N^o d'ordre : 3579

THÈSE

Présentée devant

L'UNIVERSITÉ DE RENNES I

Pour obtenir

Le Titre de : DOCTEUR DE L'UNIVERSITÉ DE RENNES I

Mention : INFORMATIQUE

PAR

Andrew I. COMPORT

Équipe d'accueil: VISTA, puis lagadic (IRISA/INRIA, Rennes)

École doctorale: Mathématiques, Télécommunications, Informatique, Signal, Systèmes, Electronique

Composante universitaire: Institut de Formation Supérieure en Informatique et Communication (IFSIC)

Titre de la thèse :

Towards a Computer Imagination: Robust Real-time 3D Tracking of Rigid and Articulated Objects for Augmented Reality and Robotics

Soutenue le 23 Septembre 2005, devant la commission d'examen composée de :

Patrick	BOUTHEMY	Président
Pascal	FUA	Rapporteurs
Marie-Odile	BERGER	
Bradley	NELSON	Examinateurs
Éric	MARCHAND	
François	CHAUMETTE	

To my parents.

Acknowledgments

The research described in this thesis was carried out from October 2001 to June 2005 at the “Institute de Recherche en Informatique et Systémé Aléatoire” IRISA/INRIA in Rennes. This thesis was initially undertaken in the ViSTA project (directed by Patrick Bouthemy) and following this in the lagadic project (directed by François Chaumette) from January 2004.

First of all I would like to thank the members of the jury for having accepted to assess my thesis:

- Patrick BOUTHEMY, Research Director, IRISA/INRIA Rennes, to have welcomed me in his group and to have honored me by presiding the jury.
- Pascal FUA, Professor of the Computer Vision Lab, Swiss Federal Institute of Technology (EPFL), Lausanne Switzerland for his encouragement and time spent in reading, analyzing, accessing and writing the report.
- Marie-Odile BERGER, Associate Researcher, INRIA Lorraine & LORIA, France for her particularly rigorous reading, analysis, constructive criticism and time spent in writing the report.
- Bradley NELSON, Professor of the Institute of Robotics and Intelligent Systems, Swiss Federal Institute of Technology ETH Zurich, Switzerland for his interest in my work and for having accepted to be a member of the jury.

I would particularly like to express my deepest gratitude to my thesis supervisors:

- Eric MARCHAND, Associate Researcher and thesis supervisor, IRISA/INRIA, Rennes.
- François CHAUMETTE, Research Director and thesis supervisor, IRISA/INRIA, Rennes.

They proposed me with a passionate research topic and have been true mentors for me. Without their guidance this thesis would not be what it is today. I would like to thank them for their confidence in me and their availability throughout my thesis. The numerous discussions and their motivating investment in the research surrounding my thesis is more than greatly appreciated.

A part of this thesis is the result of a collaboration within a national RIAM project called SORA. The aim of this project was to transfer technology to Total Immersion Inc. I would like to thank Emmanuel Marin, Valentin Lefevre and Bruno Uzzan for their interest in my work and to have validated it within real-world commercial applications. A special thanks to Cedric Riou who has worked closely with me to transfer, test and improve the software developed during my thesis from Linux to Windows.

I would like to thank David Suter for his invitation to visit his Digital Perception Laboratory and the interesting discussions that we had during my stay. I would also like to thank Danica Kragic for her interest in my work and for the fruitful discussions that we had.

Thanks to my friends and colleagues for their friendship and support including those who have shared their office with me: Nicholas Ducoin, Chafik Kermad, Nicholas Mansard, Anthony Remazeilles, and Anne-Sophie Tranchant and others including Elise Arnaud, Sid-Ahmed Berrani, Odile Bourquardez, Cybèle Ciofolo, Marc Chaumont, Nathalie Peyrard, Gwenaelle Piriou-Lenevez, Muriel Pressigout, Clementine Nebut, Omar Tahri, Thomas Veit. I would like to thank the group engineer Fabien Spindler and the secretaries Edith Blin and Stephanie Lemail for their support.

This thesis has followed a serious motorcycle accident which took place on the 13th of April 2001. I would like to take this opportunity to thank all the people who saved my life either directly or indirectly following this accident. They include the firemen, the bystanders, the air-ambulance crew, the emergency doctors, the surgeons, the nurses, the hospital staff, the reeducation staff. I would also like to thank all the people who have helped and supported me with the consequences of this accident.

I would like to make a special thanks to my mother Helen, my brothers Matthew, Timothy and Michael, my grandfather Ray. Last but not least I would like to send my love to my baby girl Fleur and my wife Delphine who knows how much I owe her.

Contents

SCALAR QUANTITIES	
Index for any vector	i
Index for any matrix	(i, j, k)
Number of parameters to estimate:	m
Number of features	n
Number of features for component i	n_i
Number of joints	l
Number of intrinsic parameter	p
Number of components in an multi-body object	κ
Number of cameras in an multi-camera system	s
Number of features in image i	s_i
Dimension	d
Dimension for feature type i	d_i
Number of feature types	t
Number of linearly independent components	ι
Focal distance of the camera	f
Ratio between object and camera motion	ϱ
Feedback control gain	λ
MATHEMATICS	
Scalars	lowercase
Vectors	**bold lowercase**
Matrices	**BOLD CAPITAL**
Geometric primitive	Calligraphy
Vertical concatenation of matrices, vectors and scalars	$\mathbf{M} = (\mathbf{M}_1,\ \mathbf{M}_2,\ ...,\ \mathbf{M}_n)$
Horizontal concatenation of matrices, vectors and scalars	$\mathbf{M} = \left[\ \mathbf{M}_1\quad \mathbf{M}_2\quad ...\quad \mathbf{M}_n\ \right]$
n dimensional Real Space	$\mathbb{R}^n$
n dimensional Euclidean Space	$\mathbb{E}^n$
n dimensional Projective Space	$\mathbb{P}^n$
Special Euclidean Group or Lie Group	$SE(3)$
Special Orthogonal Group	$SO(3)$
Lie algebra	$se(3)$
Transpose of $\mathbf{S}$	$\mathbf{S}^\top$
Inverse of $\mathbf{S}$	$\mathbf{S}^{-1}$
Pseudo-inverse of $\mathbf{S}$	$\mathbf{S}^+$
Skew symmetric matrix of the vector $\mathbf{u} \in \mathbb{R}^3$	$[\mathbf{u}]_\times$
Homogeneous twist matrix of the vector $\mathbf{v} \in \mathbb{R}^6$	$[\mathbf{v}]_\wedge$
The only exception being that 3D primitives(i.e. points, lines, ellipses, ...) are noted in capital even though they are not matrices.	

Table 1.1: The notation used in this thesis.

GEOMETRY	
Angle of a line in 2D	θ
Tangent angle	α
The distance from the origin of a line in 2D	ρ
3D coordinates	X, Y, Z
2D coordinates	x, y, z
3D parametric equation parameters	$A, B, C, D, E, F, ...$
2D parametric equation parameters	$a, b, c, d, e, f, ..., k_1, k_2, ...$
Plane parameters	α, β, γ
Reference frame i	$\mathcal{F}_i$
The camera reference frame	$^c.$
2D distance	d
3D point	$\mathbf{P}$
2D point	$\mathbf{p}$
Homogeneous 3D point	$\bar{\mathbf{P}}$
Homogeneous 2D point	$\bar{\mathbf{p}}$
3D line	$\mathcal{L}$
2D line	$\mathbf{l}$
2D moment	m_{ij}
2D centered moment	u_{ij}
2D normalized centered moment	n_{ij}
The center of gravity	(x_g, y_g)
Contour	$\mathcal{C}$
Projected region of a contour in the image	$\mathcal{R}$
Rigid body mapping	$\mathbf{m}(\,)$
Homogeneous matrix for rigid body mapping	$\mathbf{M}$
Adjoint map for transforming twists	$\mathbf{V}$
Rotation matrix	$\mathbf{R}$
Identity matrix	$\mathbb{I}$
Camera Intrinsic parameter matrix	$\mathbf{K}$
Interaction matrix/vector for a primitive $\mathbf{s}$	$\mathbf{L_s}$
Combination matrix for the case when $dim(\mathbf{L}) > m$	$\mathbf{C}$
Simplifying parameters for line interaction matrix	$\alpha, \lambda_1, \lambda_2, \lambda_\rho, K_1, K_2$

Table 1.2: The notation used in this thesis.

GEOMETRY Continued...	
3D primitive	S
2D primitive	s
Vector of 3D primitives	$\mathbf{S}$
Vector of 2D primitives	$\mathbf{s}$
Error vector	Δ
Task function error vector	$\mathbf{e}$
The pose vector	$\mathbf{r}$
Translation vector	$\mathbf{t}$
Rotation vector	$\boldsymbol{\Omega}$
Twist vector	$\mathbf{v}$
Camera calibration parameters	ξ
Camera calibration parameter velocities	$\dot{\xi}$
Combined twist vector and camera calibration velocities	$\mathbf{w}$
Generalized pose vector for an articulated object	$\mathbf{q}$
Generalized twist vector for an articulated object	$\dot{\mathbf{q}}$
Linear velocity vector	$\boldsymbol{\upsilon}$
Angular velocity vector	$\boldsymbol{\omega}$
General coordinate vector for interaction matrix derivation	$\mathbf{X}$
General 2D primitive parameterizations function	$h()$
General 2D primitive parameterizations function	$g()$
General projection and visibility function	$\mu()$
General parameter vector for 3D primitive	$\boldsymbol{\Phi}$
General parameter vector for 2D primitive	$\boldsymbol{\phi}$

Table 1.3: The notation used in this thesis.

PROBABILITY and STATISTICS	
Likelihood	L
Statistical weight	w
Weighting matrix for iteratively re-weighted least squares	$\mathbf{D}$
Outlier threshold	τ
Number of samples	N
Number of data in a sample	s
The number of consensus data (within a distance threshold	eta
Standard deviation	σ
Hypothesis	H
Log ratio of the likelihood function	ϵ
Modified likelihood criterion	ζ
Probability density	P
General parameters	Θ
Cumulative normal distribution	Φ
Negative logarithm of the probability density	$\rho()$
Influence function	ϕ
Expectation of i	E_i
Influence function	ψ
Rate of change of the scale	χ

Table 1.4: The notation used in this thesis.

CHAPTER 1

Introduction

1.1 Introduction

Much research has been focused on the problem of how best to describe the relationship between objects in the physical world. A series of 6 pose parameters can be considered to fully define the location and orientation between two rigid-bodies in 3D space and time. In this thesis, interest will be focused on using vision sensors to observe this basic geometric relationship between different objects. As will be shown, this can be achieved by building a virtual representation of the external world and relating it to a virtual image.

1.1.1 Computer Imagination

Computer imagination refers to an artificial visual system inspired by different aspects of human imagination. The main concept involves a vision system which perceives and mirrors reality using a virtual 2D image, a-priori knowledge about the environment and imaginary 3D actions which affect this image. More specifically, imagination is defined as: "The act or power of forming a mental image of something not present to the senses or never before wholly perceived in reality" [Merriam-Webster04]. As will be shown, a virtual or imaginary image provides the central unifying point between the different concepts that will be presented in this thesis. This virtual image is not only considered as static but also varying temporally and independently of the real world. Furthermore, imagination also " ... signifies a lively conception of objects of sight" [Merriam-Webster04]. In this thesis the lively conception of visual objects is defined as imagining in real-time with respect to the measurement rate of a visual sensor. In the Diderot-Alembert Encyclopedia [Diderot51] a further distinction is made between two types of imagination:

- passive imagination - concerns the representation of an image that is involuntarily present to the senses,
- active imagination - consists in bringing together different visual memories to form a new and dynamic representation. This refers to a more creative state of the mind where the composition of completely virtual images takes place.

Although, passive imagination will be the main topic of interest in this thesis, active imagination will also be considered within an Augmented Reality (AR) context. In this case, completely virtual images are superimposed onto real images so that they are aligned correctly with the real world. Creation of completely virtual images in an automatic way, however, is related to a more abstract fusion between ideas, memory, learning, virtual imagery, etc, and is thus related to many domains that are not within the scope of this study.

The core problem studied in this thesis is multi-disciplinary and can essentially be found at the intersection between the visual servoing and computer vision domains, however, other scientific domains are also important. Automatic control and feedback control loops play a major role in the minimization of error. Statistical methods provide a solid foundation for making decisions within an uncertain and varying world. Differentiable geometry provides the basis for describing the three-dimensional space in which we live.

With regard to visual servoing, a computer imagination is shown to replicate the perception-action cycle in traditional image-based visual servoing (IBVS) within a virtual model. Visual servoing is an approach to the control of robots based on visual perception. It involves the use of visual information to control the position of a robot relative to its environment. The control of this robot is usually specified by a particular task. In a similar manner in this study, the notion of a virtual image is considered to vary in time with respect to a virtual 3D model of the environment. This relation is therefore centered on the spatial-temporal relationship between a virtual 3D environment and a virtual image. In this way, the classical formulation of visual servoing is considered in a Virtual Visual Servoing context. As will be defined in detail, the different general concepts involved in this thesis can be summarized as:

- the sensing of visual information from the environment with a monocular camera,
- a 3D model representing the *a priori* knowledge of the environment,
- a virtual 2D image which evolves in real-time both spatially and temporally,
- the perception of visual information at both low and high levels,
- virtual actions defined by an image-based closed loop control scheme,
- the creation of augmented images by overlaying virtual objects into the real scene.

1.1.2 Human Visual Perception

Much research in computer vision is strongly motivated by the human visual system [Marr81]. In particular, the continuous closed loop model is strongly supported by psychophysical evidence which shows the dense flow of information from many different sources. It is easy to see that information input from the senses is kept as knowledge within the brain, however, it is more difficult to show the impact of this knowledge on the senses. It can be shown, however, that changes in the appearance of a familiar object can be attributed to external causes [Meer04, Goldstein87]. A classic example of the backward flow of information is the Ames room experiment [Ames52]. This room is built such that objects that are closer, are made smaller than objects that are further, so that the perspective foreshortening compensation is violated. In other words, the human visual system attempts to compensate by using the knowledge that the image of objects which are closer are usually larger.

A similar experiment can be found walking through the Washington DC, Smithsonian sculpture garden where a model house violates these same rules (see Figure 1.1), however, in this case the perceptive effect works without a monocular peep-hole. While walking around the house, the perception of the object is disturbing due to the contradictory effect between stereopsis and a-priori information.

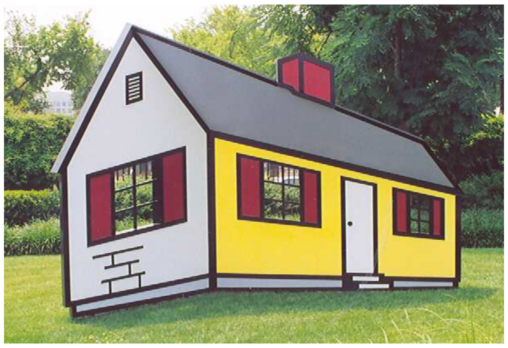


Figure 1.1: A model house which visually violates the perspective interpretation of the physical object. The human visual system perceives the 3D structure of the house using a-priori knowledge about the proportion of distances and angles. If you look closely, there is a disconcerting effect on your perception as there are visual cues emanating from the illumination of the house which contradict the a-priori perception of the 3D structure (In fact, this house is actually a concave structure).

1.2 Objective

The main aim of this thesis is to study the problem of maintaining an estimate of the relative position between a monocular camera sensor and the position of 3D objects via the use of visual information. A major assumption is that 3D knowledge about the environment is known *a priori*. One of the objectives is to center the solution to this problem on techniques employed within both the visual servoing and real-time 3D tracking domains. In this way, important parallels will be made between visual servoing techniques and estimation techniques. Furthermore, this allows a visual servoing platform to be used as an experimental test-bed for validating and visualizing the behavior of the estimation procedures employed.

Traditionally research in the visual servoing community has been targeted at the movement of robotic manipulators in real-time by exploiting image sensor information. This is achieved via a feedback control law or task function which is closed via the image. General tasks involve moving an end-effector to a certain pose with respect to certain objects and features with the image. These methods have proven to be very reactive to visual stimuli due to the tight coupling between image features and camera movements.

On the other hand, the visual tracking community has focused on a similar problem which is related to visual servoing via non-linear iterative parameter estimation. In a 3D tracking scenario, depth information is often obtained from a-priori knowledge of the environment. The main aim of visual 3D tracking is to maintain an estimate of the location of the viewpoint of a camera, from an image acquired by this camera, and a 3D model of the environment. This estimation then provides unknown parameters to a more global model that also contains the configuration of different elements that compose a scene. In a first part, this configuration is defined by the 3D pose of a rigid-object with respect to the camera. The pose is composed of six parameters representing the three degrees of translation and the three degrees of rotation that completely define the rigid transformation between the camera and the object frames of reference. Later this configuration will also be extended to include the relationship between different rigid elements within the environment. This extended configuration space will also include the relative translation and rotation between rigid components.

In order to estimate the pose it is necessary to extract enough visual information from the image to determine the geometric relationship between different elements of the scene. Thus, in order to well constrain all parameters, it is necessary to have well conditioned visual information. Therefore, it is interesting to consider various different types of geometric visual information. Furthermore, in order to estimate using real data it is also necessary to take into consideration noisy measurements or outliers. Therefore an important objective is to consider redundant visual information. In order to remove noisy measurements and outliers, another objective is to develop robust algorithms within a visual feedback control loop.

Based on a solid foundation for tracking of rigid objects, the aim is then to extend the method to non-rigid objects so that a more general class of objects may be considered. The objective is to relax the rigidity constraints so that an articulated class of objects may be considered. An articulated object is therefore considered as rigid components that are linked together by joints that have varying degrees of freedom.

One of the innovative aspects of the is work is the aim to focus on real-time computation. Not only is the computational efficiency of utmost importance for many applications but it also reduces supporting theories to their simplest form. In this way, the search for efficient methods leads to focusing on the essential parts of the problem which are often general and applicable to other domains.

Another aim is to use a monocular camera system. Although multi-camera systems [Martin02] clearly provide extra information about the scene along with a wider field of view, much research remains to be carried out on the understanding of perception with single camera without needing to combine two or more of them. Indeed, the aim is to define a general method which can be extended to further cameras without much effort. Furthermore, there is now wide spread use web-cameras which open the door to many immediate applications.

1.3 Overview

As mentioned, the aim of this thesis is to use visual servoing techniques for real-time 3D tracking by performing visual servoing virtually [Kinoshita94, Sundareswaran98, Martin02, Marchand02a], known as Virtual Visual Servoing (VVS) [Marchand02a]. In this case, the pose estimation problem is considered as synonymous to moving a virtual camera so as to minimize an error criterion in the image. More precisely this involves using a CAD model of an object and projecting this onto the image using an initial 'guess' pose of the camera or the previously known position of the camera. This projection is then compared with the corresponding measures extracted from the image representing the object's actual position in the image. This then forms a minimization criterion and the actual pose is then converged upon using a non-linear iterative minimization technique. Indeed, several techniques have shown the advantages of model based techniques to calculate the pose of the camera [Gennery82, Lowe87, Lowe92].

In Figure 1.2, an overview is given of the tracking feedback control loop. The Virtual Visual Servoing control law is equivalent to an iterative minimization procedure. It represents the major components of the system including, on the left of the diagram, the perception of visual features from an input image. These features are compared with the virtual position of the features according to the previously estimated pose. This error then interacts with the pose to produce a virtual action upon a virtual camera. The result of this is the projection of a known CAD model onto a virtual image which is then compared with the real visual features.

Low level visual information is extracted from the image via efficient one-dimensional edge searching. A one-dimensional method is possible because the search is made to the normal of projected object contours in the image. Furthermore, edges contain a high level of geometric information while remaining invariant to changes in illumination. The oriented Moving Edges algorithm [Bouthemy89] provides a solid basis for efficient low level tracking with convolution efficiency.

The errors corresponding to the projected contours form the basis for distance type visual features. In this thesis, distances to the perspective projection of 3D points, lines, planar contours (ie. circles) and the

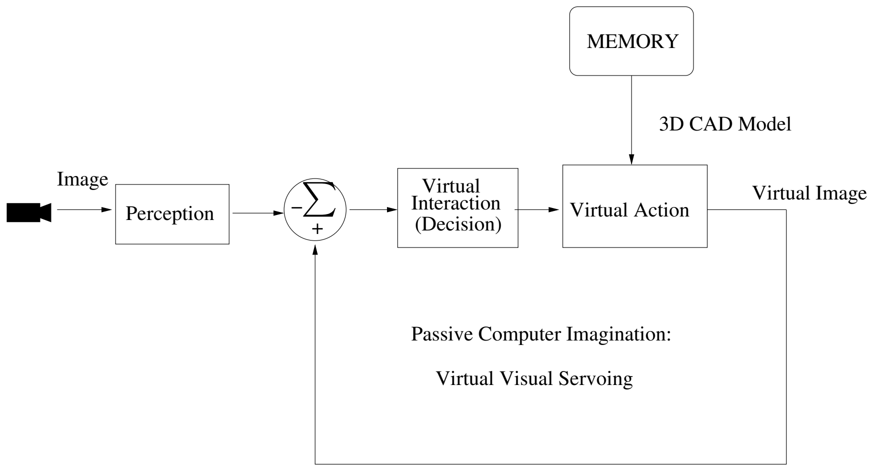


Figure 1.2: A control diagram showing how the pose is estimated using Virtual Visual Servoing. The control loop maintains an estimate of the pose and a virtual image that contains a projection of a known CAD model. This diagram is related to passive computer imagination since the virtual image is closely related to the perception of the real image. At the input is the image from which visual features are perceived. The visual features are compared with virtual features and this is used to produce a virtual action upon a virtual model.

volumetric surfaces (ie. cylinders) are considered. This involves deriving an interaction matrix for each type of primitive using an adequate modeling approach [Chaumette90, Espiau92].

One of the major problems in many estimation techniques is to address the important problem of robustness. Robust statistics have emerged from the literature as an efficient way to reject a general class of outlier during the minimization process. In this thesis, robust M-estimators [Huber81] are integrated into the Virtual Visual Servoing control law via an iteratively re-weighted least squares methodology.

Another limiting factor of tracking 3D objects is the rigidity assumption. Rigid-body tracking is extended to a more general class of non-rigid articulated objects by constraining rigid-body motion using velocity constraints. The velocity constraints are defined according to different mechanical joints. This allows to define the different orthogonal kinematic subsets of motion belonging to a multi-body object and to derive a closed-loop control law for maintaining an estimate of the parameters describing the configuration of the object in 3D space.

1.4 Historic

In what follows is an abbreviated, and by no way comprehensive, overview of the roots of the visual pose estimation problem derived from accounts by [Murray94], [Eklundh01] and [Haralick87]. A good introductory text for computer vision is [Trucco98].

Of course the history of mathematics and geometry can be traced far into history. For the purpose of this thesis, it can be noted that expressions for rigid motion came to light in the early 1800s when Chasles proved that a rigid body can be moved from any one position to any other by a movement consisting of rotation about a straight line followed by translation relative to that line. Poinsot showed that any system of forces acting on a rigid body can be replaced by a single force applied along a line combined with a torque about that same line. Using this work, Robert S. Ball published a complete theory of screws [Ball00]: A treatise on the Theory of Screws. Nearly one hundred years later, pose estimation techniques, known then as absolute orientation estimation, had matured significantly. The dissertation by [Szczepanski58] is reported to survey nearly 80 different solutions beginning with one given by Schrieber of Karlsruhe in the year 1879.

Computer vision started close to the beginning of computing. Indeed in 1951 John von Neumann proposed to analyze micro-graphs using computers by comparing intensities at adjacent locations in the images. An early work which considered three-dimensional models of scenes consisting of polyhedral objects was published by [Roberts65]. He considered perspective transformations and used differentiation to detect lines. Computational vision models suggested that the paramount importance of a vision system lies in its 3D perception of the world. Early vision methods focused on low level image processing [Sobel70, Marr80]. Many initial efforts required specialized hardware to overcome the major bottle neck in real time image processing which was, and still is, the computational complexity of treating a large quantity of data in series. Clearly, the exponential growth in computing power highly benefits study in this area. Of course, many initial efforts in real-time target tracking (see [Bar-Shalom93]) were oriented towards military applications. The paradigm of Marr in [Marr82] proposes a model for 3D visual perception consisting of three principle steps: segmentation, reconstruction and recognition. Stereo [Longuet-Higgins81] or active vision [Aloimonos87] are both duly considered important steps in 3D reconstruction and perception of the world. Of course the distinction between these different principles is not so clear cut, especially given the dense flow of information in all directions within the human model. Indeed evidence suggests that higher order processes rely substantially on the backward flow of information from a-priori knowledge of the world towards the senses. One fundamental a-priori model can be found in the relation between two views of the world [Lucas81, Longuet-Higgins81] such that the epipolar constraint exists. In the early eighties, published work emerged focusing on 2D scene registration with 3D models [Gennery82]. Pose estimation was mostly performed from a single view [Kanade81] using image descriptors [Lowe80], cited in [Lowe91] including line drawings [Barrow81].

A more recent area of study is the domain of robustness which is based on estimating unknown parameters correctly in the presence of external sources of disturbance. This notion is very important when theoretical models are applied to real world data. In pure mathematics, by the end of the 1900s estimation techniques had been studied significantly and several landmark textbooks had been written on the notion of robust statistical estimation [Huber81, Rousseeuw87]. The first robust pose estimation solution in computer vision was given by [Fischler81].

Three-dimensional tracking has proven to be essential to many domains and recent applied fields include visual servoing and augmented reality. The field of visual servoing has developed somewhat in parallel with computer vision, however, with a fixed focus on interaction with the 3D world [Weiss87, Chaumette90, Espiau92]. More recently, one of the first useful examples of Augmented Reality was published in [Drascic91] based on a virtual pointer superimposed on a newly designed stereoscopic video system.

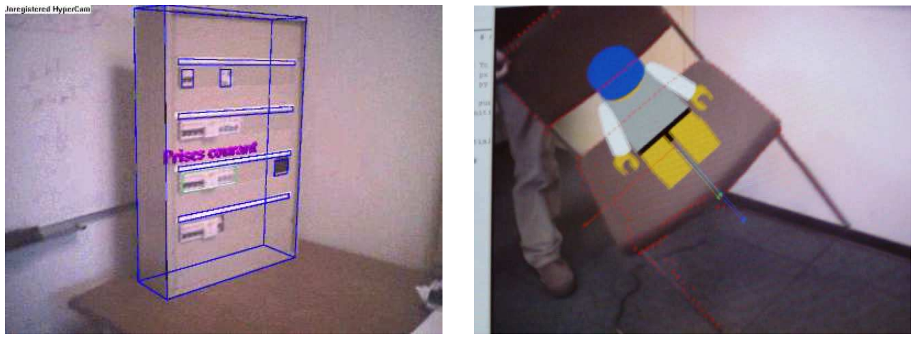
1.5 Applications

In this thesis, one of the principal objectives is to apply visual servoing and 3D tracking research to the young and developing field of Augmented Reality (AR). A concerted effort has also been made to use the example of AR as a platform for developing and testing more general and fundamental techniques such as robust statistical estimation and non-rigid mechanics. Furthermore, visual servoing applications have also benefited from this work.

1.5.1 Augmented Reality

Augmented Reality, by definition [Azuma97], is an environment in which computer-generated virtual objects are seamlessly integrated into the real world (see Figure 1.3). AR differs from virtual reality where the user is completely immersed in a virtual environment. Alternatively, in augmented reality, virtual objects are superimposed onto the real world. In [Milgram94b, Milgram94a], the distinction between real and virtual worlds has been studied and the relationship between them was shown to be continuous as shown by Milgram's reality-virtuality continuum (see Figure 1.4).

Real-time computational efficiency and interaction with a 3D world are core concepts in closed-loop Virtual Visual Servoing techniques. Likewise in AR, an important defining notion is the requirement for AR to be *interactive*, as given in [Azuma01] . This means that it is necessary to have both a 3D relationship of virtual objects, as seen by the user, and the incrustation of these objects in real-time into the real-world. This definition also means that post-production techniques are excluded from this definition. Indeed, techniques already exist to mix virtually animated 3D models and real-images, such as in the film "Jurassic Park", however, these methods are full post-production techniques that are highly time consuming and have little overlap with AR techniques. Even with this classification of techniques, there exists a continuity of methods corresponding to different levels of automation and computational requirements. Furthermore, the improvement in processing power continues to improve meaning that more and more complex techniques can be considered. In this way, some near real-time post-production techniques are defined as AR (eg. [Berger99]). Thus, when researching new techniques, it is important to compare the performance of algorithms and to balance different performance factors against computational efficiency.



(a) (b)

Figure 1.3: Augmented reality examples resulting from the work presented in this thesis: (a) A maintenance scenario on a real electrical panel with virtual instructions in 3D, (b) A game with a virtual Lego-man on a real chair.

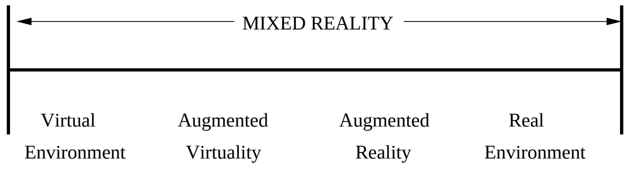


Figure 1.4: Milgram's reality-virtuality continuum adapted from [Milgram94b]

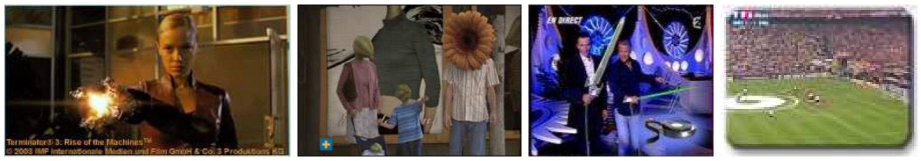
The principal research problems in vision-based augmented reality are summarized as:

1. the display technology,
2. the real-time estimation of the viewpoint of the imaging device,
3. the human computer interaction and applications.

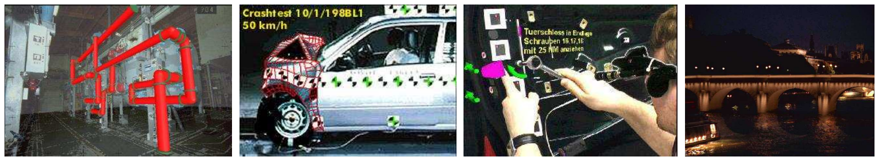
Of course, the hardware support for combining virtual and real images poses a major obstacle to any AR application. In the past, head-mounted displays, such as those found in military fighter jets [Caudell92, Caudell94, Sims94], have been considered an interesting way of achieving this. Different head-mounted display (HMD) technologies include those which perform video-see-through [Ellis94, Rolland95] whereby the user looks at an image of the real environment superimposed with virtual objects. The advantage of this method is that there is no delay between the display of virtual and real images. The disadvantages, however, include a completely digitized image and a small field of view. Another technique is called optical-see-through, where either a semi-transparent display is used and virtual objects rendered onto this display medium (eg. [Holmgren92, Cakmakci04]) or the image is projected directly onto the retina (e.g. [Viirre98, Chinthammit03]). The advantage here is that the user still sees a full resolution view of the environment and their field of view is not reduced. The disadvantage of optical-see-through is that the calibration procedure is further complicated and could require the estimation of calibration parameters online. In general, however, HMD techniques are quite intrusive since they require wearing equipment on the head. More recently, the trend has been towards less intrusive technology such as mobile phones [Mohring04], PDAs [Beier03, Pasman03], tablet PCs [Klein04] or even projectors [Bimber02, Ehnes04]. Indeed, the easiest way to avoid using specialized hardware is to perform video-see-through on a standard monitor. So as to avoid complication, a monitor will be used as the display device in this thesis. Of course other standard hardware is required for a complete AR system including a camera device and a computer. Without going into detail, hardware factors include the frame rate of the camera, the resolution of the array, the distortion introduced by the lens and the efficiency of the computer. Specialized hardware also exists for graphical display and image processing, however, the aim in this thesis is to use standard hardware so as to focus on the efficiency of the algorithm.

Another major problem of AR is real-time viewpoint estimation which is the main focus of this thesis. Viewpoint estimation is related to the alignment of virtual objects in the scene with real objects present in such a way that they are visually acceptable. This requires recovery of the pose between the environment and the camera. Hardware solutions to this problem exist such as magnetic sensors [Roetenberg03], accelerometers, gyroscopes [Satoh04], etc, which perform AR within an open loop system. However, this usually results in visual miss-alignment which is not acceptable to the observer. To date, hardware solutions (alone) have not offered the flexibility and robustness that camera sensors do. Apart from being highly flexible and general, the main advantage of a camera sensor is that the control loop can be closed via the image. This means that the alignment between real and virtual objects is verified directly within the image so that the resulting estimated pose is precise and visually acceptable. As will be shown these techniques are easily adapted so as to accurately register a virtual object in a dynamic scene environment. Of course, other adjacent problems exist, such as the need to manage occlusions between real and virtual objects [Lepetit00], which are not considered here.

The third problem of human computer interaction (HCI) depends heavily on the underlying application. Various applications are shown in Figure 1.5. Augmented Reality touches upon many domains including computer graphics, virtual reality, artificial intelligence, psychology, etc. Clearly, the core AR technology is based on the hardware and the registration techniques. Consequently, solutions to these two fundamental problems provide the basis for a general computing tool which can be applied in many areas. Inversely, the concept of AR is motivated by many applications which each have their



Audiovisuel : cinema (Terminator 3), advertisements (Renault/Realviz),
television (France 2, Total-immersion), sport (TF1, Symah-vision)



Industry : study of conformity (Siemens)
automobile industry (Arvika), manufacturing (Arvika), impact of new lighting (Loria)

Various : video games (Aqua-gauntlet, MR-lab), military (US Marines Corps),
medical (UNC Chapel Hill)

Figure 1.5: Examples of applications that are based on augmented reality.

specific requirements. In the medical domain applications include visualization of 3D ultrasound images [Bajura92, State94], visual guidance of needles [State96, Nicolau04], MRI and CT scan visualization [Grimson95, Mellor95, Lorensen93]. Maintenance and repair scenarios for a laser printer have been presented in [Feiner93] and for an engine in [Rose95, Tuceryan95]. Other applications include industrial manufacturing [Caudell92, Janin93, Sims94, Navab99], driver assistance [Kealy04], 3D computer interfaces [Feiner93] where traditional 2D windows are placed within a 3D environment or books with 3D markers that tell a 3D story [Billinghurst01], entertainment such as ARQuake [Piekarski02] or the television media [Woolard03], military purposes such as battlefield augmented reality in urban environments [Julier99], robot guidance [Drascic93, Milgram93] and the list goes on... A detailed survey of AR applications is given in [Azuma97, Azuma01].

Until recently, most real-time pose estimation methods applied to AR have been based on the manual placement of visual markers in the scene [Kato99]. Another marker based technique was presented in [Sundareswaran98], where the analogy between pose estimation and image-based visual servoing was first applied to an AR applications. More recently, Virtual Visual Servoing methods have been combined with marker-less image processing and robust statistics to perform AR in real-time, such as that presented here [Comport03a]. In this thesis a marker-less model-based algorithm is used for the tracking of 3D objects in monocular image sequences. In Figure 1.6, an overview is given where augmented reality is combined with the 3D pose estimation control loop.

1.5.2 Visual Servoing

Another application of 3D tracking techniques is visual servoing. Visual servoing is an approach to the control of robots based on the visual perception of the environment. This perception of the environment

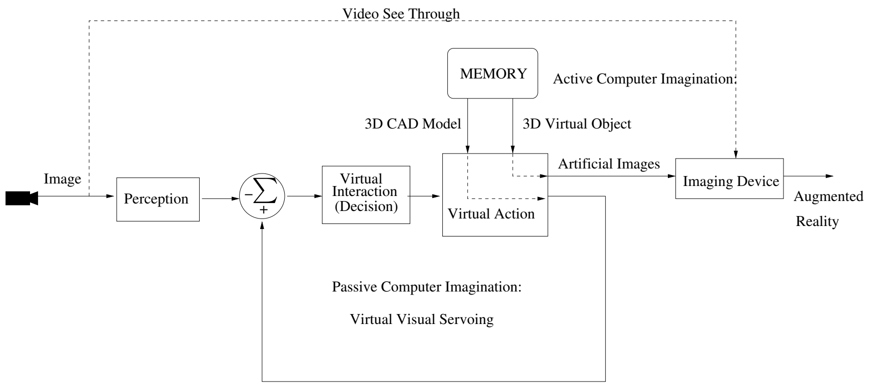


Figure 1.6: A control diagram showing how the estimated pose is used to produce an augmented image. On the left is the input image. In the middle is the input of both the CAD model of the target as well as the model of a virtual object which is to be augmented into the scene. The control law maintains an estimate of the pose between the camera and a target object. This diagram is related to both passive and active computer imagination. The passive part is related to the tracking of the pose between the camera and the object. The active part is related to the incrustation of virtual objects into the image such that they are correctly aligned with objects in the real image. In this way the real image is augmented with extra visual information.

is then used to control the robot within this environment. Thus, the tracking of known 3D objects within the environment is directly related to an improved sense of visual perception. A 3D tracking step provides the complete 3D configuration between different elements within the environment to a visual servoing task. More precisely, it provides depth information along with information about hidden parts of the scene.

In the past, various techniques have been proposed for visual servoing including position based (PBVS) and image based visual servoing (IBVS) [Hutchinson96]. A classical scenario being to control an eye-in-hand robot w.r.t. to its environment. They differ primarily in the input of the closed loop control scheme, the first being based on a pose while the second being to based on images. The advantage of PBVS is that it allows for direct control of a robot in 3D space, however, it requires the pose to be known at the input stage. The pose is usually estimated by a pose estimation step. On the other hand, the advantage of image based control is that it is based in the sensor frame of reference making measurement errors easy to handle. Furthermore, it is easiest to ensure that visual features remain in the field of view. Generally, the final image is given to the robot via either a teaching by showing approach or the projection of a CAD model using a desired pose. Image based techniques have been shown to be accurate due to the closed loop control law (see Figure 1.7), to be flexible by permitting tasks to be defined easily [Chaumette90, Espiau92] and to be robust to camera calibration errors [Malis98]. A difficulty of this approach is proving convergence due to local minima or singularities in the interaction matrix. It has been shown, however, that it is possible to avoid these singularities by using more complex features in the interaction matrix such as lines instead of points [Chaumette98]. In a classical context this method essentially involves calculating the interaction matrix from images captured in real time, to provide nonlinear velocity control to a robotic system. The resulting pose of the arm after performing IBVS is essentially the solution to the final pose. In this thesis, several advantages of IBVS will be brought to PBVS by using Virtual Visual Servoing.

A fundamental step towards broadening the use of real world image-based visual servoing is to deal with the important issues of reliability and robustness to failure. Few methods on visual servoing propose a solution to this problem in the literature. Recently, a robust cue integration approach was proposed for a stereo eye-to-hand visual servoing task [Kragic01]. This method involved combining visual cues and

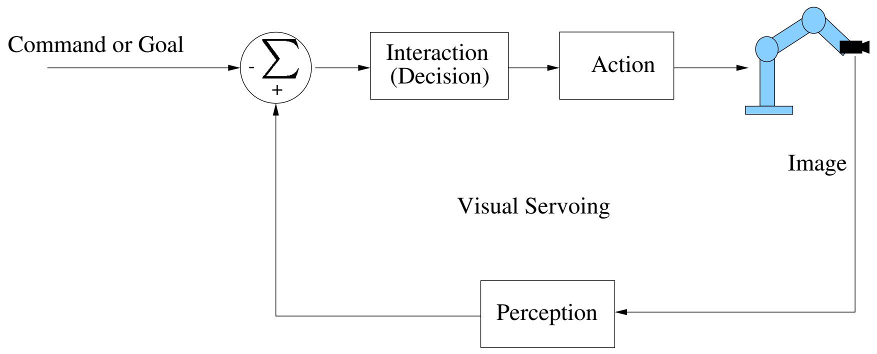


Figure 1.7: A classic image-based control diagram showing an eye-in-hand visual servoing system. The input command is usually a set of desired image features, however, it is also possible to use a known CAD model and project this onto the image using a desired pose. The execution of the task is complete when the error between features in the image fall below a certain threshold.

rejecting outliers in the feature extraction step. This approach, however, requires the definition of visual cues and relies on many manually chosen thresholds. Furthermore, in a real-time scenario it is not very efficient because a voting approach is used which performs a binary mapping of a feature's probability to either 1 or 0. In this thesis, a closed loop control law is presented that simultaneously accomplishes a visual servoing task and is robust to a general class of external errors. This generality allows concurrent consideration of a wide range of errors including: noise from image feature extraction, small scale errors in the tracking and even large scale errors in the matching between current and desired features.

1.6 Contributions

Vision sensors are devices which provide an enormous amount of information, therefore making them general and widely applicable, often adding to the difficulty and progress of research in this area. In this section, an outline of several open problems will be given in the areas of 3D tracking, augmented reality and visual servoing. Following this, contributions to these problems will be described.

A problem of primary importance is the design of vision based tracking methods which perform in real time, thus requiring efficient algorithms. Indeed, it is possible to obtain good tracking results with large amounts of off-line computation, however, off-line methods are not sufficient for many applications such as augmented reality. Real-time performance is not only important for AR, but it opens the door to a new realm of potential applications. Thus it is important to recognize major bottle necks in current techniques and to propose efficient solutions.

Another area where much work remains to be carried out is in the extraction of meaningful information from the image. In most work on pose estimation only simple point features are used and more complex features are often neglected. Indeed, real-world objects are composed of all sorts of shapes and forms which can not be perceived solely using simple point features. Some works have extended tracking to line features or planar texture patches, however, the entire set of visual information is far from being fully exploited. Therefore, it is important to design methods which can be generalized to deal with more complex geometric visual information.

Another major obstacle to the usability of tracking techniques in the real-world is their robustness to failure. In the literature, many limiting assumptions are made including: the assumption of specific colors such as the use of skin color for body tracking which results in a loss of generality; the use of correlation techniques that are dependent on illumination conditions; the use of texture which is not always present; or the avoidance of occlusion with other parts of the environment. Experiments are often designed to show the benefits of a particular method rather than its weaknesses. For example, background textures are carefully chosen so as not to disturb tracking methods; colors or textures which are similar to those being tracked are avoided; lighting conditions are carefully setup to enhance tracking; movements are often performed slowly and smoothly; etc. Indeed, a simple live demo of many real-time tracking techniques suffices to see that many problems remain to be solved. In summary, it is clear that more general methods are required which work in a larger range of situations and can deal with unexpected events. Indeed, it is possible to account for each type of error by further modeling, however, it is desirable to have techniques which are general and do not require modeling a potentially infinite source of error. In other words it would be interesting to reject any general external source of disturbance.

A fundamental assumption made in the definition of many three-dimensional models is the rigidity assumption relating to the geometric structure of the world. However, in a real world setting, objects may take on many different non-rigid forms. Non-rigid forms range from less constrained such as gases and fluids to more constrained forms such as hinges or screw links. Therefore, it would be interesting to relax the rigidity constraints so that any type of non-rigid physical configuration may be considered.

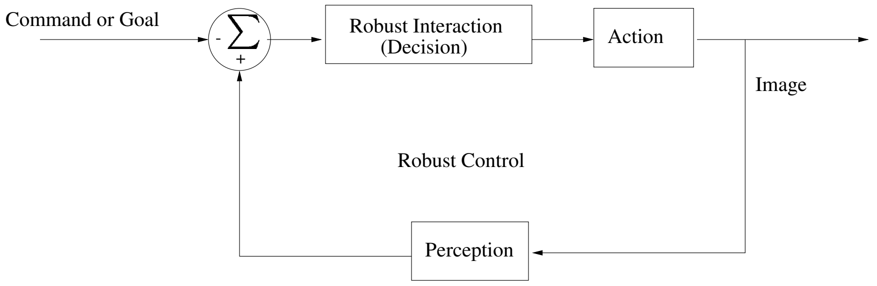


Figure 1.8: An overview of the robust visual control law proposed in this thesis for 3D tracking and visual servoing. It is based on an Iteratively Re-weighted Least Squares implementation of robust maximum likelihood estimation. The difference with respect to previous cases is the robust rejection of outliers at the interaction step.

In the following, the major contributions of this thesis will be highlighted. It should be noted that in the remainder of this thesis, contributions in the domain of visual 3D tracking will be presented in detail and contributions in the domain of visual servoing will only be mentioned briefly. Contributions can be exhibited at different levels:

- **Virtual Visual Servoing Based 3D Tracking** - In this thesis, real-time pose computation is formulated in terms of a full scale non-linear optimization: Virtual Visual Servoing (VVS). A contribution proposed in this thesis involves closing the control loop through a monocular image stream for full spatio-temporal tracking of 3D objects. This provides real-time and accurate 3D tracking of complex objects. Furthermore, different scene models are investigated including camera-based and object-based movement models. Hidden surface removal is integrated into the 3D object model so that complete 3D objects may be used instead of planar contours.

- **Distance-to-Contour Interaction Matrices** - A central problem in the design of a 3D tracking control law is the derivation of the interaction matrix which relates the movement of different types of visual features to the 3D movement within the scene. A general method is recalled to derive interaction matrices for any type or distance-to-contour based image feature. Distances are used for their real-time efficiency and because they provide a common basis for many different types of contours. Different types of interaction matrices are derived and analyzed including distances to the perspective projection of 3D points, lines, planar contours (circles) and volumetric surfaces (cylinders). Furthermore, a continuous, one-step simultaneous estimation procedure is obtained by 'stacking' feature Jacobian to combine objects of different types. Furthermore, this type of feature provides high computational efficiency for real-time operation. Further efficiency can be obtained by using a constant interaction matrix.

- **Scale Invariant Moving Edges** - An oriented edge detector is used in order to provide real-time tracking of points normal to the object contours. This type of local approach is ideally suited to real-time tracking due to an efficient 1D search. Edge detection involves making local spatio-temporal edge hypotheses along the normal. Normally, these hypotheses are then tested using a detection threshold, which holds the edge scale information, to determine if an edge exists at a given location. A method is proposed in this thesis for propagating the uncertainty and scale of these locations to the global pose uncertainty and scale in an efficient and accurate way. In this way, no intermediary decision is made as to whether or not an edge exists at a given location and no predetermined threshold (or scale) is needed. The decision as to whether or not an edge exists is then made simultaneously with the global estimation of the pose by using robust estimation techniques which either reject or accept the point.

- **Statistically Robust Control** - As mentioned previously, in a 'real world' scenario some features may be incorrectly tracked. This may be due to many reasons including occlusion, changes in illumination and miss-tracking. This problem is overcome by designing a robust control law based on robust statistics (see Figure(1.8). This involves estimating the pose from a highly redundant number of distance-to-contour correspondences so that in-liers may be determined and outliers may be rejected. In particular, the statistical technique of robust maximum likelihood (M-estimation [Huber81]) is employed. This is introduced directly into the visual control law by an iteratively re-weighted least squares method. The Median Absolute Deviation (MAD) is used as an estimate of the standard deviation of the inlier data and Tukey's influence function was used to completely reject the effect of outliers. This combination is advantageous because of its high efficiency, high breakdown point and desirable influence functions. The robustness and stability of the control law is shown to be dependent on a subsequent measure of position uncertainty. Furthermore the convergence criteria of the control law are investigated.

- **Kinematic Sets for Articulated Objects Tracking** - Until now, models of non-rigid motion or more particularly articulated motion have been defined by either using kinematic chains [Lowe91] or using Lagrange multipliers [Drummond02] to apply constraints to rigid configuration spaces. In the kinematic chain case error is propagated via a chain back to the root which leads to non-symmetric distribution of the error. In the second case, a two step method is required where the full pose of rigid components are estimated in a first step and then constraints are applied in a second estimation step. This case is worse since not only does it propagate error as in a kinematic chain, but it is also inefficient and prone to error in cases where there is not enough information to estimate the full pose of a particular component. In this thesis a new method is proposed based on minimal parameter subsets which avoids all the inconveniences of these previous methods. Furthermore, this new method is shown to handle the complete and general class of articulated joints including non-linear joints. In retrospective, this kinematic set method has also been related to the Lagrange-d'Alembert formulation in classical physics.

- **Augmented Reality** - The major motivating application in this thesis is that of video see through Augmented Reality. The AR pose computation problem is formulated by way of a 1D/3D non-linear closed loop tracking methodology. The technique is based on an efficient distance-to-contour features as well as a robust statistical framework. This system is shown to be capable of treating complex scenes in a realistic live real-time environment without the need to modify the environment. The closed loop scheme provides accurate and stable estimation of the pose so that virtual objects are rendered onto a real image in a visually acceptable way. The system is based on CAD models of manufactured objects or urban environments.

- **Model-Based Tracking for Visual Servoing and Robust Visual Servoing** - Since Virtual Visual Servoing methods are used for 3D tracking much of this research also benefits traditional visual servoing or image-based camera control. A first improvement over traditional methods consists in designing a robust control law for visual servoing. A second contribution involves using robust model based tracking in place of simple feature tracking within a visual servoing task. These contributions can be summarized as follows:

    1. **Robust Visual Servoing** - A robust visual control law has been designed based on well known statistical M-estimation techniques. This allows continuous estimation of the uncertainty in visual features and rejection of outliers during each step of a real-time visual servoing task. Simulations were performed on small and large data sets to test the theory which was also tested on real visual servoing tasks. Different convergence criteria were also analyzed.
    2. **3D Tracking for Visual Servoing** - Real-time 3D tracking methods are proposed in this thesis which mirror a real visual servoing task via Virtual Visual Servoing. In this way visual features are tracked using VVS for a real VS task or, in other words, the pose of the robot end-effector is estimated at each iteration of an image based visual servoing task and the CAD model is re-projected onto the image. This allows precise and robust estimation of the current position of visual features. A second benefit of VVS is that it allows to improve over existing techniques as it takes advantages of both IBVS and PBVS. This is achieved by performing 3D pose estimation at each iteration of a visual servoing control law. This means that image-based visual servoing can be performed with respect to complex 3D objects. As is usual the desired pose can be used as input in order to project a complex polyhedral CAD models onto the image plane, providing the final image for an image-based visual servoing procedure. The 3D tracking approach introduces advantages including by estimating the pose at each iteration, the depth parameter is known which improves the stability and behavior of the system. Furthermore, this provides all the essential information for globally convergent methods such as $2.5D$ visual servoing. Most importantly, however, is the fact that by using this method, visual servoing tasks can be carried out even when the final parts of the object are not initially visible, therefore making the visual control more flexible. Furthermore, by minimizing using IBVS, the target can more easily be kept within the image frame.

- **Transfer to Industry** A final contribution of this thesis involved the transfer of marker-less augmented reality technology to industry. This was carried out within a national French RIAM project named SORA. The transfer of technology was carried out with the industrial partner Total Immersion who has also integrated this work with their rendering engine in several complex image sequences. This work has also been included in many live demos such as that performed at DEMOgod and they have been awarded for their performance as a European IST prize winner. The transfer of software has been carried out locally via direct collaboration with the engineer Cederic Riou who's aim was to optimize and transfer this work to the windows operating platform.

1.7 Outline

Much thought has gone into the best way to present the various concepts involved in this thesis. It was decided that a traditional bottom up approach would be best by first considering the most basic elements and building up to more complex ones. However, this is more easily said than done and in the end it was also necessary to introduce a certain degree of a top down approach. In particular, the topic is introduced by an overall view in Chapter 3. In the following is an introduction to each chapter.

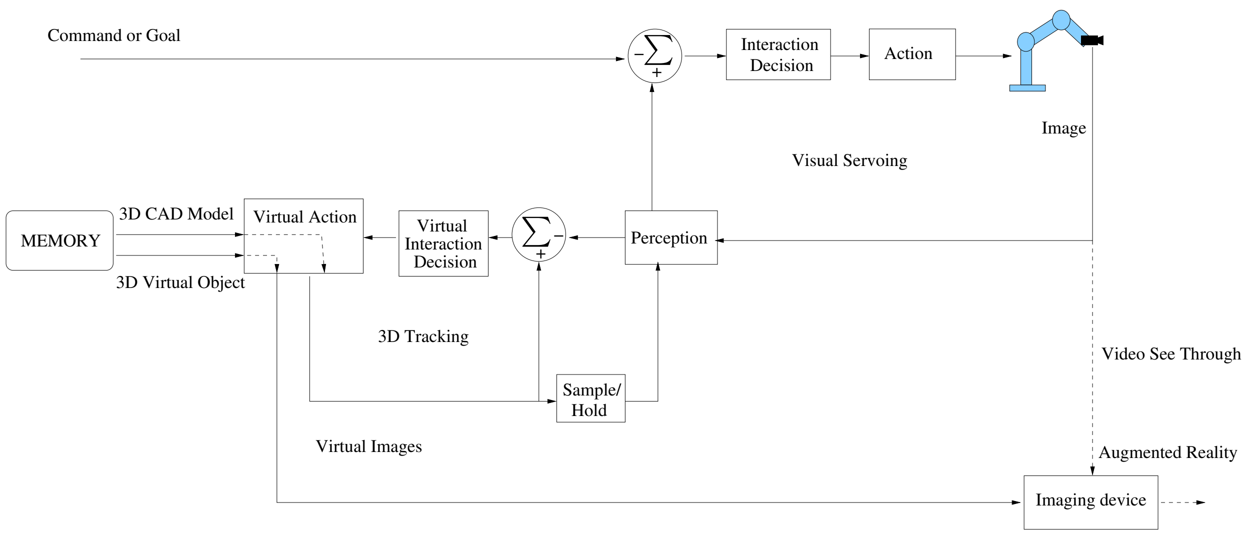


Figure 1.9: A combined block diagram with all the different elements considered in this thesis. On the left are the inputs to the system including the control command, a-priori knowledge about the environment and virtual objects. The only other input is the image stream from the camera. On the right side are the outputs including the action of the robot and an augmented image. The two closed control loops can be seen for both visual servoing and Virtual Visual Servoing.

In Figure 1.9, an overview is given with both visual servoing and Virtual Visual Servoing feedback control loops. It can be seen that Virtual Visual Servoing makes up an important component of the computer imagination. The output of the estimated pose is used to project imaginary objects onto the virtual image and the augmented reality is given at the bottom of the diagram. It can be seen that the perception block is of central importance to both systems. The perception refers to the extraction of visual information from the image. The interaction blocks refer to the interaction matrix for any type of visual feature. The decision to reject or keep visual features is also carried out within the robust interaction block. The mirroring effect between the visual servoing and Virtual Visual Servoing is apparent. Furthermore, it can be seen how both these loops potentially affect each other. Indeed, one would like, in a 3D tracking context that the imaginary loop mimics the real loop as closely as possible.

In Chapter 2, the mathematical foundations of 3D rigid-body motion are presented. In a first part, a pure rotation matrix is defined between reference frames in 3D space. This is then combined with translational motion to define a minimal parameter pose vector along with its homogeneous transformation matrix form. Next the case of rigid body velocity is derived by starting with the rotational velocity and then generalizing this, using the translational velocity, to a full rigid body velocity. The minimal parameter velocity twist is also defined along with a velocity transformation matrix. Lastly, the exponential map is derived so as to complete the mapping between the velocity twist space tangent to the pose manifold space.

In Chapter 3, a general overview is given of the Virtual Visual Servoing method of pose estimation. The chapter begins with an overview of the state of the art in visual tracking and reasons are given for the different choices which have been made. The perspective camera model is given along with the parametric 3D object model. Different estimation configurations are compared to traditional visual servoing schemes including eye-in-hand visual servoing with a camera-based motion model, eye-to-hand based visual servoing with an object based motion model and a combination of these approaches. The core part of this chapter is the definition of the VVS control law which forms the basis of a non-linear iterative minimization scheme. Different convergence criteria are considered and results are presented.

Chapter 4 is based on the problem of low level information extraction from the image. A literature review reveals that edge tracking is very efficient and most interesting for real-time applications. This is due to a one-dimensional search to the normal of a contour. In a first part an in-depth review of both Haralick's facet detector and Canny's edge detector is made. Following this the Moving Edges method, used in this thesis, is presented. This method is based on oriented local edge hypotheses which are tested using a log ratio test criterion. This method is then adapted to the case of 3D object tracking and results are presented. Furthermore, it is shown that edge detection can be made invariant to scale by propagating this unknown to global pose estimation. This means that no edge detection threshold is needed and that stronger edges are not factorized over weaker strength edges(in terms of intensity difference).

Chapter 6 extends the visual control law to a statistically robust estimation paradigm. In a first part, a literature review is given and separated into two groups. There are those methods which came about from the computer vision community such as RANSAC and those which have emerged from the statistics community such as LMedS and M-estimation. All estimation methods are shown to fit under the banner of M-estimation with an auxiliary scale estimate. Different methods for the estimation of scale are also reviewed. The theory of M-estimation is then integrated into the visual control law via an iteratively re-weighted least squares implementation. Different convergence criteria are analyzed and results are presented for simulated scenarios as well as real situations.

In Chapter 7, rigid-body tracking is generalized to articulated objects. After a review of the literature, two prominent methods are presented. The first being the well known kinematic chain model and the second being a Lagrange multiplier approach to applying constraints. Next, a new method is proposed which takes the advantages from previous methods to design a kinematic set based approach. In a first part the overall objective is defined in terms of the articulation matrix which maps rigid-body velocities to a minimal subspace. A general class of joints is defined via a velocity constraint matrix. Joint velocities are then defined and mapped to a common basis by using the velocity transformation matrix. Lastly all the individual subspaces are mapped together resulting in a decoupled articulation matrix. Results are presented which include a selection of different articulated objects with different mechanical links and a comparison is made with rigid object tracking.

In Chapter 8, a conclusion is made and future perspectives are given.

CHAPTER 2

Mathematical Foundations of 3D Motion

2.1 Introduction

The three-dimensional world forms the basis for the study of the motion, kinematics and 3D tracking of rigid and articulated objects. The techniques presented in this chapter are classic and are well founded concepts. This chapter therefore gives a short overview of the essential terminology and definitions which form the basis of the remainder of this thesis. In particular, the set of parameters describing the position and orientation of rigid bodies in 3D Euclidean space will be derived. Since positions and orientation are relative to a reference frame, the definition of a pose can be shown to be equivalent to rigid body motion and rigid-body displacement will also be defined. General equations will be given for transforming and manipulating these quantities. Following this, the set of parameters describing velocities of rigid bodies in 3D space will be defined from the tangent to the 3D space of displacements. Similarly, a general set of equations for transforming and manipulating these quantities will be given. Finally, a relationship will be derived between the space of displacements and that of velocities.

2.2 Rigid Body Motion

Rigid motion or displacement is defined as a transformation which preserves distances and angles between reference frames. In the remainder of this text, 3D rigid motion is referenced by a 6-dimensional *pose* vector and is referred to as Cartesian *rigid-body* motion.

Let $\mathbb{E}^3$ represent three-dimensional Euclidean space defined by the five axioms of Euclid. The position of a point in 3D Euclidean space is defined to be relative to an inertial Cartesian coordinate frame so that a point $\mathbf{P} \in \mathbb{E}^3$ is identified with a point in $\mathbb{R}^3$. The set of three orthonormal axes, representing a reference frame, form the basis for defining the position of a point in 3D using three coordinates as:

$$\mathbf{P} = (X, Y, Z) \in \mathbb{R}^3. \tag{2.1}$$

The motion trajectory of a point can be represented as a parameterized curve:

$$\mathbf{P}(t) = (X(t), Y(t), Z(t)) \in \mathbb{R}^3. \tag{2.2}$$

An idealized rigid body is a collection of particles such that the distance between any two particles remains fixed, regardless of any motions or external forces exerted on the body.

$$||\mathbf{P}_1(t) - \mathbf{P}_2(t)|| = ||\mathbf{P}_1(0) - \mathbf{P}_2(0)|| = \text{constant}. \tag{2.3}$$

Furthermore, the definition of a rigid body is such that the orientation between different points on a rigid body must also be preserved. In other words, the orientation of vectors must be preserved.

A rigid-body transformation is therefore defined as follows:

Definition 2.2.1. *Special Euclidean Transformation and Rigid Body Motion.*

A mapping $\mathbf{m} : \mathbb{R}^3 \rightarrow \mathbb{R}^3$ satisfies the following properties:

1 *Length is preserved: $||\mathbf{m}(\mathbf{P}_1) - \mathbf{m}(\mathbf{P}_2)|| = ||\mathbf{P}_1 - \mathbf{P}_2||$ for all points $\mathbf{P}_1, \mathbf{P}_2 \in \mathbb{R}^3$.*

2 *The cross product is preserved: $\mathbf{m}_*(\mathbf{v} \times \mathbf{w}) = \mathbf{m}_*(\mathbf{v}) \times \mathbf{m}_*(\mathbf{w})$ for all vectors $\mathbf{v}, \mathbf{w} \in \mathbb{R}^3$, where vectors transform according to $\mathbf{m}_*(\mathbf{v}) = \mathbf{m}(\mathbf{P}_1) - \mathbf{m}(\mathbf{P}_2)$.*

The set of all such transformations is denoted as the special Euclidean group $SE(3)$.

There is an inherent duality between the representation of 3D rigid motion over time and a rigid displacement between two reference frames at one time instant. The relative *pose* or rigid motion between two reference frames is, therefore, a spatio-temporal term. In this treatment, rigid body motion will be considered to be a transformation which describes both the spatial position of rigid bodies and how points on a rigid body change with time.

2.2.1 Rotation Matrix

First of all, the special case of rotational motion is introduced. In order to consider the definition of rotations which conserve the rigid body properties (2.2.1) it is necessary to consider a rotation matrix.

The properties of a rotation matrix are such that its columns are mutually orthonormal, $\mathbf{R}\mathbf{R}^\top = \mathbf{R}^\top\mathbf{R} = \mathbf{I}$. The coordinate frame is chosen to be right handed such that $det(\mathbf{R}) = +1$. The space of rotation matrices is defined by:

$$SO(3) = \left\{\mathbf{R} \in \mathbb{R}^{3\times3} : \mathbf{R}\mathbf{R}^\top = \mathbf{I}, det(\mathbf{R}) = +1\right\}. \tag{2.4}$$

$SO(3) \subset \mathbb{R}^{3\times3}$ is called the special orthogonal group under the operation of matrix multiplication and satisfies the axioms of closure, identity, inverse and associativity.

In this treatment, rotations will be parameterized as either rotations around the coordinate axes or by Euler's parameters, however, many other parameterizations exist. These include Quaternions [Altmann86] which give a global parameterizations of rotation at the cost of using four parameters. Canonical exponential coordinates representation of the rotation matrix [Brockett84] which is a minimal representation related to Rodrigues' parameters [Rodrigues40]. Different forms of Euler angles also exist such as ZYZ Euler angles, ZYX Fick angles, YZX Helmholtz angles and ZYX Pitch, Roll, Yaw angles all having the inconvenience of having singularities.

Let $\mathbf{R} \in \mathbb{R}^{3\times3}$ be a rotation matrix. One way of deriving this rotation matrix is to consider Euler's famous theorem:

Theorem 2.2.1. *Any rotation* $\mathbf{R}$ *can be represented as a single rotation* $\theta \in [0, 2\pi)$ *around a fixed axis* $\mathbf{u} \in \mathbb{R}^3$.

Let θ be the angle of rotation and the unit vector $\mathbf{u} = (u_1, u_2, u_3)^\top$ define the axis of rotation. This parameterizations is redundant in the number of parameters, however, this can be eliminated by introducing the constraint that $\mathbf{u}$ has unit norm by dividing by $\sqrt{u_1^2 + u_2^2 + u_3^2}$. The rotation matrix is then given by:

$$\mathbf{R} = \mathbb{I}\cos\theta + (1-\cos\theta)\begin{bmatrix} u_1^2 & u_1u_2 & u_1u_3 \\ u_2u_1 & u_2^2 & u_2u_3 \\ u_3u_1 & u_3u_2 & u_3^2 \end{bmatrix} + \sin\theta \begin{bmatrix} 0 & -u_3 & u_2 \\ u_3 & 0 & -u_1 \\ -u_2 & u_1 & 0 \end{bmatrix} \tag{2.5}$$

A complete derivation for Euler's parametrization can be found in many textbooks on rigid-body motion including [Goldstein02, Shabana89]. It can be noted also that the derivation of this representation is closely related to the derivation of Rodrigues' formula (see Section 2.4) which relates velocities to displacements.

A second method is to consider rotations parameterized by α, β, γ around the principle x, y and z axes respectively. Each elementary rotation is defined by the following matrices:

$$\mathbf{R}_X(\alpha) = \begin{bmatrix} 1 & 0 & 0 \\ 0 & \cos\alpha & -\sin\alpha \\ 0 & \sin\alpha & \cos\alpha \end{bmatrix} \quad \mathbf{R}_Y(\beta) = \begin{bmatrix} \cos\beta & 0 & \sin\beta \\ 0 & 1 & 0 \\ -\sin\beta & 0 & \cos\beta \end{bmatrix} \quad \mathbf{R}_Z(\gamma) = \begin{bmatrix} \cos\gamma & -\sin\gamma & 0 \\ \sin\gamma & \cos\gamma & 0 \\ 0 & 1 & 1 \end{bmatrix}, \tag{2.6}$$

with the matrix describing the overall rotation being composed of the product of these elementary matrices:

$$\mathbf{R} = \mathbf{R}_X\mathbf{R}_Y\mathbf{R}_Z = \begin{bmatrix} \cos\beta\cos\gamma & -\cos\beta\sin\gamma & \sin\beta \\ \sin\alpha\sin\beta\cos\gamma + \cos\alpha\sin\gamma & -\sin\alpha\sin\beta\sin\gamma + \cos\alpha\cos\gamma & -\sin\alpha\cos\beta \\ -\cos\alpha\sin\beta\cos\gamma + \sin\alpha\sin\gamma & \cos\alpha\sin\beta\sin\gamma + \sin\alpha\cos\gamma & \cos\alpha\cos\beta \end{bmatrix}. \tag{2.7}$$

Note that the order of multiplication of elementary rotation matrices is not commutative.

It is often useful to perform the inverse operation and obtain the rotation parameters from the coefficients r_{ij} $(i = 1 \ldots 3,\ j = 1 \ldots 3)$ of the rotation matrix $\mathbf{R}$. If the rotation matrix has been obtained numerically, it is important to make sure that $\mathbf{R}$ is really an orthogonal matrix by testing the properties given in (2.4).

In the case of equation (2.5), $\mathbf{u}\theta$ is obtained by using the following formula:

$$\theta\mathbf{u} = \frac{1}{2\mathrm{sinc}\theta}\begin{pmatrix} r_{32} - r_{23} \\ r_{13} - r_{31} \\ r_{21} - r_{12} \end{pmatrix} \tag{2.8}$$

where $\theta = \cos^{-1}((r_{11} + r_{22} + r_{33} - 1)/2)$ and $\mathrm{sinc}\theta = \sin\theta/\theta$ which is a C^∞ function which cancels out at $(2n+1)\pi, \forall n \in \mathbb{Z}$. A division by zero occurs in (2.8) when $\theta = \pi$ which can be considered explicitly by determining $\mathbf{u}$ as the eigenvector of $\mathbf{R}$ associated with an eigenvalue of 1.

In the case of equation (2.6), α, β and γ can be recovered from the rotation matrix simply as:

$$\begin{aligned} \alpha &= -tan^{-1}\left(\tfrac{r_{23}}{r_{33}}\right) \\ \beta &= sin^{-1} r_{13} \\ \gamma &= -tan^{-1}\left(\tfrac{r_{12}}{r_{11}}\right) \end{aligned} \tag{2.9}$$

2.2.2 Pose and Location

To define a complete rigid motion consider Chasles' theorem which says:

Theorem 2.2.1. *Every rigid body motion can be realized by a rotation about an axis combined with a translation parallel to that axis.*

This parametrization of rigid motion is commonly referred to as a screw motion which will be shown later to have a direct relation with rigid-body velocities.

In general, a rigid transformation $\mathbf{m} : \mathbb{R}^3 \rightarrow \mathbb{R}^3$ is considered as being composed of rotational and translational motions. The minimal representation of a rigid transformation is parameterized as a six parameter pose vector, composed of translation and rotation parameters:

$$^a\mathbf{r}_b = (^a\mathbf{t}_b, ^a\boldsymbol{\Omega}_b) \in \mathbb{R}^6, \tag{2.10}$$

where $\mathbf{r}$ represents the position and orientation between a reference frame a and another frame b, with $^a\mathbf{t}_b = (^at_{b,x}, ^at_{b,y}, ^at_{b,z})$ being the vector of translation parameters between frame a and b along the x, y and z axes and $\mathbf{^a\Omega_b} = (^a\Omega_{b,x}, ^a\Omega_{b,y}, ^a\Omega_{b,z})$ being the rotation parameters between frame a and b around the x, y and z axes.

According to Chasles theory (2.2.1), a 3D point $\mathbf{P}$, relative to a reference frame a (denoted $^a\mathbf{P}$), has coordinates relative to a reference frame b which can be obtained by applying a rotation followed by a translation between the two reference frames:

$$\boxed{^a\mathbf{P} = {^a\mathbf{R}_b}{^b\mathbf{P}} + {^a\mathbf{t}_b}. \tag{2.11}}$$

More generally, the set of all possible configurations or poses of a rigid body can be defined by the space of Special Euclidean transformations:

$$SE(3) = \left\{\mathbf{m} = (\mathbf{R}, \mathbf{t}) : \mathbf{R} \in SO(3), \mathbf{t} \in \mathbb{R}^3\right\}. \tag{2.12}$$

The homogeneous representation of $\mathbf{m}$ is then obtained in matrix form as:

$$\boxed{^a\mathbf{M}_b = \begin{bmatrix} ^a\mathbf{R}_b & ^a\mathbf{t}_b \\ \mathbf{0}_3 & 1 \end{bmatrix}, \tag{2.13}}$$

where $\mathbf{M}$ is an equivalent representation of the minimal vector form $\mathbf{r}$ given in equation (2.10). This set of homogeneous transformations belongs to the 6-dimensional Lie Group of rigid body motions in $SE(3)$.

The transformation of a homogeneous point between two reference frames can therefore be written as:

$$\begin{bmatrix} ^a\mathbf{P} \\ 1 \end{bmatrix} = {^a\mathbf{M}_b} \begin{bmatrix} ^b\mathbf{P} \\ 1 \end{bmatrix}, \tag{2.14}$$

Furthermore, a pose between frame a and frame c can be expressed as a composition of homogeneous transformation matrices as:

$$^a\mathbf{M}_c = {^a\mathbf{M}_b}{^b\mathbf{M}_c}. \tag{2.15}$$

where $^b\mathbf{M}_a = {^a\mathbf{M}_b^{-1}}$ and the inverse transformation is:

$$^a\mathbf{M}_b^{-1} = \begin{bmatrix} ^a\mathbf{R}_b^\top & -{^a\mathbf{R}_b^\top}{^a\mathbf{t}_b} \\ \mathbf{0}_3 & 1 \end{bmatrix}. \tag{2.16}$$

2.3 Rigid Body Velocity

The infinitesimal version of rigid motion is called a *twist* where a twist represents the instantaneous velocity of a rigid body in terms of linear and angular components. A rigid body velocity is defined as a 6-dimensional twist vector $\mathbf{v} = (\boldsymbol{\upsilon}, \boldsymbol{\omega})$ where $\boldsymbol{\upsilon} = (\upsilon_x, \upsilon_y, \upsilon_z)$ is the linear component of the velocity vector and $\boldsymbol{\omega} = (\omega_x, \omega_y, \omega_z)$ the angular velocity. A twist vector is the tangent vector to an element $\mathbf{m}(t)$ of $SE(3)$.

In order to derive a velocity twist, first consider a point $\mathbf{P}$ with coordinates in a spatial reference frame a and another body reference frame b. The body reference frame is attached rigidly to the point so that the body frame does not vary with time. Re-consider the homogeneous relation between a single point as seen in two different reference frames from (2.14) as:

$$^{a}\bar{\mathbf{P}}(t) = {}^{a}\mathbf{M}_b(t)^{b}\bar{\mathbf{P}}, \tag{2.17}$$

where the bar on $\bar{\mathbf{P}} = (X, Y, Z, 1)$ signifies a homogeneous point. Note here that point's motion with respect to the body reference frame is null.

The velocity of the point with respect to frame a_* (where the symbol $*$ refers to an *instantaneous* reference frame) is obtained by deriving the motion of its coordinates with respect to time:

$$^{a_*}\dot{\mathbf{P}}(t) = \frac{d}{dt}{}^{a}\mathbf{P}(t). \tag{2.18}$$

Differentiating equation (2.17) with respect to time gives the velocity of the point in frame a_* as:

$$^{a_*}\dot{\bar{\mathbf{P}}} = {}^{a_*}\dot{\mathbf{M}}_b{}^{b}\bar{\mathbf{P}}, \tag{2.19}$$

where the homogeneous velocity of a point is $^{a_*}\dot{\bar{\mathbf{P}}} = (\upsilon_x, \upsilon_y, \upsilon_z, 1)$.

The velocity of the point with respect to the spatial reference frame a can then be related to its coordinates in the *instantaneous* spatial reference frame $^{a_*}\mathbf{P}$ by using composition (2.15):

$$^{a_*}\dot{\bar{\mathbf{P}}} = {}^{a_*}\dot{\mathbf{M}}_b{}^{a}\mathbf{M}_b^{-1}{}^{a}\bar{\mathbf{P}} = {}^{a_*}\dot{\mathbf{M}}_b{}^{b}\mathbf{M}_a{}^{a}\bar{\mathbf{P}}, \tag{2.20}$$

remembering that $^{a}\mathbf{M}_b^{-1} = {}^{b}\mathbf{M}_a$.

The 4×4 velocity mapping in equation (2.20) is called a twist and is then easily obtained by using equation (2.13) as:

$$^{a_*}\dot{\mathbf{M}}_b{}^{a}\mathbf{M}_b^{-1} = \begin{bmatrix} ^{a_*}\dot{\mathbf{R}}_b & ^{a_*}\dot{\mathbf{t}}_b \\ \mathbf{0}_3 & 0 \end{bmatrix} \begin{bmatrix} ^{a}\mathbf{R}_b{}^{\top} & -{}^{a}\mathbf{R}_b{}^{\top a}\mathbf{t}_b \\ \mathbf{0}_3 & 1 \end{bmatrix} = \begin{bmatrix} ^{a_*}\dot{\mathbf{R}}_b{}^{a}\mathbf{R}_b{}^{\top} & -{}^{a_*}\dot{\mathbf{R}}_b{}^{a}\mathbf{R}_b{}^{\top a}\mathbf{t}_b + {}^{a_*}\dot{\mathbf{t}}_b \\ \mathbf{0}_3 & 0 \end{bmatrix}, \tag{2.21}$$

where $^{a_*}\dot{\mathbf{t}}_b$ is the translational velocity of a point in frame a_* with respect to a point in frame b. In order to obtain the 4×4 transformation matrix $^{a_*}\dot{\mathbf{M}}_b{}^{a}\mathbf{M}_b^{-1}$ of the moving frame b with respect to frame a in a more compact form, it is necessary to examine its special properties as will be described below. To simplify the derivation, the rotational velocity will first be considered in Section 2.3.1 and in Section 2.3.2 the full twist matrix will be defined from its minimal parameter form.

2.3.1 Rotational Velocity

To find the minimal representation of the velocity twist in equation (2.21), first consider the case where a pure rotational velocity exists with $^{a}\mathbf{t}_b = 0$. In this case equation (2.17) is given as:

$$^{a}\mathbf{P}(t) = {}^{a}\mathbf{R}_b(t)^{b}\mathbf{P}, \tag{2.22}$$

where homogeneous coordinates are not needed.

Next deriving with respect to time, as was done in equations (2.19), gives the rotation velocity mapping as:

$$^{a*}\dot{\mathbf{P}} = {}^{a*}\dot{\mathbf{R}}_b {}^b\mathbf{P}. \tag{2.23}$$

Transforming using composition as in (2.20) gives:

$$^{a*}\dot{\mathbf{P}} = {}^{a*}\dot{\mathbf{R}}_b {}^a\mathbf{R}_b{}^{-1a}\mathbf{P} = {}^{a*}\dot{\mathbf{R}}_b {}^a\mathbf{R}_b{}^{\top a}\mathbf{P}, \tag{2.24}$$

where $^{a*}\dot{\mathbf{R}}_b {}^a\mathbf{R}_b{}^\top$ is defined as the rotation twist.

In order to find a compact representation for the rotation transformation twist in (2.24), it can be shown that the matrix $(^{a*}\dot{\mathbf{R}}_b {}^a\mathbf{R}_b^{-1})^\top = -{}^{a*}\dot{\mathbf{R}}_b {}^a\mathbf{R}_b^{-1}$. This can therefore be identified as a 3×3 skew symmetric matrix. More precisely, the defining property of a skew-symmetric matrix is that $[\mathbf{u}]_\times^\top = -[\mathbf{u}]_\times$, for the column vector $\mathbf{u} = (u_1, u_2, u_3)$. A 3×3 skew-symmetric matrix represents a mapping from $\mathbb{R}^3$ to $\mathbb{R}^3$ as:

$$[\mathbf{u}]_\times = \begin{bmatrix} 0 & -u_3 & u_2 \\ u_3 & 0 & -u_1 \\ -u_2 & u_1 & 0 \end{bmatrix}, \tag{2.25}$$

where the operator, $[.]_\times$, is chosen to represent a skew symmetric matrix.

The space of all skew symmetric matrices is denoted by $so(3)$:

$$so(3) = \left\{ [\mathbf{u}]_\times \in \mathbb{R}^{3\times 3} : \mathbf{u} \in \mathbb{R}^3 \right\}. \tag{2.26}$$

Using this definition, the independent components of rotation are therefore defined as the compact vector $\boldsymbol{\omega} \in \mathbb{R}^3$ which is obtained from the elements of the skew-symmetric matrix $[\boldsymbol{\omega}]_\times = {}^{a*}\dot{\mathbf{R}}_b {}^a\mathbf{R}_b{}^\top$.

2.3.2 Velocity Twist

The 4×4 twist matrix from equation (2.21) can now be shown to be related to a minimal 6 parameter vector by defining the angular component $^{a*}\boldsymbol{\omega}_a \in \mathbb{R}^3$ and the linear component $^{a*}\boldsymbol{\upsilon}_a \in \mathbb{R}^3$ of the velocity of a point in the body frame b (possibly imaginary) passing through instantaneous spatial frame a as:

$$^{a*}\mathbf{v}_a = \begin{bmatrix} ^{a*}\boldsymbol{\upsilon}_a \\ [^{a*}\boldsymbol{\omega}_a]_\times \end{bmatrix} = \begin{bmatrix} -{}^{a*}\dot{\mathbf{R}}_b {}^a\mathbf{R}_b{}^{\top a}\mathbf{t}_b + {}^{a*}\dot{\mathbf{t}}_b \\ ^{a*}\dot{\mathbf{R}}_b {}^a\mathbf{R}_b{}^\top \end{bmatrix}, \tag{2.27}$$

Therefore, there exists an operator, based on the skew-symmetric operator $[.]_\times$ for rotations, which transforms a full twist matrix to its minimal vector form defined as:

$$[\mathbf{v}]_\wedge = \begin{bmatrix} [\boldsymbol{\omega}]_\times & \boldsymbol{\upsilon} \\ 0 & 0 \end{bmatrix}. \tag{2.28}$$

The space of velocity twists can therefore be written as a 4×4 homogeneous twist matrix $[\mathbf{v}]_\wedge$ as:

$$se(3) = \left\{ [\mathbf{v}]_\wedge \in \mathbb{R}^{4\times 4} : [\omega]_\times \in so(3), \boldsymbol{\upsilon} \in \mathbb{R}^3 \right\} \subset \mathbb{R}^{4\times 4}, \tag{2.29}$$

where $se(3)$ is the Lie Algebra of the Lie Group $SE(3)$.

Note that the notation in equations (2.18), (2.21) and (2.27) have been chosen carefully so that $^{a*}\dot{\bar{\mathbf{P}}} \neq {}^{a*}\dot{\mathbf{t}}_b \neq {}^{a*}\boldsymbol{\upsilon}_b$ respectively. In fact, $^{a*}\dot{\bar{\mathbf{P}}}$ is the homogeneous velocity of a point in the spatial reference frame a_*. $^{a*}\dot{\mathbf{t}}_b$ is the velocity of the origin of the body reference frame b with respect to the origin of reference frame a_*. Finally, ${}^{a_*}\boldsymbol{v}_b$ is the velocity of a point (possibly imaginary) on the rigid body b which is traveling through the origin of the spatial frame a_* at time t.

Using the results of the previous section from equations (2.17) and (2.27), the velocity of a point $\mathbf{P}$ relative to a spatial reference frame a_* is given as:

$$ {}^{a_*}\dot{\mathbf{P}} = -{}^{a_*}\boldsymbol{v}_a - {}^{a_*}\boldsymbol{\omega}_a \times {}^{a}\mathbf{P}, \tag{2.30}$$

where the vectorial cross product $\times$ is equivalent to multiplication by the skew-symmetric matrix $[\boldsymbol{\omega}]_\times \in so(3)$ or:

$$\boxed{ {}^{a_*}\dot{\bar{\mathbf{P}}} = [{}^{a_*}\mathbf{v}_a]_\wedge {}^{a}\bar{\mathbf{P}}, \tag{2.31}}$$

where $[\mathbf{v}]_\wedge \in se(3)$ is the twist matrix given in (2.28).

2.3.3 Velocity Transformation

Unlike in the case of the homogeneous pose matrix $\mathbf{M}$, the homogeneous twist transformation of a velocity can not be directly applied to a change of reference frame for twists. In this case, a twist transformation can be obtained by first considering the velocity of a rigid-body with respect to the instantaneous body frame b_*. The derivation begins by re-stating the velocity of a point (2.17) in terms of the body frame:

$$ {}^{a}\bar{\mathbf{P}}(t) = {}^{a}\mathbf{M}_b(t){}^{b}\bar{\mathbf{P}}. \tag{2.32}$$

Differentiating equation (2.32) gives, as before, equation (2.18).

Now, in order to obtain the instantaneous velocity of the body with respect to the body reference frame a homogeneous transformation is applied to (2.18) which gives:

$$ {}^{b_*}\dot{\bar{\mathbf{P}}} = {}^{a_*}\mathbf{M}_{b_*}^{-1}{}^{a_*}\dot{\mathbf{M}}_b{}^{b}\bar{\mathbf{P}} = {}^{b_*}\mathbf{M}_{a_*}{}^{a_*}\dot{\mathbf{M}}_b{}^{b}\bar{\mathbf{P}}, \tag{2.33}$$

where the body reference frame is considered to be in movement and the system is invariant to the spatial frame.

Since ${}^{a_*}\mathbf{M}_{b_*}^{-1}$ is unknown it is approximated by ${}^{a}\mathbf{M}_b^{-1}$ to give:

$$ {}^{b_*}\dot{\bar{\mathbf{P}}} = {}^{a}\mathbf{M}_b^{-1}{}^{a_*}\dot{\mathbf{M}}_b{}^{b}\bar{\mathbf{P}}, \tag{2.34}$$

It can also be noted here that one of the homogeneous matrices, ${}^{a}\mathbf{M}_b$, depends on time, while the other does not. This represents the fact that we seek the instantaneous velocity of a frame b^* with respect to the previous position of that same frame b.

Finally, the expression for the body-frame twist matrix is obtained as:

$$ [{}^{b_*}\mathbf{v}_b]_\wedge = {}^{a}\mathbf{M}_b{}^{-1}{}^{a_*}\dot{\mathbf{M}}_b = \begin{bmatrix} {}^{a}\mathbf{R}_b{}^{\top}{}^{a_*}\dot{\mathbf{R}}_b & {}^{a}\mathbf{R}_b^{\top}{}^{a_*}\dot{\mathbf{t}}_b \\ \mathbf{0}_3 & 0 \end{bmatrix}. \tag{2.35}$$

where the subscript b refers to the velocity of the body and the superscript b_* means with respect to the instantaneous body frame.

Similar to equation (2.30) for the instantaneous spatial reference frame a_*, the velocity of a point in terms of the body reference frame b_* can be written using (2.35) as:

$$ {}^{b_*}\dot{\mathbf{P}} = {}^{b_*}\boldsymbol{\omega}_b \times {}^{b}\mathbf{P} + {}^{b_*}\boldsymbol{\upsilon}_b. \tag{2.36}$$

The relationship between an instantaneous twist $[{}^{a_*}\mathbf{v}_a]_\wedge$, in the spatial reference frame and an instantaneous twist $[{}^{b_*}\mathbf{v}_b]_\wedge$ in the body reference frame is obtained by homogeneous composition as:

$$[{}^{a_*}\mathbf{v}_a]_\wedge = {}^{a_*}\dot{\mathbf{M}}_b{}^{a}\mathbf{M}_b^{-1} = {}^{a}\mathbf{M}_b({}^{a}\mathbf{M}_b{}^{-1}{}^{a_*}\dot{\mathbf{M}}_b){}^{a}\mathbf{M}_b{}^{-1} = {}^{a}\mathbf{M}_b[{}^{b_*}\mathbf{v}_b]_\wedge{}^{a}\mathbf{M}_b{}^{-1}, \tag{2.37}$$

which gives:

$$\begin{aligned} {}^{a}\mathbf{M}_b[{}^{b_*}\mathbf{v}_b]_\wedge{}^{a}\mathbf{M}_b{}^{-1} &= \begin{bmatrix} {}^{a}\mathbf{R}_b & {}^{a}\mathbf{t}_b \\ \mathbf{0}_3 & 1 \end{bmatrix} \begin{bmatrix} [{}^{b_*}\boldsymbol{\omega}_b]_\times & {}^{b_*}\boldsymbol{\upsilon}_b \\ \mathbf{0}_3 & 0 \end{bmatrix} \begin{bmatrix} {}^{a}\mathbf{R}_b^\top & -{}^{a}\mathbf{R}_b^\top{}^{a}\mathbf{t}_b \\ \mathbf{0}_3 & 1 \end{bmatrix}, \\ &= \begin{bmatrix} {}^{a}\mathbf{R}_b[{}^{b_*}\boldsymbol{\omega}_b]_\times{}^{a}\mathbf{R}_b^\top & -{}^{a}\mathbf{R}_b[{}^{b_*}\boldsymbol{\omega}_b]_\times{}^{a}\mathbf{R}_b^\top{}^{a}\mathbf{t}_b + {}^{a}\mathbf{R}_b{}^{b_*}\boldsymbol{\upsilon}_b \\ \mathbf{0}_3 & 0 \end{bmatrix}. \end{aligned} \tag{2.38}$$

From this, the calculation can be simplified by using the following identity:

$$\mathbf{R}[\boldsymbol{\omega}]_\times\mathbf{R}^\top = [\mathbf{R}\boldsymbol{\omega}]_\times, \tag{2.39}$$

for which a proof can be found by direct calculation.

Using (2.39) along with the identity for a skew-symmetric matrix $\mathbf{u}_1 \times \mathbf{u}_2 = -\mathbf{u_2} \times \mathbf{u_1}$ gives the twist transformation matrix in (2.38) as:

$$\boxed{ {}^{a_*}\mathbf{V}_{b_*} = \begin{bmatrix} {}^{a}\mathbf{R}_b & [{}^{a}\mathbf{t}_b]_\times{}^{a}\mathbf{R}_b \\ \mathbf{0}_3 & {}^{a}\mathbf{R}_b \end{bmatrix}, } \tag{2.40}$$

which is a 6×6 matrix that transforms twists from reference frame a_* to reference frame b_*. This is commonly known as the *adjoint* map.

Thus a twist in frame b_* can be expressed in frame a_* by using the adjoint matrix:

$${}^{a_*}\mathbf{v}_a = {}^{a_*}\mathbf{V}_{b_*}{}^{b_*}\mathbf{v}_b. \tag{2.41}$$

2.4 Exponential Map

It is often useful to know the relationship between the velocity of a moving body and its pose. This relation is called the exponential map which transforms, exponentially, the velocity vector $\mathbf{v}$ to its corresponding pose $\mathbf{r}$. The exponential map is usually written using homogeneous matrices as:

$$\boxed{ \mathbf{M}_{t+1} = \mathbf{M}_t e^{(\mathbf{v})}, } \tag{2.42}$$

where $\mathbf{M}_t$ is a pose before the applied velocity and $\mathbf{M}_{t+1}$ is the result.

The exponential map relation is found by re-considering the velocity of a point in equation (2.30) as:

$$ {}^{a}\dot{\mathbf{P}}(t) = -{}^{a}\boldsymbol{\upsilon}_b - {}^{a}\boldsymbol{\omega}_b \times {}^{a}\mathbf{P}(t), \tag{2.43}$$

where it is assumed that the rigid-body is moving at a constant velocity.

In homogeneous form this can be written as:

$$ {}^{a}\bar{\boldsymbol{\upsilon}}(t) = [{}^{a}\mathbf{v}_b]_{\wedge}{}^{a}\bar{\mathbf{P}}(t). \tag{2.44}$$

This is a time-invariant linear differential equation which is integrated to give the solution:

$$ {}^{a}\bar{\mathbf{P}}(t) = e^{(t[{}^{a}\mathbf{v}_b]_{\wedge})}{}^{a}\bar{\mathbf{P}}(0), \tag{2.45}$$

where ${}^{a}\bar{\mathbf{P}}(0)$ is the initial position of the point and the scalar $t \in \mathbb{R}$ represents the unit of time. Since the velocity twist transformation is independent of time, this is chosen as $t = 1$ to simplify. Note that the value of time can be considered as a scale factor. This is utilized in the exponential co-ordinate representation of a rotation (mentioned in Section 2.2.1) in order to represent a rotational amount.

In order to compute the exponential map, a pure rotational motion is first considered. The exponential from equation (2.42) is then $e^{[\boldsymbol{\omega}]_\times}$, where $[\boldsymbol{\omega}]_\times \in so(3)$ as given in equation (2.27). The matrix exponential $e^{[\boldsymbol{\omega}]_\times}$ can be written as a Taylor series expansion:

$$ e^{[\boldsymbol{\omega}]_\times} = I + [\boldsymbol{\omega}]_\times + \frac{([\boldsymbol{\omega}]_\times)^2}{2!} + \frac{([\boldsymbol{\omega}]_\times)^3}{3!} + \cdots, \tag{2.46}$$

which is an infinite series and not easily computable.

Equation (2.46) can be obtained in closed form by considering a skew-symmetric matrix $[\mathbf{a}]_\times \in so(3)$ such that $||[\mathbf{a}]_\times|| = 1$. Two basic properties of this matrix are:

$$ [\mathbf{a}]_\times^2 = -(I - \mathbf{a}\mathbf{a}^\top), \tag{2.47}$$

$$ [\mathbf{a}]_\times^3 = -[\mathbf{a}]_\times, \tag{2.48}$$

where it can be noted that equation (2.47) is the projection of $\mathbf{a}$ onto its orthogonal vector space so that $-(I - \mathbf{a}\mathbf{a}^\top)^n = (I - \mathbf{a}\mathbf{a}^\top)$.

Therefore, it is possible to re-write all the terms of (2.46) by first normalizing $[\boldsymbol{\omega}]_\times$ so that:

$$ \mathbf{a}_\times = \frac{[\boldsymbol{\omega}]_\times}{||[\boldsymbol{\omega}]_\times||}, \tag{2.49}$$

where it can be noted that in [Murray94] this normalization is done by setting $t = \frac{1}{||\boldsymbol{\omega}||}$ from (2.45).

Equation (2.46) can then be factored by using property (2.48). Pairing odd and even powered terms of $[\boldsymbol{\omega}]_\times$ and normalizing as in (2.49) then gives:

$$ \begin{aligned} e^{[\boldsymbol{\omega}]_\times} = & \; I + \left(||[\boldsymbol{\omega}]_\times|| - \tfrac{||[\boldsymbol{\omega}]_\times||^3}{3!} + \tfrac{||[\boldsymbol{\omega}]_\times||^5}{5!} - \cdots \right) \tfrac{[\boldsymbol{\omega}]_\times}{||[\boldsymbol{\omega}]_\times||} \\ & + \left(\tfrac{||[\boldsymbol{\omega}]_\times||^2}{2!} - \tfrac{||[\boldsymbol{\omega}]_\times||^4}{4!} + \tfrac{||[\boldsymbol{\omega}]_\times||^6}{6!} - \cdots \right) \tfrac{[\boldsymbol{\omega}]_\times^2}{||[\boldsymbol{\omega}]_\times||}, \end{aligned} \tag{2.50}$$

where the two parentheses contain the Taylor series expansion of $\sin(||[\boldsymbol{\omega}]_\times||)$ and $1 - \cos(||[\boldsymbol{\omega}]_\times||)$.

From equation 2.50, the famous Rodrigues formula [Rodrigues40], for a rotation matrix, is obtained:

$$\boxed{e^{[\boldsymbol{\omega}]_\times} = \mathbb{I} + \frac{[\boldsymbol{\omega}]_\times}{||\boldsymbol{\omega}||}\sin||\boldsymbol{\omega}|| + \frac{[\boldsymbol{\omega}]_\times^2}{||\boldsymbol{\omega}||^2}(1-\cos(||\boldsymbol{\omega}||)), \qquad (2.51)}$$

where $\boldsymbol{\omega} \in \mathbb{R}^3$ and the exponential map gives $\mathbf{R} = e^{[\boldsymbol{\omega}]_\times}$.

Now considering the combined translational and rotational part, the matrix exponential $e^{[\mathbf{v}]_\wedge}$ can also be written as a the Taylor series expansion:

$$e^{[\mathbf{v}]_\wedge} = \boldsymbol{I} + [\mathbf{v}]_\wedge + \frac{([\mathbf{v}]_\wedge)^2}{2!} + \frac{([\mathbf{v}]_\wedge)^3}{3!} + \cdots. \qquad (2.52)$$

In order to determine the exponential map for a complete and general twist the next step consists of simplifying (2.52) so that the previous result for the angular component given in equation (2.51) can be easily inserted into the equation. This is done by transforming the full twist into a form for which the rotational and translational velocities are decoupled. This is achieved by using the following transformation identity:

$$e^{[\mathbf{v}]_\wedge} = e^{\mathbf{M}[\mathbf{v}]'_\wedge\mathbf{M}^{-1}} = \mathbf{M}e^{[\mathbf{v}]'_\wedge}\mathbf{M}^{-1}, \qquad (2.53)$$

where the twist $[\mathbf{v}]'_\wedge$ is invariant to the angular component.

The transformation which creates an invariant twist is given as:

$$\mathbf{M} = \begin{bmatrix} \boldsymbol{I} & \boldsymbol{\omega}\times\boldsymbol{\upsilon} \\ 0 & 1 \end{bmatrix}. \qquad (2.54)$$

Expanding this transformation gives:

$$\begin{aligned} [\mathbf{v}]'_\wedge &= \mathbf{M}[\mathbf{v}]_\wedge\mathbf{M}^{-1}, \\ &= \begin{bmatrix} \boldsymbol{I} & -\boldsymbol{\omega}\times\boldsymbol{\upsilon} \\ 0 & 1 \end{bmatrix} \begin{bmatrix} \frac{[\boldsymbol{\omega}]_\times}{||\boldsymbol{\omega}||} & \boldsymbol{\upsilon} \\ 0 & 0 \end{bmatrix} \begin{bmatrix} \boldsymbol{I} & \boldsymbol{\omega}\times\boldsymbol{\upsilon} \\ 0 & 1 \end{bmatrix}, \\ &= \begin{bmatrix} \frac{[\boldsymbol{\omega}]_\times}{||\boldsymbol{\omega}||} & \frac{[\boldsymbol{\omega}]_\times^2}{||\boldsymbol{\omega}||}\boldsymbol{\upsilon} + \boldsymbol{\upsilon} \\ 0 & 0 \end{bmatrix}, \end{aligned} \qquad (2.55)$$

where the normalizing factor, $||\boldsymbol{\omega}||$, which was introduced for the angular velocity in equations (2.49) and (2.50), carries across onto the linear velocity term $\boldsymbol{\upsilon}$ in the term $[\mathbf{v}]_\wedge$. Here the last term can be simplified by using the identity (2.47) to give:

$$\frac{[\boldsymbol{\omega}]_\times^2}{||\boldsymbol{\omega}||}\boldsymbol{\upsilon} + \boldsymbol{\upsilon} = -\left(1 - \frac{[\boldsymbol{\omega}]_\times^2}{||\boldsymbol{\omega}||}\right)\boldsymbol{\upsilon} + \boldsymbol{\upsilon} = \frac{\boldsymbol{\omega}\boldsymbol{\omega}^\top}{||\boldsymbol{\omega}||}\boldsymbol{\upsilon}. \qquad (2.56)$$

Equation (2.55) is now in a form which is invariant to rotation due to the fact that the calculation can be simplified using $[\omega]_\times\omega = 0$. The infinite series components of the Taylor expansion of $e^{[\mathbf{v}']_\times}$ then becomes:

$$([\mathbf{v}']_\wedge)^2 = \begin{bmatrix} [\boldsymbol{\omega}]_\times^2 & 0 \\ 0 & 0 \end{bmatrix}, \qquad ([\mathbf{v}']_\wedge)^3 = \begin{bmatrix} [\boldsymbol{\omega}]_\times^3 & 0 \\ 0 & 0 \end{bmatrix}, \qquad \cdots \qquad (2.57)$$

Finally, the exponential map of the transformed twist $[\mathbf{v}]'_\wedge$ is obtained by using the exponential map for an angular velocity (2.51) in the Taylor expansion for a full twist exponential (2.52) along with the transformed velocity from equation (2.55) to give:

$$e^{[\mathbf{v}]'_\wedge} = \begin{bmatrix} e^{[\boldsymbol{\omega}]_\times} & \frac{\boldsymbol{\omega}\boldsymbol{\omega}^\top}{||\boldsymbol{\omega}||}\boldsymbol{v} \\ 0 & 1 \end{bmatrix}. \tag{2.58}$$

Re-transforming the map by using identity in (2.53), the final exponential map is obtained as:

$$\boxed{e^{[\mathbf{v}]_\wedge} = \begin{bmatrix} e^{[\boldsymbol{\omega}]_\times} & \frac{(I-e^{[\boldsymbol{\omega}]_\times})([\boldsymbol{\omega}]_\times\boldsymbol{v})+\boldsymbol{\omega}\boldsymbol{\omega}^\top\boldsymbol{v}}{||[\boldsymbol{\omega}]_\times||} \\ 0 & 1 \end{bmatrix}, \quad \text{if} \quad \boldsymbol{\omega} \neq 0 \ . \qquad (2.59)}$$

2.5 Further Readings

The mathematical derivation of rigid-body motion was based on descriptions given in various different sources. For a short but concise description of rigid-body physics applied to visual servoing the reader can refer to the tutorial by Hutchinson et al. in [Hutchinson96]. Another treatment applied to visual servoing is given in [Chaumette02a]. The derivation given in this chapter was inspired from several books which give a more in depth treatment of the topic. Notably, [Murray94] gives a mathematical approach to robot manipulation which has been incorporated into [Ma04] with a computer vision focus. Another good book, however, slightly older, is [Shabana89] which lays down the fundamentals before going on to talk about multi-body systems. For a more mechanics and physics based approach two good books are [Rosenberg77] and [Goldstein02].

2.6 Conclusion

This chapter has provided the fundamental definitions and associated terminology for the study of motion and velocity of rigid objects. This definition of physical 3D space forms the basic model which will be used to relate images acquired from vision based sensors to the 3D structure of objects within the environment. In Chapter 7 the assumptions of rigidity based on the preservation of distances and angles, as given in this chapter, will be reconsidered in the case of non-rigid articulated motion.

CHAPTER 3

3D Tracking: Pose Estimation

Introduction

The aim of this chapter is to define the underlying model proposed for estimating the motion of a rigid-body system as has been defined in Chapter 2 for Augmented Reality (AR) and Visual Servoing (VS) application. Although other models exist, here the motion of a rigid-body system is fully defined with respect to two reference frames in 3D space via notion of a pose and a twist. The notion of 3D object tracking is considered here to be equivalent to pose estimation with prior knowledge about the pose. In order to track or localize objects in a real 3D environment various models exist for relating geometric measurements and *a priori* knowledge to a set of unknown motion parameters. The 3D tracking algorithm thus consists in estimating these parameters from information obtained from a measurement sensor and any *a priori* information about the environment (see Figure 3.1). The 3D tracking problem is difficult to treat in the context of augmented reality and visual servoing applications because it is necessary to simultaneously satisfy real-time constraints (or near real-time) along with estimation precision so that the method remains exploitable. This chapter is therefore dedicated to the development of a model-based tracking algorithm.

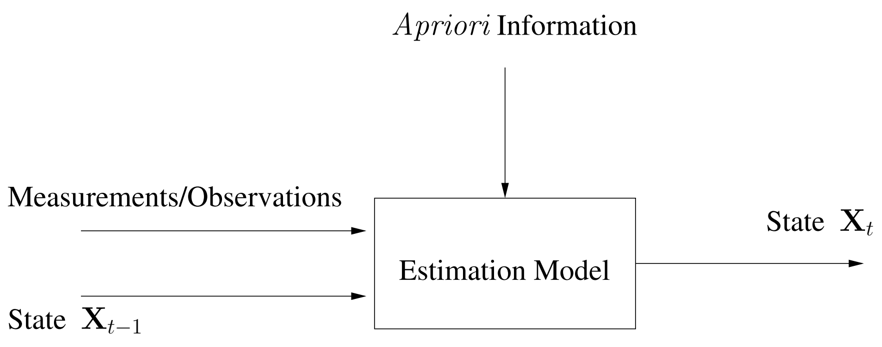


Figure 3.1: Estimation block diagram. The state vector is kept general here, however, for this chapter the state vector represents the relative pose between two rigid-bodies in 3D space.

The measurement sensor considered in this thesis is a camera sensor which can be easily integrated into augmented reality or visual servoing experiments. The problem is thus based on determining the pose of an object as seen from a single view [Dhome89, Lowe91, Gennery92, Koller93, Daucher93, Tonko00, Drummond02, Vacchetti03, Comport03a], or multiple views [Braud97, Martin02]. The later allowing a better precision to be obtained and temporary occlusions of the object to be considered. Image formation is an important issue and it is necessary to choose an adequate model of the measurement sensor. If the internal camera parameters are not available, it is also necessary to calibrate the camera. Both the problem of pose estimation and camera calibration are two old problems in computer vision (such as PnP [Fischler81]– *perspective from n points* –) and photogrammetry [Brown71] which continue to be studied heavily. These parameters are also known as extrinsic and intrinsic parameters respectively.

Many different approaches have been developed in the literature for estimating the relative pose between a camera with respect to the scene and it seems impossible to list them exhaustively. Nevertheless, different techniques can be categorized according to their degree of *constraint* on the unknown parameters (see Figure 3.2). An image stream essentially provides spatio-temporal information about the environment that is used to constrain a system of equations. Due to the discrete sampling process of an image sensor, the continuous relationship between spatial and temporal components is clearly split in two. The spatial constraint is present within each 2D image while the temporal constraint is based on two or more images.

As it stands, pose estimation from a single image is impossible without any *a priori* information about the environment. Thus a monocular camera sensor has been used along with a 3D knowledge about the structure of the object. This strong constraint allows to carry out registration between a 2D image and the 3D environment while remaining efficient and robust. It is possible to consider different types of 3D models which can be classed generally as Computer Aided Design (CAD) models. Indeed many types of prior knowledge are used in the literature. These constraints are usually given manually or obtained in an off-line processing step. Spatial constraints include continuity/discontinuity constraints; large databases of images defining structure from multiple views of the environment; segmented object templates relating different viewpoints of an object; or even the relative positions of interest points. Those methods with strong temporal constraints include constant velocity or even acceleration. Of course there are various levels of strong or weak spatial and temporal constraints that position different techniques with respect to one another. These different methods will be presented later in this chapter.

Pose estimation, in particular, is the focus of this chapter. The pose between a camera and an object over time forms a spatio-temporal trajectory $\mathbf{r}(t)$. Indeed, in order to obtain precise and robust estimates, it is desirable to find the balance between over-constraining and under-constraining the system of equations. This usually depends on the required performance of the system. In augmented reality and visual servoing, target objects are often unpredictable, however, it is desirable to have a precise and robust estimation. In this way, the aim is therefore to develop a method with strong spatial constraints while keeping weak temporal constraints. More particularly, those pose estimation techniques which are more closely associated with pure spatial estimation are based solely on the image at the current time instant t (i.e. [Fischler81, Ganapathy84, Horaud89, Dhome90, Haralick91]). These methods, however, have no prior information about the pose and it is therefore necessary to search the entire manifold (or feature combinations) in order to find the solution (as in the case of pure object recognition). In order to render this search more efficient, it is therefore interesting to introduce a weak temporal constraint so as to reduce this search space. This can be effectively achieved by using the previous pose $\mathbf{r}_{t-1}$, at the time instant $t-1$, to limit this search to a local neighborhood on the manifold. This can be shown to improve estimation efficiency. Thus the pose estimation problem becomes equivalent to a spatio-temporal 3D tracking problem. It will be shown in this chapter that the chosen tracking method is based on a pose estimate at time t that depends on the prior pose at the instant

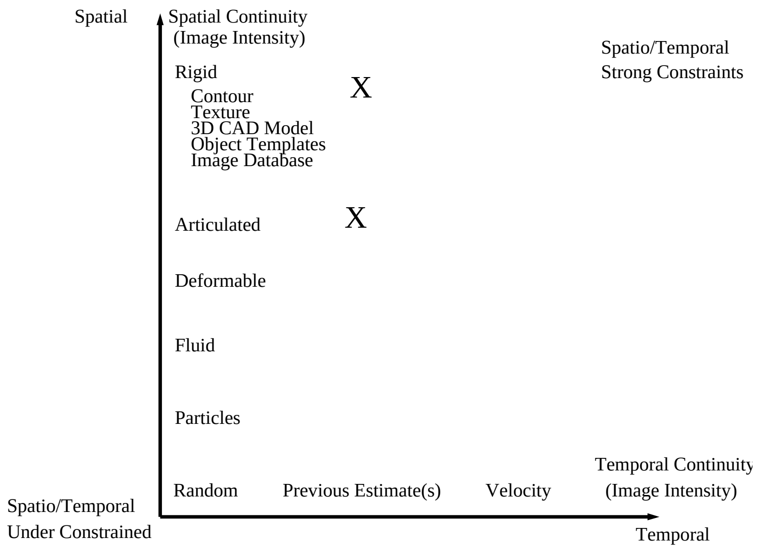


Figure 3.2: Spatio/Temporal constraints used to classify various visual tracking techniques. On the x axis are the spatial constraints and on the y axis are the temporal constraints. The Xs mark the classification of the constraints which have been considered in this thesis. It can be noted that the pose between a rigid body and the camera is a rigid 3Dformulation. If the pose is known to be within a local region of the parameter space then a temporal constraint exist on the current estimate from the previous estimate. The pose can also be used to define multiple rigid objects with rigid constraints between them leading to articulated motion.

$t - 1$. This dependence is both via the initialization of the estimation process as well as the local tracking of image features so as to make correspondences made between the model and the current image. The pose at time $t - 1$ is obtained either from the previous estimate or via an initialization procedure (spatial pose estimation).

The method proposed for estimating the pose is based on a direct analogy between pose estimation and classical image-based visual servoing. In particular, well known visual servoing methods are considered as analogous to pose estimation [Kinoshita94, Sundareswaran98, Marchand02b, Martin02], called Virtual Visual Servoing (VVS) [Marchand02a]. The estimation model is formulated in a general manner so that any number of image sensors or visual features may be used to estimate geometric parameters from a scene. Pose estimation is based on a rigid motion model, a perspective camera model, image measurements, along with a known parametric object model in 3D. Analogous camera-based and object-based motion models are defined and considered for pose estimation. The concepts presented in this chapter provide the basis for the tracking of multiple rigid-body objects, studied in detail in Chapter 7.

Measurement Sensor

Although it has been mentioned already that a monocular camera sensor will be used in this thesis, it is important to mentioned that many different types of sensors are available for measuring relative position and location parameters of objects within an environment. Certain sensors are more adapted to the task of monitoring relative positions within a dynamic and evolving scene.

Amongst the plethora of sensors that provide spatio-temporal information are:

- Intrusive devices: those requiring the placement of instruments on the object of interest including led patterns [Bajura95], markers [Mellor95, Neumann96, State96], GPS allowing position to be determined via triangulation between satellites [Piekarski02], gyroscopes [Klein03, Aron04, Casanova04] providing inertial information, accelerometers, and magnetic sensors [Webster96, State96, Rousseau02] which is limited due to sensor sensitively to metallic or conductive objects and only works in short range (typically 1m radius).

- Active devices: range sensors including 'Time of Flight' laser range finders [Jarvis83, Besl88, Nitzan88] providing distance measures, ultrasonic emitter/detector combinations [Kleeman95], striped light projectors combined with cameras so as to actively measure the depth in the image [Taylor04].
- Passive devices: Camera and or Infrared sensors providing an array of photometric measurements according to a camera configuration.

For a review of tracking technologies, one can consult [Ferrin91, Meyer92]. A more recent summary is also given in [Azuma97, Azuma01] with particular reference to augmented reality applications.

The main interest in this thesis is to obtain 3D-spatio-temporal information, however, many of these sensors do not provide this information directly and it needs to be estimated. Apart from using a single sensor it is possible to consider hybrid methods. In [State96], the magnetic tracker was integrated with video-based pose estimation. Inertial sensors were combined with a camera sensor for fast augmented reality in [Klein03, Aron04]. In the first case visual and inertial information was fused, while in the second inertial information was only used to recover when the visual tracking was lost. In computer vision, stereo [Faugeras93], trifocal [Hartley94, Shashua95, Faugeras98], multi-focal sensors, etc..., have been modeled. More particularly, in pose estimation, multi-camera pose estimation has been studied in [Braud97, Martin02]. Furthermore, much research has been invested in fusing different sources of information [Gordon93, Isard98a, Schulz01, Hue02a]. Of course estimation using a larger and more diverse source of information improves tracking strength and limits its weaknesses. Nevertheless, it is necessary to carefully study individual sensors so that when they are combined, maximum performance may be reached. In this thesis, the implementation presented is restricted to the use of a monocular vision sensor.

One of the most versatile and general sensors is the camera since it provides a very large amount of information. Being a passive sensor which observes the environment, there is no need to modify the natural environment in order to obtain useful information. Cameras provide a rich source of spatio-temporal information which can be exploited in many different ways. Indeed, this dense source of information allows imprecision in spatio-temporal estimation to averaged out so that noise and modeling errors are minimized. Furthermore, extra visual information is indispensable within non-rigid or unknown environment as will be elaborated in Chapter 7.

One of the difficulties or weaknesses of a monocular vision sensor is that it only provides degenerate 2D information about a 3D world at a particular time instant. Indeed, 3D spatial information about the scene is a core part of the localization problem. As already mentioned, a way to circumvent this problem is to assume that a 3D model of the world is already known. It should also be noted that, whatever the sensor used, it is important to create a closed feedback control loop (or an iterative minimization scheme) so that the registration error may be regulated to zero.

In Section 3.1, and overview of the fundamentals of image formation will be considered in order to determine the group of parameters representing its mathematical model. More precisely, these parameters establish the relation between a point in 3D to a point as seen in the image plane, expressed in pixels. In Section 3.2, the 3D CAD model will be described. Following this, different motion models will be defined in Section 3.4. These models include a camera movement model, an object movement model and a joint movement model. In Section 3.6, a non-linear estimation scheme based on Virtual Visual Servoing (VVS) will be given. This will provide the basis for Chapter 4 where the low level edge information will be modeled. In Chapter 5, the interaction between the 3D CAD model and the 2D image features will be derived.

3.1 Camera Model

Up-until now all reference frames of objects have been defined in metric space. In a vision system, however, the camera sensor is at the crux of information and measurements. In order to complete the link between movements measured in the image and 3D motion between the camera and an object, it is necessary to model the physical properties of the measurement sensor. Of particular interest for tracking algorithms is the geometric formation of the image. This is important because it defines a model which relates the 2D geometric information seen in the image to the 3D geometric information present in Euclidean space. In this treatment basic knowledge of optics such as focal length, field of view, angular aperture and lenses are assumed known. In the literature, a large amount of work has also been carried out on photometric properties. While not the major focus of this study, photometric information plays and important role in low level feature extraction.

Camera calibration techniques also provide the basis for accurate estimation of other parameters. In the literature, most calibration methods rely on the extraction of point features [Brown71, Faugeras87, Tsai89, Weng92, Wei94], however, approaches also exist based on lines [Caprile90] or ellipses [De Ma93, Tarel95]. In [Tsai89] the extra parameters involved with camera calibration are estimated and it is also shown to be possible to estimate other unknown parameters to calibrate a robot hand-eye system. More recently, the estimation of the calibration parameters has been performed online in [Simon99] where image registration was performed while simultaneously estimating the zoom. In [Marchand02b] camera calibration parameters, including radial distortion, were estimated simultaneously with the pose from various features. Amongst the various calibration methods it has been shown in [Brown71, Chaumette89] that those which are non-linear prove to be more accurate. It should be noted, however, that in a large majority of methods, calibration is performed off-line using a calibration pattern for precise calibration [Maybank92, Luong97]. This means that they do not need to be estimated online and that they may be estimated more precisely by using a precise calibration pattern composed of strong image features (see Figure 3.5).

It should be noted, however, that projective invariant techniques exist which render estimation process invariant to the camera calibration parameters. A complete treatment of multi-view geometry and projective geometry invariants can be found in [Faugeras93] and [Hartley01]. Nevertheless, it will be shown that it is not difficult to calibrate a camera and that the required performance may be obtained without the need for projective invariants.

Visual servoing minimization approaches can be used for estimating the camera calibration parameters as will be described in this chapter. In the literature estimation of camera calibration parameters while simultaneously performing visual servoing has been shown in [Kinoshita94]. In [Marchand02b] the pose and the camera calibration parameters were estimated simultaneously using the Virtual Visual Servoing framework.

The ***intrinsic parameters*** that define the model of the camera can be divided into three different sets of parameters:

- The focal length which is present in the projection model.
- The transformation between camera coordinates in metric space and pixel coordinates.

- The radial distortion of the image formation introduced by the camera optics.

The following section describes these sets of parameters in more detail.

3.1.1 Projection Model

In this study the sensor chosen is the *perspective projection* or *pinhole* model. This model corresponds to the real physical model of the image formation on the sensor and is in widespread use. The perspective projection of a point corresponds to the simple mapping function:

$$\pi : \mathbb{R}^3 \rightarrow \mathbb{R}^2; \quad \mathbf{P} \mapsto \mathbf{p}_m, \tag{3.1}$$

where $\mathbf{P}$ is a 3D point primitive and $\mathbf{p}_m$ is the corresponding 2D point primitive in metric coordinates. The projection of a point can be written as $\mathbf{p}_m = \pi(\mathbf{P})$.

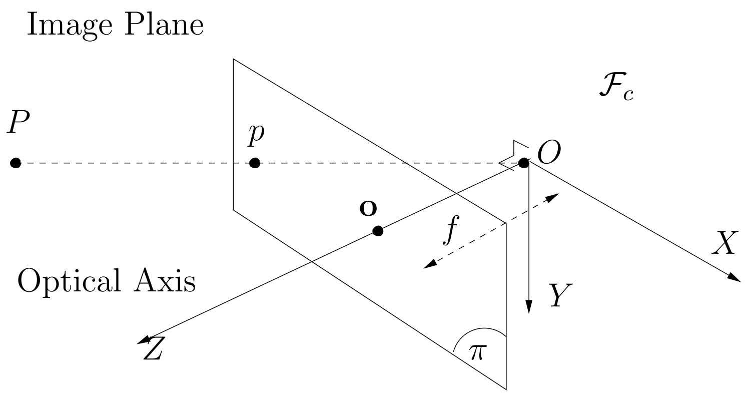


Figure 3.3: Perspective projection camera model. $\mathbf{p}$ is a 2D point corresponding to a 3D point $\mathbf{P}$. O is the center of projection and o is the intersection of the line running through O and perpendicular to the image plane known as the optical axis. f is the focal distance and $\mathcal{F}_c$ is the camera reference frame.

Consider the Figure 3.3. The distance along the optical axis between the center of projection O in 3D and the image center o is the focal length f. Given a straight line joining the point $\mathbf{P} = (X, Y, Z)$ and the center of projection of the camera O (both in 3D, the corresponding point on the image is $\mathbf{p} = (x, y)$. The *perspective projection* is found using similarity as :

$$x = f\tfrac{X}{Z}, \quad y = f\tfrac{Y}{Z}, \tag{3.2}$$

which can be rewritten in matrix form:

$$\begin{bmatrix} x \\ y \\ 1 \end{bmatrix} = \frac{1}{Z} \begin{bmatrix} f & 0 & 0 & 0 \\ 0 & f & 0 & 0 \\ 0 & 0 & 1 & 0 \end{bmatrix} \begin{bmatrix} X \\ Y \\ Z \\ 1 \end{bmatrix}, \tag{3.3}$$

or more simply denoted:

$$\bar{\mathbf{p}} = \frac{1}{Z}\mathbf{F}\bar{\mathbf{P}} \tag{3.4}$$

where the focal length corresponds to the depth of an image point $f = z$ and the 2D point is equivalent to a point $P = (x, y, f)$ in 3D. This is a non-linear equation due to the factor $1/Z$ which does not preserve distances between points or angles between lines.

3.1.2 Imaging Model

The pinhole projection equations describe how rays of light enter the camera and form an image on the imaging sensor. Incoming light is usually sampled, discretely, by CCD photo-receptors which are arranged in a rectangular array and transfer the sampled digitized images into memory. Usually, a grey-scale image contains integer intensity values in the range $\mathbb{I}(u, v) \in [0, 255]$. An important assumption throughout this thesis is that images will always be considered as grey-scale images. A digital image is therefore considered as a mapping of two positive integers u and v, representing a discrete location on the image plane, to a positive integer value describing the intensity of the pixel at that location:

$$\mathbb{I} : \Omega \subset \mathbb{Z}_+^2 \rightarrow \Gamma \subset \mathbb{Z}_+; \quad \mathbf{p}_p \mapsto \mathbb{I}(u, v), \tag{3.5}$$

where $\mathbf{p}_p = (u, v)$, the intensity domain $\Omega = [0, 255]$, the image domain $\Gamma = [0, u_{max}] \times [0, v_{max}]$ and $(u_{max} + 1) \times (v_{max} + 1)$ is the size of the image in pixels. See Figure 4.1 for a 3D plot of the intensity captured at each pixel location.

The transformation between the metric camera frame and the image pixel frame is related via the center of the image frame and the effective size of the pixels by a mapping:

$$\xi : \mathbb{R}^2 \rightarrow \mathbb{Z}_+^2; \quad \mathbf{p}_m \mapsto \mathbf{p}_p, \tag{3.6}$$

where the pixel coordinates are always positive as long as the point is in the image.

This mapping is defined by the following two equations:

$$\begin{aligned} u &= s_x x + u_o, \\ v &= s_y y + v_o, \end{aligned} \tag{3.7}$$

where u_0, v_0 are coordinates, in pixels, of the image center and s_x, s_y are the size of a pixel (in meters) in the horizontal and vertical directions respectively (see Figure 3.4).

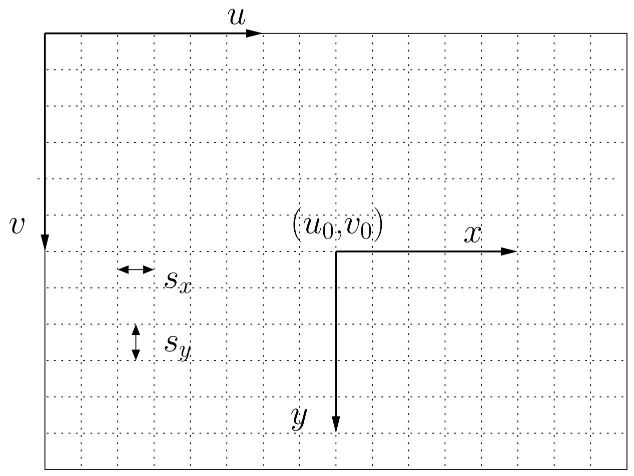


Figure 3.4: Image axes for meter-pixel transformations.

These internal camera parameters can be written as a homogeneous relation:

$$\begin{bmatrix} u \\ v \\ 1 \end{bmatrix} = \begin{bmatrix} s_x & 0 & u_o \\ 0 & s_y & v_o \\ 0 & 0 & 1 \end{bmatrix} \begin{bmatrix} x \\ y \\ 1 \end{bmatrix}, \tag{3.8}$$

This relation can be written more simply as:

$$\bar{\mathbf{p}}_p = \mathbf{K}\bar{\mathbf{p}}_m. \tag{3.9}$$

where it is reminded that the $\bar{\mathbf{p}}$ is expressed in homogeneous coordinates.

If the pixels are not square, a more general form of the imaging matrix can be considered:

$$\mathbf{K} = \begin{bmatrix} s_x & s_\theta & u_o \\ 0 & s_y & v_o \\ 0 & 0 & 1 \end{bmatrix} \tag{3.10}$$

where s_θ is a skew factor of each pixel. Typically the skew factor is very small and this is left out.

If the imaging model 3.8 is combined with the perspective projection (3.2) the matrix transformation according to the camera model becomes:

$$\boxed{\begin{bmatrix} u \\ v \\ 1 \end{bmatrix} = \frac{1}{Z}\begin{bmatrix} s_x & s_\theta & u_o \\ 0 & s_y & v_o \\ 0 & 0 & 1 \end{bmatrix}\begin{bmatrix} f & 0 & 0 & 0 \\ 0 & f & 0 & 0 \\ 0 & 0 & 1 & 0 \end{bmatrix}\begin{bmatrix} X \\ Y \\ Z \\ 1 \end{bmatrix} \qquad (3.11)}$$

3.1.3 Radial Distortion Model

In severe cases the optics used in the imaging process introduce radial distortion (see Figure 3.5). This can be modeled by the following equations:

$$\begin{aligned} x &= x_d(1 + k_1 r^2 + k_2 r^4), \\ y &= y_d(1 + k_1 r^2 + k_2 r^4), \end{aligned} \tag{3.12}$$

where (x_d, y_d) are the distorted coordinates (in meters) of a point (x, y) (in meters), $r^2 = x_d^2 + y_d^2$ and k_1, k_2 are parameters which model the radial nature of the distortion. It can be seen that in this model the radial distortion has increasing influence as the radial distance increases from the center of the image. The internal camera parameters are finally given as $\xi = (u_o, v_o, s_x, s_y, s_\theta, k_1, k_2)$.

The estimation of these parameters can be done online or in an off-line learning step. Appendix A.1 describes the process of estimating these parameters using the Virtual Visual Servoing technique described in the following chapters. From these results the radial distortion is often minimal and can be usually neglected in practice.

3.2 3D Object Model

In Chapter 2 a 3D object model was defined by simple points which were defined relative to a reference frame. The aim of this section is to consider a 3D object representation which is more complex than simple 3D points. In this way more natural and realistic objects may be considered. Since a camera sensor is used, it is also important to keep in mind that this 3D object representation should be closely linked to the visual sensor measurements so that it best represents the perceived visual information.

The definition of an object model requires taking different factors into consideration including:

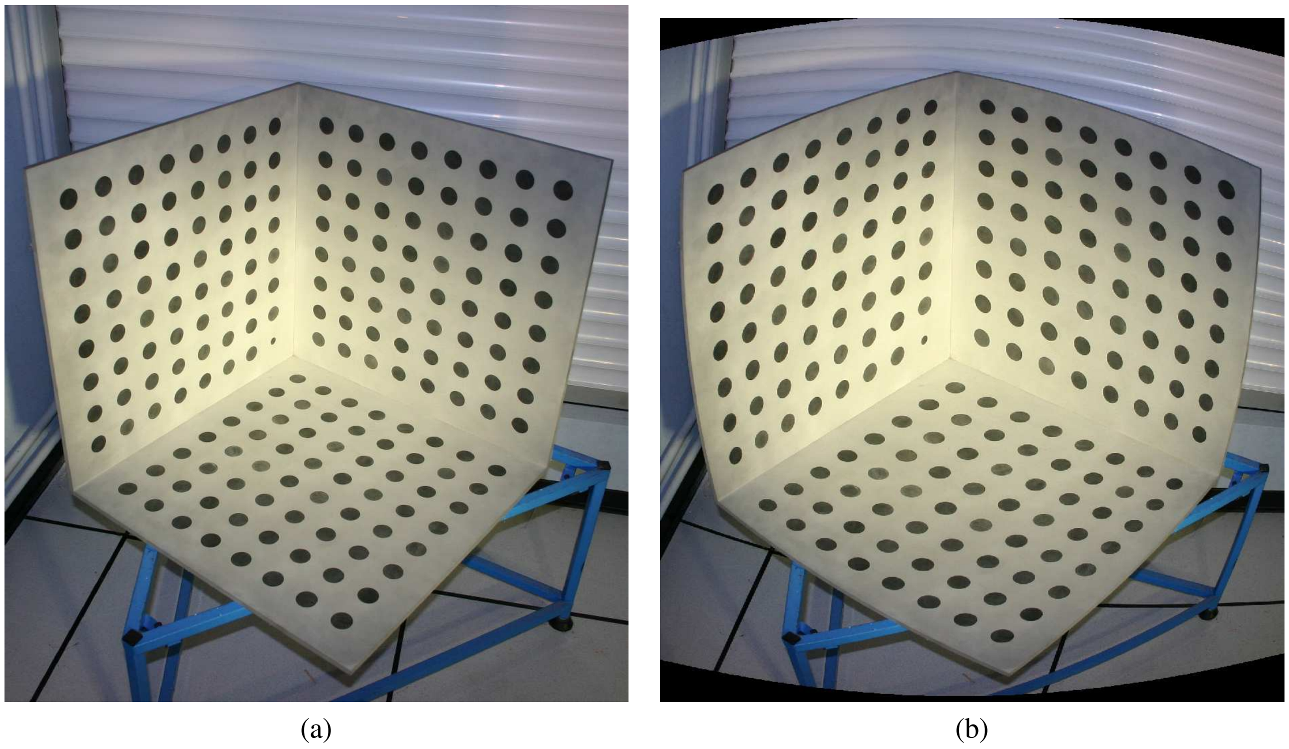


Figure 3.5: Image of a 3D calibration pattern (a) With little radial distortion, (b) With significant radial distortion

1. Representation - the model parameters which provide essential *a priori* information for the estimation process.
2. Reconstruction - How the model is obtained.

3.2.1 Image-based Models

Various methods exist for representing an environment including those which store a large database of raw images from different positions so as to form the basis for a visual memory [Wolf02, Menegatti04, Royer04, Remazeilles04]. Most of the time, however, some sort of off-line processing is performed on the images including features extraction [Harris88] and correspondence [Shi94, Lowe04] or even 3D reconstruction (see further). The difficulty of this type of approach is that images of a dynamic environment are subject to changes such as illumination conditions or even physical movement within the environment.

Two main approaches therefore exist to treating raw images. Firstly, if one is interested in a particular object within the environment it is necessary to segment the object from the background. Secondly, if one is interested in the static environment, it is necessary to determine which parts of the images may no longer be there at a later stage.

Since the interest here is object tracking the second case is the most interesting. This involves segmentation of the images so as to obtain image templates [Kervrann98, Benhimane04, Jurie01, Jurie02] which are smaller images which have been separated from the background. In [Kervrann98], the apparent motion or affine motion model parameters is estimated from 2D object templates. In [Simon02a, Simon02b, Benhimane04, Pressigout04] the planar homography constraint [Faugeras88, Zhang95, Malis00] is used to estimate motion parameters between planar structures and it is shown that it is possible to estimate the full 3D pose up to a scale factor. [Uenohara95] used templates to initialize a model and then used much smaller feature patches to track points on the model and calculate the 3D pose. In [Jurie01, Jurie02] multiple templates are used to estimate the full 3D pose and template matching is based on Hyper-planes for fast convergence. In [Masson04] 3D models are made up of small planar texture patches and used for real-time tracking. Template images are usually chosen from different critical angles around the object so that convergence upon a global minima is assured. These images then form the basis of a 3D object model. The disadvantage of this type of method is that it is highly dependent on intensity information in the template images and therefore they are subject to variation in illumination conditions.K

3.2.2 Reconstruction

It is also possible to obtain a parametric 3D model of the environment from a learning or reconstruction phase which can be performed off-line or online. Since a monocular image alone does not provide any depth information, it is necessary to consider various methods to obtain this unknown information. A large body of work has been carried out on 3D reconstruction and many different approaches exist. Several methods are summarized in the following paragraphs.

Reconstruction of surfaces has been achieved by extracting 3D curvature properties from occluding contours as was first considered in [Koenderink84]. In [Giblin87], surface reconstruction was later performed for planar motion using the orthographic projection image formation model. Perspective projection was considered for reconstruction using known motion in [Cipolla90, Vaillant92] and the central objective of each methods was to compute the unknown depth parameter. This was achieved using the continuity of occluding contours within a minimization framework. In the former method the epipolar plane is used to constrain the system while in the later the radial plane is used. This type of osculating circle approach is widely used in the literature. In [Szeliski93], for example, reconstruction was improved by simultaneously estimating the parameters of epipolar curves on the whole surface leading to robust shape recovery using a linear smoother. More recently 3D surface reconstruction has been performed using occluding contours as in [Boyer97]. Here it is shown that the local shape can be estimated given three consecutive contours without any constraints on the camera motion.

Unknown 3D depth information can also be obtained from a stereoscopic system such as in [Faugeras93] by using triangulation. In [Fua96, Ilic03] a stereo approach is used and a 3D surface is deformed to fit the object in the image. Instead of dealing with static images it is also possible to actively explore the environment with the help of a mobile robot. This method involves the so called structure from motion paradigm. [Xie89] studies dynamic vision via the reconstruction of 3D polyhedral objects using a mobile camera. [Smith94] focuses on controlled exploration of un-calibrated environments for reconstruction. This topic was further studied in [Boukir93] where reconstruction of a static 3D environment was performed using active vision. In [Chaumette96] structure from controlled motion is studied with a particular focus on what sort of motion is required for accurate estimation. In [Marchand96] strategies for actively exploring and reconstructing maps of an environments are presented.

3.2.3 Depth Sensors

As considered in Section 3 various other sensors can be used to obtain geometric information. Obtaining a 3D model of an object can be made much easier using active emitters, such as lasers, to obtain range data [Jarvis83, Besl88, Nitzan88]. In [Grimson96] a model obtained by reconstruction using a laser range-finder is presented for an augmented reality scenario and applied to a medical based application. For a review on range data used for computer vision, a good reference is [Arman93].

The central difficulty in obtaining a 3Dmodel is to find an optimal internal model representation of the object which provides the essential information to the estimation process. Once the 3D information has been obtained it is necessary to form a model from this data. In [Eck96], a method is presented for reconstructing a tensor product B-spline surface, of arbitrary topological type, from a set of previously scanned 3D points. This is achieved by defining a network of B-spline patches. A key ingredient in our solution is a scheme for automatically constructing both a network of patches and a parametrization of the data points over these patches.

3.2.4 Simultaneous Localization and Mapping

Hybrid methods include those which learn the model while tracking is performed. This type of method is known as *Simultaneously Localization And Mapping* (SLAM). This technique basically involves estimating the motion of a sensor as it moves through a largely unknown static environment and continuously updating a map of this environment. This problem has been widely studied in the mobile robotics literature and is of prime importance for exploring unknown environments.

Different types of sensors have been used in a framework based on 2D maps including: user-given landmarks [Thrun98], sonar [Chong99, Leonard91], laser range finders [Bosse04], or millimeter radar [Dissanayake01]. Few methods exist using this method with a monocular vision sensor [Davison02, Bosse03]. The former being combined with a shoulder mounted pan-and-tilt unit and the later based on an omni-directional sensor and using vanishing points and 3D lines. In [Davison03] the vision based SLAM was also presented for an augmented reality application. Unfortunately these methods have an increased computational burden, depending on the number of landmarks used, with respect to those methods that already have an a-priori model. Furthermore, it is necessary to find very stable landmarks so as to avoid drift over time.

3.2.5 3D Contour Models

The ideal environment model needs to provide efficient and accurate information to the pose estimation process. In order to obtain such a model it is necessary to determine the 3D object information that is essential to the estimation of pose parameters. This leads to the idea of retaining a 3D model which provides the most information about the relative geometric position and location of it's visual features. A volumetric object model defines an object's occupation of 3D space, however, it does not take into consideration what information is observed by the camera. In order to study visual information, it is necessary to consider image formation and low level image processing in detail. Therefore, the extraction of low level geometric information from the image will be treated apart in Chapter 4. It is noted here that contour based edge information provides a good source of geometric information while remaining computationally efficient.

In 3D the contours of an object in the image are made up from the object's surface and occlusion points. It is only under the projection of the object's surface onto the image place that contours appear. In this way, contours can be separated into three main classes: those which define the closed boundary of the object, in the image; those which are enclosed within the boundary contour and those formed from occlusion boundaries. In order to determine the appearance of object contours in the image, it is therefore necessary to consider Hidden Surface Removal (HSR) alongside the projection of the 3D object model onto the image. In this thesis, HSR is handled using standard computer graphics based techniques whereby the surface normal is searched for its intersection with other surfaces along the contours (see Appendix B.1).

Many methods in the literature have been based on parametric or wire-frame CAD models. Indeed work in the computer graphics community has significantly advanced computational modeling of 3D worlds and many computational techniques already exist. Many pose estimation methods such as [Fischler81, Haralick89, Dementhon95, Neumann96] are based on simple 3D points. In [Lowe87, Dhome89, Gennery92, Koller93, Kumar94, Tonko97] 3D straight line segments and edges were used to estimate the pose with a gain in precision and efficiency over simple point based methods. In [Liu90, Phong95] and [Ansar02] points and lines were combined to make up the 3D model and estimate the pose. In [Comport03a], lines, circles and cylinders were considered. In [Rosenhahn05], conformal geometric algebra was used to model complex objects in 3D and in [Tahri05] image moments were used to estimate the pose from complex planar shapes. Of course many complex volumetric 3D objects, modeled in the literature, could form the basis for pose estimation, including generalized cylinders [Marr78] or super-quadrics [Borges96].

It is clear that having more knowledge about the world introduces certain advantages over methods without strong models of the environment. Of course, the advantages depend directly on the amount and type of information available in the model. In the case of a full 3D model of the object and environment, these advantages include:

- being able to determine the hidden movement of the object,
- increased robustness by reducing the effect of any external disturbances to the tracking process,
- improved computational efficiency by reducing the search space or
- a more precise 3D representation of the environment without 2D approximations.

It can be noticed that the methods which tend towards the model-less end of the spectrum are better suited to learning and reconstruction of the environment because they are based on using little *a priori* knowledge.

As mentioned in the beginning of this chapter, a known 3D CAD model is assumed to be known *a priori*. Parametric 3D models are considered for points, lines, circles, ellipses, cylinders and spheres. Since the focus of this thesis is on the tracking procedure, techniques for automatically segmenting and reconstructing the model are considered outside the scope of this study. In practice, the CAD model is measured by hand or is known from a design plan of a manufactured object. The 3D parametric surface model is thus defined as follows:

Three-dimensional Surface Parameterizations

A parametric function describing the type of 3D primitive and its configuration in the scene is given by:

$$\boxed{h(X, Y, Z, \Phi) = 0, \qquad (3.13)}$$

where $\mathbf{P} = (X, Y, Z)$ is a 3D point and $\mathbf{\Phi} = (\Phi_1, \dots, \Phi_n)$ are n parameters describing the configuration of the object in 3D. The 3D point $\mathbf{P}$ belongs to the set of all points which constitute the surface model in 3D, defined by $\mathbf{\Phi}$ if (3.13) is satisfied.

Assumption: The assumption is made that the configuration of the 3D object is restricted to displacements on an open set $\mathbb{R}^6$ of SE_3 such that, $\forall \mathbf{r} \in \mathbb{R}^6$, no degenerate cases occur. Of course, this depends on the type of object in question. This assumption means, for example, that a sole line, circle or square configured perpendicular to the image plane cannot occur since they degenerate to a point and segments respectively resulting in a loss rank of the system constraints. However, it can be noted that this degeneracy is rare for more complex 3D objects which often have a highly redundant system of constraints since they are composed of multiple types of visual features with different 3D spatial distributions.

In Chapter 5 explicit models for equation (3.13) will be given for a set of objects. Furthermore, this 3D model will be projected onto the image plane using the perspective projection equation (3.2). Hidden surface removal will be performed so that the visual appearance of a 3D object can be determined from varying viewpoints. This derivation will be given conjointly with a distance-to-contour based interaction matrix derivation.

3.3 Tracking Techniques

There exists a full spectrum of geometric motion parameters which can be used to track objects using image sensor information (refer back to Figure 3.2. Motion parameters can range from the spatio-temporal trajectory of a single parameter particle to the trajectory of a full six parameter 3D-pose and beyond. The dimension of the parameters could potentially tend to an infinite number of unknown parameters incorporating any level of model complexity. In the literature, many variables have been considered including the camera calibration parameters along with the pose parameters. It has been shown in [Espiau93, Malis98, Malis99], however, that image-based non-linear minimization methods are robust to a certain degree of modeling error. Therefore, the model only needs to be accurate and robust enough for the given task.

Pure spatial or temporal methods do not exist since both are intimately related to one another. Even so, as was mentioned previously, various degrees of constraints exist for both spatial and temporal components in a tracking procedure. In the following, both strong temporal and spatial methods will be reviewed separately before looking at more balanced spatio-temporal techniques.

3.3.1 Temporal Motion

Temporal motion estimation methods are based on inter-image measurements and it is rare to find a pure temporal estimation method since an image provides inherently spatial information. Nevertheless, the methods considered here are essentially those which exploit the temporal aspect of this information and the spatio-temporal estimation techniques will be treated in more detail later.

An example of a technique that falls into this class of tracking methods would be temporal correlation of the image intensity between successive images. The least structural elements which can be found in a scene are points [Shi94]. Therefore, the methods based on simple point features could be classed towards the temporal end of the spectrum. Another point based method is [Laptev03] where temporal interest points are fused with spatial interest points so as to provide an elemental spatio-temporal motion event detector. In this case abrupt temporal changes signify temporal interest points. Abrupt temporal changes are often related to a significant event such as the movement extremities of a waving hand or the point of impact of a ball.

Temporal Filtering and Prediction

In the literature there exists a class of methods which are based on the assumption of temporal continuity. These methods filter and predict the system state vector from one time instant to the next. Even though this class of methods is essentially spatio-temporal, the spatial aspect of the method depends on the state vector. Therefore, the temporal filtering is considered here as a temporal aspect. The most common method for introducing strong temporal constraints is that of Kalman filter [Bar-Shalom93]. In [Dickmanns88], the Kalman filter was used for real-time control of a road vehicle within an image sequence. In this paper a road vehicle was shown to be driven automatically along a road using correlation detectors to detect the edges of the road at up to 100km/hr. In [Koller97] a linear acceleration model was used for the camera motion. The linearity of a traditional Kalman filter is not compatible with non-linear state estimation. In this case it is more interesting to use an extended Kalman filter [Bar-Shalom93]. In [Wu89], an extended Kalman filter was used to estimate the 3D motion of a rigid object. In [Taylor03], the extended Kalman filter was used to fuse model cues for model based object tracking.

These types of filtering and prediction are very well suited to cases in which a strong temporal constraint exists. For example, tracking targets at large velocities or with large masses is constrained by gravity and Newton's laws of physics. Unfortunately, these types of methods are not very reactive in situations where the target makes an abrupt change in motion such as the impact of an object with a wall. In this case, undesirable tracking overshoot, or even tracking failure, is observed in the prediction equations. Indeed there are applications such as augmented reality whereby it is not possible to predict the motion of a target since no significant prior temporal information is available. This is particularly the case in user centered Augmented Reality situations where the camera is mounted on a human user.

More recent particle filtering or importance sampling have also become popular in computer vision [Isard98b]. Particle filtering methods are able to handle non-linear motion and can also be used to track in cases where either temporal continuity exists or it does not. Furthermore, these types of methods have been shown to handle multiple targets [Hue02b] or combined with Markov state-machine based methods shown to fuse different state processes [Hue02a, Taylor03]. Another advantage of the particle filter is its robustness to varying sources of noise. Unfortunately, the disadvantage of this filter is that it is necessary to keep track of several states simultaneously. Since in the pose computation case there are 6 degrees of freedom for each state, the number of states which must be sampled can become very large and consequently computationally inefficient and time consuming. Subsequently, in a real-time setting, these methods remain approximate and less precise, however, they are robust an can treat a wide range of situations.

3.3.2 Spatial Motion

Contrary to temporal estimation methods, there are many different techniques for determining the motion of an object or a camera using spatial constraints. In this case it is assumed that there is no prior information about the position or location of the object so that the temporal component is minimal. Thus the main problem considered by spatial estimation approaches is to find the spatial configuration of an object from a single image and using no prior temporal information about this configuration. This type of approach is often required and used to initialize more efficient spatio-temporal tracking methods, which exploit the prior motion of the target.

Object Recognition

The problem of spatial motion estimation is closely related to the problem of *recognizing* or *detecting* an object in the image. Object recognition is based on the idea that the a model of the target object is known *a priori* (not the previous state). In the case where one is only interested in tracking a single target then the amount of prior information is small, however, the aim of object recognition is also to be able to recognize a particular object out of a large number of possibilities. Prior information is usually obtained in the form of images and many preprocessing techniques exist including extracting useful descriptors or even 3D reconstruction. The problem of recognition therefore consists in matching an index image to images in a data-base by comparing the similarity of their descriptors. In this way, if multiple views of a single object are stored, it is possible to recognize the position of the object in the current image relative to the position of the *a priori* images.

Different research problems arise in the area of object recognition including: the difficulty of indexing high-dimensional spaces (the dimensionality curse) and finding stable, transformation-invariant descriptors or objects (including pose estimation).

The problem of organizing and structuring large amounts of data is important so as to allow easy and rapid indexing. Two traditional techniques exist: partition-index methods [Berchtold98] that divide the data space according to the actual values of the data and space-partitioning index methods [Henrich98] that divide the data space along predefined lines regardless of the actual values (see [Bohm01] for a comprehensive survey). The process of indexing these structures consists in searching for the nearest neighbor to an index query [Berrani04].

Content-based image retrieval is also highly dependent on building good image descriptors that can be used for accurate and efficient detection. In the literature a single global descriptor has traditionally been computed for each image such as a color histogram or Principal Component Analysis (PCA) [Murase95, Nayar96a]. This global information allows to retrieve the closest descriptor from a database to the query descriptor. This results in returning globally similar images, however, it cannot retrieve partial matches such as all images containing a certain object. More recently, local image descriptors have been shown to allow "fine grained" indexing [Amsaleg04] and to be robust in terms of image transformations. Indeed much effort has gone into developing descriptors which are invariant to different changes which may occur with the environment. This leads to developing descriptors which are invariant to photometric changes [Schmid97] or geometric changes within the environment such as scale invariant descriptors [Lowe04] or affine invariant descriptors [Rothganger05]. Of course transformation invariant features become very important when different 3D views of an object are considered.

Pure spatial pose estimation methods also exist in the literature which allow combining local and global object descriptors mentioned previously. This class of methods involves estimating the pose of an object within a query image using a 3D model selected from a database. The closest match can then be obtained by comparing the error-norm for each estimated pose and corresponding 3D model. In [Lowe85, Lowe87] a model based recognition was performed where an image was initially segmented into line structures that were likely to be invariant from a wide range of viewpoints. Probabilistic ranking was then used to group the features and reduce the size of the search space. The pose was then estimated from different sets of features and the best estimated pose was chosen. [David02] proposed a method for simultaneous pose and correspondence estimation. Indeed, pose estimation is a particularly good way to make the object detection invariant to geometric location. In [Lepetit04] a recent method is proposed which treats point matching as a classification problem for pose estimation. Until recently, these methods have been too inefficient to be computed continuously in real time. By the use of clustering techniques performed off-line, along with fast and robust feature matching, this method is able to detect and compute the pose in 200ms. In this case each class corresponds to the set of all possible views of a point feature. The system is therefore trained by synthesizing a large number of views for each key-point and a compact description is obtained using classification tools. This type of method has the advantage over tracking techniques in that if the detection fails due to occlusion or leaving the image, it is detected again as soon as it reappears in the image. Although the results are very impressive, this method still remains less efficient than pure tracking techniques. In [Lepetit03] this technique has been used to initialize a more efficient tracking technique.

Pose Estimation

The pose, as defined in Chapter 2, is the representation of rigid-body motion in $SE(3)$ such that distances and angles are preserved under transformations. It is represented as a 6 parameter vector $\mathbf{r}(t)$ composed of 3 degrees of freedom in translation and 3 in rotation. The pose is a spatio-temporal representation of the motion of the object and can be estimated using the spatial information $\mathbf{I}(t)$ measured by a camera at a particular time instant t. Pose estimation can be considered as a spatial estimation method if it is assumed that no prior information about the pose is known. In this section the spatial aspect of pose estimation will therefore be considered.

Many papers in the literature have studied the 3D localization and pose estimation problem. In early approaches, the pose was estimated from a single perspective view along with a 3D model of the target. In this section, all methods considered estimate the pose using a known model of the target. Different approaches include: those which treat the feature matching problem, those that use different linear or non-linear minimization strategies and those which treat robustness.

As in object recognition, pose estimation is highly dependent on correspondences with image features. Indeed, the correspondence problem is a major obstacle for automatic pose estimation. In model based pose estimation this involves making correspondences between a 3D model and a 2D image. Another similar problem is to make correspondences between 2D images [Lucas81, Tommasini98]. The latter approach is wide spread in recognition as well as stereo based systems and it is also needed in model based systems where the representation of the model is in the form of template images. A common technique in the computer vision literature is known as Random Sample and Consensus (RANSAC). The idea is to randomly choose small sets of correspondences between the image and an object model [Fischler81] (see Chapter 6) and to compare their pose estimates for consensus. In this case the sets are made as small as possible so that the pose can still be estimated from them. The pose is then calculated for each set and their poses are compared so as to reach a consensus on the correct pose. One drawback of this method is that testing many combinations can become time consuming. Apart from this technique, it is possible to consider Bayesian approaches. In [Gennery81] a probabilistic approach was used for testing feature correspondences with a model. A probabilistic tree decision structure was used for making correspondences with parts of a model in [Grimson87] and in [Brooks81], combinations of potential matches are evaluated for consistency using symbolic reasoning between 3D models and 2D images. In the next section, it will be shown that this search for correspondences can be made much more efficient when *a priori* temporal information about the pose is available.

Apart from the correspondence problem is the question of solving for the unknown parameters. The 3D pose estimation problem from a single perspective view is inherently non-linear in nature, however, many closed form solutions exist. In order to perform registration, correspondences are typically exploited in the form of a image error to be minimized.

Purely geometric estimation approaches (e.g. [Dhome89]) involve a rigorous analysis of the set of geometric equations relating a camera to objects in the scene. Analytically the minimum number of points required to solve the system of equations for pose calculation is three. This is known as the perspective-3-point problem [Dhome89, Dhome90, Haralick91], however, well known ambiguities in the number of solutions remain [Navab93]. Of course, pose estimation for a symmetric object unavoidably produces certain singularities. In order to solve the problem uniquely, the perspective-4-point algorithm was proposed [Ganapathy84, Horaud89]. It is also possible to consider an analytical derivation for visual features other than points. In [Dhome89, Dhome90] a derivation is given for pose estimation from the perspective projection of three lines and in [Ansar02] linear pose estimation was performed from either points or lines.

Linear approaches use a least-squares method to estimate the pose and *Full-scale non-linear optimization techniques* (e.g. [Lowe87, Lu00, Drummond02]) consists of minimizing the error between the observation and the forward-projection of the model. In this case, minimization is handled using numerical iterative algorithms such as Gauss-Newton or Levenberg-Marquardt. The Gauss-Newton method minimizes by using a first order Taylor series expansion to find a first approximate solution which is then solved again for the next approximate solution iteratively. If the initial starting point close to the solution then this method converges quickly, however, if it is far from the solution the system is ill-conditioned and the estimation may converge slowly or fail to converge. The Levenberg-Marquardt method can be considered like a steepest descent method far from the solution and as a Gauss-Newton as the estimation approaches the solution. In this way the algorithm can avoid the aforementioned problems far from the solution and be fast and accurate close to the solution. The main advantage of these approaches are their accuracy. The main drawback is that they may be subject to local minima and, worse, divergence. Approaches that use a Levenberg-Marquardt approach for pose estimation include [Lowe91] who also used a weighted stabilization method to smooth the estimate. [Haralick87, Haralick88, Haralick89] also worked on pose estimation using a non-linear iterative technique. Unfortunately, these iterative approaches generally converge slowly far from the solution or converge upon local minima and require a large number of measurements for stability, however, in a tracking scenario these techniques are ideal since the inter-frame movement can be considered as small.

Combined *Numerical and iterative* [Dementhon95] approaches may also be considered. In this method a scaled orthographic projection is used in order to linearize the estimation process. A single solution using this approach can be determined when local minima are considered explicitly as in [Lu00, Gao03]. Other multi-step methods exist such as in the calibration procedures [Tsai89, Wei94, Weng92] where in a first pass a linear estimation is considered (for example in the case of Tsai a constraint is added to the radial alignment) for certain parameters and an iterative estimation is used to minimize the rest.

Different combinations of 2D and 3D estimation approaches have also been considered in the literature. In [Lu00] globally convergent pose estimation is performed on video images by minimizing a collinearity error in object space. In this paper an orthogonal iterative minimization algorithm is used for 3D-3D pose estimation. In [Haralick89] different minimization models such as 2D/3D, 2D/2D and 3D/3D are compared for pose estimation. The 2D based minimization approaches, however, allow to minimize the error in the image which is useful for applications such as augmented reality where visual alignement is important.

Various types of features have been used for pose estimation. Most techniques are based on a non-linear iterative minimization process. In [Lowe91, Lowe92] line segments were first segmented from the image and non-linear estimation was performed to estimate the pose. In [Kumar94] line features were also used and a sensitivity analysis was also given. In [Liu90] and [Phong95] points and lines were used to iteratively estimate the pose. In [Marchand02a] a general approach was proposed for considering any type of visual feature and examples of lines, circles and cylinders were considered. More recently more complex visual features have been considered using conformal algebra [Rosenhahn05] and moments [Tahri05].

Robustness and accuracy is also another important aspect of pose estimation. Amongst the various techniques used for pose estimation some use a minimal number of parameters. Methods using minimal sets of features necessary for resolving a set of simultaneous equations include RANSAC and LMedS. In this case, however, the resulting estimation is generally sensitive to noise and the estimation process is inefficient. On the other hand, some methods are based on a redundant number of features [Haralick89] where statistical M-estimation is used to reject data which are at the extremities of a robust distribution. [Kumar94] has also considered and compared two statistical estimation methods, namely M-estimation and LMedS, for estimating the pose. Since there are many open issues the entire Chapter 6 is dedicated to this issue.

3.3.3 Spatio-Temporal Motion

In the following subsections different spatio-temporal techniques which are well adapted to tracking 3D objects are described and compared. Indeed there are a full spectrum of spatio-temporal methods available in the literature ranging from low level event detection to global motion estimation. In this section the focus will be on model based methods with a global tracking focus. In a first part, the evolution of the research carried out in the local VISTA research project is outlined separately. This has been done so as to highlight the origins of the 3D tracking method proposed later in this chapter along with the fact that some research has been undertaken in parallel with techniques proposed in the wider literature. Following this section, 3D tracking methods in general are reviewed.

Local Historic

Within the local VISTA research project, previous attempts estimate spatio-temporal motion are deeply rooted in the research. In light of the 3D model-based approach presented here, this review will concentrate on this type of technique. However, it will be shown in Chapter 4 that the Moving Edges algorithm [Bouthemy89] also provides a fundamental base for spatio-temporal feature extraction and low level tracking.

Previous methods proposed based on 3D CAD models have included [Giordana00], where an affine motion model was first used. In this work, the first step involved initialization of the object's contour in the image using a known 3D object model. Next normal displacements were estimated as in [Bouthemy89] along the projected contour of the object. In a third step, the affine motion was estimated from a modified multi-resolution estimation method [Odobez95]. Apart from introducing a model based initialization, the main contribution of this work was to combine local deformations [Kervrann94] with the affine estimation in order to account for insufficiencies in the 2D model representation. The general idea was to model the shape change as:

$$X^{t+1}(\Theta, \delta) = \Phi_{\Theta}(X^t) + G\delta, \tag{3.14}$$

where X is a vector of sampled points along the projection of the contour, Θ_f is the estimated 2D affine deformation, G is a local deformation factor of the affine motion model.

The affine motion model as in [Kutulakos95] attempts to account for 3D motion as seen in the image plane. The affine 2D transformation is given by:

$$\begin{bmatrix} x^{'} \\ y^{'} \end{bmatrix} = \begin{bmatrix} a_1 & a_2 \\ a_3 & a_4 \end{bmatrix} \begin{bmatrix} x \\ y \end{bmatrix} + \begin{bmatrix} T_x \\ T_y \end{bmatrix} \tag{3.15}$$

where (x, y) is an image coordinate and $(x^{'}, y^{'})$ is the transformed coordinate. The transformation corresponds to an affine transformation given by the parameters $a_1, \ldots, a_4$ and the translation parameters T_x and T_y.

Unfortunately, this model is not particularly adapted to full 3D environments since it cannot completely account for perspective effects of the imaging sensor including large rotational movements or environments with large disparities. However, it is widely used in video analysis where the determination of the 3D structure of the environment is difficult [Odobez95]. In particular, attempts to account for these effects include introducing quadratic displacement parameters (equivalent to a homographic model) which results in significant improvements, however, this is only valid for planar objects.

A further progression was made in [Marchand99b], which involved using the 3D model to remove the bias introduced by the affine motion model instead of using local deformations. The method proceeds as in [Giordana00] for the first three steps. In a fourth step, a common pose estimation algorithm was used to transform the affine estimation into a full 3D pose estimation. The pose estimation consisted of a linear method [Dementhon95], using corresponding object points from the quadratic estimation, followed by a more accurate non-linear pose estimation algorithm [Lowe92]. Unfortunately, this multi-step method meant that affine bias was still present in the estimated parameters. Thus it was necessary to further refine the fit of the data. This was performed by fitting the projection of the known CAD model to the spatial intensity gradient in the image using the following optimization equation (also used in [Haag99]):

$$E(\Phi) = \int_{\Gamma_\Phi} \frac{|\nabla \mathbf{I}_\zeta \cdot \mathbf{n}_\Phi(s)|}{||\nabla \mathbf{I}_\zeta||^2} ds, \tag{3.16}$$

where $\mathbf{n}_\Phi$ is the normal direction of the projected contour, Φ is the pose, Γ_Φ represents the visible part of the object model and $\nabla \mathbf{I}_\zeta$ is the spatial gradient of the intensity function at a point s projected at ζ in the image.

The fitting based on this cost function was implemented using an explicit discrete search algorithm. Due to efficiency constraints a recursive search algorithm was used, inspired by fast block matching [Jain81]. For each discrete search-pose the integral of the gradient along the contour was maximized to determine the next step. Pose estimation was shown to work better than simple 2D displacement estimation and various scenarios and complex objets were considered. Calibration parameters were also estimated using this procedure and robot positioning experiments were presented. This precursor to more recent techniques provided a solid basis for progression in the 3D tracking domain. In retrospective this technique provided slightly less precise results than current state of the art methods mainly due to a multi-step estimation process. In particular, this imprecision lead to significant jitter in the image. Other disadvantages included, the final discrete estimation process never guaranteed converging upon a global minimum. Manually tuned parameters were needed, including the definition of the magnitude and direction for the discrete steps in the discrete search. The pose estimation algorithm was reported to run at 3Hz and 10Hz for two different objects of different size on a 400Mhz PC under Linux OS.

The visual servoing framework [Weiss87, Feddema89, Chaumette90, Espiau92, Papanikolopoulos93a] was used for 3D pose estimation in [Sundareswaran98]. Indeed, the visual servoing framework is similar to traditional non-linear minimization methods, however, the original framework also gives a very general and methodological framework for determining any visual feature's Interaction Matrix (otherwise known as Image Jacobian). This tracking algorithm was used in the new and emerging context of Augmented Reality, however, only marker based point features were used to achieve this.

In [Sundareswaran98, Marchand02b, Martin02, Marchand02a] spatial pose estimation was performed using visual servoing minimization techniques. In [Marchand02a] this approach was coined Virtual Visual Servoing (VVS). In this paper, the authors avoided using simple markers by incorporating the Moving Edge tracker for lines, cylinders and circles [Bouthemy89]. Apart from avoiding the use of markers, the advantage of this approach was an efficient 1D search to the normal of the feature contour. Higher level features such as lines were then estimated using a least squares estimation procedure before using these higher level features to estimate the pose (as in [Lowe92]). As in [Giordana00, Marchand99b], manual initialization was performed by clicking on features in the image and making a correspondence with a 3D model. Given the initial correspondence between a tracked feature and a projected model feature, a Virtual Visual Servoing, task-based control law [Chaumette90, Espiau92] was used to estimate the pose.

This last approach has formed the basis of the contributions made in this thesis. As will be shown later, an oversight in this approach was that the feedback loop was not closed temporally across images to create a fully fledged spatio-temporal tracking process. In this case errors built up in the system until failure (see results in Section 3.7). More specifically, line, circle and cylinder parameters were tracked globally in 2D along the sequence using the Moving Edge algorithm and their positions were not corrected using the estimated pose. This meant that the rigidity constraints of the CAD model did not constrain the geometric 3D relationship between the intermediary features. As such the 2D features continued to deviate from the global model and accumulate error until failure. Another difference between this method and more recent methods is that this approach involved multiple steps rather than a single minimization procedure. Edge points were detected at the low level, higher level feature parameters were then estimated in an intermediary least-squares step and finally the pose was estimated from these intermediary features. Due to these discontinuities small error was introduced into the estimated parameters causing jitter in the pose. It can be noted, however, that this estimation precision is traded-off for a slight gain in computational efficiency. Another issue with this method was its low robustness to outliers in the data. Robust techniques could not be used effectively with this method since the features used for estimation were far from redundant.

3D Tracking

It will be shown here that the various spatial pose estimation methods may be adapted to full spatio-temporal 3D tracking. The adaptation of these methods means that it is necessary to introduce some sort of temporal constraint into the estimation procedure. As mentioned previously, however, strong temporal constraints are undesirable for many applications where the target may change path unpredictably. It is, however, possible to introduce a weak temporal constraint by using the fact that the inter-frame movement of the object is relatively small. In this case it is therefore possible to use the pose $\mathbf{r}_{t-1}$ to initialize the estimation at time t and also to constrain the search space to a local manifold around the previous solution. This can be shown to yield a significant improvement in computational efficiency leading to real-time tracking. Furthermore, this assumption that the initial pose is locally close to the solution means that non-linear iterative minimization methods are well adapted.

Model-based tracking methods are particularly popular in many applications such as in [Koller93] where the model is composed of segments which are used to track cars and road traffic scenes in real-time. In [Koller94] this technique was extended for robust multiple car tracking with occlusion reasoning. More recently in [Koller97] a 3D real-time pose calculation method was proposed for Augmented Reality. In [Wunsch97] another edge based real-time 3D tracking algorithm is presented of which also handles dynamically occlusions. In [Haag98, Haag99], 3D model-based vehicle tracking is also performed for the case of complex driving maneuvers. In this method edge elements are combined with optical flow Estimates. In [Papanikolopoulos93b, Papanikolopoulos95] a visual tracking method is presented for obtaining visual information to control a six degree-of-freedom hand-eye system. Model-based approach are also useful in mobile robotics applications such as in [Ait-Aider02]. In [Yesin04] a model based tracking system is presented for the application of visually guided micro-assembly and the effect of small scale systems is also investigated including precision and micro vision sensor calibration.

Throughout the literature a multitude of 3D tracking and estimation methods exist with different advantages and disadvantages. One of the earliest 3D model-based tracking systems was proposed in [Gennery82]. It was based on tracking edges within a 5 pixel search range using the Sobel edge detector and only a 1D search is required rather that a full 2D search as in the case of a point feature. The efficiency gain by tracking of edges in this way is a very significant advance for real-time applications. In this particular article the pose estimate was updated by a Kalman filter approach. Many early methods for pose estimation were multi-step processes introducing unnecessary error and creating computational inefficiencies. The well known model based tracking method proposed by Lowe [Lowe91, Lowe92] first required the estimation of higher level line parameters before estimating the pose from these features in a second step. More specifically, line segments were extracted from the image and chained together. These segments were then matched with model segments using a probabilistic approach. The lines were then searched for the correct pose via best-first search in a decision tree where the nodes contain at least 4 segments so as to constrain an articulated object of 7 degrees of freedom. The residual error after a least-squares pose estimate was then used to accept or reject a certain pose. Non-linear least squares estimation is performed using a Levenberg-Marquardt method with stabilization using the variance of the measures. The probability of each line belonging to the set in each node of the tree are updated accordingly. Real-time line detection required the use of special hardware. Other multi-step methods have followed in a similar way including [Bray90, Gennery92, Daucher93, Wunsch97], where high level edge features are first estimated, followed by pose estimation.

Another interesting class of methods are those based directly on image intensities. In [Koller97] pose estimation is performed by directly matching polyhedral models to gray value gradients. In this paper, the visual aspect of the contours are modeled with Gaussian noise. This has the interesting property that no low level edge detection is required. An information theoretic approach was proposed in [Viola95], where intensity values are also used directly. The pose estimation technique proposed in this paper is based directly on aligning strips of image intensity and a known model of the target. The pose is then estimated by maximizing the mutual information between the image and a model. More recent approaches based directly on image intensity include [Masson04] where 3D objects are tracked in real-time using 3D models made up of small planar texture patches. In [Pressigout04] both contours and textures are combined. In [Rosenhahn05] pose estimation is performed using 3D free-form contours. This is achieved by using conformal geometric algebra which is a geometric algebra which models entities as stereographically projected entities in a homogeneous model. Impressive results are given for tracking of both rigid and non-rigid objects.

Of course, the different approaches can also be differentiated by the types of information used including different 3D models as presented in Section 3.2 and different visual information, as will be presented in 4. Pose estimation from various types of features has also been mentioned in Section 3.3.2. In [Drummond99] visual tracking is performed using affine transformations of contours which are tracked using an active snake. Points on the image contour are then integrated into the Lie Algebra to control a five degree of freedom robot using visual feedback. In [Marchand02a] spatial pose estimation is also combined with the moving edges contour tracking method to perform pose estimation from lines, circles and cylinders. These and earlier approaches have shown to be efficient since they are based on an efficient 1-dimensional search.

An interesting technique is therefore to use complex 3D polyhedral objects while retaining real time efficiency via the use of contours an a 1D search to the normal. Closed loop estimation methods have also proven to be precise and computationally efficient. In [Drummond99, Drummond02] such a method is presented for tracking complex rigid 3D structures for visual servoing. Distance-to-contour based estimation is performed based on a Lie algebraic framework. Contours model the object in 3D combined with a binary space partition tree for hidden surface removal [Paterson90]. Another advantage of this type of approach over previous approaches is the use of a common distance measure which allows the use of robust statistics such as M-estimation. Unfortunately, in this particular approach, iteratively re-weighted least square is not performed for each image rendering the robust estimation ineffective. Heuristics are also used to increase the robustness of the system including reducing the probability of a match when multiple edges occur, reducing the probability of a point when it approaches the edge of the image, using a test threshold for edge strength and weighting those measures which condition better the Least-squares matrix. Visual servoing is performed via a teaching by showing approach combined with the tracking algorithm. This approach is compared to the approach proposed in this thesis in more detail in Chapter 5.

Another recent tracking method with impressive results is [Vacchetti04b]. In this paper stable real-time 3D tracking of textured objects is achieved by taking advantage of online and off-line information. Here the tracking problem is formulated in terms of local bundle adjustment and stability is obtained by establishing image correspondences that can handle both short and wide-baseline matching. The wide baseline matching is due to merging information from preceding frames with information obtained in key-frames which were created during a training stage. This has been shown to avoid jitter and drift. Real-time capabilities are reported but depend on hardware accelerated functions to implement many critical parts.

From this review, it is clear that estimating the unknown parameters simultaneously has advantages over having multiple estimation steps (i.e. low level feature extraction, intermediary feature estimation and then pose estimation). A closed loop estimation process is also advantageous so that and errors in the system are minimized. The technique proposed in this thesis is based on the Virtual Visual Servoing based control law. In this way, it is possible to draw strong analogies between pose estimation and image-based visual servoing techniques. The aim of the following sections is to present the visual servoing framework and its analogous Virtual Visual Servoing method. In the next section the different visual servoing configurations will be described and the equivalent Virtual Visual Servoing motion models will be defined.

3.4 Motion Model

There are different motion models which may be considered for estimating the pose including the movement of the camera, object or both. In this section these different models will be presented and analogies will be drawn with their visual servoing counterpart. In particular, camera and object-based minimization schemes will be considered. It is noted in particular that the object-based motion model forms the basis for articulated motion which will be treated in Chapter 7.

3.4.1 Image-Based Visual Servoing

The fundamental principle of the proposed approach is to define the pose computation problem as the dual problem of 2D visual servoing [Chaumette90, Espiau92, Hutchinson96]. Visual servoing techniques consist in using the information provided by one or several cameras in order to control the motion of a robotic system. More often than not, visual servoing systems are eye-in-hand systems whereby a camera is mounted on a robot end-effector. A visual feedback loop is then closed via the image in order to control the robot degrees of freedom. It is also possible to consider an eye-to-hand based system, however, in this case it is more difficult to close the visual feed-back loop as it involves tracking the end-effector of the robot.

The goal of classical 2D image-based visual servoing [Chaumette90, Espiau92] is essentially to minimize the error Δ between a vector of n image sensor features $\mathbf{s}$ seen at the current state and their desired value $\mathbf{s}^*$ extracted from the current image (see Figure 3.6(a)). The desired features are usually provided to the robot controller in the form of a desired image that is often obtained from a teaching by showing approach. The formalism of visual servoing is based on the definition of a visual servoing closed-loop control law which regulates this error to zero. This error is defined according to the task function approach [Samson91]. If the vector of visual features is well chosen, there is only one final position of the camera that allows this minimization to be achieved.

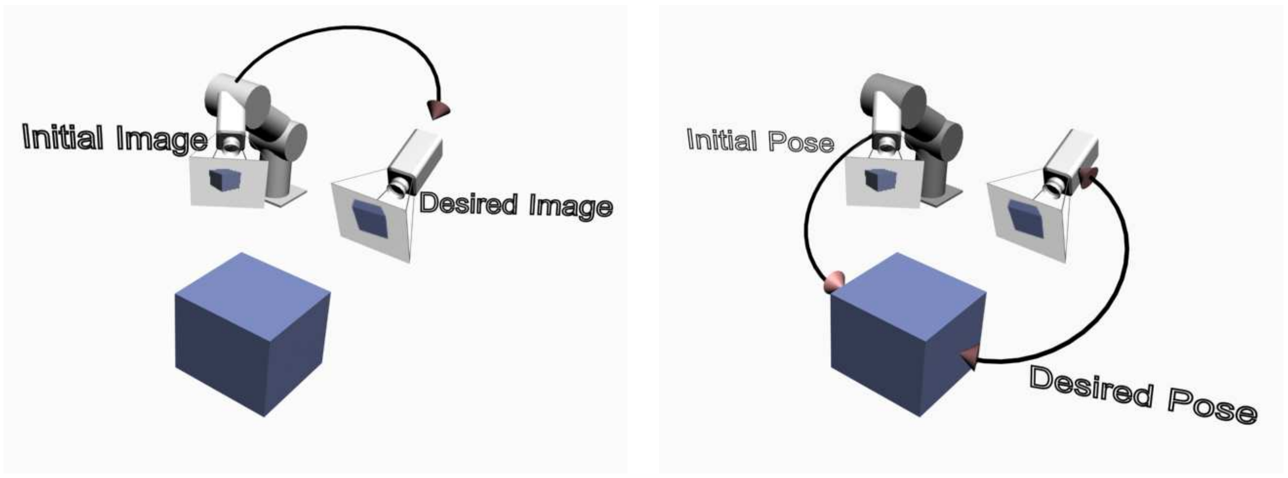
The input to the control law is thus the sensor measures and the output is a velocity command which is sent to control the m degrees of freedom of a robot. A visual sensor provides a potentially large range of information which can be used to control the robot where classically, features such as points, lines and ellipses [Chaumette90, Espiau92] as well as more complex ones such as moments [Chaumette04] have been used.

The regulation of the error Δ is therefore defined as:

$$\Delta = \Big(\mathbf{s}({}^c\mathbf{r}_o(t)) - \mathbf{s}^*\Big), \tag{3.17}$$

where the current position of the visual features, $\mathbf{s}({}^c\mathbf{r}_o(t))$, depends on the pose of the camera (placed on the robot end-effector) with respect to the object reference frame.

The counterpart of the image-based visual servoing method is the position-based method [Hutchinson96, Wilson96, Martinet97] whereby the desired pose ${}^{c^*}\mathbf{r}_o$ is provided to the robot or estimated from the desired



(a) (b)

Figure 3.6: (a) 2D Image-based Visual Servoing where the error to be minimized is defined by features within the image plane, (b) Position-based visual servoing where the error is defined by the difference between two poses in SE(3).

image (see Figure 3.6(b)). In this case the error is defined from the desired pose and the current pose ${}^c\mathbf{r}_o$ in 3D Euclidean space. This approach therefore depends on a good pose estimation technique to provide the pose of the robot. With this type of control law, however, it is difficult to ensure that the object of interest will remain in the field of view of the camera. A more detailed comparison of these two approaches can be found in [Cervera03, Deng03].

The position-based visual servoing method therefore minimizes:

$$\Delta = \left({}^c\mathbf{r}_o(t), {}^{c^*}\mathbf{r}_o\right) = \left({}^c\mathbf{r}_{c^*}\right), \tag{3.18}$$

where the differential pose ${}^c\mathbf{r}_{c^*}(t)$ is the pose between the current pose ${}^c\mathbf{r}_o(t)$ and the desired pose ${}^{c^*}\mathbf{r}_o$. This is obtained by multiplying the corresponding homogeneous matrices using equation (2.13) as ${}^c\mathbf{M}_{c^*} = {}^c\mathbf{M}_o{}^o\mathbf{M}_{c^*}$. The pose vector to be minimized is therefore obtained using the exponential map.

It is possible to use the pose to perform image-based visual servoing with respect to a complex 3D object as soon as a 3D CAD model of the object is available. To illustrate the principle, consider a 3D object model with various 3D features $\mathbf{S}$ as defined in Section 3.2. The desired pose of the camera reference frame with respect to the object frame is given as the pose ${}^{c^*}\mathbf{r}_o$. The desired pose can then be used to project the 3D model of the object onto the image providing an equivalent to the desired image features in the traditional image-based case.

The image-based visual servoing method therefore minimizes:

$$\Delta = \left(\mathbf{s}({}^c\mathbf{r}_o(t)) - \mathbf{s}^*({}^{c^*}\mathbf{r}_o)\right) = \left[\mathbf{s}({}^c\mathbf{r}_o(t)) - pr({}^{c^*}\mathbf{r}_o, \xi, {}^o\mathbf{S})\right], \tag{3.19}$$

where the current position of the visual features still depends on the pose of the camera $\mathbf{s}({}^c\mathbf{r}_o)$ and the final position of the end-effector is defined by forward-projection $pr(.)$ of the 3D model features, according to the desired pose ${}^{c^*}\mathbf{r}_o$ and the camera calibration parameters ξ.

3.4.2 Camera-based Motion Model

Historically, the central concepts of *virtual displacement* and *virtual work* have allowed the extension of mechanics from the Newtonian point of view to the Lagrangian. In a similar way, the notion of *virtual* *velocity* is used to formalize the estimation of position and orientation between two reference frames in Euclidean space. Combined with a minimization process, this notion of virtual velocity can be considered under the Virtual Visual Servoing framework. In Section 3.6 the minimization control law will be used to design a 3D tracking and pose estimation algorithm.

An explanation is now be given as to why the pose computation problem is very similar to visual servoing. Once again an object CAD model is available with various 3D features $\mathbf{S}$. A virtual camera is defined whose position in the object frame is defined by ${}^c\mathbf{r}_o$. The approach consists in estimating the real pose by minimizing the error Δ between the observed data $\mathbf{s}^*$ (usually the position of a set of features in the image) and the position $\mathbf{s}$ of the same features computed by forward-projection according to the current pose:

$$\Delta = \Big(\mathbf{s}({}^c\mathbf{r}_o(t)) - \mathbf{s}^*\Big) = \Big[pr({}^c\mathbf{r}_o(t), \xi, {}^o\mathbf{S}) - \mathbf{s}^*\Big], \tag{3.20}$$

It is supposed here that intrinsic parameters ξ are available but it is possible, using the same approach to also estimate these parameters.

In this formulation of the problem, a virtual camera is moved (initially at ${}^c\mathbf{r}_o(0)$) using a visual servoing control law in order to minimize this error Δ. At convergence, the virtual camera reaches the position ${}^{c^*}\mathbf{r}_o$ which minimizes this error (${}^{c^*}\mathbf{r}_o$ will be the real camera pose). Considering that $\mathbf{s}^*$ is computed (from the image) with sufficient precision is an important assumption (this will be treated in further detail in following chapters).

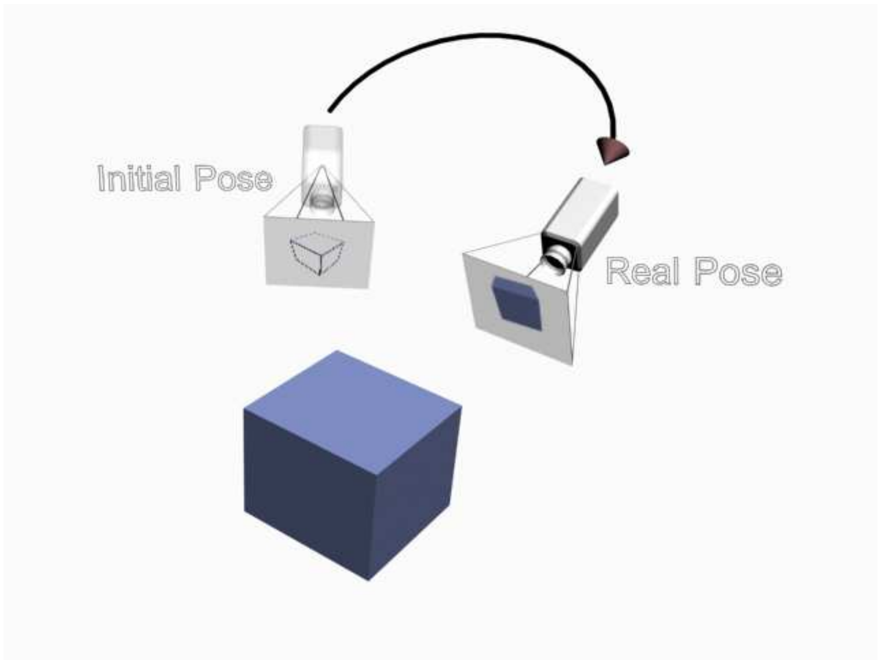


Figure 3.7: Camera Based - Virtual Visual Servoing (Eye-in-hand). The initial pose of the virtual camera is shown on the left by a half visible camera. The model of the box is projected onto the image of this virtual camera and an error is defined between the projected contours and the real image of the box. This error is minimized when the virtual camera reaches the real pose (relative to the solid camera on the right side).

In terms of visual servoing, most algorithms fall into the equivalent of a camera-based motion model where the camera is placed on the end-effector of the robot [Hashimoto93, Hutchinson96]. If the movement seen in the image is attributed uniquely to the movement of the camera then the dual in terms of visual servoing is an eye-in-hand system. This can be seen in Figure 3.7.

An important difficulty in minimizing (3.20) is to derive the relation $\dot{\mathbf{s}}$ which links the movement of features in the image to the velocity twist of the camera ${}^c\mathbf{v}_{c^*}$. Note that the velocity twist is the instantaneous velocity between the camera reference frame c and the desired camera reference frame c^*. This relation is defined as:

$$\dot{\mathbf{s}} = \frac{\partial \mathbf{s}}{\partial {}^c\mathbf{r}_{c^*}} \frac{d {}^c\mathbf{r}_{c^*}}{dt} = \mathbf{L_s} {}^c\mathbf{v}_{c^*}, \qquad (3.21)$$

where $\mathbf{L_s}$ is called the interaction matrix [Chaumette90, Espiau92] or image Jacobian [Hutchinson96] representing the mapping from the variation of the pose to the variation of a feature in the image.

3.4.3 Object-based Motion Model

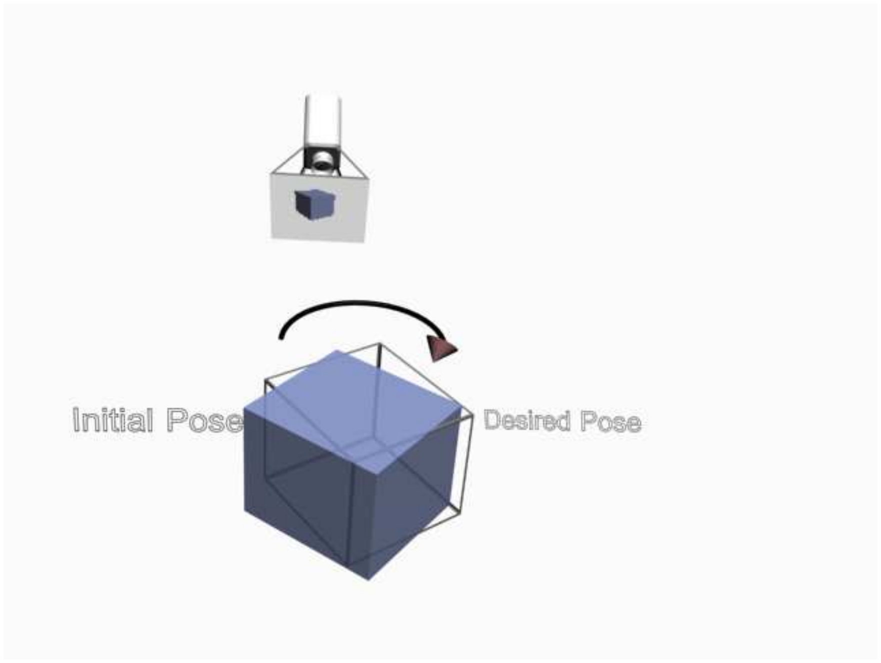


Figure 3.8: Object Based - Virtual Visual Servoing (Eye-to-hand). The initial pose of the virtual object is shown by the wire-frame model of the object. This model is moved to minimize the error in the image. This error is minimized when the virtual object reaches the real pose (relative to the solid object).

Here an object-based motion model is presented (refer to Figure 3.8) and it is noted that this type of motion will form the basis for more complex inter-object motion models in Chapter 7 on articulated motion.

If the movement seen in the image is attributed uniquely to the movement of the object then the dual in terms of visual servoing is an *eye-to-hand* system. The link between $\dot{\mathbf{s}}$ and the movement of the object ${}^o\mathbf{v}_o*$ is by definition:

$$\dot{\mathbf{s}} = \frac{\partial \mathbf{s}}{\partial {}^c\mathbf{r}_{c^*}} \frac{\partial {}^c\mathbf{r}_{c^*}}{\partial {}^o\mathbf{r}_{o^*}} \frac{d {}^o\mathbf{r}_{o^*}}{dt} = \mathbf{L_s} {}^c\mathbf{V}_o {}^o\mathbf{v}_{o^*}, \qquad (3.22)$$

where ${}^o\mathbf{v}_o*$ is the instantaneous velocity twist between the object reference frame o and the desired object reference frame o^*. ${}^c\mathbf{V}_o({}^c\mathbf{r}_o)$ is the twist transformation matrix given in equation (2.38) on page 40 which depends on the current virtual pose ${}^c\mathbf{r}_o$ between the camera frame c and the object frame o. Note that ${}^o\mathbf{r}_{o^*}$ can be considered as analogous to a robot's joint parameters, however, in this case the robot is a single link with a full six degree of freedom joint to the camera or base reference frame. This idea will be taken further in Chapter 7 where articulated objects will be considered. A change in sign can be noted between equations (3.21) and (3.22) due to the change in the configuration of the visual features with respect to the unknown parameters (in this case the velocity of the object). For example, if the camera moves to the right w.r.t. an object, then the visual features in the image move to the left. In order to obtain the same movement of the features in the image by moving the object, then it is necessary to move the object to the left with respect to the camera (thus a change in sign).

In terms of visual servoing, methods that use the equivalent eye-to-hand model include [Nelson94, Kelly96]. Further methods include [Allen93, Cipolla97, Hager95a, Horaud98, Ruf99a] in which a pair of external stereo cameras are used. Ideally an object-based minimization error should also be obtained so that the errors in the model are minimized. It can be noted that various 2D and 3D minimization functions have been studied for pose estimation in depth in [Haralick89].

3.4.4 Camera and Object Motion Model

In eye-in-hand visual servoing it is often possible to decouple the movement of the objects in the scene from the movement of the camera since the movement of the robot can be measured separately. In the Virtual Visual Servoing counter-part, the movement seen in the image is ambiguously attributed to either the movement of the object or the movement of the camera. Even so, if these movements are considered as distinct, then the link between $\dot{\mathbf{s}}$ and the movement of the camera ${}^c\mathbf{v}_{c^*}$ and the object ${}^o\mathbf{v}_{o^*}$ is given as:

$$\dot{\mathbf{s}} = \mathbf{L_s}{}^c\mathbf{v}_{c^*} - \mathbf{L_s}{}^c\mathbf{V}_o{}^o\mathbf{v}_{o^*}, \tag{3.23}$$

If both ${}^c\mathbf{v}_{c^*}$ and ${}^o\mathbf{v}_{o^*}$ are unknown, then it is possible to redefine (3.23) as:

$$\dot{\mathbf{s}} = \varrho\mathbf{L_s}{}^c\mathbf{v}_{c^*} - (1-\varrho)\mathbf{L_s}{}^c\mathbf{V}_o{}^o\mathbf{v}_{o^*}, \tag{3.24}$$

where ϱ is a constant representing the portion of the movement attributed to the camera and $(1-\varrho)$ represents the portion of movement attributed to the object.

If ϱ is unknown there are an infinite number of possible solutions. Indeed, with only one camera and one object, the estimated motion remains ambiguous. In general, however, it is not possible to discriminate between the two types of movement without a higher level contextual model of the scene or extra information. It has been shown in [Odobez95, Csurka99] that the dominant motion in the image can in general be attributed to the camera while the subsidiary motions are attributed to moving objects. Other options include using extra sensors such as a second external camera or a position sensor which can be used to differentiate between the motion of the first camera and the object. In robotics applications the odometry or the position of the robot is usually readily available. This technique is therefore left open for these specific scenarios and in the remainder of this thesis only camera or object-based motion models will be considered.

3.5 Multiple Feature Stacking

Within the formulation of the interaction matrix there are two general cases of visual information which can be considered. The first case being when the number of linearly independent components ι in $\mathbf{s}$ is equal to the number of degrees of freedom m (i.e. $\iota = m$, where $m = 6$ in the case of the pose). In the case the unknown parameters may be resolved, however, it is important to verify that no singularities or local minima exist. For example, in the case of point features at least 4 non co-linear points are needed so as to avoid ambiguities (i.e. $dim(\mathbf{L_s}) = 8 \times 6$). The second case being when the system is rank deficient and the number of linearly independent components ι are less than the number of degrees of freedom (i.e. $\iota < m$). In this case the system of equations can be said to be under-constrained. If this occurs then tracking may continue, however, certain degrees of freedom will remain uncontrolled.

Another case which can be considered is the number of linearly dependent equations. In the case where there are no linearly dependent equations then a single feature controls each degree of freedom. On the other hand, when there exists a redundant number of equations in $\mathbf{s}$ (i.e. $n > m$, where $n = dim(\mathbf{s})$), then the noise and errors in the measurements are averaged across all linearly dependent equations leading to improved statistical robustness in a least squares sense.

Next it is possible to consider combining various different features in $\mathbf{L_s}$. First of all the rows of $\mathbf{L_s}$ correspond to the visual features in $\mathbf{s}$. It is possible to consider features of different dimension d which provide constraints on the linear system and correspond to d rows of $\mathbf{L_s}$ for each feature. In this case the dimension of the interaction matrix is $\left(\sum_{i=0}^{t} n_i d_i\right) \times m$ where t is the number of different features, n_i is the number of features of type i, d_i is the dimension of feature type i and $m = 6$ for the unknown pose.

The combination of different features is therefore achieved by adding features to vector $\mathbf{s}$ and by concatenating each feature's corresponding interaction matrix into a large interaction matrix as:

$$\begin{bmatrix} \dot{\mathbf{s}}_1 \\ \vdots \\ \dot{\mathbf{s}}_n \end{bmatrix} = \begin{bmatrix} \mathbf{L}_1 \\ \vdots \\ \mathbf{L}_n \end{bmatrix} \mathbf{v} \tag{3.25}$$

where n is the number of rows $(\sum_{i=0}^{t} n_i d_i)$

Furthermore, if the number or the nature of visual features is modified over time, the interaction matrix $\mathbf{L_s}$ and the vector error Δ is easily modified consequently.

3.6 Registration and Virtual Visual Servoing

As already mentioned, the central issue of 3D tracking is pose estimation. Here a focus is made on the estimation or image registration technique. As already mentioned, Virtual Visual Servoing is a non-linear iterative minimization procedure which aims to minimize the difference between the projection of a 3D model of the environment and its corresponding features detected in the real image. This is achieved by using the task function approach, which consists of regulating an error $\mathbf{e}$ to zero:

$$\boxed{\mathbf{e} = \mathbf{C}\Delta = \mathbf{C}(\mathbf{s}({}^c\mathbf{r}_o(t)) - \mathbf{s}^*), \tag{3.26}}$$

where the difference between the task function and the image feature error given in equation (3.20) is the matrix $\mathbf{C}$, called the combination matrix, which is used to combine different numbers of visual features. $\mathbf{C}$ is a $d \times n$ matrix chosen so that $\mathbf{C}\widehat{\mathbf{L}}_\mathbf{s}$ is of full rank d. The minimum number of independent equations required for estimating the rigid set of pose parameters is $d = 6$. Therefore, there must exist a minimum of $n = 6$ linearly independent rows of $\mathbf{e}$ so as to fully constrain the 6 degrees of freedom of the pose $\mathbf{r}$. In general the combination matrix $\mathbf{C}$ is used to combine features in a redundant system so that a linear relation can be given between the set of visual features and the parameters to be estimated.

The next step is to minimize the error by taking the derivative of (3.26) and setting equal to zero. Since (3.26) is a non-linear equation in the pose parameters, it is necessary to perform a Taylor series expansion of (3.26) leading to:

$$e_i({}^c\mathbf{r}_o, \delta\mathbf{r}) = e_i({}^c\mathbf{r}_o) + \delta t_x \frac{\partial e_i}{\partial t_x} + \delta t_y \frac{\partial e_i}{\partial t_y} + \delta t_z \frac{\partial e_i}{\partial t_z} + \delta\Omega_x \frac{\partial e_i}{\partial \Omega_x} + \delta\Omega_y \frac{\partial e_i}{\partial \Omega_y} + \delta\Omega_z \frac{\partial e_i}{\partial \Omega_z} + O(|\delta\mathbf{r}|^2), \quad (3.27)$$

where e_i is a single row i of (3.26), ${}^c\mathbf{r}_o$ is the previous camera pose between the camera and the object, $O(|\delta\mathbf{r}|^2)$ are the higher order terms and $\delta\mathbf{r} = (\delta t_x, \delta t_y, \delta t_z, \delta\Omega_x, \delta\Omega_y, \delta\Omega_z)^\top$ is the difference between the initial and desired camera poses.

It is important to note that the difference $\delta\mathbf{r}$ between the initial and final poses can be achieved in two ways. A common approach is simply the direct difference between the initial and final poses:

$$\delta\mathbf{r} = {}^c\mathbf{r}_o - {}^{c^*}\mathbf{r}_o, \quad (3.28)$$

however, this quantity does not respect the distances and angles under transformations in $SE(3)$. The second approach is to determine the difference in pose between the current and desired camera positions as:

$$\delta\mathbf{r} = {}^c\mathbf{r}_{c^*} \quad (3.29)$$

where this differential pose is obtained from the homogeneous transformation of the pose using equation (2.13), on page 36, as was done for the position based visual servoing in (3.18) (i.e. ${}^c\mathbf{M}_{c^*} = {}^c\mathbf{M}_o \left({}^{c^*}\mathbf{M}_o\right)^{-1}$). This difference can be considered as equivalent to a twist ${}^c\mathbf{v}_{c^*} = \delta\mathbf{r}$ which is coherent with the Lie algebra representation of the difference between the two poses so that ${}^c\mathbf{r}_{c^*} = e^{[{}^c\mathbf{v}_{c^*}]}$ (see Chapter 2). Note that this representation would be similar in the case of an object motion model with $\delta\mathbf{r} = {}^o\mathbf{v}_{o^*}$.

If a first order approximation is made and equation (3.29) is used, this leads to:

$$e_i({}^c\mathbf{r}_o, {}^c\mathbf{r}_{c^*}) \approx e_i({}^c\mathbf{r}_o) + \nabla e_i({}^c\mathbf{r}_{c^*}){}^c\mathbf{v}_{c^*}, \quad (3.30)$$

where the second term depends on ${}^c\mathbf{r}_{c^*}$ (due to (3.29)) and the gradient is $\nabla e_i({}^c\mathbf{r}_{c^*}) = \left(\frac{\partial e_i}{\partial t_x}, \ldots, \frac{\partial e_i}{\partial \omega_z}\right)^\top$.

Written in matrix form for the entire error vector $\mathbf{e}$, (3.30) gives:

$$\mathbf{e}({}^c\mathbf{r}_o, {}^c\mathbf{r}_{c^*}) \approx \mathbf{e}({}^c\mathbf{r}_o) + \dot{\mathbf{e}}({}^c\mathbf{r}_{c^*}). \quad (3.31)$$

Using the task function (3.26) and a camera-based motion model (3.21), the derivative of $\mathbf{e}$ with respect to time is given by:

$$\begin{aligned} \dot{\mathbf{e}}({}^c\mathbf{r}_{c^*}) &= \frac{\partial\mathbf{e}}{\partial\mathbf{s}} \frac{\partial\mathbf{s}}{\partial {}^c\mathbf{r}_{c^*}} \frac{d\,{}^c\mathbf{r}_{c^*}}{dt}, \\ &= \mathbf{C}\mathbf{L}_\mathbf{s}{}^c\mathbf{v}_{c^*} + \frac{d\mathbf{C}}{dt}\left(\mathbf{s} - \mathbf{s}^*\right). \end{aligned} \quad (3.32)$$

where it can be noted that the combination matrix $\mathbf{C}$ is potentially varying with time.

In general, the assumption is made that $\mathbf{C}$ is constant so that $\frac{d\mathbf{C}}{dt}\left(\mathbf{s} - \mathbf{s}^*\right) = 0$. Equation (3.32) is therefore:

$$\dot{\mathbf{e}}({}^c\mathbf{r}_{c^*}) = \mathbf{C}\mathbf{L}_\mathbf{s}{}^c\mathbf{v}_{c^*}, \quad (3.33)$$

where $\mathbf{L}$ is the Interaction matrix of $\mathbf{s}$ given by:

$$\mathbf{L}_\mathbf{s}({}^c\mathbf{r}_{c^*}) = \begin{pmatrix} \nabla s_i^\top({}^c\mathbf{r}_{c^*}) \\ \vdots \\ \nabla s_n^\top({}^c\mathbf{r}_{c^*}) \end{pmatrix} = \begin{pmatrix} \frac{\partial s_1}{\partial t_x}({}^c\mathbf{r}_{c^*}) & \cdots & \frac{\partial s_p}{\partial \Omega_z}({}^c\mathbf{r}_{c^*}) \\ & \ddots & \\ \frac{\partial s_n}{\partial t_x}({}^c\mathbf{r}_{c^*}) & \cdots & \frac{\partial s_n}{\partial \Omega_z}({}^c\mathbf{r}_{c^*}) \end{pmatrix}. \quad (3.34)$$

Note that this matrix is called an Interaction Matrix when equation (3.29) is used and when equation (3.28) is used the matrix $\mathbf{J}({}^c\mathbf{r}_o) = \nabla\mathbf{s}$ is called the Jacobian.

In order to minimize the error $\mathbf{e}$ an iterative minimization procedure is used. An exponentially decoupled decrease of the error is defined so that:

$$\boxed{\dot{\mathbf{e}} = -\lambda\mathbf{e}, \qquad (3.35)}$$

where λ is the gain controlling the rate of decrease of the error.

Combining equations (3.33) and (3.35) and solving for the velocity twist ${}^c\mathbf{v}_{c^*}$, the following is obtained:

$${}^c\mathbf{v}_{c^*} = -\lambda(\mathbf{C}\widehat{\mathbf{L}}_\mathbf{s})^{-1}\mathbf{e}, \qquad (3.36)$$

where $\widehat{\mathbf{L}}_\mathbf{s}$ is an approximation of the "true" interaction matrix $\mathbf{L}_\mathbf{s}$.

To simplify the control law the combination matrix $\mathbf{C}$ is chosen as being the pseudo inverse of $\widehat{\mathbf{L}}_\mathbf{s}$. $\widehat{\mathbf{L}_s}^+ = (\widehat{\mathbf{L}}_\mathbf{s}^\top\widehat{\mathbf{L}}_\mathbf{s})^{-1}\widehat{\mathbf{L}}_\mathbf{s}^\mathbf{T}$ if $\widehat{\mathbf{L}}_\mathbf{s}$ is a $n \times p$ full rank matrix with $n \geq p$. In practice the pseudo inverse is computed using Singular Value Decomposition (SVD). Using equation (3.26) and (3.36) gives:

$$\boxed{{}^c\mathbf{v}_{c^*} = -\lambda\widehat{\mathbf{L}_s}^+(\mathbf{s}({}^c\mathbf{r}_o) - \mathbf{s}^*). \qquad (3.37)}$$

Once the control velocity has been obtained it is simple to obtain the corresponding camera pose by using the exponential map as given in (2.59) as:

$${}^c\mathbf{r}_{c^*} = e^{[{}^c\mathbf{v}_{c^*}]}, \qquad (3.38)$$

which give the relative pose between the initial and desired camera positions.

The pose between the camera and the object is obtained from a simple homogeneous transformation:

$${}^{c^*}\mathbf{M}_o = {}^c\mathbf{M}_{c^*}^{-1}{}^c\mathbf{M}_o, \qquad (3.39)$$

from which the final pose ${}^{c^*}\mathbf{r}_o$ is obtained iteratively.

The overall view of the algorithm is given by way of a flow chart given in Figure 3.9.

3.6.1 Convergence Issues

When $\widehat{\mathbf{L}}_\mathbf{s}$ is constant, a well known criteria to ensure the global asymptotic stability of a non-linear system is the Lyapunov positivity condition [Samson91]:

$$\boxed{\widehat{\mathbf{L}}_\mathbf{s}^+\mathbf{L}_\mathbf{s} > 0, \quad \forall t, \qquad (3.40)}$$

where t is the number of iterations of the control law.

In general it is not possible to demonstrate the global stability of a visual control law, however, it is possible to obtain the local stability for different cases of $\widehat{\mathbf{L}}_\mathbf{s}$. The different cases which may be considered

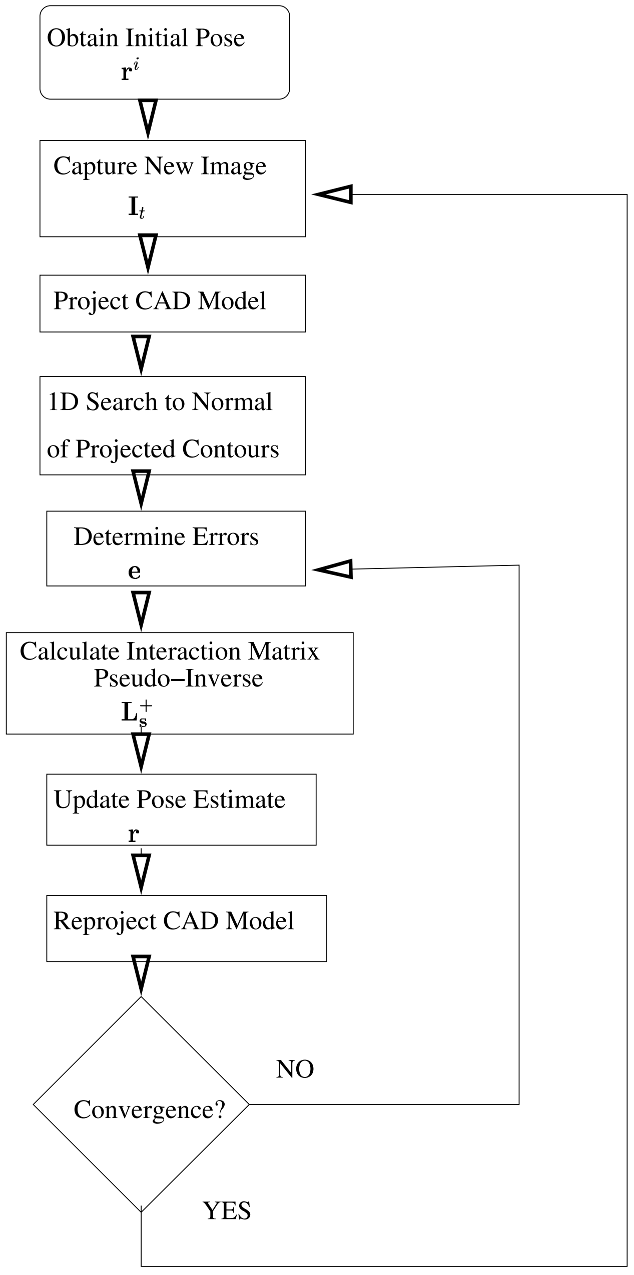


Figure 3.9: Flowchart of the 3D tracking algorithm presented in this chapter. Note that the extraction of distances-to-contours is treated later in Chapter 4 and the determination of the interaction matrix is treated in Chapter 5.

depend strongly on the application and which data is available. In [Chaumette98] three different cases were studied for a visual servoing control law. In visual servoing the objective is not to estimate unknown parameters but to control a robot to execute a desired task. In this case, it is possible to estimate the final pose before executing the task. On the other hand, visual tracking involves the estimation of these pose parameters. In a registration task, the available information includes both the desired position of the image features as well as the current position of the image features, however, only the current estimate of the depth parameter is available. Therefore, in the tracking case the following cases may be considered:

- A first feasible possibility is to estimate the interaction matrix at each iteration t. In this case, a current estimate of the features $\mathbf{s}(\mathbf{r}(t))$ and a current depth estimate $\mathbf{Z}(t)$ are made at each iteration of the control loop. Thus:
$$\widehat{\mathbf{L}}_{\mathbf{s}}^{+} = \widehat{\mathbf{L}}_{\mathbf{s}}^{+}(\mathbf{s}(\mathbf{r}(t))). \tag{3.41}$$
This choice allows the system to follow, as closely as possible, the intended behavior ($\dot{\mathbf{e}} = -\lambda\mathbf{e}$). However, it is impossible to ensure the global stability, since $\widehat{\mathbf{L}}_{\mathbf{s}}$ is not constant. Thus this case has $(\widehat{\mathbf{L}}_{\mathbf{s}})^{+}\mathbf{L}_{\mathbf{s}} = \mathbb{I}, \forall t$ but only local stability can be obtained.

- A second case may be considered where it is possible to consider using a constant Interaction Matrix by using the initial pose along with the initial or desired features. The interest in doing this is because the pseudo-inverse of the large interaction matrix does not have to be computed at each iteration leading to computational efficiency.
$$\widehat{\mathbf{L}}_{\mathbf{s}}^{+} = \mathbf{L}_{\mathbf{s}}^{+}(\mathbf{s}, \hat{\mathbf{r}}(0)), \quad \text{or} \quad \widehat{\mathbf{L}}_{\mathbf{s}}^{+} = \mathbf{L}_{\mathbf{s}}^{+}(\mathbf{s}^{*}, \mathbf{r}(0)) \tag{3.42}$$
Here condition (3.40) is only ensured in a neighborhood of the desired pose, however, this criterion is ensured because in a tracking scenario the target and camera have minimal movement between frames.

Of course, other combinations of parameters exist, however, they do not present any more advantages over and above the previously mentioned combinations. It is also necessary to ensure that $\mathbf{L}_{\mathbf{s}}$ is always of full rank (6 to estimate the pose). It should be noted here that degenerate cases are not considered, such as the duality in the image between a projected line and a point, a circle and a straight line or a cylinder and a circle.

It has been shown that only local stability can be demonstrated. This means that the convergence may not be obtained if the error $\mathbf{s}-\mathbf{s}^{*}$ is too large. However, in tracking applications $\mathbf{s}$ and $\mathbf{r}$ are obtained from the previous image, thus the motion between two successive images acquired at video rate is sufficiently small to ensure the convergence. In practice it has been observed that the convergence is obtained, in general, when the camera displacement has an orientation error less that 30^{o} on each axis. Thus, potential problems only appear for the very first image where the initial value for $\mathbf{r}$ may be too coarse.

3.7 Rigid-body Tracking Results

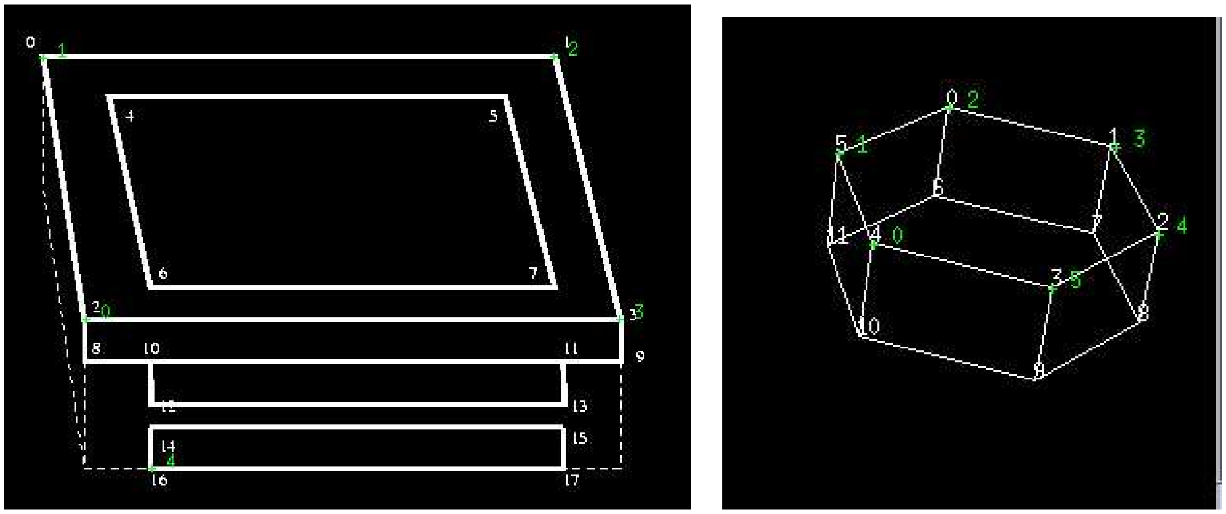
In this section, some preliminary results are given so as to give an overview of the estimation and tracking procedure. It is important to highlight that these results are based on results from the following chapters and only results related to the different estimation strategies are presented here.

The algorithm was tested on various sequences of images using a contour tracking algorithm along with the VVS control law. The tracking experiments were performed on relatively long sequences of images using a Pentium IV 1,7 GHz computer. A development environment called Augmented Reality Platform (ARP) has been created to test the algorithms. This platform is protected under the French Software protection (APP).

3.7.1 Initialization

Initialization of the pose between the camera and a target object is based on correspondences between the CAD model and the initial image in an image stream. Different manual and automatic techniques have been considered.

One manual solution which requires well structured objects is based on using the mouse to make these correspondences between the CAD model and the initial image. An image based representation of the CAD model is used for making these correspondences as shown for two objects in Figure 3.10. Depending on the composition of the various objects it is possible to make point correspondences with points defined on the object. It can be seen that these CAD models are composed of line features, however, the corners of the object are used for the initialization. More complex CAD models will be considered further in Chapter 5. Once an initial correspondence has been made an initial pose calculation was made using a four point algorithm [Dementhon95]. The difficulty of this approach is that it requires well defined points on the object. If point features are not available, such as in the case of more complex configurations including circles, infinite lines, etc, it is necessary to use a different approach or guess point correspondences.



(a) (b)

Figure 3.10: A representation of the projection of the 3D CAD model for (a) a micro-controller and (b) a screw. These models were used to manually make correspondences between the model and the initial image so as to initialize the pose estimation process.

For the reason mentioned previously, it is necessary to consider another initialization method for objects without well defined point features. In this case manual initialization is performed by projecting the CAD model onto the image with an initial guess pose and the manipulating each degree-of-freedom of its pose separately with the mouse. This has proven to be an effective way to initialize all types of objects.

Since manual initialization of correspondences is often tedious and time consuming, a third method was implemented which can automatically initialize textured objects. An example of the initialization step can be seen in Figure 3.11. This approach is based on the recent scale invariant key-point matching method (SIFT [Lowe04]) which is kindly provided online. This technique is used along with reference images of the object and a linear pose computation method to perform an automatic initialization of the pose in the first image. Unfortunately, much improvement is required to obtain a fully automatic procedure for all types of objects including those without texture.

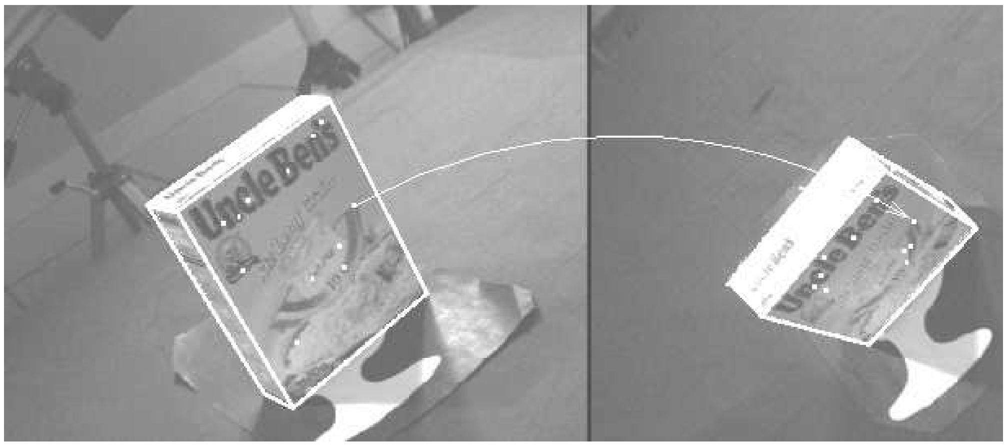


Figure 3.11: Automatic initialization of the pose in the first image for two different starting positions. Reference image on the left with matched point s indicated on each image.

3.7.2 Virtual Visual Servoing: Pose Estimation

In this section, results are presented for the Virtual Visual Servoing approach presented in [Marchand02a] which has formed the basis of the work in this thesis. These results are presented so that a comparison may be made with the 3D tracking algorithm that has been developed. Note that only the estimation techniques are compared here. In this way the basic elements of the algorithms have been made a similar as possible including the using same low level tracking and no robust estimation as will be presented in Chapter 3.

The VVS algorithm presented here can be classed as a three-step pose estimation algorithm. At the low level the algorithm is based on the extraction of low level edge elements using the Moving Edges algorithm [Bouthemy89]. This method will be described in full detail in the following chapter. In this algorithm the low level edge elements are then used in a second step to estimate global feature parameters such as lines, and ellipses. In a last step the global feature parameters are used to estimate the pose using visual servoing techniques [Chaumette90, Espiau92].

A long micro-controller sequence, as can be seen in Figure 3.12, has been used as a test case to compare the multi-step method with that proposed in this thesis. It can be noted that the major drawback of this approach was that a closed-loop control law was only carried out virtually and the feedback loop was not close visually with the image. Considering the flow diagram in Figure 3.9, the projection of the model onto the image was not performed in the outer loop at step 3. This meant that the estimated features (i.e. lines) were tracked globally and continued to track independently of their projected position according to the pose estimate. As can be seen in Figure 3.12(d), the error norm continues to increase as the global features drift from the actual position of these features in the image (or lock onto background structures). In fact, tracking effectively fails at around 130 iterations during an abrupt movement of the camera. It can also be seen in Figures 3.12(e) and (f) that the velocity as well as the pose is prone to significant jitter.

3.7.3 3D Tracking

In this section, a full 3D tracking method is presented which considers the global constraint imposed by the CAD model on the lower level features. In a first part it is interesting to compare the estimation procedure proposed in this thesis with its predecessor presented previously. The results are given in Figure 3.14 and have been performed on the same micro-controller sequence as used in Figure 3.12. Furthermore, in order to only compare the base estimation procedures it was necessary to remove unrelated advances which will be discussed in further chapters. The main objective of these results is to show that it is much more beneficial to use the re-projected model for the initialization of low level tracking. Furthermore, in the case presented here the tracking loop is closed with the image eliminating the accumulation of error.

It can be seen in these results that this 3D tracking method has an average error norm in Figure 3.13(d) of approximately 3×10^{-3}m, while the previous approach is in the order of 0.2m. Furthermore, tracking continues further and only breaks down at around 325 iterations as can be seen in Figure 3.13(e). This breakdown is expected because several object enter the image and severely occlude the micro-controller. In fact, this method is able to resist some occlusion without any robust statistical treatment, however, the occlusion becomes too severe at the end of the sequence. It can also be seen in Figure 3.13(f) that the pose estimate is much smoother up until failure. This failure will be addressed further in a Chapter 6 on robust statistics.

3.7.4 Camera and Object based motion models

In this next series of results, the 3D tracking estimation procedure used in this thesis is presented in more detail. The main analysis is based on the micro-controller scenario presented previously and following this final results are presented for two different types of objects. More results are given in the following chapters. It should, however, be noted that these results are obtained by using the full algorithm including the various contributions yet to be presented. Nevertheless, only the estimation results are analyzed here.

In Figure 3.15 both (a) the camera motion model and (b) the object motion are given. The sweeping rotational movement of the camera can be seen in 3.15(a) and the return movement of the camera along the same path can be seen to follow the forward trajectory unlike in Figures 3.14 and 3.13. In 3.15(a) the object can be seen to translate with small rotations on its axis.

It was verified experimentally that the object-based and camera-based motion models have the same effect on the pose between the camera and the object by comparing the resulting poses between the camera and the object. If an object based motion model given in (3.22) is used then repeating the derivation of the control law from (3.37) leads to:

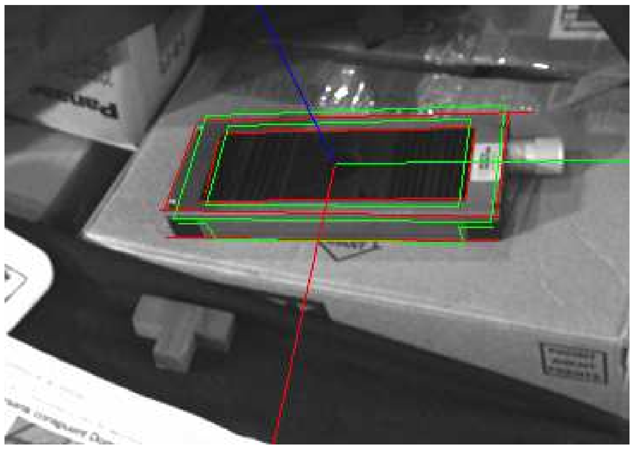
$$^{o}\mathbf{v}_{o^*} = -\lambda(\widehat{\mathbf{L}}_{\mathbf{s}}{}^{c}\mathbf{V}_{o})^{+}(\mathbf{s}(^{c}\mathbf{r}_{o}) - \mathbf{s}^*). \tag{3.43}$$

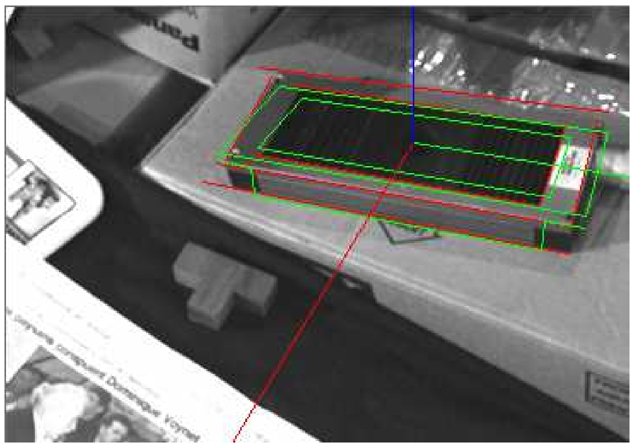
The corresponding pose is then:

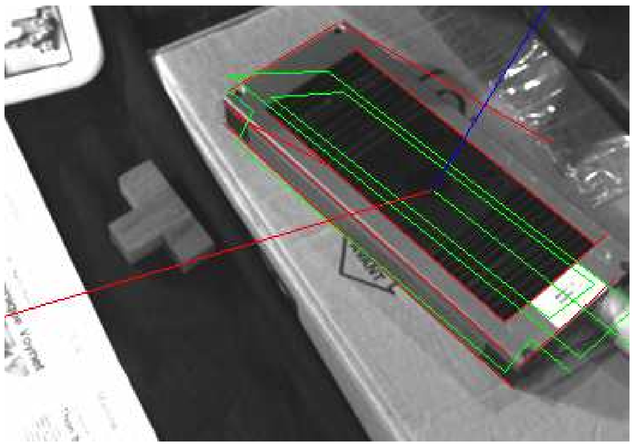
$$^{o}\mathbf{r}_{o^*} = e^{[^{o}\mathbf{v}_{o^*}]}, \tag{3.44}$$

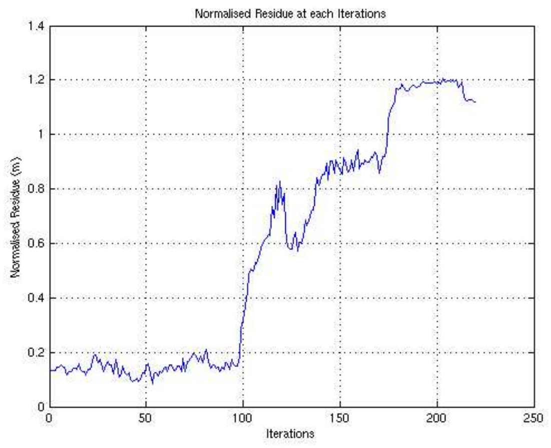
and subsequently the pose between the camera and the object is found as:

$$^{c}\mathbf{M}_{o^*} = {}^{c}\mathbf{M}_{o}{}^{o}\mathbf{M}_{o^*}. \tag{3.45}$$

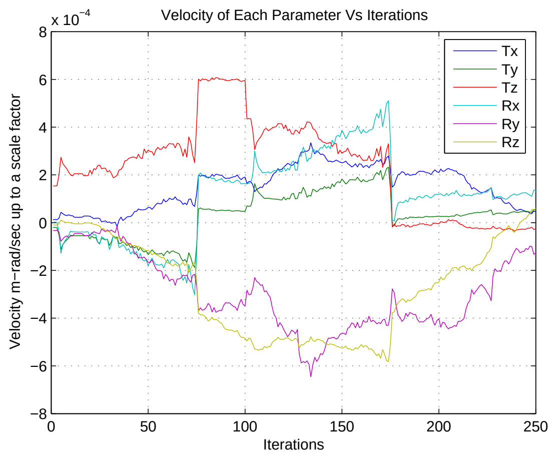

(a)


(b)

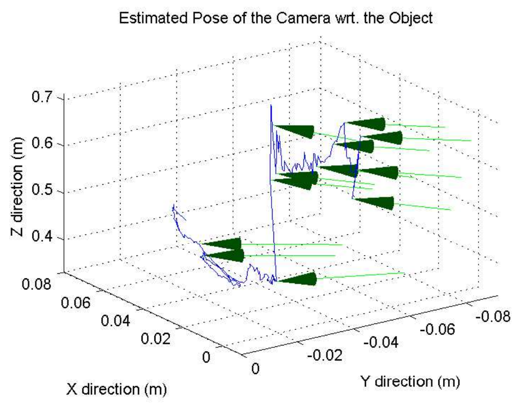

(c)



(d)

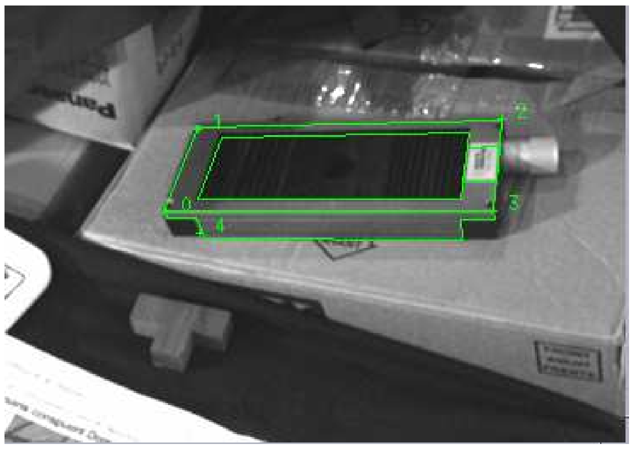


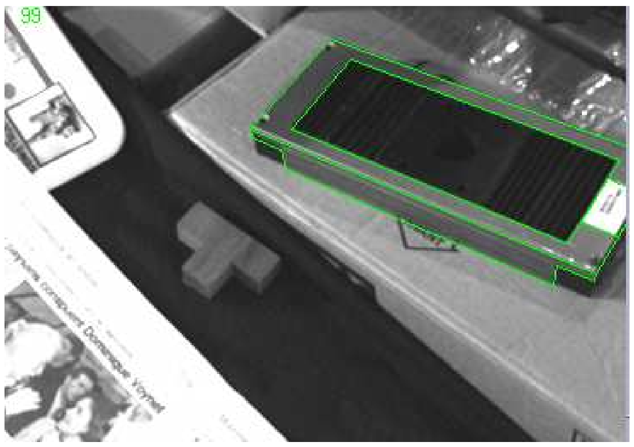
(e)

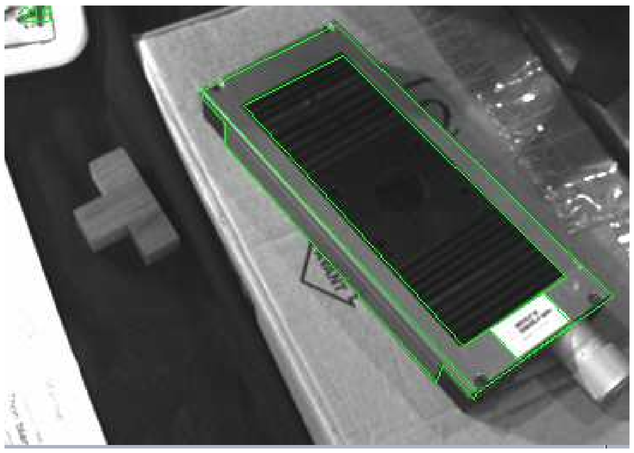


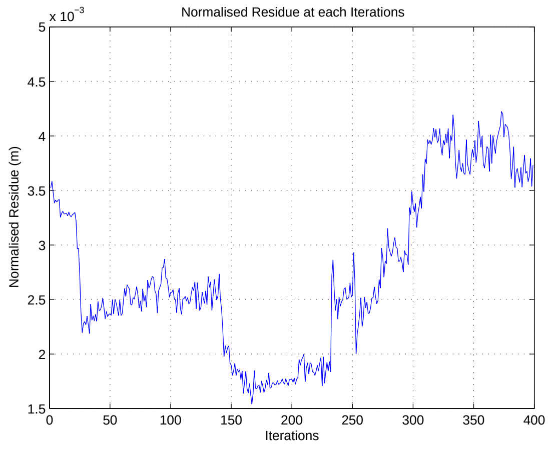
(f)

Figure 3.12: The Virtual Visual Servoing framework based on estimated lines [Marchand02a]. In the images here and in the remainder of the figures presented, the object axes are drawn in red, blue and green. The projected model using the current estimation of the pose is drawn in green while the tracked features are drawn in red. (a) the 1st image in a sequence (b) the 50th image in the sequence, (c) the 150th image in the sequence where it can be seen that the tracked edges in red have drifted away from the re-projection of CAD model leading to tracking error. It can also be noted that pose is completely lost after iteration 250. (d) the evolution of the mean square error of the feature parameters (tracking starts to fail at around 100 iterations), (e) the estimated velocity of the camera with respect to the object, (f) the evolution of the estimated pose with the camera drawn as a cone at regular intervals.

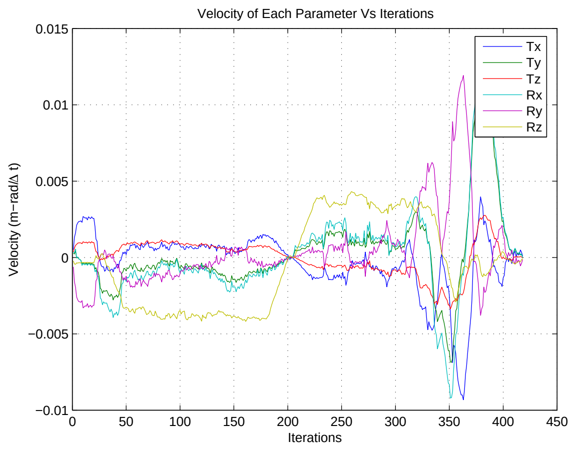

(a)


(b)

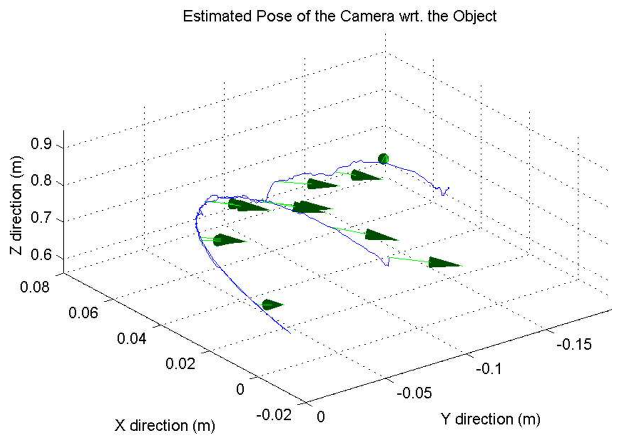

(c)



(d)



(e)



(f)

Figure 3.13: Tracking of a micro-controller using the camera-based closed loop 3D tracking method proposed in this thesis. Again the object axis is drawn and the projection of the CAD model in green can be seen to be well aligned with the object. The real movement of the camera involves a rotational movement around the object. The camera makes this rotational movement in one direction and then returns to the initial position via the same path. (a) the 1st image in a sequence (b) the 99th image in the sequence, (c) the 219th image in the sequence, (d) the evolution of the mean square error of the feature parameters (tracking continues up until around 325 iterations), (e) the estimated velocity of the camera with respect to the object, (f) the evolution of the estimated pose with the camera drawn as a cone at regular intervals. A tiny difference in trajectory can be observed between the forward and return paths.

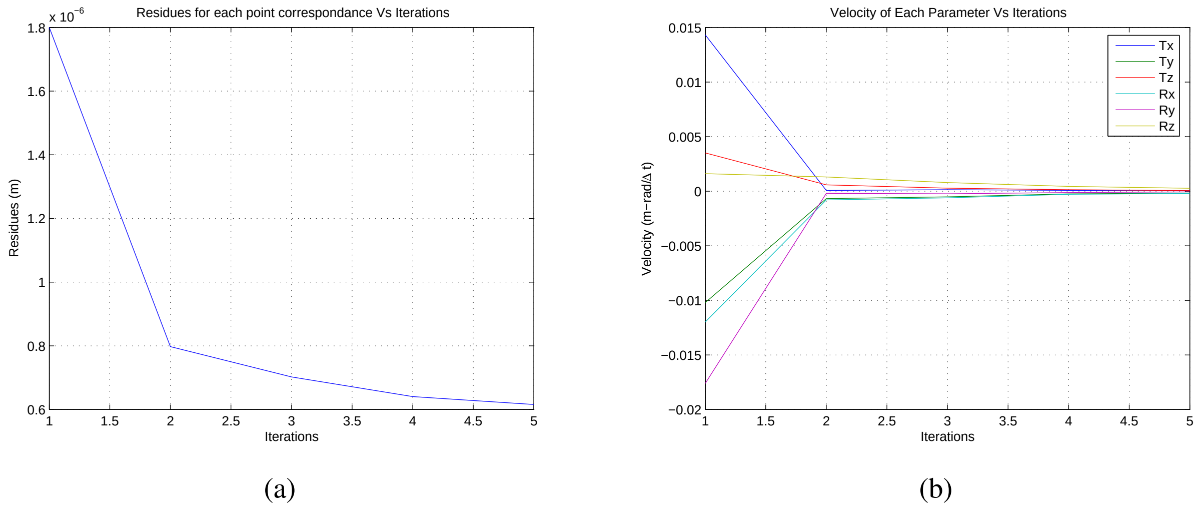


Figure 3.14: Virtual Visual Servoing iterative estimation is shown for a single image. It can be seen that 5 iterations are required to converge upon an error were the convergence test is when the average square error varies less that $1 \times 10^{-6}m$. In general very few iterations are required since the inter-frame movement is relatively close to the previous pose. (a) shows the average squared error at each iteration and (b) shows the virtual instantaneous velocity of the camera relative to the object during one estimation step. It can be seen that an exponential decrease of the error is maintained according to equation (3.35).

Since the object based and camera-based motion models yield an equivalent pose it can be said that:

$$^{c^*}\mathbf{r}_o \equiv {}^{c}\mathbf{r}_{o^*}. \tag{3.46}$$

In the velocity plots of Figures 3.15(a) and (b) it can be seen that the velocities are much more stable and smooth when compared with previous results in Figures 3.14 and 3.13. This has a direct improvement on the visual appearance of the projected objects and jitter in the image is minimal.

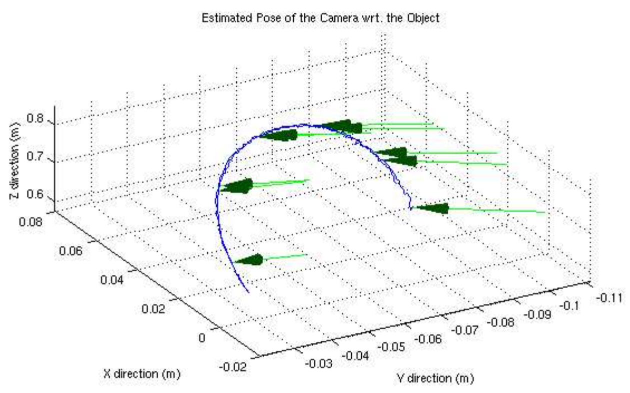
In Figure 3.16 another experiment is presented based on the tracking of a nut. This sequence is difficult due to low intensity contrast between the different faces of the object along with the presence of shadows and a total self occlusion by the original face of the nut. In order to project correctly this model it is necessary to use hidden surface removal techniques. It can be seen that a large rotational movement is made such that the side of the object is not at all within the field of view in the first image and it comes fully into view by the end of the sequence.

3.7.5 Iterative vs non-iterative minimization

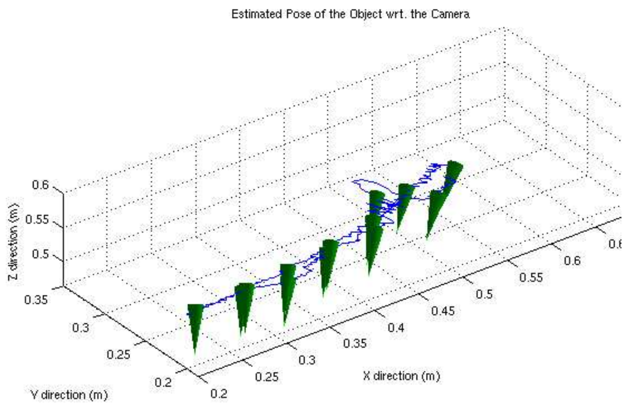
In this section, iterative minimization will be compared with a linearized non-iterative approximation. In particular the linearized method given in [Drummond02] has been implemented and compared with our iterative method using a Uncle Bens rice packet as a model.

The accuracy without iterating gives a higher normalized averaged error across the sequence than when iterations are performed (see Figure 3.18). Obviously iteratively minimizing for each frame obtains a more precise minimum of the objective function. Furthermore the larger deviation of error seen in the plot for Method 1 can also be explained by the failure of robust outlier rejection which will be studied in Chapter 6. This failure is essentially due to the dependence of the robust estimation upon iterative convergence.

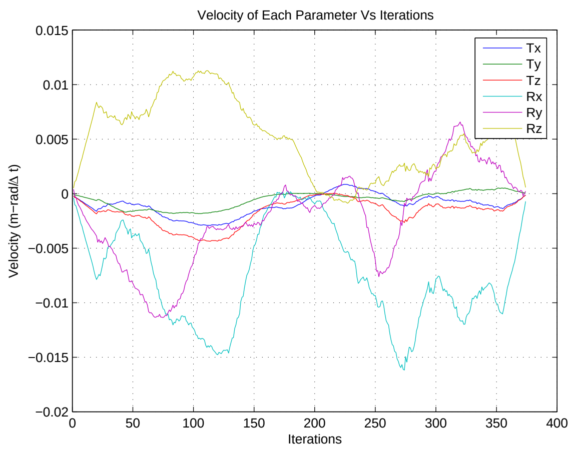
The goal of the first experiment was to determine whether it is advantageous to use an iterative (Method 2) or a non-iterative minimization scheme (Method 1). The accuracy without iterating gives an



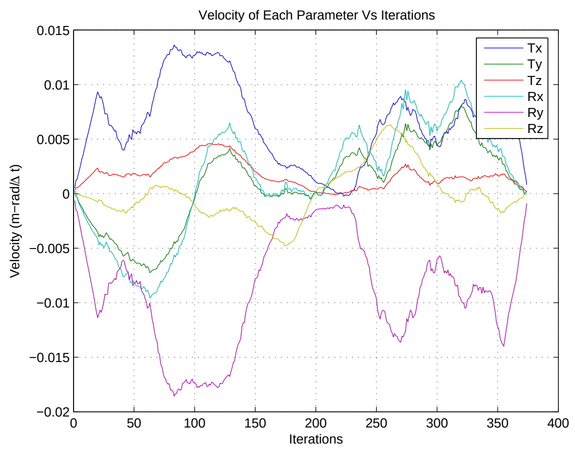
(a)



(b)

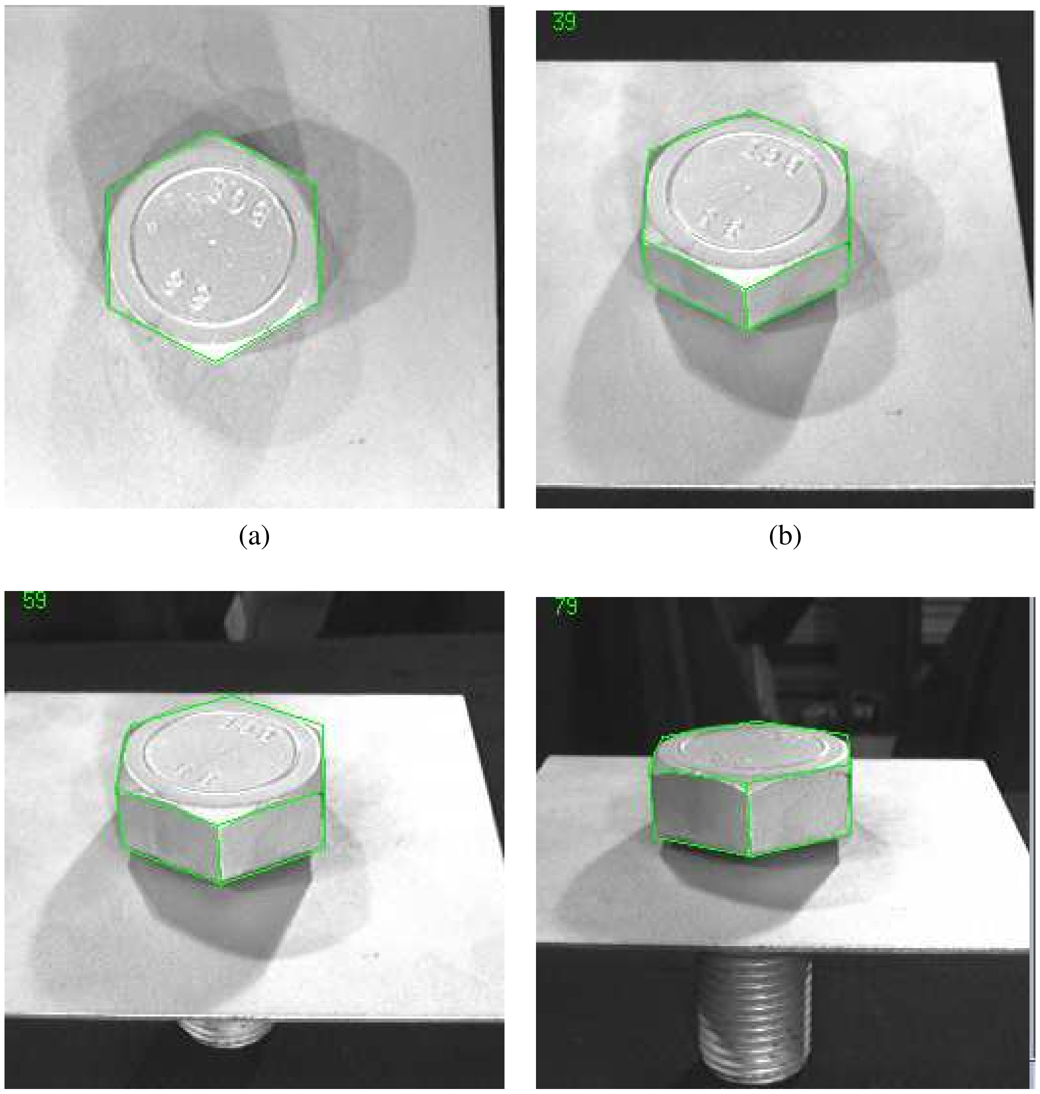


(c)



(d)

Figure 3.15: Tracking of a micro-controller using the 3D tracking methods proposed in this thesis. The real movement of the camera involves a rotational movement around the object. The camera makes this rotational movement in one direction and then returns to the initial position via the same path. Both object-based and camera-based interpretations of this motion are given. (a) The trajectory of the object relative to the camera (compare with Figure 3.13). The reference frame is drawn as a cone at regular intervals. It can be seen that tracking is maintained for the entire sequence (the end of the sequence is at around 375 iterations). It can be seen that the camera makes a rotational movement around the object and then returns to its initial position along the same path. (b) The pose of the object with respect to its last known position for the same sequence. (c) The estimated instantaneous velocity of the camera relative to the object along the entire sequence. The units are $(m/\Delta t)$, where Δt is the time taken for one frame capture which is normally around $1/30$ for a 30Hz (frames per second) camera. (d) The estimated instantaneous velocity of the object relative to itself along the entire sequence.



(a) (b) (c) (d)

Figure 3.16: Tracking of a Nut using a 3D model composed of straight lines. The projection of the model using the estimated pose is in green. (a) The initialization of the tracking in the very first image, (b) Frame 39, (c) Frame 59 and (d) frame 79. The computational performance was better that real-time due to an efficient low level tracking procedure described later in Chapter 4.

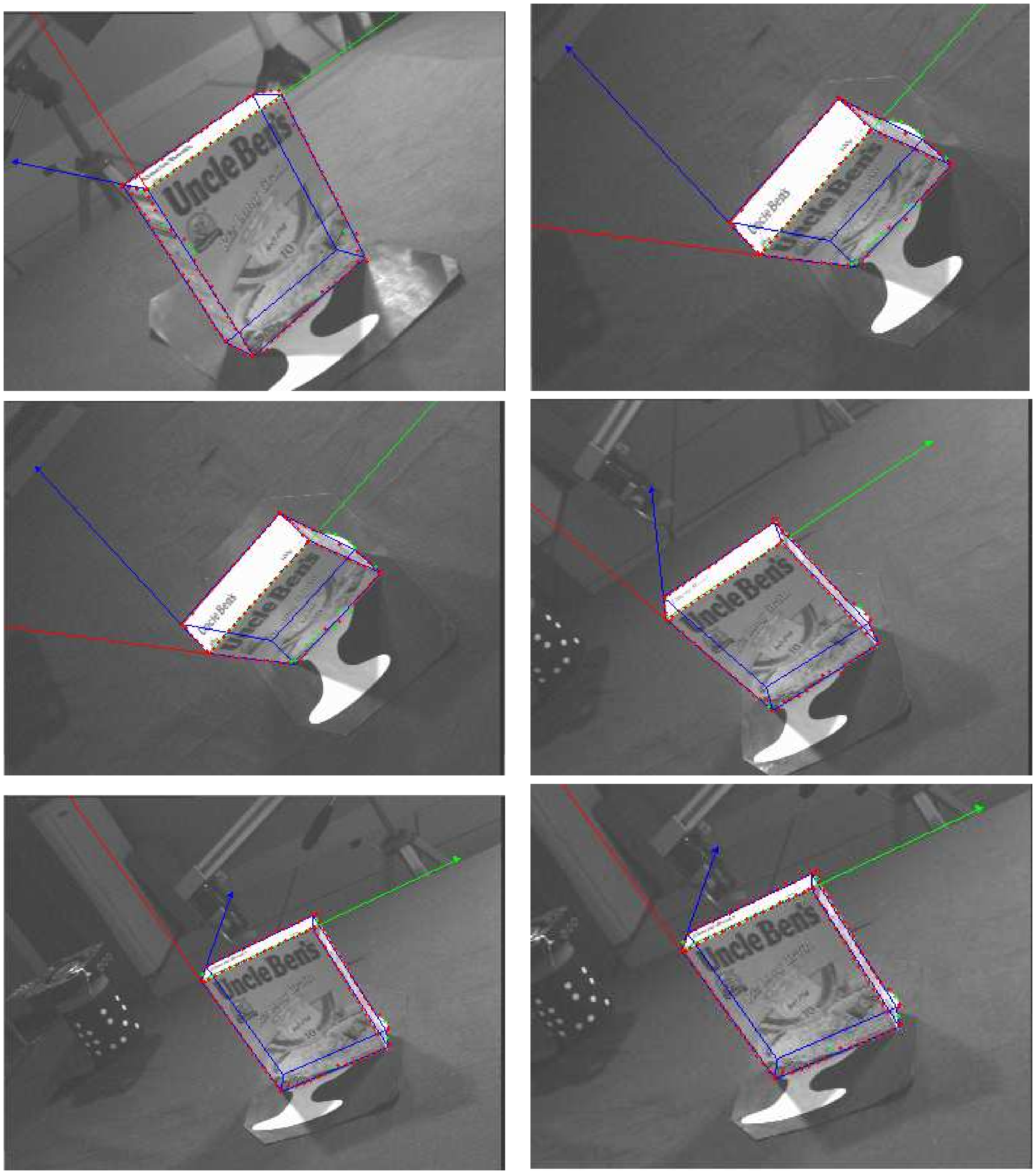
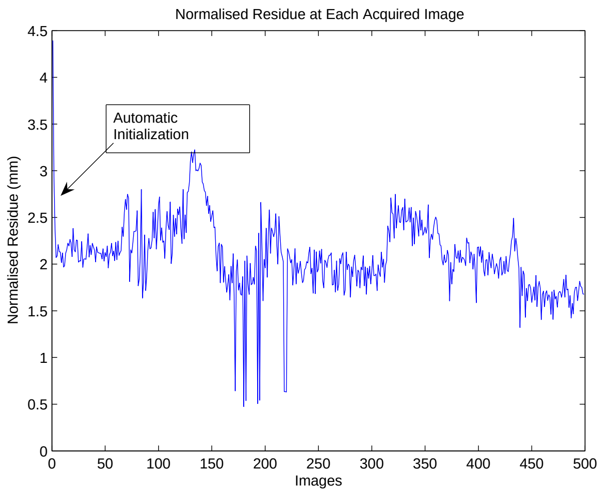
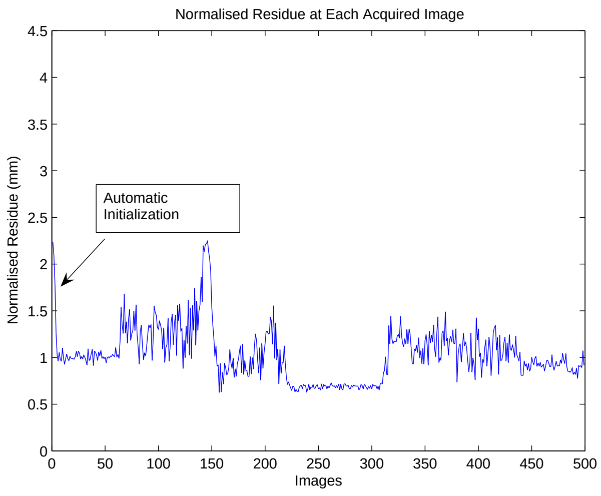


Figure 3.17: The Uncle Ben's sequence with the projected model drawn in blue and the object axes are drawn in red, green and blue. The points drawn in red are used for determining the image features and will be discussed in the following Chapter. The points drawn in green (at the bottom of the box) have been detected as erroneous. This method will be treated in Chapter 6.



(a)



(b)

Figure 3.18: Comparison of error norm over a 500 image sequence. (a) Method 1 without iterating for each image and, (b) Method 2 using an iterative scheme with a gain $\lambda = 0.7$. The iterative method provides better minimization along with a better outlier rejection.

higher normalized averaged error across the sequence (see Figure 3.18). When iterations are performed the average normalized error is smaller. Obviously iterating minimization for each frame allows to reach a better minimum of the objective function. Furthermore the larger deviation of error for Method 1 can also be explained by the failure of the M-estimator which does not obtain an accurate scale estimate (standard deviation of the inliers data) without iterating. This could be chosen manually for each sequence, however, automatic determination of scale is preferred.

3.8 Conclusion and Perspectives

The main contribution of this chapter has been to propose a tracking model analogous to both eye-in-hand and eye-to-hand image based visual servoing models. Although the different camera-based and object-based methods are essentially estimating the same unknowns, the different estimated velocities are required for different applications. Furthermore, the importance of closed loop control and iterative minimization has been shown to be important for accurate and smooth tracking results.

As will be shown later the estimated parameters may be used in both augmented reality and visual servoing applications. The object-based motion model described in this Chapter has formed the basis for tracking more complex inter-object movements. As such it will be shown in Chapter 7 that this model plays a very important role in the estimation of articulated object parameters.

In perspective, it would be interesting to consider a higher level objective function based purely on image intensity as was done in [Haag99]. In this way the 3D model could also be exploited further by considering the 3D transformation of low level image intensities and the scale of this information would be estimated simultaneously with the pose. Recent particle filtering techniques [Isard98a] have proven to be an effective way of tracking non-linear systems and maintaining multiple hypotheses. It would be interesting to consider tracking in this way with the state-space being the pose of the system.

CHAPTER 4

1D Tracking: Search with Edge Hypothesis

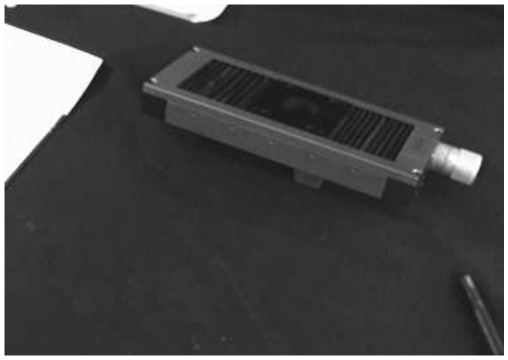
This section considers the low level foundations for position estimation from monocular vision *sensor measurements*. It will be shown that a 1D search leads to efficient real time tracking. In particular, this section will review and describe modeling of spatio-temporal edges in an image sequence and the various modeling issues that arise. This will provide the basis for the following chapter where these low level distance to contour features will be integrated into a global framework. Further chapters will continue to reference low level feature measurements including a chapter on statistically robust estimation.

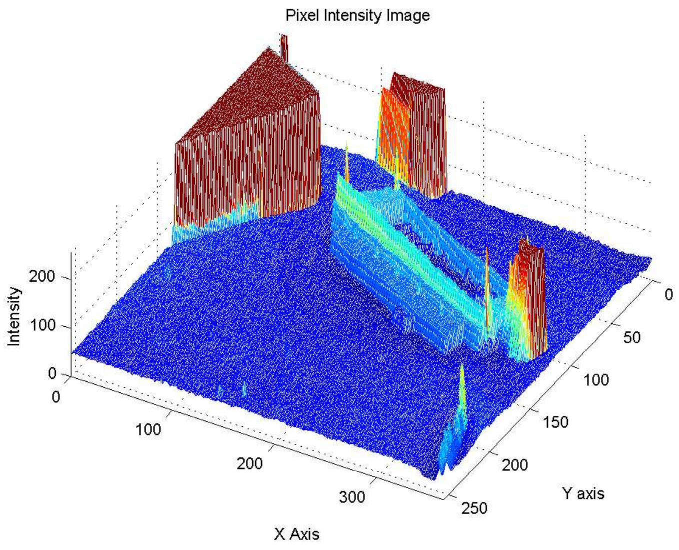
As mentioned in Chapter 3, it is important to exploit the use of high level *a priori* information so as to improve tracking techniques. In particular a 3D CAD model provides precious information about the allowable configurations of low-level information. Other than this CAD model it is also assumed that *a priori* information about the previous pose between the camera and a target object is available. As has been noted this forms the basic difference between tracking and recognition methods. This results computationally efficient searches for corresponding image elements because it is possible to search a much smaller pos configuration space between the target and sensor. It will be shown that the forward and backward flow between global and low level information is very important for obtaining performant tracking algorithms. In particular, low level methods will be considered here while keeping in mind the implications on precision, computational efficiency and robustness.

4.1 Local Movement

Modeling low level measurements provides the first step in the large majority of image processing algorithms. Indeed, since an image stream potentially contains an infinite amount of information it becomes rapidly necessary to extract important and pertinent information from the image. Furthermore, in a real-time setting it is extremely important that the extracted information holds a high amount of information while remaining compact and small in size. In terms of the human visual system, [Marr80] makes reference to early studies on the visual cortex with low level cells which are tuned to detect bars and edges of various widths and orientation.

At the lowest level of an image sensor an image intensity measurements can exist as a single grey level intensity or as a triplet of intensities representing elements of a color measurement. An example of raw


(a)



(b)

*Figure 4.1: A 3*D *plot of the intensities captured by a* 365×256 *pixel grey-scale image (a) The original image, (b) The grey level intensity plot obtained from the sensor measurements gives an intuitive view of the nature of the intensity information and the contours in an image. This scene has a homogeneous background making feature extraction simple.*

intensity information available from a single image is given in Figure 4.1. As can be seen in the 3D plot, there are many abrupt changes in intensity leading to useful information.

In the literature there exists two broad groups of methods for extracting position information from an image stream. Namely, the spatial domain in a single image and the temporal domain across subsequent images. Ideally, a feature model will incorporate both of these into a spatio-temporal model. In order to extract these types of information from the image there are another two, yet complementary techniques. More particularly, these are the techniques based on differentiation versus the techniques based on similarity. The different combinations can be summarized as follows:

- Spatial
  - Intensity differentiation between neighboring pixels.
  - Spatial auto-correlation testing.
- Temporal
  - Intensity differentiation for the same pixel between images.
  - Intensity smoothness constraints by correlation between images.

The objective of these methods is usually to match or make correspondences between images of objects present in the scene. There are various ways to estimate parameters using these general models including using an objective function and minimizing. Others include using a test criterion such as the Signal to Noise Ratio (SNR) or using a likelihood hypothesis test.

A well known spatio-temporal technique for extracting local position and motion information from the image is the Image Flow Constraint Equation (IFCE). This equation relates velocity vector v to the spatial gradient of the image intensity ∇f and its temporal derivative $\partial f / \partial t$ as follows [Horn81, Schunck86]:

$$v \cdot \nabla f + \frac{\partial f}{\partial t} = 0. \tag{4.1}$$

From this is possible to remark that only the component parallel to the intensity gradient, or that is orthogonal to the contour, is measurable:

$$v^{\perp} = \frac{-\partial f / \partial t}{||\nabla f||} \tag{4.2}$$

This is well known as the aperture problem. In Figure 4.2 a depiction of the edge normal component is given. It can be seen that if the edge moves along itself then its motion is not observable. Although this model of image flow highlights the local effect of edges, a direct implementation of this model introduces certain drawbacks. In particular it is not invariant to illumination changes such as shadows. In highly textured zones the local intensity is no longer linear while in uniform zones no information is available, it fails in occlusion situations such as a moving object passing in front of another moving object and it is restricted to local movement due to linear approximations. Theses weaknesses have lead to adding extra constraints which compensate. Approaches that work well include those that are combined with global optimization strategies such as parametric models or multi-resolution approaches. Global constraints using the pose have already been considered in Chapter 3.

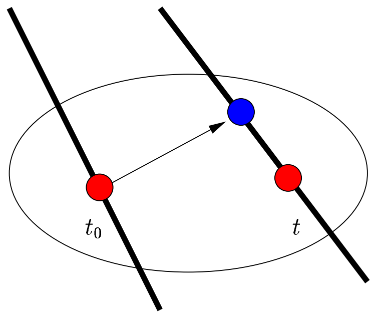


Figure 4.2: The aperture problem. Only the perpendicular motion component is observable locally on an edge and therefore only the velocity component perpendicular to the edge holds useful information.

From an information theoretic perspective on tracking, it is necessary to study visual features which hold the greatest amount of information about the spatio-temporal three-dimensional geometrical configuration of the scene while maintaining the best compression in terms of the representation of this information so as to attain optimal efficiency. This trade-off can be investigated further by analyzing the Minimum Message Length (MML) [Wallace87] or Minimum Description Length (MDL) [Rissanen87] principles concerning the model and the message related to particular visual features. Under this principle the best model is the one which allows the smallest encoding of:

1. the model, and
2. the data *given* the model.

which has been extensively applied to images in the domain of compression techniques. This notion is in accordance with the used of a global CAD model which provides an extremely compact representation of the data and the model when compared to purely local approaches.

4.2 Visual Features

Perceptually relevant information leads to the notion of feature based tracking [Hager95b]. In order to determine important and relevant features, it is also important to consider image formation. Photometry begins by the study of the light energy or intensity which originates from different light sources. Different models exist for light sources taking into account the intensity, type and direction of illumination. This light is projected onto the image plane after having been reflected off surfaces which also have their special properties. Finally, the light arrives into the camera and different models are used to model the transformation the sensor has on the signal. Refer to Figure 4.3 for a summary of the different types of visual features present in a three-dimensional scene.

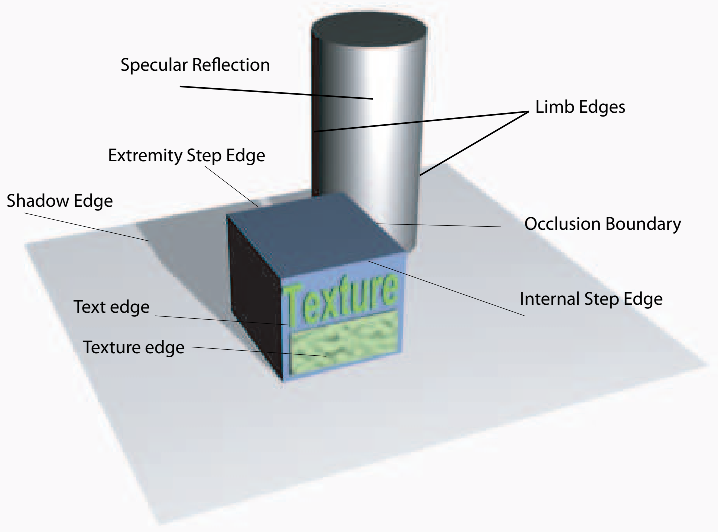


Figure 4.3: Types of visual features in a simplified model of the environment.

The problem of extracting useful information is a largely unsolved problem. This can be exemplified by the fact that in real world applications, such as Augmented Reality, many designers resort to the use of markers for robust and reliable real-time tracking. Indeed, the most common geometric features used in pose computation which are suitable for AR applications include indoor fiducial/marker based [Billinghurst01, Kato00, Marchand02a, Sundareswaran98, Zhang02] and outdoor fiducial/marker based [Neumann99] the latter shows how the size of the marker contributes to robustness and ease of use.

Although marker based methods are still used, an enormous amount of research and effort has been invested in low level feature extraction and more particularly the case of points and edges. As a first step an intensity image is usually segmented or searched for occurrences different image features. Even though it is possible to obtain scene information in varying ways such as shape from shading, shape from texture, or stereo processing, in general, regions of high variation hold more dense information than homogeneous regions, therefore leading to the notion of exploiting edges or points as visual features from the image. As already mentioned in Chapter 3, geometric visual features used for pose estimation span a very large domain. Low level visual features have also been studied in the wider literature including points and corners for which a second spatial derivative is tractable [Harris88, Tommasini98] have shown much success and have proven to be very efficient for matching and image indexing. More recently scale invariant points [Lindeberg98, Lowe04, Mikolajczyk04] have been shown to highly improve recognition accuracy and efficiency. Higher level features include moving edges based on lines and ellipses [Bouthemy89, Boukir98], circles [Hoff96], ellipses [Vincze01], contours [Blake93, Bascle94, Isard96, Blake98, Ricquebourg97, Ricquebourg00]... Less geometric methods can also be considered such as color, frequency, regions of interest [Hager98b] or principle component analysis [Murase93, Nayar94, Murase95].

All visual features can be classified to be either low level or high level according to the number of parameters needed to represent each feature. More particularly, a distance between two points is a one-dimensional feature, a point is a two-dimensional feature, an ellipse contour is a five dimensional feature and a region feature is more general and can be represented in different ways. An example region model is based on principal component analysis which is well suited to textured environments whereby the n largest components are selected. It is clear that there is a continuity of feature representations between the lowest dimension features to the highest dimension. Both low level features and high level features have certain advantages and disadvantages. In particular, low level features are generally very simple and easy to manipulate and do not over-constrain the representation of the data. On the other hand, they are not very robust for tracking nor are they very precise. On the other hand, high level features become rapidly complex, however, they are more robust due and precise due to aggregation across many low level measurements. Therefore, an ideal tracking algorithm will find a good balance between these two approaches by taking advantage of both low and high level features.

In order to obtain precise and computationally efficient features, it is important to find the smallest model which best represents the data in the most precise manner. Research in this direction has lead to energy based techniques. In [Venkatesh89, Venkatesh90], the first 2D implementation of "local energy" or phase congruency method is defined in the frequency domain to find image features. Alternatively to edge compression techniques, the phase congruency model assumes that a compressed image should be high in information or entropy and low in redundancy. Therefore, this method involves searching for ordered patterns in the phase component of the Fourier transform. In [Rosenthaler92] a similar method uses a set of oriented Gabor-based filters for determining local energy. A local energy implementation based on wavelets was given by [Xie95] and a phase congruency method based on wavelets has been given by [Kovesi96]. Unfortunately, these types of methods can become computationally intensive when compared to convolution efficient filters.

In order to obtain improvements in computational efficiency it can be shown, however, that the use of global information at the lower levels has an advantage over purely local methods. In order to highlight this efficiency gain, consider a line and a point feature. The efficiency of the line feature over a point feature can be observed in both segmentation and search algorithms. With regard to image filtering or segmentation, edges can be extracted with a 1-dimensional search to the normal of the line by using the higher level information about the orientation of the line. On the other hand, a lower level point feature requires a 2D search which is clearly less efficient. It can be noted, however, that to estimate line parameters at least two correspondences are required, however, a line feature can model the intensity values of many more pixels than a point feature for the same number of parameters. As will be shown, it is possible to extend this gain in "energy" to more complex contours to that a local 1D search strategy may be exploited.

This reasoning thus leads to the notion of an edge. Edges are used in a very large number of systems as a preliminary image processing step. The advantage of edge detection, over correlation based methods, is that it is invariant to intensity changes or changes in illumination conditions. Even though many general edge detection procedures exist, it is necessary, however, to formulate the detection methodology in the appropriate context. As proposed by [Canny83], the first step in the modeling of any edge detection method is to design a set of performance criteria.

Well known edge detection methods exist based on the first and second derivative of the intensities in the image. Some early methods such as [Roberts65] (2×2 mask) and Sobel [Sobel70] (3×3 mask) detectors used the first derivative or local gradient information but only detected edges having certain orientations and performed poorly in the presence of noisy edges. The result of the Sobel edge detector can be seen in Figure 4.4 based on the image of a micro-controller already given in 4.1. [Shen92] used a symmetrical exponential filter for edge detection. [Marr80] proposed a second derivative or Laplacian method with a broad Gaussian in order to find a balance between position and bandwidth. The Laplacian is, however, more sensitive to noise in the image.

Another type of edge detection methodology consists in modeling edge surfaces by basis functions and estimating edge parameters [Prewitt70, Haralick81, Haralick82]. An example of the Prewitt operator is given in Figure 4.4. Results of Haralick's edge detector are given in a more in depth treatment in Section 4.3. In this method image intensities are projected in the direction of the basis functions in order to decouple the surface parameters to be estimated. As will be shown this method allows more direct estimates of the edge properties such as position and orientation and can consider different surface models.

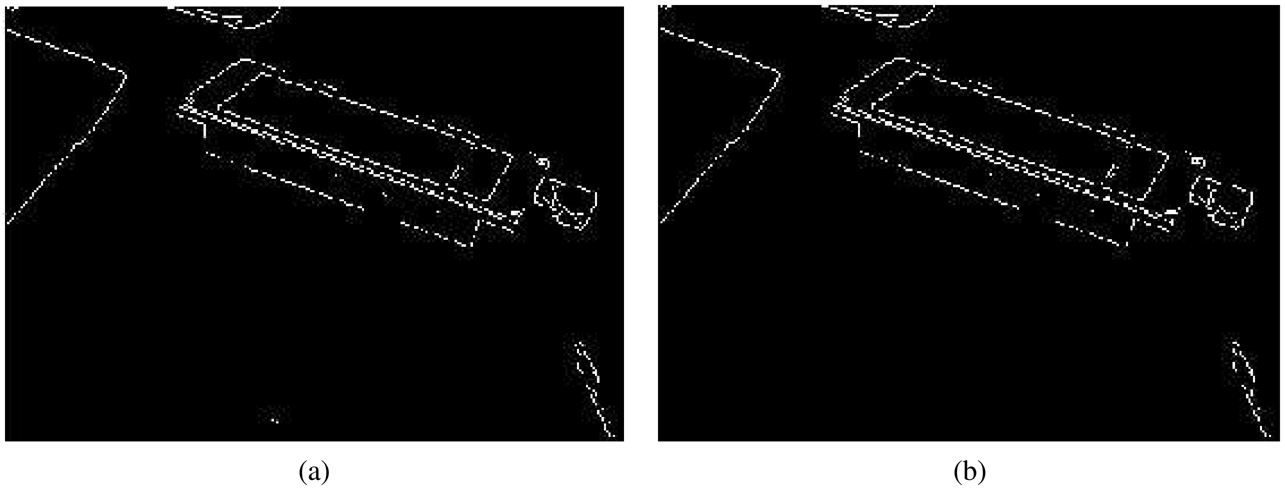


Figure 4.4: The Sobel and Prewitt edge detectors are applied to the micro-controller image. The original image of a micro-controller as seen in Figure 4.1(a) convolved with simple edge detectors (a) the Sobel edge filter (b) the Prewitt edge filter. It is clear that the information is highly compressed when compared to the pure intensity plot in Figure 4.1(b).

Since these early methods, a multitude of methods now exist which provide incremental improvement over the earlier ones. In the following sections a more in depth review will be given for three different approaches in order to highlight the main directions available. Due to the reasons given previously, the following methods will only be considered in 1D even if they are applicable in 2D. Furthermore, the approaches described here will only focus on step changes in intensity. As was shown by Canny and others, step edges are important because they typically contain sharp changes of surface normals or to the boundaries between objects in a scene. In Section 4.3, one of the models given by Haralick is described based on the zero-crossing of the second derivative. In Section 4.4, Canny's performance criteria are described for designing a filter based on the first derivative of the image intensities after Gaussian smoothing. In Section 4.5, the method used in this thesis is given. It is based on an elegant maximum likelihood formulation with an oriented spatio-temporal edge model [Bouthemy89]. As will be shown in Chapter 5, the spatio-temporal edge formulation can be easily integrated into the more global estimation scheme already described in Chapter 3.

4.3 An Edge Surface Model

In [Haralick81, Haralick82] and [Haralick92], the facet model principle was proposed whereby the image is considered as an underlying continuous grey level intensity surface and that the observed digital image is a noisy, discrete sample of a distorted version of this surface. Edge detection therefore proceeds by first estimating surface parameters. Commonly used general forms of this model are piecewise constant, piecewise linear, piecewise quadratic or piecewise cubic. Although this method also applies to gradient maxima detection, here a summary will be given for the zero-crossing of the second directional derivative of the image intensity function taken in the direction of a non-zero gradient at the pixel's center.

The particularity of this approach lies in the use of a functional form of the intensity surface consisting of a linear combination of the tensor products of discrete orthogonal polynomials of up to degree three. The directional derivatives are then easily computed from this kind of function. The underlying functions from which the directional derivatives are calculated are represented as linear combinations of the polynomials in any polynomial basis set. The discrete orthogonal polynomial basis set is chosen because it allows independent estimation of each coefficient.

4.3.1 Surface Fitting

Given a rectangular neighborhood whose row index set is R and whose column index set is C, first assume that the underlying grey level intensity function f is given by a two dimensional cubic:

$$f(i,j) = k_1 + k_2 i + k_3 j + k_4 i^2 + k_5 ij + k_6 j^2 + k_7 i^3 + k_8 i^2 j + k_9 ij^2 + k_{10} j^3, \tag{4.3}$$

where $(i,j) \in R \times C$ and k are the coefficients of the polynomial surface.

Given that the index set R is symmetrical(a window) with N elements, it is possible to construct the set of discrete orthogonal polynomials $\{P_0(i), ..., P_{N-1}(i)\}$ over R. For each index i an intensity value is observed $I(i)$ and the polynomial equation is given as:

$$I(i) = \sum_{n=0}^{N-1} a_n P_n(i), \tag{4.4}$$

where the data is fit to the function by estimating the parameters $a_0, ..., a_{N-1}$.

The exact fitting problem consists in determining all $N-1$ coefficients. The approximate fitting problem consist in estimating $K \leq N-1$ coefficients. The approximate minimization equation is therefore given as:

$$e^2 = \sum_{i \in R}^{K} \left[I(i) - \sum_{n=0}^{K} a_n P_n(i) \right]^2, \tag{4.5}$$

where a particular coefficient a_m, (4.5) is minimized by taking the partial derivative with respect to a_m and setting equal to zero.

The orthogonality property simplifies the determination of the coefficients so that by multiplying both sides of equation (4.4) by $P_m(i)$ which corresponds to a particular coefficient a_m and summing over all i can be solved to give:

$$a_m = \frac{\sum_{i \in R} P_m(i) I(i)}{\sum_{i \in R} P_m^2(i)}, \tag{4.6}$$

where the discrete orthogonal polynomials can be recursively generated by the relation given in [Haralick92] and [Forsythe57]:

$$P_{t+1}(r) = iP_t(i) - \beta_t P_{t-1}(i), \tag{4.7}$$

and where:

$$\beta_t = \frac{\sum_{i\in R} rP_t(i)P_{t-1}(i)}{\sum_{i\in R} P_{t-1}(i)^2}, \quad P_0(i) = 1 \quad \text{and} \quad P_1(i) = i. \tag{4.8}$$

Once the coefficients have been estimated the corresponding polynomial is given as:

$$\boxed{Q(i) = \sum_{n=0}^{K} a_n P_n(i)} \tag{4.9}$$

which can now be treated as a continuous real-valued function which can be derived and manipulated easily.

4.3.2 Directional Derivatives

The next step in determining an edge detector is to take the directional derivative of this function. The directional derivative at (i, j) in the direction α (clockwise from the column axis) is given as:

$$f_\alpha^{'}(i,j) = \lim_{h\to 0} \frac{f(i + h\sin\alpha, j + h\cos\alpha) - f(i,j)}{h}, \tag{4.10}$$

resulting in:

$$f_\alpha^{'}(i,j) = \frac{\partial f}{\partial i}(i,j)\sin\alpha + \frac{\partial f}{\partial j}(i,j)\cos\alpha. \tag{4.11}$$

and similarly the second directional derivative is given by:

$$f_\alpha^{''}(i,j) = \frac{\partial^2 f}{\partial i^2}(i,j)\sin^2\alpha + 2\frac{\partial^2 f}{\partial ij}(i,j)\sin\alpha\cos\alpha + \frac{\partial^2 f}{\partial j^2}(i,j)\cos^2\alpha. \tag{4.12}$$

Substituting the parametric representation $i = \rho\sin\alpha$ and $j = \rho\cos\alpha$ into the surface equation (4.3) gives:

$$\begin{aligned} f_\alpha(\rho) = \quad & k_1 + (k_2\sin\alpha + k_3\cos\alpha)\rho + (k_4\sin^2\alpha + k_5\sin\alpha\cos\alpha + k_6\cos^2\alpha)\rho^2 \\ & +(k_7\sin^3\alpha + k_8\sin^2\alpha\cos\alpha + k_9\sin\alpha\cos^2\alpha + k_{10}\cos^3\alpha)\rho^3. \end{aligned} \tag{4.13}$$

and solving gives the relation between parameter representations:

$$\sin\alpha = \frac{k_1}{(k_2^2+k_3^2)^5}, \quad \cos\alpha = \frac{k_3}{(k_2^2+k_3^2)^5}. \tag{4.14}$$

Therefore, the second directional derivative is finally obtained as:

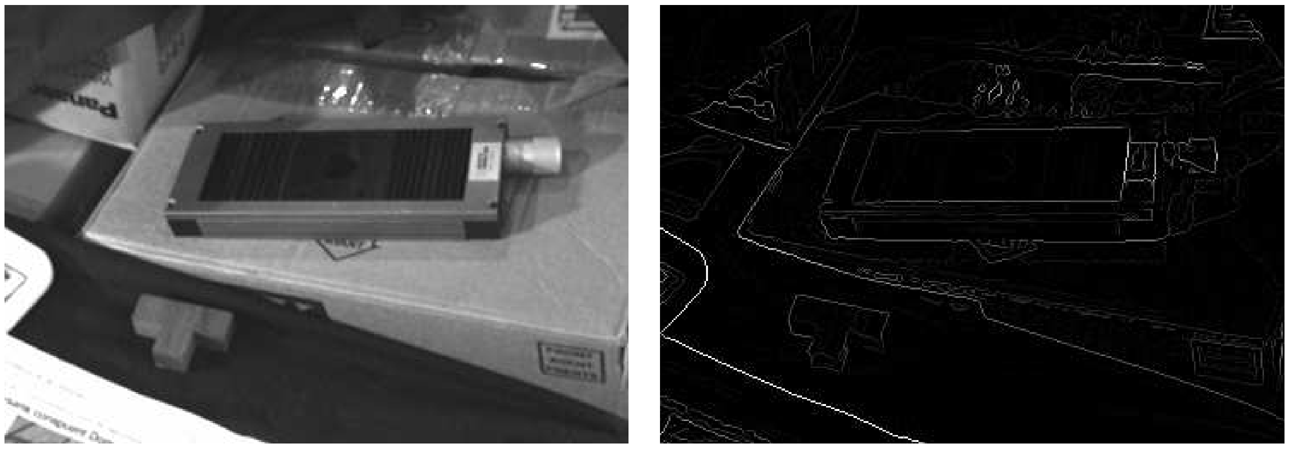
$$\begin{aligned} f_\alpha^{''}(i,j) = \quad & (6k_7\sin^2\alpha + 4k_8\sin\cos\alpha + 2k_9\cos^2\alpha)i \\ & +(6k_10\cos^2\alpha + 4k_8\sin\alpha\cos\alpha + 2k_8\sin^2\alpha)j \\ & +(2k_4\sin^2\alpha + 2k_5\sin\alpha\cos\alpha 2k_6\cos^2\alpha), \end{aligned} \tag{4.15}$$

and substituting (i, j) for (ρ, α) gives:

$$\boxed{\begin{aligned} f_\alpha^{''}(\rho) = \quad & 6[k_7\sin^3\alpha + k_8\sin^2\alpha\cos\alpha + k_9\sin\alpha\cos^2\alpha + k_{10}\cos^3\alpha]\rho \\ & +2[k_4\sin^2\alpha + k_5\sin\alpha\cos\alpha + k_6\cos^2\alpha]. \end{aligned}} \tag{4.16}$$

Given the estimated parameters for this surface, an edge is detected for a given ρ if $|\rho| < \rho_0$, $f''+\alpha(\rho) = 0$ and $f'_\alpha(\rho) \neq 0$. In other words, if there is a zero crossing of the second directional derivative in the direction of the gradient.

The results of performing this type of edge detection on the micro-controller image is shown in 4.5. It can be seen that the edges are more evenly detected in all orientations. In this formulation is it noted, interestingly, that the discrete image intensity values are used to estimate an underlying continuous valued function. Unfortunately, the method does not provide convolution efficiency and requires a mask convolution for each surface coefficient. Even so, this is the optimal number of masks required to determine the local orientation of the contour and the number of parameters could be reduced if only the perpendicular distance was estimated without any orientation information. As will be shown in Section 4.5, this amounts to estimating an edge position or spatio-temporal planar patch.



(a) (b)

Figure 4.5: (a) An image of a micro-controller in a cluttered background used in the results given in the remainder of this chapter. (b) Results of applying the zero-order crossing mask to the micro-controller image with a coefficient of 1.0.

4.4 Edge Detection Modeling Criteria

The approach proposed by [Canny83, Canny86] is based on the definition of three performance criterion. The three performance criterion are:

1. Detection - This indicates that there is a good probability that an edge is detected and a low probability that a false positive is made. Since both these probabilities are monotonically decreasing functions of the signal to noise ratio (SNR), this criterion corresponds to maximizing the SNR.

2. Localization - The detected edges should be as close as possible to the real position of the edge in the image.

3. Multiple detection criterion - Only one response to a single edge.

4.4.1 Detection Criterion

The first criterion states that if an edge exists, it must be detected with high probability and vice versa. The corresponding detection criterion is chosen as the signal to noise ratio. The impulse response of the filter is denoted $f(x)$ and the edge itself is denoted by $G(x)$. The edge is centered at $x = 0$. The filter of this edge is given by a convolution integral:

$$H_G = \int_{-W}^{+W} G(-x) f(x) dx, \tag{4.17}$$

where G denotes the response of the edge signal only and the filter is bounded between $-W$ and $+W$. The root-mean-squared response to the noise $n(x)$ is:

$$H_n = n_o \left[\int_{-W}^{+W} f^2(x) dx \right]^{1/2}, \tag{4.18}$$

where n denotes the response of the noise only and n_0^2 the mean squared noise amplitude.

The signal to noise ratio or detection criterion is therefore:

$$SNR = \frac{\left| \int_{-W}^{+W} G(-x) f(x) dx \right|}{n_o \sqrt{\int_{-W}^{+W} f^2(x) dx}}. \tag{4.19}$$

4.4.2 Localization Criterion

Criterion two states that the detected edges must be as close as possible to the real edge position. This is chosen to be the reciprocal of the RMS distance between the detected edge and the center of the true edge, determined as the reciprocal of the standard deviation of x_0. First of all, given that the SNR detects the maximum signal strength of the operator $f(x)$, the first derivative of the response will be zero at these points, thus giving:

$$H_n^{'}(x_0) + H_G^{'}(x_0) = 0, \tag{4.20}$$

where it is assumed that there is a maximum in the response at the point $x = x_0$.

The Taylor expansion of $H_G(x_0)$ about the origin gives:

$$H_G^{'}(x_0) = H_G^{'}(0) + H_G^{''}(0) x_0 + O(x_0^2). \tag{4.21}$$

Canny reasons that the response of H_G will always be symmetric no matter what type of edge $G(x)$ exists so that the odd terms of equation (4.21) are zero. Considering that $H_G^{'} = 0$ at an edge, that the higher order terms $O(x_0^2)$ are negligible and combining equations (4.20) and (4.21) gives:

$$H_G^{''}(0) x_0 \approx -H_n^{'}(x_0). \tag{4.22}$$

where the variance of the random Gaussian quantity $H_n^{'}(x_o)$ is given by its mean squared value:

$$E[H_n^{'}(x_0)^2] = n_0^2 \int_{-W}^{+W} f^{'2}(x) dx, \tag{4.23}$$

where $E[.]$ is the expectation. Combining this with (4.22) and substituting for $H_G^{''}(0)$ gives:

$$E[x_0^2] \approx \frac{n_0^2 \int_{-W}^{+W} f'^2(x)dx}{\left[\int_{-W}^{+W} G'(-x)f'(x)dx\right]^2} \approx \delta x_0^2, \tag{4.24}$$

where δx_0 is an approximate standard deviation of the position.

The criterion is therefore defined by the reciprocal of the RMS distance between the detected edge and the center of the true edge given by the reciprocal of the standard deviation of x_0:

$$\boxed{Localization = \frac{\left|\int_{-W}^{+W} f'(x)dx\right|}{n_0\sqrt{\int_{-W}^{+W} f'^2(x)dx}}. \tag{4.25}}$$

The objective of the first two criteria then becomes to maximize the product of equations (4.19) and (4.25).

4.4.3 Multiple Detection Criterion

The final criterion is designed to eliminate multiple responses. In summary this involves the determining a criterion which gives the expected number of noise maxima N_n in the region of the filter $[+W, W]$ as:

$$\boxed{N_n = \frac{2W}{x_{max}} = \frac{2}{k}, \tag{4.26}}$$

where x_{max} is the distance between adjacent maxima in the noise response of f which is determined as twice the distance between zero crossings of $f^{'}$, where the functional is given in [Rice45] as:

$$x_{max} = 2\pi \left(\frac{\int_{-W}^{+W} f'^2(x)dx}{\int^{+W} -W f''^2 dx} \right)^{1/2} = kW, \tag{4.27}$$

and where k fixes the number of noise maxima that could lead to a false response.

The original derivation of these criterion were given in [Canny83]. Given these performance criteria it is possible to determine optimal filters by numerical optimization. Canny determines the optimal filter under these criteria for a *step edge* signal. It is then shown that the optimal filter is well approximated by the first derivative of a Gaussian giving the final algorithm as:

$$\mathbf{J} = \nabla G * \mathbb{I}, \tag{4.28}$$

where $\mathbf{G}$ is a Gaussian with zero mean and standard deviation σ. It should be noted that it is necessary to choose the standard deviation of the Gaussian filter as well as an edge detection threshold. It can be noted here that criterion three is invariant to changes in scale.

4.4.4 An Optimal Mask Convolution

Thus Canny gives an analytical form of three criteria as functionals of the operator impulse response. It is also possible to enhance this detector in several ways including non-maximum suppression which thins edges to be 1-pixel wide, or using hysteresis thresholding to obtain chains of connected contours in order to avoid streaking. Combining different operators can be done in order to determine optimal mask width and edge orientation. This is achieved by selecting the mask width with the best SNR and the orientation with the highest probability.

The work by Canny reaffirms an important result obtained in [Marr80] and even in earlier papers [Bracewell65]. The notion being that the localization in the spatial domain is in conflict with the frequency domain. In the case of Canny's 1D step operator, the analytical derivation of criterion one and two can be proven to be in contradiction [Canny83], therefore obeying this uncertainty principle. In particular, if the size of the 1D mask is increased in the direction of the normal to the edge then the SNR increases, however, at the same time the localization precision or variance of the position increases. In other words, the localization precision is inversely proportional to the signal to noise ratio. Inversely, however, it can be shown that with increasing mask width, parallel to the edge, both localization and SNR increase as long as the application of the operator does not deviate from the edge.

Deriche [Deriche87] uses Canny's criteria to derive another "optimal operator" for which the filter is assumed to have infinite extent. The result of applying this edge detector to the micro-controller image is given in Figure 4.6. It can be seen that the results are similar to that of Haralick's facet detector, however, this time edge detection is performed with convolution efficiency. This improved filter is shown to be sharper than the derivative of the Gaussian and a recursive filter is proposed to improve efficiency. Shen and Castan [Shen92] describe another operator which extends that proposed by Deriche. The filter is further shown to be sharper owing to the fact that with increasing scale the central region of the edge becomes flatter, unnecessarily reducing localization precision. This is resolved by considering information close to a center "discontinuity" of the filter as being more probable than towards the edges via a weighting function.

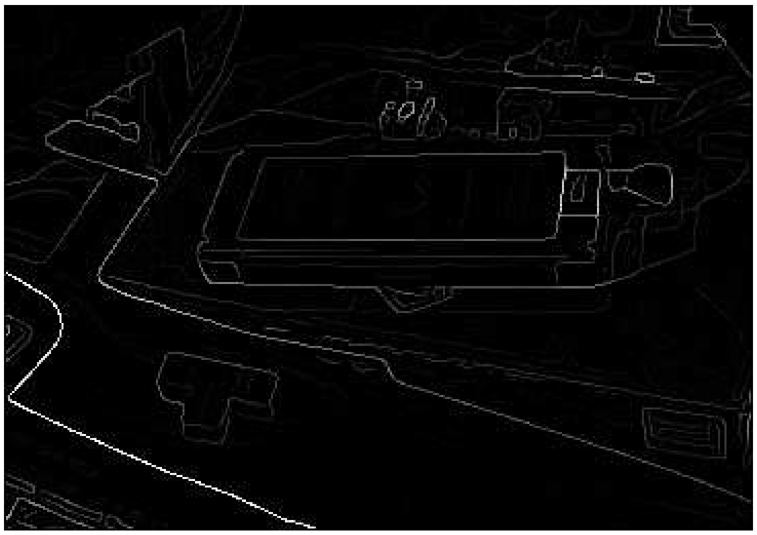


Figure 4.6: Results of applying the Canny-Deriche edge detector to the micro-controller image with $\alpha = 1$.

4.5 Spatio-Temporal Edges and Maximum Likelihood

The previous edge detection methods have been designed for pure edge detection in a static image and have not considered subsequent images in an image stream or moving edges explicitly. The third method presented here is based on the spatio-temporal Moving Edges algorithm. The Moving Edges algorithm originally developed in [Bouthemy89] is a local and efficient tracker that detects as well as determines edges in a sequence of discrete images. It is founded on a statistical maximum likelihood framework for simultaneously determining:

- position,
- spatial orientation,
- velocity component perpendicular to the edge.

The moving edge algorithm is designed so that it is not directly related to the spatio-temporal intensity changes. In [Ricquebourg97, Ricquebourg00], local smoothness constraints were added to the Moving Edges methodology using a Markov formulation so as to consider articulated movement. This implementation also considered sub-pixel precision [Ricquebourg95]. The original formulation of the moving edges is further optimized in [Boukir98] by using the contour orientation angle from the previous image rather than estimating it. Furthermore, this method implements a parallel computation procedure based on hardware and a more global least squares estimation of line parameters.

A Moving Edge (ME) is modeled as a spatio-temporal planar surface. This model follows the premise that only partial information is available from a local observation and this is referred to as the aperture problem [Marr81]. In the formulation used here, a planar patch is used (see Figure(4.7(a))), however, different parametric models as well as non-parametric based models could also be considered. An alternative model has been given in [Ricquebourg97, Ricquebourg00] for estimating the edge angle as well by using a 3D mask (see Figure(4.7(b))).

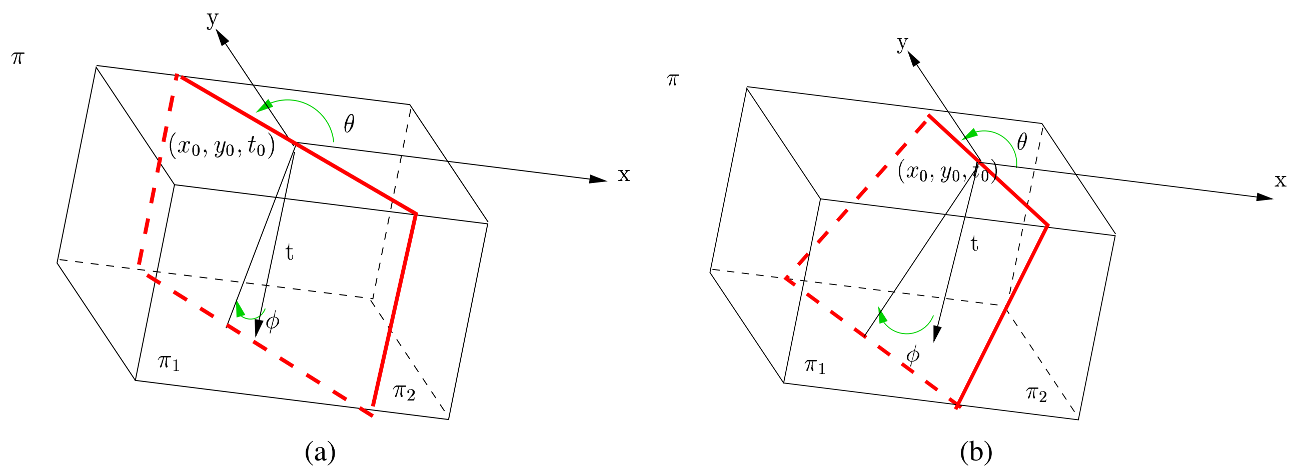


Figure 4.7: Local modeling of a moving edge where the x and y coordinates define the image plane. The t coordinate defines the temporal direction. π represents the entire volume while the volumes π_1 and π_2 represent the volumes left either side of the dividing patch. (a) depicts the case of a planar patch perpendicular to the edge in the image where the angle ϕ defines the new position of the line in a subsequent image, (b) depicts a non-planar patch with local angle θ in addition to the direction ϕ.

In this treatment, only the planar patch model given in Figure 4.7(a) is considered because this leads to a tractable solution with convolution like efficiency. The planar patch is defined within an elementary volume π in the 3D space (x, y, t). The geometric configuration of the spatio-temporal surface patch S is defined by the parameter vector ϕ.

The configuration ϕ of a planar patch can be defined by a spatio-temporal position and two angles:

- ϱ being a position within (x, y, t) space, corresponding to an edge's location in the image sequence.
- θ being the spatial angle of the edge in the image plane with respect to the x axis such that $\theta \in [0, \pi)$,
- ψ being the angle corresponding to the observed edge-perpendicular velocity with respect to the t axis such that $\psi \in (-\pi/2, \pi/2)$,

The temporal angle ψ corresponds to a perpendicular velocity component in the image given as:

$$v^{\perp} = \tan \psi. \tag{4.29}$$

As will be shown, this model approximates well a general class of edges and provides a good trade off between precision and computational efficiency.

4.5.1 Competing Hypotheses

The basic planar patch model, presented previously, is next used as a basis for a decision criterion to decide if the contour exists at a certain location in the image or not. Two competing hypotheses for detecting Moving Edges are defined as follows :

- H_0: Non-detection - the entire volume π is modeled as a constant intensity level c_0 plus a Gaussian noise with mean $\mu = 0$ and variance σ^2.
- H_1: Detection - the volume π is modeled as two sub-volumes π_1 and π_2, separated by the surface patch $S(\phi)$. Each sub-volume having a constant intensity c_1 and c_2 respectively and the noise is modeled as in H_0.

Contrary to Haralick's model, the surface patch is defined as a spatio-temporal separation and not just spatial. In the moving edge methodology it is possible to take into account piecewise linear intensity distributions, however, this is traded off for convolution efficiency. It can be seen that the spatial part of this model is so far equivalent to Canny's model, of a step edge signal with mean-zero Gaussian noise, if the planar patch is chosen. However, in addition to the spatial edge detection, a temporal component is introduced in the moving edge formalism giving added robustness.

It can also be noted here that the hypotheses are not complete if multiple edges exist within the elementary volume. In order to avoid this scenario, it is assumed that the volume is sufficiently small for this situation to be rare. Another incompleteness is the situation where the edge “disappears”or is occluded from one image to the next. This can be avoided by basic heuristics or by using robust statistics as will be shown in following chapters. The next problem is how to choose one hypothesis over another.

4.5.2 Likelihood Ratio Criterion

A likelihood ratio test is used as a decision criterion for choosing between these hypotheses. The likelihood functions are evaluated using Gaussian distributions of mean c and variance σ^2, as follows:

$$L_0 = (2\pi\sigma^2)^{-n/2} \prod_{p_i \in \pi; i=1}^{n} exp\Bigg(-\frac{1}{2\sigma^2}(f_{p_i} - c_0)^2 \Bigg), \tag{4.30}$$

where f_{p_i} are the observed intensity values at points p_i and n is the number of points.

$$L_1 = (2\pi\sigma^2)^{-(n_1+n_2)/2} \prod_{p_j \in \pi_1; j=1}^{n_1} exp\Bigg(-\frac{1}{2\sigma^2}(f_{p_j} - c_1)^2 \Bigg) \cdot \prod_{p_k \in \pi_2; k=1}^{n_2} exp\Bigg(-\frac{1}{2\sigma^2}(f_{p_k} - c_2)^2 \Bigg), \tag{4.31}$$

where f_{p_j} and f_{p_k} are the observed intensities at points in volumes π_1 and π_2 respectively.

In order to decide between hypotheses the log ratio is employed. The log ratio ξ of the likelihood functions in equations (4.30) and (4.31) is given by:

$$\boxed{\xi = ln\Big(\frac{L_1}{L_0}\Big), \tag{4.32}}$$

where this ratio depends on the parameter vector $\Theta = (\phi_1, \ldots, \phi_m, c_0, c_1, c_2)$ and where ϕ represents the geometry parameters of the surface patch and c the intensity distribution. When referring to the Figure 4.7, it can be seen that the parameters θ and ψ characterize the surface patch, however, it will be shown in Chapter 5 that θ does not need to be estimated as it is already known from the previous position of the edge. The variance parameters σ^2, at this point, are assumed constant and therefore do not appear in the parameter vector. Moreover, as the sampled volume is rather small, it is reasonable to assume that second order statistics are not applicable.

The resulting criterion for determining whether or not a local edge is present in the search area is given by:

$$max_{\phi} max_{c_1,c_2} min_{c_0} \xi(\Theta) \geq \lambda, \tag{4.33}$$

where λ is a threshold criteria which also takes into account the variance σ^2.

The likelihood ratio can be expanded as:

$$\xi(\Theta) = \frac{1}{2\sigma^2}\sum_{i=1}^{n}(f_{p_i} - c_0)^2 - \frac{1}{2\sigma^2}\sum_{j=1}^{n_1}(f_{p_j} - c_1)^2 - \frac{1}{2\sigma^2}\sum_{k=1}^{n_2}(f_{p_k} - c_2)^2, \tag{4.34}$$

where it is clear that the sum domain boundaries in terms two and three depend on the geometric parameters ϕ, while the integrand depends on the image intensity values. Thus, it is not possible to obtain a closed form solution to estimate an optimal parameter set and it is necessary to use predetermined configurations.

In order to obtain a closed form solution, a set of predetermined configurations $\{\phi_g = 1, \ldots, G\}$ are considered as already defined in Section 4.5. In summary, this involves pre-determining the likelihood masks in the grid of discrete spatial angles θ and discrete temporal angles ϕ. This determination can, however, be reduced to discrete spatial angles θ since all operators with a given spatial angle are the same along different temporal angles (i.e. along the normal of the contour at time t). Therefore, given these configurations, it is possible to determine a set of optimal parameters c_0, c_1, c_2 from the following relation:

$$\frac{\partial \xi(\phi_g, C)}{\partial c_r} = 0, r = 0, 1, 2, \tag{4.35}$$

where $C = (c_0, c_1, c_2)$.

This is evaluated as:

$$\hat{c}_0 = \tfrac{1}{n}\textstyle\sum_{i=1}^{n} f_{p_i}, \quad \hat{c}_1 = \tfrac{1}{n_1}\textstyle\sum_{j=1}^{n_1} f_{p_j}, \quad \hat{c}_1 = \tfrac{1}{n_2}\textstyle\sum_{k=1}^{n_2} f_{p_k} \tag{4.36}$$

which is simply the mean of the intensity distribution within each respective volume.

This log-likelihood ratio criterion can be compared to Canny's SNR decision criterion. In the numerator of the SNR is the convolution of a step function with a Gaussian which is equivalent to the two Gaussian hypotheses L_1 with the same mean and variances. In the denominator of the SNR is a noise term based on the Root Means Squared (RMS) error. This is comparable to the moving edge hypothesis of constant grey level with Gaussian noise L_0, however, in the latter case only the mean error is used. In this case, the main difference is that the *mean error* relates to the a-posteriori probability of a false *positive* whereas the RMS error relates to the a-posterior probability of a false *negative* [Trucco98].

4.5.3 Edge Mask Derivation

In this section, it will be explained how the log-likelihood ratio is transformed into a common mask convolution which can be convolved with the edge normal in the image as given in [Bouthemy89]. This involves re-writing the ratio given in equation 4.34 such that the mask coefficients are decoupled from the image intensity value.

In a first step, expanding equation 4.34 gives:

$$\begin{aligned}\xi(\Theta) = \tfrac{1}{2\sigma^2} \; &\Big(\textstyle\sum_{i=1}^{n} f_{p_i}^2 - 2\sum_{i=1}^{n} f_{p_i} c_0 + \sum_{i=1}^{n} c_0^2 \\ &- \textstyle\sum_{j=1}^{n_1} f_{p_j}^2 - 2\sum_{j=1}^{n_1} f_{p_j} c_1 + \sum_{j=1}^{n_1} c_1^2 \\ &- \textstyle\sum_{k=1}^{n_2} f_{p_k}^2 + 2\sum_{k=1}^{n_2} f_{p_k} c_2 - \sum_{k=1}^{n_2} c_2^2 \Big),\end{aligned} \tag{4.37}$$

where this can be simplified by using the following observation:

$$\sum_{i=1}^{n} f_{p_i}^2 = \sum_{j=1}^{n_1} f_{p_j}^2 + \sum_{k=1}^{n_2} f_{p_k}^2, \tag{4.38}$$

Considering the simplified version, equation (4.36) can be substituted into (4.37) to give:

$$\xi(\Theta) = \frac{1}{2\sigma^2}\left(n_1\hat{c}_1^2 + n_2\hat{c}_2^2 - n\hat{c}_0^2\right). \tag{4.39}$$

Re-writing observation (4.38) using equation (4.36) as:

$$nc_0 = n_1c_1 + n_2c_2, \tag{4.40}$$

allows the mean c_0, corresponding to hypothesis H_0, to be eliminated from equation (4.39). After simple algebraic development the coefficients and the intensity dependant means can be decoupled to give:

$$\xi(\phi_g) = \frac{n_1 n_2}{2(n_1+n_2)\sigma^2}(\hat{c}_1 - \hat{c}_2)^2 \tag{4.41}$$

Equation (4.41) can then be applied as a mask related directly to the intensity values by modifying the likelihood function to be the square root of ξ:

$$\zeta(\phi_g) = \left| \sum_{\eta \in M} a_\eta^g f_{\varrho+\eta} \right| \tag{4.42}$$

where $f_{\varrho+\eta}$ is the intensity observed at each position $\varrho + \eta$ in M, where M is the set of points in the volume π centered on the test position ϱ along the edge normal.

From (4.36) and (4.41) the coefficients are determined as:

$$\boxed{\begin{array}{ll} a_\eta^g = \frac{1}{\sigma}\left(\frac{n_2}{2n_1(n_1+n_2)}\right)^{1/2}, & if\ \ \eta \in \pi_1 \\ a_\eta^g = -\frac{1}{\sigma}\left(\frac{n_1}{2n_2(n_1+n_2)}\right)^{1/2}, & if\ \ \eta \in \pi_2 \end{array}} \tag{4.43}$$

so that a pair of masks can be pre-determined for each geometric configuration g of the planar patch.

From this it is possible to take into consideration two or more images in a sequence such that:

$$\zeta(\phi_g) = \left| \sum_{t=t_1}^{t_2} \sum_{\eta \in M_t} a_\eta^g f_{\varrho+\eta} \right| \tag{4.44}$$

where the sets M_t correspond to a volume which incorporates all previous volumes up until time t.

The test criterion then becomes:

$$\boxed{\max_{\phi_g} \zeta(\phi_g) \geq \lambda,} \tag{4.45}$$

where the standard deviation is often considered to be merged into the test threshold λ.

A further refinement is taken into account which takes into consideration the grid structure of the image (see Figure 4.8). A weighting factor is applied to the mask coefficients so that the area of a pixel belonging to the volume π_1 or π_2 is considered. Therefore equation (4.43) becomes:

$$\alpha_\eta^{rq} = \nu_t \cdot a_\eta^g, \tag{4.46}$$

where rq represent a given configuration of θ, ψ and ν_t is a weighting factor given by:

$$\nu_t = \frac{|\epsilon_1 - \epsilon_2|}{\epsilon_1 + \epsilon_2}, \tag{4.47}$$

where ϵ_1 and ϵ_2 denote the area of a pixel belonging to volume π_1 or π_2 respectively.

In the original formulation [Bouthemy89], both the spatial orientation θ as well as the position of the edge to the normal ϕ were sought. This required applying a set of masks at each pixel location corresponding to a set of potential edge angles θ. In this case, to alleviate problems of computational costs, as well as to take into account the fact that the angles must correspond to discrete image positions, only 4 or 6

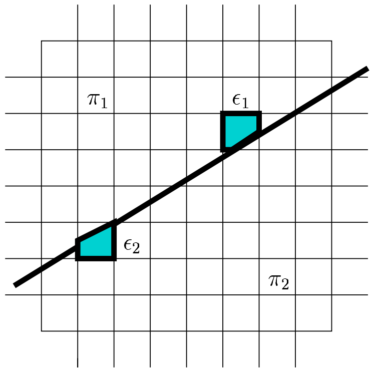


Figure 4.8: The weighting of mask values to compensate for the discrete geometry of pixels. π_1 and π_2 represent the part of the image which corresponds to the two volumes in the spatio-temporal edge model. ϵ_1 and ϵ_2 correspond to the area of each half of a pixel that is cut in two by a line.

angles were considered (i.e. $\theta \in \{0^o, 45^o, 90^o, 135^o\}$ or $\theta \in \{0^o, 30^o, 60^o, 90^o, 120^o, 150^o\}$). In a later implementation [Boukir98], global line parameters were estimated and it was possible to consider using the same angle θ as the line from the previous image. In this case a 1D search was made along the normal to the line, however, the normal directions were chosen to be discrete angles $\delta \in \{0^o, 45^o, 90^o, 135^o\}$ closest to the angle θ meaning that any gain in precision was lost.

Similarly, in the implementation considered in this thesis, the spatial angle θ is assumed to be locally the same as the angle of the projected contour using the previously estimated pose. However, in this thesis the search angle is also considered to be much finer accuracy $\delta = 1^o$ and the nearest pixel location corresponding to positions along the normal are considered (see Figure 4.9). Points corresponding to new edge positions are thus found by searching to the normal of the contour in subsequent images and applying a mask which corresponds to the configuration of the planar patch. This is achieved without any extra computational cost since the edge masks corresponding to different angles are calculated off-line and indexed in a look-up table online. Furthermore, the closest pixel location along the true search direction is easy to compute. These precisions greatly improve the accuracy and the reliability of the normal displacement.

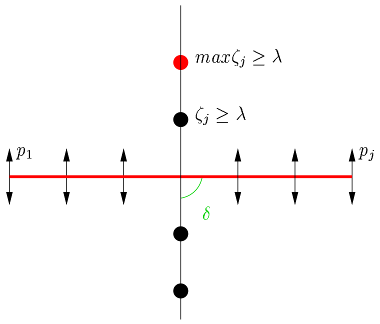


Figure 4.9: Search for a corresponding point in the current image using the moving edge algorithm. The points p_1 to p_j are sampled along the line in 2D and a search is made along the normal (of angle δ) at discret pixel positions. A 2D mask is applied at each of these locations and the likelihood of an edge being present is obtained up to a scale factor.

4.6 Automatic Scale Selection and Uncertainty Propagation

In order to model the noise in the intensity measurements a Gaussian distribution has been used in equations (4.30) and (4.31). This distribution depends on an unknown value describing the variation σ^2 of the data. The *scale* is defined here by the standard deviation σ of this intensity distribution. Even though the previously described method merges scale into the threshold criterion, *local* edge information is strongly dependent on this scale. Indeed, some level of image smoothing is needed so as to obtain good localization properties as mentioned by Canny in the design of his detection and localization criteria. A precise value of scale is particularly important when the magnitude of the signal is close to that of the noise. However, many parts of an image differ and have varying levels of noise and types of edges. If the assumption is made that the scale is constant everywhere, this can lead to failure. This section presents a method for automatic selection of the scale for the detection of the low level 1-dimensional features described previously.

Previously, much research has been invested into finding a solution to the problem of scale. The coordinate system related to scale is known in the literature as scale space [Witkin83]. It was proven in [Babaud86] that when 1D signals are smoothed with a Gaussian filter, it is the only filter for which the zero-crossings of the second derivatives disappear when moving from fine to coarse scale but new ones are never created, thus making it possible to track zero-crossings across a range of scales. More importantly, this unique property means that no image features are created and image integrity and information is maintained under filtering operations. In [Schunck87], an algorithm based on Canny's edge detector is introduced for detection across multiple scales. Edges are first detected at multiple scales. The edges obtained with a larger scale filter then correspond to major edges while edges obtained with decreasing scale contain an increasing number of edges. Edge strengths are then analyzed by their cumulative detection likelihood across all scales. [Bergholm87] proposes the notion of edge focusing whereby edge detection is performed using adaptive thresholding. This is aimed at counter-balancing the blurring effect of Gaussian smoothing by tracking an edge across scales, thus, finding a good balance between detection and localization. In Figure 4.10, the effect of changing the scale can be observed. The results have been obtained from the micro-controller image shown in Figure 4.5(a).

Since the early research on automatic scale estimation many solutions have been proposed. More recently, Lindeberg [Lindeberg98] proposed a framework based on two edge strength criteria:

$$\mathcal{G}_{\gamma-norm}L = \sigma^2(L_x^2 + L_y^2), \tag{4.48}$$

$$\mathcal{T}_{\gamma-norm}L = \sigma^3(L_x^3 L_{xxx} + L_y^3 L_{yyy} + 3L_x^2 L_y L_{xxy} + 3L_x L_y^2 L_{xyy}, \tag{4.49}$$

where the scale space representation L is obtained from a Gaussian convolution of variance σ with the image, and γ is used to tune the selection process by varying the diffusion of the edge. Equation (4.48) represents the gradient magnitude while equation (4.49) is related to the sign of an edge. This definition of an edge combines sign, magnitude, scale and γ. Scale is then determined by maximizing with respect to scale. The downside of this method is that it is necessary to choose discrete scales and is computationally inefficient due to the need to compute higher order derivatives.

In [Elder98], the emphasis is made on edge diffuseness. This approach is local and it is argued that it is not possible to determine the importance of edges at a low level. A local scale control algorithm is proposed which detects all edges over a broad range of contrast and blur. The algorithm requires no parameters except for the second moment of the sensor noise. The idea is based on selecting the minimum reliable scale at which the possibility of detecting edges is below a specified tolerance. The blur information has been shown to be related to the depth map.

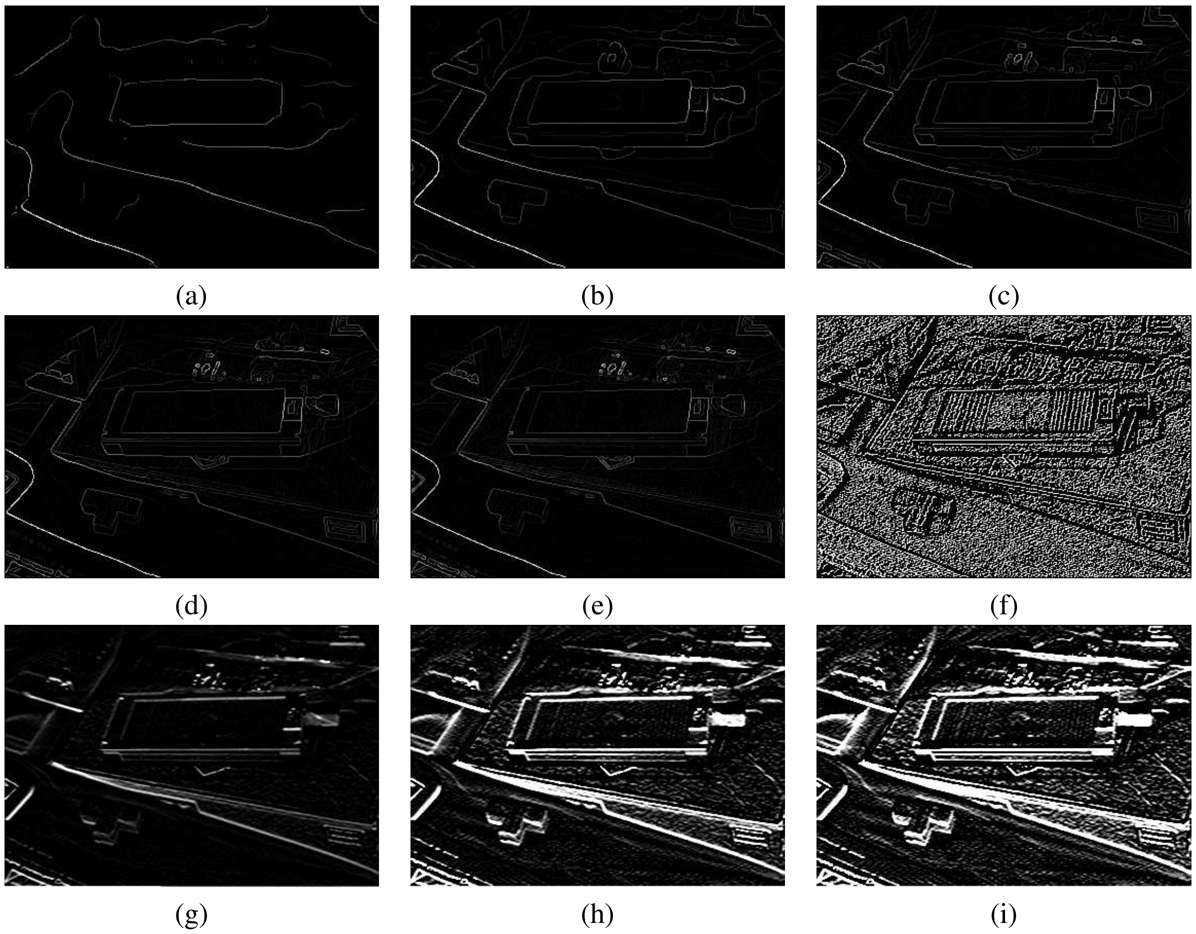


Figure 4.10: The output of an a edge filter convolved with the micro-controller image in a cluttered scene. The first row is based on the Deriche-Canny edge filter with scale values of $\alpha = 0.1, 0.5$ and 1 respectively for (a,b) and (c). The second row is based on the basic Canny gradient filter with scale values of $\sigma = 1, 0.5$ and 0.1 respectively for (d,e) and (f). The third row is based on the oriented moving edge detector applied to a single image. The edge is orientated horizontally and various scale values of $\sigma = 1, 0.5$ and 0.1 respectively. This edge tracker is, however, designed to be applied between two images and only the spatial component shown here. The effect of an oriented detector can be seen and it can be noted that no thinning has been performed on the edges. Overall it can be noted that various contours that make up the micro-controller have different scales. These differences in scale are essentially due to internal object edges vs the external contour of the object. The code used to provide the results for the Deriche-Canny edge detector is reported in [Monga91].

Another recent approach involves making use of the heat equation and the convolution of a Gaussian filter with the image. The scale is considered as the time component in the traditional heat equation. The linear heat equation eliminates noise via diffusion, however, it also blurs the edges isotropically. [Perona90] propose a solution to this problem by using a multi-scale image representation based on anisotropic diffusion. This is achieved by making the diffusion coefficient in the heat equation a function of space and scale. In other words the diffusion coefficient is made to vary with the image gradient leading to spatially adaptive smoothing which tends to preserve the location of edges across scale. The papers mentioned here as well as [Basu02] provide a more in depth survey of these different techniques.

The method proposed in this section is a simpler method which aims to take advantage of higher level estimation introduced in the following chapters. It can be considered as similar to [Kollnig95] where a global CAD model is estimated directly from image intensity values avoiding the need for edge detection and threshold selection. In this case and the case presented in Chapter 3, a known CAD model is available with a known object size and shape. The method proposed here, therefore involves propagating the low level uncertainty and scale to the higher level estimation process. At the higher level, robust decision methods are then used to locally adapt the scale and uncertainty of the features simultaneously according to their agreement with the global model.

The local Moving Edges method described previously determines points along the normal of a contour using a maximum likelihood approach. The decision as to whether or not a spatio-temporal edge exists is made by thresholding the local likelihood value. ζ_{j^*} is chosen to be an edge providing that it is greater than a threshold λ. This threshold is usually chosen manually and it depends on both:

- the likelihood of a particular spatio-temporal configuration denoted γ,
- the scale σ of the Gaussian which is related to the size of the mask being applied.

Therefore equation (4.45) can be rewritten as:

$$max_{\phi_g}\zeta(\phi_g) \geq \gamma\sigma, \tag{4.50}$$

Since the scale and the threshold are *a priori* unknown and need to be chosen manually for different situations, it is preferable to have an automatic method for estimating this. A method is presented here to propagate the local likelihood and scale of the points to the global likelihood of the pose. The global uncertainty will be introduced in chapter 6.

This involves directly relating the global measure of uncertainty to each local measure of scale and uncertainty $\zeta_j(\phi_g, \sigma, \gamma)$. For the moment consider the global uncertainty and the local uncertainty as a weighting factor. Let the local weighting factor be defined as $w_{\zeta_j} \in [0, 1]$, corresponding to each test site j. The local weighting factor is defined by a normalized value of uncertainty:

$$\boxed{w_{\zeta_j} = \frac{max_{\phi_g}\zeta_{\phi_g}}{max_j\zeta_j}, \qquad (4.51)}$$

where ζ_{ϕ_g} is a local maximum likelihood at a particular test site to the normal of the contour and $max_j\zeta_j$ is the maximum likelihood amongst all the edge points detected. Note also that it is still necessary to verify that the maximum likelihood is not close to zero.

This has the effect of giving the most certainty to strong contours in terms of the local likelihood. When combined with the global estimation of uncertainty, the statistical technique employed effectively decides upon acceptance or rejection of a particular hypothesis. In other words, the global estimation decides upon which correspondences should be kept instead of a manually chosen threshold. This is advantageous when different scenes are considered along with different size masks or scenes with edges of different scales. In fact, the global estimation can be shown to estimate the scale of the contour via robust scale estimation, while the decision threshold remains constant. If this is the case the weighting factor can be considered as a normalized estimate of the scale.

Several methods in the literature such as [Drummond02] suggest choosing the closest likelihood in the search path that is above a certain threshold. This has the advantage of being more efficient because it is not necessary to consider the remaining sites once an edge has been detected. The disadvantage is that a manually chosen threshold is needed. In this case the automatic scale estimation and uncertainty propagation cannot be used. If multiple edges exist within the search path this method may fail. In particular, it will fail if the wrong edge is closer that the actual edge which is often the case when and edge is appearing from a self-occlusion. Another possibility is to choose the closest maxima in the search path. In this case no threshold is needed but the multiple edge problem remains. This method is also extremely sensitive to noise. Thus, if the SNR ratio is small this leads to imprecise localization. This type of method would be better adapted to the zero-crossing of the Laplacian, however, this is also more sensitive to noise than the gradient. A third method involves choosing the maximum likelihood in the search path. This has the advantage of not needing a manually chosen threshold. Furthermore, this works well if the contours of the object are well defined, however, if there are multiple edges in the search path the strongest edge will be chosen. In practice, this method is used, however, ideally it would be interesting to consider multiple hypotheses at the global estimation level as has been considered in [Vacchetti04a].

4.7 Results

In this section, several basic results are presented. The technical implementation of the Moving Edge method and experiments carried out in different scenarios have allowed to refine the algorithm. In particular, several subtle differences will be made showing that the algorithm needs to be adapted to the problem.

4.7.1 Oriented Edges

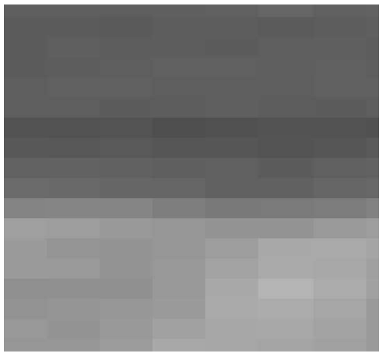
In figure 4.11(a) and (b), a close up view of two typical edges is given. It can be seen that this type of edge closely fits the planar patch model whereby two regions of constant intensity exist either side of the edge. When the Moving edge mask is applied to these edges with the same orientation (see Figure 4.11(c) and (d)) a clear response is given and a single maximum exists in the search path corresponding to the edge. Indeed the orientation of the filter has a large impact on the robustness of the results. In Figure 4.12 masks with different orientations have been applied to the micro-controller image. In this case it can be seen that badly oriented edges are easily avoided.

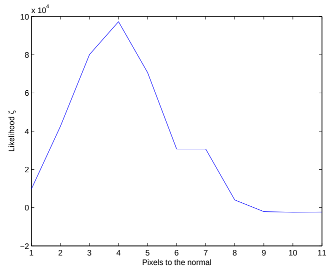
4.7.2 Search to the Normal

The Moving Edge detector is applied between two subsequent images of a sequence. In Figure 4.13 several objects with known CAD models are considered. The object's projected contour is shown in blue. A search is performed to the normal of the contour in the image, indicated in red.

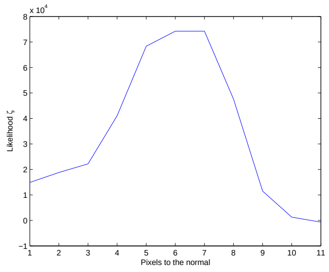
While this is beneficial for obtaining 1D computational efficiency, it also has several drawbacks. In particular, no one-to-one correspondence exists from one image to the next. In this case the information needs to be modeled as a point-to-contour type feature. This will be investigated further in following chapters.

(a)


(b)

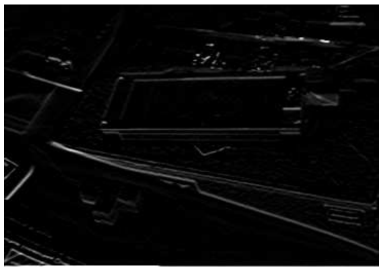


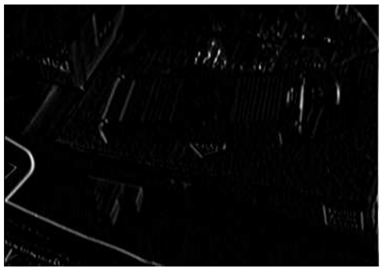
(c)



(d)

Figure 4.11: A blown up view of two real edges (a) and (b) taken from the bottom part of the micro-controller shown in Figure 4.5(a). In (c) and (d) the likelihood values corresponding to these test sites are given along the normal to the contour. The mask size used was 6×6 and the contour was searched 5 pixels either side of the initial position.


(a)


(b)

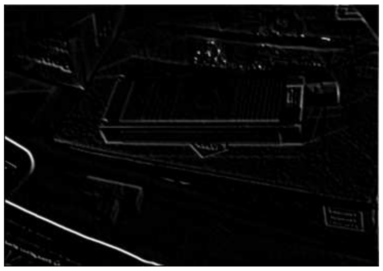
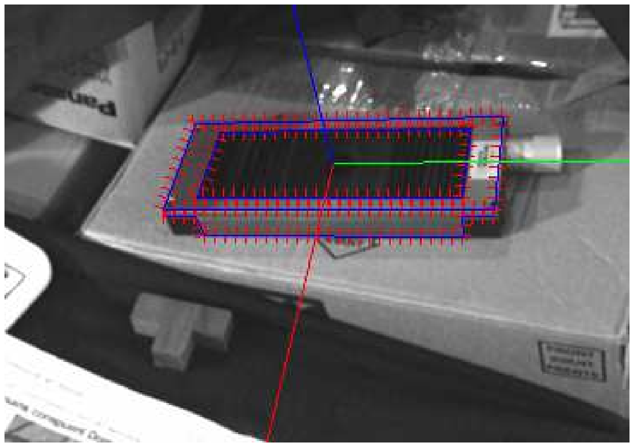
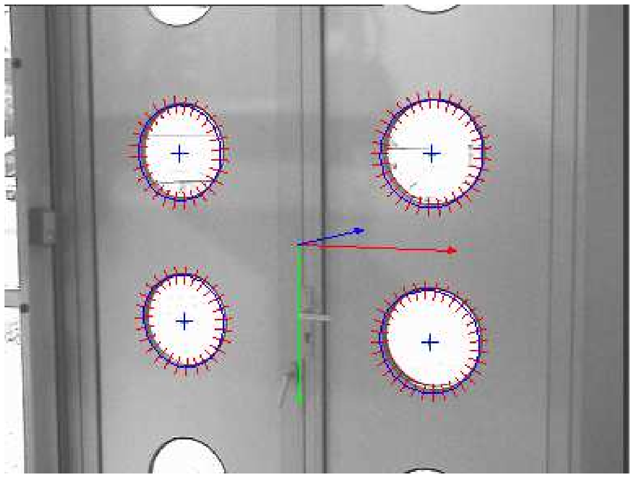
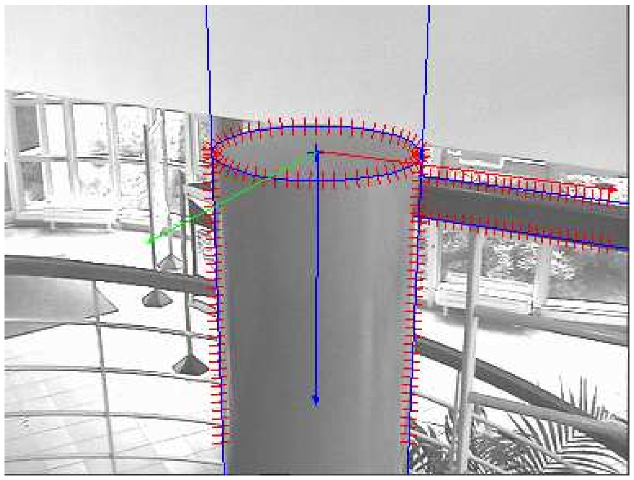

(c)

Figure 4.12: Different spatial Moving Edge mask orientations applied to the micro-controller in Figure 4.5(a). (a) a horizontal mask, (b) a vertical mask and (c) a diagonal mask. It can be seen that the masks are quite selective on the orientation of the contour.


(a)


(b)


(c)

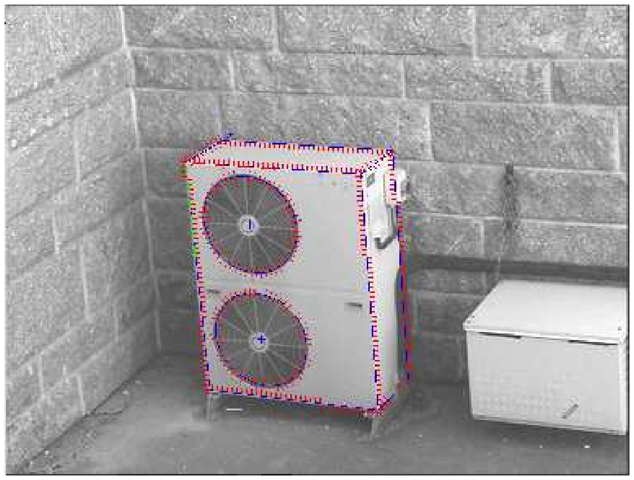

(d)

Figure 4.13: A search for a corresponding edge in an image with starting from the previously known position. (a) the micro-controller sequence, (b) the door sequence, (c) the entrance at IRISA and (d) an outdoor air-conditioner. Note that the search path is marked in red while the model contour are drawn in blue.

The drawback of this is in the implementation of equation (4.44) which takes into account the *temporal* nature of the contour. In this equation it is made clear that the temporal components are separable so that the likelihood equation between two subsequent images can be re-written as:

$$\zeta(\phi_g) = \left| \sum_{\eta \in M_{t0}} a^g_\eta f_{\varrho+\eta} + \sum_{\eta \in M_t} a^g_\eta f_{\varrho+\eta} \right| \tag{4.52}$$

where the left term is the mask applied at t_0 and the right at time t.

In order to evaluate this equation it is necessary to have a one-to-one correspondence between images. This is of particular importance at corners or on textured objects where the matching of a point at t_0 to another point nearby at time t is flagrant.

In this case the solution is to re-apply the mask convolution after the estimation of the object position. In this way there is a one-to-one mapping of the edge point between t_0 and t_1. The downside of this is the fact that the computational cost is doubled. The effect of this was performed experimentally and no significant improvement was obtained out of the temporal component across one image. If more than one image is considered the bias introduced by assuming that a one-to-many correspondence is actually one-to-one is extremely important. Furthermore, the computational cost is no longer feasible in real-time.

4.7.3 Contrast Criterion

In the original formulation, apart from the test criterion given in (4.44), another double inequality criterion is usually added in practice to test for large swings in contrast:

$$\mu_1 \leq \left| \frac{\zeta_t(\phi)}{\zeta_{t0}(\phi)} \right| \leq \mu_2, \tag{4.53}$$

where μ_1 and μ_2 are two predetermined thresholds.

After experiments were performed using this original criterion it was observed to fail in real scenarios. In fact, even though the criteria is designed to be both invariant to illumination changes and to reject large swings of contrast, it can be shown that in a real scene large and instantaneous changes of contrast occur frequently if the object is moving through the environment. For illustrative purposes consider a white object over a black background at time t_0 which moves to a new position with a grey background at time t_1(Refer to Figure 4.14). In this case the likelihood can be taken as being at maximum with $\zeta_{t0}(\phi) = 1$ and at the second time instant can be considered to have half the likelihood with $\zeta_t(\phi) = 0.5$. In this case, it would therefore be necessary to have $\mu_2 > 2$ so that this valid transition is not rejected. Furthermore, it can be shown that as the grey background approaches a likelihood of 0 (i.e. a weak contour) the contrast approaches infinity meaning that the required value of $\mu_2 \rightarrow \infty$ approaches infinity. Of course the inverse scenario is also true for $\mu_1 \rightarrow 0$ (i.e. white object on the grey background moving towards the black). Therefore it can be said that criterion (4.53) does not add any valuable assumptions in many real world target tracking scenarios. This is particularly relevant in environments with a cluttered background or textured backgrounds. It can be said that this scenario is mainly relevant the external contour of the object, however, it can be noted that severe shadows or specular reflection may also interfere with inter-object contours. On the other hand, if pure camera motion is observed and the environment remains static then the assumption of contrast continuity is much more relevant. Even though a full range of contrasts can potentially occur in a real scene, a compromise could be made between temporal continuity and extreme contrast swings by choosing $\mu_1 = 0.5$ and $\mu_2 = 2$. Unfortunately, this does not account for all edge configurations. Therefore, the use of this criterion is considered to be better suited to under-constrained systems where little other *a priori* information is available, such as a CAD model in the case of this thesis.

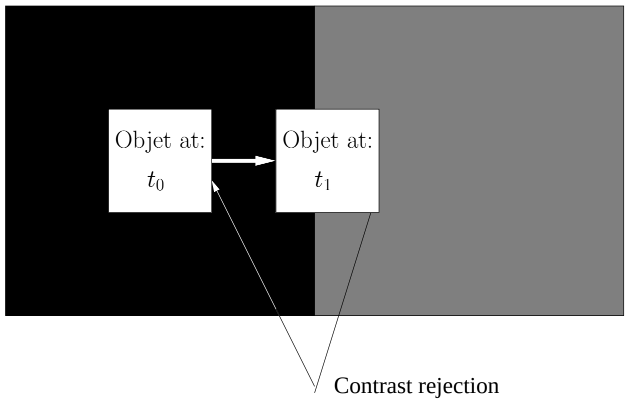


Figure 4.14: Search for a corresponding point in the current image using the moving edge algorithm.

4.8 Conclusion and Perspectives

Although low level feature analysis has been the object of much study throughout the past, much more work remains to fully take advantage of forward and backward propagation of information between high and low levels. Treating this aspect more thoroughly may result in more general methods to solve many of the open problems.

Amongst those problems which remain open include:

1. How to treat the occurrence of more than one edge in the search path.
2. Only "step" edges are considered and more complex edges are potentially intractable.
3. While methods are shown to obey the given model, robustness under the influence of external influence has not been considered.

More particularly, the first problem makes reference to the occurrence of multiple edges in the search path. At present this is avoided by assuming that the search path is sufficiently small to contain only one edge and that the edge must have the same orientation. It is clear, however, that this is a limiting factor on the observable displacement magnitude and subsequently hinders the maximum speed with which an object can be tracked in the image (apart from the physical video cadence limitation). Several solutions to this are foreseeable:

- Particle Filters - [Isard98a, Isard98b, Doucet00, Doucet01] particle filters provide a means of maintaining multiple hypotheses simultaneously. In the case of low level tracking this could be employed to keep track of multiple edge hypotheses during the pose estimation and dynamically selecting the best hypothesis.
- Bayesian decision tree - [Ricquebourg97, Ricquebourg00] implemented a Markov formulation which takes into consideration local constraints such as local smoothness linked to an energy function.
- Robust estimators - The use of multiple robust functions that can be used to selection a single edge [Vacchetti04a].
- Inference - It could be possible to consider more hypothesis by using MML to infer a Mixture Model of many sources of intensity variation, including multiple edges.

Of course the choice of such a method would necessitate to take into account many performance criteria including computational efficiency, and discontinuity problems such as when two edges appear from one or vice versa.

The second point points out the fact that many different types of surfaces and edges exist in a real image. In the literature many different types of edges have been considered including roof edges or curved edges. The conclusion in the studies presented in this paper came to the conclusion that the step edge was the most general form and could take into account different types of edges easily. Furthermore, it has been shown that it is possible to separate one dimensional component for this type of edges. Even so, if more estimation precision is needed this becomes an important factor. Indeed, if a model of the object already exist, as will be proposed in the following chapter, this type of model could also be *a priori* known. Otherwise, it could be possible to extend the set of test hypotheses so that different types of edges could be inferred.

The third point consists in taking into consideration the 3D nature of the environment. In particular many real dynamic and moving environments involve the occlusion of edges or illumination conditions which saturate the image intensity at given location. A possible solution could involve exploiting the full extent of the information available in an image stream such as color [Comaniciu02] or texture. This would also require some sort of framework similar to those mentioned previously which can handle multiple cue fusion [Moreno-Noguer04].

As has been shown in Chapter 3, the inclusion of global three dimensional estimation allows for use of low level information within a global model. The following chapter will complete the description of the pose estimator by showing how local contour information is used within the global estimation process. Furthermore, Chapter 6 will introduce robust statistics into the global estimation problem allowing to handle a large majority of these difficulties including multiple edges, occlusion and saturation.

CHAPTER 5

1D-3D Interaction Model

5.1 Introduction

Now that the foundations of rigid-body dynamics, image formation and low level feature extraction have been laid, this chapter will address the cohesion of these concepts into a fully fledged real-time 3D pose estimation algorithm. At the core of the pose estimation algorithm is the interaction between the movement of 1D distance features in the image and the movement of the camera and object relative to one another in 3D.

The overall model can therefore be summarized as being composed of:

- a 3D spatio-temporal model,
- a pinhole camera model,
- a low level edge illumination model,
- a general class of parametric 3D rigid object models,

where the velocity twist model will relate the camera to a 3D parametric object while the camera model will transform the 3D metric coordinates into image coordinates.

Subsequently, the interaction between the movement of the low level search features and a more global 3D model of the object and camera will be derived in this chapter.

5.2 Literature Review

Although the review of different objective functions and visual features has been largely covered in the previous chapters, a short, yet more focused review is given here on the interaction matrix. The interaction between features in the image and the 3D pose between an object and a camera is defined by a matrix which maps the image features to a six-dimensional configuration space, known as a twist vector, as described in Chapter 2. This matrix can be determined in many different ways depending mainly on the type of visual information used, as well as the different approximations made in defining it.

In summary, the most common features used for pose calculation include points [Horaud89, Haralick89, Dementhon95], segments [Dhome89, Deriche90, Gennery92, Marchand99a]. Common volumetric features considered for pose estimation include conics [Safaee-Rad92, De Ma93], cylinders [Dhome90], superquadrics [Borges96] and generalized cylinders [Liu93, Zerroug95]. More complex objects have been considered by using the polar signature or Fourier Transform of a planar contour [Collewet02], using conformal geometric algebra [Rosenhahn05], or using a moment-based approach [Chaumette02b, Tahri05]. The interaction matrix is then determined analytically from the different parametric equations. The difficultly of using greater order visual features is that they require a complex and costly segmentation step which is often side-stepped by using markers. Furthermore, this complex segmentation phase remains very sensitive to noise in the image as well as the complex 3D configuration of the environment. Even so, one of the interests of a moment-based approach is that it allows pose estimation from a correspondence between a single image feature and the entire object. In this way the expensive correspondence problem is avoided. Furthermore it has been shown to be stable and globally convergent in front of the image plane which also means this technique is well adapted to the object recognition problem.

Other techniques have been proposed which avoid the complex segmentation problem by working directly with the image intensities. A common method, based on appearance [Murase93, Nayar94, Murase95], relies on principal component analysis where the coefficients corresponding to the largest Eigen Vectors are selected as image features. The interaction matrix is then learnt by generating movements using each degree of freedom separately and observing the corresponding movement of the features in the image. This type of representation is useful for visual features of complex objects such as in the case of face recognition [Pentland94]. These methods have also been applied to robot vision [Deguchi96, Nayar96b] for approximating the interaction matrix. The principal advantage of learning the interaction matrix is that the complex modeling phase is avoided. Therefore it becomes rapidly possible to take into account complex environments. Furthermore, this approach is reasonably robust to changes in the environment which often only have a small effect on the principal components. The disadvantage is, however, that it remains impossible to prove the stability of the system and the behavior remains unpredictable and approximate. The modeling phase also remains in the form of teaching methods which need to be devised carefully so as to minimize introducing error into the learnt matrix.

A second group of methods is based on modeling the geometric nature of the visual information. It is also possible to *learn* the interaction matrix corresponding to a particular geometric feature. This can be achieved via purely numerical estimation of entries in the interaction matrix [Weiss87, Suh93, Wells96, Hosoda94] and [Jägersand97]. Methods that estimate parameters in an off-line learning stage include [Weiss87, Ruf99a] and using a batch technique [Lapresté04]. It is also possible to use neural networks [Suh93, Wells96]. Another approach is to continually estimate and update the estimation online [Kinoshita94, Hosoda94, Boukir93, Chaumette96, Jägersand97]. Amongst these methods there are those which provide an analytical geometric relationship between parametric models of visual information and the 3D pose of the camera with respect to objects in the scene and then only estimate the unknown parameters. These parameters can be the pose parameters [Chaumette90, Espiau92], the intrinsic camera parameters [Kinoshita94, Marchand01, Marchand02a], the depth parameters [Chaumette96] or articulated object parameters [Ruf99a]. The advantage of the analytic approach is therefore that it is more precise and it is possible to analyze closely the stability and convergence characteristics. Nevertheless, it remains difficult to model increasing complexity of objects and their interaction in a general manner.

Considering both the approach based on image intensities and the geometric approach, in [Hager98a] a parametric model of geometry and illumination is minimized. This method is robust and learns its image Jacobian from a template image which is compared to the current image and related by affine deformations of the image. This belongs to a class of correlation based methods and in this case the Sum of Squared Difference is used to minimize an objective function between templates. This approach, however, is limited to 2D templates. In [Jurie01, Jurie02] several templates of a 3D object are employed to overcome these problems. Another method has been presented in [Kollnig95] where the pose of a vehicle is estimated by directly matching a polyhedral vehicle model to image gradients without an edge segment extraction process. In this work a synthetic gradient image is compared to the actual image intensity gradient of the current frame using a MAP pose estimation process. As it turns out, this method is nearly equivalent to using facet based edge detector, however, the notion of directly using image intensities inspires a continuous estimation process.

With regard to the complexity of the object model, in practical terms the tendency in the computer graphics literature is towards local approximations which provide an efficient representation without added complexity. The advantage of an analytical derivation being that it is both more precise and allows one to obtain a convergence proof.

5.2.1 Lie group formulation of a distance feature

A recent method has been proposed by Drummond and Cipolla [Drummond02] which is relevant to the method proposed in this thesis. In this work, the velocity field is defined from Lie Group generators which are projected in the direction of the contour normal so as to represent different rigid motion parameters as components of a distance (see Figure 5.1). An introduction to Lie groups and Lie algebra is given in [Varadarajan74, Sattinger86] and the application of Lie Algebras to visual servoing is given in [Drummond00a].

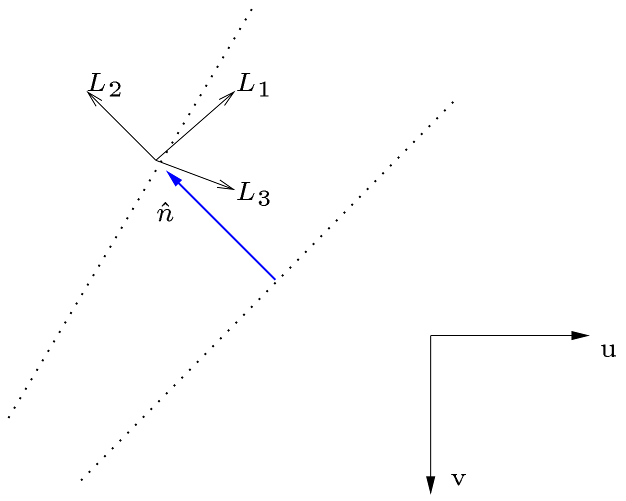


Figure 5.1: Computing the generator fields

For the simple case of a point in 3D, a projection matrix is defined from the intrinsic camera parameters homogeneous transformation matrix $\mathbf{K}$, given in equation (3.9), and the extrinsic pose parameters homogeneous pose transformation matrix $\mathbf{M}$, given in equation (2.13) and the homogeneous transformation matrix $\mathbf{F}$, given in equation (3.4). Combining both these matrices, the image coordinates of a 3D point are then given by:

$$\begin{bmatrix} u \\ v \\ 1 \end{bmatrix} = \frac{1}{Z}\mathbf{KFM}\begin{bmatrix} X \\ Y \\ Z \\ 1 \end{bmatrix}, \tag{5.1}$$

where these transformation matrices have been discussed in detail in Chapters 2 and 3.

In order to perform pose estimation, the derivative of $SE(3)$ corresponding to the tangent velocity vector space or Lie Algebra is considered. The 4×4 basis matrices of the Lie Algebra are used explicitly. These basis matrices are called generators and are chosen in a standard way to represent translations in the x, y and z directions along with rotations around the x, y and z axes. These six generators are given as:

$$\mathbf{G}_1 = \begin{bmatrix} 0 & 0 & 0 & 1 \\ 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 \end{bmatrix}, \quad \mathbf{G}_2 = \begin{bmatrix} 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 1 \\ 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 \end{bmatrix},$$

$$\mathbf{G}_3 = \begin{bmatrix} 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 1 \\ 0 & 0 & 0 & 0 \end{bmatrix}, \quad \mathbf{G}_4 = \begin{bmatrix} 0 & 0 & 0 & 0 \\ 0 & 0 & -1 & 0 \\ 0 & 1 & 0 & 0 \\ 0 & 0 & 0 & 0 \end{bmatrix}, \tag{5.2}$$

$$\mathbf{G}_5 = \begin{bmatrix} 0 & 0 & 1 & 0 \\ 0 & 0 & 0 & 0 \\ -1 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 \end{bmatrix}, \quad \mathbf{G}_6 = \begin{bmatrix} 0 & -1 & 0 & 0 \\ 1 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 \end{bmatrix},$$

The Lie Group of displacements is related to the Lie Algebra of velocities via the exponential map given in equation (2.59), so that:

$$\mathbf{M} = exp(\alpha_i \mathbf{G}_i), \tag{5.3}$$

where $\mathbf{M}$ is the homogeneous form of the pose given in (2.13) and each α_i corresponds to an element of the kinematic screw or twist representing the inter-frame transformations. The motion in the image is related to the twist in 3D by taking the partial derivative of projective image coordinates with respect to the i^{th} generating motion:

$$\begin{pmatrix} u_i' \\ v_i' \\ 1 \end{pmatrix} = \mathbf{KFMG}_i \begin{pmatrix} X \\ Y \\ Z \\ 1 \end{pmatrix}. \tag{5.4}$$

Taking the derivative of the perspective projection of a point leads to the motion of a point in the image as:

$$\mathbf{l}_i = \begin{bmatrix} \dot{x}_i \\ \dot{y}_i \end{bmatrix} = \begin{bmatrix} \frac{u_i' Z_i'}{Z_i} - \frac{u_i Z_i'}{Z_i} \\ \frac{v_i' Z_i'}{Z_i} - \frac{v_i Z_i'}{Z_i} \end{bmatrix}. \tag{5.5}$$

Note that this notation has been kept consistent with [Drummond02] and the derivation of its matrix form will be reconsidered in more detail in the next section. Furthermore, it will be shown in the next section that this equation is fundamental to the study of more complex geometric features.

Continuing with the determination of the inter-frame movement, the different generating motions are projected in the direction of the normal of the contour as:

$$f_i = \mathbf{l}_i.\hat{\mathbf{n}}, \tag{5.6}$$

where $\hat{\mathbf{n}}$ is the normal direction.

5.3 Interaction Matrices

Definition:
The interaction matrix $\mathbf{L_s} \in \mathbb{R}^{n \times m}$ represents the *projection* of a m-dimensional velocity field, related to the pose between a camera and the scene, onto a n-dimensional velocity field in the image. In this way each row of the interaction matrix relates the 3D configuration between a camera and a three-dimensional primitive to its corresponding visual information $\mathbf{s}$. Moreover, this matrix relates the measurements in then image to the 3D movement in the scene.

As mentioned in Section 5.2, there are two general methods for determining an interaction matrix for any type of image feature. The first way is to solve for it numerically by using learning techniques and the second is to derive it analytically and estimate unknown parameters. In this thesis, preference is given to the analytical derivation due to its known and accurate representation of the interaction between the image feature velocities and 3D rigid body velocities. Furthermore, an analytical derivation also allows one to analyze convergence and stability characteristics related to different feature configurations as well as difference scene configurations.

As will be shown, all interaction matrices can be derived by reusing the simplest interaction matrix, being that of a point. This central interaction matrix is founded on the twist model between coordinate frames combined with the perspective projection model. The interaction matrix is therefore based on the perspective projection of twist velocities. This projection matrix is at the heart of the interaction between the 2D image primitives and the 3D image primitives due to the fact that 2D image measurements are inherent in the physical camera sensor and 3D primitives are based on geometric properties of Special Euclidean space.

Although a general formalism has been provided for determining an interaction matrix for any type of visual feature [Chaumette90, Espiau92], up until recently derivations have only been given for visual features which have a dimension greater than or equal to 2 (i.e. $d \geq 2$). Even so, one dimensional features have been considered by using a single parameter from a higher dimensional visual feature such as the x or y coordinate of a point or the center of gravity. More recently, moment based features have provided the basis for one-dimensional visual features such as the area of a planar object [Tahri03, Chaumette04]. In this chapter, the formalism will be used to derive an interaction matrix for various one-dimensional type distance primitives ($d = 1$) related to the projection of 3D contours. In particular, the case of a point-to-ellipse distance is analyzed in detail and the algebraic as well as geometric distances are considered. The use of distance features allows for new many-to-one mapping of feature correspondences, where the corresponding feature parameters $\boldsymbol{\phi}$ are considered in equation (5.15). This mapping is one of the major differences with traditional methods, where one-to-one mappings were previously considered, such as the case of a point-to-point or line-to-line correspondence. In these cases it was necessary to estimate local feature parameters before estimating global parameters therefore creating an inefficient two step process. In this thesis, many to one feature correspondences are considered for various types of contours so as to provide a distance-to-contour type minimization equation. The advantages of distance features are described in the following paragraph.

The remainder of this section describes the derivation of distance interaction matrices between any projected contour of a rigid 3D object and multiple corresponding points in the image. A 3D object is defined by a collection of piecewise parametric equations. Distance primitives are advantageous for many reasons, including:

1. They are computationally efficient due to a one-dimensional search for correspondences in the image.
2. They exhibit a high level of precision and accuracy due to the simultaneous consideration of all the distances corresponding to a complex 3D object in a non-linear framework.
3. They can be easily compared to one another in order to obtain robust statistical information.

Moreover, a distance feature is considered as a scalar quantity representing an error in the image between local point features obtained from a fast image processing step and the contours of a more global CAD model. It is important to note, however, that a point found normal to a contour is nothing more than a scalar quantity, even if it is considered as a point. This is because the point has been found to the normal of the contour and a one-to-many correspondence exists. In other words, no two-dimensional information is known. The desired value of the distance is equal to zero unlike the case of one-to-one type interaction matrices. Furthermore, by estimating the entire pose simultaneously, intermediate estimation of features such as lines or ellipses (as was done previously in [Lowe92, Marchand99b]) can be avoided. All distances are then treated according to their corresponding contour.

The general derivation of the interaction matrix starts by reconsidering the simple case of a point in 3D. Thus, consider the homogeneous 3D point $\bar{\mathbf{P}}$ already expressed in the camera frame $\mathcal{F}_c$ so that no extrinsic transformation is required. The perspective projection of the point is performed as in equation (3.4) and the relative motion between a point and the camera is defined in equation (2.13) leading to:

$$\bar{\mathbf{p}} = \frac{1}{Z}\mathbf{F}\bar{\mathbf{P}}. \tag{5.7}$$

where the homogeneous transformation matrix $\mathbf{F}$ has been given in equation (3.4) on page 50.

The relative motion between a point $\mathbf{P}$ and the camera as defined in equation (2.31) is:

$$\dot{\bar{\mathbf{P}}} = [\mathbf{v}]_\wedge \bar{\mathbf{P}}, \tag{5.8}$$

where $\mathbf{v} = (\boldsymbol{\upsilon}, \boldsymbol{\omega})$ is the velocity twist vector that is transformed into a homogeneous matrix by the operator $[.]_\wedge$ and where $\boldsymbol{\upsilon} = (\upsilon_x, \upsilon_y, \upsilon_z)$ is the translational component of the velocity vector and $\boldsymbol{\omega} = (\omega_x, \omega_y, \omega_z)$ the angular velocity.

If equation (5.8) is expanded into components this gives:

$$\begin{aligned} \dot{P}_x &= -\upsilon_x - \omega_y Z + \omega_z Y, \\ \dot{P}_y &= -\upsilon_y - \omega_z X + \omega_x Z, \\ \dot{P}_z &= -\upsilon_z - \omega_x Y + \omega_y X. \end{aligned} \tag{5.9}$$

To obtain the relationship between the velocity of the 3D point $\mathbf{P}$ and the velocity of a point $\mathbf{p}$ in the image (as was mentioned briefly in equation (5.5)), the time derivative of equation (5.7) using the quotient rule gives the motion field:

$$\dot{\mathbf{s}_p} = \frac{Z\dot{\mathbf{P}} - \upsilon_z \mathbf{P}}{Z^2}, \tag{5.10}$$

where the focal distance is assumed to be $f = 1$ so that $\dot{\mathbf{s}}_p = (\dot{s}_x, \dot{s}_y)$ in expanded form is:

$$\begin{array}{l} \dot{s}_x = \frac{v_z x - v_x}{Z} - \omega_y + \omega_z y + \omega_x xy - \omega_y x^2, \\ \dot{s}_y = \frac{v_z y - v_y}{Z} + \omega_x - \omega_z x - \omega_x xy - \omega_y x^2 \end{array} \tag{5.11}$$

and in matrix form is written as:

$$\dot{\mathbf{s}} = \mathbf{L}_p \mathbf{v}. \tag{5.12}$$

Finally, the interaction matrix $\mathbf{L}_p$ for a point is:

$$\boxed{\mathbf{L}_p = \begin{bmatrix} -Z^{-1} & 0 & xZ^{-1} & xy & -(1+x^2) & y \\ 0 & -Z^{-1} & yZ^{-1} & 1+y^2 & -xy & -x \end{bmatrix}, \qquad (5.13)}$$

which provides the fundamental interaction between the movement of a point in the image and the movement of the pose between the camera and the object. As will be shown, this forms the fundamental relation for deriving many different interaction matrices for various types of geometric features.

5.4 Generalized Interaction Matrix

This section is concerned with calculating the interaction matrix $\mathbf{L}_\mathbf{s}$ for any kind of three-dimensional geometrical primitive of the object and a corresponding point in the image.

Indeed, in classical image-based techniques [Chaumette90, Espiau92, Chaumette00], a general framework exists to analytically compute $\mathbf{L}_\mathbf{s}$. Using this framework, visual servoing tasks have been demonstrated which implement various types of image information including points, lines, circles, quadratics, etc...

Definitions

The general derivation of the interaction matrix for any type of geometric primitive is based on the 3D object representation already given generally in equation (3.13) combined with the definition of the following equations:

- **Visible Contour Parametrization**

  This equation is based on the intersection of the line of sight between the image plane and the visible parts of the contour. The visible part of the contour must therefore be defined by a parametric equation of the form:

$$\boxed{\mu(x, y, \Phi) = Z, \qquad (5.14)}$$

  where $\mathbf{p} = (x, y)$ is a 2D point and the existence of a unique function around $\mathbf{P}_o$ is ensured by the implicit function theorem.

  In the case of planar features, this parametric equation can be found by the intersection of a plane passing through the origin of the object coordinate system and parallel to the image plane. On the other hand, if volumetric objects are considered, it is important to take into account and self-occlusion of the object. In this case the corresponding visible contour is referred to as the limb surface. In the planar case, consider, for example, a three-dimensional line defined by the intersection of two three-dimensional planes h_1 and h_2. In this case, the corresponding visibility of equation μ can be derived directly from either the two corresponding parametric 3D equations h_1 or h_2 since the feature is always visible. In the case of a three-dimensional primitive it is necessary to determine its limb surface or contour in the image such as in the case of cylinder.

Assumption: Here the important assumption is made that there is no other source of occlusion other than in the case of the limb surface for a geometric object. However, as mentioned in Chapter 3, various piecewise features can be combined together to create a more complex object. However, in this case it is also necessary to take these different components in the calculation of 'visible' piecewise contours and limb surfaces. In this way self occlusions are effectively handled by the use of well known Hidden Surface Removal (HSR) techniques such as Back Face Culling, Binary Space Partition Trees or ray-tracing(see Appendix B.1).

- **Two-dimensional Contour Parametrization**

  A parametric function describing the contour of the 3D object projected onto the image plane:

$$\boxed{g(x, y, \phi) = 0, \qquad (5.15)}$$

  where $\boldsymbol{\phi} = (\phi_1, \ldots, \phi_m)$ are m parameters describing the projected contour of the object.

Assumption: In this equation the assumption is made that it uniquely parameterizes the two-dimensional primitive. For example, in the case of a line it is necessary to define a direction of the line in order to obtain a unique parametrization.

5.4.1 General Interaction Matrix Derivation

From this general set of equations it is possible to derive the general interaction matrix by first differentiating a primitive $\mathbf{s}$ with respect to time, where its relation to the parameters is given by a function $\mathbf{s} = f(\boldsymbol{\phi})$. Given the definition of equation (5.15), for a general primitive $\mathbf{s}$ in the image, it is possible to start deriving the general interaction matrix. First derive the 2D primitive according to (3.31) with respect to time as follows:

$$\frac{\partial \mathbf{s}}{dt} = \frac{\partial \mathbf{s}}{\partial \boldsymbol{\phi}} \frac{\partial \boldsymbol{\phi}}{\partial \mathbf{r}} \frac{\partial \mathbf{r}}{dt}, \qquad (5.16)$$

where the 2D primitive depends on the 2D parameters $\boldsymbol{\phi}$, which subsequently depends on the pose $\mathbf{r}$ and which finally depends on time t.

Differentiation of (5.15) gives:

$$\frac{\partial g}{\partial \boldsymbol{\phi}}(\mathbf{p}, \boldsymbol{\phi})\dot{\boldsymbol{\phi}} = -\frac{\partial g}{\partial \mathbf{p}}(\mathbf{p}, \boldsymbol{\phi})\dot{\mathbf{p}}. \qquad (5.17)$$

Here the well known optical flow equation is found on the right hand side which is directly related to the interaction matrix of a point (5.13). In other words, the generalized interaction matrix is based on the point interaction matrix.

Substituting the corresponding interaction matrix for $\dot{\mathbf{p}}$, into equation (5.17), gives:

$$\frac{\partial g}{\partial \phi}(\mathbf{p},\phi)\dot{\phi} = -\frac{\partial g}{\partial \mathbf{p}}(\mathbf{p},\phi)\mathbf{L_s}(\mathbf{p},Z)\mathbf{v}. \tag{5.18}$$

The perspective projection of the feature parameters is done by substituting equation (5.14) into the interaction matrix for a point from equation (5.13) giving equation (5.18):

$$\frac{\partial g}{\partial \phi}(\mathbf{p},\phi)\dot{\phi} = -\frac{\partial g}{\partial \mathbf{p}}(\mathbf{p},\phi)\mathbf{L_s}(\mathbf{p},\mathbf{\Phi})\mathbf{v}. \tag{5.19}$$

This equation is then solved explicitly for particular primitives, or alternatively, it is possible to solve this by identification [Chaumette90, Espiau92].

5.5 Point-to-Line Distance

In this section, the most simple 1D feature will be considered before looking at more complex features. Since the technique presented here is based on efficient 1D searches, the most simple feature that can be considered is the distance between a corresponding point and the projection of a 3D model line in the image. The line is then sampled in 2D and a 1D search is made to the normal of the line. In Figure 5.2 $\mathbf{p}$ is the desired point feature position and $\mathbf{l}$ is the current line feature position. If the aperture problem [Bouthemy89] is present then the distance between $\mathbf{p}$ and $\mathbf{l}$ is characterized by the distance perpendicular to the segment. In other words the distance parallel to the segment does not hold any useful information unless a correspondence exists between a point on the segment and $\mathbf{p}$.

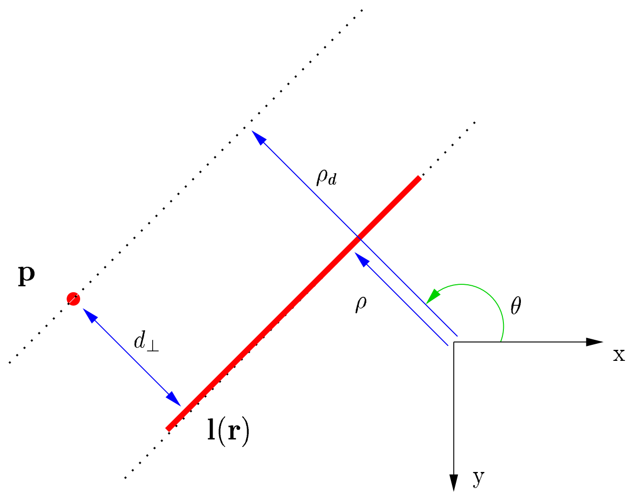


Figure 5.2: Distance of a point normal to a segment

5.5.1 Defi nition

To derive the interaction matrix for a point-to-line distance, the 3D configuration is considered as a line and its 2D configuration is considered as a distance.

For the 3D configuration a line can be represented as the intersection of two planes:

$$\begin{aligned} h_{l1}(\mathbf{P}, \mathbf{\Phi}_{l1}) &= A_1X + B_1Y + C_1Z + D_1 = 0, \\ h_{l2}(\mathbf{P}, \mathbf{\Phi}_{l2}) &= A_2X + B_2Y + C_2Z = 0, \end{aligned} \tag{5.20}$$

where $\mathbf{\Phi}_{l1} = (A_1, B_1, C_1, D_1)$ and $\mathbf{\Phi}_{l2} = (A_2, B_2, C_2)$. The configuration of the line is then $h_l(\mathbf{P}, \mathbf{\Phi}_l) = (h_{l1}(\mathbf{P}, \mathbf{\Phi}_{l1}), h_{l2}(\mathbf{P}, \mathbf{\Phi}_{l2}))$. This parameterizations is not minimal, however, it allows complete description of a line in 3D.

Since this is essentially a two-dimensional or planar primitive the visibility equation (5.14) is trivial and can be obtained from h_{l1} as:

$$\frac{1}{Z} = \alpha x + \beta y + \gamma, \tag{5.21}$$

where $\alpha = -A_1D_1^{-1}$, $\beta = -B_1D_1^{-1}$ and $\gamma = -C_1D_1^{-1}$. The degenerate case of the projection of a line to a point in the image($D_1 = 0$) is excluded.

The perspective projection equation (5.7) applies directly to h_{l1} and is used to obtain the perspective projection for a line giving:

$$g_{l1}(\mathbf{p}, \boldsymbol{\phi}_{l1}) = ax + by + c = 0, \tag{5.22}$$

where $a = A_2$, $b = B_2$ and $c = C_2$.

Given that the parametrization (a, b, c) is not minimal, the corresponding representation (ρ, θ) is chosen as (see Figure 5.2):

$$g_{l2}(\mathbf{p}, \boldsymbol{\phi}_{l2}) = x\cos\theta + y\sin\theta - \rho = 0, \forall(x, y) \in s_l, \tag{5.23}$$

where $\theta = \arctan(ba^{-1})$ and $\rho = -c(a^2 + b^2)^{-1/2}$.

Since the 2D configuration is a distance, the 2D configuration is defined as the perpendicular distance between a point and a line. This is achieved by considering the corresponding point found normal to the line, as belonging to a second straight line, with the same orientation as the projected contour(see Figure 5.2). As will be shown, the interaction matrix is subsequently derived from an interaction matrix between two lines, where one line is the current segment of the model and the other is a line going through p and running parallel to l. The 2D configuration is determined as:

$$g_{dl}(\mathbf{p}, \boldsymbol{\phi}_{dl}) = (\rho - \rho^*) - d_l = 0, \tag{5.24}$$

where ρ^* is the distance from the origin to the point found normal to the contour. This can be expanded by considering that both ρ and ρ^* have the same orientation:

$$g_{dl}(\mathbf{p}, \boldsymbol{\phi}_{dl}) = (\rho - (x^*\cos\theta + y^*\sin\theta)) - d_l = 0. \tag{5.25}$$

5.5.2 Derivation

From these basic definitions, the interaction matrix is now derived. Firstly, deriving (5.24) with respect to time and considering the parameters of the line as time varying, leads to:

$$\dot{d}_l = \dot{\rho} - \dot{\rho^*} = \dot{\rho} + \alpha\dot{\theta}, \tag{5.26}$$

where $\alpha = x^* \sin\theta - y^* \cos\theta$. The remark can be made here that, the interaction matrix for a point-to-line distance is essentially a combination of a line's parameters and $\mathbf{L}_{dl} = \mathbf{L}_\rho + \alpha\mathbf{L}_\theta$. The derivation for a line has already been given in [Chaumette90, Espiau92] and proceeds as follows.

In order to determine $\dot{\rho}$ and $\dot{\theta}$ equation (5.23) is derived with respect to time giving:

$$\dot{\rho} + (x\sin\theta - y\cos\theta)\dot{\theta} = \dot{x}\cos\theta + \dot{y}\sin\theta, \quad \forall(x,y) \in \mathcal{L}. \tag{5.27}$$

By using (5.23), x is expressed as a function of y when $\cos\theta \neq 0$(or y as a function of x in the other case). By using (5.7) and (5.21), (5.27) is written as:

$$(\dot{\rho} + \rho\tan\theta\dot{\theta}) + y\left(-\frac{\dot{\theta}}{\cos\theta}\right) = K_2\mathbf{v} + yK_1\mathbf{v}, \quad \forall y \in \mathbb{R}, \tag{5.28}$$

where

$$\begin{array}{rcl} \mathbf{K}_1 & = & [\lambda_1\cos\theta \quad \lambda_1\sin\theta \quad -\lambda_1\rho \quad \rho \quad \rho\tan\theta \quad (\cos^\theta)^{-1}] \\ \mathbf{K}_2 & = & [\lambda_2\cos\theta \quad \lambda_2\sin\theta \quad -\lambda_2\rho \quad \sin\theta \quad -\cos\theta - \rho^2(\cos\theta)^{-1} \quad -\rho\tan\theta] \end{array}$$

and $\lambda_1 = (-a_i\tan\theta + b_i)d_i^{-1}$ and $\lambda_2 = (a_i\rho(\cos\theta)^{-1} + c_i)d_i^{-1}$.

This leads to:

$$\begin{array}{l} \dot{\rho} = (K_2 + \rho\sin\theta K_1)\mathbf{v}, \\ \dot{\theta} = -K_1\cos\theta\mathbf{v}. \end{array} \tag{5.29}$$

Equation (5.29) written in matrix form leads to the interaction matrix for a line:

$$\begin{array}{rcl} L_\rho & = & [\lambda_\rho\cos\theta \quad \lambda_\rho\sin\theta \quad -\lambda_\rho\rho \quad (1+\rho^2)\sin\theta \quad -(1+\rho^2)\cos\theta \quad 0], \\ L_\theta & = & [\lambda_\theta\cos\theta \quad \lambda_\theta\sin\theta \quad -\lambda_\theta\rho \quad -\rho\cos\theta \quad -\rho\sin\theta \quad -1] \end{array}, \tag{5.30}$$

with $\lambda_\rho = (A_2\rho\cos\theta + B_2\rho\sin\theta + c_2)D_2^{-1}$ and $\lambda_\theta = (A_2\sin\theta - B_2\cos\theta)D_2^{-1}$.

Using (5.30) and considering that $\mathbf{L}_{dl} = \mathbf{L}_p + \alpha\mathbf{L}_\theta$ the interaction matrix for a point-to-line distance is finally deduced as:

$$L_{d_l} = \begin{bmatrix} \lambda_{d_l}\cos\theta \\ \lambda_{d_l}\sin\theta \\ \lambda_{d_l}\rho \\ (1+\rho^2)\sin\theta - \alpha\rho\cos\theta \\ -(1+\rho^2)\cos\theta - \alpha\rho\sin\theta \\ -\alpha \end{bmatrix}^\top, \tag{5.31}$$

where $\lambda_{d_l} = \lambda_\rho + \alpha\lambda_\theta$ and the subscript of d_{d_l} refers to a point-to-line distance.

5.6 Directional Point-to-Point Distance

The previous method models the 3D primitive as a line feature and projects its 6-dimensional twist onto the image via the interaction matrix. Subsequently, the points are sampled along the projection of the contour in 2D and corresponding points are searched for in the direction of the normal. It is possible to reconsider this formulation as the distance between two points and projected in the direction of the normal. In a first part the interaction matrix will be determined for a directional distance between two points. Following this, a comparison will be made with a point-to-line distance as given in the previous section.

In the present case samples are made along a 3D contour, instead of a 2D contour, and these points are projected onto the image. As can be seen in Figure 5.6, a search is been made in the direction of the normal to a contour starting at point $\mathbf{p}$ and ending at the corresponding point $\mathbf{p}^*$.

This technique can therefore be applied to a large range of contours without considering the interaction of these contours explicitly. Unfortunately, however, it is still necessary to determine the normal of the contour in the image. The easiest way to achieve this is to project the 3D contour onto the image and determine the tangent at point $\mathbf{p}$. Thus this approach still requires the projection of the 3D contour as was performed in the previous section.

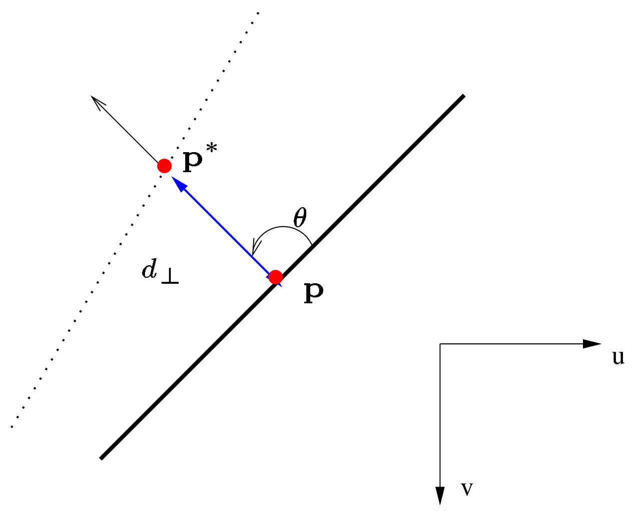


Figure 5.3: Directional distance between two points.

The first step is to define the 3D parametric equation (3.13) of the object model, which in this case is simply a 3D point. The visibility equation (5.14) is simply the depth of the point $Z = Z_0$. The 2D parametric equation (5.15) is again simply a 2D point. Therefore, it remains to define a distance in 2D. This can be done by projecting each x and y coordinate's distance in the direction of the normal as:

$$g_{dp}(\mathbf{p}, \phi) = (x - x^*)\cos\theta + (y - y^*)\sin\theta - d_p = 0. \tag{5.32}$$

where d_p refers to the directional distance between two points as opposed to a point-to-line distance as given in equation (5.31).

This means that the x and y coordinates of a point, on the contour, are considered as time varying, instead of the contour parameters themselves. Therefore, deriving (5.32) with respect to time gives:

$$\dot{d}_p = \dot{x}\cos\theta + \dot{y}\sin\theta, \tag{5.33}$$

which can be written more generally as:

$$\dot{d}_p = \frac{\partial d_p}{\partial x}\dot{x} + \frac{\partial d_p}{\partial y}\dot{y} \tag{5.34}$$

where it can be seen that $\dot{x}$ and $\dot{y}$ correspond to the interaction matrix of a point times the twist velocity.

Therefore the interaction matrix related to (5.34) corresponds to projecting the twist velocity related to a point feature in the direction of the normal θ as:

$$\mathbf{L}_{d_p} = \cos\theta \mathbf{L}_x + \sin\theta \mathbf{L}_y, \tag{5.35}$$

where $\mathbf{L}_p$ is the interaction matrix for a point.

The interaction matrix of a directional point-to-point distance is then given as:

$$\boxed{\mathbf{L}_{d_p} = \begin{bmatrix} -Z^{-1}\cos\theta \\ -Z^{-1}\sin\theta \\ Z^{-1}x\cos\theta + Z^{-1}y\sin\theta \\ xy\cos\theta + (1+y^2)\sin\theta \\ -(1+x^2)\cos\theta - xy\sin\theta \\ y\cos\theta - x\sin\theta \end{bmatrix}^\top . \qquad (5.36)}$$

Although the formulation of the method proposed in [Drummond02] is different, the interaction matrix of a point-to-line distance can be shown to be equivalent. Let the method proposed in [Drummond02] be known as Method 1 and the method presented in this thesis as Method 2. First of all, the equivalence between the notation of Method 1 and the matrix notation of Method 2 is established. The velocities α_i to be estimated in the first method are elements of the velocity twist and can be written as $\mathbf{v} = (\alpha_1 \dots \alpha_6)$. The index i is chosen to index a feature in place of ξ in the first method. The distances are defined similarly giving $d_l = d_\xi$ with error vector defined as $\mathbf{s} = (d_1, \dots, d_n)$, where n is the number of features. In Method 1 components of the interaction matrix were determined individually by the use of Lie Group generators and homogeneous point projection matrices.

The derivation of the interaction matrix of a directional point-to-point distance in the second method can be written in its matrix form by considering each $\mathbf{l}_i$ from (5.5) to correspond to a column of the interaction matrix for a point so that the interaction matrix for a point is given by $\mathbf{L}_p = [\mathbf{l}_1, \dots, \mathbf{l}_6]$. The dot product operator that appears in (5.6) is equivalent to projecting each point co-ordinate's interaction vector in the direction of the normal as in (5.35). Therefore the resulting interaction matrix is the equivalent.

5.6.1 Comparison of Point-to-Line and Directional Point-to-Point

The distance in the image between a point and a line given in equation (5.25) can be rewritten to eliminate ρ by using the equation of a line (5.23). This results in exactly the same equation as a directional point-to-point distance as given in (5.32). The difference lies in the fact that x, y are considered as varying in the case of a directional point-to-point distance, whereas, θ is considered as varying in the case of a point-to-line distance. Furthermore, the case of a point-to-line distance is more complete because it takes into account the projection of the contour normal onto the image plane.

Another difference in the two derivations lies in the computation complexity of the algorithm. This is due to the fact that for the case of a distance-to-point, sampling of the contour occurs in 3D, whereas, in the case of a distance-to-line this is done in 2D. Furthermore, each method does a conversion from meters to pixels in a different way. In the case of a distance-to-point, the 3D points are sampled, projected and then converted to pixels, whereas, in the case of a distance-to-line, the projected contours are both converted to pixels and sampled in 2D. It can be shown that the computational complexity of a distance-to-point based interaction matrix, for the projection for all sampled points onto the image, is order $O(n)$ and for the intrinsic parameter conversion, from meters to pixels, is also $O(n)$ where n is the number of tracked points along the contour. In the case of a distance-to-line, both extrinsic projection and intrinsic conversion are performed in order $O(m)$, where m is the number of higher level features and where $m << n$.

In summary, both methods can be considered as equivalent mathematically yet slightly different in terms of modeling the interaction matrix. The notable difference in the two derivations, although, lies in the completeness of the derivation and in computational complexity of the algorithm.

5.7 Point-to-Contour Distance

In this section, more complex 3D objects will be considered using a moment based parametrization of an ellipse as given in [Chaumette90, Espiau92]. A more recent derivation has been given in [Chaumette04] and [Tahri05] where a general class of planar moments and have been shown to treat very complex planar objects in a general manner.

In order to derive a point-to-contour interaction matrix, a planar 3D circle is considered. The closed contour of a circle is shown to project onto the image in the form of an ellipse. A moment based parametrization is then used to define the closed contour of an ellipse. The interaction matrix for the distance between a point and an ellipse can then be formulated using the general derivation described above.

The two following distances will be considered:

1. Algebraic distance (see Figure 5.4(a))
2. Geometric distance (see Figure 5.4(b))

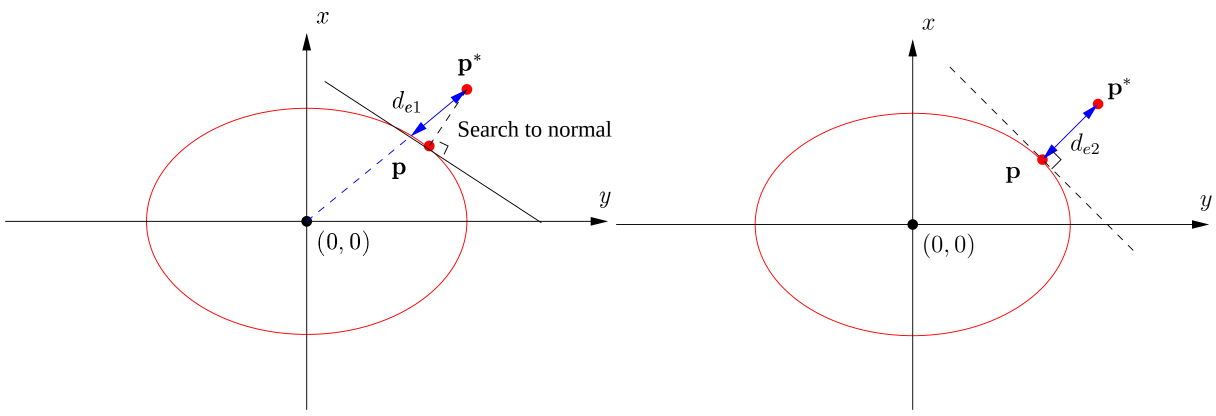


Figure 5.4: The distance of a point $\mathbf{p}^*$ found by searching along the normal to an ellipse at a sampled location $\mathbf{p}$ (a) Algebraic distance determined by using the equation of an ellipse. (b) Orthogonal geometric distance determined by the tangent.

5.7.1 Definition

A circle can be modeled in three-dimensions as the intersection of a sphere and a plane as:

$$\begin{aligned} h_{c1}(\mathbf{P}, \mathbf{\Phi}_{c1}) &= (X - X_0)^2 + (Y - Y_0)^2 + (Z - Z_0)^2 - R^2 = 0, \\ h_{c2}(\mathbf{P}, \mathbf{\Phi}_{c2}) &= A(X - X_0) + B(Y - Y_0) + C(Z - Z_0) = 0. \end{aligned} \tag{5.37}$$

where h_{c1} is obviously the sphere with parameters $\mathbf{\Phi}_{l1} = (X_0, Y_0, Z_0, R)$ and h_{c2} the plane with parameters $\mathbf{\Phi}_{l2} = (A, B, C, X_0, Y_0, Z_0)$. The configuration of the circle in 3D is then $h_c(\mathbf{P}, \mathbf{\Phi}_c) = (h_{c1}(\mathbf{P}, \mathbf{\Phi}_{c1}), h_{c2}(\mathbf{P}, \mathbf{\Phi}_{c2}))$.

Once more this is a planar primitive and the visibility equation μ_c is easily obtained from h_{c2} by using the perspective projection (5.7):

$$\frac{1}{Z} = \alpha x + \beta y + \gamma c. \tag{5.38}$$

where

$$\alpha = \frac{A}{(AX_0+BY_0+CZ_0)}, \quad \beta = \frac{B}{(AX_0+BY_0+CZ_0)}, \quad \gamma = \frac{C}{(AX_0+BY_0+CZ_0)} = \frac{(1-\alpha X_0 - \beta y_0)}{Z_0}$$

By substituting equation (5.38) into h_{c1} the equation of the projected circle is obtained in the image. After factoring the equation in terms of x and y the two dimensional equation of an ellipse is obtained as:

$$g_{e1}(\mathbf{p}, \boldsymbol{\phi}_{e1}) = ax^2 + by^2 + 2cxy + 2dx + 2ey + f = 0 \tag{5.39}$$

where g_{e1} represents this parametrization of an ellipse and where:

$$\begin{array}{ll}
a = \alpha^2(X_0^2 + Y_0^2 + Z_0^2 - R^2) + 1 - 2\alpha X_0, & b = \beta^2(X_0^2 + Y_0^2 + Z_0^2 - R^2) + 1 - 2\beta Y_0, \\
c = \alpha\beta(X_0^2 + Y_0^2 + Z_0^2 - R^2) - \beta X_0 - \alpha Y_0, & d = \alpha\gamma(X_0^2 + Y_0^2 + Z_0^2 - R^2) - \gamma X_0 - \alpha Z_0, \\
e = \beta\gamma(X_0^2 + Y_0^2 + Z_0^2 - R^2) - \gamma Y + 0 - \beta Z_0, & f = \gamma^2(X_0^2 + Y_0^2 + Z_0^2 - R^2) + 1 - 2\gamma Z_0,
\end{array}$$

It should be noted that this ellipse can degenerate into a line in certain cases.

Equation (5.39) can be factorized under the common form:

$$g_{e2}(\mathbf{p}, \boldsymbol{\phi}_{e2}) = \frac{(x - x_g + k_2(y - y_g))^2}{k_0^2(1 + k_2^2)} + \frac{(y - y_g - k_2(x - x_g))^2}{k_1^2(1 + k_2^2)} - 1 = 0 \tag{5.40}$$

where g_{e2} represents this intuitive parametrization of an ellipse and where:

$$\begin{array}{l}
x_g = \left(\frac{bd-ce}{c^2-ab}\right), \quad y_g = \left(\frac{ae-cd}{c^2-ab}\right), \\
k_2 = \left(b - a \pm \frac{\sqrt{(b-a)^2+4c^2}}{2c}\right), \quad k_0 = \left(\frac{2(ax_c^2+2cx_cy_c+by_c^2-f)}{a+b\pm\sqrt{(b-a)^2+4c^2)}}\right), \quad k_1 = \left(\frac{2(ax_c^2+2cx_cy_c+by_c^2-f)}{a+b\mp\sqrt{(b-a)^2+4c^2)}}\right),
\end{array}$$

where x_g and y_g are the coordinates of the center of gravity of the ellipse, k_0 and k_1 its major and minor axes respectively and $k_2 = tan\alpha$ with α being the orientation of the major axes. This parametrization degenerates when the ellipse becomes a circle. In this case k_2 is ambiguous and is the same for any orientation. For this reason it is preferable to represent an ellipse by the coordinates defined by the center of gravity and the normalized centered moments of order two.

Parametrization using moments is now considered. In order to define an ellipse in terms of moments it is necessary to introduce the definitions of a moment, the area based on moments, the center of gravity, the centered moments and the normalized centered moments. First, let it be noted that the order of a moment is determined by the value of $i + j$.

1. **Moment**: A general moment related to a set of image points is given by:

$$m_{ij}(t) = \int\int_{\mathcal{R}(t)} f(x, y)dxdy, \tag{5.41}$$

where $f(x, y) = x^i y^j$ and $\mathcal{R}$ is the projected region of the object in the image. Note that in the case of an ellipse only the contour $\mathcal{C}(t)$ is considered.

2. **Area**:
   The area of a contour is equivalent to the moment of order zero denoted:
$$a = m_{00}. \tag{5.42}$$
3. **Center of gravity**:
   Similarly, the first order moments m_{10} and m_{01} are related to the center of gravity of the object by:
$$x_g = \tfrac{m_{10}}{m_{00}}, \quad y_g = \tfrac{m_{01}}{m_{00}}. \tag{5.43}$$
4. **Centered moments**:
   The center of gravity can then be used to define the centered moments:
$$\mu_{ij} = \int\int_{\mathcal{R}} (x - x_g)^i (y - y_g)^j dxdy, \tag{5.44}$$
5. **Normalized centered moments**:
   The normalized centered moments of order two are defined as:
$$n_{ij} = \frac{\mu_{ij}}{a} \tag{5.45}$$
   with
$$\mu_{20} = m_{20} - ax_g^2, \quad \mu_{02} = m_{02} - ay_g^2, \quad \mu_{11} = m_{11} - ax_g y_g.$$

By direct integration along the contour of an ellipse (5.40) the area or zero order moment of an ellipse can be obtained from (5.41) as:
$$m_{00} = \pi k_0 k_1, \tag{5.46}$$
Using (5.43) and (5.42) the first order moments of an ellipse are obtained as:
$$\begin{aligned} m_{10} &= \pi k_0 k_1 x_g, \\ m_{01} &= \pi k_0 k_1 y_g. \end{aligned} \tag{5.47}$$
Using equation (5.45) and (5.44) the normalized centered moments of order two for an ellipse are obtained by direct integration as:
$$\begin{aligned} n_{20} &= \tfrac{(k_0^2 + (k_1 k_2)^2)}{4(1+k_2^2)}, \\ n_{02} &= \tfrac{((k_0 k_2)^2 + k_1^2)}{4(1+k_2^2)}, \\ n_{11} &= \tfrac{k_2(k_0^2 + k_1^2)}{4(1+k_2^2)}, \end{aligned} \tag{5.48}$$
where a mathematic software package has been used in [Chaumette90] to do the integration between defined extremities around the ellipse.

From which the inverse relationships can be determined as:
$$\begin{aligned} k_0^2 &= 2(n_{02} + n_{20} + \sqrt{(n_{20} - n_{02})^2 + 4n_{11}^2}), \\ k_1^2 &= 2(n_{02} + n_{20} - \sqrt{(n_{20} - n_{02})^2 + 4n_{11}^2}), \\ k_2 &= (n_{02} - n_{20} + \sqrt{(n_{20} - n_{02})^2 + 4n_{11}^2})(2n_{11})^{-1} \end{aligned} \tag{5.49}$$

By reinserting (5.49) back into (5.40) the ellipse can be represented by the moments of order less than or equal to two($i + j \leq 2$) giving the parametrization:
$$\begin{aligned} g_{e3}(\mathbf{x}, \boldsymbol{\phi}_{e3}) = \; & n_{02}x^2 + n_{20}y^2 - 2n_{11}xy + 2(n_{11}y_g - n_{02}x_g)x + 2(n_{11}x_g - n_{20}y_g)y \\ & + n_{02}x_g^2 + n_{20}y_g^2 - 2n_{11}x_g y_g + n_{11}^2 - n_{20}n_{02} \end{aligned} \tag{5.50}$$
where g_{e3} is the moment parametrization of an ellipse.

5.7.2 Point-to-Ellipse Algebraic Distance

Using the moment formulation from (5.50), the two-dimensional parametrization of an algebraic point-to-ellipse distance is simply:

$$\begin{aligned} g_{de}(\mathbf{p}, \boldsymbol{\phi}_{de}) = \;& n_{02}x^2 + n_{20}y^2 - 2n_{11}xy + 2(n_{11}y_g - n_{02}x_g)x + 2(n_{11}x_g - n_{20}y_g)y \\ & + n_{02}x_g^2 + n_{20}y_g^2 - 2n_{11}x_gy_g + n_{11}^2 - n_{20}n_{02} - d_e \end{aligned} \tag{5.51}$$

The algebraic point-to-ellipse distance can then be differentiated with respect to time giving:

$$\dot{d}_e = \begin{pmatrix} 2(n_{11}(y-y_g) + n_{02}(x_g - x)) \\ 2(n_{20}(y_g - y) + n_{11}(x - x_g)) \\ ((y-y_g)^2 - n_{02}) \\ 2(y_g(x+x_g) + x_gy + n_{11}) \\ ((x-x_g)^2 - n_{20}) \end{pmatrix}^{\top} \begin{pmatrix} \dot{x_g} \\ \dot{x_g} \\ \dot{n_{20}} \\ \dot{n_{11}} \\ \dot{n_{02}} \end{pmatrix} \tag{5.52}$$

As in the case for the point-to-line distance, the interaction matrix for a point-to-circle distance can be related to that of a circle by the relation $\mathbf{L}_{dc} = \mathbf{L}_{ext}\mathbf{L}_{c3}\mathbf{v}$, where $\mathbf{L}_{ext}$ represents the first term on the right hand side of equation (5.52) and $\mathbf{L}_{c3}$ is the interaction matrix for an ellipse parameterized by moments. The interaction matrix for an ellipse was first derived in [Chaumette90] and later extended to a general class of moments in [Chaumette04]. The derivation proceeds as follows.

According to the general derivation of an interaction matrix the general equation of a moment (5.41) is derived as in equation (5.17) with respect to time. The time derivative of an integral is achieved by using Green's theorem [Stewart03] given as:

$$\dot{m}_{ij} = \int\int_{\mathcal{R}} \left[\frac{\partial f}{\partial x}\dot{x} + \frac{\partial f}{\partial y}\dot{y} + f(x,y)\left(\frac{\partial \dot{x}}{\partial x} + \frac{\partial \dot{y}}{\partial y}\right)\right] dxdy. \tag{5.53}$$

By using the visibility equation (5.38) in (5.13), the different components of (5.53) can be determined as:

$$\begin{aligned} \dot{x} &= -(\alpha x + \beta y + \gamma)v_x + x(\alpha x + \beta y + \gamma)v_z + xy\omega_x - (1+x^2)\omega_y + y\omega_z, \\ \dot{y} &= -(\alpha x + \beta y + \gamma)v_y + y(\alpha x + \beta y + \gamma)v_z + (1+y^2)\omega_x - xy\omega_y + x\omega_z, \\ \tfrac{\partial \dot{x}}{\partial x} &= -\alpha v_x + (2\alpha x + \beta y + \gamma)v_z + y\omega_x - 2x\omega_y, \\ \tfrac{\partial \dot{y}}{\partial y} &= -\beta v_y + (\alpha x + 2\beta y + \gamma)v_z + 2y\omega_x - x\omega_y. \end{aligned} \tag{5.54}$$

Substituting this into equation (5.53) along with the trivial derivations $f(x,y) = x_iy_j$, $\frac{\partial f}{\partial x} = ix^{i-1}y^j$ and $\frac{\partial f}{\partial y} = jx^iy^{j-1}$ the interaction vector for a general moment can be expressed as:

$$\mathbf{L}_{m_{ij}} = \begin{bmatrix} L_{m_{ij},vx} & L_{m_{ij},vy} & L_{m_{ij},vz} & L_{m_{ij},\omega x} & L_{m_{ij},\omega y} & L_{m_{ij},\omega z} \end{bmatrix}, \tag{5.55}$$

where

$$\begin{aligned} L_{m_{ij},vx} &= -i(\alpha m_{ij} + \beta m_{i-1,j+1} + \gamma m_{i-1,j}) - \alpha m_{ij}, & L_{m_{ij},vy} &= -j(\alpha m_{i+1,j-1} + \beta m_{ij} + \gamma m_{i,j-1}) - \beta m_{ij}, \\ L_{m_{ij},vz} &= -(i+j+3)(\alpha m_{i+1,j} + \beta m_{i,j+1} + \gamma m_{ij}) - \gamma m_{ij}, & L_{m_{ij},\omega x} &= (i+j+3)m_{i,j+1} + jm_{i,j-1}, \\ L_{m_{ij},\omega y} &= -(i+j+3)m_{i+1,j} - im_{i-1,j}, & L_{m_{ij},\omega z} &= im_{i-1,j+1} - jm_{i+1,j-1}. \end{aligned}$$

Different interaction vectors for moments of any order can be obtained using this general derivation by just choosing the values of i and j. Thus the different interaction vectors corresponding to the parametrization of an ellipse can be obtained. Firstly, the interaction vector for an area is simply obtained as:

$$\mathbf{L}_a = \begin{bmatrix} -a\alpha & -a\beta & a(3Z_g^{-1} - \gamma) & 3ay_g & -3ax_g & 0 \end{bmatrix}. \tag{5.56}$$

The interaction vectors for the coordinates of the center of gravity are obtained from $x_g = m_{10}m_{00}^{-1}$ and $y_g = m_{01}m_{00}^{-1}$ as:

$$\begin{array}{rcl} \mathbf{L}_{x_g} &=& [\begin{array}{cccccc} -Z_g^{-1} & 0 & x_{g_{vz}} & x_{g_{\omega x}} & x_{g_{\omega y}} & y_g \end{array}], \\ \mathbf{L}_{y_g} &=& [\begin{array}{cccccc} 0 & -Z_g^{-1} & y_{g_{vz}} & y_{g_{\omega x}} & y_{g_{\omega y}} & -x_g \end{array}], \end{array} \tag{5.57}$$

where

$$\begin{array}{lll} x_{g_{vz}} = \frac{x_g}{Z_g} + 4(\alpha n_{20} + \beta n_{11}), & y_{g_{vz}} = \frac{y_g}{Z_g} + 4(\alpha n_{11} + \beta n_{02}), & x_{g_{\omega x}} = y_{g_{\omega y}} = x_g y_g + 4n_{11}, \\ x_{g_{\omega y}} = -(1 + x_g^2 + 4n_{20}), & y_{g_{\omega x}} = -(1 + y_g^2 + 4n_{02}), & \end{array}$$

where n_{00}, n_{02} and n_{11} are the second-order, normalized, centered moments given in equation (5.48). Finally the interaction vectors for the normalized centered moments can be obtained by deriving (5.48) as:

$$\mathbf{L}_{n_{ij}} = \frac{\mu_{ij}\mathbf{L}_a - a\mathbf{L}_{\mu_{ij}}}{a^2} = \frac{\mathbf{L}_{\mu_{ij}}}{a} - \frac{n_{ij}\mathbf{L}_a}{a}. \tag{5.58}$$

In order to determine this it is necessary to determine the interaction for the centered moments. The centered moments are derived according to Green's theorem as was done in the case of non-centered moments (5.54):

$$\begin{array}{rcl} \dot{\mu}_{ij} = & \int\int_{\mathcal{R}} & \Big[i(x - x_g)^{i-1}(y - y_g)^j(\dot{x} - \dot{x}_g) \\ & & + j(x - x_g)^i(y - y_g)^{j-1}(\dot{y} - \dot{y}_g) \\ & & + (x - x_g)^i(y - y_g)^j\left(\frac{\partial \dot{x}}{\partial x} + \frac{\partial \dot{y}}{\partial y}\right)\Big]\, dxdy \end{array} \tag{5.59}$$

Using equation (5.54) and the interaction vectors for the center of gravity (5.57), the general interaction vector for centered moments is obtained from (5.59) as:

$$\mathbf{L}_{\mu_{ij}} = \begin{bmatrix} L_{\mu_{ij},vx} & L_{\mu_{ij},vy} & L_{\mu_{ij},vz} & L_{\mu_{ij},\omega x} & L_{\mu_{ij},\omega y} & L_{\mu_{ij},\omega z} \end{bmatrix} \tag{5.60}$$

$$\begin{array}{l} L_{\mu_{ij},vx} = -(i+1)\alpha\mu_{ij} - i\beta\mu_{i-1,j+1}, \\ L_{\mu_{ij},vy} = -j\alpha\mu i + 1, j - 1 - (j+1)\beta\mu_{ij}, \\ L_{\mu_{ij},vz} = -\alpha L_{\mu_{ij},\omega y} + \beta L_{\mu_{ij},\omega x} + (i+j+2)\gamma\mu_{ij}, \\ L_{\mu_{ij},\omega x} = (i+j+3)\mu_{i,j+1} + ix_g\mu_{i-1,j+1} + (i+2j+3)y_g\mu_{ij} - 4in_{11}\mu_{i-1,j} - 4jn_{02}\mu_{i,j-1}, \\ L_{\mu_{ij},\omega y} = -(i+j+3)\mu_{i+1,j} - (2i+j+3)x_g\mu_{ij} - jy_g\mu_{i+1,j-1} + 4in_{20}\mu_{i-1,j} + 4jn_{11}\mu_{i,j-1}, \\ L_{\mu_{ij},\omega z} = i\mu_{i-1,j+1} - j\mu_{i+1,j-1}. \end{array}$$

Using the centered moment interaction vector (5.60), the interaction vector for an area (5.56), the area (5.42), the normalized moment (5.48) and knowing that the third order moments are all equal to zero for an ellipse$(\mu_{30}, \mu_{21}, \mu_{12}, \mu_{03} = 0)$ then the interaction vectors for the normalized centered moments are:

$$\begin{array}{rcl} \mathbf{L}_{n_{20}} &=& [\begin{array}{cccccc} L_{n_{20},vx} & 0 & L_{n_{20},vz} & L_{n_{20},\omega x} & L_{n_{20},\omega y} & 2n_{11} \end{array}], \\ \mathbf{L}_{n_{02}} &=& [\begin{array}{cccccc} 0 & L_{n_{02},vy} & L_{n_{01},vz} & L_{n_{02},\omega x} & L_{n_{02},\omega y} & -2n_{11} \end{array}], \\ \mathbf{L}_{n_{11}} &=& [\begin{array}{cccccc} L_{n_{11},vx} & L_{n_{11},vy} & L_{n_{11},vz} & L_{n_{11},\omega x} & L_{n_{11},\omega y} & n_{02} - n_{20} \end{array}], \end{array} \tag{5.61}$$

where

$$\begin{array}{ll} L_{n_{20},vx} = -2(\alpha n_{20} + \beta n_{11}), & L_{n_{02},vy} = -2(\alpha n_{11} + \beta n_{02}, \\ L_{n_{11},vx} = -\alpha n_{11} - \beta n_{02}, & L_{n_{11},vy} = -\alpha n_{20} - \beta n_{11}, \\ L_{n_{ij},vz} = -\alpha L_{n_{ij},\omega y} + \beta L_{n_{ij},\omega x} + 2\gamma n_{ij}, & L_{n_{20},\omega x} = 2(y_g n_{20} + x_g n_{11}), \\ L_{n_{20},\omega y} = -4x_g n_{20}, & L_{n_{02},\omega x} = 4y_g n_{02}, \\ L_{n_{02},\omega y} = -2(y_g n_1 1 + x_g n_{02}), & L_{n_{11},\omega x} = 3y_g n_1 1 + x_g n_{02}, \\ L_{n_{11},\omega y} = -y_g n_{20} - 3x_g n_{11}. & \end{array}$$

Finally leading to the interaction vector for a point-to-circle distance:

$$\mathbf{L}_{de} = \begin{bmatrix} 2(n_{11}(y-y_g) + n_{02}(x_g - x)) \\ 2(n_{20}(y_g - y) + n_{11}(x - x_g)) \\ ((y-y_g)^2 - n_{02}) \\ 2(y_g(x+x_g) + x_g y + n_{11}) \\ ((x-x_g)^2 - n_{20}) \end{bmatrix}^{\top} \begin{bmatrix} \mathbf{L}_{x_g} \\ \mathbf{L}_{y_g} \\ \mathbf{L}_{n_{20}} \\ \mathbf{L}_{n_{02}} \\ \mathbf{L}_{n_{11}} \end{bmatrix} \tag{5.62}$$

5.7.3 Point-to-Ellipse Geometric Distance

As with the case of a distance between a point and a line it is also possible to consider the case of a directional distance between two points. In this case sampling is performed in 3D and it is necessary to determine the angle of the contour normal in the image. As will be seen in this section, the use of a directional distance between points has certain advantages. More particularly, it allows the geometric distance between a point and an ellipse to be considered as opposed to the algebraic distance, given in the previous section, of a point to an ellipse.

In the case of the geometric distance, the distance model is based on the orthogonal distance between a point and the ellipse (see Figure 5.4). In this case there is no high curvature bias and Euclidean distances are preserved. The orthogonal distance is defined as the distance between a point and the closest corresponding point on the ellipse. It has been shown in [Zhang97] that to solve analytically for the distance it is necessary to formulate the distance in terms of the tangent of the ellipse so that the tangent is orthogonal to the line joining the point p^* to the closest point on the ellipse p_e. In this case, the computational efficiency is lost because there are two possible solutions to this equation.

The directional point-to-point distance formulation, however, allows to model the geometric distance between a point and an ellipse without the need to resolve the two solutions for the analytic formulation of the geometric point-to-ellipse distance. The different effects of the estimation bias introduced by the algebraic distance between a point and an ellipse, when compared to the geometric distances, has been investigated in depth in [Zhang97]. In the majority of cases, the major advantage for using the algebraic distance over the geometric distance is its computational efficiency. On the other hand, the disadvantages are, firstly, that the algebraic distance introduces bias in regions of high curvature. Secondly, the distance is usually not invariant under Euclidean transformations.

In the case presented here, the multiple correspondence solutions found when tracing a line between a point and orthogonal points on an ellipse can be avoided if the corresponding point is already known. This is the case here because it has been sampled so as to make a search to the normal of the contour. Thus as was done in the case of a line, the interaction vector for a point-to-ellipse distance can be considered as the distance between a point and a tangent to the ellipse.

The geometric distance between a point and the closest point on an ellipse is therefore considered as a directional distance as in the case of a line. The first step is to define the 3D parametric equation (3.13) of the object model, which in this case is simply a 3D point. The visibility equation (5.14) is simply the depth of the point $Z = Z_0$. The 2D parametric equation (5.15) is again simply a 2D point. Therefore, it remains to define a distance in 2D. This can be done by projecting each x and y coordinate's distance in the direction of the normal as:

$$g_{de2}(\mathbf{p}, \boldsymbol{\phi}) = (x - x^*)\cos\theta_{e2} + (y - y^*)\sin\theta_{e2} - d_{e2} = 0. \tag{5.63}$$

where d_e2 is the geometric distance between the two points in the image and θ is the angle of the tangent.

Deriving (5.63), the point-to-ellipse distance (5.51) becomes:

$$\begin{aligned} \dot{d}_{e2} &= \tfrac{\partial d_{e2}}{\partial x}\dot{x} + \tfrac{\partial d_{e2}}{\partial y}\dot{y} \\ &= (2\mu_{02}x - 2\mu_{11}y + 2(\mu_{11}y_g - \mu_{02}x_g))\,\dot{x} + (2\mu_{20}y - 2\mu_{11}x + 2(\mu_{11}x_g - \mu_{20}y_g))\,\dot{y}. \end{aligned} \tag{5.64}$$

Since the angle of the tangent, given a point on the ellipse p_e , can be determined as:

$$\theta_{e2} = \arctan\frac{dy}{dx}, \tag{5.65}$$

then it can be shown that:

$$\tan\theta_{e2} = \frac{dy}{dx} = \frac{(2\mu_{20}y - 2\mu_{11}x + 2(\mu_{11}x_g - \mu_{20}y_g))}{(2\mu_{02}x - 2\mu_{11}y + 2(\mu_{11}y_g - \mu_{02}x_g))} = \frac{\sin\theta_{e2}}{\cos\theta_{e2}} \tag{5.66}$$

and the velocity of a distance is equivalent to the simplified version of a line (5.33).

The new interaction matrix is thus obtained by considering $L_{d_{e2}} = \cos\theta_{e2}\mathbf{L}_x + \sin\theta_{e2}\mathbf{L}_y$ as:

$$\boxed{L_{d_{e2}} = \begin{pmatrix} -Z^{-1}\cos\theta_{e2} \\ -Z^{-1}\sin\theta_{e2} \\ Z^{-1}x\cos\theta_{e2} + Z^{-1}y\sin\theta_{e2} \\ xy\cos\theta_e + (1+y^2)\sin\theta_{e2} \\ -(1+x^2)\cos\theta_{e2} - xy\sin\theta_{e2} \\ y\cos\theta_e - x\sin\theta_{e2} \end{pmatrix}, \qquad (5.67)}$$

5.8 Point-to-Projected Volumetric Surface Distance

In this section a 3D volumetric surface is considered. In the previous section the distance between a point and the projection of a circle was considered. A 3D circle was represented by the intersection of a plane with a sphere. Thus the interaction matrix for a 3D sphere can be easily derived in a similar way. For the interested reader, the complete derivation for a sphere has been given in [Chaumette90, Espiau92]. In order to study the distance between a point and the projection of a volumetric surface, the example of a cylinder will be considered here. A cylinder can be defined by a set of circles with a radius R whose center lies on a line $\mathcal{L}$ which is perpendicular to the plane of each circle.

First of all, a circle is defined as the intersection of a sphere with a plane as in equation (5.37):

$$\begin{aligned} h_{c1}(\mathbf{P}, \mathbf{\Phi}_{c1}) &= (X - X_i)^2 + (Y - Y_i)^2 + (Z - Z_i)^2 - R^2 = 0, \\ h_{c2}(\mathbf{P}, \mathbf{\Phi}_{c2}) &= A(X - X_i) + B(Y - Y_i) + C(Z - Z_i) = 0, \end{aligned} \tag{5.68}$$

where $\mathbf{P}_i = (X_i, Y_i, Z_i)$ is the center of the circle.

To define the constraint that the center of a circle belongs to a line axis, $\mathcal{L}$, the parametric equation for a line is written as:

$$\begin{array}{rcl} X_i & = & X_0 + A\lambda, \\ Y_i & = & Y_0 + B\lambda, \quad \forall \mathbf{P}_i \in \mathcal{L} \quad \text{and} \quad \forall \lambda \in \mathcal{R}, \\ Z_i & = & Z_0 + C\lambda, \end{array} \tag{5.69}$$

where (X_0, Y_0, Z_0) is a reference point on the line and all (X_i, Y_i, Z_i) are the points belonging to the axis of the cylinder.

In order to solve for lambda, the parametric equation for a line (5.69) is first substituted into (5.68):

$$\begin{array}{ll} \lambda^2 - 2\lambda[A(X - X_0) + B(Y - Y_0) + C(Z - Z_0)] & \\ +(X - X_0)^2 + (Y - Y_0)^2 + (Z - Z_0)^2 - R^2 & = 0. \end{array} \tag{5.70}$$

It is now possible to solve for λ and by choosing $A^2 + B^2 + C^2 = 1$ this simplifies to:

$$\lambda = A(X - X_0) + B(Y - Y_0) + C(Z - Z_0). \tag{5.71}$$

The three-dimensional parametric equation for a cylinder is then determined as:

$$\begin{array}{rl} h_{cyl}(\mathbf{P}, \phi_{cyl}) = & (X - X_0)^2 + (Y - Y_0)^2 + (Z - Z_0)^2 \\ & -[A(X - X_0) + B(Y - Y_0) + C(Z - Z_0)]^2 - R^2 = 0. \end{array} \tag{5.72}$$

The perspective projection equation (5.7) is used to project the cylinder onto the image. When factorized as a polynomial in terms of Z^{-1} this gives:

$$\begin{array}{rl} Z^{-2} & \Big((X_0^2 + Y_0^2 + Z_0^2 - (AX_0 + BY_0 + CZ_0)^2 - R^2\Big) \\ - & 2Z^{-1}\Big((1 - A^2)X_0 - ABY_0 - ACZ_0)X - (ABX_0 - (1 - B^2)Y_0 + BCZ_0)Y \\ & -ACX_0 - BCY_0 + (1 - C^2)Z_0\Big) \\ + & \Big(X^2 + Y^2 + 1 - (AX + BY + C)^2\Big) = 0, \end{array} \tag{5.73}$$

where it can be seen that the projection of the limbs of a cylinder in the image degenerates when the camera center of projection is on the surface of the cylinder $((X_0^2 + Y_0^2 + Z_0^2) = (AX_0 + BY_0 + CZ_0)^2 + R^2))$ or when the center of projection is inside the cylinder $((X_0^2 + Y_0^2 + Z_0^2) - (AX_0 + BY_0 + CZ_0)^2 - R^2 < 0)$. Therefore, only the non-degenerate cases are considered.

In order to determine the portion of the projected cylinder, which is seen in the image, it is first noted that the contours of the cylinder which intersect with the image plane are unique (as in the case of a line). The solutions of equation (5.73) can be found using the discriminant Δ:

$$\Delta = b^2 - 4ac = 0, \tag{5.74}$$

where, a and b are respectively the coefficients of Z^{-2} and Z^{-1} in equation (5.73), and c is the last term.

The visibility equation is then easily determined by noting that $Z^{-1} = b(2a)^{-1}$ giving:

$$\begin{array}{rl} \frac{1}{\mu(x,y,\Phi_{cyl})} = \frac{1}{Z} = & \frac{1}{a}\Big((1 - A^2)X_0 - ABY_0 - ACZ_0)X \\ & +(ABX_0 - (1 - B^2)Y_0 + BCZ_0)Y \\ & -ACX_0 - BCY_0 + (1 - C^2)Z_0\Big) \end{array} \tag{5.75}$$

Furthermore, solving the determinant in equation (5.74) allows two solutions to be obtained which correspond to two lines. These two lines are the limbs of the projected cylinder and can be written in ρ, θ format so as to obtain the following interaction matrix:

$$\begin{array}{lcl} L_{\rho_1} & = & [\lambda_{\rho_1}\cos\theta_1 \quad \lambda_{\rho_1}\sin\theta_1 \quad -\lambda_{\rho_1}\rho_1 \quad (1+\rho_1^2)\sin\theta_1 \quad -(1+\rho_1^2)\cos\theta_1 \quad 0], \\ L_{\theta_1} & = & [\lambda_{\theta_1}\cos\theta_1 \quad \lambda_{\theta_1}\sin\theta_1 \quad -\lambda_{\theta_1}\rho_1 \quad -\rho_1\cos\theta_1 \quad -\rho_1\sin\theta_1 \quad -1], \\ L_{\rho_2} & = & [\lambda_{\rho_2}\cos\theta_2 \quad \lambda_{\rho_2}\sin\theta_2 \quad -\lambda_{\rho_2}\rho_2 \quad (1+\rho_2^2)\sin\theta_2 \quad -(1+\rho_2^2)\cos\theta_2 \quad 0], \\ L_{\theta_2} & = & [\lambda_{\theta_2}\cos\theta_2 \quad \lambda_{\theta_2}\sin\theta_2 \quad -\lambda_{\theta_2}\rho_2 \quad -\rho_2\cos\theta_2 \quad -\rho_2\sin\theta_2 \quad -1] \end{array} \qquad (5.76)$$

with $\lambda_{\rho_i} = (A_2\rho_i\cos\theta_i + B_2\rho_i\sin\theta_i + c_2)D_2^{-1}$ and $\lambda_{\theta_i} = (A_2\sin\theta_i - B_2\cos\theta_i)D_2^{-1}$.

Using (5.76) and considering that $\mathbf{L}_{dl} = \mathbf{L}_p + \alpha\mathbf{L}_\theta$ as was the case for the interaction matrix for a point-to-line distance, the distance between a point and the projection of the limb surface of the cylinder is finally deduced as:

$$L_{dl,i} = \begin{bmatrix} \lambda_{d_l,i}\cos\theta_i \\ \lambda_{d_l,i}\sin\theta_i \\ \lambda_{d_l,i}\rho_i \\ (1+\rho_i^2)\sin\theta_i - \alpha\rho_i\cos\theta_i \\ -(1+\rho^2)\cos\theta_i - \alpha\rho\sin\theta_i \\ -\alpha \end{bmatrix}^\top, \qquad (5.77)$$

where $\lambda_{d_l,i} = \lambda_{\rho_i} + \alpha\lambda_{\theta_i}$.

5.9 Results

In the experiments presented,"real" images are acquired using a commercial digital camera. Various augmented reality experiments which are based on the 3D tracking approach are reported. The goal, in these results, is not to build a complete AR-based scenario but just to illustrate the fact that the proposed tracker is robust enough to provide the necessary information to build such a system. Tracking is always performed at below frame rate. In all the reported experiments, the tracking and 3D localization is achieved at 50 Hz on a simple PC (3.0 GHz) running Window XP. Other experiments in a robotic context have been reported in [Comport04c].

The 3D model-based tracking has been implemented as a software platform known as the Augmented Reality Platform (ARP). The marker-less tracking component of this software is protected under the French APP protection laws and is submitted under the name Marker-less (Markerless-based Augmented Reality KERnel). This software environment has been implemented in C++ under both Linux and Windows XP. It has been integrated into D'Fusion, a software developed by Total-Immersion Inc.

In the following experiments, the image processing is potentially very complex. Indeed extracting and tracking reliable points in real environment is a non trivial issue. Avoiding the constraints introduced by using markers in the environment is an important step toward the industrialization of augmented reality systems. A large number of 3D localization and tracking algorithms are available in the literature (see Section 5.2). However, they are not always suitable for augmented reality applications for various reasons (reliability, computation time, ...). Even so, some marker-less tracking algorithms have been already used in this context [Simon99, Lepetit03, Klein03, Comport03a].

In this section, results will be presented for various types of visual primitives. The use of more complex features such as the distance of points to the projection of 3D circles, lines, and cylinders is demonstrated.

In all experiments, the distances are computed using the Moving Edges algorithm previously described. It will be shown that different types of visual features, as well as the amount of visual information, has a direct impact on the stability of our method. For example, enough visual features to constrain the 6 degrees of freedom is first necessary. The greater the information for each d.o.f. the greater the stability of the algorithm with respect to noise in the measurements.

Tracking with different contours. In this experiment the result of tracking four 3D circles is shown. This long sequence (more than 2000 images) tests both translational and rotational movement of the camera. Some of the circles partially leave the image and return into view without any affect on the tracking of the algorithm.

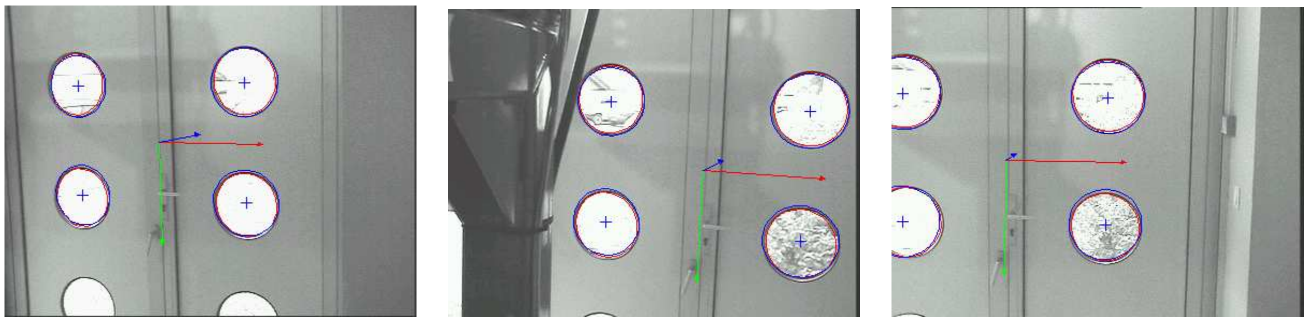
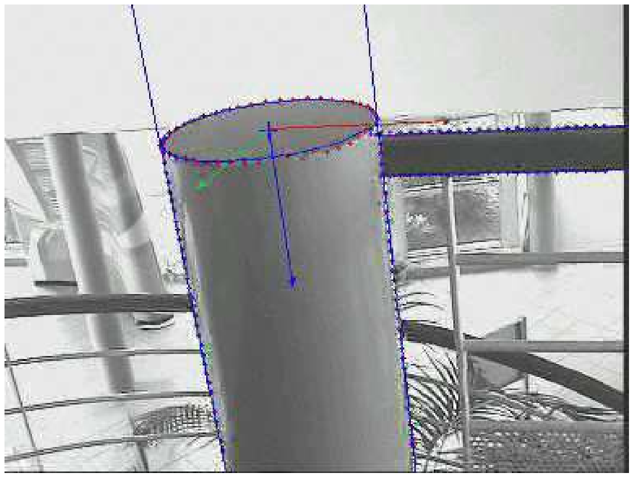
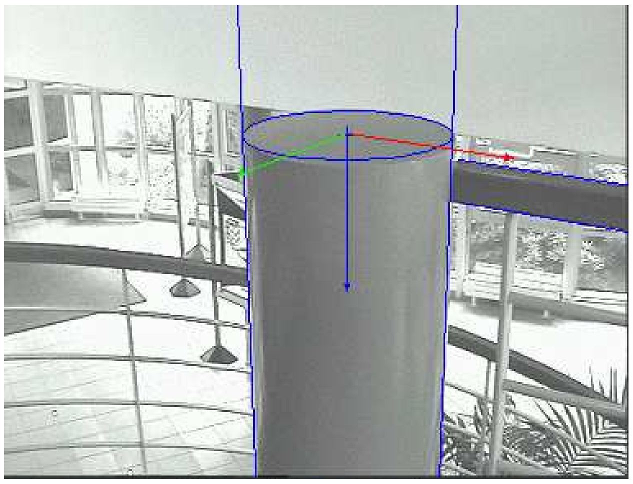


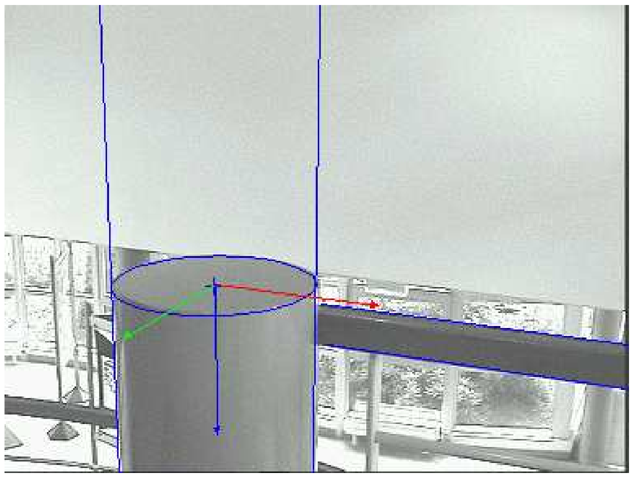
Figure 5.5: Tracking 3D circles. Four circles are tracked along a 2000 images sequence. Some of the circles partially leave the image.

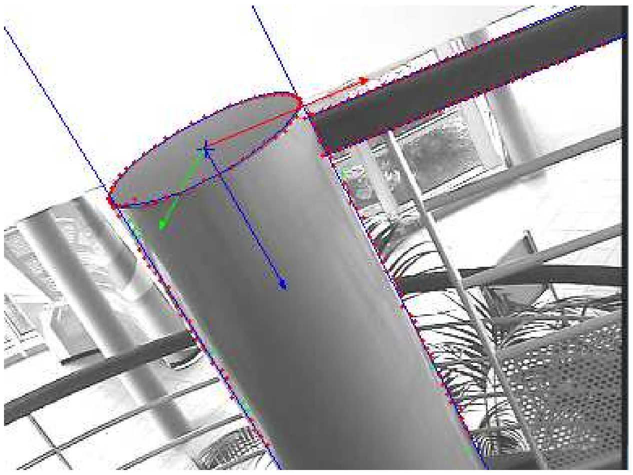
In the second experiment an object whose model is composed of a cylinder, a circle and two lines is considered (see Figure 5.6). This illustrates the capabilities of the tracking algorithm to consider various different features within the same minimization process.

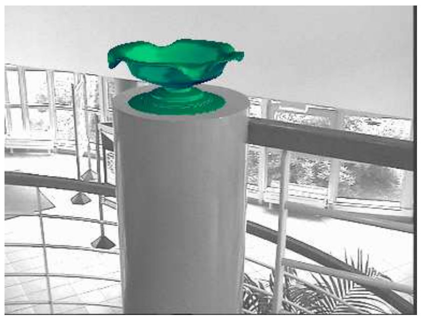
It can be seen from the graphs in Figure 5.7 that the estimation of the pose takes only 6 iterations which is fast enough for tracking to be achieved at camera frame-rate. The maximum velocity of the camera relative to the object can be estimated from Figure 5.7 as $0.08 * 30 = 2.4m/s$ where camera frame-rate is taken as 30 frames per second. Note that this is only a relative velocity and it is not the instantaneous velocity of the object or the camera. It can be seen that the error norm is maintained at around 1.2 mm in the image. The trajectory of the camera and the object are clearly seen in the pose plots. Furthermore, at around iteration 175 the bottom part of the CAD model starts to leave the image and continues to leave the image up until iteration 300. At this point the object returns gradually into the image up until iteration 350. It can be seen when referring to 5.7(d), that this correlates with the increase in the error norm, indicating the sensitivity of our algorithm with respect to the amount and type of visual information within the image.

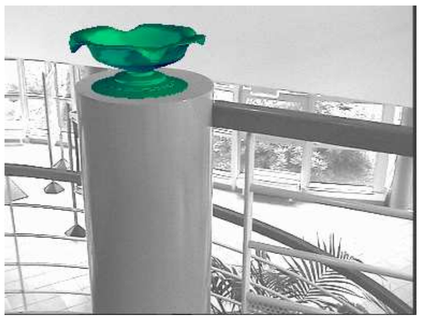

(a)


(b)


(c)


(d)


(e)


(f)

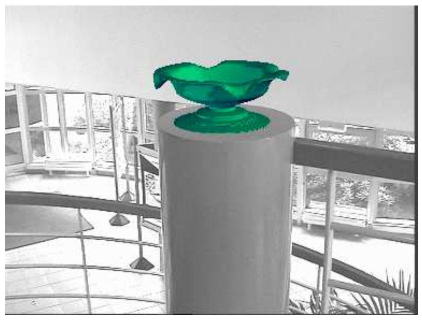
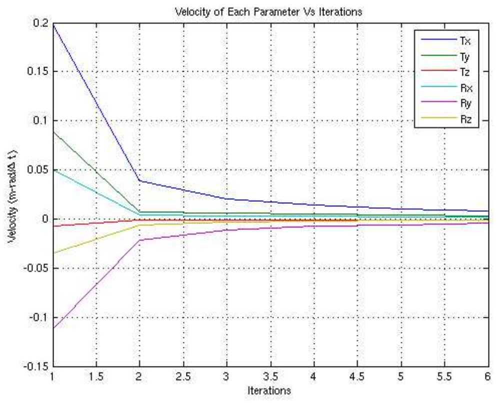
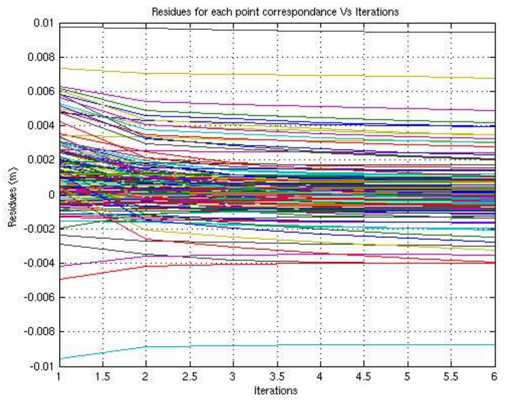

(g)

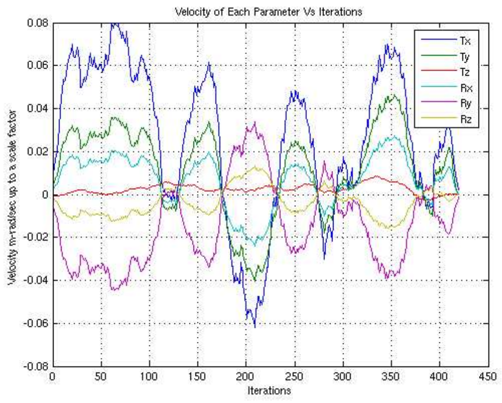
Figure 5.6: Tracking considering a circle, a cylinder and two straight lines. (a-d) The projection of the CAD model in blue and the points tracked along the normal in red. Large rotational movements are made and parts of the object leave the image. (e-f) The resulting augmented reality sequence using the 3D pose estimated from the tracker to project virtual objects onto the image.



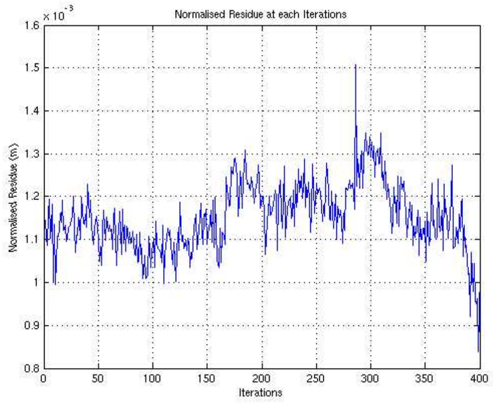
(a)



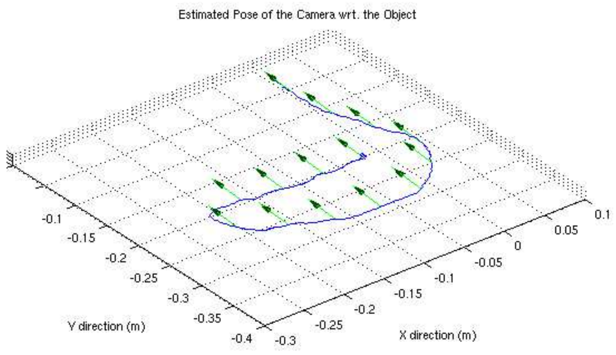
(b)



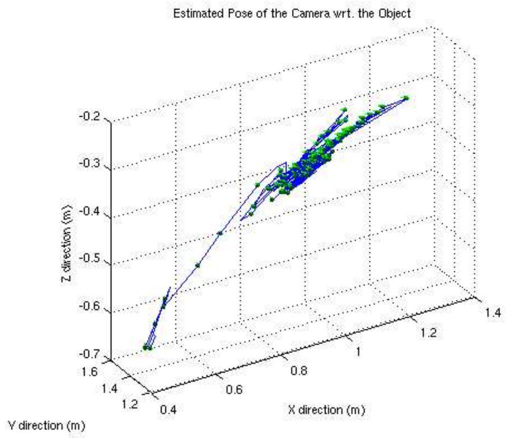
(c)



(d)



(e)



(f)

Figure 5.7: Analysis of the tracking of a circle, a cylinder and two straight lines. (a,b) The convergence of the Virtual Visual Servoing control law on the first iteration of the sequence for the virtual velocity and visual errors respectively. It can be seen that the number of iterations to converge upon an error of $10e^{-}5$ is 6 which has been obtained for a gain of $\lambda = 0.7$. Some of the errors remain large because they have been rejected by a robust estimator treated in Chapter 6. (c) the estimated velocity of the camera at each image, (d) the error norm at each image in the sequence, (e) the trajectory of the camera relative to the object and (f) the trajectory of the object relative to itself.

5.9.1 Tracking in an Augmented Reality Scenario

In this next example (Figure 5.8) the 3D tracking algorithm is used to track a domestic or industrial object that is the subject of a maintenance operation. In this case it is an electrical panel, however, many other manufactured objects could be considered using the same technique. Within the experiment, large motions of the object are made along with large camera motions. Forward projection of the model appears in blue and the image of the object is augmented with various textual information which is positioned in alignment with the object.

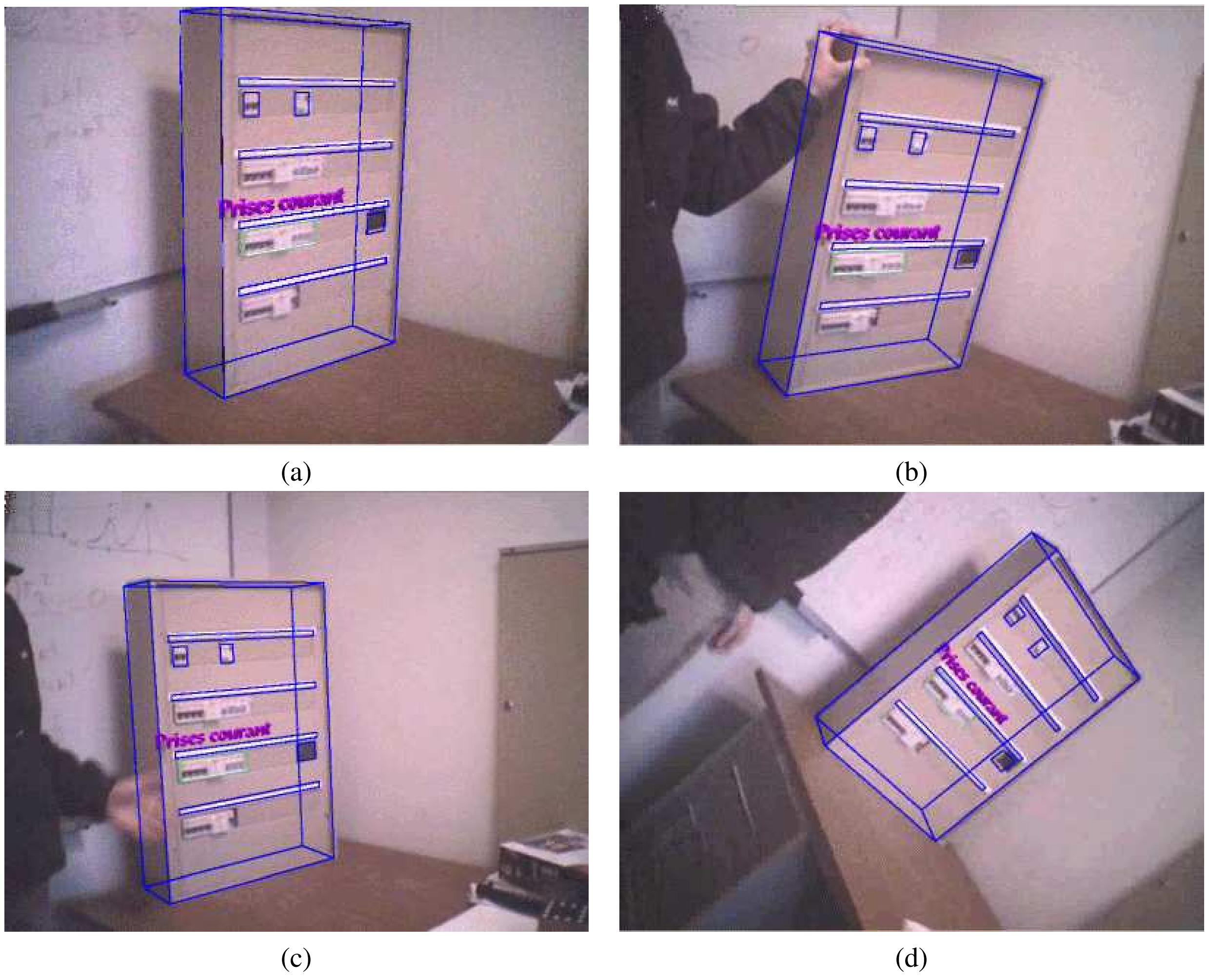
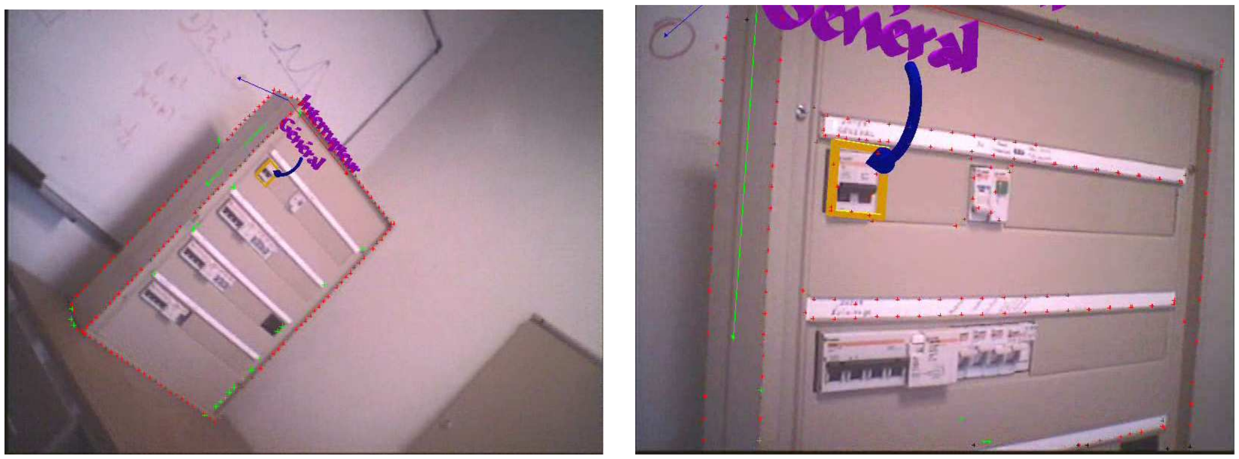


Figure 5.8: Tracking of an electrical panel with augmented text and the projection of the CAD model in blue (a). The sequence features large motion of the object (b), partial occlusions (c), and large motion of the camera(d).

The model of the object is represented as a hierarchical scene graph (encoded in VRML) that allows online modification of the model used in the tracking process. Since the relative position between the camera and the tracked object is subject to large movements, it is indeed necessary to automatically select which features to use online. Indeed when the camera moves away from the object, some details become too small to be useful for tracking (or may even induce miss-tracking). In Figure 5.9 the white bands (modeled by two lines in the CAD) are used when the camera is close from the object (see the red points on the right image and on the video), while when the camera moves away from the object (left image), these two lines are too close each another to provide reliable information. Finally, in order to maintain tracking at video rate, the number of tracked points is restricted to be lower than 400 (**L** is then a 400×6 matrix). This is achieved by constantly modifying the sampling density of points along the lines of the model.



(a) (b)

Figure 5.9: Tracking of an electrical panel: illustration of the dynamic modification of the model whereby some parts are added or removed according to the relative camera/object position. (a) Only the outline of the box is necessary when far from the object, (b) More detail is useful closer to the object since many primitives are no longer within the view of the camera and able to constrain the pose.

5.9.2 High Velocity Tracking and Modeling Error

In Figure 5.10 a chair is tracked and a virtual Lego-man is augmented in the view. The model of the chair is very rough. Each leg of the chair is only modeled by a simple line in the VRML model (and the back of the chair by a simple polygon). The motion of the chair in the video stream was very fast.

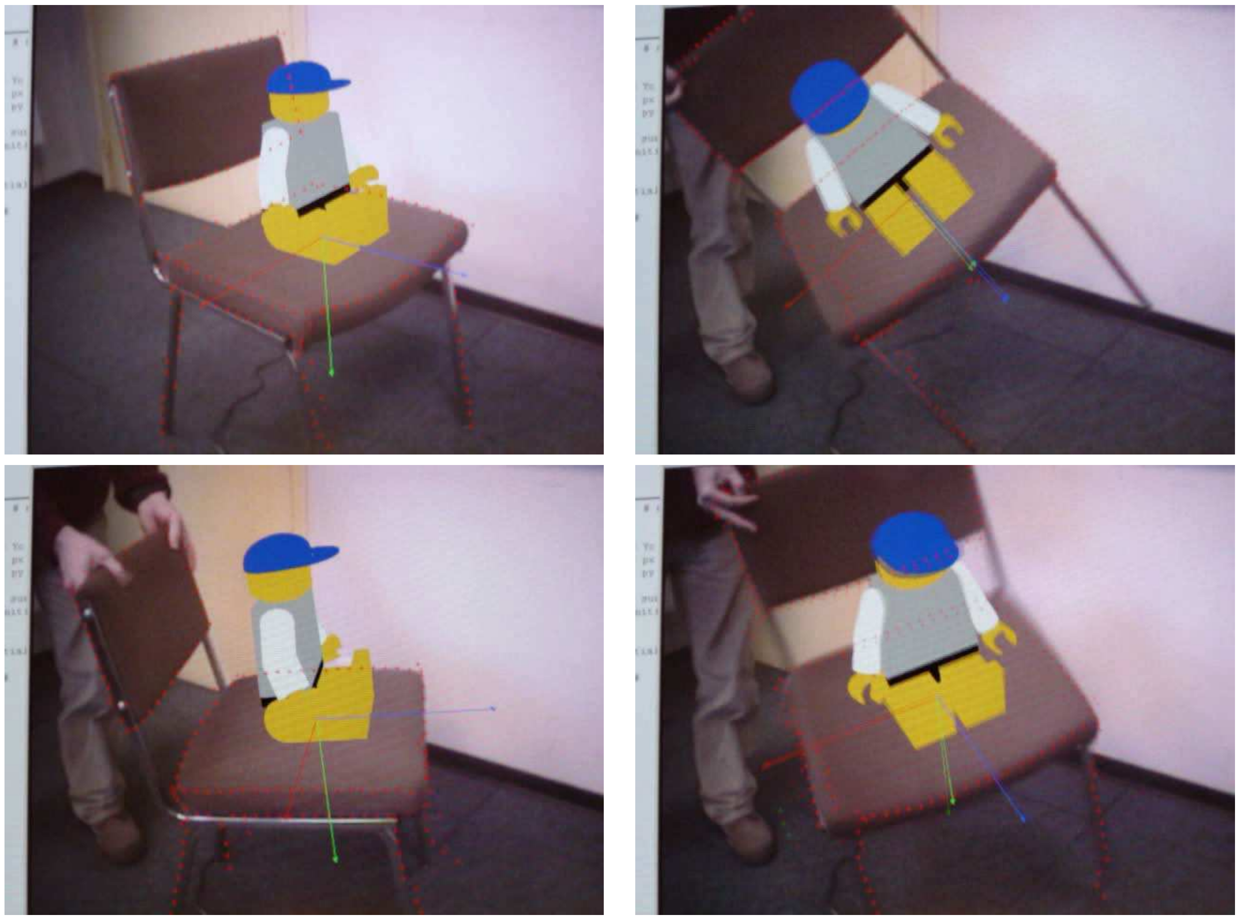


Figure 5.10: The Lego-man experiment: tracking a chair. This experiment illustrates the robustness of the algorithm with respect to rough CAD model as well as to fast object motion.

5.9.3 Environment Tracking

In this experiment an entire conference room was modeled. Tracking starts from one corner of the room and the camera pans around the room. In this way the camera completely leaves its initial position an returns to its initial position after a long period of time without any accumulation of tracking error.

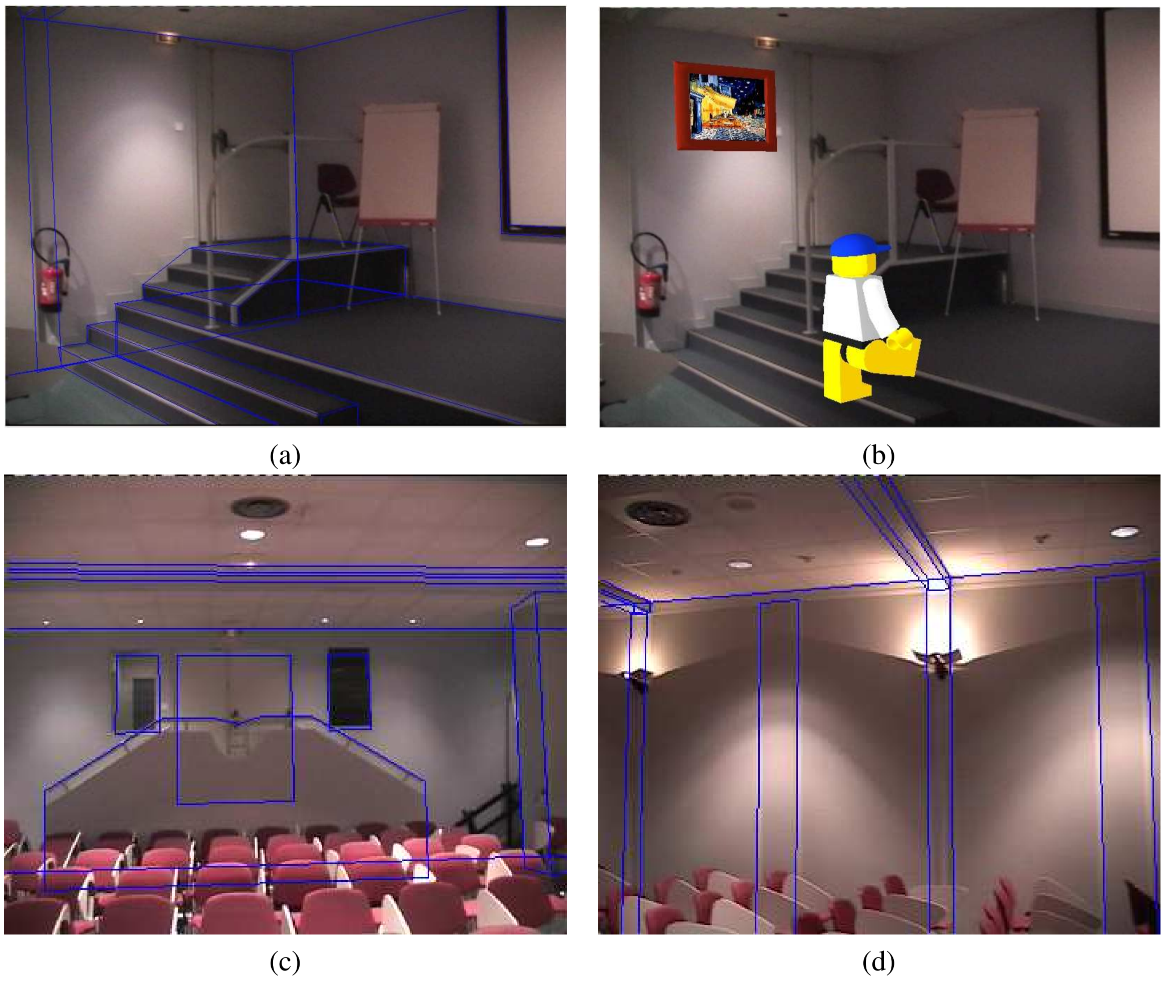


Figure 5.11: Tracking and augmenting a conference room. The camera pans around the room and returns to the initial position without any accumulated error. (a) The initial position, (b) the augmented room, (c,d) intermediary positions with the CAD model displayed.

5.9.4 Interactive Augmented Reality

In this section two augmented reality experiments are presented involving interaction between the real world and the virtual world. In Figure 5.12 the 3D tracker proposed in this thesis has been used to track a castle ruin (the tracked object and the lines used in the tracking are shown in red). A virtual car model can then be rendered and its motion is handled using an animation engine (ODE). The car is subject to gravity and to collision with the ruin model. The ruin model moves accordingly to the real ruin motion and the car interacts with its environment in realistic way.

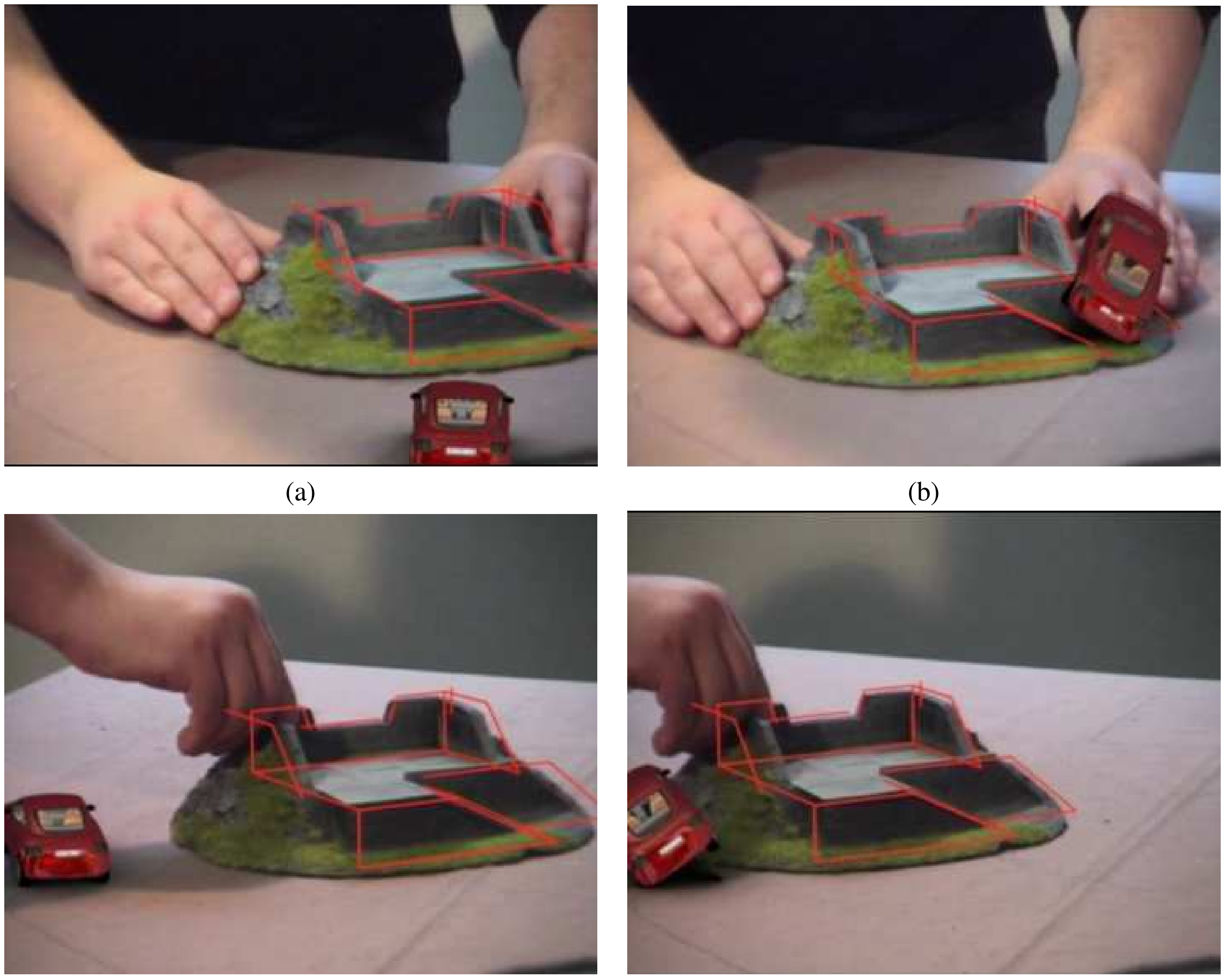


Figure 5.12: Interactive augmented reality based on tracking the ruins of a castle. The object model contours used for tracking are displayed in red. A remote-control virtual car is augmented into the video stream. The user can control the virtual car using a game controller. The car is able to interact with the real environment thanks to the tracking algorithm combined with an animation engine (ODE). [1]

[1]Images obtained in collaboration with Total-Immersion

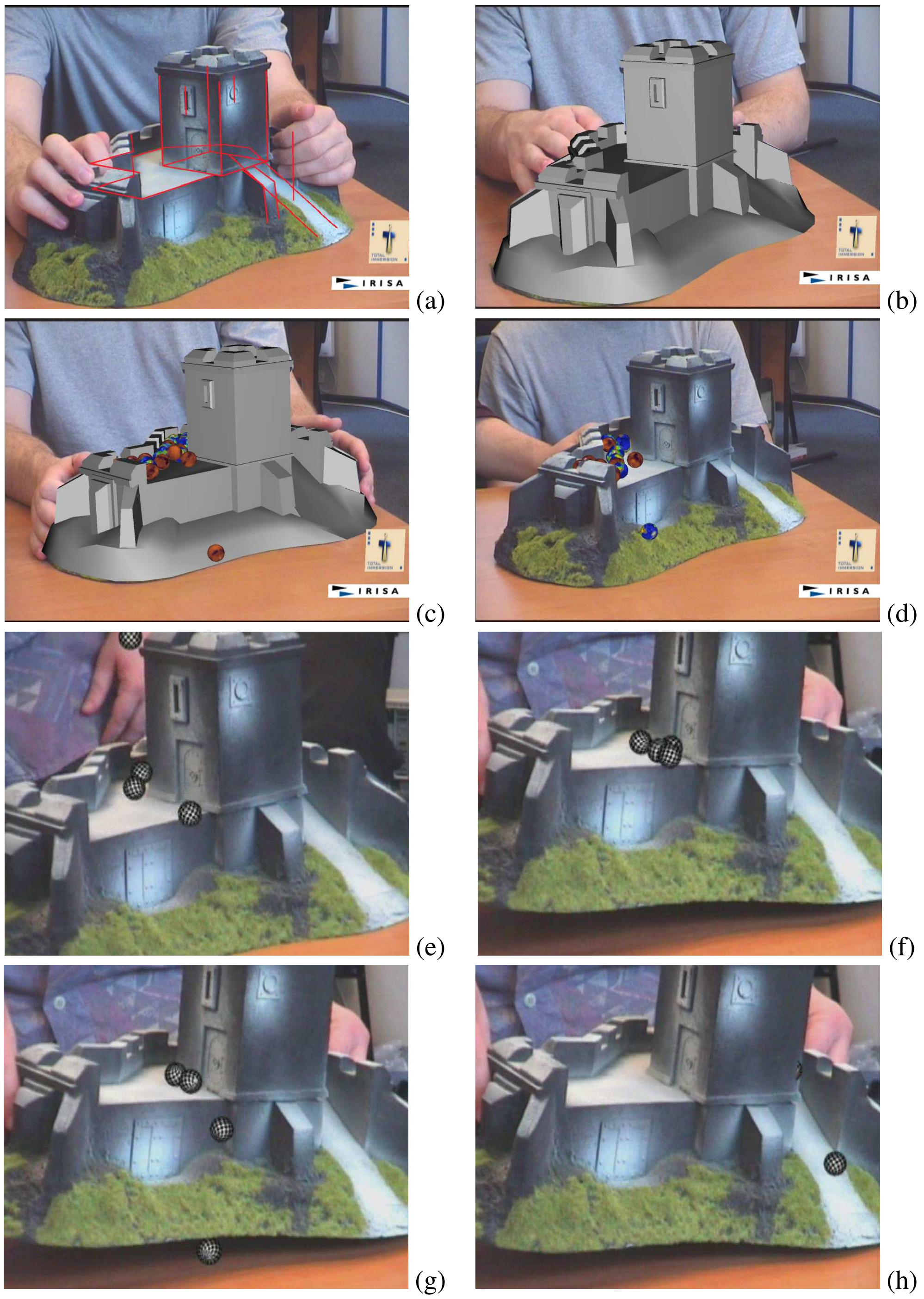


Figure 5.13: Interactive augmented reality based on 1D/3D tracking. The algorithm tracks the castle while the motion of the balls is handled by a dynamic engine. (a) real castle and tracked features in red, (b) castle model (c) castle model with virtual ball, (d) final results, (e-h) four images acquired during the experiment.[2]

Another interactive augmented reality game is shown in Figure 5.13 where the 3D tracker has been used

[2]Images obtained in collaboration with Total-Immersion

to track an entire castle. In Figure 5.13, the tracked object and the lines used in the tracking are shown. An augmented model can then be rendered (Figure 5.13b) along with some interactive balls (Figure 5.13c) which are handled using an animation engine (ODE). The balls are subject to gravity and to collision with the castle model and the other balls. The castle model moves accordingly to the real castle motion and the balls move in realistic way. Figures 5.13e-h show four images of the final result.

5.10 Conclusion and Perspectives

In conclusion, different distance-to-contour based features have been modeled and verified experimentally. In particular distances to points, lines, circles and cylinders have been considered. The results have proven to be accurate and computationally efficient. This work has laid the basis for further investigation into 3D tracking. With respect to another recent method in the literature [Drummond02], a theoretical comparison has been made and a unifying framework has been given.

Nevertheless, it would be interesting to consider deriving interaction matrices for various other types of visual information including planar features and texture such as in [Pressigout04]. It could also be interesting to continue the development of interaction matrices for more complex contours modeled by B-Splines or Nurbs [Pressigout04]. More complex 3D models could also be integrated into such a framework by using super-quadrics [Terzopoulos91, Bardinet94, Leonardis97]. Indeed, it would be interesting to take advantage of the global convergence properties of moment based interaction matrices [Tahri05], however, it would be necessary to develop this theory to include 3D moments and not only 2D moments.

Another perspective is to render this type of tracking technique more automatic by considering both reconstruction of the 3D world so as to obtain a CAD model as well as automatic initialization. The reconstruction could be approached by learning or generating a higher level parametric representation of the world via techniques such as information theory and Bayesian inference. In particular, inference techniques such as the Minimum Message/Description Length principles(MML/MDL) could be used to generate a compressed 3D map of the environment. This type of method would allow the extraction of only the pertinent information by using a Bayesian decision tree which adapts to the best model of the environment. Furthermore, it would be interesting to consider hidden surface removal and Binary Space Partition Trees (BSP-Tree) within this type of framework to account for the full 3D nature of the environment.

CHAPTER 6

Robust Estimation

6.1 Introduction

Real-time model-based tracking of 3D objects is subject to varying degrees of error. If base correspondences or tracking contain errors then registration usually fails or converges upon an imprecise position. Therefore it is necessary to improve these techniques to be robust to various sources of error so that a good estimate of the global pose parameters is obtained.

The control law presented previously falls into the more general category of estimation techniques or more particularly, non-linear iterative least-squares minimization. Estimation techniques have been studied extensively both in pure mathematics and in applied domains. In this chapter the more general class of *robust* estimation techniques will be studied. This term robust is used in many varying contexts, however, in this thesis it will be used in a precise sense. In the theoretical statistics literature, the term robust was coined by G.E.P. Box in 1953 [Press02], where *robustness signifies insensitivity to small deviations from the assumptions* [Huber81]. In the control literature, robustness of a control law has traditionally been defined as “stability results which remain true in the presence of *modeling errors* or certain classes of *disturbance*" [Samson91]. In the latter, two typical solutions emerge. The first is to create a more accurate model of the system and the other is to treat the different classes of disturbances.

With regards to system modeling errors, it is clear that when ideal measurement are available a particular model defines the accuracy and performance boundaries of an objective function. In computer vision and visual servoing literature, this has lead to modeling of the camera in terms of orthographic or perspective projection and estimating the depth parameter online [Maybank92, Grosso96, Chaumette96, Taylor99]. Intrinsic parameters have been modeled using the pixel size, the origin of the image, radial distortion and many more which have been shown to reduce modeling error and improves results [Tsai89, Luong97, Zhang00]. Image features have been modeled extensively as covered in Chapters 4 and 5. Sources of modeling error that remain include those introduced by local detection and matching of features between the first and desired images. Overcoming this class of error is often achieved by better modeling by improving the quality of tracking algorithms [Tommasini98] and feature selection methods [Papanikolopoulos93a]. Other approaches attempt to continuously update model parameters online such as in SLAM techniques mentioned in Chapter 3. In the control literature, improving estimation by modeling has lead to the development of techniques such as the H_∞ and μ-synthesis for robust control system design which factor quantifiable plant uncertainty into the design process.

The solutions mentioned in the previous paragraph are partial solutions to categorized error by further modeling, however, a real time image sequence represents a practically infinite source of information which is extremely difficult to model exhaustively by analytical development. This includes dynamic problems such as moving objects, multiple occlusion, changes in illumination due to varying light sources, specular reflections, different visual medium such as water and air, etc... It is clear that handling all the potential sources of error by analytical classification is near impossible when an infinite source of information is available. In order to represent all the possible external sources of error the correspondences may contain a statistical measure of position uncertainty is sought.

Therefore the different sources of error can be categorized into certain groups:

1. Gross error or "large outliers" - this type of error usually has a destructive effect on the estimation process.

2. Small scale noise - this type of error is usually averaged out via maximum likelihood frameworks, however, it may have a small influence on the error norm.

3. Bias - the types of error causing bias fall into the class of small scale noise at the initial iterations of the minimization process, however, they are rejected as large outliers at the end of the minimization process. They are more problematic when far from the solution. This type of error usually has a small deviating influence on the trajectory and efficiency of the estimation process.

It can be noted that for the local tracking method considered in Chapter 4, the small search window ensures that outliers have a bounded effect on the estimation process. Thus in this case small scale errors are more frequent. If the search window is widened then faster inter-frame movements can be observed and larger errors including outliers need to be considered as well.

In the robust statistics, computer vision and visual servoing literature, many different robust estimation techniques exist. Some of these have come about independently and provide different perspectives on similar problems with sometimes, slightly different goals. In Section 6.2, different techniques developed in the computer vision community are described. In Section 6.2.2 the LMedS method is reviewed in a visual control context and in Section 6.2.3 the M-estimation framework is introduced. In Section 6.2.6, scale estimation is considered and shown to be a determining factor in robust estimation. In Section 6.3 a robust visual control scheme is proposed. This is achieved by introducing a weight matrix in the error minimization.

6.2 Robust Estimation Techniques

Various robust estimation techniques exist in the literature. In this first part a short overview of these various techniques will be given. Following this each of the main approaches will be presented in more detail. In particular there are those methods that have originated in the computer vision literature and those that have originated in the statistical literature or other applied domains.

One of the oldest and most well known robust techniques in computer vision is the Hough transform which was published in a US patent [Hough62] and originally implemented for bubble chamber picture analysis [Hough59]. This method essentially extracts low dimensional parameters from measurements with noise. In the computer vision literature the Random Sample and Consensus paradigm (RANSAC) [Fischler81] is popular. In its original formulation it was applied to the pose calculation problem or the perspective n-point problem, however, it is applicable to model fitting in general.

More recently, another class of robust methods exist which have had success in producing robust computer vision algorithms. This class involves integrating multiple cues from different measurements [Brautigam98, Kragic99, Li04]. While this type of approach is far from optimal, the use of many sources of information provides good results for immediate real-world application.

Apart from these techniques there are also those which have originated in the statistics literature from a more theoretical point of view. In the statistics literature, the definition of robustness is of core importance in designing a robust estimation technique. In particular several performance factors have been considered and they can be summarized by the following principle factors which have played a determining role in the design of robust estimators:

1. Influence Function [Huber81, Hampel86] - is related to the change in an estimate from its "true" value when outliers are added to the data as a function of the distance of the data to the true estimate. For example, the influence function for the least squares estimator is proportional to the distance of a measure from the estimate.

2. Breakdown point - is the smallest percentage of outlier contamination that will force the estimated parameters to deviate arbitrarily far from their "true" estimate [Rousseeuw87]. For example the breakdown point of the mean is zero since a single outlier can break the estimate.

3. Relative Efficiency - is defined as the ratio between the lowest achievable variance for the estimated parameters (the Cramer-Rao bound) and the actual variance provided by the given method. This also depends on the underlying noise distribution [Lindgren93]. For example, in the case of Gaussian noise, the mean square estimator has an asymptotic efficiency of 1 (which is the maximum value) [Mosteller77]. Asymptotic efficiency as the number of data tends to infinity as well as small sample efficiency needs to be considered.

4. Computational Efficiency or Time Complexity - is related to the number of measures n and the number of parameters to be estimated p. For example, the time complexity for the least squares method is $O(np^2)$ and the feasibility of the computation requires at most $O(n^2)$ [Meer91].

Even with these different comparison criteria, however, it is important to keep in mind the application at hand. In recent work, Meer highlights this fact by proposing a different definition of robustness which involves the application. "*An estimator is considered as robust only when the estimation error is guaranteed to be less than what can be tolerated in the application*" [Meer04]. For example in computer vision applications a very high breakdown point is needed since it is often necessary to take into account multiple instances of the same data. Under the definition of breakdown point considered above this would mean that a robust computer vision algorithms would need to be able to tolerate 100% outlier data even if the match is good or to consider multiple instances.

Theoretical statistics started in the early 1960s, with the maximum likelihood estimator (M-estimator) being introduced in the 70s [Maronna76, Huber81, Huber96]. This approach focuses on a single geometric error function with a robust distribution. *Outliers* are then rejected which do not correspond to this definition. Standard M-estimators have been shown to have a breakdown point of less than $1/(1+p)$ where p is the number of estimated parameters [Hampel86, Li85]. The time complexity is the same as least squares, namely $O(np^2)$. The relative efficiency for large sample sizes is $n^{-1/2}$.

Under a similar framework other methods have been proposed such as R-estimates [Jaeckel72] which are based on ranking the error and using a monotonic scoring function such as the Wilcoxon rank test. An important advantage of this method over M-estimators is its invariance to scale. L-estimates use linear combinations of order statistics such as the median or the α-trimmed mean.

Another type of statistical robust estimator is the least median of squares (LMedS), which was introduced by Rousseeuw [Rousseeuw84, Rousseeuw87]. LMedS is essentially a sampling based method which is similar to RANSAC. This estimator achieves a breakdown point of 0.5. The time complexity of this method can be improved using a Monte Carlo type speed-up technique. In this case the computational complexity is $O(mn\log(n))$. The relative efficiency for large sample sizes, when Gaussian noise is present, is $n^{-1/3}$, which is low compared to M-estimators.

Robust statistical techniques are now widely used in the computer vision community. Haralick [Haralick88] proposed to use M-estimation for 2D/3D pose estimation. [Lee89] used M-estimation to estimate full 3D motion parameters. Quantitative results have shown that it is possible to obtain a correct solution with up to 30% miss-match errors. In [Tirumalai88] the least median of squares method was used for robust surface approximation and in [Kim89] it was proposed for computer vision in general. [Kumar89] used the least median of squares to solve the pose estimation problem.

The two significant algorithms in the statistics literature are both LMedS and M-estimation. M-estimation based techniques differ largely from the sampling based methods. Both will be considered more in-depth in the following sections and formulated in terms of pose calculation. The core difference between these two classes of methods can be related to their trade-off between breakdown point and efficiency. For the interested reader, good reviews of robust estimation in computer vision can be found in both [Meer91] and [Stewart99b]. In [Zhang97] and [Press02], robust estimation is also covered in a general context.

In the following subsections various important robust algorithms will be presented in more detail including Random Sample and Consensus, Least Median Squares, Maximum Likelihood Estimation, Cue Fusion, Combined techniques and robust scale estimation (robust standard deviation) .

6.2.1 Random Sample and Consensus

The Random Sample and Consensus (RANSAC) principal was proposed by Fischler and Bolles [Fischler81] for robust model-based pose estimation for computer vision applications, however, it is also applicable to a general class of model fitting. Other than pose estimation, this method has been studied for various different problems including homography estimation [Hartley01], the fundamental matrix estimation [Torr93, Torr97].

The base criterion of the RANSAC method is to maximize the number of data points η within a user defined boundary. This boundary if defined by a distance threshold τ which relates to the probability of a particular error distance being an outlier given a particular model. It is assumes that the errors are Gaussian with zero mean and standard deviation σ. Since the objective is to minimize the square of the error then the data belong to a Chi-squared distribution χ^2_m with m degrees of freedom. The rejection/inclusion error bound is thus defined as:

$$\boxed{\begin{array}{l} \text{if} \quad \left(\mathbf{s}_i(\mathbf{r}) - \mathbf{s}_i^*\right)^2 \leq \tau^2, \quad \text{then} \quad \text{outlier.} \\ \text{else} \quad \text{inlier.} \end{array} \qquad (6.1)}$$

where $\tau^2 = \Phi_m^{-1}(\alpha)\sigma^2$, with Φ being the cumulative Chi-squared distribution. The value of α is usually chosen so that a 95% probability is ensured in the case of Gaussian noise. With $\alpha = 0.95$ and $m = 6$ this leads to $\tau^2 = 12.6\sigma^2$.

The RANSAC algorithm involves randomly selecting data subsets of size m from the entire data set of size n. In general, the size of m is the minimum number of data points required to obtain a unique solution to a given estimation problem. In the case of this thesis, the data are image features. The pose estimation problem has a minimum subset corresponding to the perspective-4-point problem. According to [Hartley01] The various steps can then be summarized as follows:

1. Randomly select a sub-sample S_i of size s from the entire data and compute the model parameters (i.e. $s = 4$ points in the case of the pose $\mathbf{r}$).
2. Determine the number of data η_i which are within the threshold bound τ around the solution of set i. These point are considered to be in consensus with this model.
3. *If* the number of data η_i is greater than some threshold, then re-estimate the model using all the data and stop.

   Else select a new sub-sample and goto 1.
4. After N trials the largest consensus set S_i is selected.

where N must be chosen so that the computation time is not too long. This factor is considered in the next section.

As noted in [Wang04a] this depends entirely on a good estimate of the scale. In this paper an adaptive scale method is proposed based on mean-shift probability maxima and minima estimation. Scale estimation will be treated in more detail in Section 6.2.6.

Computational efficiency

When the number of data n increases the number of possible sub-samples increases drastically and it is impossible to treat exhaustively all of them. Assuming that the whole set of features contains up to ε outliers (e.g., $\varepsilon = 50\%$), the probability that at least one of the N sub-samples contains only inliers is given by [Rousseeuw87]:

$$P = 1 - [1 - (1 - \varepsilon)^s]^N. \tag{6.2}$$

where s is the size of the subsamples.

In order to obtain a good probability of detection ($P \simeq 1$), N can be computed from (6.2) given P, N and ε:

$$\boxed{N = \frac{\log(1 - P)}{\log[1 - (1 - \varepsilon)^s]}.} \tag{6.3}$$

For example, when $n = 20$ and $s = 6$ an exhaustive search leads to $N_{max} = \binom{20}{6} = 38760$ sub-samples where as, with $P = 0.99$ and $\varepsilon = 50\%$, only 293 sub-samples are required. These N samples can be drawn using a Monté Carlo technique.

This process can be made more efficient if "guided sampling" is employed. In this case it is necessary to have prior information about the nature of the data which is often not the case. In [Ruppert90], however, guided local line searches have been performed. Other approaches involve using heuristics.

6.2.2 Least Median of Squares

Both RANSAC and LMedS can be considered as similar algorithms which have been developed independently in the computer vision and statistics literature respectively. Even so, both domains now have much crossover and LMedS has also been used in many computer vision problems such as the recovery of epipolar geometry in [Zhang95].

Both RANSAC and LMedS are algorithms which select sample sets from the data and estimate a solution based on these subsets. The difference is that RANSAC maximizes the number of inliers while seeking the best model while LMedS minimizes a robust residual measure of the model [Meer91]. These methods remain largely inefficient from a computational point of view compared to a simple least squares solution because they must search the entire space of possible estimates generated from the data with a time complexity of $O(n^2)$ [Edelsbrunner88].

The Least Median of Squares (LMedS) [Rousseeuw87] method estimates the parameters by solving the non-linear minimization problem:

$$\boxed{\min \; Med_{i=1,\dots n}\, r_i^2, \qquad (6.4)}$$

where r_i is the residual for each available measure and n is the number of measure.

Consider a non-linear system of equations to be composed of a vector of unknown parameters $\mathbf{v}$ and a matrix $\mathbf{L}$ that projects these parameters onto the error space $s_i - s_i^*$, where s and s^* are the initial and desired values of the data. In order to simplify the relationship between the general case and the minimization approach presented in this thesis, the notation of the variables are reused, however, another non-linear system of equations could equally be used. Thus in the VVS approach $\mathbf{v}$ corresponds to the virtual camera velocity which is estimated by solving the following linear system: $\mathbf{L_s}\mathbf{v} = -(\mathbf{s} - \mathbf{s}^*)$, the residual r_i is given by:

$$r_i^2 = (\mathbf{L}_{s_i}\mathbf{v} + (s_i - s_i^*))^2$$

where $\mathbf{L}_{s_i}$, s_i and s_i^* are, respectively, the i-th line of the interaction matrix $\mathbf{L_s}$ and of the vector $\mathbf{s}$ and $\mathbf{s}^*$.

The LMedS method is solved by a search in the space of possible estimates generated from the data. This space is usually very large with n independent measurements, each with m parameters, the number of sets generated from the data is $\binom{nm}{d}$ where the dimension of the data d is greater than or equal to the minimal number of linearly independent equations required to solve the system of equations ($6 \leq d < nm$).

The algorithm is outlined as follows. Given n measures $s_i, i = 1 \dots n$:

1. draw N subsamples $\mathbf{s}_J, J = 1 \dots N$ of d independent visual features. The maximum number of subsamples is $N_{\max} = \binom{nm}{d}$, therefore if nm is large $N_{\max}$ may be very large and a Monte Carlo technique may needed to draw the N subsamples so as to improve computational complexity (see Section 6.2.1).

2. for each subsample J, the unknown parameters $\mathbf{v}_J$ are computed according to:

$$\mathbf{v}_J = -\mathbf{L}^+_{\mathbf{s}_J}(\mathbf{s}_J - \mathbf{s}^*_J)$$

3. For each $\mathbf{v}_J$, the median of square residuals is determined, denoted M_J, with respect to the whole set of features, that is:

$$M_J = Med_{i=1\dots n}(\mathbf{L}_{s_i}\mathbf{v}_J + (s_i - s_i^*))^2$$

4. The minimal value M^* is retained amongst all N M_J's along with each corresponding $\mathbf{v}_J$.

In order to detect and remove outliers. Rousseeuw [Rousseeuw87] proposes assigning a binary weight to each feature according to:

$$w_i = \begin{cases} 1 & \text{if } |r_i| \leq 2.5\widehat{\sigma} \\ 0 & \text{else} \end{cases} \tag{6.5}$$

where $\widehat{\sigma}$ is a scale estimate defined more clearly in Section 6.2.6. The following scale estimate is directly related to LMedS (see [Rousseeuw87] for details):

$$\widehat{\sigma} = 1.4826(1 + 5/(n-d))\sqrt{M^*} \tag{6.6}$$

Computational Efficiency

Since LMedS is similar computationally to RANSAC, the method described previously in Section 6.2.1 for calculating the number of required samples can also be used here. Even so, the computational cost is $O(Nn \log n)$ which remains significant, meaning that this method is not well suited to real time applications. Furthermore, if an iterative non-linear minimization method is used and LMedS is performed at each step then this cost increases dramatically. In terms of asymptotic efficiency, this method has been shown to perform poorly in the presence of Gaussian noise. The main advantages of LMedS are its 0.5 breakdown point as well as the fact that it can be used with globally convergent pose estimation methods such as [Dementhon95, Tahri03].

6.2.3 M-Estimation

M-estimators can be considered as a more general form of Maximum Likelihood Estimators(MLE). They are more general because they permit the use of different minimization functions not necessarily corresponding to normally distributed data. The maximization of this likelihood would then correspond to minimizing the log-likelihood of the observations. Many functions have been presented in the literature which allow uncertain measures to be less likely considered and in some cases completely rejected, thus inferring that the data is not normally distributed.

The following review is separated into three sections. The first explaining the effect of the loss function, ρ, upon the estimates of the parameters otherwise known as the location estimates. The second explains how the robustness of these location estimates are influenced by a scale estimate, i.e. the standard deviation of the 'good' measures. Finally different objective functions, otherwise known as influence functions will be investigated for their different properties.

Location Estimates

Once again, consider a vector of measurements $\mathbf{s}^*$ and a vector of parameters to be estimated $\mathbf{r}$. In a non-robust maximum likelihood minimization framework it is assumed that noise in the measurements is additive leading to:

$$s_i^* = \bar{s}_i^* + \delta s_i^*, \quad s_i^* \in \mathcal{R}^p, \quad i = 1, \ldots, n, \tag{6.7}$$

where $\bar{s}_i^*$ is the "true" value of the noisy measurement s_i^*.

In a statistical sense it is generally assumed that each feature measure has independent and identically distributed (i.i.d) error:

$$\delta s_i^* \approx \mathcal{N}(0, \sigma^2 \mathbf{C}), \tag{6.8}$$

where $\mathcal{N}(.)$ is a normal (Gaussian) distribution around the "true" model $\mathbf{s}(\mathbf{r})$. σ is the standard deviation or scale of the data and $\sigma^2\mathbf{C}$ is the noise covariance with $\mathbf{C} = \mathbf{I}$ in the case of i.i.d noise.

The minimization function can be re-written so that the objective becomes to maximize the probability or likelihood of a model, given normally distributed data. This is written as:

$$P(\mathbf{r}|\mathbf{s}) \propto \prod_{i=1}^{n} \left\{ \exp\left[-\frac{1}{2} \left(\frac{s_i(\mathbf{r}) - s_i^*}{\sigma_i} \right)^2 \right] \delta s_i \right\}, \tag{6.9}$$

where P is the probability of a model given the data and n is the number of measures.

Taking the log of (6.9) leads to the Chi-square merit function. In order to simplify it is assumed that the standard deviation of the image feature errors σ_i belong to the same distribution $\sigma_i = \sigma$. In this case the merit function Q to be minimized is:

$$Q(\mathbf{r}, \sigma) \equiv \left[\sum_{i=1}^{n} \frac{[s_i(\mathbf{r}) - s_i^*]^2}{2\sigma^2} \right] - n \log(\delta s), \tag{6.10}$$

where the constant terms, n, σ and δs are dropped as they do not influence the minimization procedure leading to:

$$\hat{\mathbf{r}} = \underset{\mathbf{r}}{\operatorname{argmin}} \left(\sum_{i=1}^{n} [s_i(\mathbf{r}) - s_i^*]^2 \right), \tag{6.11}$$

which can be minimized using a non-linear least squares minimization procedure.

In robust M-estimation the assumption is made that the image feature measures are not normally distributed around the "true" value. Therefore, the maximum likelihood formula for the estimated pose parameters $\mathbf{r}$ becomes:

$$P(\mathbf{r}|\mathbf{s}) \propto \prod_{i=1}^{n} \left\{ \exp\left[-\rho \left(\frac{s_i(\mathbf{r}) - s_i^*}{\sigma_i} \right) \right] \delta s_i \right\}, \tag{6.12}$$

where ρ is the negative logarithm of the probability density, and the standard deviation of each feature measure σ_i is independent.

Transforming this into a minimization problem is again done by taking the log of (6.12) and minimizing. Thus the M-estimate of the pose parameters $\mathbf{r}$ is given by minimizing the log-likelihood merit function:

$$Q(\mathbf{r}, \sigma) = \sum_{i=1}^{n} \rho \left(\frac{s_i(\mathbf{r}) - s_i^*}{\sigma_i} \right), \tag{6.13}$$

where $\rho(u)$ is a robust loss function that grows sub-quadratically and is monotonically nondecreasing with increasing $|u|$.

The robust estimate is therefore obtained from (6.13) as:

$$\hat{\mathbf{r}} = \underset{\mathbf{r}}{\operatorname{argmin}} \left(\sum_{i=1}^{n} \rho \left(\frac{s_i(\mathbf{r}) - s_i^*}{\sigma_i} \right) \right), \tag{6.14}$$

Equation (6.14) can then be minimized by taking the derivative w.r.t. time and minimizing:

$$\sum_{i=1}^{n} \psi \left(\frac{s_i(\mathbf{r}) - s_i^*}{\sigma_i} \right) \frac{1}{\sigma_i} \frac{ds_i(\mathbf{r})}{d\mathbf{r}} = 0, \tag{6.15}$$

where σ_i is assumed constant and $\psi(u)$ is called the influence function with $\psi(u) = \rho'(u)$.

Iteratively Re-weighted Least Squares (IRLS) [Deming43] is a common method of minimizing the M-estimator [Holland77] and [Huber81](pp. 179-192). It converts the problem into a weighted least-squares problem. This transformation is defined by a weight function $w(.)$ such that $\psi(u) = w(u).u$ giving:

$$\boxed{\sum_{i=1}^{n} w\left(\frac{s_i(\mathbf{r}) - s_i^*}{\sigma_i}\right)\left[\frac{s_i(\mathbf{r}) - s_i^*}{\sigma_i^2}\right]\frac{ds_i(\mathbf{r})}{d\mathbf{r}} = 0, \qquad (6.16)}$$

which can be considered as a weighted modification of the non-robust objective function. For instance, consider the case where the data is normally distributed with $\rho(u) = 1/2u^2$. In this case $\psi(u) = u$ and $w(u) = 1$.

Up until now the notion of the robust probability distribution $\rho(.)$ has remained rather abstract. In Section 6.2.3, different functions of robust distributions will be given. Following this robust statistical measures of the variation or scale of the distribution will be determined. It will be shown that this also involves determining a robust center of the distribution.

Influence Function

In theoretical statistics, the analysis of robust properties are based on a weak-star topology in the space of all probability measures such that:

$$F \rightarrow \int \psi dF \qquad (6.17)$$

is continuous for all bounded continuous functions ψ, where $F(.)$ is a distribution function.

Without going into the theoretical proofs of consistency and asymptotic mini-max theory [Huber81], it is important to consider the influence function criterion. The influence curve, first derived by [Hampel86] provides a unified description of the asymptotic properties for any arbitrary functional $T(.)$. The influence function is derived as:

$$IC(x; F, T) = \lim_{t \rightarrow 0} \frac{T((1-t)F + t\delta_x) - T(F)}{t} \qquad (6.18)$$

where $T(.)$ is a given functional, δ is a point mass 1 at x and $0 \leq t \leq 1$ corresponds to a t-quantile range. The influence function gives the influence of a single observation with value x as the sample size $n \rightarrow \infty$.

In the case of an M-estimate, the influence curve is proportional to $\psi(.)$. Many influence functions exist in the literature which provide different infinitesimally robust characteristics of the underlying probability distribution. In all cases the function must be symmetric and positive definitive so as to ensure convergence upon a minimum value. The different functions can be analyzed and compared in terms of their convergence rates and breakdown points.

In this section some of the various forms of ρ, ψ and w are compared. Of the various loss and corresponding influence functions that exist in the literature Huber's monotone function is considered, Cauchy's and McLure's soft re-descending functions are considered with $\psi(\infty)$=0 and Tukey's hard re-descending function is considered with $\psi(u) = 0$ for $|u| \geq b$, where b is a constant. Figure(6.2) and Table(6.1) permit easy comparison of the different functions and their convergence characteristics. A more detailed analysis can be found in Section 6.4 where simulations are performed for various functions.

When referring to Table 6.1, in the case of Tukey's influence function, $\frac{u}{b}$ corresponds to the point at which an outlier is totally rejected. Furthermore, the proportionality factor of b=4.6851 ensures 95%

Type / Function	ML Function $\rho(u)$	Weight Function $w(u)$
Least Squares	$\frac{u^2}{2}$	1
Huber	$\frac{1}{2}u^2, \quad \lvert u\rvert \leq a$ $\frac{1}{2}a(2\lvert u\rvert - a) \quad \lvert u\rvert > a$	$1, \quad \lvert u\rvert \leq a$ $a\frac{1}{\lvert u\rvert}, \quad \lvert u\rvert > a$
Tukey	$\frac{b^2}{6}\left(1-(1-(\frac{u}{b})^2)^3\right) \quad \lvert u\rvert \leq b$ $\frac{b^2}{6} \quad \lvert u\rvert > b$	$\left(1-(\frac{u}{b})^2\right)^2, \quad \lvert u\rvert \leq b$ $0, \quad \lvert u\rvert > b$
Cauchy	$\frac{c^2}{2}log\left(1+(\frac{u}{c})^2\right)$	$\frac{1}{(1+(u/c)^2)}$
McLure/Geman	$\frac{(u^2)/2}{(1+u^2)}$	$\frac{1}{((1+u^2)^2)}$

Type / Function	Influence Function $\psi(u)$	Chi Function $\chi(u)$
Least Squares	u	u
Huber	$u, \quad \lvert u\rvert \leq a$ $a\frac{u}{\lvert u\rvert}, \quad \lvert u\rvert > a$	$-\frac{u^2}{\sigma}, \quad \lvert u\rvert \leq a$ $-\frac{a\lvert u\rvert}{\sigma}, \quad \lvert u\rvert > a$
Tukey	$-\frac{u}{\sigma}\left(1-(\frac{u}{b})^2\right)^2, \quad \lvert u\rvert \leq b$ $0, \quad \lvert u\rvert > b$	$\frac{u^2}{2}\left(1+\frac{3u^2}{b}(1-\frac{5u^2}{3b^3})\right) \quad \lvert u\rvert \leq b$ $\frac{1}{3}\sigma b^2, \quad \lvert u\rvert > b$
Cauchy	$\frac{u}{(1+(u/c)^2)}$	$\frac{u^2}{(1+(\frac{u}{c}))^2}$
McLure/Geman	$\frac{u}{((1+u^2)^2)}$	$\frac{c^2}{2}log(\frac{c^2+u^2}{c^2})$

Table 6.1: Different Cost Functions and their corresponding Influence Functions. $\rho(.)$ corresponds to the robust probability distribution, $\psi(.)$ is the partial derivative of $\rho(.)$ w.r.t. the parameters $\mathbf{r}$, $\chi(.)$ is the partial derivative of $\rho(.)$ w.r.t. the scale σ and $w(.)$ is the corresponding weight factor for an IRLS implementation.

efficiency in the case of Gaussian noise. To ensure 95% efficiency in for Cauchy's function a proportionality factor of c=2.3849 is required. For the standard normal distribution, Huber's function requires a tuning constant of a=1.2107 for 95% efficiency.

The first two functions in the table guarantee unique solutions and give a partial derivative with respect to the scale which has a single global minimum. This allows a convergence proof to be obtained, however, they do not completely eliminate the effect of large outliers. On the other hand the last three functions in the table do not guarantee a unique solution and do not give a χ function which has an absolute minimum at $x = 0$. They are, however, favorable in terms of outlier rejection. Tukey is the only function given here which completely rejects outliers and gives them a zero weight.

Considering Figure 6.2 it can be seen that the McLure/Geman function is the most severe outlier rejector with its weighing function tending rapidly towards zero. However, further from the center of the error distribution this function only approaches zero whereas the Tukey function cuts directly to zero removing the effect of outliers completely. For Tukey's function the cutoff point for total rejection of outliers corresponds to 4.7 deviations of the residue from its initial median. It also has a similar shape to the Cauchy function. The other functions asymptotically reduce the influence of an outlier towards zero. The Huber estimator assumes that all values within the bounds of 95% of the data given the normal distribution are 100% correct and makes a gradual elimination for features outside of this region. Figure 6.2(c) shows how the scale varies with the residue for different influence functions. It can be seen here that Huber's function is the only function with an absolute minimum at $\chi(0) = 0$, and can therefore be the only function which gives a minimum σ simultaneously with a minimum residue as suggested by Huber.

6.2.4 Cue Fusion

It is possible to improve the previous robust estimation techniques by considering the fusion of multiple cues. Indeed, an important technique that almost always improves estimation robustness is the use of highly redundant number of measurements. In the case of Gaussian noise, not only does this make aberrant measurements less likely but it also makes good measurements more likely. In most estimation problems the input measurements are related to unknown parameters via a model. In the case where the measurements contain highly aberrant data or large outliers, the estimation process fails and it is necessary to use robust techniques mentioned previously. The aim of cue fusion in computer vision is therefore to exploit the many sources of information which are available from a vision sensor such as luminance, color, shape, motion, or even information from other sensors. In this way, extra redundant information is introduced into the estimation process with the aim being to improve the robustness of the algorithm.

In one hand, many cue fusion methods are based on optimization theory whereby it is necessary to correctly model the physical interaction of each type of measurement with the unknown parameters to be estimated [Clark90, Bulthoff87, Aloimonos89, Pirjanian98]. Progress in this area, however, has been relatively slow due to the difficulty in identifying valid mathematical models and the fusion of these models into one simultaneous estimation process. More recently much research has been invested in the fusion of different cues within the computer vision literature with very promising results [Isard98a, Isard98b, Pérez04, Li04]. Unfortunately, these techniques remain imprecise, are computationally expensive and do not yet run at frame-rate for the pose estimation problem.

In the other hand, another approach is to avoid the complex fusion process and avoid the interdependence of mathematical models by using a binary voting approach such as in [Brautigam98]. In other words, visual cues were fused while avoiding dependence upon each individual model by determining uncertainty using a binary voting scheme. In this way it was possible to cue incommensurable cues such as intensity and surface orientation. Furthermore, the use of a binary weighting function avoids uncertainty normalization. In this

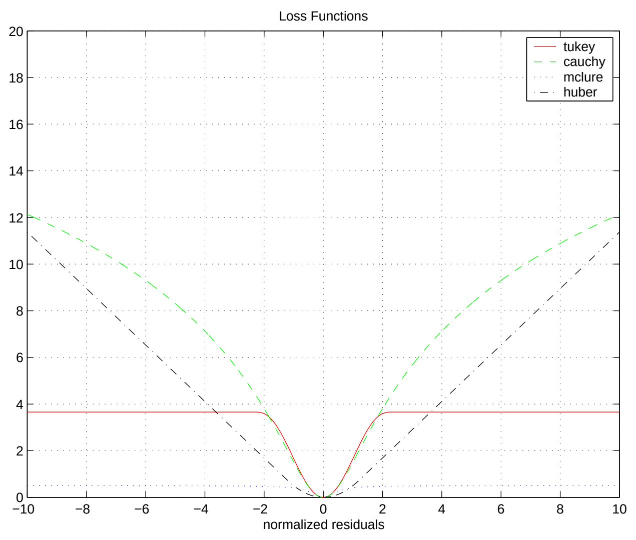
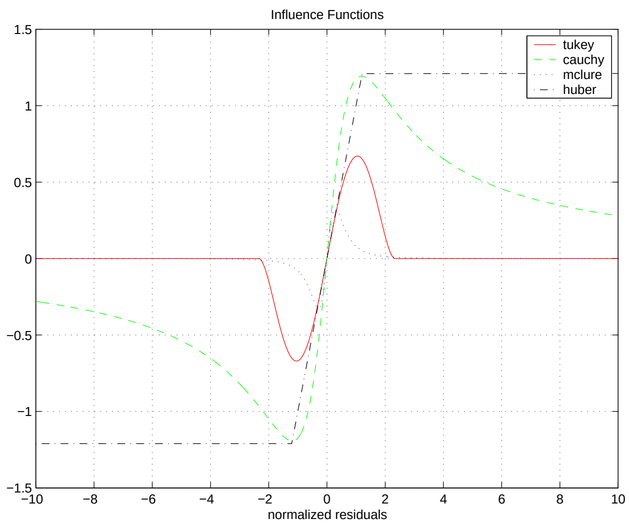
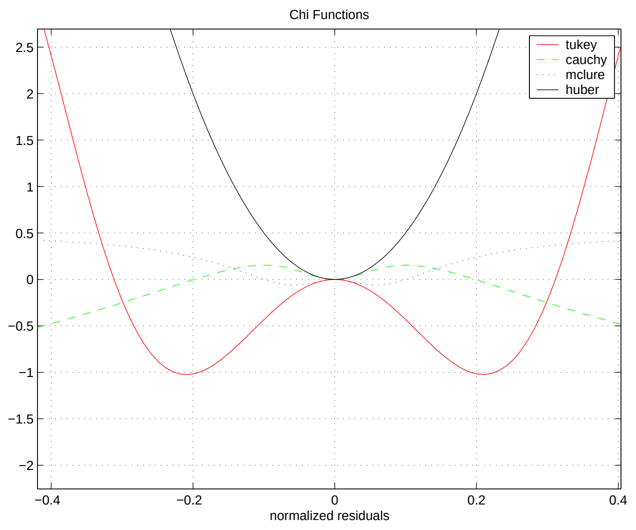
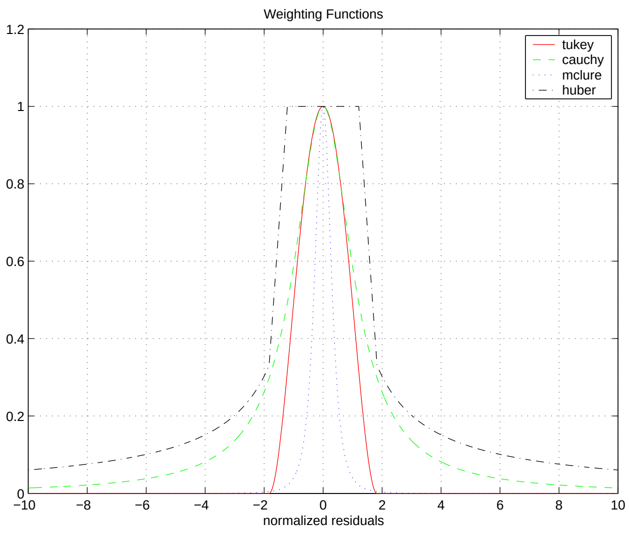
a) ML Function $\rho(u)$	b) Influence Function $\psi(u)$
$\sigma = 0.5$	$\sigma = 0.5$
	
c) Chi Function $\chi(u)$	d) Weight Function $w(u)$
$\sigma = 0.5$	$\sigma = 0.5$
	

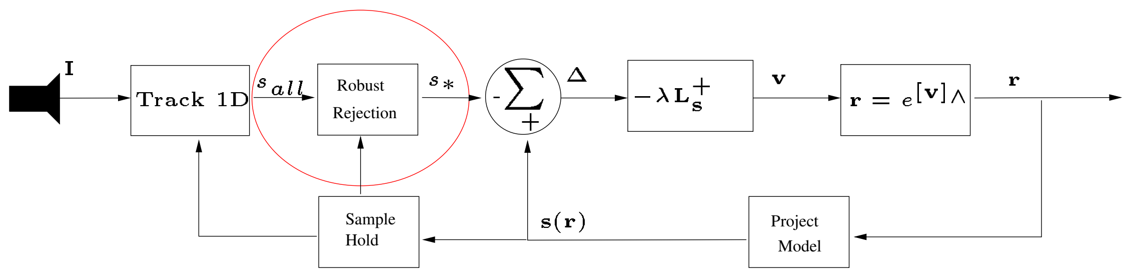
Table 6.2: Simultaneous plot of different robust functions: a) The robust probability distribution, b) The partial derivative of ρ w.r.t. $\mathbf{r}$, c) The partial derivative of ρ w.r.t. σ and d) The corresponding weight function.

paper, the technique was applied to the detection of planar surfaces from binocular images. In [Kragic01] this binary method was applied to visual servoing. This approach provides a robust input estimate (cue) to a control loop and as such treats outlier rejection, in an image processing step, prior to the visual servoing task. Furthermore, this type of method requires the tuning of parameters to deal with the complexity of the system. Of course, this type of approach provide rapid prototypes for real-world situations, however, this solution is far from optimal in terms of precision and efficiency.

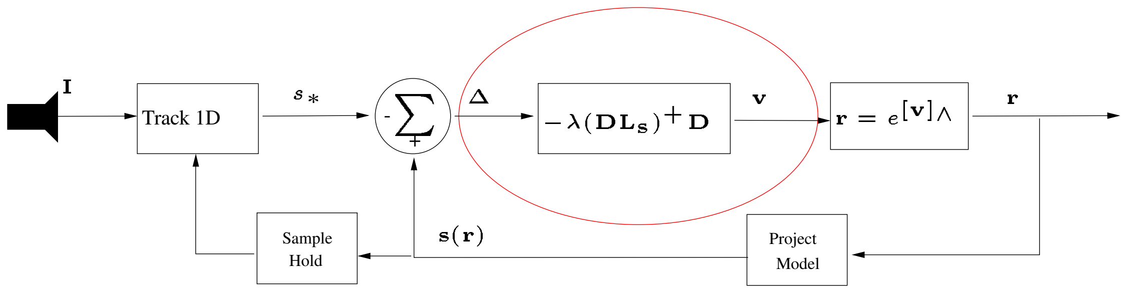
The difference between these two approaches is summarized in Figure 6.1 which treats the binary weighting method as a two-step method so the two groups of estimation algorithms emerge:

1. Remove outliers by applying binary outlier rejection rules.

2. Estimate unknown parameters simultaneously from all sources of information.



(a)


I
Track 1D
$s*$
$\sum$
-
+
Δ
$-\lambda(\mathbf{DL_s})^+\mathbf{D}$
v
$\mathbf{r} = e^{[\mathbf{v}]\wedge}$
r
Sample Hold
s(r)
Project Model

(b)

Figure 6.1: (a) Two step outlier rejection: outlier rejection is performed in an initial step prior to a second step involving accurate parameter estimation. (b) One step robust estimation: the outlier rejection is performed simultaneously with parameter estimation.

As will be shown, not only are two step approaches discontinuous but they are also a source of unwanted error. This is due to the fact that only binary weights are used as opposed to a continuous range of uncertainty values in the one-step case. In this way false positives and negatives are more easily made with respect to a simultaneous estimation process. This either results in fewer good measurements being used in the estimation process or on the other hand more outliers are retained within the estimation process. In a visual servoing scenario this error can be exacerbated since outliers can be difficult to recognize unless the robot is already close to the final position. Empirically, it can also be shown that one step robust estimation procedures are more efficient [Hampel86] due to their smooth transition between rejection and acceptance.

6.2.5 Combined Techniques

A recent trend has been to consider using random sample style methods such as LMedS or RANSAC combined with an M-estimator as was done by [Rousseeuw84] for robust statistical estimation or combining RANSAC with M-estimation as was done by [Torr97] for the Fundamental Matrix estimation. In this implementation RANSAC or LMedS was essentially used to ensure global convergence while the M-estimator was used for fine tuning in the presence of Gaussian Noise. This did not take into account the loss of precision introduced by false rejections, and it was necessary to manually tune parameters using RANSAC.

In [Marchand04], the LMedS method, was combined with an M-estimator [Comport03b] in a visual servoing context. This implementation aimed at reducing any bias in the M-estimator because any bias introduces a small deviation in the trajectory of the robot. Small bias can be present in an estimator when outliers which are not initially detected disturb the estimator. It can be noted, however, that this only occurs when outliers are of the same magnitude as the errors to be minimized. In [Marchand04], this was overcome by using LMedS to initialize the weights of an M-estimator. The M-estimator was then used to ensure robustness during execution of the task. The LMedS could not be used at each iteration because it is not computationally feasible to compute it at the cadence of the control law. The more efficient M-estimator is therefore necessary to ensure robustness to dynamic and transient errors which can appear during the movement of the robot.

The difficulty in this implementation was that the LMedS method often rejected good points along with the bad leading to false rejection. This was overcome by gradually reintroducing the initially rejected points via a manually tuned process. This meant that full robust control was handed over to the M-estimator by the end of the convergence. Unfortunately, this also meant that the visual servoing task had a breakdown point of zero during the transition period.

In the analogous VVS context this dual method would be too time consuming, however, it could be interesting to consider random sampling for pose initialization. In any case, the bias in the trajectory is not important in a virtual estimation process and any outliers not detected initially are usually rejected towards the end of the convergence.

6.2.6 Scale Estimates

In order to obtain a robust objective function, a value describing the certainty of the measures is required. The scale is a robust estimate of the standard deviation of the good data and is at the heart of the robustness of the estimation procedure. It is sometimes dubbed a nuisance parameter for location and regression problems because pure scale problems are rare. In non-linear regression such as Virtual Visual Servoing, this estimate of the scale can vary dramatically during convergence.

It can be shown that both sampling and M-estimation based methods can be considered under a unifying framework [Meer00]. In this case all techniques are considered as an M-estimator with an auxiliary scale estimate. In this way the influence function for LMedS and RANSAC are considered as binary (i.e. either a measure is good or bad). The significant difference between these two techniques then lies in the estimation of the *scale*.

Pure Statistical methods such as LMedS and M-estimators [Huber81] treat scale as something to be estimated while RANSAC and the Hough transform traditionally require a tuning variable. In LMedS, however, the scale estimation can be considered as being more robust since it is based on the residual error after a minimization step whereas in the M-estimation case the scale can only be estimated from the initial error. This effectively gives a breakdown point of 50% for LMedS compared to a lesser percentage for M-estimation. Even so, M-estimation combined with a scale estimator such as the Median Absolute Deviation

(MAD) can be shown to empirically work well even if a break down proof does not exist. No breakdown proof is available because this depends on the discontinuity of the median operator.

Robust Mean and Normalization

The standard deviation of a distribution is relative to the center of the distribution. Therefore, in order to determine a robust standard deviation, it is first necessary to define a robust center of the distribution. The usual center of the distribution is its mean, however, this has a breakdown point of zero. The median has traditionally been considered as a robust estimate of the mean.

When using a robust estimate of the mean, it is usual to normalize the distribution so as to center the data around zero. In the case of the median operator, the normalized residue is given by:

$$\bar{\Delta}_i = \Delta_i - \mathrm{Med}(\Delta_i), \forall i, \tag{6.19}$$

where $\Delta_i = [s_i(\mathbf{r}) - s_i^*]$ and $\mathrm{Med}(.)$ corresponds to the median value taken across all the errors.

Median Absolute Deviation

In the past scale has been treated as a tuning variable which can be chosen manually for a specific application [Odobez95]. Alternatively, a robust statistic can be used to calculate it. One robust statistic is the Median Absolute Deviation (MAD), given by:

$$\boxed{\hat{\sigma} = \frac{1}{\Phi^{-1}(0.75)} \cdot \mathrm{Med}_i(|\bar{\Delta}_i|). \tag{6.20}}$$

where $\frac{1}{\Phi^{-1}(0.75)} = 1.48$ represents one standard deviation of the normal distribution and $\Phi()$ is the cumulative normal distribution function. This estimated standard deviation can accordingly be used with a tuning variable which makes a particular function mini-max [Huber81] with respect to bias. In Table 6.1, a, b and c represent the different tuning variables calculated by a Monté-Carlo method for different functions.

The MAD is widely used and appears in popular algorithms such as [Tommasini98]. A convergence proof for non-linear regression using the MAD only exists if it is calculated once as an ancillary scale estimate due to its lack of asymptotic properties [Holland77]. Empirical results, however, show that determining the MAD at each iteration not only converges, but drastically improves the results. In the case of an estimation problem the introduced discontinuity does not affect the application. In a visual servoing scenario, however, it is necessary to smooth the discontinuity so as to converge along a smooth trajectory to the desired position.

6.3 Robust Visual Control

In this section the aim will be to address the problem of robust estimation in the context of visual control. A secondary objective will be to design a robust technique which may be applied to both tracking and visual servoing domains. Following on from the review of different robust techniques, it is clear that robust statistical M-estimation has many advantages over the other techniques presented and have even been considered to be a unifying banner for these estimation techniques [Meer00]. In particular, the computational efficiency can be shown to improve upon sampling style methods which is of major importance for applications with real-time constraints. Furthermore, this technique has strong theoretical foundation which addresses closely the remaining performance factors including, influence function, relative efficiency and breakdown point. The core framework is therefore based on robust statistics which is extremely general in that these methods may be applied to a general class of optimization problems.

Therefore, the *robust* visual control law proposed in this thesis is based on statistical M-estimation which provides a solid statistical basis for detailed analysis. The estimation of the scale using the Median Absolute Deviation (MAD) is used due to its computational efficiency and so that no tuning is required. In order to integrate this theory into the control law various techniques exist. For the group of methods which compute scale as an auxiliary estimate two variants exist. They are the modified residuals method and the modified weights method. The most common implementation is the modified weights method which is commonly known as an Iteratively Re-weighted Least Squares (IRLS) method [Holland77]. As will be shown, this method allows simple integration directly into the visual control law.

Pose estimation approaches are numerous including the RANSAC approach [Fischler81] which was originally based on pose estimation. [Haralick89] presented work on robust pose estimation based on M-estimation techniques. In [Kumar94] a detailed sensitivity analysis was given for robust pose computation and in [Stewart99b] a survey is given for computer vision. The 3D tracking method proposed in [Drummond02] was also based on M-estimation. Unfortunately this method required manual tuning and it did not perform iterative estimation at each image. Subsequently, it was also necessary to augment robustness with various heuristics.

In the augmented reality literature statistically robust estimation techniques have also proven to be use full as given in [Berger96, Simon98, Berger98]. In these works M-estimation was considered in the framework of pose estimation of a bridge from local point data. Scale estimation is also performed using an LMedS style approach. Work presented in [Comport03a] was also based on M-estimation and MAD scale estimation for robust augmented reality.

Of the robust methods focused on visual servoing control there is [Kragic01] who used a two step technique based on cue fusion. This approach was analyzed in detail in section 6.2.4. More recently the approach presented here based on M-estimation has been presented in [Comport03b]. In these results the accuracy, efficiency and stability in the face of outlier data were demonstrated. An important difference between robust VVS and VS is that the trajectory in a VS scenario is more important. Furthermore, the sources of error are different in each case. In the VVS case, error may come from the CAD model or the image, while in the VS case the error may come from the initial or desired images as well as the depth parameter.

6.3.1 Robust 3D Tracking

As in non-robust 3D tracking, the aim of robust tracking is to minimize the error between a pose dependent feature $\mathbf{s}(\mathbf{r})$ and a desired feature $\mathbf{s}^*$. Robust tracking, however, also has the added objective to minimize outlier data. In order to minimize outlier data a weight w_i is associated with each image feature s_i. This weight represents the confidence or uncertainty of a measured feature related directly to its deviation from an underlying statistical distribution. This weight effectively modifies the previous assumption of a normal or Gaussian distribution so that those measurements which deviate significantly from the true or ideal distribution are considered as less likely in the sense of Maximum Likelihood.

For the moment, the focus will be made on the integration of the weights into the visual tracking control law. The determination of the weights can be obtained from the previously mentioned methods in Section 6.2.1, 6.2.2 and 6.2.3. In the following section, the combination of low level uncertainty along with the weights will be considered.

The first step is to reconsider the control law originally given in equation (3.26). Introducing the weighting factor according to an Iterative Re-weighted Least-Squares procedure [Holland77] gives the new weighted objective function as:

$$\mathbf{e} = \mathbf{C}\mathbf{D}(\mathbf{s}(\mathbf{r}) - \mathbf{s}^*), \tag{6.21}$$

where

- where matrix $\mathbf{C}$ is a common combination matrix of dimension $m \times k$ where k is the number of features and m the number of controlled robot degrees of freedom (6 in the case of a rigid-body system),
- and $\mathbf{D}$ is a diagonal weighting matrix given by

$$\mathbf{D} = \begin{pmatrix} w_1 & & 0 \\ & \ddots & \\ 0 & & w_n \end{pmatrix}$$

  and where the computation of weights w_i corresponding to various influence functions was given in Table 6.1.

As is done traditionally, the reasonable assumption is made that $\mathbf{C}$ is constant, since it usually does not vary much close to the solution.

The partial derivative of equation (6.21) is then given by:

$$\dot{\mathbf{e}} = \frac{\partial \mathbf{e}}{\partial \mathbf{s}} \frac{\partial \mathbf{s}}{\partial \mathbf{r}} \frac{d\mathbf{r}}{dt} = \mathbf{C}\mathbf{D}\mathbf{L_s}\mathbf{v} + \frac{\partial \mathbf{D}}{\partial \mathbf{s}} \mathbf{C}(\mathbf{s}(\mathbf{r}) - \mathbf{s}^*), \tag{6.22}$$

where it can be noted that $\mathbf{D}$ is potentially varying with time.

In the case where $\mathbf{D}$ is considered to be varying with $\mathbf{s}$, the minimization of equation (6.22) is highly non-linear. In order to take into account this variation it is necessary to derive the chosen weight function with respect to $\mathbf{s}$. This loss function ultimately depends on a scale factor which potentially varies with time. As soon as the MAD operator is used in any of these calculations, the possibility of obtaining a continuous derivative vanishes due to the median. It could be possible to approximate the exponential decrease of the median using initial values, however, since the ultimate goal is to determine the steepest gradient, the approximate changes in the topology introduced by the variation of the weights would have little influence on the minimization. For these reasons, the assumption of $\frac{\partial \mathbf{D}}{\partial \mathbf{s}} = 0$ is made. It can be noted that the variation of the weights will affect the trajectory of the camera, however, this is not important in a tracking context.

Thus if both $\mathbf{C}$ and $\mathbf{D}$ are constant, the derivative of equation (6.21) is then given by:

$$\dot{\mathbf{e}} = \frac{\partial \mathbf{e}}{\partial \mathbf{s}} \frac{\partial \mathbf{s}}{\partial \mathbf{r}} \frac{d\mathbf{r}}{dt} = \mathbf{C}\mathbf{D}\mathbf{L_s}\mathbf{v}. \tag{6.23}$$

The exponential decoupled decrease of the error $\mathbf{e}$ still holds as in equation (3.35). Combining equation (6.21) and (6.23) using (3.35) and solving for $\mathbf{v}$ now gives:

$$\mathbf{v} = -\lambda(\mathbf{C}\widehat{\mathbf{D}}\widehat{\mathbf{L}}_\mathbf{s})^{-1}\mathbf{e}, \tag{6.24}$$

where $\widehat{\mathbf{L}}_\mathbf{s}$ is a model or an approximation of the real matrix $\mathbf{L_s}$ and $\widehat{\mathbf{D}}$ a chosen model for $\mathbf{D}$.

Since $\widehat{\mathbf{D}}\widehat{\mathbf{L}}_{\mathbf{s}}$ is a $k \times m$ matrix with full rank $m < k$, the control law can be simplified by choosing the combination matrix $\mathbf{C}$ to be the pseudo inverse $(\widehat{\mathbf{D}}\widehat{\mathbf{L}}_{\mathbf{s}})^{+}$ of $\widehat{\mathbf{D}}\widehat{\mathbf{L}}_{\mathbf{s}}$. More precisely this is:

$$\begin{aligned}\mathbf{C} = (\widehat{\mathbf{D}}\widehat{\mathbf{L}}_{\mathbf{s}})^{+} &= ((\widehat{\mathbf{D}}\widehat{\mathbf{L}}_{\mathbf{s}})^{\top}\widehat{\mathbf{D}}\widehat{\mathbf{L}}_{\mathbf{s}})^{-1}(\widehat{\mathbf{D}}\widehat{\mathbf{L}}_{\mathbf{s}})^{\top},\\ &= (\widehat{\mathbf{L}}_{\mathbf{s}}^{\top}\widehat{\mathbf{D}}^{\top}\widehat{\mathbf{D}}\widehat{\mathbf{L}}_{\mathbf{s}})^{-1}(\widehat{\mathbf{D}}\widehat{\mathbf{L}}_{\mathbf{s}})^{\top},\\ &= (\widehat{\mathbf{L}}_{\mathbf{s}}^{\top}\widehat{\mathbf{D}}^{2}\widehat{\mathbf{L}}_{\mathbf{s}})^{-1}(\widehat{\mathbf{D}}\widehat{\mathbf{L}}_{\mathbf{s}})^{\top},\end{aligned}$$

since $\mathbf{D}$ is a diagonal matrix and $\mathbf{D}^{\top} = \mathbf{D}$.

Considering an exponential decrease of the error as $\dot{\mathbf{e}} = -\lambda\mathbf{e}$ along with equations (6.21) and (6.24), the robust control law is finally obtained as:

$$\boxed{\mathbf{v} = -\lambda(\widehat{\mathbf{L}}_{\mathbf{s}}^{T}\widehat{\mathbf{D}}^{2}\widehat{\mathbf{L}}_{\mathbf{s}})^{-1}\widehat{\mathbf{L}}_{\mathbf{s}}\widehat{\mathbf{D}}(\mathbf{s}(\mathbf{r}) - \mathbf{s}^{*}). \qquad (6.25)}$$

The corresponding control diagram, given in Figure 6.2, is similar to the non-robust case. This diagram presents different ways of presenting this robust control law for the case of the distance. The first diagram considered the visual error, to be minimized, as the error between a point and a contour.

Robust Iterative Convergence

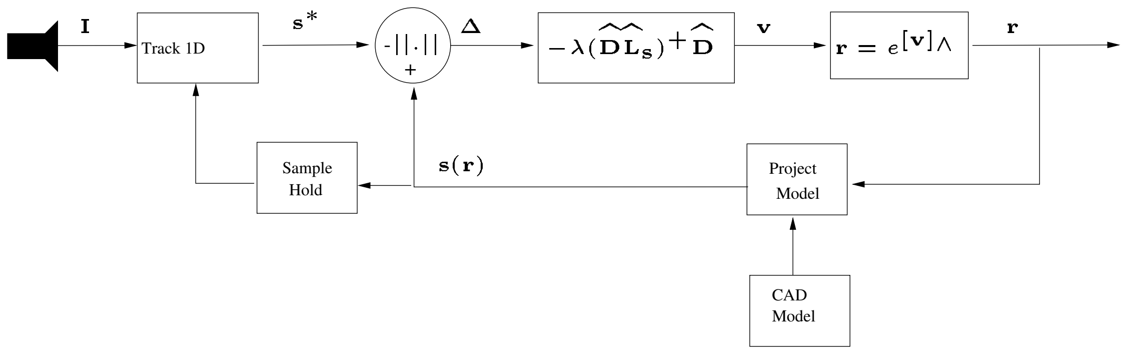
The Lyapunov positivity criterion [Samson91], given previously in equation (3.40), ensures global asymptotic stability of the non-linear system. In the re-weighted case, a sufficient criteria to ensure global asymptotic stability of the system is given by:

$$\boxed{(\widehat{\mathbf{D}}\widehat{\mathbf{L}})^{+}\mathbf{D}\mathbf{L} > 0, \quad \forall t, \qquad (6.26)}$$

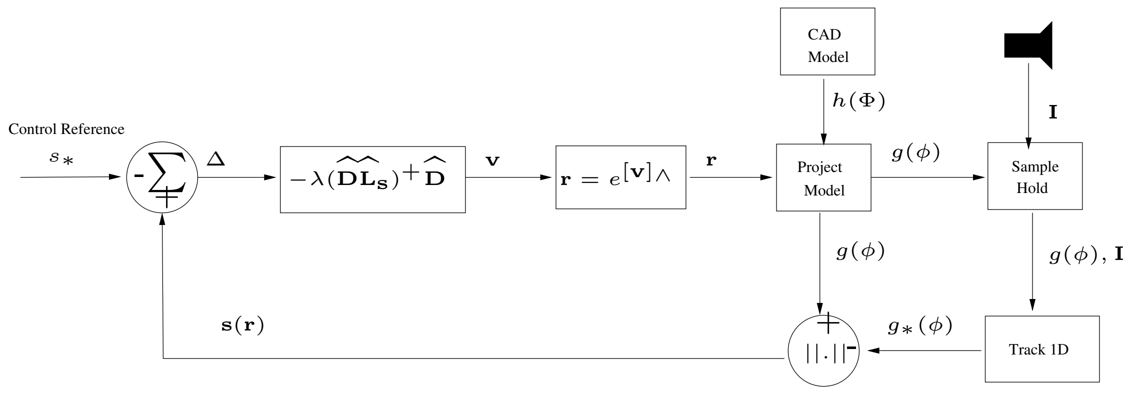
where t is the number of iteration of the control law.

As usual for a visual control law, it is impossible to demonstrate the global stability, however, it is possible to obtain the local stability for different cases of $\widehat{\mathbf{L}}$ and $\widehat{\mathbf{D}}$. The different cases which may be considered depend strongly on the application and which data is available. In the case of visual tracking, one has access to the desired position of the image features as well as the current position of the image features. On the other hand, only a current estimate of the weights and depth parameter are available. Therefore, the following cases may be considered:

- A first feasible possibility is to estimate both the interaction matrix and the weights numerically at each iteration t. In this case, a current estimate of the features $\mathbf{s}(\mathbf{r}(t))$, a current depth estimate $\mathbf{Z}(t)$ and a current estimate of the weights $\mathbf{D}(\mathbf{s}(\mathbf{r}(t)), \hat{\mathbf{Z}}(t))$ are made at each iteration of the control loop. In this case:
$$(\widehat{\mathbf{D}}\widehat{\mathbf{L}})^{+} = [\mathbf{D}(\mathbf{s}(\mathbf{r}(t)))\mathbf{L}(\mathbf{s}(\mathbf{r}(t)), \hat{\mathbf{Z}}(t))]^{+}. \qquad (6.27)$$
This choice allows the system to follow, as closely as possible, the intended behavior ($\dot{\mathbf{e}} = -\lambda\mathbf{e}$). However, it seems impossible to prove analytically that condition (6.26) is satisfied, since $\mathbf{D}$ and $\mathbf{L}$ are not constant (refer to (6.23)). Thus if the weights are correctly computed then $(\widehat{\mathbf{D}}\widehat{\mathbf{L}})^{+}\mathbf{D}\mathbf{L} = \mathbf{I}, \forall t$ and only local stability can be obtained as in the standard visual control law. The convergence of



(a)



(b)

Figure 6.2: (a) The same robust 3D tracking control law represented in only two different ways. In both, the weighting matrix $\mathbf{D}$ is applied via Iteratively Re-weighted Least Squares within the control loop and is determined simultaneously with the estimation of parameters. (a) The desired image features are considered as the tracked 2D points and the current image features are considered as sampled contours, (b) The current image feature is considered as a distance and therefore the desired features or reference signal is zero.

the weights depend on the MAD which has a maximum breakdown point of 0.5 for symmetric error distributions so reasonable performance may be expected. Even though it has been assumed that $\frac{\partial \mathbf{D}}{\partial \mathbf{s}} = 0$, the actual variation of the weights between each iteration is relatively small since the MAD is also minimized at each iteration. On the other hand, no convergence proof exists for simultaneously iterating the MAD and the pose estimation, however, empirical evidence is good.

- Another case is to consider using a constant Jacobian by using the estimated depth from the first iteration $Z(0)$ along with the initial or desired features, $\mathbf{s}(0)$ or $\mathbf{s}^*$. The interest in doing this is because the pseudo-inverse of the large interaction matrix does not have to be computed at each iteration leading to computational efficiency. Furthermore, in the case of a re-weighted control law, simplifications can be observed. Consider using the initial depth $\mathbf{Z}(0)$ and the initial or final value of the features $\mathbf{s}(\mathbf{r}(0)), \mathbf{Z}(0)))$. Furthermore, the weighting matrix in the initial step is assumed to be $\widehat{\mathbf{D}}(0) = \mathbf{I}$ so that the interaction matrix is given by:

$$(\widehat{\mathbf{D}}\widehat{\mathbf{L}})^+ = \mathbf{L}(\mathbf{s}(\mathbf{r}(0), \mathbf{Z}(0))^+. \tag{6.28}$$

where it can be seen that the initial weights do not intervene in the pseudo-inverse of the Jacobian leading to a simpler control law:

$$\mathbf{v} = -\lambda \widehat{\mathbf{L}}^+ \mathbf{e} = -\lambda \widehat{\mathbf{L}}^+ \mathbf{D}(\mathbf{s}(r_i) - \mathbf{s}^*) \tag{6.29}$$

and a simpler convergence criteria:

$$\mathbf{L}(\mathbf{s}(0), \mathbf{Z}(\mathbf{0}))^+ \mathbf{D}\mathbf{L} > 0. \tag{6.30}$$

Note also that, even if the model (6.28) is constant, the evolution of the weights during the realization of the control law are taken into account through the computation of $\mathbf{e}$, as in (6.27). Furthermore, the weights $\mathbf{D}(0)$ could be computed instead of choosing them to be equal to 1, however, these initial weights may be equally incorrect if the outliers are initially of the same magnitude as the image feature error. Once again, only the local stability of the system can be demonstrated since equation (6.30) is only satisfied around $\mathbf{s}^*$.

Of course, other combinations of parameters exist, however, they do not present any more advantages over and above the previously mentioned combinations. It is also necessary to ensure that a sufficient number of features will not be rejected so that $\mathbf{DL}$ is always of full rank (6 to estimate the pose). It should be noted here that degenerate cases are not considered, such as the duality in the image between a projected line and a point, a circle and a straight line or a cylinder and a circle.

6.3.2 Robust Visual Servoing

In this section a brief overview is given on the visual servoing counterpart to the Virtual Visual Servoing presented previously. Although the methods are intrinsically similar, they have subtle differences in their implementations.

As can be seen in Figure 6.3, the control law acts on a real robot and not on a virtual camera. The desired features are now obtained from the desired image and not from the tracked features. The current features are now the features tracked in the image as opposed to the projection of a CAD model at a given pose. Another difference between visual servoing and tracking is that a VS task usually controls the robot along a long trajectory whereas in tracking inter-image differences are relatively small. In visual servoing it is also

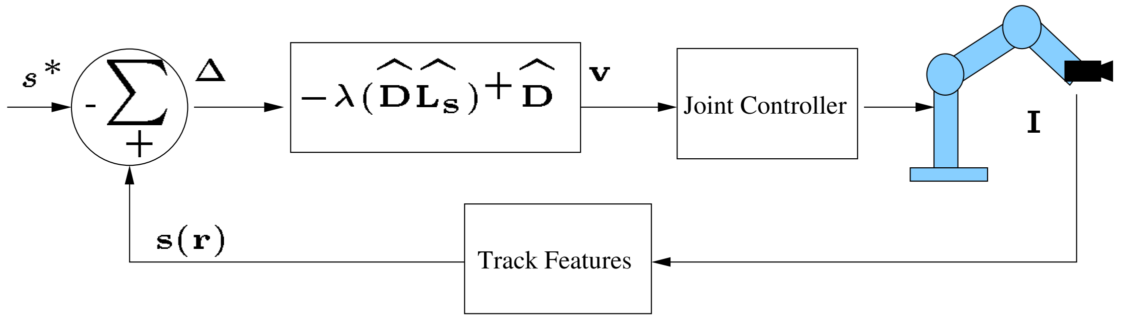


Figure 6.3: Robust control law

possible to have the final depth Z^* and not $Z(0)$ as in the Virtual Visual Servoing case. Other than these subtle differences, the control law and weight computation is carried out in the same manner.

In visual servoing the trajectory of the robot is particularly important. As mentioned previously, the variation of the weighting matrix modifies the distribution of the data and subsequently the trajectory of the camera. This is particularly important when the MAD is used as a scale estimate since the Median is not continuous and may produce a rough motion. This is best avoided by smoothing the Median.

It can also be noted that some small bias may influence the trajectory of the robot around the ideal trajectory. This occurs when outliers are of similar magnitude to the real data. In the case where these outliers are present at the beginning of the visual task, they are usually rejected towards the end of the task.

It should be noted that small sample sizes are traditionally used in visual servoing methods. In this case, many of the asymptotic proofs from the statistics literature may no longer be valid and it is necessary to carefully consider all scenarios [Raudys91].

6.3.3 Automatic Scale Selection and Propagation of Uncertainty

The local ME method described in Chapter 4 determines points along the normal of a contour using a maximum likelihood approach. The decision as to whether or not a spatio-temporal edge exists is made by thresholding the local likelihood value. ζ_{j^*} is chosen to be an edge providing that it is greater than a threshold λ. This threshold is usually chosen manually and it depends on both the contrast of the contours in the image as well as the size of the mask being applied. A method is presented here to propagate the local likelihood of the points to the global likelihood of the pose.

Assuming that the local measure of uncertainty $\zeta_{j^*} \in [0, 1]$ is independent of the global measure of uncertainty w_i, the weights are given by:

$$w_{p_i} = w_i * \zeta_{j^*}, \tag{6.31}$$

where w_{p_i} is the propagated weight and matrix $\mathbf{D}$ is then given by

$$\mathbf{D} = \begin{bmatrix} w_{p_1} & & 0 \\ & \ddots & \\ 0 & & w_{p_n} \end{bmatrix}$$

This has the effect of giving the most certainty to strong contours in terms of the local likelihood and, amongst those correspondences, the M-estimator converges upon those which conform globally to the 3D shape of the object. Effectively the robust estimator chooses which correspondences should be considered instead of a manually chosen threshold. This is advantageous when different scenes are considered along with different size masks.

6.4 Results

As stated previously, one of the aims of robust estimation is to avoid many external sources of error by using a single general function. Common external errors in 3D tracking include partial occlusions, lighting variations, miss-tracking, etc... The results given in this section will demonstrate the application of a robust control law to real-world scenarios. As mentioned this involves a Virtual Visual Servoing control law combined with robust M-estimation.

The control law used in each example is that which has been given in equation (6.25) using a camera-based motion model. As two different iterative minimization possibilities were considered for convergence in Section 6.2.6 both these cases will be presented. However, it can be said that empirical results were better when $(\mathbf{DL})^+$ is calculated at each iteration. Thus all the experiments will use this technique except in the simulation where the constant case was considered. The desired features were used in each case to calculate the interaction matrix since it allows better accuracy at the solution, however, it could equally be computed from the initial features.

The convergence criteria of the residue r is modified to include the effect of the weights is given as:

$$r^i = \sqrt{\frac{\sum_{i=1}^{n} \mathbf{D}^i(\mathbf{s}^i - \mathbf{s}^*)^2}{\sum_{i=1}^{n} \mathbf{D}_i}}, \tag{6.32}$$

where convergence is achieved if $|r^i - r^{i-1}| \times 10^{-8} = 0, \forall i$.

It should also be noted that when the scale is estimated adaptively using the MAD a lower bound is required so that it does not converge to zero and create large weight variations. This threshold was was chosen so as to avoid it reaching the level of the noise in the image. The basic resolution of 1 pixel in the image is approximately 0.0017 of a meter. Thus the lower bound on sigma was given as:

$$\sigma \geq 0.0017. \tag{6.33}$$

This value was confirmed practically by converging upon a solution of the pose with no outliers in the image. This image was then reused and the starting pose was given as the pose previously obtained. At this level the noise in the image is present and it was noted to converge upon varying poses and occasionally enter into a state of non-convergence if the threshold was not present. When present the threshold resolved this problem.

In a first part simulated experiments will be considered for pose estimation with various sources of error, various influence functions, various minimization techniques, various 3D motions. Following this experiments will be considered on real images with markers as features, thus allowing easy manipulation of the image feature to test for small and large scale errors. In a last part, real 3D tracking scenarios will be considered including indoor and outdoor sequences, scenes with heavy occlusion, large lighting variation. The 3D tracking is also applied to augmented reality and visual servoing scenarios.

6.4.1 Simulations

In order to compare and validate the different algorithms it was necessary to look at the convergence of the estimated pose upon the correct pose in the presence of outlier noise. With this in mind a simulated object with precisely known coordinates was created and projected onto an image using a virtual camera pose. In practice the coordinates of the object in the image are defined as feature correspondences between an object model and the captured image and the camera pose is what we are looking to calculate. The difference between the projected object and the corresponding points seen in the image was then minimized and a result was converged upon using a non-linear estimation algorithm, called Virtual Visual Servoing.

The simulations have been performed using basic point features based on the interaction matrix given in equation (5.13). The CAD model is a 3D cube composed of 16 3D points such that the side lengths of the cube are $5cm$ long. Each corner of the cube has two points associated. The cube is placed $1m$ in front of the simulated real camera so that the pose to be estimated is $\mathbf{r}^* = (0, 0, 1, 0, 0, 0)$. The initial guess pose of the virtual camera is then chosen to be $\mathbf{r}(0) = (0.1, 0.31, 1.4, 0.1, 0.3, 0.1)$ where the rotation is in radians. Another control variable that was chosen to be constant throughout these tests is the gain $\lambda = 0.1$ which was chosen to be small so that the transient effects of the various test are more apparent. For all the test cases where the influence function was not investigated Tukey's function was used so as to keep a control case on the comparisons. The residues were always normalized using the MAD given as in equation (6.19) and a separate M-estimate was calculated for each coordinate.

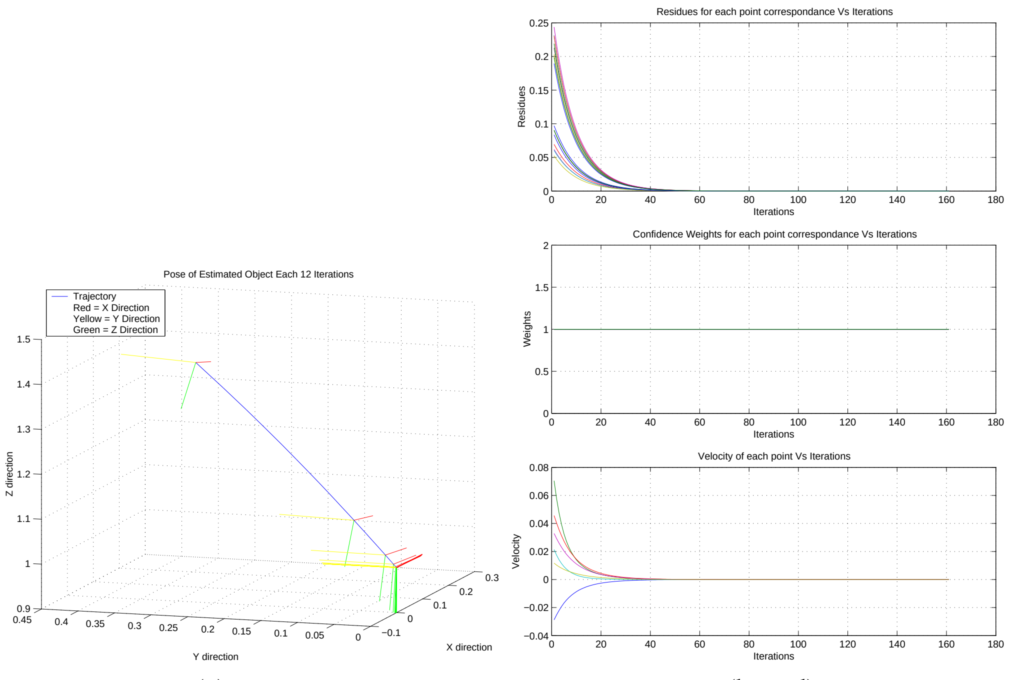
In the following test cases several of the ideal correspondences between the image and the model points were perturbed in order to simulate outliers observed in practice. In each test case two very large outliers are introduced along with two small errors. The large errors are produced by summing large values to the actual image features. Since these errors are large they are detected immediately and their weight is set to 0. The small errors are obtained by miss-matching adjacent correspondences between the model and the simulated image features. These errors are considered as small since they remain within the standard deviation of the error during several iterations of the minimization process. As such the transient response of their weights is visible during minimization and they can be seen to converge upon a rejection weight of zero.

An analysis of the robustness of this approach is based on various factors. To allow easy interpretation the first subsection presents an ideal case, a worst case (i.e. when there are outliers and no M-estimator) and a control case. The control case is given by a particular implementation of a M-estimator. Trials are presented as follows:

- Ideal case
- Outliers only
- Control case of a M-estimator
- A combined M-estimate of un-coupled x,y coordinates.
- A coupled M-estimate of the distance between two points.
- Different influence functions: Tukey, Cauchy, McLure/Geman and Huber
- Simultaneous minimization of the location and scale
- The effect of pure translation and rotation
- The effect of the number of outliers
- Calculation of the interaction matrix at each iteration vs calculation only for the first iteration
- Static images

Ideal Case

The case presented in Figure 6.4 has no added errors and is a simulation of an ideal situation where the point correspondences obtained are considered perfect.



(a) (b, c, d)

Figure 6.4: The control case with no outliers. The evolution of the (a) pose (b) residuals (c) weights (d) velocity at each iteration.

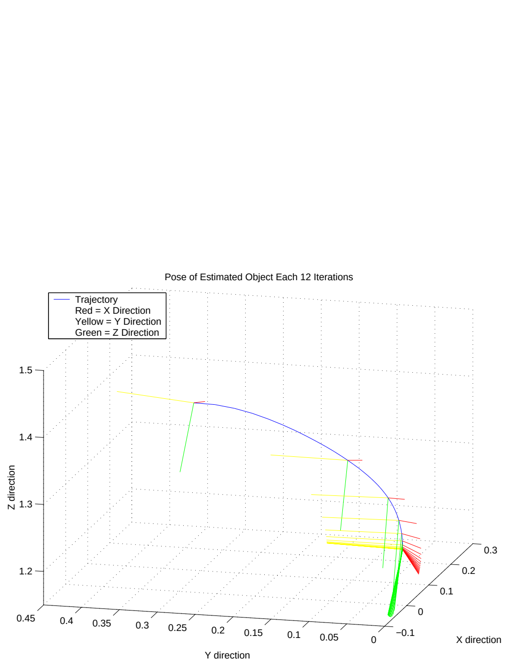
This clearly converges easily to the desired position with the final pose $\mathbf{r} = (0, 0, 1, 0, 0, 0)$. In Figure(6.4(b)) the convergence of two sets of residues grouped together is noticeable. This represents the residues of each coordinate, x and y. It can also be noted that the convergence takes 162 iterations. The number of iteration is representative of the computational efficiency of the algorithm.

Outliers

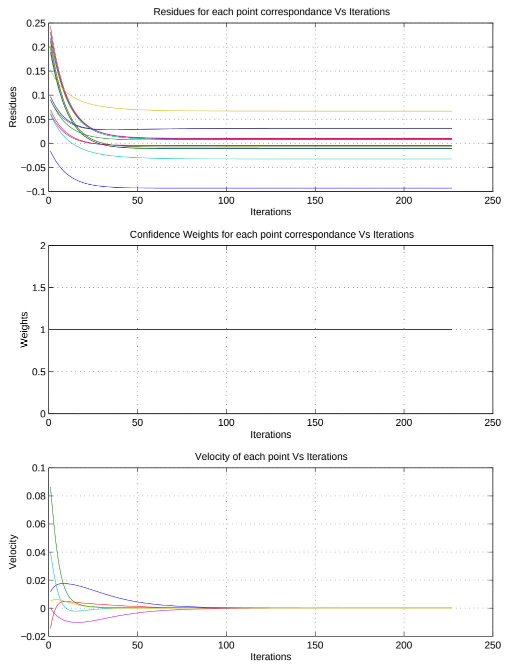
The outliers introduced here are representative of the errors found in practice. One large error is introduced and two other points are mismatched. After 227 iterations of the algorithm the effect of the introduced outliers can be seen in Figure 6.5 and in the accuracy of the final pose which was erroneous (i.e. $\mathbf{r} = (0, 0, 1, 1, 41, 1)$). This pose has a considerable amount of error particularly in the rotation. This is evident in the plot of the camera pose trajectory (Figure 6.5(a)) as well as in the plots showing convergence of the error (Figure 6.5(a)) and the plot showing convergence of the pose (Figure 6.5(d)).

Tukey's influence function

Firstly a control case is established for comparison of the simulations carried out in the following sections. In the simulation reported in Figure 6.6 Tukey's function was used and the MAD was calculated at each



(a)



(b, c, d)

Figure 6.5: The effect of outliers without robust estimation. The evolution of the (a) pose (b) residuals (c) weights (d) velocity at each iteration.

iteration. As will be shown later in the report this leads to the necessity to rely on the MAD as being a ’robust’ and ’invariant’ estimate of scale.

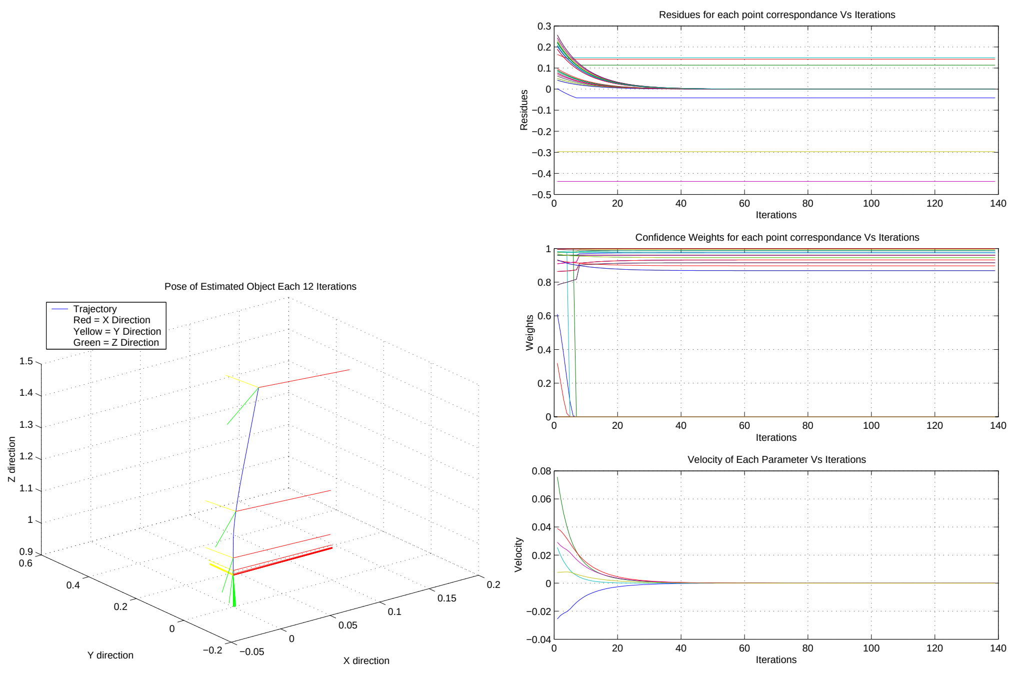
Furthermore, if we calculate the median at each iteration it is possible to use this estimate to normalize the data and provide an unbiased set of data. The weights are then given by:

$$w_i = \frac{\psi(\bar{\Delta}_i)}{\bar{\Delta}_i}, \tag{6.34}$$

where the normalized residue $\bar{\Delta}_i$ is given by equation(6.19) at each iteration.

In addition an M-estimate is made for the ensemble of x coordinates and another for the ensemble of y coordinates, in-effect decoupling the M-estimate. This can be compared to Figure 6.8 where a distance measure is used to couple the M-estimate and Figure 6.7 where a single M-estimation is made for the combined ensemble of x, y coordinate residues.

It can be seen here that the mismatched outlier points are rejected very rapidly in around 5 iterations. Due to the separation of coordinates it can be seen in Figure 6.6(c) that each coordinate of a pair has a separate weight. The weights of the ’outliers’ are given as 0 representing a complete rejection. Note that when one coordinate is rejected the other is manually discarded as well. The large outliers are rejected immediately and thus do not appear in Figure 6.6(b). Convergence is achieved in 139 iterations which is the same as the ideal case. The weights for the inlier data are kept close to 1 compared to the following simulations.



(a) (b, c, d)

Figure 6.6: Decoupled x and y weight computation. The convergence of (a) the pose with outliers and an M-Estimator (b) residuals (c) weights (d) velocity at each iteration

A combined M-estimate of x, y coordinates

The effect of treating the residues of the x and y coordinates as equivalent and calculating the weights upon an entire vector of residues is presented in Figure 6.7.

It is clear that this case is not as good as computing weight for x and y coordinates separately because the x and y coordinates are not decoupled resulting in a bad distribution of residues. This results in a wide spread of weights given to good values and total outlier rejection occurs much later at around 20 iterations.

M-estimate of the distance measure

In the following subsections an investigation is made between either defining a residue for the entire point (i.e. a distance) or calculating different M-estimates for each coordinate as given in the control case.

The distance measure is given by:

$$\bar{\Delta}_d = \sqrt{\bar{\Delta}_x^2 + \bar{\Delta}_y^2},$$

where $\bar{\Delta}_d$ represents the distance of the normalized residues x and y.

Note in Figure 6.8(c) that there are no longer weights for x and y coordinates but weights for each distance. It can be seen that this converges and does so in 166 iterations. The mismatched points or the outlier points are rejected at around 20 iterations which has similar rejection properties to the previous section. The outlier weights make a slight oscillation before being rejected which could be attributed to the movement of the median value of the residues during convergence. The weights for the correct correspondences are acceptable in this case.

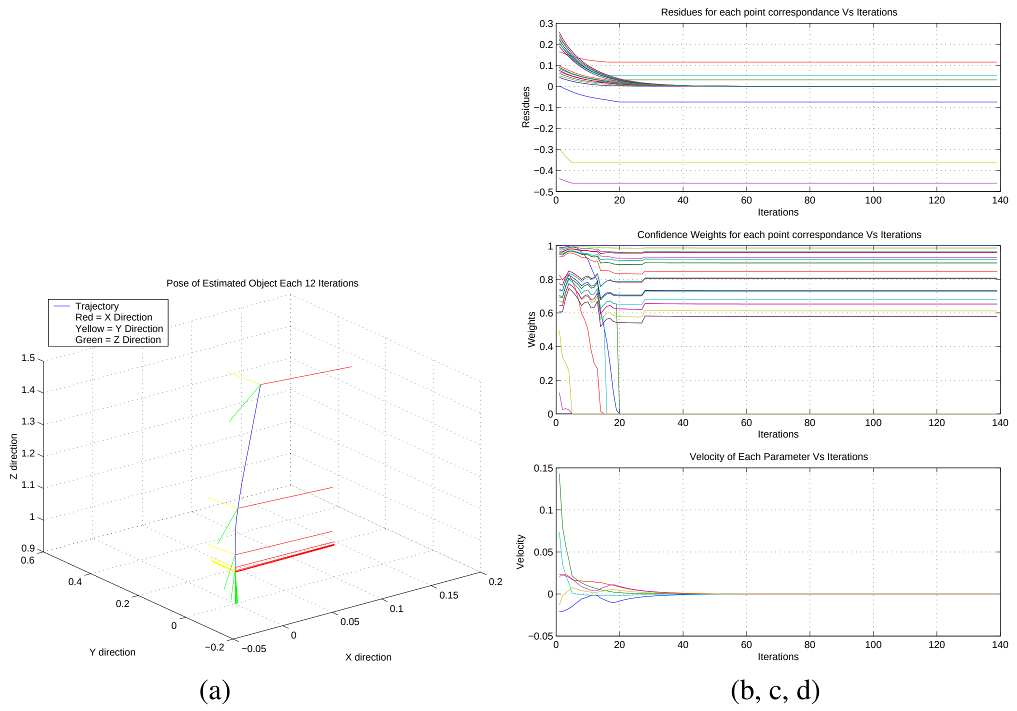


Figure 6.7: Coupled (x, y) weight computation. The convergence of (a) the pose with outliers and an M-Estimator (b) residuals (c) weights (d) velocity at each iteration.

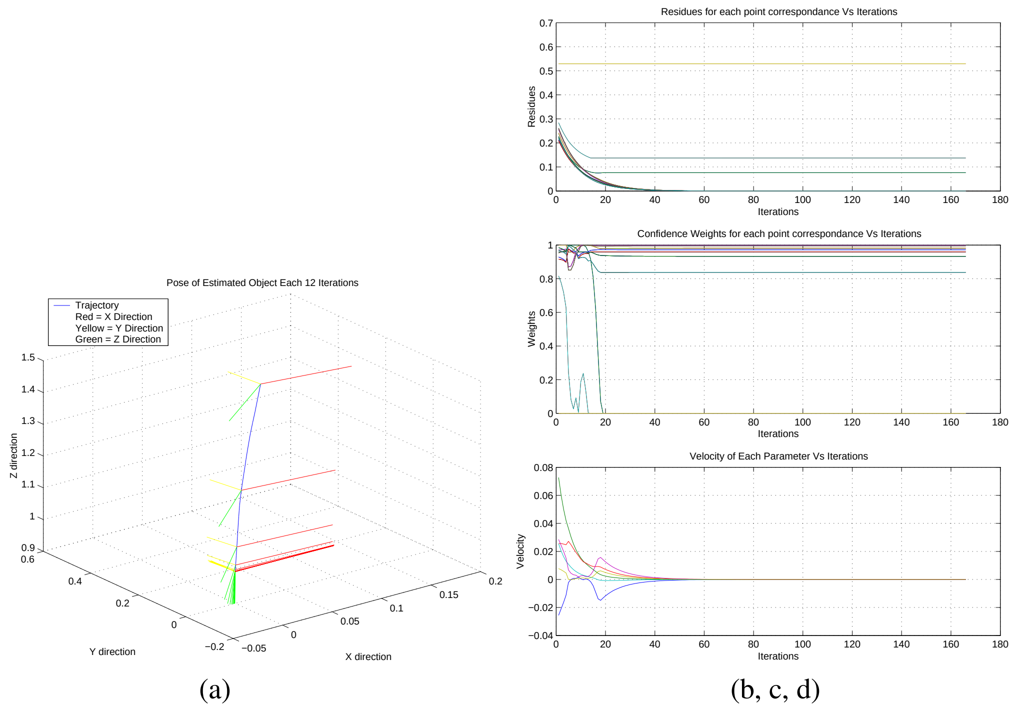
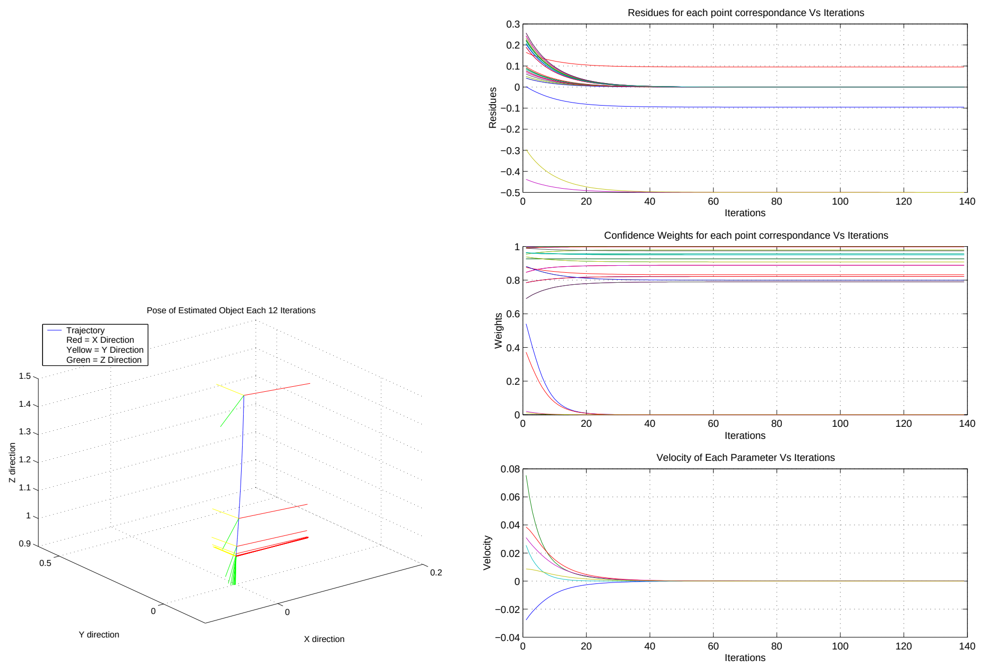


Figure 6.8: A distance measurement. The convergence of (a)the pose (b)the residuals (c)the weights (d)the velocity at each iteration

The reason why the separation of the coordinates as given in Figure 6.6 performs better in certain situations can be explained by the decoupling of the x and y translation. It should be noted that this case only rejects outliers more quickly when the object is away from the translation diagonal ($y = x$ or $-y = -x$) in the image frame. If the object lies on the diagonal it rejects the outliers in a similar fashion to the control case presented above. The only difference being that in this case the correct values are more widely spread and result in worse weights being given after outlier rejection.

Cauchy

In Figure 6.9 the response of the soft re-descending Cauchy M-estimator with the same perturbation as used previously. The normalized residue is used in conjunction with decoupled M-estimate of the x and y coordinates.



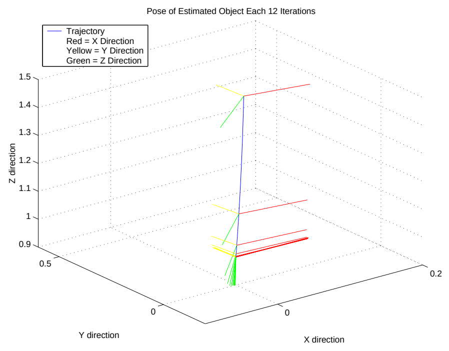
(a) (b, c, d)

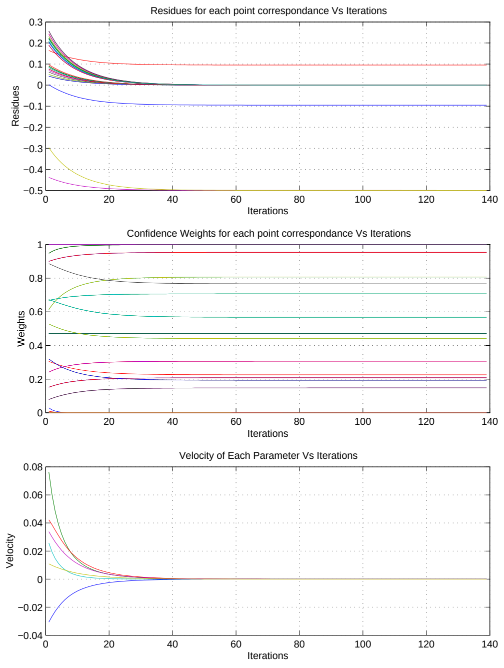
Figure 6.9: Cauchy's influence function : The convergence of (a)the pose (b)the residuals (c)the weights (d)the velocity at each iteration

Convergence was achieved in about 180 iterations with a final pose of $\mathbf{r} = (0, 0, 1, 0, 0, 0)$. It can be seen in Figure 6.9 that the weights taper off gradually as opposed to Tukey's function where the weights are quickly rejected.

McLure/Geman

Figure 6.10 gives the response of the soft re-descending McLure M-estimator with the same disturbance as used previously. The normalized residue is used in conjunction with decoupled M-estimates for the x and y coordinates.





(a) (b, c, d)

Figure 6.10: Geman/McLure influence function: The convergence of (a)the pose (b)residuals (c)weights (d)velocity at each iteration

Convergence was again achieved in 139 iterations with the correct final pose of $\mathbf{r} = (0, 0, 1, 0, 0, 0)$. When considering Figure 6.10(c) it can be noted that the outliers are rejected rather quickly, however, the remainder of the good data is attributed a spread of weights. This is due to the fact that the weighting function descends more sharply than the other functions and thus results in a greater spread for the correct values.

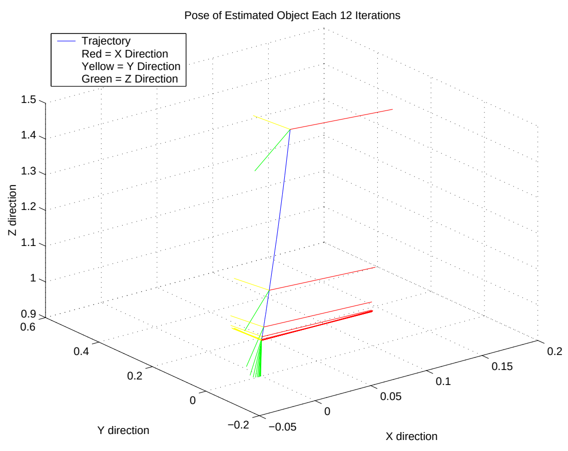
Huber

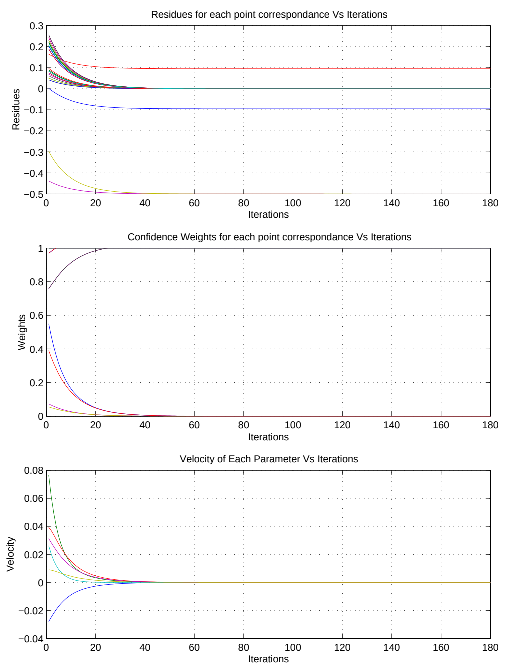
Figure 6.11 gives the response of the Huber monotone re-descending M-estimator with the same perturbation as used previously. The normalized residue is used in conjunction with decoupled M-estimates for the x and y coordinates. The MAD is updated at each iteration.

Convergence was achieved in 197 iterations with a final pose of $\mathbf{r} = (0, 0, 1, 0, 0, 0)$. It is apparent that the M-estimator is successful in finding both outliers. The weighting given to the outlier correspondences descends smoothly according to the parabolic maximum likelihood function. The Huber function attributes the highest weights to the correct correspondences, out of all three functions tested here.

Translation vs Rotation

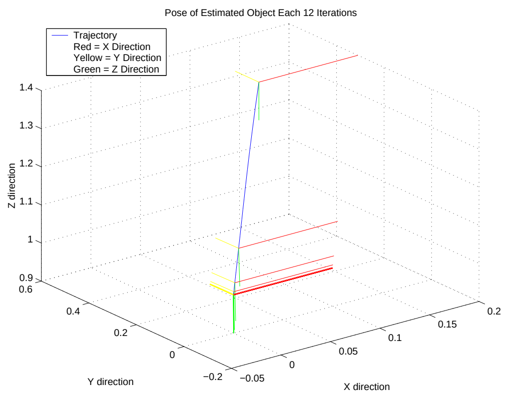
The case given above for the Tukey M-Estimator, with two points mismatched, is re-applied in the case of a pure translation (see Figure 6.4.1) and pure rotation (see Figure 6.13). The case of pure translation takes 166 iterations to converge while the rotational estimation case took only 142 iterations simply because the magnitude of each error was not the same.





(a) (b, c, d)

Figure 6.11: Huber's influence function: The convergence of (a)the pose (b)residuals (c)weights (d)velocity at each iteration



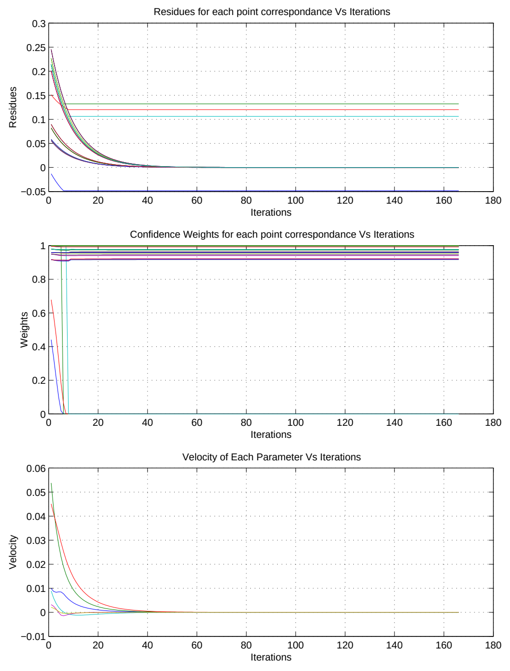


Figure 6.12: The case of pure Translation.

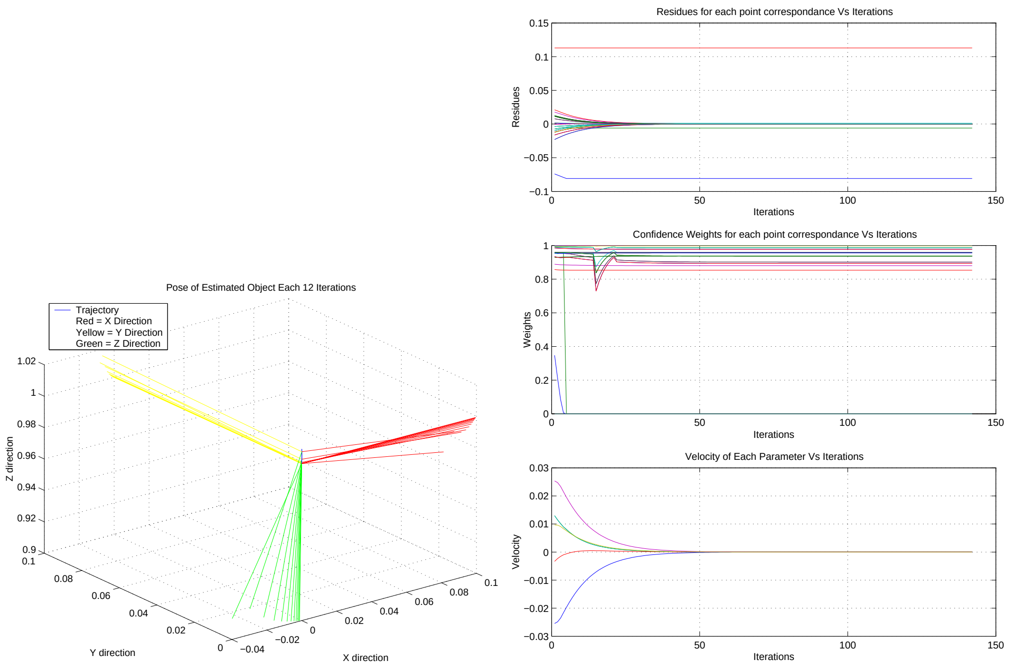


Figure 6.13: The case of pure Rotation.

The outliers are rejected successfully in both cases, however, it can be noted that in the case of the rotational weights, the spread of weights is larger than the translational case. If no lower bound threshold is given for the scale then this spread of weights is much more evident. It can be seen in Figure 6.13(c) the point where the lower bound on the scale is reached. Apart from this difference, a trial was also made for separate translational and rotational motion using the distance given by equation (6.4.1). In the case of translation the outliers are not rejected completely until around iteration 20, however, in the case of rotational motion the outliers are rejected immediately. This suggests that the distance measure is a better M-estimate for rotational error.

These differences between rotational and translational motion are due to the fact that a rotational motion induces a wide range of errors whereas the translational motion gives a constant error in the image. For example, with a square object, the contours that are closer to the center of rotation have smaller errors that those which are towards the extremities. This means that the errors are distributed across a wide range of values and subsequently the M-estimator incorrectly tries to compensate for this. However, since the errors tend towards a normal distribution the estimation converges along with the spread of the weights. Furthermore, the decoupled coordinates only improves the weight computation for the translational motion and not the rotations since the rotational error is not decoupled by these projections of the error.

Number of Outliers

This part looks at the effect of increased outliers. The Tukey function is used with the normalized residue and separate M-estimates for x and y coordinates. First of all 25% outliers are tested and it can be see to compare well with Figure (6.6) with only four outliers.

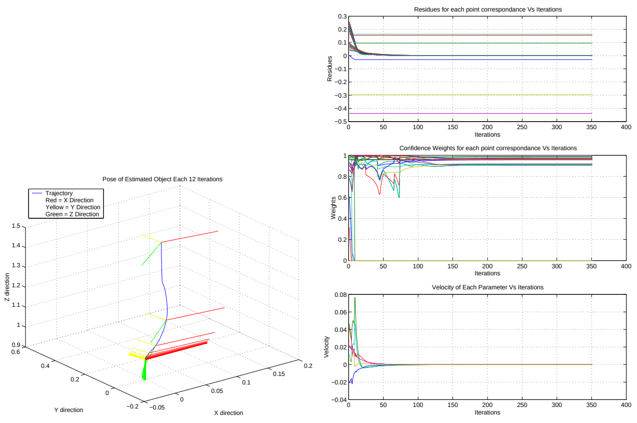
From Table 6.3 it can be noted that the 50% outlier case does not converge. Several tests were made in attempt to achieve convergence in this case and it was noted that it is only possible to achieve 50% outlier rejection in particular cases. Although the MAD can be shown to have an absolute breakdown point of 25%, it can be shown that symmetric data distributions approach a maximum breakdown point of 50% [Huber81]. Furthermore, if the MAD is calculated iteratively, no convergence proof is available [Holland77]. As expected the 62.5% case doesn't converge as the MAD breaks down. It can be seen here that this has the effect of rejecting too many correspondences, resulting in divergence.

Indeed, the median is a very simplistic method for calculating the scale. Another simple solution could be to manually select a value of σ based on the application. Future research must be directed in the direction of using improved estimates such as simultaneous pose and scale estimation or the use of Kernel Density estimation [Meer04, Wang04a] techniques.

Constant interaction matrix

As mentioned previously in section 6.3.1, several choices exist for the interaction matrix and weights according to the control law given in equation(6.29). In these experiments the effect of using a constant interaction matrix is investigated. All variables are kept the same as in the control case.

The first case considered in Figure 6.14 is such that the pseudo-inverse is calculated in the first iteration using the position of the features in the image calculated by the guess pose. Furthermore the weights $\mathbf{D}$ were assigned to $\mathbb{I}$ in this first iteration thus making the assumption that all correspondences are correct. The number of iterations 351 was much larger than the control case, however, the algorithm still converged upon the correct pose.



(a) (b, c, d)

Figure 6.14: The convergence of (a)the pose (b)residuals (c)weights (d)velocity at each iteration.

The second case considered in Figure 6.15 is such that the pseudo-inverse is calculated in the first iteration using the position of the features in the desired image. The number of iterations 338 was also much larger than the control case with a final pose $\mathbf{r}^* = (0, 0, 1, 0, 0, 0)$. It was also noted that when increasing the gain of the system significantly, this system took longer to converge when compared to the optimal case. Furthermore, failure to converge was observed with large gains. In conclusion, it is better to use an interaction matrix calculated at each iteration.

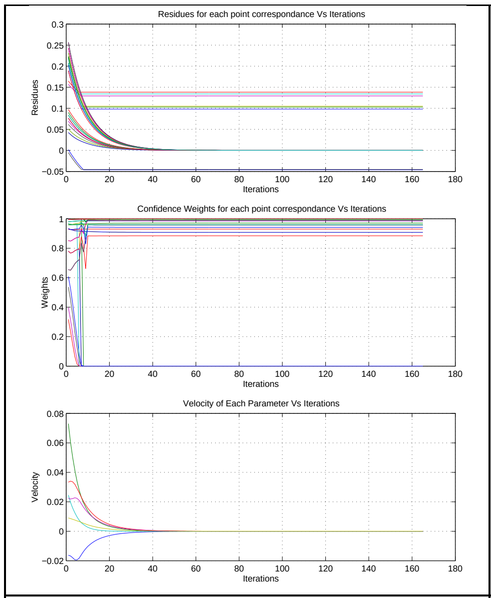
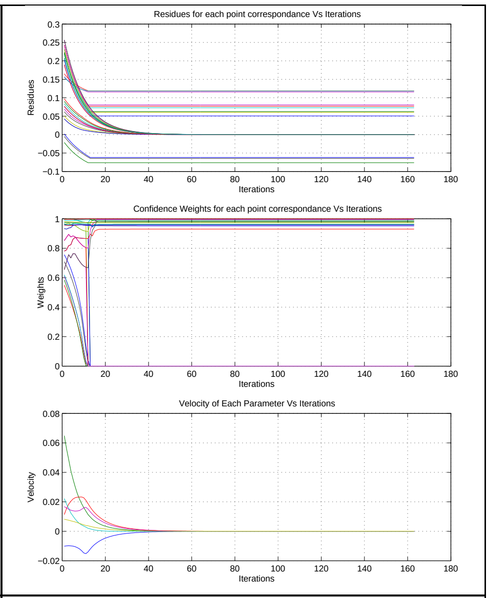
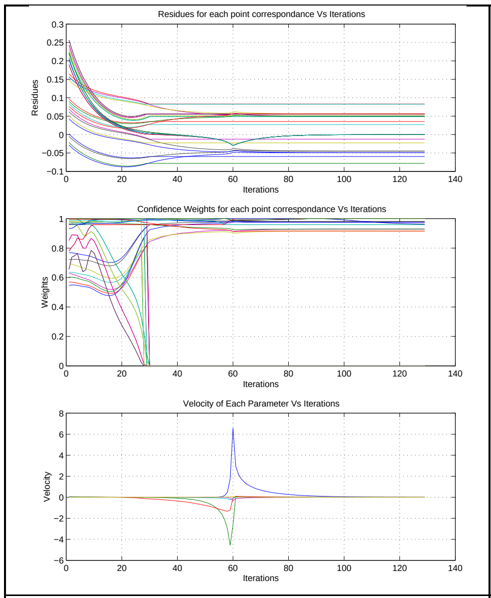
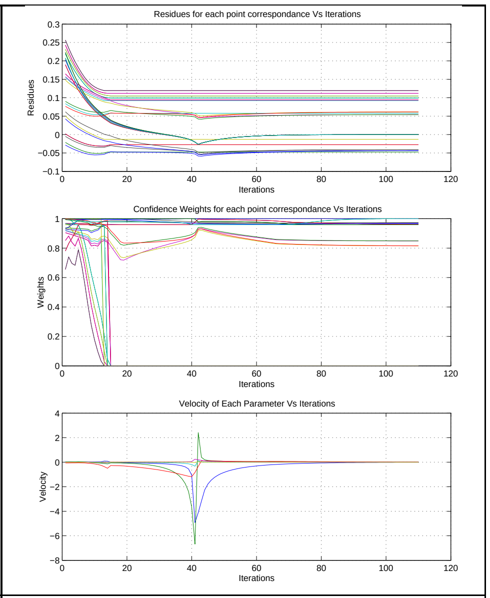
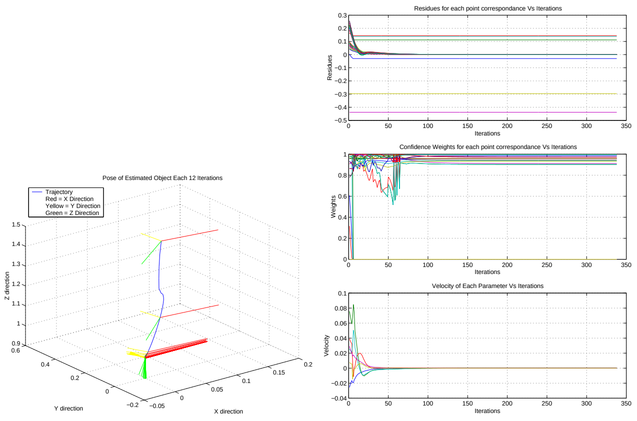
25%	37.5%
	
Iterations = 169, Final pose $\mathbf{r}^* = (0,0,1,0,0,0)$	Iterations = 163, Final pose $\mathbf{r}^* = (0,0,1,0,0,0)$
50%	62.5%
	
Iterations = 129, Final pose $\mathbf{r}^* = (0,1,20,73,63,-77)$	Iterations = 110, Final pose $\mathbf{r}^* = (0,1,18,48,-77,51)$

Table 6.3: The convergence upon the pose for different percentages of outliers.



(a) (b, c, d)

Figure 6.15: The convergence of (a)the pose (b)residuals (c)weights (d)velocity at each iteration.

6.4.2 Static images

The simulations carried out previously were confirmed on real images. In Figure 6.16 a lplanar square object is considered. The correspondences between the image and the CAD model were chosen manually. The green cross-hairs represent the manually selected correspondences and the red cross-hairs are computed from the green cross-hairs by determining the center of gravity of the white area. These red points then become the desired points which correspond to points on the CAD model. It can be seen that one large incorrect correspondence was made. Less evident in the image is the miss-matching of two point correspondences. The initial pose is shown in blue. In this example Tukey's influence function was used along with the normalized residue and a separate M-estimate for x and y coordinates. In this test an iteratively computed interaction matrix was also used and the current image features were used to calculate these interaction matrices.

The final pose was checked by comparing it to the pose calculated with no outliers created. The system converged upon the correct pose and in Figure 6.16(c) it can be seen that the rejected points do not even appear and are immediately given a weight of zero. It can be seen Figure 6.16(a) that the residues for the outliers remain large while the rest of the errors converge.

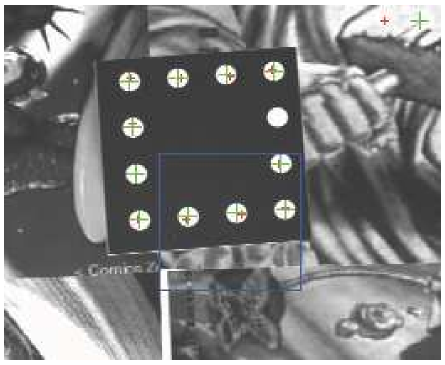
6.4.3 Robust Visual Servoing

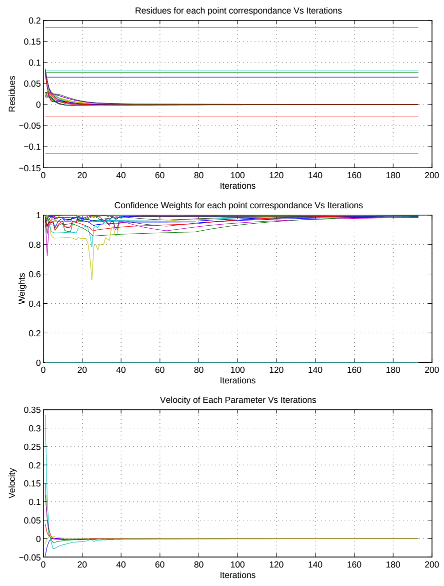
Since robust Virtual Visual Servoing is equivalent to visual servoing, the robust control law can easily be applied directly to a visual servoing task. As seen in figure 6.17, the resulting control law successfully converges in the presence of outliers. More complete results are given in [Comport03b] and [Marchand04].

6.4.4 Real-time robust tracking

Tracking a micro-controller

In the following section, several real-time tracking experiments are performed. The experiment reported in Figure 6.18 was performed using a long sequence of a micro-controller. This sequence presented difficulties associated with complex objects such as close object edges which lead easily to mismatching when using a


(a)



(b)

Figure 6.16: (a) The test image with correspondences displayed with cross-hairs. (b) The convergence of (a)the pose (b)residuals (c)weights (d)velocity at each iteration.

ME algorithm. It can be seen in the sequence that external objects rapidly occlude the micro-controller. Furthermore, the micro-controller partially leaves the camera field of view during motion. Rapid and dramatic changes in illumination due to shadowing introduces further sources of errors. The outlier rejection and overall tracking was shown to be able to track this object throughout the entire sequence and to effectively reject erroneous points. Real-time tracking (i.e. framerate) performance was also achieved.

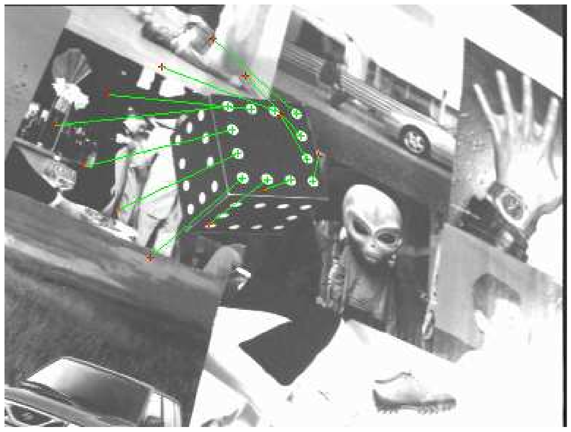
Tracking in an indoor environment

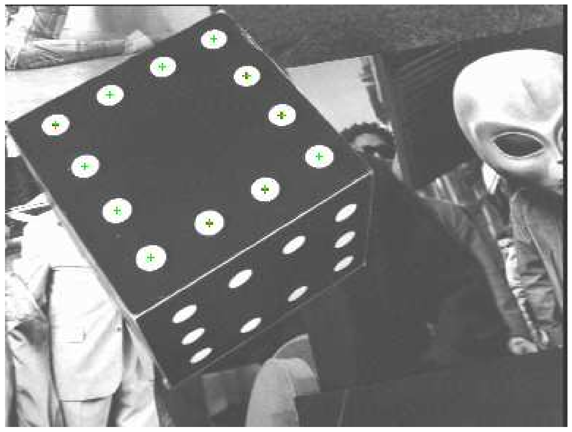
In this experiment an object whose model is composed of a cylinder, a circle and two lines is considered (see Figure 6.19). This long sequence features numerous occlusions. Although the images are quite simple in this experiment, if no robust estimation is considered tracking fails after a few images because the minimization process has to deal with miss-tracking and problems due to occlusion.

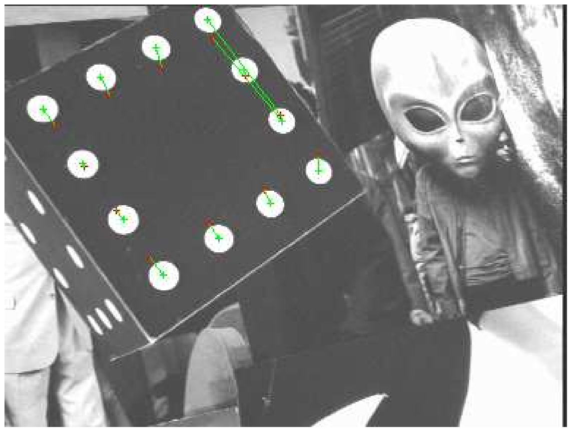
Tracking in an outdoor environment

In the next experiment (see Figure 6.20), an outdoor scene is considered. Here, distance to the projection of a 3D cylinder and to two 3D lines are used to compute the pose. Despite very noisy images (wind in the trees, multiple occlusions, etc.) tracking is achieved along a 1400 image sequence. The images display the tracked lines and cylinder limbs as well as 3D information inserted after the pose computation (the reference frame and the projection of the cylinder and lines (in blue)).

It can be noted that in this case the target object is static while the camera moves. Therefore it is possible to consider using the full spatio-temporal contour tracker including the temporal test on the edge contrast (see Section 4.5 in Chapter 4 for details). In this scenario this contrast test helps when the person passes behind the pillar since the pillar provides the only vertical constraints on the pose. This interference is not


(a)


(b)


(c)

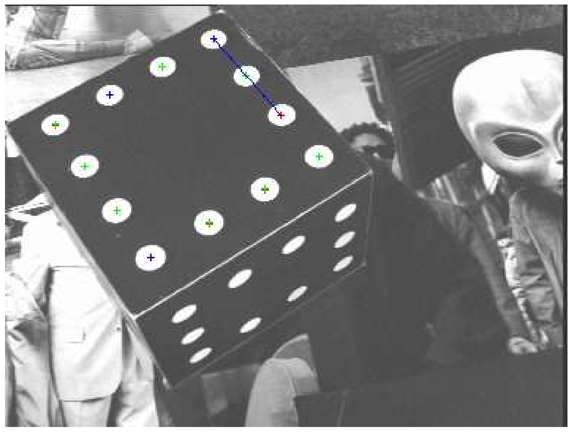

(d)

Figure 6.17: A classical positioning task with a classical control law and a robust control law. The desired or final image has been acquired off-line by a teaching by showing approach. Image (a) shows the initial image. The three remaining images correspond to images acquired after execution of the positioning task. (b) corresponds to a classical control law without any outliers(the reference experiment), (c) uses the same control law but false correspondences are made between current and desired points, introducing outliers (this is a possible source of error, however, other cases are also possible [Comport03b, Marchand04]). As one would expect, the robot converges towards a local minimum, (d) considers the same experiment but with a robust control law. In spite of outliers, the positioning task converges correctly.

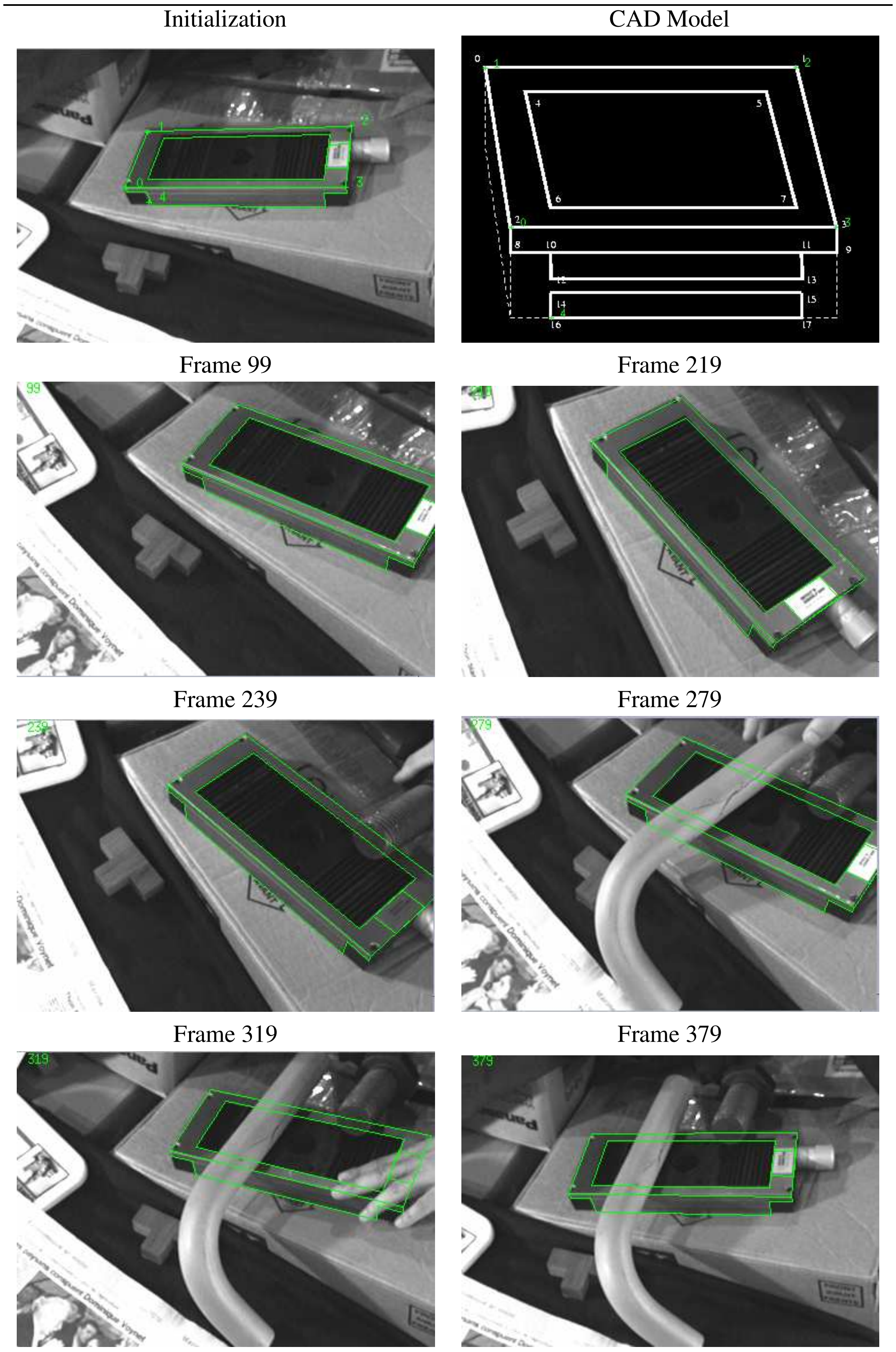


Figure 6.18: Tracking of a micro-controller using a 3D CAD model composed of lines. A a large quantity of occlusions can be observed (hand, tools, etc).

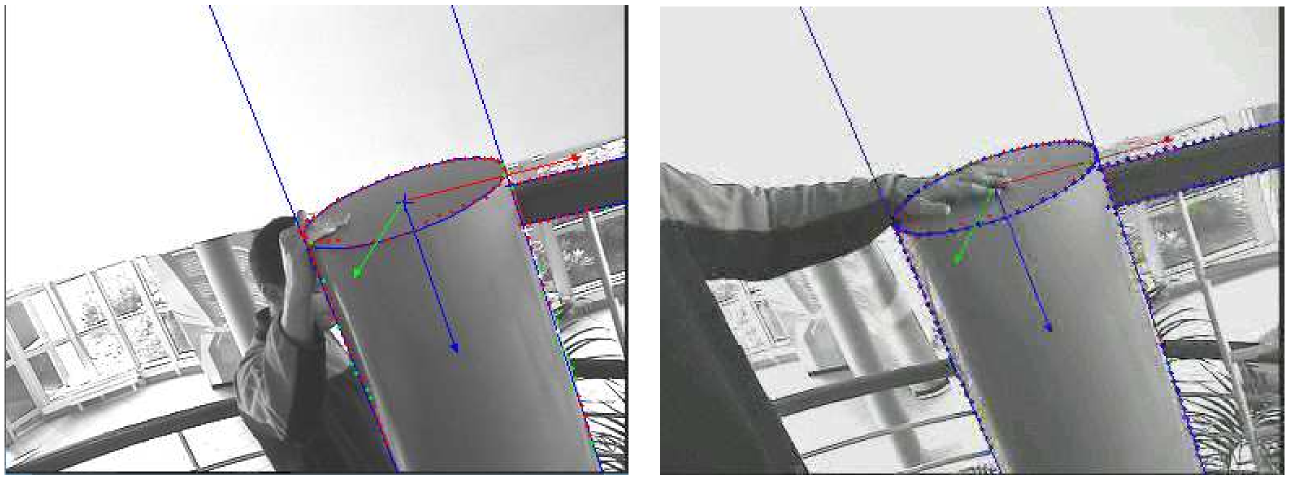


Figure 6.19: Tracking considering a circle, a cylinder and two straight lines with occlusion and severe background noise. This sequence features multiple occlusions of some of the features.

detected by the M-estimator since the person provides a strong contour that is initially within the certainty region and then gradually pulls an entire contour away from the actual position of the model line as they continue to move.

In Figure 6.21 this approach was applied to a real-time augmented reality application. These images are extracted from the sequence after the insertion of virtual objects. Due to both the introduction of the robust estimation and to the high redundancy of visual features, visual alignment is satisfactory to the human eye.

6.4.5 Computational Effi ciency

The computational efficiency was measured by timing the critical inner loop for execution time. The pie chart in Figure 6.22 is the result of an analysis of the computation time occupied by the different functions. These include:

1. It can clearly be seen that the major factor for computational efficiency is the convolution for determining edge locations at 38%. This could be improved by implementing the mask convolution in specialized hardware or in standard graphics cards.

2. Coming in second at 21% is the computation of the velocity twist. This computation involves simple matrix multiplication and the computation of the pseudo-inverse of the interaction matrix in the control law. In order to improve this computation a parallel SVD computation could be considered. Other than this it could be interesting to consider a constant interaction matrix so that the pseudo-inverse is only computed once for each estimation.

3. In equal second are groups all the remaining computation after all the interesting factors had been extracted. This includes various communication between matrices and functions.

4. In third is the computation of the interaction matrix involving several computations for each feature including projection, pixel-to-meter transformation, etc... It can be noted here that it is preferable to use the interaction matrix in its fully derived form as proposed in Chapter 6. More particularly, the

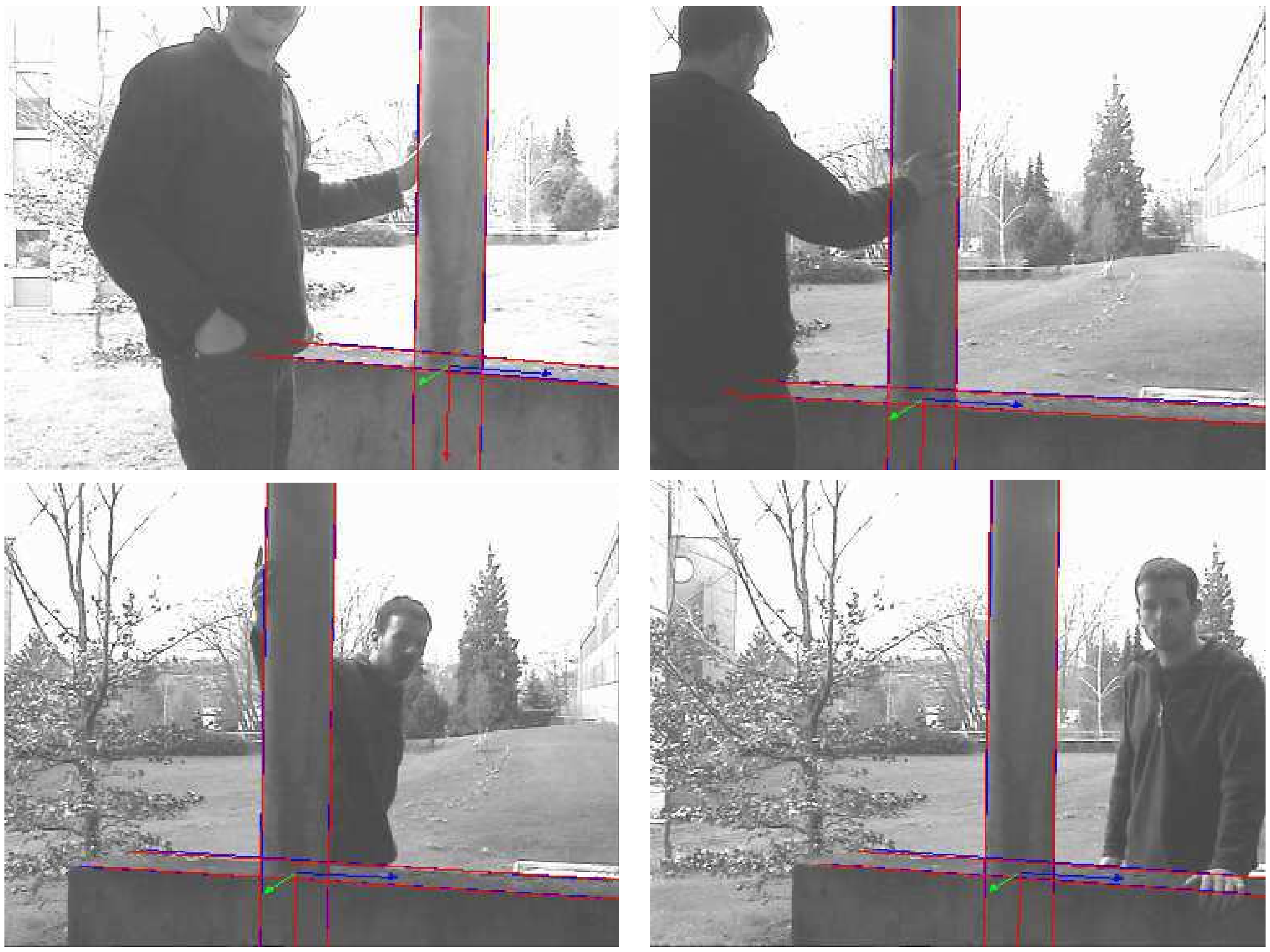


Figure 6.20: Tracking in an outdoor environment. Despite multiple occlusions and disturbances, tracking is still very reliable and handled in real-time.

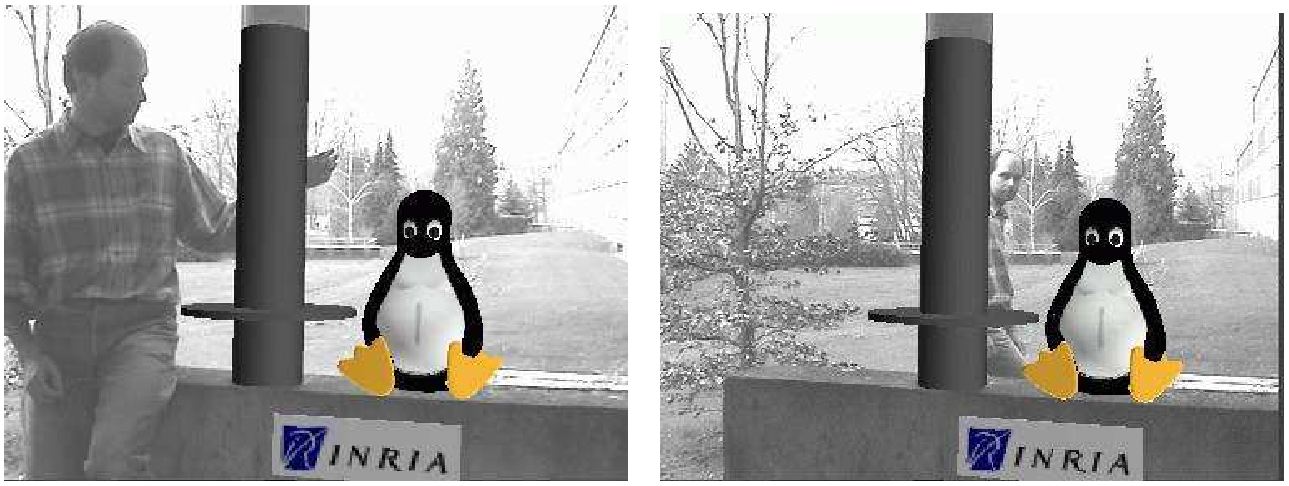


Figure 6.21: Tracking considering a cylinder and two straight lines with application to augmented reality.

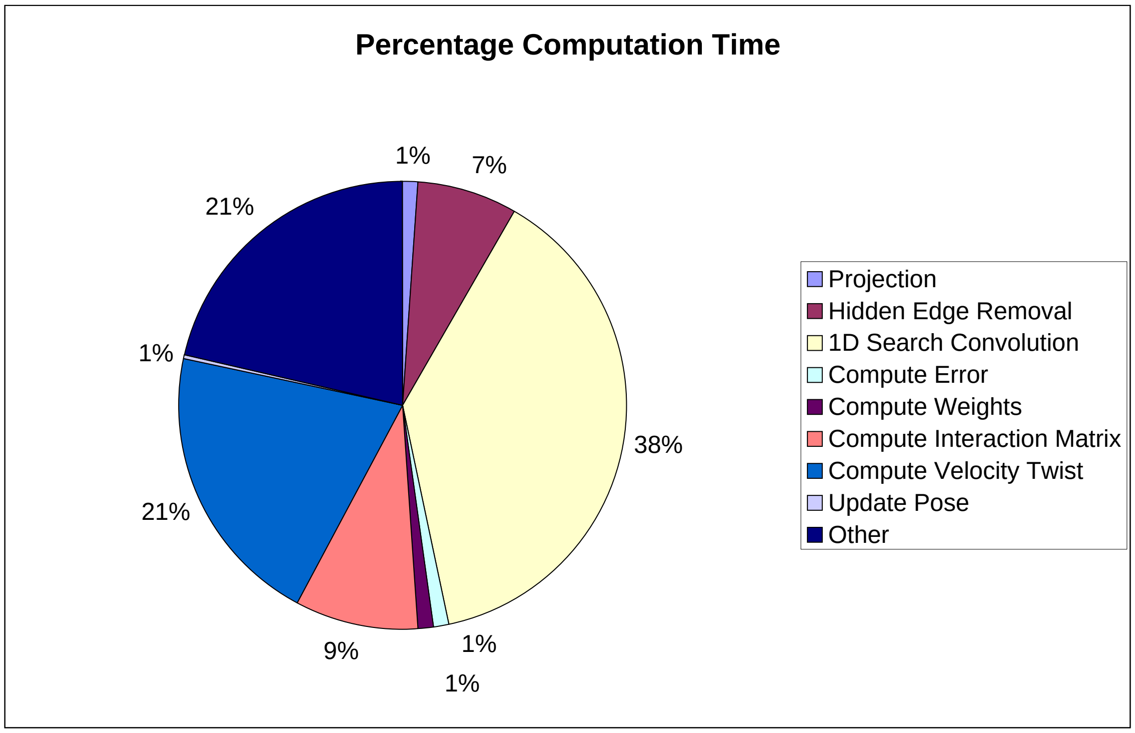


Figure 6.22: Percentage computation time of different parts of the algorithm. Note that the 'other' category contains code such as the contour display or weight display times as well as other code that does not impact on the tracking efficiency.

method give in [Drummond02] was compared to distance-to-line based interaction matrices and it was shown to be more computationally expensive due to the need to calculate the intermediary steps of projection and meters-to-pixel for each generator at each iteration.

5. The last significant computation is at 7% involving hidden surface removal. This code could be optimized somewhat and most standard graphics hardware can be used to compute this.

An augmented reality application for system maintenance

In this example (see Figure 6.23) the 3D tracking algorithm is used to track a domestic or industrial object. A mock AR scenario is presented in Figure 6.23(e-h) for guided maintenance of an air-conditioning system. Many lines with different contrasts along with two circles are considered in the pose computation. This illustrates the capabilities of the algorithm for considering various features within the same minimization process. Large images of size 768×576 pixels are used in the tracking and real-time performance is still obtained. The scenario deals with heavy occlusions and the effects of video interlacing. In the images displayed in Figure 6.23, red dots correspond to inlier data and green dots correspond to the outliers rejected by M-estimation. Forward projection of the model appears in blue. In Figure 6.23(c) a large occlusion is present. One can see that most of the dots on the left part of the object are correctly detected as outliers along with small areas on the bottom right of the object where the model is not perfect. Despite these large occlusions, tracking is correctly handled.

Severe Background Noise

To demonstrate the robustness of the algorithm, a small non convex object was placed in a highly textured environment (an electronic board) as shown in Fig. 6.24. Multiple temporary and partial occlusions were also made by a hand and self occlusions of the object itself were imposed during the tracking task. Despite these difficulties, tracking was correctly achieved and once again images were acquired and processed at video rate.

6.4.6 Robust Tracking for Visual Servoing

A positioning task using a CCD camera mounted on the end effector of a six d.o.f robot has been considered. This application requires both a fast and reliable tracking algorithm. From an initial position, the robot has to reach a desired position expressed as a desired position of the object in the image (depicted in blue in the images). The object of interest in this experiment was a micro-controller device.

To prove the robustness of the algorithm, the micro-controller was placed in a highly textured environment as shown in Figure 6.25. Tracking and positioning tasks were correctly achieved. Images were acquired and processed at video rate (25Hz). Multiple temporary and partial occlusions by an hand and various tools as well as modification of the lighting conditions were imposed during the realization of the positioning task.

The robust 3D tracking algorithm has also been used to track and perform model-based visual servoing. Indeed the tracking algorithm has been widely used at IRISA in visual servoing applications. Other than its robustness to occlusions (due to the integration of M-estimation into the control law), to lighting variations (due to the robustness of the low level ME), to random movements (not compatible with the use of prediction filters such as Kalman), it also provides 3D and 1D localization at a cadence compatible with a closed-loop control law. In Figures 6.26 and 6.27, polyhedral objects have been placed in a highly textured environment. 2 1/2D visual servoing [Malis99] has been implemented. Many partial occlusions (hand, tools, etc) as well

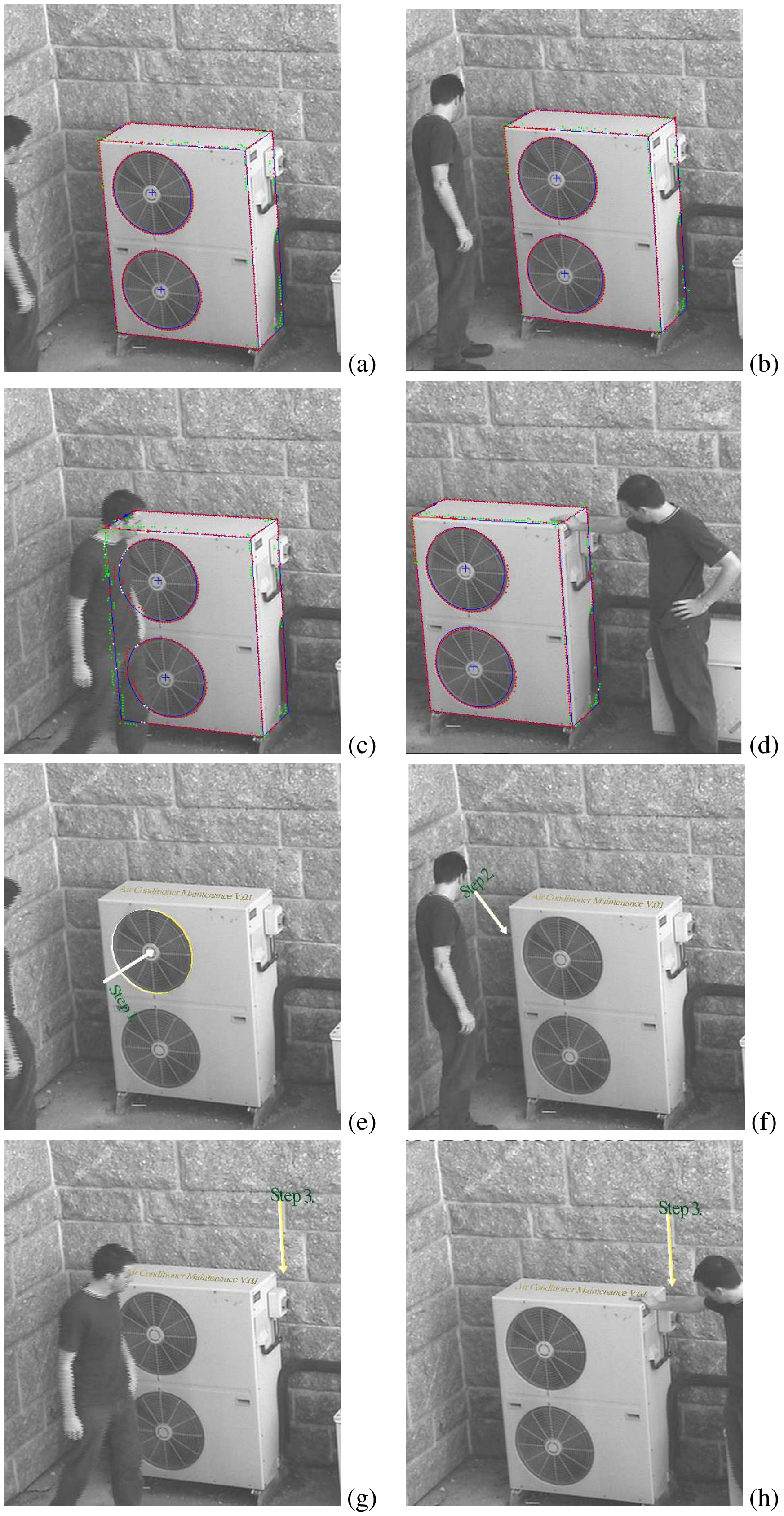

(a)
(b)
(c)
(d)
Air Conditioner Maintenance V.01
Step 1
(e)
Step 2.
Air Conditioner Maintenance V.01
(f)
Step 3.
Air Conditioner Maintenance V.01
(g)
Step 3.
Air Conditioner Maintenance V.01
(h)

Figure 6.23: Tracking and AR for a constructed maintenance scenario. Even with heavy occlusion and disturbances, tracking is still very reliable and handled in real-time. (a-d) Tracking results (e-h) AR sequences corresponding to images (a-d) respectively

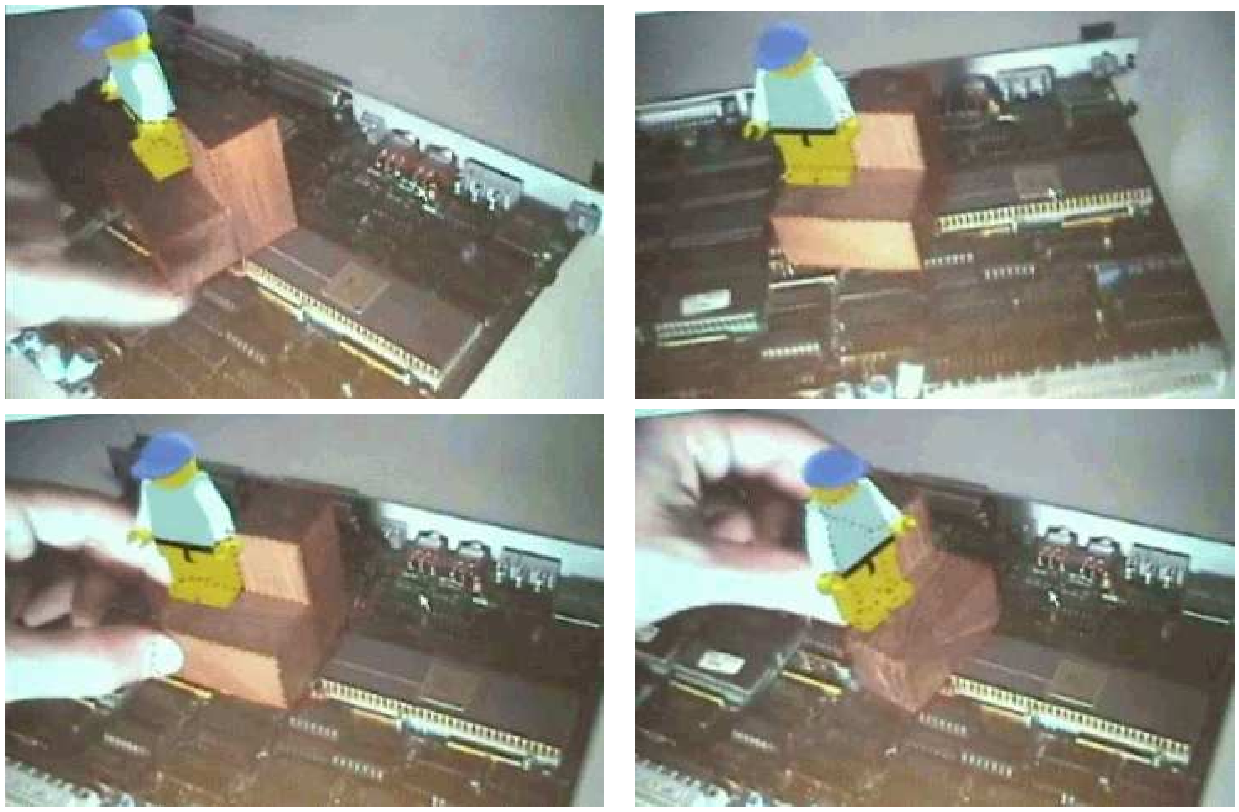


Figure 6.24: Tracking a non convex object on a highly textured environment.

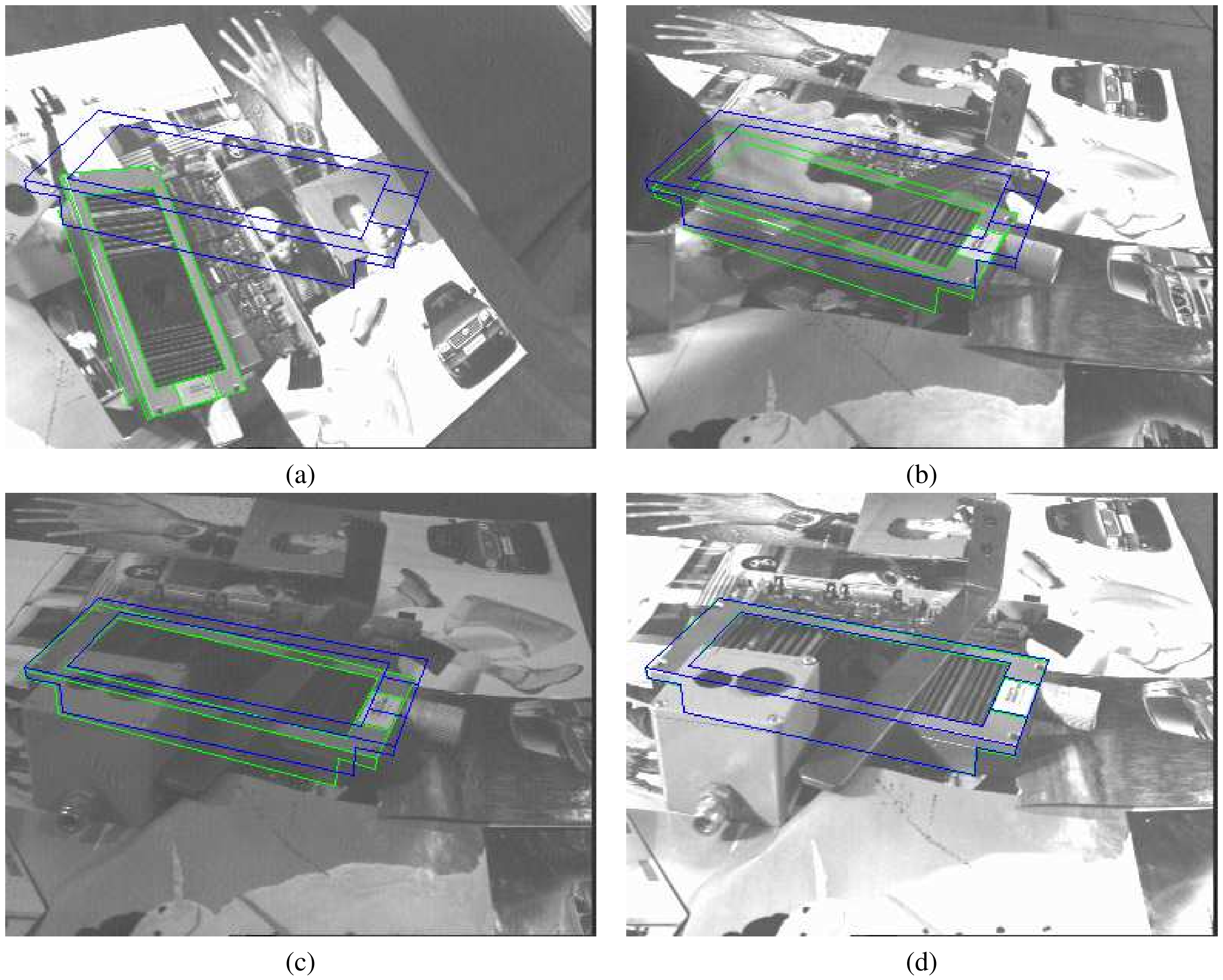


Figure 6.25: Tracking in complex environment within a classical visual servoing experiments: Images are acquired and processed at video rate (25Hz). Blue : the desired position defined by the user. Green : position measured after pose calculation. (a) first image initialized by hand, (b) partial occlusion with hand, (c) lighting variation, (d) final image with various occlusions

as large lighting variations were introduced during the execution of these visual servoing tasks. In all the experiments, the tracking was always performed at a cadence compatible with video rate (often less than 10 ms).

6.5 Conclusion and Perspectives

In conclusion, a method has been presented for robustly tracking complex objects in an image sequence at a high processing rate (50Hz) using visual servoing techniques. High robustness has been demonstrated with a robust model-based approach using an iteratively re-weighted least squares implementation of a M-estimator. The use of a non-linear minimization approach coupled with a redundant number of distance-to-contour visual features leads to efficiency and robustness. The presented method was implemented in an Augmented Reality scenario and was also used for fast and accurate positioning of a eye-in-hand robot with respect to real objects (without any landmarks) in complex situations. The algorithm has been tested on various real scenarios demonstrating a real usability of this approach.

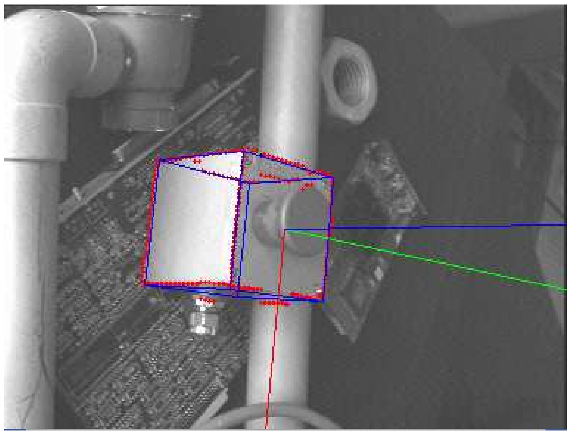
In perspective, it would be interesting to research the initialization and recognition problem. One possible approach could be to use sampling style methods such as RANSAC to initialize correspondences between an image and the 3D CAD model at the beginning of a tracking or visual servoing task. Interest point detectors such as [Harris88] could then be used to extract useful information from the initial image. Globally convergent pose estimation methods such as [Dementhon95, Tahri03] could then be used to find a consensus between different sets of correspondences.

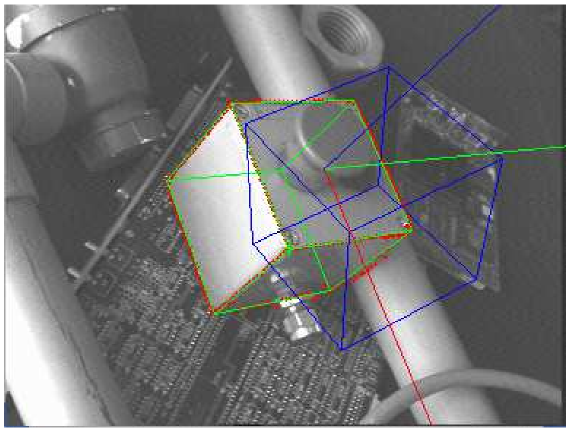
Other techniques have recently become popular based on factored sampling [Isard98a]. In this type of method multiple hypotheses are maintained over time making the estimated state of the system more robust. It would be interesting to explore compatible real-time approaches to this algorithm. Furthermore, this framework also allows for the fusion of multiple cues. Indeed it would be beneficial to consider the local spatial dependence in the observation of visual contours such as in [MacCormick98]. Indeed there are many sources of information available in an image stream such as color which could also improve robustness. Local spatial consistency constraints could also be introduced into a probabilistic framework at the lower level of the estimation process. Primitive likelihood information could also be used to guide the sampling process such as in [Stewart99a, Armstrong95]. In the later primitive stability is considered by varying uncertainty over time, computed as a decaying average of the frequency with which the primitive is found.

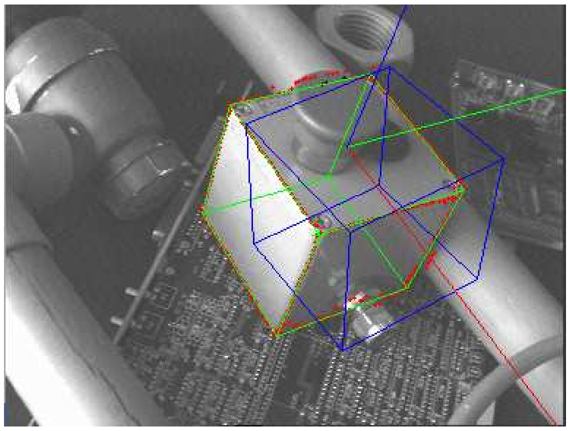
Another avenue for research is based on the fact that orthogonal least squares will produce a sub-optimal estimate because the residuals to be minimized do not have Gaussian distributions [Bookstein79, Sampson82]. The variance is therefore *heteroscedastic* meaning that it depends on the measurement frame of reference. Sampson proposed a first order approximation of the geometric distance. In [Weng89] Sampson's method was adapted to the estimation of the Fundamental matrix and in [Shapiro95, Torr97] outlier detection was considered for orthogonal regression.

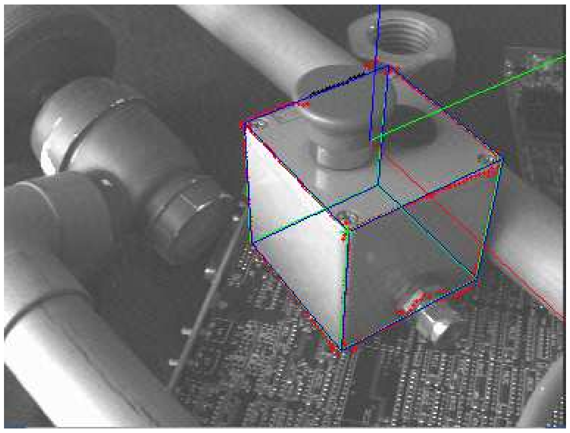
With respect to the propagation of uncertainty, the proposed method is useful to avoid manually setting edge detection thresholds. However, the method presented in this thesis is based on two steps, the first involving feature extraction and the second, pose estimation. It has been shown in [Kollnig95], however, that the global CAD model is estimated directly from image intensity values avoiding the need for an edge extraction step. This method also avoid the requirement of edge detection thresholds.

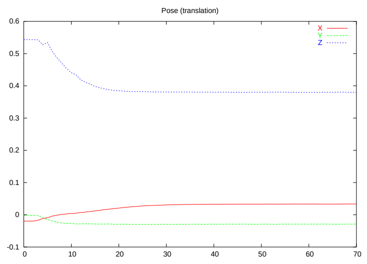
Finally, the scale estimation using the MAD is only a very simplistic method. Indeed, one of the major limitations of traditional methods is that they have relatively low breakdown points which are unsatisfactory in computer vision applications. Current methods are unable to handle multiple instances of the same


(a)

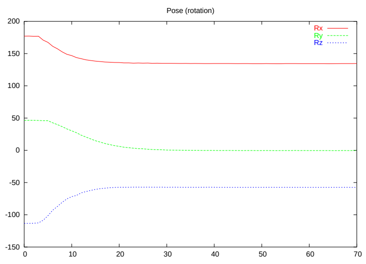

(b)


(c)

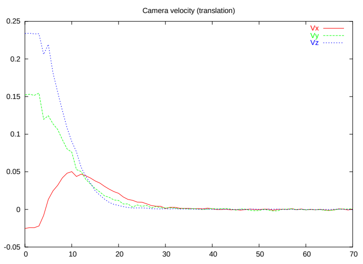

(d)



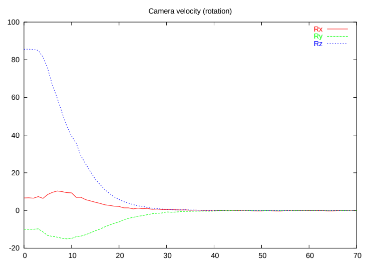
(e)



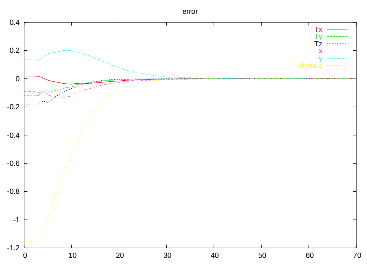
(f)



(g)



(h)



(i)

Figure 6.26: Tracking of an object during a 2 1/2D eye-in-hand visual servoing experiment. (a-d)The object tracked is in green and the desired position of the object is in blue. The red points correspond to the points found with a 1D search to the normal using the modified ME algorithm. Plots correspond to (e) Pose (translation) (f) Pose (rotation) (g-h) camera velocity in rotation and translation (i) error vector $\mathbf{s} - \mathbf{s}^*$

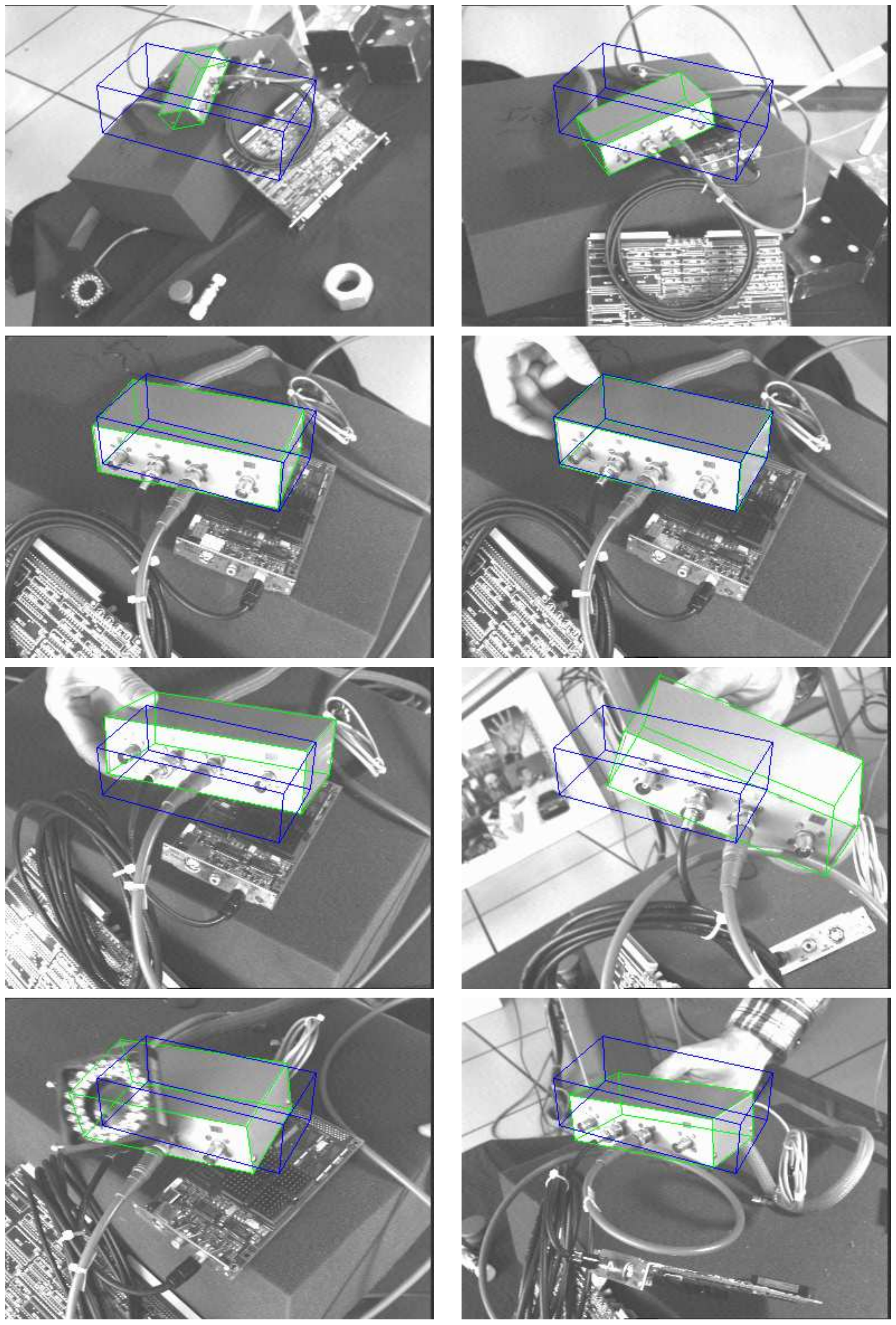


*Figure 6.27: Tracking of an object during a 2 1/2*D *eye-in-hand visual servoing experiment. The object tracked is in green and the desired position of the object is in blue. The first four images correspond to the initial positioning steps of the robot. In the following images, both the object and the robot are moving.*

structure [Meer00]. More recently, the mean-shift procedure originally investigated for pattern recognition [Fukunaga75] has become an interesting way of analyzing the feature space so as to improve traditional robust techniques. In recent computer vision literature, this issue had been popular and relevant work can be found in [Suter04, Wang04a, Wang04b, Comaniciu02, Comaniciu03]. The use of these techniques in visual control would be of great interest.

CHAPTER 7

3D Tracking of Non-rigid Objects

7.1 Introduction

A large amount of research has been invested in tracking objects from a stream of images based on visual information such as contours, color, texture, appearance, shape or combinations of these. In the previous chapters, rigid 3D tracking methods based on a 3D model and contour information have shown to be efficient and robust while remaining computationally efficient. The rigidity assumption, however, largely restricts the use of these techniques to a certain class of applications. Many real-world scenarios that involve non-rigid motion are either impossible to track or tracked poorly using these techniques. This is particularly the case when only limited or rank deficient visual information is available or when strong noise is present in the data. On the other hand, tracking of non-rigid and deformable structures is a relatively new field with few publications focused purely on the aspect of visual perception of these types of objects. In this chapter, a new approach is given for real-time visual tracking of the articulated class of non-rigid objects in 3D. The main idea consists in symmetrically modeling the motion and velocity of an articulated object via a kinematic set approach. The method proposed is a model based approach composed of the geometric structure of rigid components along with mechanical joint definitions. It is shown to be precise and perform in real-time in complex environments.

An "articulated" object is defined as a multi-body system composed of at least two rigid *components* and at most five independent degrees of freedom between any two components. With articulated motion, a non-rigid but constrained dependence exists between the components of an *object*. In the past, articulated motion has been well studied in many domains including geometry, mechanics, and more recently in the computer graphics and animation communities. In computer animation the main interest is *inverse kinematics* and positioning of articulated figures. This is usually achieved via a classical kinematic chain approach [Badler91, Zhao94]. On the other hand in visual tracking of articulated structures the research is based on a *forward kinematics* approach whereby the aim is to determine the positioning of an articulated structure from a view of this structure. This is particularly the case in model-based approaches to tracking where the projection of a known model onto the image requires the use of inverse kinematics [Lowe91]. As will be shown, a tracking approach may involve both forward and inverse kinematics since, in a Virtual Visual Servoing approach it is also necessary to control a virtual model of the real environment.

A taxonomy of non-rigid motion has been given in [Aggarwal94, Aggarwal98] including articulated and elastic motion. Of particular interest are those methods which are based within a computer vision context. Non-rigid motion has been classed into three categories [Huang90] describing different levels of constraints on the movement of a body:

- Fluid - the motion of elements that make up a fluid do not necessarily remain connected creating a discontinuous type of motion and can involve topological variations and turbulent deformation (eg. [Zhou01, Corpetti02]),
- Elastic - motion with a fixed deformation factor involving continuity or smoothness (eg. [Metaxas02])
- Articulated - preserves rigid distances and angles for components but globally motion between components is not entirely rigid (eg. [Comport04a]).

According to these different classes of motion the different approaches to modeling them are wide and varying. At the least constrained end of the spectrum are the methods based on tracking fluids such as in oceanography [Das Peddada96], meteorology [Ottenbacher97] or even medical imagery [Song91]. In this case accurate and precise tracking is extremely difficult due to the number of unknown parameters as well as the observability of pertinent information. As a consequence, techniques designed for estimating this class of motion are approximative and lengthy off-line calculations are common place. Thus the probability of the measures plays a large role in the validation of this type of motion analysis.

On the other end of the spectrum are rigid tracking methods which have already been presented in Chapter 3 which are more deterministic and highly constrained. Other approaches which further restrict the motion include constraining the estimation to a certain number of possible system states of an articulated object by using techniques such as Markov chains [Wu03]. Such an example can be found in the restriction of the motion of a walking person to a gait cycles, where each cycle is composed of a limited set of allowable positions.

In the middle of the spectrum is a group of methods which draw from both extremes to track deformable objects. An example of elastic movement includes the movement of the heart and an example of permanently deformable material is bending of metal. Some approaches to modeling non-rigid motion draw from techniques more adapted to the under-constrained end of the spectrum. The aim is then to constrain the motion so as to obtain an optimal system model for a given application. For example, in [Fua97] hard constraints were applied to deformable snake like models via optimization within orthogonal subspaces. On the other hand, some approaches draw from rigid-body techniques, which can be considered as over-constrained systems, and a bottom up approach would seek to relax these constraints until the model of an ideal system is reached. For example, in [Plankers03] articulated 3D soft objects were defined via implicit surfaces for which rigid constraints were relaxed. There are also those applications which require the unification of the different approaches such as tracking of a human body where deformable tissue or clothing are attached to an underlying rigid skeleton.

In the literature, relatively few recent methods focus on relaxing constraints on physical rigid-body geometric models for articulated motion tracking. In this chapter, certain advantages of these model-based methods such as precision, efficiency, robustness to occlusion etc... are presented within an articulation motion model.

7.1.1 Applications

Motivating the articulated motion model presented previously are a large number of applied domains which range from robotics to human motion analysis and beyond.

In the visual servoing literature, tracking of articulated objects is most interesting for eye-to-hand robot control. In this application, a camera looks externally upon a robot manipulator and the visual information is used to control the robot. The major advantage of this approach to robot control is that it allows a feed-back loop to be easily closed via the image leading to precise and robust results. The main challenge in closing the feedback loop for an eye-to-hand system is to track a robot manipulator and all its articulations in 3D space. The eye-to-hand technique differs from the more common eye-in-hand based visual servoing systems because of the view angle offered by each system respectively. In the eye-to-hand case it is possible to easily have a wide view of both the target and the manipulator. In the eye-in-hand case it is difficult to close the control loop between the target and the manipulator for grasping objects because the camera is mounted on the end-effector. In this case the camera only has a limited view when the end-effector is close to the target. Furthermore, since the camera is mounted on the end-effector and next to the manipulator it is difficult to obtain a view of both the target and the manipulator, thus requiring precise calibration between the camera and the manipulator. The tracking application presented in this chapter is therefore specifically interesting for accurate performance of visually guided object grasping. An example of this approach is [Horaud98] where visual alignment is performed between two solids in three-dimensional (3-D) projective space using an un-calibrated stereo-rig. This method was used for hand-eye coordination of a robotic manipulator. A particularity of this approach was a 3-D projective representation which is view-invariant or independent of camera parameters. Another similar approach was given in [Ruf99a, Ruf99b] where rigid and articulated motion is also modeled and estimated via an un-calibrated stereo rig.

Another major application domain is the analysis of human motion or even animal sequences [Bregler04]. The goal of human tracking is a very interesting and challenging research topic which has been the object of much in depth attention. The problem of human motion tracking is very complex since it requires the solution of many important issues which are yet to be solved making research in this area difficult. As such most methods which have been proposed in the literature are very approximative and less precise than the rigid-body tracking method already proposed in previous chapters. The different challenges include but are not limited to:

- 3D Model: obtaining a geometric description of a 3D model. Human models vary and are not defined by simple geometric objects.
- Low level information: The low level image processing aspect for human subjects is difficult and requires taking into account deformable motion such as the movement of clothing. This problem could be avoided by using silhouettes or skin color for example and approximating the parameter estimation.
- Joint types: Human joints are once again difficult to model but are often approximated by simple linear rotational joints.

With regard to the image processing, human subjects can be very difficult to track locally. To date many common tracking algorithms have been employed such as in [Bregler98], where image processing was performed using an affine motion region tracking. One of the most common techniques is based on the extraction of shape information using silhouette contours such as in [Delamarre99, Plankers03]. In [Metaxas00] spatio-temporal analysis of the subject's silhouette is performed. Other simplifications can be found including simple background subtraction or using skin color.

Silhouette extraction is usually performed with the help of multiple cameras. Indeed in [Delamarre99, Metaxas00] and [Plankers03], multiple views were used to capture the shape and motion of human subjects. In [DiFranco99] another multi-view approach was used for tracking of dancers. In this paper the motion of the figures in the image plane were described by 2D Scaled-Prismatic Model (SPM) and a 3D reconstruction problem was formulated as a batch optimization over a series of SPM measurements. Another approach using 4 cameras along with color blob tracking is found in [Caillette04] where 3D voxel-based reconstruction is performed.

Amongst all the approaches presented here including [Bregler98, Bregler04, Plankers03, Deutscher05], processing is performed off-line in a post-production step and are therefore limited to video based applications. Furthermore, filtering and smoothing is also often used to improve results along with manual user intervention to select key-frames such as in [DiFranco99]. In [Drummond01] real-time results are reported with a single camera, however, in the single camera experiment image processing is simplified by using a stick figure and moving the stick figure off-line to avoid occlusion. The tracker was then allowed to converge in a static situation online before introducing the next movement off-line.

In order to further improve robustness, 3D CAD models are often introduced. In [Metaxas00] upper body extremities are modeled in three-dimensional by a deformable model which is related to the subject's silhouette. In [Drummond01] the intersection of pairs of rigid quadrics are used. In [Plankers03] articulated soft objects are modeled by an implicit surface approach. In this case a distance function is defined between data points and models which can be minimized allowing accurate description of both the shape and the motion using a fairly small number of parameters. The generic soft model is deformed so that it conforms to the image data. The major advantage of these model based approaches is that they permit prediction of the expected location of image features, thereby increasing the robustness.

Amongst all these techniques, however, lies the fundamental problem of articulated motion modeling. Indeed, the articulated motion model forms an underlying basis for tracking of articulated objects. In [Bregler98], people are tracked using twists within a classic kinematic chain, formulated using the product of exponentials formula. In [Plankers03] a classical robotics approach is also used to handle articulated bodies. Other approaches rely on more statistical approaches to apply constraints such as [Metaxas00] where the positions and orientations of each modeled body part of the subject's body are related using a Kalman filter framework or [Caillette04] where a statistical framework is used to update blobs.

In summary, methods proposed in the literature for real-time tracking of human structures have required simplified image-processing, specialized hardware and very approximate pose estimation leading to easy failure. On the other hand it is possible to estimate human movement in an off-line processing step, however, this limits the application domain. Even though this is a very interesting problem, it is important also to create a solid foundation for the construction of such applications. The methods presented in this chapter therefore aim to address the fundamental problem of modeling physical articulated motion and subsequently the techniques presented here would naturally apply to tracking human body motion.

7.1.2 Multiple Cameras

In many of the papers presented previously, results have been improved significantly by using multiple cameras. In this case it is possible to solve both the 3D reconstruction problem and the registration problem simultaneously. Furthermore, with an adequate set of viewpoints, the state space is fully observable meaning that the state estimate will be well conditioned. Apart from those multi-camera methods for human motion tracking, include [Drummond00b] where multiple cameras are used to track manufactured objects and [Ruf99a, Ruf99b] who also tracks manufactured objects using a stereo-rig.

There exists a significant similarity between multi-body motion models and the modeling of multiple cameras which are rigidly linked together. As such most geometric model-based techniques can be easily extended to a multi-view system by simply performing a twist transformation of all camera twists so that they may all be minimized within a common reference frame. As such the multi-camera approach presented in [Drummond00b] was first based on a monocular rigid-body tracking formulation. Another monocular based tracking method was given in [Nickels01] where an *a priori* 3D model was used to provide the missing 3D information. The approach proposed in this chapter is a model based monocular tracking method that can be easily extended to a multi-camera scenario.

7.1.3 Filtering

A group of techniques used for articulated object tracking can be classed as filtering methods. This framework is based on the propagation of probabilistic uncertainty from one time instant to the next and is usually used to make predictions about the future state of the system. Amongst the various filtering methods, different techniques exist. In [Metaxas00] a simple Kalman filter was used, however, this type of filter is not well adapted to non-linear 3D motion equations. More realistic filters such as the extended Kalman filter take into account the non-linearities in the prediction step. In [Borges96] an iterative extended Kalman filter is used to apply constraints between poses with no prior uncertainty. In [Nickels01] the extended Kalman filter is also used to update the state of a 3 degrees of freedom robotic arm including joint positions and all extra degrees of freedom in the model. Each degree of freedom in the model has an uncertainty associated with it, indicating the confidence in the current estimate for that degree of freedom. These uncertainty estimates are updated after each observation. In [Metaxas02] a spatio-temporal probabilistic filtering framework is also used to estimate forces that are computed from an error between the model and the data.

Another approach has been recently presented in [Deutscher05] based on a stochastic search of the configuration space. In this paper, a multi-camera system was used for marker-less human motion capture. The approach presented in this paper was called annealed particle filtering, based on the well known particle filtering framework [Isard96]. This approach initially generates particles within the configuration space of the articulated motion and particle weights are updated according to the measurements observed. This method has been shown to be effective at searching high dimensional configuration spaces (30+ dimensions). The advantages of particle filtering include robust tracking in clutter because it can represent arbitrary functional shapes. Furthermore, less likely model configurations are not discarded immediately but given the possibility to prove themselves later by propagating multiple hypotheses. This therefore makes tracking robust, however, it is also computationally expensive and does not run in real-time.

The use of filtering methods have not been employed in this chapter since there are many situations where movement is unpredictable such as in the case of human motion, or human manipulation of manufactured object. It would be interesting to consider multiple state vectors simultaneously as in the particle filter case, however, this would require a significant improvement in computational efficiency to run in real-time.

7.1.4 Real-time Processing

A large majority of the methods proposed in the literature to date have been focused on post-processing of video sequences due to the heavy computational costs in estimating articulated motion parameters. Indeed most of the papers presented based on human motion tracking are post-production. The reasons for these inefficiencies vary and can be attributed to various factors including estimation techniques, the number of hypotheses tested and low level image processing techniques.

Many of the algorithms that are reported in the literature to perform in real-time make reference to different benchmarks. More often than not, real-time efficiency refers to the requirement of a particular application. For example in [Caillette04] real-time is reported at around 15fps whereas for applications considered in this thesis, real-time is taken to be at around a frame-rate of around 30fps.

Approaches based on filtering techniques have proven computationally expensive including [Borges96, Metaxas02, Deutscher05], where in the later case, computational cost can be attributed to the simultaneous treatment of multiple high level hypotheses. More recently, however, with advances in articulated object modeling along with computer processing power, articulated object tracking has become more efficient and real-time performance has been demonstrated using filtering techniques such as in [Drummond01], however, simplified image processing was required.

The advantages of a real-time processing goal are numerous and include a much wider application domain. Furthermore, this goal forces the design of such methods to be focused on the essential factors required to estimate articulated motion. In the 3D rigid tracking results given in previous chapters, it has been shown that the largest computational cost comes from the low level image processing and feature extraction stages as well as the computation of the matrix inverse within the global estimation procedure. Thus an efficient 1D search to the normal of the projected contour in the image is an important factor for real-time performance.

Therefore, amongst the various real-time algorithms proposed for articulated motion tracking, the large majority are based on a known CAD model. The use of a CAD model allows to globally constrain the search space of possible solutions leading to more efficient 1D searches. In this way the low level image processing stages are more efficient such as the 1D search presented in Chapter 4. Even so some model-based approaches have also required off-line processing such as in [Nickels01]. Here poor efficiency was attributed to expensive low level tracking based on the Sum of Squared Difference of image regions.

7.1.5 CAD and Motion Models

As has been discussed already in Chapter 3, there exists a full range of information that can be used in a 3D tracking context. In summary, this information can be obtained in various ways. First of all it is possible to provide *a priori* information to an estimation process. This *a priori* information is associated with a CAD model. On the other hand it is possible to estimate unknown parameters online. Indeed various degrees of *a priori* and unknown information is present in tracking methods presented in the literature. Another approach is to add more information via different measurements such as multiple cameras or different types of sensors. For the same reasoning as in the rigid-body case, in this chapter a complete 3D CAD model is used due to greater robustness and computational efficiencies. Furthermore, knowing the object's structure helps to predict hidden movement, which is particularly interesting in the case of articulated objects because there is an increased amount of self-occlusion. Knowing the model also allows an analytic relation for the system dynamics to be more precisely derived. It is also possible to consider parameters such as elastic constants and material types within the CAD model.

In the articulated motion tracking literature various approaches have been made either with or without an *a priori* model of the object. Alternatively to the rigid-motion model, non-rigid motion depends on further parameters which define the relationship between rigid components. In the particular case of articulated motion, this extended model includes the type and position of mechanical joints. This type of model has been used for a long time in the robotics and mechanics literature. Since the CAD model must be obtained at some point in time, *a priori* knowledge can also be associated with off-line processing techniques. *a priori* knowledge the encompasses many different types of information including joint positions, joint types, contours, texture, camera calibration parameters etc...

In [Lowe91] a CAD model composed of lines is combined with articulated joint parameters as well as camera calibration parameters. In [Drummond02] the CAD model is also composed of 3D contours, articulated joint parameters and camera calibration parameters. In [Ruf99a, Ruf99b] point features are combined with articulated joint features, however, the model is simplified since a stereo rig is used to provide unknown depth information online. An advantage of this approach is that it is invariant to camera calibration parameters since it was implemented using projective geometry whereas many other approaches need estimates of the camera calibration parameters. In [Nickels01] an appearance model is provided along with the kinematic structure.

On the other hand, if certain parameters cannot be given *a priori*, it is often necessary to estimate them online. In [Lowe91] rigid pose parameters were estimated simultaneously with the length of object segments. In [Marchand01] it was shown that camera calibration parameters could be estimated online. Another example is [Metaxas02] where a deformable model's elastic parameters are estimated simultaneously with unknown forces related to a physics based geometric model. Deformable models include implicit models.

Amongst the various parameters that are used to define the motion of an object, different mathematical models exist which relate these parameters to the measurements. In the articulated motion case several techniques are presented here. In [Lowe91, Bregler98, Ruf99a, Ruf99b, Nickels01] articulated object tracking was performed using kinematic chains in a tree structure. In [Delamarre99, Drummond00b] a similar algorithm is applied to multiple articulated structures. Poses are first computed independently for individual components and constraints are then imposed using Lagrange multipliers. These techniques will be analyzed in more detail in Sections 7.4.1 and 7.4.2.

In summary, the monocular articulated tracking method proposed in this chapter is based on the previously presented results including local tracking of distance-to-contour features as well as robust statistics. In a first part, an overview of the objective of articulated object tracking is made. In Section 7.3 a articulated joint motion model is proposed for a general class of articulated objects by using velocity constraints. In Section 7.4 several different methods for modeling the a minimal set of articulated object motion parameters is presented. These methods include kinematic chains, Lagrange multipliers and a new approach called kinematic sets. Finally, the closed loop control law is derived in Section 7.5 aimed at minimizing a set of distance errors in the image and estimating the system parameters. In the results, the proposed articulated tracking system will be shown to be efficient and precise leading to real-time performance (frame-rate) and accurate measurements. Experimental results also show prismatic, rotational and helical type links and up to eight general parameters.

7.2 Articulated Tracking Overview

As in the rigid-body tracking presented in Chapters 3 4 5 6, the extension of this method to a new class of articulated object is also based on an *a priori* CAD model. The new model includes *a priori* information about an articulated object which is composed of rigid-body components as well as joints. 3D parametric equations describe the structure of a rigid component with respect to an object reference frame $\mathcal{F}_o$. Articulated objects are represented as rigid components(or feature sets) with joints that link the components together. Each joint has an associated reference frame $\mathcal{F}_i, \forall i = 1, ..., l$, where l is the number of joints, and an associated joint type. The representation of a joint or link is a central part of this model and it will be derived in detail in Section 7.3. In a first instance an overview of the approach will be presented.

Apart from an *a priori* model is the motion model which is to be estimated online and in real-time. The articulated object motion model is first based on rigid body differential geometry. Each component of an articulated object is individually defined by a full rigid-body pose $\mathbf{r} \in \mathbb{R}^6$, originally defined in (2.10). In summary, the Special Euclidean Group, $SE(3)$, is the configuration space of a rigid body and is also known as a Lie Group. An element of $SE(3)$ is referred to as a pose or location and belongs to a 6-dimensional differential manifold. A rigid-body pose is defined so that Euclidean transformations preserve distances and orientations of object features with respect to one another. In order to estimate the pose of a rigid object from visual information, rigid-body velocities are required. An element $\mathbf{v} \in se(3)$ is known as a twist as defined previously in (2.27). Finally, rigid-body velocities $\mathbf{v}$ are directly related to rigid-body poses $\mathbf{r}$ by Rodriguez's exponential map (2.59). The Lie Group is defined so that Euclidean transformations preserve distances and orientations of object features with respect to one another.

The objective of the articulated object tracking approach is to maintain an estimate of a set of minimal parameters describing the configuration of a 3D articulated object in Euclidean space. This is defined by the vector $\mathbf{q} \in \mathbb{R}^m$. This is a minimal vector (or set) of m parameters describing the configuration of an articulated object in $SE(3)^\kappa$, where κ is the number of rigid components making up an articulated object. Instead of modeling articulated motion directly using the pose it will be shown that a more general approach may be taken by modeling the velocities of an articulated object. Rigid components are then related to one another by constraining the allowable velocity twists of each rigid component (Section 7.3.2). In the definition made here, the space of articulated pose is defined so as to incorporate all different types of articulated objects including the case where both components are completely independent, however, in practice this space is configured to contain an explicit articulated object and is usually much smaller.

Thus, non-rigid bodies, or articulated objects, belong to a m-dimensional configuration space $\mathbf{q} \in SE(3)^\kappa$ in which the extra degrees of freedom describe internal parameters allowing intra-object movement. It is a m-dimensional differential manifold. The tangent space of velocities is denoted $se(3)^\kappa$. In this chapter, the main focus of modeling articulated motion will be based on the tangent space. An element of this generalized configuration space, $\dot{\mathbf{q}} \in \mathbb{R}^m$, is defined as:

$$\dot{\mathbf{q}} = (\dot{q}_1, \dots, \dot{q}_m), \tag{7.1}$$

For this generalized configuration space, the Lie Group defines Euclidean transformations which preserve different *subsets* of distances and orientations of object features. The subsets of velocity parameters within this manifold are also related to their Special Euclidean counterpart by Rodriguez's exponential map.

In order to maintain an estimate of $\mathbf{q}$, the underlying idea is to minimize a non-linear system of equations so that the projected contour of the object model in the image is aligned with the actual position of the contours in the image. To perform the alignment, an error Δ is defined in the image.

$$\Delta = \Big(\mathbf{s}(\mathbf{q}) - \mathbf{s}^*\Big) \tag{7.2}$$

where $\mathbf{s}(\mathbf{q})$ is a vector of feature parameters projected from a 3D model onto the image and $\mathbf{s}^*$ is a vector of corresponding features found by a local tracking procedure in the image (desired features). In this Chapter distance features are also used (see Chapter 5).

Once again, a monocular *camera sensor* is used where the center of projection is referenced by the camera reference frame $\mathcal{F}_c$. The formation of the image is a standard perspective projection model. The error in equation (7.2) can be written as:

$$\Delta = \Big[pr(\mathbf{q},{}^o\mathbf{S}) - \mathbf{s}^*\Big], \tag{7.3}$$

where ${}^o\mathbf{S}$ are the 3D parameters of any type of visual feature associated with an object which are expressed in the object frame of reference o. $pr(\mathbf{q},{}^o\mathbf{S})$ is the camera projection model according to the generalized pose parameters $\mathbf{q}$. The features of each component are projected onto the image using their associated camera poses ${}^c\mathbf{r}_{\mathcal{F}_i}(\mathbf{q})$ where each component's camera pose is composed of a 6 dimensional subset obtained from the parameters $\mathbf{q}$.

For the very first image at t_0, the parameters of the object are initially needed and they can be computed using different globally convergent methods. These include manually making the correspondence of four point features with the model [Dementhon95], a manual manipulation of the object projected on the image (using the mouse) or an automatic procedure in the case of textured objects [Lowe04]. The initial estimate is then refined using the Virtual Visual Servoing [Comport03a, Marchand02a] non-linear iterative minimization approach. Following an approximate initialization procedure, the *closed-loop* tracking procedure proposed in this chapter takes over. Since a closed-loop method is employed, any errors in the initialization procedure are eliminated. As mentioned already, the advantages of using a non-linear minimization procedure include accuracy, robustness, and real-time efficiency. Although non-linear iterative methods are not globally convergent, they are ideal for tracking since any inter-frame movement of the object is relatively small. Even if large movements are observed within the image, the estimation process still converges since a large cone of convergence exists around the previous solution. A proof of convergence is given in Section 7.5.

As presented in Chapter 6, in order to render the minimization of these errors more robust, they are minimized using a robust approach based on M-estimation techniques.

$$\Delta_{\mathcal{R},i} = \rho\Big(s_i(\mathbf{q}) - s_i^*\Big), \tag{7.4}$$

where $\rho(u)$ is a robust function [Huber81] that grows sub-quadratically and is monotonically nondecreasing with increasing $|u|$. As in the rigid tracking case, Tukey's function is used because it allows complete rejection of outliers.

This is integrated into an iteratively re-weighted least squares (IRLS) minimization procedure so as to render those errors at the extremities of the distribution less likely.

7.3 3D Non-rigid Geometry Model

In this section, an *a priori* CAD model for an articulated object is defined in detail. As mentioned previously in Chapter 3 a 3D CAD model provides many advantages to a real-time tracking problem over those approaches which assume no prior knowledge is available. Indeed, this allows many interesting results to be obtained including estimations which are robust to many sources of error and high computational efficiency. Furthermore, in many modern applications prior information is readily available such as in the case of CAD diagrams for manufactured objects. Therefore, in this section the aim is to extend the definition of a rigid geometric model to an articulated model.

It is first necessary to define clearly what assumptions have been made and to define clearly the problem. As mentioned already, articulated motion is defined by rigid components which are linked together by joints. The rigid component model is defined as in the rigid tracking case, with the geometric structure known *a priori*. In addition to this, it is also assumed that the mechanical relationship between the different components, as well as the location of the joint, are also known *a priori*.

In Section 7.3.1 rigid-object tracking is revisited and a notation is introduced for dealing with multiple objects. Following this, constraint matrices are introduced with the aim of constraining both the full pose and velocity of a rigid object. The configuration and location of a joint is then used to build a general Jacobian Matrix which relates individual rigid body velocities (twists) to an underlying minimal subspace.

7.3.1 Multiple Objects

In general, the methods which have been proposed in the past for articulated object tracking rely on a good rigid tracking method. In this treatment the rigid-body tracking method presented in this thesis is therefore extended to include multiple rigid components. In this way the articulated motion model is also considered within the context of Virtual Visual Servoing. This duality, has been explained in depth for rigid objects, however, articulated objects are slightly different. In articulated object tracking, minimizing a visual error in the image can be considered as analogous to controlling an arm-to-eye system so as to observe the arm at a given position in the image. In the articulated object case, however, the base of the object (robot) is not necessarily fixed to the ground.

The rigid 3D CAD model is now redefined to include multiple objects with their own corresponding visual features. In this way, each cluster of rigidly linked 3D visual information is associated with a unique reference frame in space. This is known as the component reference frame. Thus the definition of multiple objects require certain extra notation. The error in the image of the two components are defined *separately* as follows:

$$\begin{pmatrix} \Delta_1 \\ \vdots \\ \Delta_\kappa \end{pmatrix} = \begin{pmatrix} \mathbf{s_1}({}^c\mathbf{r}_{\mathcal{F}_1}) - \mathbf{s^*}_1 \\ \vdots \\ \mathbf{s}_\kappa({}^c\mathbf{r}_{\mathcal{F}_\kappa}) - \mathbf{s^*}_\kappa \end{pmatrix} \tag{7.5}$$

where the visual features $\mathbf{s}$ of each component $i \le \kappa$, depend on the pose $\mathbf{r}$ between the camera and each individual component i.

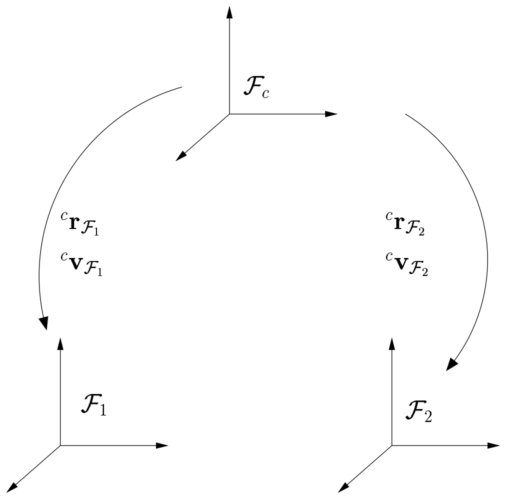

$\mathcal{F}_c$
${}^c\mathbf{r}_{\mathcal{F}_1}$
${}^c\mathbf{v}_{\mathcal{F}_1}$
${}^c\mathbf{r}_{\mathcal{F}_2}$
${}^c\mathbf{v}_{\mathcal{F}_2}$
$\mathcal{F}_1$
$\mathcal{F}_2$

Figure 7.1: Case 1: Pose calculation for two components

Thus differentiating equation (7.5) with respect to time and using a Taylor series expansion as in Chapter 3 gives:

$$\begin{pmatrix} \dot{\Delta}_1 \\ \vdots \\ \dot{\Delta}_\kappa \end{pmatrix} = \begin{pmatrix} \frac{\partial \mathbf{s_1}}{\partial {}^c\mathbf{r}_{c^*}} \frac{d\,{}^c\mathbf{r}_{c^*}}{dt} & \cdots & \mathbf{0}_{n_1 \times 6} \\ \vdots & \ddots & \\ \mathbf{0}_{n_\kappa \times 6} & & \frac{\partial \mathbf{s}_\kappa}{\partial {}^{c_\kappa}\mathbf{r}_{c^*_\kappa}} \frac{d\,{}^{c_\kappa}\mathbf{r}_{c^*_\kappa}}{dt} \end{pmatrix} = \begin{pmatrix} \mathbf{L_{s1}}{}^c\mathbf{v}_{\mathcal{F}1} & \cdots & \mathbf{0}_{n_1 \times 6} \\ \vdots & \ddots & \\ \mathbf{0}_{n_\kappa \times 6} & & \mathbf{L}_{\mathbf{s}_\kappa}{}^c\mathbf{v}_{\mathcal{F}_\kappa} \end{pmatrix} \tag{7.6}$$

where :

- $\mathbf{L}_{\mathbf{s}i}$ is the interaction matrix of dimension $n_i \times 6$ of component i. It relates the change of the image features $\mathbf{s}_i$ to the velocity twist between the camera coordinate frame c and the desired camera coordinate frame c^*.
- n_i is the number of features of component i or the dimension of $\mathbf{s}_i$.
- ${}^c\mathbf{v}_i(\boldsymbol{v}, \omega)$ is the velocity twist between camera and a component composed of the velocity of translation and rotation vectors.

The separate poses are easily determined using a visual control law for each individual component as previously given in equation (3.37). The velocities are converted into poses using the exponential map and the pose between the camera and the object can be computed using the initial pose ${}^c\mathbf{r}_{\mathcal{F}_i}$ for object coordinate frame $\mathcal{F}_i$. It can be noted that in this multi-object model the individual poses, related to each component, are completely decoupled and the number of estimated parameters is $\kappa \times 6$. In this form the parameters estimated are not minimal.

The Lagrange Multiplier approach given in Section 7.4.2 uses this individual estimation of parameters. Alone, the approach presented here is very simple and easy to implement, however, as will be shown in the results, individual tracking can be unstable and not very robust in extreme situations. In the following sections, the minimal form of the pose parameters will be modeled by constraining the rigid formulation presented here. As will be shown, this leads to improved computational efficiency as well as more precise estimation of parameters.

7.3.2 Constraint Matrices

There are two main avenues one may take to define a mechanical joint. The first is to derive analytically the mechanical configuration of each joint separately according to the type of parameters defined within that joint. The second method is to define the space of all joints and *constrain* this space so that it represent the required configuration. The later method is interesting because it is based on a general definition of the entire configuration space rather than requiring analytical derivation for each different object configuration. Furthermore, the 3D configuration spaces $SE(3)$ and $se(3)$ have already been defined for rigid objects. Therefore, in this section, constraint matrices will be considered for constraining these spaces.

The aim here is to first present the mathematical concepts behind constraining a linear system of equations with the final aim being to define a general class of mechanical joints. In general, a constraint matrix is used to define two subspaces of a vector space. A single constraint matrix defines two subspaces since each of these subspaces is defined so that the Image of one subspace coincides with the Kernel of the other and vice versa [Edwards95]. As will be shown later, these subspaces can be easily used to define the configuration of an articulated joint.

In the next section, pose constraint matrices will be considered and in the following section velocity constraint matrices will be considered.

Pose Constraints

Since the Virtual Visual Servoing concept provides a direct analogy with visual servoing, it is possible to consider constraint techniques which have already been formulated for visual servoing. In visual servoing, *visual* constraints have been considered for achieving hybrid visual servoing tasks [Samson88, Samson91, Chaumette93]. Hybrid visual servoing aims at combining a principal task with a secondary one by constraining the second task so that it does not interfere with the first. An example being a road following task whereby the principal task is to minimize the error between the parallel lines found on a road, while the second being to advance along the road. Traditional visual servoing constraint techniques fall into the first class of pose constraints presented in this Section.

In visual servoing, a hybrid task is composed of two tasks which are specified by task or error vectors $\mathbf{e}_1(\mathbf{q}, t)$ and $\mathbf{e}_2(\mathbf{q}, t)$, where these are the *principal* and *secondary* tasks respectively and $\mathbf{q}$ is the robot joint vector. In visual servoing $\mathbf{e}_1$ is defined by visual features in the image and $\mathbf{e}_2$ is defined by a secondary contraint. The primary task is fully defined so that $m \leq n$ independent components exist. For this visual task constraints are defined by carefully selecting visual information for the first task that leaves certain degrees of freedom uncontrolled. In this way the principal task is defined so that it leaves free the range of allowable parameters to be controlled by the secondary task $\mathbf{e}_2$. The free parameters therefore correspond to movements which do not create any error in the image for the visual features that correspond to the principal task. In this way, the constraints are considered within the task space of $\mathbf{e}$. The downside of this approach is that it is necessary to find visual features that correspond to the desired constraint.

Following on, once the principal task has been well chosen it is necessary to combine it with the secondary task in a redundancy framework. Therefore the main task is chosen to be of dimension m while the secondary task is defined by a cost function h_s. The different choices of cost functions can be found in [Samson91]. Minimizing h_s under the constraint that $e_1 = 0$ requires the subspace of motions left free by this constraint to be determined. This is equivalent to knowing the null-space of the interaction matrix (Jacobian) $\mathbf{L}_1$ for the principal task $(Ker(\mathbf{L}_1))$, which is equivalent to knowing the range of $\mathbf{L}_1^\top$. Therefore it is necessary to define a $m \times n$ subspace projector from the Jacobian of the principal task as:

$$\ker\mathbf{W} = \ker\mathbf{L}. \tag{7.7}$$

The subspace projector can then be used as in [Samson88, Samson91] to define the hybrid task to be minimized as:

$$\mathbf{e} = \mathbf{W}^{+}\mathbf{e}_1 + \beta(\mathbb{I}_6 - \mathbf{W}^{+}\mathbf{W})\frac{\partial h_s^\top}{\partial \mathbf{r}}, \tag{7.8}$$

where

- β is a positive scalar which determines the priority of task 1 over task 2,
- $\mathbf{W}^{+} = (\mathbf{W}^\top\mathbf{W})^{-1}\mathbf{W}^\top$ is the pseudo-inverse of $\mathbf{W}$,
- $\mathbb{I}_6 - \mathbf{W}^{+}\mathbf{W}$ is an orthogonal subspace projection operator of the Kernel of $\mathbf{W}$.

In the mechanics literature, constraints of this form are of common use and it is possible to consider a general holonomic constraint which restricts the motion of a rigid body to a smooth manifold. In this case, a holonomic constraint is represented locally as an algebraic constraint on the configuration space [Murray94]:

$$e_h(\mathbf{r}) = 0, \qquad h = 1, ..., d. \tag{7.9}$$

where h indexes the *class* of the constraint which is of dimension d. This matrix constrains the pose $\mathbf{r}$ which is the configuration space of a rigid object.

Each function $e_h(\mathbf{r})$ is a mapping from the configuration space Q to $\mathbb{R}$ which restricts the motion of the system. It is assumed that the constraints are linearly independent giving the full rank matrix:

$$\frac{\partial e}{\partial \mathbf{r}} = \begin{bmatrix} \frac{\partial e_1}{\partial r_1} & \cdots & \frac{\partial e_1}{\partial r_6} \\ \vdots & \ddots & \\ \frac{\partial e_d}{\partial r_1} & & \frac{\partial e_d}{\partial r_6} \end{bmatrix}. \tag{7.10}$$

where it is assumed that the unconstrained system is of dimension $d \times 6$, where d is the dimension or class of the constraint and 6 is the dimension of the pose vector.

The difficulty of this first approach is that it requires algebraically defining the space of joint positions by defining allowable positions. This was achieved in the visual servoing case by geometrically defining visual information to constrain this space. As will be shown in the next section, the form of the constraint can be made much more simple by using velocity constraints. In Table 7.2, various different classes of configuration types are given.

Velocity Constraints

In this section a formalism will be presented for applying velocity constraints to twists. It will be shown that this constraint formalism gives a more general definition than the constraints expressed in the task space. The definitions provided will form the basis for modeling articulated motion, however, they also provide much potential for application to visual servoing as well as other computer vision problems.

Velocity constraints are widely known in the classical mechanics literature as Pfaffian constraints [Rosenberg77]. A velocity twist is constrained as follows:

$$\mathbf{S}^{+}(\mathbf{r})\mathbf{v} = 0, \qquad \mathbf{S}(\mathbf{r}) \in \mathbb{R}^{d \times dim(\mathbf{v})}, \tag{7.11}$$

where d is the class of the joint and the dimension of $\mathbf{v}$ is 6 in the case of a velocity twist.

The constraint matrix $\mathbf{S}$ represents d velocity constraints and $\mathbf{S}^{+}$ is the pseudo-inverse of $\mathbf{S}$. It is assumed that constraints are point-wise linearly independent meaning that $\mathbf{S}(\mathbf{r})$ is full row rank at $\mathbf{r} \in SE(3)$.

As shown in Chapter 2, there exists a natural duality between the space $SE(3)$ of the pose parameters and the tangent space of velocities $se(3)$ via the exponential map. Thus it is easy to pass from a twist to a pose with ease. Likewise, according to the definition of a Pfaffian constraint, if it is holonomic, it is therefore integrable and it can be used to determine an algebraic constraint in $SE(3)$ as presented previously.

A velocity constraint is defined to act on the range of allowable movements for a particular configuration. It is defined as a projector onto a standard basis that represents the type of constraint. The constraint matrix is given in transposed form w.r.t equation (7.10) as:

$$\mathbf{S}_j^{\perp} = \begin{bmatrix} \frac{\partial S_1}{\partial r_1} & \cdots & \frac{\partial S_d}{\partial r_1} \\ \vdots & \ddots & \\ \frac{\partial S_1}{\partial r_6} & & \frac{\partial S_d}{\partial r_6} \end{bmatrix} = \left\{ \begin{matrix} \frac{\partial S_1}{\partial r_1} & \frac{\partial S_2}{\partial r_1} & \frac{\partial S_d}{\partial r_1} \\ \vdots & \vdots & \vdots \\ \frac{\partial S_1}{\partial r_6}, & \frac{\partial S_2}{\partial r_6}, & \frac{\partial S_d}{\partial r_6} \end{matrix} \right\}, \tag{7.12}$$

where this is a holonomic constraint matrix and is defined such that each column vector defines one degree of freedom at the corresponding constraint. Once again, the particular case of motion in $se(3)$ is considered so that the dimension of $\mathbf{v}$ is 6, however, this dimension may be different for other applications. This constraint applies to the range of allowable velocities defined by the rigid twist vector.

As mentioned previously, the number of non-zero columns of $\mathbf{S}_j^{\perp}$ is referred to as the *class* d of the link. The rows of a column define the type of the link by defining which combination of translations and rotations are permitted as well as their proportions.

Let $\mathbf{S}_j^{\perp}$ be the Image matrix subspace so that its corresponding Kernel subspace is given by (with abuse of notation):

$$\mathbf{S}_j = Ker((\mathbf{S}_j^{\perp})^{\top}). \tag{7.13}$$

where $\ker\mathbf{S}^{\perp} := \{\mathbf{v} \in se(3) : \mathbf{S}^{\perp}\mathbf{v} = 0\}$.

Thus, a velocity constraint matrix has been defined which constraints a general class of rigid configurations by constraining sets of allowable velocities. Each constraint can be easily defined according to the desired geometrical behavior of a rigid object.

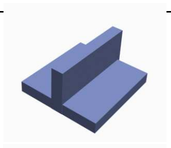
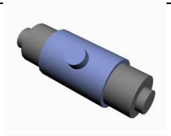
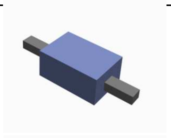
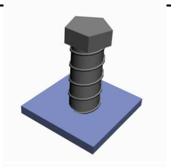
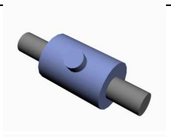
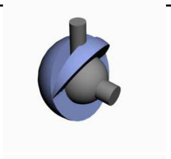
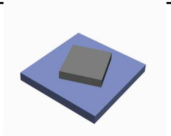
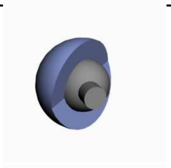
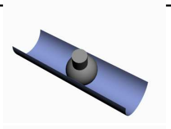
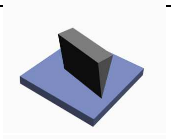
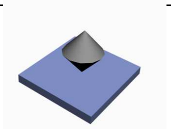
Type of Joint	**Degrees of Freedom** (Class)	**Translation** (T_x, T_y, T_z)	**Rotation** $(\Omega_x, \Omega_y, \Omega_z)$	**Diagram**
Rigid	0	0	0	
Revolute	1	0	1	
Prismatic	1	1	0	
Helical	1	$\frac{1}{a}$	a	
Cylindrical	2	1	1	
Constrained Ball Joint	2	0	2	
Planar	3	2	1	
Ball Joint	3	0	3	
Sliding Ball Joint	4	1	3	
Rectilinear Sliding Wedge	4	2	2	
Spindle	5	2	3	

Figure 7.2: Different classes of mechanical joints, where a is the ratio of translation to rotation for a helical type joint.

7.3.3 3D Mechanical Joint Model

Following on from the *a priori* model-based reasoning for rigid-body tracking, it is assumed here that the CAD model also contains known information about mechanical joints which connect different components of an object. Therefore it is necessary to define a general joint. As will be shown, any type of joint or link can be fully defined by a velocity constraint matrix and a pose of the joint with respect to a fixed reference frame.

A joint matrix for joint j is denoted by:

$$\mathbf{J}_j(\mathbf{S}_j^{\perp}, \mathbf{r}_j), \tag{7.14}$$

where:

$\mathbf{S}^{\perp}{}_j$ is a velocity constraint matrix as was defined in Section 7.3.2 used to define the mechanical constraint of the joint j. This information is considered to be defined *a priori* and it remains constant over time. Some examples are given further in this section.

$\mathbf{r}_j(t)$ is a 6-parameter pose vector, defined *a priori*, representing the location and orientation of the joint in 3D which varies over time. This is used to further derive the different parts of the articulation matrix.

In Section 7.4 various different joint matrices will be given for the different motion models proposed.

Joint Type

In this section a velocity constraint matrix is used to define two subspaces of a single rigid mechanical link. One belonging to the rigid part of the link and the other to the non-rigid part.

In terms of the physical joint, the set of velocities that a first component can undertake which leaves a second component invariant is defined by $S^{\perp} \subset se(3)$. This is the orthogonal compliment of the subspace $S \subset se(3)$ which constitutes the velocities which are in common between two components. Since a component, that is linked to another, is composed of these two subspaces, it is possible to project component twists onto these subspaces with these standard bases for the kernel and the image.

It should be noted that due to the assignment of sets to matrices, the column vectors of the matrices should be considered as unordered and dependent upon numerical calculation methods. This unknown order is avoided by defining projection operators. In the following, the matrix $\mathbf{S}_j$ and its orthogonal compliment $\mathbf{S}_j^{\perp}$, defined in equations (7.12) and (7.13), are used to project the kinematic twist (velocities) onto two orthogonal subspaces. In Section 7.4.3, the more complicated case of more than one link is considered.

A subspace projection matrix is given as:

$$\begin{aligned} \mathbf{P}_j &= \mathbf{S}_j\mathbf{S}_j^{+}, \\ \mathbf{P}_j^{\perp} &= \mathbf{S}_j^{\perp}\mathbf{S}_j^{\perp+} = \boldsymbol{I}_6 - \mathbf{P}_j. \end{aligned} \tag{7.15}$$

This ensures that the resulting projected velocities are defined according to a common basis defined by the parameters of the velocity vector in equation (2.27). Most importantly, this allows the twist transformations, given in the following section, to be applied to these quantities.

Examples

In this example, the constraint matrices for two different types of class 1 links are given. These two joints are shown in Figure 7.3.

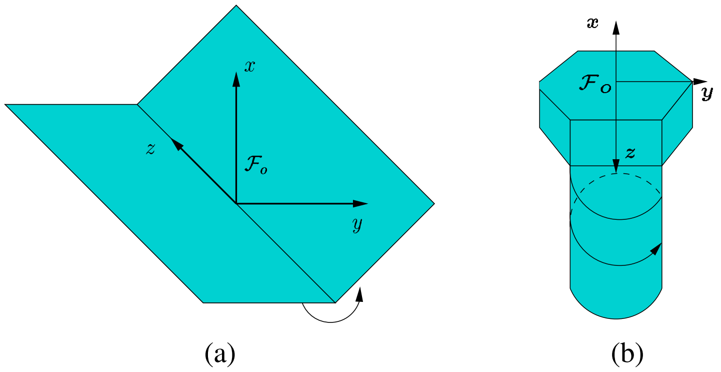


Figure 7.3: (a) A class one rotational hinge link, where component 1 and component 2 are both rectangular components , (b) A class one helical link where component 1 is a screw and component 2 is a square plate that is not shown in the diagram.

A single rotational link is defined around the x axis by a constraint vector:

$$\mathbf{S}_j^{\perp} = \begin{bmatrix} 0 \\ 0 \\ 0 \\ 1 \\ 0 \\ 0 \end{bmatrix}, \quad \mathbf{S}_j = \begin{bmatrix} 1 & 0 & 0 & 0 & 0 \\ 0 & 1 & 0 & 0 & 0 \\ 0 & 0 & 1 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 1 & 0 \\ 0 & 0 & 0 & 0 & 1 \end{bmatrix}. \tag{7.16}$$

where the corresponding constraint matrix is found by solving for the Kernel of $\mathbf{S}_j^{\perp}$ and where $\mathbf{S}_j^{\perp}$ belongs to a set of 6 constraint vectors with the remaining vectors being NULL.

It is possible to verify that the constraint in (7.16) effectively constrains the velocities of a twist as in equation (7.11) as:

$$\begin{aligned} (\mathbf{S}_j^{\perp})^{+}\mathbf{v}_j &= \omega_x, \\ \mathbf{S}_j^{+}\mathbf{v}_j &= (v_x,\ \ v_y,\ \ v_z,\ \ \omega_y,\ \ \omega_z). \end{aligned} \tag{7.17}$$

where it can be seen that the first subspace is composed of a pure rotational velocity around the x axis and the compliment subspace contains the remaining degrees of freedom.

A helical link around and along the z axis has a slightly different constraint vector:

$$\mathbf{S}_j^{\perp} = \begin{bmatrix} 0 \\ 0 \\ a \\ 0 \\ 0 \\ 1 \end{bmatrix}, \quad \mathbf{S}_j = \begin{bmatrix} 1 & 0 & 0 & 0 & 0 \\ 0 & 1 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & -\frac{1}{a} \\ 0 & 0 & 1 & 0 & 0 \\ 0 & 0 & 0 & 1 & 0 \\ 0 & 0 & 0 & 0 & 1 \end{bmatrix}, \tag{7.18}$$

where the value of 'a' relates the translation along the z axis corresponding to a single rotation around the z axis. Note again that $\mathbf{S}_j$ is determined automatically by determining the Kernel of $\mathbf{S}_j^{\perp}$.

Once again it is possible to verify that the constraint effectively constrains the velocities of the system as in equation (7.11):

$$\begin{aligned} (\mathbf{S}_j^{\perp})^{+}\mathbf{v}_j &= av_z + \omega_z, \\ \mathbf{S}_j^{+}\mathbf{v}_j &= (v_x,\ \ v_y,\ \ \omega_x,\ \ \omega_y,\ \ -\tfrac{1}{a}v_z + \omega_z). \end{aligned} \tag{7.19}$$

where it can be seen that the determined velocities no longer correspond to the common basis velocities defined by the twist vector $\mathbf{v}$. This potentially poses a problem if one wants to transform this vector using the adjoint map. This problem is overcome by considering a projection matrix.

For the example of a helical link, defined in equation (7.18), the mechanical configuration projector becomes:

$$\mathbf{P}_j = \begin{bmatrix} 1 & 0 & 0 & 0 & 0 & 0 \\ 0 & 1 & 0 & 0 & 0 & 0 \\ 0 & 0 & \frac{1}{a^2+1} & 0 & 0 & -\frac{a}{a^2+1} \\ 0 & 0 & 0 & 1 & 0 & 0 \\ 0 & 0 & 0 & 0 & 1 & 0 \\ 0 & 0 & -\frac{a}{a^2+1} & 0 & 0 & \frac{a^2}{a^2+1} \end{bmatrix}, \quad \mathbf{P}_j^{\perp} = \begin{bmatrix} 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & \frac{a^2}{a^2+1} & 0 & 0 & \frac{a}{a^2+1} \\ 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & \frac{a}{a^2+1} & 0 & 0 & \frac{1}{a^2+1} \end{bmatrix}, \tag{7.20}$$

The columns are now orthogonal with respect to the chosen twist basis as desired. It can also be noted here that the columns are not linearly independent and they must be reduced to their linearly independent form before being used in the following developments. In their reduced form the dimension of the column space for $\mathbf{P}_j$ and $\mathbf{P}_j^{\perp}$ is still 5 and 1 respectively.

Joint Pose and the Adjoint Map

In addition to the type of a mechanical joint defined in the previous section, it is also necessary to define the pose of a joint with respect to a 3D reference frame. The joint location and orientation is fully defined by the well known pose vector. The joint pose may be defined relative to any arbitrary reference frame, however, the most natural choice is with respect to one of the components. Other choices include the camera reference frame or the reference frame associated with object center of gravity. It is assumed here that the joint frame coincides with a component frame. Several cases simplify the definition of joint positions including the case where all components have their reference frames aligned with the joint degree of freedom, however, this is not always possible. The full joint pose vector is used here so as to provide a general description of the position of a joint for any type of joint configuration including the case where the joint and component frames do not coincide. This situation is necessary when a single component has multiple joints all at different locations. In the simple case where both component and joint reference frames coincide, the pose ${}^{\mathcal{F}_1}\mathbf{r}_{\mathcal{F}_2}$ between the two components corresponds to the unknown joint parameter which is to be estimated.

The general joint pose vector for joint j is therefore defined with respect to a connected component's reference frame as:

$$\mathbf{r}_j = {}^{\mathcal{F}_i}\mathbf{r}_{\mathcal{F}_j} = (t_x, t_y, t_z, \theta_x, \theta_y, \theta_z), \tag{7.21}$$

where $\mathcal{F}_i$ indicates an adjoining component's reference frame and $\mathcal{F}_j$ represents joint j's reference frame.

Since the previously defined mechanical constraint projector $\mathbf{P}_j$ is only valid in the link frame of reference, it is necessary to determine the kinematic twist velocities as seen at the joint reference frame. This is achieved by a generic kinematic twist transformation matrix which is used to obtain the velocity of the link frame with respect to the camera. In representation theory, the Lie algebra provides the adjoint matrix $\mathbf{V}(\mathbf{r})$ for transformation of vector quantities as given in equation (2.40). This is a kinematic twist transformation from frame a to frame b. As will be shown in section 7.4, it is possible to choose different methods of applying the constraint matrix within the joint reference frame.

7.4 Articulated Motion Models

Within the context of modeling articulated motion, there are several methods which have been proposed for estimating the pose of an articulated object. In this section, different articulated motion models are given and compared. In Section 7.4.1, kinematic chains are considered and in Section 7.4.2, Lagrange multipliers are considered. Following this a new kinematic set method is proposed which combines the advantages of the previous two methods.

7.4.1 Kinematic Chains

A well known method for modeling 3D articulated motion based on a kinematic chain representation is presented here. A standard method for implementing a kinematic chain is the Denavit-Hartenberg formulation [Denavit55] which lays out a complete symbolic notation for defining the kinematics of a multi-body system. In particular, a methodology is given for selecting successive link coordinates such that a transformation across a joint is only the function of four parameters. This is directly related to the screw relationship between any two lines in space (a screw is given by translation and rotation about the same line). Another method for deriving a kinematic chain is known as the product of exponentials formula [Brockett84] which is usually based on a world reference frame rather than multiple reference frames as in the previous case. In terms of visual tracking, a kinematic chain is used to describe the velocities of a sequence of components connected by joints making up a mechanism. The velocity of the different components and joints is determined by taking the partial derivative of their position along the chain back to the root. See Figure 7.4 for the visual representation of a kinematic chain in terms of visual tracking.

In the literature, a good example of articulated object tracking based on kinematic chains appears in work by Lowe [Lowe91]. In this work the partial derivatives of image features are taken with respect to object pose and articulation joint parameters which vary with time. In this paper, the kinematic chain of articulations is represented as a tree structure of internal rotation and translation parameters and the model points are stored in the leaves of this tree. The position and partial derivatives of each point in camera-centered coordinates is determined by the transformations along the path back to the root.

Recently, more complex features have been used for non-rigid object tracking in [Nunomaki00]. They make use of deformable super-quadric models combined with a kinematic chain approach. However, real-time performance is traded-off for more complex models. Furthermore, this method requires multiple view-points in order to minimize the system of equations. As Lowe points out, the tendencies in computer graphics have been toward local approximations via polyhedral models. Ruff and Horaud [Ruf99a] give another kinematic-chain style method for the estimation of articulated motion with an un-calibrated stereo rig. They introduce the notion of projective kinematics which allows rigid and articulated motions to be represented within the transformation group of projective space. The authors link the inherent projective motions to the Lie-group structure of the displacement group. The minimization is determined in projective space and is therefore invariant to camera calibration parameters.

In the following the kinematic chain approach will be derived using the Virtual Visual Servoing approach. An articulated object is described as a tree of components whereby there is a root component which is rigidly connected to the camera and the subsidiary components extending down to the leaves of the tree are dependent on all parents along a path back to the root. Since all subsidiary components depend on their parent component and subsequently all components depend on the root component, the order of selection of the components is significant. As will be shown, with this approach is possible to consider the calculation of the velocities and the location parameters for all degrees of freedom simultaneously. This has the effect of canceling the errors across all components of the object and is more accurate than calculating the poses separately.

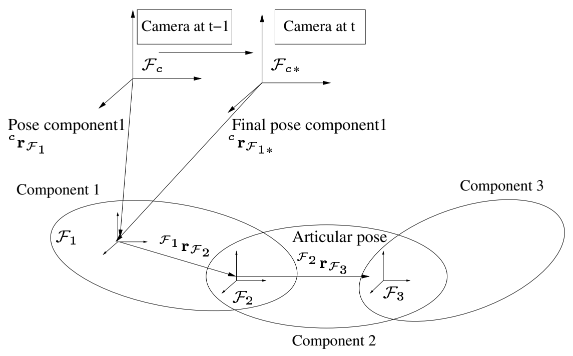


Figure 7.4: Kinematic chain method: The pose of an articulated object is determined via a kinematic chain of rigid bodies extending to sub components. Each circle represents a rigid component. The root component is rigidly linked to the camera via the pose ${}^c\mathbf{r}_{\mathcal{F}_1}$ and each subsequent component is linked by the parameters contained within a pose such as ${}^{\mathcal{F}_1}\mathbf{r}_{\mathcal{F}_2}$ from component 1 to 2.

To achieve this goal, one of the two components is related, as in subsection 7.3.1, to the camera frame of reference by its pose. The error Δ is defined as in equation (7.5). By doing this the pose of the first component is calculated by using its error in the image with:

$$\dot{\Delta}_1 = \mathbf{L_s}_1 {}^c\mathbf{v}_{c^*}. \tag{7.22}$$

The pose for the child components can be determined by the simple observation that the pose between the root component and a child component can be related by using the poses between all the components along the chain back to the root as (see Figure (7.4)):

$$\begin{aligned} {}^c\mathbf{M}_{\mathcal{F}_i} &= {}^c\mathbf{M}_{\mathcal{F}1} {}^{\mathcal{F}1}\mathbf{M}_{\mathcal{F}2} \ldots {}^{\mathcal{F}_{i-1}}\mathbf{M}_{\mathcal{F}_i}, \\ &= e^{\left[{}^c\mathbf{v}_{\mathcal{F}1}\right]} e^{\left[{}^{\mathcal{F}1}\mathbf{v}_{\mathcal{F}2}\right]} \ldots e^{\left[{}^{\mathcal{F}_{i-1}}\mathbf{v}_{\mathcal{F}_i}\right]}, \end{aligned} \tag{7.23}$$

where i is any arbitrary component in the chain such that the components are numbered in increasing order along the chain with 1 being the root component. The homogeneous transformation matrix $\mathbf{M}$ as defined in equation (2.13) corresponds to the poses $\mathbf{r}$. The second line corresponds to the product of exponentials formula which defines this relation in terms of twists.

By considering equation (7.23) it can be seen that the visual error (7.5) depends on the poses along the kinematic chain back to the root as:

$$\Delta_i = \mathbf{s}_i({}^c\mathbf{r}_{\mathcal{F}_i}) - \mathbf{s}^*{}_i = \mathbf{s}_i({}^c\mathbf{r}_{\mathcal{F}_1}, {}^{\mathcal{F}_1}\mathbf{r}_{\mathcal{F}_2}, \ldots, {}^{\mathcal{F}_{i-1}}\mathbf{r}_{\mathcal{F}_i}) - \mathbf{s}^*{}_i. \tag{7.24}$$

where it can be noted that the inter-component poses ${}^a\mathbf{r}_b$ can be reduced to their minimal form ${}^a\mathbf{q}_b$ which have a dimension corresponding to the dimension of the corresponding joint.

Now that the visual features have been defined for a general component i it is necessary relate this information to the movement of the camera and the object (including all joint movement). In order to achieve this it is necessary to define common reference frames amongst the different components so that the resulting simultaneous set of equation are defined within a common basis for minimization. It is assumed here that all joint frames coincide with all *parent* component frames. Thus the first joint is defined in the first components reference frame, the second joint is defined in the second components reference frame, etc... The basis reference frames will therefore be defined accordingly with the exception of the root component which is also linked to the camera reference frame. In this case the basis frames is chosen to be the camera reference frame.

It is now interesting to reconsider the object-based motion model as was considered in Chapter 3. This model makes it possible to determine the relationship between the movement of each component's visual features and the movement of each individual joint. Since each component's twist is initially defined within the camera reference frame it is necessary to transform each components twist from the camera reference frame to each joint reference frame. Using the poses between each joint and the camera (computed using the components poses), the adjoint transformation matrix $\mathbf{V}$ given in (2.40) can be used to perform this transformation according to the following relation:

$$ {}^{l}\mathbf{v}^{i}_{l*} = {}^{c}\mathbf{V}_{l}{}^{c}\mathbf{v}^{i}_{c*}, \tag{7.25} $$

where ${}^{c}\mathbf{V}_{l}$ is the adjoint map which depends on the object configuration from the previous estimation step. ${}^{l}\mathbf{v}^{i}_{l*}$ is the twist related to component i's features, expressed in the joint reference frame l and ${}^{c}\mathbf{v}^{i}_{c*}$ is the twist of component i in the camera reference frame. This can be used to relate the error in the image of each component to the movement of each unknown parameter defined in a particular reference frame.

Using a Taylor series expansion of (7.24) the image feature velocities become:

$$ \begin{aligned} \dot{\mathbf{s}}_i &= \frac{\partial \mathbf{s}_i}{\partial^{c}\mathbf{r}_{c*}}\frac{d^{c}\mathbf{r}_{c*}}{dt} + \frac{\partial \mathbf{s}_i}{\partial^{c}\mathbf{r}_{c*}}\frac{\partial^{c}\mathbf{r}_{c*}}{\partial^{\mathcal{F}_1}\mathbf{r}_{\mathcal{F}_1^*}}\frac{\partial^{\mathcal{F}_1}\mathbf{r}_{\mathcal{F}_1^*}}{\partial^{\mathcal{F}_1}\dot{\mathbf{q}}_{\mathcal{F}_1^*}}\frac{d^{\mathcal{F}_1}\dot{\mathbf{q}}_{\mathcal{F}_1^*}}{dt} + \cdots + \frac{\partial \mathbf{s}_i}{\partial^{c}\mathbf{r}_{c*}}\frac{\partial^{c}\mathbf{r}_{c*}}{\partial^{\mathcal{F}_{i-1}}\mathbf{r}_{\mathcal{F}_{i-1}^*}}\frac{\partial^{\mathcal{F}_{i-1}}\mathbf{r}_{\mathcal{F}_{i-1}^*}}{\partial^{\mathcal{F}_{i-1}}\dot{\mathbf{q}}_{\mathcal{F}_{i-1}^*}}\frac{d^{\mathcal{F}_{i-1}}\dot{\mathbf{q}}_{\mathcal{F}_{i-1}^*}}{dt} \\ &= \mathbf{L}_{\mathbf{s}_i}{}^{c}\mathbf{v}_{c*} + \mathbf{L}_{\mathbf{s}_i}{}^{c}\mathbf{V}_{\mathcal{F}1}{}^{\mathcal{F}_1}\mathbf{P}_{\mathcal{F}_2}{}^{\mathcal{F}_1}\dot{\mathbf{q}}_{\mathcal{F}_1^*} + \cdots + \mathbf{L}_{\mathbf{s}_i}{}^{c}\mathbf{V}_{\mathcal{F}_{i-1}}{}^{\mathcal{F}_{i-1}}\mathbf{P}_{\mathcal{F}_i}{}^{\mathcal{F}_{i-1}}\dot{\mathbf{q}}_{\mathcal{F}_{i-1}^*} \end{aligned} \tag{7.26} $$

where:

- $\mathbf{L}_{\mathbf{s}_i}({}^{c}\mathbf{r}_{\mathcal{F}_j}, \mathbf{s}_i)$ is the interaction matrix derived in Chapter 5. It defines the interaction between the visual features from component i and the velocity of the camera (or the object).
- ${}^{a}\mathbf{V}_{b}({}^{a}\mathbf{r}_{b})$ is the 6×6 twist transformation matrix or adjoint matrix defined in equation (2.40). It is used here to transform a twist defined in the camera reference frame to a corresponding joint reference frame.
- $\mathbf{P} = \mathbf{S}\mathbf{S}^{+}$ is a $6 \times d_i$ joint subspace projector that is based on the velocity constraint matrix $\mathbf{S}$ defined in equation (7.15), where d_i is the dimension or class of the joint for component i. It is used here to constrain the velocity twist to a subset corresponding to a particular joint type.
- ${}^{c}\mathbf{v}_{c*}$ is the camera twist and ${}^{a}\dot{\mathbf{q}}_{b}$ are the joint parameter velocities.

Therefore the full unknown parameter vector is:

$$ \dot{\mathbf{q}} = (\mathbf{v}, {}^{\mathcal{F}_1}\dot{\mathbf{q}}_{\mathcal{F}_1^*}, \cdots, {}^{\mathcal{F}_{i-1}}\dot{\mathbf{q}}_{\mathcal{F}_{i-1}^*}). \tag{7.27} $$

where the dimension of this vector is $(\dim(\dot{\mathbf{q}}) = 6 + \sum_{i-1}^{l} d_i)$ where d_i is the dimension of the joint i and where $l = \kappa - 2$ is the number of joints (i.e. there is no joint between the camera and the root component).

It can be seen that the joint constraint projector, as given in equation (7.15), is applied in the joint reference frame by first transforming a twist to the joint reference frame using the adjoint map $\mathbf{V}$ and then applying the joint projector $\mathbf{P}$. Furthermore, it can be seen that the first term relates the visual information of a component to the pose between the camera and the root component. Each subsequent term relates the visual information of a component to the joint parameters along the chain. Therefore the dimension of the resulting Jacobian matrix is minimal.

The set of simultaneous equations can now be defined for the articulated object as in the case of multiple components (7.6). The velocity model for an articulated object is therefore derived as:

$$\boxed{\begin{aligned}\begin{bmatrix}\dot{\mathbf{s}}_1\\ \dot{\mathbf{s}}_2\\ \vdots\\ \dot{\mathbf{s}}_\kappa\end{bmatrix} &= \begin{bmatrix}\mathbf{L}_{\mathbf{s}1} & & \mathbf{0}_{n_1\times 6}\\ & \mathbf{L}_{\mathbf{s}2} & \\ \vdots & \ddots & \\ \mathbf{0}_{n_i\times 6} & & \mathbf{L}_{\mathbf{s}_i}\end{bmatrix}\begin{bmatrix}\mathbf{I}_{6\times 6} & \mathbf{0}_{6\times d_2} & \cdots & \mathbf{0}_{6\times d_\kappa}\\ \mathbf{I}_{6\times 6} & {}^{c}\mathbf{V}_{\mathcal{F}1}{}^{\mathcal{F}_1}\mathbf{P}_{\mathcal{F}_2} & \cdots & \mathbf{0}_{6\times d_\kappa}\\ \vdots & \vdots & \ddots & \vdots\\ \mathbf{I}_{6\times 6} & {}^{c}\mathbf{V}_{\mathcal{F}1}{}^{\mathcal{F}_1}\mathbf{P}_{\mathcal{F}_2} & \cdots & {}^{c}\mathbf{V}_{\mathcal{F}_{\kappa-1}}{}^{\mathcal{F}_{\kappa-1}}\mathbf{P}_{\mathcal{F}_\kappa}\end{bmatrix}\begin{bmatrix}{}^{c}\mathbf{v}_{c^*}\\ {}^{\mathcal{F}_1}\dot{\mathbf{q}}_{\mathcal{F}_1^*}\\ \vdots\\ {}^{\mathcal{F}_{\kappa-1}}\dot{\mathbf{q}}_{\mathcal{F}_{\kappa-1}^*}\end{bmatrix}\\ \dot{\mathbf{s}} &= \mathbf{L}_\mathbf{s}\mathbf{H}\dot{\mathbf{q}},\end{aligned}} \quad (7.28)$$

where

- $\dot{\mathbf{s}}$ is the combined feature velocity vector in the image and κ is the number of components,
- $\mathbf{L}_\mathbf{s}$ is a block diagonal interaction matrix with rigid component interaction matrices on the diagonal. They link the image feature velocities and to each components individual twist. $\mathbf{L}_\mathbf{s}$ is of dimension $(\sum_{i=1}^{\kappa} n_i \times (6\kappa))$ where $(\sum_{i=1}^{\kappa} n_i)$ is the dimension of the combined error vector.
- $\mathbf{H}$ is the full Jacobian between the rigid velocity twists of each component and the minimal space of velocities associated with the camera velocity plus all the joint velocities. The dimension of $\mathbf{H}$ is $(6\kappa \times (6 + \sum_{i=1}^{l} \dim(d_i))$ where d_i is the dimension of each joint i which are summed across all l joints.
- $\dot{\mathbf{q}}$ is the combined twist vector of minimal dimension.

It is important to point out that in this formulation the combined interaction matrix $\mathbf{H}$ is block triangular of size $n \times m$ where n is the total number of visual features and m is the dimension of the minimal parameter vector. Once this relation has been defined it is easy to obtain the velocity screw of the root component with respect to the camera frame of reference, along with the joint velocities with respect to the joint reference frames.

Firstly, by solving (7.28) for $\dot{\mathbf{q}}$ the following relation is obtained:

$$\dot{\mathbf{q}} = -\lambda \mathbf{L}_\mathbf{s}\mathbf{H}^+\dot{\mathbf{s}}, \quad (7.29)$$

which forms the basis of a visual control law. The articulated motion control law will be developed further in Section 7.5.

Once all the parameters have been estimated, the individual pose of each component is determined by first applying the constraints and transformations of equation (7.28) in the reverse order as:

$$\mathbf{v} = \mathbf{H}\dot{\mathbf{q}}, \quad (7.30)$$

where $\mathbf{v}$ is the stacked vector of full poses between the camera and each component.

The the exponential map can then be used to obtain the poses associated with each component. The updated pose of the camera is simply obtained from the first twist in the stack $\mathbf{v}$. This pose is ${}^{c}\mathbf{M}cc^{*} = e([{}^{c}\mathbf{v}_{c^*}])$. The pose of the root component is then obtained by updating the original pose of the root component with this transformation as ${}^{c^*}\mathbf{M}_o = {}^{c}\mathbf{M}_{c^*}^{-1}{}^{c}\mathbf{M}_o$. In order to obtain the poses of all sub-components, the adjoint map is used to transform this initial pose down the chain as:

$$ {}^{c^*}\mathbf{v}_{\mathcal{F}_i} = {}^{c}\mathbf{V}_{\mathcal{F}1}{}^{\mathcal{F}1}\mathbf{V}_{\mathcal{F}2} \dots {}^{\mathcal{F}_{i-1}}\mathbf{V}_{\mathcal{F}_i}{}^{\mathcal{F}_{i-1}}\mathbf{v}_{\mathcal{F}_i}, \tag{7.31}$$

where the adjoint maps depend on the previously calculated poses: ${}^{c}\mathbf{V}_{\mathcal{F}1}({}^{c}\mathbf{r}_{\mathcal{F}_1})$, ${}^{\mathcal{F}1}\mathbf{V}_{\mathcal{F}2}({}^{\mathcal{F}_1}\mathbf{r}_{\mathcal{F}_2})$, etc... so that the calculation of the poses depends on the calculation of the parent poses along the chain and where the pose between two components is easily computed as:

$$ {}^{a}\mathbf{M}_b = {}^{c}\mathbf{M}_a^{-1}{}^{c}\mathbf{M}_b. \tag{7.32}$$

To verify if this derivation is in agreement with the multiple free object case, consider that there is a free link between the two components and the two components are defined in the same coordinate frame then:

$$ \mathbf{P}^{\perp} = \mathbb{I}_{6\times 6}, \quad \mathbf{P} = \emptyset, \quad \text{and} \quad {}^{\mathcal{F}1}\mathbf{V}_{\mathcal{F}2} = \mathbb{I}_{6\times 6}, \tag{7.33}$$

and when this is inserted into equation (7.28) it can be easily shown to be equivalent to equation (7.22).

The drawback of this formulation is that it requires selection of a 'master' component and any errors retained in determining its pose are propagated along the chain and into subcomponents. This renders the estimation imprecise and iterative convergence may take longer. Consequently, it would be preferable to have the most robust and accurate pose as the primary component. As will be shown in Section 7.4.3, this can be overcome by considering a symmetric kinematic set approach.

7.4.2 Lagrange Multipliers

Another approach has been proposed in [Delamarre99] for modeling articulated objects and using multiple views for tracking people using silhouettes. Poses are stored independently for individual components and constraints are then imposed using Lagrange multipliers. More recently, Drummond and Cipolla [Drummond00a, Drummond02] also considered articulated objects as groups of rigid components with constraints between them, expressed directly in camera coordinates (see Figure 7.5). In this formulation, the full pose of each rigid component is initially computed independently requiring the estimation of a non-minimal number of parameters. In a second step, Lagrange multipliers are then used to constrain these parameters according to simple link definitions. In this approach, Lie Algebras are used to project the measurement vector (distances) onto the subspace defined by the Euclidean transformation group (kinematic screw). In previous works, only linear links have been considered, however, it will be shown that the constraint matrices defined in Section 7.3.2 easily allow extension of this method to non-linear joints.

In this formulation, estimation of unknown parameters of a multi-body articulated object is based on a multi-step estimation process. In a fist step the individual poses are estimated for each component separately as in Section 7.3.1. The poses are defined directly in the camera reference frame by c twists $\hat{\mathbf{v}}_i$, where c is the number of components. The hat is used to signify an approximation due to the fact that the visual information of the other components has not been taken into account in the estimation process.

The subsequent estimation steps involve applying constraints to all components of an articulated object one-by-one while taking into account all previous adjustments of the estimates. The number of secondary

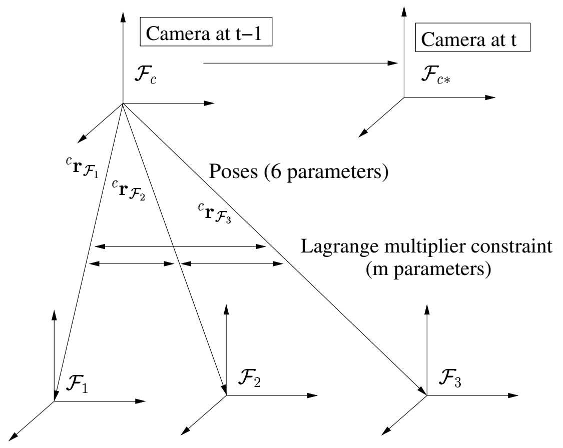


Figure 7.5: Lagrange Multiplier method: The poses between the camera and each part of the object ${}^c\mathbf{r}_{\mathcal{F}_1}$, ${}^c\mathbf{r}_{\mathcal{F}_2}$ and ${}^c\mathbf{r}_{\mathcal{F}_c}$ are calculated directly in a first step. Constraints between the components are then enforced in a second minimization step via Lagrange multipliers. In this case, a kinematic chain or tree is also used to represent the articulated structure.

estimation steps correspond to the number of joints. For each joint, the velocity constraint matrix $\mathbf{S}$, as defined in equation (7.10), is used to create another constraint equation between two components:

$$\mathbf{S}\mathbf{v}_1 - \mathbf{S}\mathbf{v}'_2 = 0, \tag{7.34}$$

where the twists $\mathbf{v}$ are the true velocities of the fully constrained system which are to be determined here. $\mathbf{v}'_2$ corresponds to the transformation of component two's twist into component one's reference frame as:

$$\mathbf{v}'_2 = ({}^c\mathbf{V}_{\mathcal{F}_2}{}^{-1c}\mathbf{V}_{\mathcal{F}_1})\mathbf{v}_2 = {}^{\mathcal{F}_2}\mathbf{V}_{\mathcal{F}_1}\mathbf{v}_2, \tag{7.35}$$

where $\mathbf{V}$ is the adjoint matrix (2.38). It should be noted, however, that this basis reference frame is arbitrary. This equation effectively defines the subset of velocities in common between the two components (corresponding to a joint) within the same reference frame.

A secondary minimization equation can then be written by considering the difference between the ideal velocities $\mathbf{v}$ and the previously estimated approximate velocities $\hat{\mathbf{v}}$ such that:

$$\delta = (\mathbf{v}_1 - \hat{\mathbf{v}}_1)^\top \mathbf{L}_{\mathbf{s}_1}^\top \mathbf{L}_{\mathbf{s}_1}(\mathbf{v}_1 - \hat{\mathbf{v}}_1) + (\mathbf{v}'_2 - \hat{\mathbf{v}}'_2)^\top \mathbf{L}'^\top_{\mathbf{s}_2}\mathbf{L}'_{\mathbf{s}_2}(\mathbf{v}'_2 - \hat{\mathbf{v}}'_2) \tag{7.36}$$

is minimized subject to equation (7.34), where $\mathbf{L}'_{s2}$ is the interaction matrix corresponding to $\mathbf{v}'_2$.

This is then easily minimized by using Lagrange multipliers as:

$$\nabla\Big((\mathbf{v}_1 - \hat{\mathbf{v}}_1)^\top \mathbf{L}_{\mathbf{s}_1}^\top \mathbf{L}_{\mathbf{s}_1}(\mathbf{v}_1 - \hat{\mathbf{v}}_1) + (\mathbf{v}_2 - \hat{\mathbf{v}}'_2)^\top \mathbf{L}'^\top_{\mathbf{s}_2}\mathbf{L}'_{\mathbf{s}_2}(\mathbf{v}'_2 - \hat{\mathbf{v}}'_2)\Big) + \lambda\nabla(\mathbf{S}\mathbf{v}_1 - \mathbf{S}\mathbf{v}'_2) = 0 \tag{7.37}$$

where λ is a vector of scalars each corresponding to a column of the constraint matrix $\mathbf{S}$, which gives:

$$2\begin{pmatrix} \mathbf{L}_{\mathbf{s}_1}^\top \mathbf{L}_{\mathbf{s}_1}(\mathbf{v}_1 - \hat{\mathbf{v}}_1) \\ \mathbf{L}'^\top_{\mathbf{s}_2}\mathbf{L}'_{\mathbf{s}_2}(\mathbf{v}'_2 - \hat{\mathbf{v}}'_2) \end{pmatrix} + \begin{pmatrix} \lambda\mathbf{S} \\ -\lambda\mathbf{S} \end{pmatrix} = 0 \tag{7.38}$$

and solving for the unknowns gives:

$$\boxed{\begin{aligned} \mathbf{v}_1 &= \hat{\mathbf{v}}_1 - \tfrac{1}{2}(\mathbf{L}_{\mathbf{s}_1}^\top \mathbf{L}_{\mathbf{s}_1})^{-1}\lambda\mathbf{S}, \\ \mathbf{v}'_2 &= \hat{\mathbf{v}}'_2 + \tfrac{1}{2}(\mathbf{L}'^\top_{\mathbf{s}_2}\mathbf{L}'_{\mathbf{s}_2})^{-1}\lambda\mathbf{S} \end{aligned} \qquad (7.39)}$$

The Lagrange multiplier constraint ensures that the twist of the first component is dependent on the second and vice-versa. In this way it is possible to describe the error of the entire object in the image as a function of the pose of the different components.

It now remains to calculate λ. This is achieved by substituting (7.34) into (7.39) to give:

$$\mathbf{S}\hat{\mathbf{v}}_1 + \mathbf{S}\hat{\mathbf{v}}'_2 - \frac{1}{2}\lambda\left(\mathbf{S}(\mathbf{L}_{\mathbf{s}_1}^\top \mathbf{L}_{\mathbf{s}_1})^{-1}\mathbf{S}^\top - \mathbf{S}(\mathbf{L}_{\mathbf{s}_2}'^\top \mathbf{L}'_{\mathbf{s}_2})^{-1}\mathbf{S}^\top\right) = 0. \tag{7.40}$$

and the unknown is then solved as:

$$\lambda = \left(\mathbf{S}(\mathbf{L}_{\mathbf{s}_1}^\top \mathbf{L}_{\mathbf{s}_1})^{-1}\mathbf{S}^\top - \mathbf{S}(\mathbf{L}_{\mathbf{s}_2}'^\top \mathbf{L}'_{\mathbf{s}_2})^{-1}\mathbf{S}^\top\right)^{-1} 2(\mathbf{S}\hat{\mathbf{v}}_1 + \mathbf{S}\hat{\mathbf{v}}'_2) \tag{7.41}$$

The second component's twist can then be transferred back to its own reference frame and the exponential map can be used again to obtain the poses corresponding to the twists.

The major disadvantage of this approach is that the estimation of unknown parameters is not simultaneous and is carried out in a multi-step procedure. This has repercussions such as the propagation of error to subcomponents or the necessity to further constrain the components if their visual features are not of full rank 6. This method allows unwanted noise and error to enter into the estimation process at the initial rigid pose calculation step. Furthermore, as in the kinematic chain case, noise is propagated to its subparts rendering estimation in-efficient and potentially in-accurate.

In previous work on this method [Drummond02], each estimation step was considered in a chain order and this method was thus likened to a kinematic chain. However, it is noted here that multiple paths may be chosen simultaneously through a graph of rigid-kinematic velocities until all unknowns have been estimated. In this way this approach may be considered as being more symmetric than the kinematic chain approach. This idea of a more symmetric estimation process is now developed into a new theory which combines the advantages of both kinematic chains and Lagrange multipliers approaches.

7.4.3 Kinematic Sets

In this section, a new method is proposed for modeling 3D *articulated* motion. This method is called a *kinematic set* approach as opposed to a chain or tree approach. The formalism aims at modeling the interaction between visual features in the image and the movement of the articulated object simultaneously as in the case of a kinematic chain, however, another aim is to achieve this symmetrically (as is possible using Lagrange multipliers). In particular, this section describes how the minimal parameter, generalized velocity vector is obtained. In this case, a novel subset approach is used whereby the minimization is carried out on decoupled subsets of parameters by defining subspace projectors from the joint definitions. These subsets are considered by using velocity constraints which allows a general treatment of the entire space of configurations. This translates as a definition of a class of mechanical joints which includes both linear joints such as rotational, translational, planar etc.. as well as non-linear joints such as helical joints. This formulation allows the error seen in the image to be partially decoupled from the velocities of the object by determining the independent sets of velocities present in object space.

The kinematic set theory has been traced back to the Lagrange-Alembert theory in classical physics. This theory states that instead of applying constraints which apply forces to a system, the coordinate basis is changed to be orthogonal to the constraint so that the only forces in the system are the real physical forces [Rosenberg77]. However, in terms of modeling and tracking an articulated object, it is only necessary to derive the equations in terms of 3D location and orientation velocities as opposed to the more complicated equations of potential and kinetic energy involved in the Lagrange-Alembert theory.

When compared with existing Kinematic Chain approaches, it will be shown that there is no need to sum partial derivatives along a kinematic chain back to the root in a visual system as the camera has a *direct* visual link with the movement of each component of an articulated object and not only the root component. In this case errors in the pose between the camera and the root component are propagated to subcomponents, potentially leading to tracking failure.

On the other hand, when compared to the Lagrange multiplier approach, there is no need to estimate in multiple steps. The advantage is then that components with under-constrained visual information w.r.t. the pose can now be considered. Furthermore, unwanted noise and error is eliminated from the estimation process due to a simultaneous closed-loop estimation process. These advantages lead to improved computational efficiency and greater accuracy.

In the proposed kinematic set approach, the joint reference frame plays an important role. This reference frame can be thought of as the point of intersection between two rigid components which together make up an articulated object. In the model presented in this chapter, a general mechanical joint is defined leading to simple a generic articulation descriptor.

In motion estimation problems, there exists an inherent duality between the motion of the sensor and motion of the scene. More precisely, the motion between a single camera sensor and a rigid object can be attributed to the movement of either the camera or the object. Similarly, in the case of multiple objects, the movement can equally be considered as movement of multiple camera sensors. Furthermore, with articulated motion, unlike the case of rigid motion, the subsets of movement which may be attributed to either the object or camera are not unique. In this treatment, the focus will be placed on a monocular camera observing a constrained multi-body object, however, these methods may equally be applied to the case of multiple cameras [Delamarre99, Plankers03].

The principal advantages of the kinematic set approach are that it:

- takes into consideration any type of mechanical link (i.e. helical, translational, rotational, ball joint etc...).
- allows tracking of under-constrained or rank deficient components (e.g: can track components with less than 3 points).
- eliminates the propagation of errors between free parameters (Important for noisy sensor measurements and approximations).
- is efficient in terms of computation as well as representation space.
- models closely the real behavior of the system. (i.e. multiple objects or multiple cameras).

In the remainder of this section, an overview of the Jacobian mapping required to pass from the rigid velocity vector of each component to a generalized minimal parameter velocity vector is first considered. Following this, the velocity constraint matrices defined in Section 7.3.2 are used to constrain rigid-body twists at the joints. This provides the basis for creating two subspace projection operators for each joint. Following this, an articulation matrix is provided which combines all joint subspaces in a combinatorial fashion so as to determine all the subspaces of the object. These subspaces are referred to as Kinematic Sets. The full Jacobian mapping is referred to as the Articulation matrix.

The Generalized Twist

The approach laid out here aims at reducing multiple *rigid-body* six dimensional velocity twists $\mathbf{v}_1, ., \mathbf{v}_\kappa$ describing the motion of κ rigid components to a minimal set of parameters $\dot{\mathbf{q}}$ describing the configuration of an articulated object.

First, reconsider the interaction matrix which defines the rigid-body mapping between a vector of visual feature velocities, $\mathbf{s}$, and the velocity twist, $\mathbf{v}$, containing linear and angular velocities of the pose as given in equation (7.1).

$$\begin{bmatrix} \dot{\mathbf{s}}_1 \\ \vdots \\ \dot{\mathbf{s}}_{n_i} \end{bmatrix} = \begin{bmatrix} \mathbf{L}_1 & & \mathbf{0}_{n_1 \times 6} \\ & \ddots & \\ \mathbf{0}_{n_i \times 6} & & \mathbf{L}_{n_i} \end{bmatrix} \mathbf{v}_i \tag{7.42}$$

where $\mathbf{v}$ is the stacked twist vector composed of each individual components 6-dimensional twist vector.

The configuration of a *multi-body* articulated object is completely defined by a differential mapping from $(se(3))^\kappa$ to $\mathbb{R}^m$, where m is the minimum number of parameters and $(m < 6\kappa)$. The central idea is to relate the movement $\dot{\mathbf{s}}$ of the set of sensor features to the movement of the generalized object parameters. This is defined as:

$$\begin{bmatrix} \dot{s}_1 \\ \vdots \\ \dot{s}_{f_1} \\ \vdots \\ \\ \dot{s}_n \end{bmatrix} = \begin{bmatrix} \begin{bmatrix} L^1_{1,1} & \dots & L^1_{1,6} \\ \vdots & \ddots & \\ L^1_{f_1,1} & & L^1_{f_1,6} \end{bmatrix} & & \mathbf{0}_{f_1 \times 6} \\ & \ddots & \\ \mathbf{0}_{f_n \times 6} & & \begin{bmatrix} L^\kappa_{1,1} & \dots & L^\kappa_{1,6} \\ \vdots & \ddots & \\ L^\kappa_{f_\kappa,1} & & L^\kappa_{f_\kappa,6} \end{bmatrix} \end{bmatrix} \begin{bmatrix} A_{1,1} & \dots & A_{1,m} \\ & & \\ \vdots & \ddots & \\ & & \\ A_{6\times\kappa,1} & & A_{6\times\kappa,m} \end{bmatrix} \begin{bmatrix} \dot{q}_1 \\ \vdots \\ \dot{q}_m \end{bmatrix}, \tag{7.43}$$

where n is the number of image features, f_i is the number of features in rigid component i, κ is the number of components, and m is the minimal number of parameters.

This is summarized in matrix form as:

$$\boxed{\dot{\mathbf{s}} = \mathbf{L_D}\mathbf{A}\dot{\mathbf{q}}, \tag{7.44}}$$

where

• $\mathbf{L_D} \in \mathbb{R}^{n \times 6\kappa}$ is composed of multiple interaction matrices, $\mathbf{L}_{\mathbf{s}_i}$, along the diagonal. The different rigid bodies or components have a corresponding interaction matrix (7.42) with $\dot{\mathbf{s}}_i = \mathbf{L}_{\mathbf{s}_i}\mathbf{v}_i$ for all $i = 1...\kappa$. Each block's column width is of dimension 6 corresponding to the six parameters of each $\mathbf{v}_i$. The row vectors each correspond to one of n visual feature parameters.

• $\mathbf{A} \in \mathbb{R}^{6\kappa \times m}$ is an Articulation matrix describing the differential relation between component's velocities $\mathbf{v}_i$ and the minimal parameter subspace $\dot{\mathbf{q}}$. This matrix is composed of different subspaces which will be derived in the remainder of this Section.

• $\mathbf{L_D}\mathbf{A}$ being the Jacobian between the visual features and the configuration of the entire object.

The Articulation matrix is the central issue of this work and it is defined separately as the mapping $\mathbb{R}^{6\kappa} \rightarrow \mathbb{R}^m$:

$$\mathbf{v} = \mathbf{A}\dot{\mathbf{q}} \tag{7.45}$$

where $\mathbf{v} \in (se(3))^\kappa$ is a vector of 'stacked' 6-dimensional twists each corresponding to the full velocity twist in $se(3)$ of each component.

This series of definitions implies that relative environmental modifications of the geometrical kind are the only ones allowed to vary. Each component can be represented individually as a rigid body and there exists a minimal subspace related to the entire object's geometrical variations.

The subsets of parameters which make up the object parameters are illustrated by a Venn diagram in Figure 7.6. In order that the generalized sets can be obtained, it is necessary to find the basis vectors representing the minimal subspace. As will be shown in the following sections, this depends on the configuration of the object.

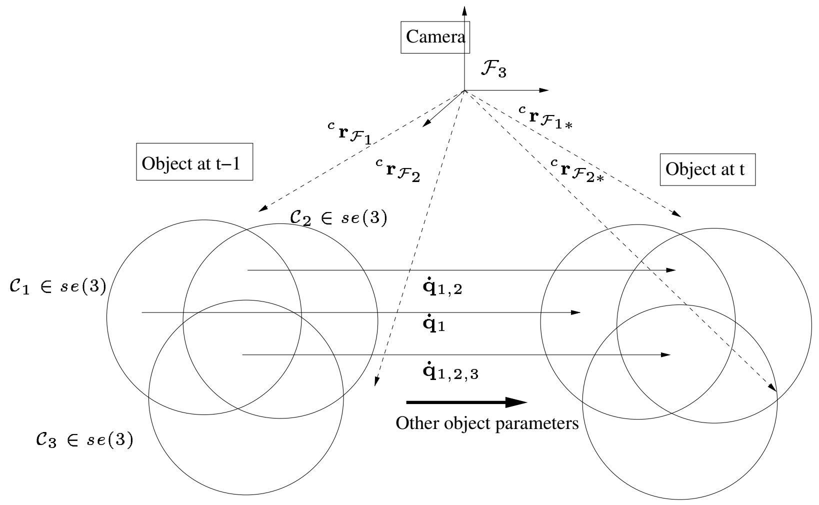

Camera
$\mathcal{F}_3$
${}^c\mathbf{r}_{\mathcal{F}_1}$
${}^c\mathbf{r}_{\mathcal{F}_2}$
${}^c\mathbf{r}_{\mathcal{F}_{1*}}$
${}^c\mathbf{r}_{\mathcal{F}_{2*}}$
Object at t–1
Object at t
$\mathcal{C}_2 \in se(3)$
$\mathcal{C}_1 \in se(3)$
$\mathcal{C}_3 \in se(3)$
$\dot{\mathbf{q}}_{1,2}$
$\dot{\mathbf{q}}_1$
$\dot{\mathbf{q}}_{1,2,3}$
Other object parameters

Figure 7.6: Kinematic set method: Each circle represents a component as set of velocities in $se(3)$ which are rigidly linked together. The intersection of these sets define the minimal subspaces that correspond to the minimal parameters $\dot{\mathbf{q}}$, where $\dot{\mathbf{q}}_{12}$ is a vector of twist parameters resulting from the intersection between $\mathcal{C}_1$ and $\mathcal{C}_2$. Since each subset is orthogonal, there is no interaction between subsets and decoupling occurs within the articulation matrix.

Generalized Joint Projectors

The general form of the articulation matrix, that links the rigid velocity twists of different components, can be derived by first generalizing the application of a joint constraints within the joint frame of reference. The idea is to fully define a general joint projection matrix **J** and its complement so that the velocities of a component can be divided into two orthogonal subsets at a joint.

These subspaces can be determined by re-considering the definition of a joint. The velocity constraint projector as given in (7.15) must be applied in the joint's spatial reference frame. The joint reference frame is given by the joint pose (7.21). The full rigid twist must first be transformed to the joint reference frame using the Adjoint map (2.38) before the constraint may be applied. However, before defining the final joint projector, it is necessary to consider the generality of the resulting velocities. In order to minimize an objective function it is also necessary to have all linear equations defined within the same basis frame. Therefore, a general joint projector must also be defined such that it projects all velocity twists onto the same basis frame. The choice of this reference frame is rather arbitrary and it is possible to choose either the camera or the object reference frames as considered in Chapter 3. In what follows, the camera reference frame is chosen to be the common basis. The interested reader can refer to previous works [Comport04b] where the object reference frame was used.

These choices lead to a generic subspace projection operator which defines the two subspaces of a joint as:

$$\boxed{\begin{aligned} \mathbf{J}_j &= Im({}^{\mathcal{F}_c}\mathbf{V}_{\mathcal{F}_j}\,\mathbf{P}_j\,{}^{\mathcal{F}_j}\mathbf{V}_{\mathcal{F}_c}), \\ \mathbf{J}_j^{\perp} &= Ker(\mathbf{J}_j) = Im({}^{\mathcal{F}_c}\mathbf{V}_{\mathcal{F}_j}\,\mathbf{P}_j^{\perp}\,{}^{\mathcal{F}_j}\mathbf{V}_{\mathcal{F}_c}), \end{aligned} \qquad (7.46)}$$

where Im represents the Image operator which reduces the column space to its mutually independent basis form. The first transformation ${}^{\mathcal{F}_c}\mathbf{V}_{\mathcal{F}_j}$ maps the rigid velocities of a component to the link frame l where the mechanical constraint is applied. The second transformation ${}^{\mathcal{F}_j}\mathbf{V}_{\mathcal{F}_c}$ then re-maps the subspace velocities back to the common camera reference frame.

The projector between features on a rigid object being $\mathbf{J}_j = \mathbf{I}_6$ and for two completely independent objects $\mathbf{J}_j = \mathbf{0}_6$

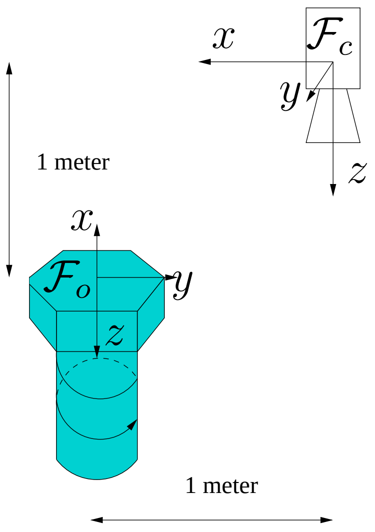


Figure 7.7: Example considering a helical joint 1 meter along the x and z axes of the camera.

Example

For simplicity, let the object be 1 meter along the z-axis and 1 meter along the x-axis of frame $\mathcal{F}_c$ and is in front of the camera (see Figure 7.7). By definition, the object reference frame is chosen to be the same as the joint reference frame for this example. The adjoint matrix ${}^{c}\mathbf{V}_o$ is then given by:

$$ {}^{c}\mathbf{V}_o = \begin{bmatrix} 1 & 0 & 0 & 0 & -1 & 0 \\ 0 & 1 & 0 & 1 & 0 & -1 \\ 0 & 0 & 1 & 0 & 1 & 0 \\ & \mathbf{0}_3 & & & \boldsymbol{I}_3 & \end{bmatrix}. \qquad (7.47)$$

If a helical joint configuration is considered, then using equation (7.20), (7.48) and (7.46), the joint constraint matrix is determined as:

$$\mathbf{J}_j = \begin{bmatrix} 1 & 0 & 0 & 0 & 0 \\ 0 & 1 & 0 & 0 & 0 \\ 0 & 0 & 1 & 0 & -\frac{1}{a} \\ 0 & 0 & 0 & 1 & 0 \\ 0 & 0 & 1 & 0 & 0 \\ 0 & 0 & 0 & 0 & 1 \end{bmatrix} \quad \text{and} \quad \mathbf{J}_j^{\perp} = \begin{bmatrix} 0 \\ -1 \\ a \\ 0 \\ 0 \\ 1 \end{bmatrix}. \tag{7.48}$$

Since the transformation is orthogonal, the rank of the matrix's column space before and after transformation stays the same. Once taking the kernel and the image of the resulting matrices as in equation (7.46), the dimension of the column space is equal to its rank.

Generalized Object Subspaces: Kinematic Sets

The aim of this section is to define the form of the articulation matrix $\mathbf{A}$ which itself will define how all joint subspaces are tied together. In particular, the two previously defined subspaces of each joint ($\mathbf{J}_j$, $\mathbf{J}_j^{\perp}$) are combined with the subspaces of other joints giving a certain number of subspaces for the entire object. This will in-turn define a basis for the minimum number of parameters which define the objects configuration.

An intuitive interpretation of the different subspaces can be obtained when referring to Figure 7.6. With one joint and two components, there are three distinct subspaces, the subspace in common between the components plus the independent subspace of each component. With two joints and subsequently one virtual joint, there are possibly seven distinct subspaces. This is determined by looking at all the intersections and non-intersections between each subspace. It can be deduced that an articulated object that contains κ components has possibly $2^{\kappa}-1$ distinct subspaces. Likewise each individual component has $2^{(\kappa-1)}$ possible subspaces which are in common with the other components.

Virtual Links

To maintain the geometric symmetry of the problem, it is necessary to have a description of the links between all rigid bodies of an object. If no link is defined between any two components, a 'virtual link' exists such that:

$$\boxed{\begin{aligned} \mathbf{J}_{ab} &= \textstyle\bigcap_{\forall l \in \mathcal{P}} \mathbf{J}_j, \\ \mathbf{J}_{ab}^{\perp} &= \textstyle\bigcap_{\forall l \in \mathcal{P}} \mathbf{J}_j^{\perp} = Ker(\mathbf{J}_{ab}), \end{aligned} \qquad (7.49)}$$

where $\mathcal{P}$ is a path of joints connecting component a to b and where the intersection between two subspaces, represented by the basis matrices $\mathbf{U}$ and $\mathbf{V}$, is defined as:

$$\mathbf{U} \cap \mathbf{V} = Ker \begin{pmatrix} \mathbf{U}^{\perp} \\ \mathbf{V}^{\perp} \end{pmatrix}. \tag{7.50}$$

The result of this intersection being an unordered basis for the intersecting subspace.

Kinematic Sets

In order to determine all the subspaces of an object and eventually determine the form of the articulation matrix $\mathbf{A}$, it is necessary to use the intersection operator to create projectors for all combinations of joint Image subspaces. Let the set $\mathcal{O}$ be the set of all combinations of joint subspaces for the entire object, where $\dim(\mathcal{O}) = 2^{\kappa} - 1$. First, consider the basic cases of two and three components.

For two components and subsequently one joint, $\mathcal{O}$, excluding the empty-set $\emptyset$, is given by:

$$\begin{aligned} \mathcal{O} &= \{(\mathbf{v}_1 \cap \mathbf{v}_2), (\mathbf{v}_1 \notin \mathbf{v}_2), (\mathbf{v}_2 \notin \mathbf{v}_1)\} \setminus \emptyset, \\ &\equiv \left\{\mathbf{J}_{12}, \mathbf{J}^{\perp}_{12}, \mathbf{J}^{\perp}_{12}\right\} \setminus \emptyset, \end{aligned} \tag{7.51}$$

where in the first line each $\mathbf{v}_i \in se(3)$ and the intersection between two sets is related to, but not equal to, the intersection operator in equation (7.50). In the second line each element refers to a joint with $l = 12$ being the joint between component 1 and component 2. Note also that the subspace $\mathbf{J}^{\perp}_{12}$ is the same for both $(\mathbf{v}_1 \notin \mathbf{v}_2)$ and $(\mathbf{v}_2 \notin \mathbf{v}_1)$.

For three components, there always exist three joints if virtual joints are included. The reason that virtual joints need to be determined is to provide a set of symmetrical equations with a general form. Furthermore, this allows the formulation to take into account redundant and parallel architectures. In this case, $\mathcal{O}$ is given by:

$$\begin{aligned} \mathcal{O} \;=\;& \{(\mathbf{v}_1 \cap \mathbf{v}_2 \cap \mathbf{v}_3), (\mathbf{v}_1 \cap \mathbf{v}_2 \notin \mathbf{v}_3), (\mathbf{v}_1 \cap \mathbf{v}_3 \notin \mathbf{v}_2), (\mathbf{v}_2 \cap \mathbf{v}_3 \notin \mathbf{v}_1), \\ & (\mathbf{v}_1 \notin \mathbf{v}_2 \notin \mathbf{v}_3), (\mathbf{v}_2 \notin \mathbf{v}_1 \notin \mathbf{v}_3), (\mathbf{v}_3 \notin \mathbf{v}_1 \notin \mathbf{v}_2)\} \setminus \emptyset, \\ \equiv\;& \left\{(\mathbf{J}_{12} \cap \mathbf{J}_{23} \cap \mathbf{J}_{31}), (\mathbf{J}_{12} \cap (\mathbf{J}^{\perp}_{23} \cap \mathbf{J}^{\perp}_{31})), (\mathbf{J}_{23} \cap (\mathbf{J}^{\perp}_{12} \cap \mathbf{J}^{\perp}_{31})), (\mathbf{J}_{31} \cap (\mathbf{J}^{\perp}_{12} \cap \mathbf{J}^{\perp}_{23})), \right. \\ & \left. (\mathbf{J}^{\perp}_{12} \cap \mathbf{J}^{\perp}_{22} \cap \mathbf{J}^{\perp}_{31}), (\mathbf{J}^{\perp}_{12} \cap \mathbf{J}^{\perp}_{22} \cap \mathbf{J}^{\perp}_{31}), (\mathbf{J}^{\perp}_{12} \cap \mathbf{J}^{\perp}_{23} \cap \mathbf{J}^{\perp}_{31})\right\} \setminus \emptyset, \end{aligned} \tag{7.52}$$

where the order of the intersection operator and non-intersection are commutative.

Thus using simple boolean algebra, the set $\mathcal{O}$ can be written generally for κ components as:

$$\mathcal{O} \;=\; \wp\{\mathbf{v}_1, ..., \mathbf{v}_{\kappa}\} \setminus \emptyset, \tag{7.53}$$

where $\wp$ represents the intersection of all the combinations of subsets, selected from $\mathbf{v}_1$ to $\mathbf{v}_{\kappa}$, which are not an element of the remaining sets and not including the empty-set.

These sets can now be used to define projectors for each subspace of a component corresponding to a component's 'row block' of the articulation matrix (7.45). Furthermore, these sets must be ordered so that each column for each component is aligned with the corresponding subsets of the other components.

Using this definition, all subspace projectors are defined by the following simple equation:

$$\boxed{\forall\,(s_o \subset \mathcal{O})\,, \mathbf{Q}_{s_o} = \left(\bigcap_{\forall\{\mathbf{v}_a, \mathbf{v}_b\} \in s_o} \mathbf{J}_{ab}\right) \bigcap \left(\bigcap_{\forall\{\mathbf{v}_a, \mathbf{v}_b\} \notin s_o} \mathbf{J}^{\perp}{}_{ab}\right), \qquad (7.54)}$$

where s_o is a subset of the object set $\mathcal{O}$ and $\cap$ is the subset intersection operator, defined in equation (7.50). Note that this equation creates the link between the components and the joints (including virtual joints).

Articulation matrix

The articulation matrix fully defines a generalized coordinate basis for representing the minimal parameter vector $\dot{\mathbf{q}}$. In general terms, the Articulation Matrix corresponds to:

$$\mathbf{A} = \begin{pmatrix} \frac{\partial \mathbf{r}_1}{\partial \mathbf{q}} \\ \vdots \\ \frac{\partial \mathbf{r}_\kappa}{\partial \mathbf{q}} \end{pmatrix}. \tag{7.55}$$

The different subspaces determined in equation (7.54) are to be mapped to each component's subspace $\mathcal{C}_i$. In the case of a three component system, as in equation (7.52), component 1's subset would be given as:

$$\begin{aligned} \mathcal{C}_1 &= \{(\mathbf{v}_1 \cap \mathbf{v}_2 \cap \mathbf{v}_3), (\mathbf{v}_1 \cap \mathbf{v}_2 \notin \mathbf{v}_3), (\mathbf{v}_1 \cap \mathbf{v}_3 \notin \mathbf{v}_2), (\mathbf{v}_1 \notin \mathbf{v}_2 \notin \mathbf{v}_3)\} \\ &\equiv \left\{(\mathbf{J}_{12} \cap \mathbf{J}_{23} \cap \mathbf{J}_{31}), (\mathbf{J}_{12} \cap (\mathbf{J}_{23}^{\perp} \cap \mathbf{J}_{31}^{\perp})), (\mathbf{J}_{31} \cap (\mathbf{J}_{12}^{\perp} \cap \mathbf{J}_{23}^{\perp})), (\mathbf{J}_1^{\perp} \cap \mathbf{J}_2^{\perp} \cap \mathbf{J}_3^{\perp})\right\}. \end{aligned} \tag{7.56}$$

The Articulation matrix is defined according to equation (7.45) and taking into account the joint subspaces given by equation (7.46). The Articulation Matrix can thus be written as:

$$\boxed{\begin{gathered} \forall\, (i = 1..\kappa) \;\text{ and }\; \forall (s_o \in \mathcal{O}) \;\; \mathbf{A}_{i,s_o} = \delta_{\mathbf{v}_i \in s_o} \mathbf{Q}_{s_o}, \\ \text{where,} \quad \delta = \begin{cases} 1 & \text{if } \mathbf{v}_i \in s_o, \\ 0 & \text{otherwise,} \end{cases} \end{gathered}} \tag{7.57}$$

where (i, s_o) is in (row, column) format and indexes the elements of the Articulation matrix.

The columns of $\mathbf{A}$ are grouped in $2^{\kappa-1}$ sub-matrices of dimension p, each corresponding to a particular subset of object parameters. The rows are all in sub-groups of dimension 6 each corresponds to the six degrees freedom of each component.

The decoupling effect within the articulation matrix is directly related to the zeros represented by the δ operator in equation (7.57). In particular, there are more zeros introduced into the articulation matrix than in the case of a kinematic chain. In the case of a kinematic chain, zeros are only found in the upper diagonal of the articulation matrix, whereas, in the kinematic set case there are zeros both above and below the diagonal. This has an effect on the computational efficiency as well as the propagation of measurement error.

Example

Consider an object with three components ($\kappa = 3$). The first component has a x-rotational link to the second component 1m along the camera's z axis. The second component has a y-rotational link to the third 1m along the first joint's x axis (refer to Figure 7.8).

After computation of the different subspaces, the dimension of the objects subspaces are given in Figure 7.9. It is interesting to note that different subsets are decoupled automatically when the link reference frames are in an orthogonal position. It can also be noted that $(\mathbf{J}_{31} \cap (\mathbf{J}_{12}^{\perp} \cap \mathbf{J}_{23}^{\perp}))$ is always equal to zero, unless there is a physical link defined between component one and component three.

A Maple program was written to prove the subset orthogonality properties of different mechanical and spatial configurations. It automatically creates all the matrices needed to create the articulation matrix from joint definitions ($\mathbf{S}_j$, $\mathbf{r}_j$). A Venn diagram, shown in Figure 7.9, which displays the dimensions of the different subspaces is created from the matrix calculations. It can be downloaded from: *http://www.irisa.fr/lagadic/code-eng.html*

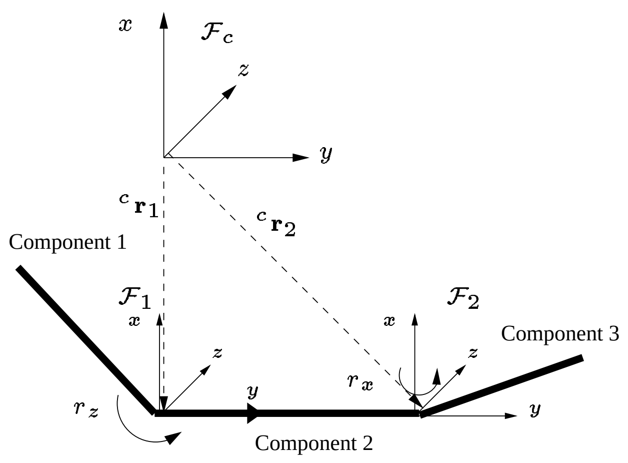


Figure 7.8: Example configuration of a camera and a three body system.

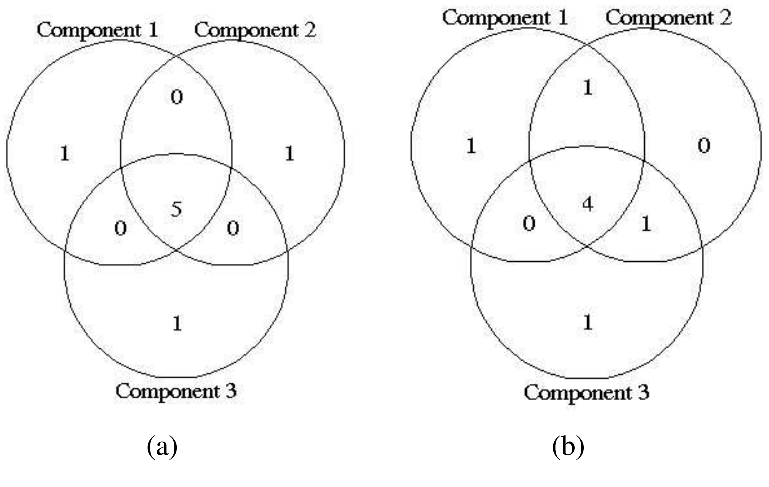


Figure 7.9: Dimensions of kinematic sets for a two component object with two rotational links. (a) Link locations are not orthogonal (b) Link locations are orthogonal

For an object with three components and two joints, and using the orthogonal subspace projectors given in equation (7.54), $\mathbf{A}$ is given by:

$$\mathbf{A} = \begin{pmatrix} \frac{\partial \mathbf{r}_1}{\partial \mathbf{q}_{123}} & \frac{\partial \mathbf{r}_1}{\partial \mathbf{q}_{12}} & \mathbf{0} & \frac{\partial \mathbf{r}_1}{\partial \mathbf{q}_{31}} & \frac{\partial \mathbf{r}_1}{\partial \mathbf{q}_1} & \mathbf{0} & \mathbf{0} \\ \frac{\partial \mathbf{r}_2}{\partial \mathbf{q}_{123}} & \frac{\partial \mathbf{r}_2}{\partial \mathbf{q}_{12}} & \frac{\partial \mathbf{r}_2}{\partial \mathbf{q}_{23}} & \mathbf{0} & \mathbf{0} & \frac{\partial \mathbf{r}_2}{\partial \mathbf{q}_2} & \mathbf{0} \\ \frac{\partial \mathbf{r}_3}{\partial \mathbf{q}_{123}} & \mathbf{0} & \frac{\partial \mathbf{r}_3}{\partial \mathbf{q}_{23}} & \frac{\partial \mathbf{r}_3}{\partial \mathbf{q}_{31}} & \mathbf{0} & \mathbf{0} & \frac{\partial \mathbf{r}_3}{\partial \mathbf{q}_3} \end{pmatrix} = \begin{pmatrix} \mathbf{Q}_{123} & \mathbf{Q}_{12} & \mathbf{0} & \mathbf{Q}_{31} & \mathbf{Q}_1^{\perp} & \mathbf{0} & \mathbf{0} \\ \mathbf{Q}_{123} & \mathbf{Q}_{12} & \mathbf{Q}_{23} & \mathbf{0} & \mathbf{0} & \mathbf{Q}_2^{\perp} & \mathbf{0} \\ \mathbf{Q}_{123} & \mathbf{0} & \mathbf{Q}_{23} & \mathbf{Q}_{31} & \mathbf{0} & \mathbf{0} & \mathbf{Q}_3^{\perp} \end{pmatrix}, \tag{7.58}$$

where $\mathbf{q}_{123}$, $\mathbf{q}_{12}$, $\mathbf{q}_{23}$, $\mathbf{q}_{31}$, $\mathbf{q}_1$, $\mathbf{q}_2$, $\mathbf{q}_3$ are vectors representing the sets of intersecting velocities and each components free parameters respectively.

It can be noted here that $\frac{\partial \mathbf{r}_1}{\partial \mathbf{q}_{123}} = \frac{\partial \mathbf{r}_2}{\partial \mathbf{q}_{123}} = \frac{\partial \mathbf{r}_3}{\partial \mathbf{q}_{123}}$ for the first column and similarly for the other columns. This is because the pose subsets (i.e. $\frac{\partial \mathbf{r}_2}{\partial \mathbf{q}_{123}}$) are defined within a common reference frame since the differential component in the Taylor series expansion of the error around each $\mathbf{r}$ (see Chapter 3) is determined for a common camera motion. An example of the different components of $\mathbf{A}$ can be found in a Maple example which can be downloaded at: *http://www.irisa.fr/lagadic/code-eng.html*

These sets are easily identified when referring to Figure 7.9. It can be verified that indeed the minimum number of parameters for a particular articulated object is:

$$\sum_{\forall s_o \in \mathcal{O}} \dim(\mathbf{Q}_{s_o}) = 6 + \sum c_j, \tag{7.59}$$

where c_j is the class of joint j.

The mapping $\mathbf{A}$ is indeed of dimension $(6\kappa) \times (6 + \sum c_j)$, remembering that κ is the number of components. Figure 7.9(a) and (b) show the case of a single object in different configurations. It can be seen that even though the dimensions of the subsets may vary for a same object, the total sum of the dimension of the subspaces remains constant. The reader can refer to [Comport04b] for the articulation matrix between two components and one joint.
The generalized parameter vector is therefore:

$$\dot{\mathbf{q}} = \begin{pmatrix} \mathbf{q}_{123}, & \mathbf{q}_{12}, & \mathbf{q}_{23}, & \mathbf{q}_{31}, & \mathbf{q}_1, & \mathbf{q}_2, & \mathbf{q}_3 \end{pmatrix}. \tag{7.60}$$

Once these velocities are obtained, they can be related back to the camera frame as in equation (7.45):

$$\begin{pmatrix} {}^c\mathbf{v}_1 \\ {}^c\mathbf{v}_2 \end{pmatrix} = \mathbf{A} \begin{pmatrix} \mathbf{q}_{123}, & \mathbf{q}_{12}, & \mathbf{q}_{23}, & \mathbf{q}_{31}, & \mathbf{q}_1, & \mathbf{q}_2, & \mathbf{q}_3 \end{pmatrix}. \tag{7.61}$$

It is important to highlight in this example the decoupling of the minimization problem. This is apparent in equation (7.57) where extra zeros appear in the Jacobian compared to the traditional case of a kinematic chain. In the particular case of three components and two joints, a kinematic chain has only 3 zeros in the articulation matrix compared to 9 in the kinematic set case.

7.5 Closed-loop Registration

In this section, a tracking control law is derived for the general class of articulated objects. The aim of the control scheme is to minimize the objective function given in equation (7.4). Thus, the error function is given as:

$$\begin{pmatrix} \mathbf{e}_1 \\ \vdots \\ \mathbf{e}_\kappa \end{pmatrix} = \mathbf{D} \begin{pmatrix} \mathbf{s}_1(\mathbf{q}) - \mathbf{s}^*{}_1 \\ \vdots \\ \mathbf{s}_\kappa(\mathbf{q}) - \mathbf{s}^*{}_\kappa \end{pmatrix}, \tag{7.62}$$

where $\mathbf{q}$ is a vector composed of the minimal set of parameters corresponding to the object's configuration and each $\mathbf{e}_i$ corresponds to an error vector for component i. $\mathbf{D}$ is a diagonal weighting matrix corresponding to the likelihood of a particular error within a robust distribution:

$$\mathbf{D} = \begin{pmatrix} \mathbf{D}_1 & & 0 \\ & \ddots & \\ 0 & & \mathbf{D}_\kappa \end{pmatrix},$$

where each matrix $\mathbf{D}_i$ is a diagonal matrix corresponding to a component which has weights w_j along the diagonal. These weights correspond to the uncertainty of the measured visual feature j. The computation of the weights have been described in Chapter 6.

If $\mathbf{D}$ were constant, the derivative of equation (7.62) would be given by:

$$\begin{pmatrix} \dot{\mathbf{e}}_1 \\ \vdots \\ \dot{\mathbf{e}}_\kappa \end{pmatrix} = \begin{pmatrix} \frac{\partial \mathbf{e}_1}{\partial \mathbf{s}_1}\frac{\partial \mathbf{s}_1}{\partial \mathbf{r}_1}\frac{\partial \mathbf{r}_1}{\partial \mathbf{q}} \\ \vdots \\ \frac{\partial \mathbf{e}_\kappa}{\partial \mathbf{s}_\kappa}\frac{\partial \mathbf{s}_\kappa}{\partial \mathbf{r}_\kappa}\frac{\partial \mathbf{r}_\kappa}{\partial \mathbf{q}} \end{pmatrix} \dot{\mathbf{q}} = \mathbf{D}\mathbf{L}_\mathbf{D}\mathbf{A}\dot{\mathbf{q}}, \tag{7.63}$$

where $\dot{\mathbf{q}}$ is the minimal velocity vector, $\mathbf{A}$ is the articulation matrix describing the mapping in equation (7.57) and $\mathbf{L}_\mathbf{D}$ is the block diagonal interaction matrix as in equation (7.44) given as:

$$\mathbf{L}_\mathbf{D} = \begin{pmatrix} \frac{\partial \mathbf{s}_1}{\partial \mathbf{r}_1} & & \\ & \ddots & \\ & & \frac{\partial \mathbf{s}_\kappa}{\partial \mathbf{r}_\kappa} \end{pmatrix} = \begin{pmatrix} \mathbf{L}_{\mathbf{s}_1} & & \mathbf{0}_6 \\ & \ddots & \\ \mathbf{0}_6 & & \mathbf{L}_{\mathbf{s}_\kappa} \end{pmatrix}. \tag{7.64}$$

If an exponential decrease of the error $\mathbf{e}$ is specified:

$$\dot{\mathbf{e}} = -\lambda \mathbf{e}, \tag{7.65}$$

where λ is a positive scalar, the following control law is obtained by combining equation (7.65) and equation (7.63):

$$\boxed{\dot{\mathbf{q}} = -\lambda(\widehat{\mathbf{D}}\widehat{\mathbf{L}}_\mathbf{D}\widehat{\mathbf{A}})^{+}\widehat{\mathbf{D}}(\mathbf{s}(\mathbf{q}) - \mathbf{s}^*), \qquad (7.66)}$$

where $\widehat{\mathbf{L}}_\mathbf{D}$ is a model or an approximation of the real matrix $\mathbf{L}_\mathbf{D}$. $\widehat{\mathbf{D}}$ is a chosen model for $\mathbf{D}$ and $\widehat{\mathbf{A}}$ depends on the previous estimation of the object configuration $\mathbf{q}$. Once these velocities are estimated they can be related back to the camera frame as in equation (7.45). The stability of this system can be proven in the same way as the various cases already considered in Chapter 3 and 6.

7.6 Articulated Object Tracking Experiments

In this section four experiments are presented for tracking of articulated objects in *real* sequences. The results presented here have been performed using a monocular vision system. Local tracking is performed via a 1D oriented gradient search to the normal of projected contours as has been detailed in Chapter 4.

It should be noted that the programming task involved in obtaining these results was not trivial due to the symmetric nature of the model. Indeed it involved modeling the storage of information within an object oriented C++ hierarchy of feature sets and correct implementation of the interaction between these feature sets represented as a graph of feature sets. The basic implementation of the algorithm gives the following pseudo-code:

```
1. Obtain initial pose.
2. Capture a new image.
3. Project the model onto the image.
4. 1D Search for corresponding points normal to the projected contours.
5. Determine the distance errors $\mathbf{e}$ in the image.
6. Calculate $(\widehat{\mathbf{D}}\widehat{\mathbf{L}}_{\mathbf{D}}\widehat{\mathbf{A}})$.
7. Determine minimal subspace velocities as in equation (7.66).
8. Update the generalized parameters as in equation (7.45).
9. Re-project onto image using each component's pose & goto 5 until the error converges.
10. Repeat to 2.
```

The information included in the following results are aimed at proving the advantages of our articulated tracking method. The experiments test for the following performance factors:

- 3D motion - In order to show that this method is capable of handling all types of 3D movement, both camera and object motion as well as articulated motion have been introduced into each experiment.
- Minimal parameters - Velocity plots show the evolution of the minimal parameter kinematic sets which are used for the tracking. At the same time the velocities attained by this system are given.
- Rank deficient visual information - one of the advantages of this method is that components with little visual information can be tracked stably and robustly since they are now supported by visual information from other components (unlike in the case of the Lagrange Multipliers or individual object tracking). Furthermore, this interdependence of components allows one to avoid badly conditioned Jacobian or noise in the image which would normally lead to failure. This involves using different configurations of distance-to-line type features including simple lines, planes and full 3D objects and comparing with individual object tracking.
- Joint types - linear and non-linear joints have been implemented so as to show the general class of joints which can be defined using our velocity constraint matrix.
- Robust estimation - also enhances the robustness of the method to external occlusion and miss-tracking.
- Precision - the performance of our method is evaluated by looking at the re-projection of the model onto the real scene which presents the data in a visual manner. Furthermore, a plot of the error norm is given, where the ground truth for the error norm is governed by the noise in the image. A pose

plot is also given in order to show the large movements imposed within the experiment as well as the smoothness of the estimated pose.

7.6.1 Rotational Link

This first experiment was carried out for a class 1 type link with a single degree of rotation on the x axis. Images at different points of one of the test sequences are shown in Figure 7.10. The constraint vector was defined as in equation (7.16) and the object frame was chosen to coincide with the joint frame. This sequence, along with others performed using the same object model, have shown real time efficiency with the tracking computation taking on average 25ms per image. In the plots given in Figure 7.11, individual tracking (a) is compared to articulated tracking (b) using kinematics sets. Both the hinge and the object were displaced during tracking without failure in the kinematic set case.

In the case of the hinged door visual information has been removed from the door to demonstrate one of the benefits of the kinematic set approach. In part of this experiment only two parallel lines which are very closely spaced are used as visual information for the door component as seen in 7.10(a) and (b). In this case the visual information from the door only constraints 5 degrees of freedom. Furthermore, since the lines are closely spaced, the rotational parameter around the line is also weakly conditioned. Other than this there are also many shadow artifacts and false lines apparent in the sequence.

Individual tracking failed in this situation. It can be mentioned that for the hinge, tracking failure is more easily observed by visual inspection since the tracking continues without fully breaking down in the plots. Even so, the plot of the individual tracking shows different velocities to that of the articulated case. This failure was due to a shadow on the box which was better defined than a contour on the object. The miss-tracking due to this shadow broke down the estimation gradually in the individual case. In the kinematic set case the miss-tracking did not disturb the pose estimation because more information from the other component balanced out these errors.

The Lagrange Multiplier method would also fail because it depends on the pose for each component before applying constraints in a second step. Furthermore, the kinematic chain approach would also fail if the root component was chosen as the one with miss-tracking. It can be seen that the kinematic set approach succeeds by better distributing the information between components so as to constrain the uncertain parameters of the first component. The improvement in estimation precision can be observed between the two plots shown in Figure 7.11.

7.6.2 Prismatic Link

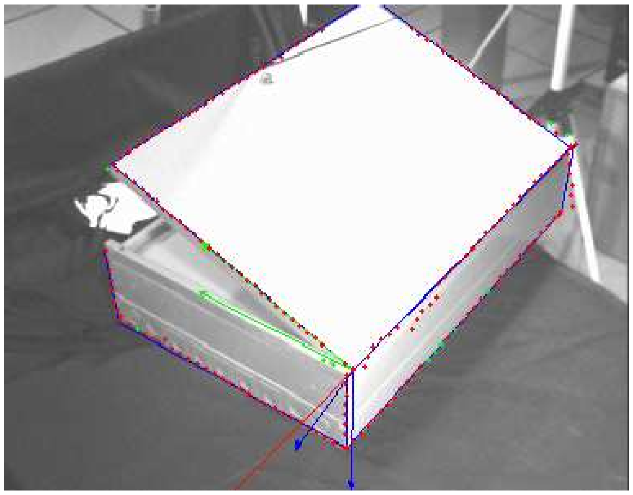
This second experiment was carried out for class one link with a single degree of translation on the x axis. The constraint vector was defined as a prismatic link along the x axis:

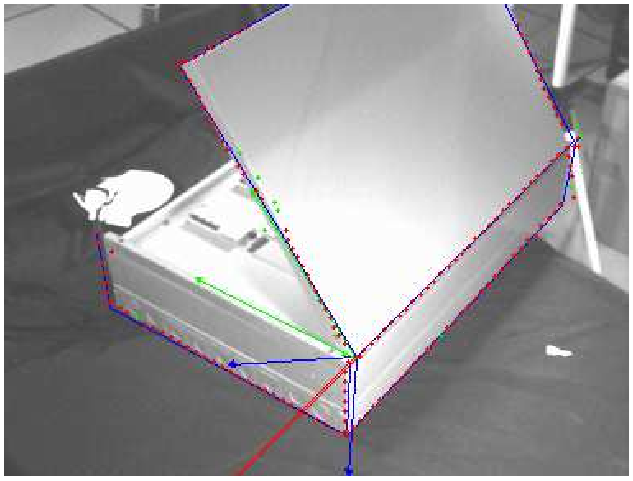
$$\mathbf{S}_{12}^{\perp} = (1, 0, 0, 0, 0, 0), \tag{7.67}$$

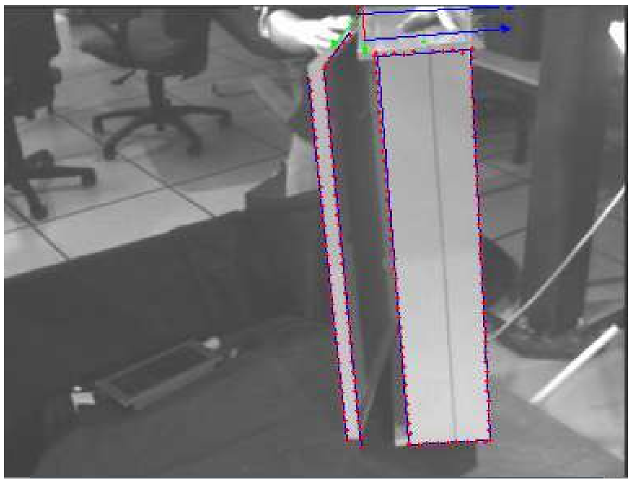
and the object frame was chosen to coincide with the joint frame.

The sequence, shown in Figure 7.12, also runs in real time with the tracking computation taking on average 20ms per image. Both the camera and the slide were displaced simultaneously. It can be seen in the image that the alignment of the projected contour with the actual image of the object is visually acceptable for the kinematic set case.

As in the previous experiment, the estimated parameter velocities are compared in Figure 7.13 for individual component tracking (a) and kinematic set tracking (b). It can be seen that tracking fails in 7.13(a) at


(a)


(b)


(c)

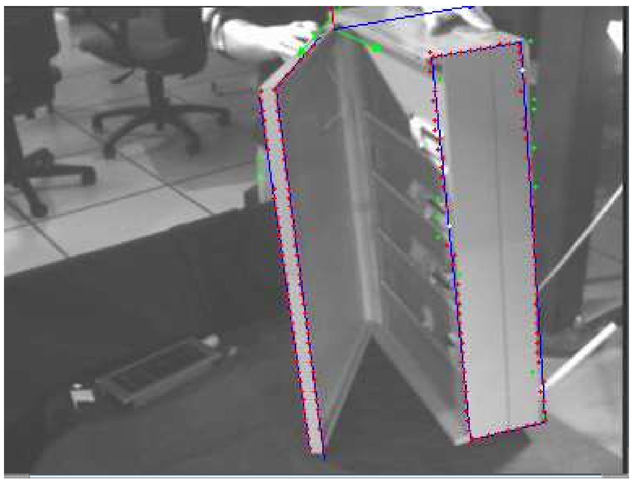
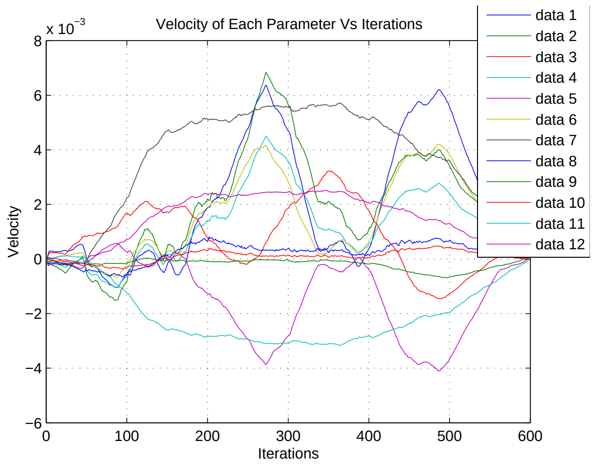
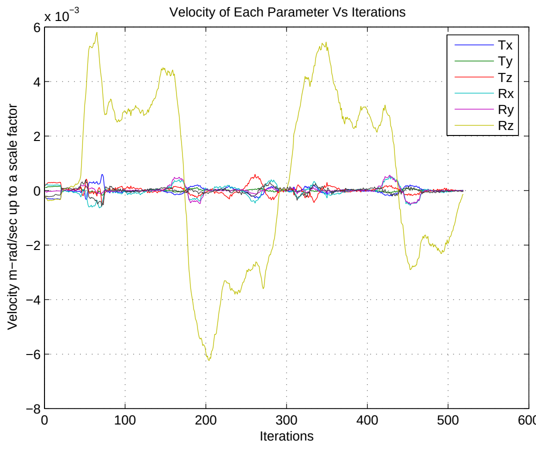

(d)

Figure 7.10: Rotation of a hinged door while undergoing various movements. In this and all the following figures the reference frames for each component are shown in green, blue and red for each axis. The projection of the CAD model contour onto the image using the estimated pose parameters is shown in blue. It can be seen that the projected position of the contours are visually acceptable. The points rejected by the M-estimator are shown in green. Changes in illumination on the surface of the door can be seen between (a,b) and (c,d) and partial occlusion by a hand is shown at the top of the box in (c) and (d).



(a)



(b)

Figure 7.11: The velocity parameters for tracking of a hinged door. In these plots only one component of the object has been moved and thus the only movement visible is the movement of the rotation of the door on its axis (see Figure 7.10(a) and (b)). The plots for the full object and hinge movement shown in the images in Figure 7.10(c) and (d) are not given here, however, later experiments will be show tracking of both components and the camera simultaneously. The units are in meters-radians per Δt, where Δt is the time for 1 frame capture of the camera. In these experiments frame capture is approximately $30fps$, therefore, $\Delta t = 1/30$. (a) Objects tracked separately with 12 parameters. In this case tracking failed almost immediately due to severe shadow interference. It can be seen that both objects have been estimated to be in movement since there are more than 6 significant velocities at a time. (b) Articulated object tracking with 7 parameters. It can clearly be seen that only rotational hinge movement was performed. This movement corresponds to the opening and closing of the door which is apparent in the figure. The maximum velocity of the door was approximately $0.15 rad/s$

around 350 iterations when individual tracking is used. In this case, there was not enough visual information from the rail to constrain a 6 parameter pose. Indeed the object model is essentially composed of two parallel lines in close proximity. In 7.13(b) tracking is successful. Once again the kinematic set approach is able to draw on the other component to fully constrain the tracking.

7.6.3 Helical Link

This experiment, reported in Figure 7.14, was carried out for class one link with helical movement simultaneously along and around the z axis. This type of link is more complex due to the nonlinear articulation matrix mapping from the different component twists to the minimal subspaces. The constraint vector was defined as in equation (7.18) and the object frame was chosen to coincide with the joint frame. Note that the constraint vector was defined by taking into consideration that for $10 \times 2\pi$ rotations of the screw, it translated 4.5 centimeters along the z axis.

Tracking of this object also ran in real-time with the main loop computation taking on average 25ms per image. Once again individual and articulated tracking are compared and the resulting plots are shown in Figure 7.15. In this experiment a case is presented where individual tracking succeeds, however, it should be noted that this sequence was chosen specifically for comparative purposes. In many similar sequences individual tracking often fails completely due to the limited contour information, background noise or occlusions. When tracked simultaneously with the plate, as an articulated object, the tracking of the screw is also based on the measurements of the plate making the tracking more stable and computationally efficient. It can be seen that tracking continuous throughout a long sequence of approximately 1100 images. Robust weights were calculated for each component's feature set individually as outlined in Chapter 6.

In the screw case, the background is essentially black against a silver object making strong visual features. Furthermore, each component is well constrained individually, therefore individual tracking succeeds, however, the computation time is twice that of the articulated case.

7.6.4 Robotic Arm

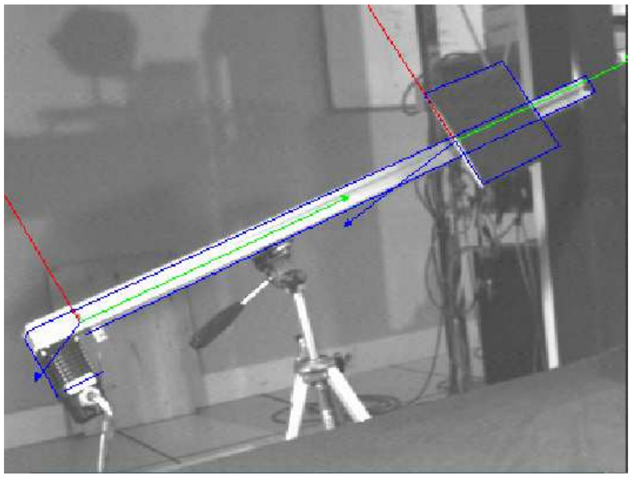
An experiment was carried out for two class one links on a robotic arm so as to generalize the approach to 3 or more joints. The articulations tracked were rotational links around the z and x axes. The constraint vectors were each defined by a pose and a constraint matrix as given in equation (7.16). Note that the constraints are no longer defined in the same coordinate system as in the previous case.

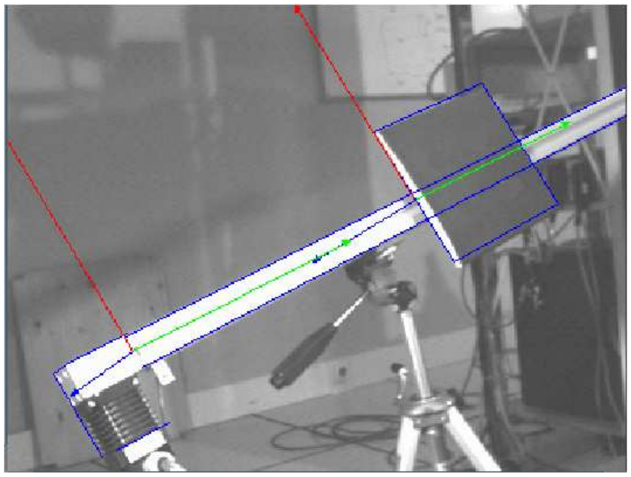
The sequence, shown in Figure 7.16, also displays real time efficiency with the tracking computation taking on average 25ms per image. As mentioned in the caption of Figure 7.17, only two line features are used to track the forearm components and are therefore not full rank for rigid pose calculation. Consequently the rigid pose plots could not be compared. As in the previous experiment, the kinematic set formulation is able to handle tracking this situation.

In what follows, a table comparing and showing the improvement in accuracy for tracking multi-body objects is given.

type / mm	Hinge	Slide	Screw	Robot
Individual	miss-track	1 badly conditioned	2.2	Rank deficient
Articulated	2.2	0.8	2	2.2

Table 7.1: A comparison of the error norm between individual tracking methods and articulated tracking via kinematic sets


(a)


(b)

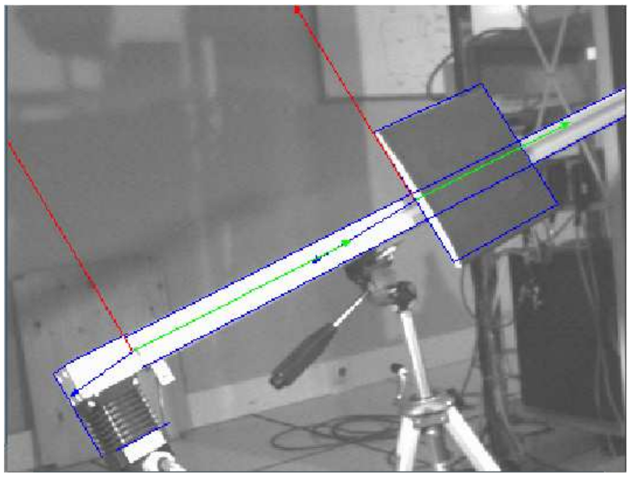
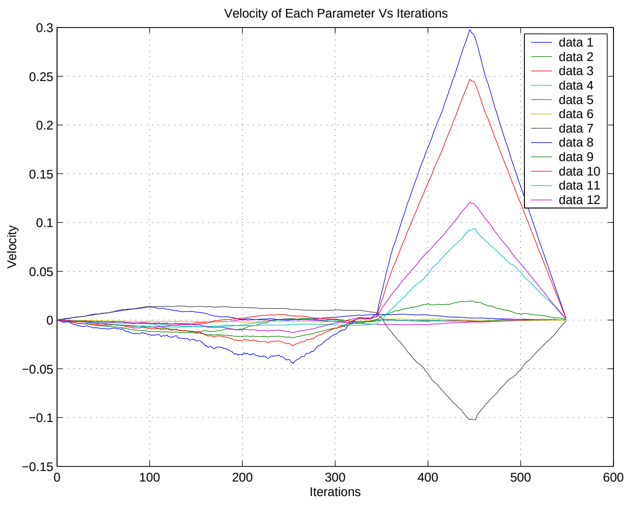
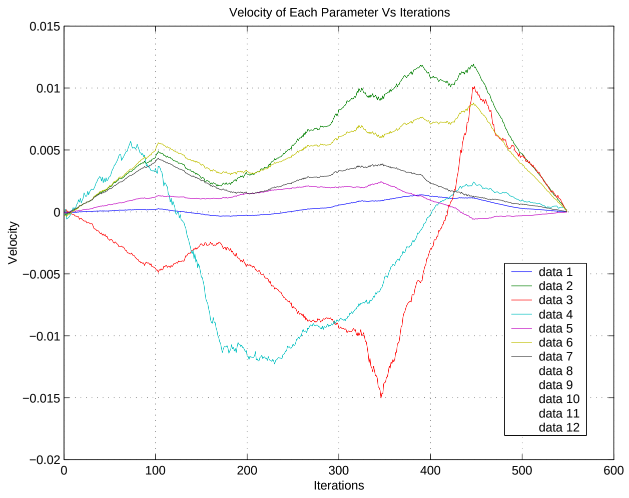


Figure 7.12: Translation along a sliding mechanism while the camera is moving. The first component is the rail and the second is the square which slides along the rail. It should also be noted that all CAD models were measured very roughly by hand using a simple ruler. Therefore, it can be seen that the tracking is able to support a certain degree of modeling error also.



(a)



(b)

Figure 7.13: The velocity parameters for a sliding mechanism: (a) Objects tracked separately with 12 parameters. It can be seen from the large spike in velocity that the tracking fails after 350 iterations. This is due to a badly conditioned Jacobian since visual information of the two parallel, closely spaced lines, belonging to the rail, provide unstable disparity information. The maximum velocity of the slide was approx. $0.45m/s$. (b) Articulated object tracking with 7 parameters. Note that translational movement of the object was performed along with movement of the camera for this experiment.

7.7 Conclusion

The method presented in this chapter demonstrates an efficient and accurate approach to real-time tracking of complex articulated objects. A framework is given for defining any type of mechanical link between components of an object based on constraining the velocities of rigid components. These constraints have been used to define a kinematic chain, a Lagrange Multiplier and a new kinematic set approaches to articulated motion modeling. The kinematic set approach has been derived via minimal object subspaces which adhere to a Lagrange d'Alembert formulation where the constraints act parallel to the forces (or velocities) in the system. It has been shown that it is possible to partially decouple the interaction between articulated components using this approach. This approach has been derived and implemented in a closed loop control law for 3D articulated object tracking. Subsequent robustness, computational efficiency and visual precision have been demonstrated. This fundamental approach has founded the basis for many new and improved applications in many areas.

In perspective, it would be interesting to explore the possibility of treating closed link systems using kinematic sets for visual servoing. First of all it would be interesting to study the use of a kinematic set approach instead of traditional kinematic chains for modeling a robotic system. In visual servoing it would also be interesting to implement a closed-loop control law for an eye-to-hand based visual servoing system by closing the control loop with a model based tracking of the robot and its end effector. This could be applied to a precise grasping task which is not always possible using classical eye-in-hand based methods. Furthermore, it would be interesting to take advantage of a wide-area view using an external eye-to-hand camera, combined with an eye-in-hand camera so as to eliminate ambiguity between object and camera movement and also draw upon the advantages of each approach.

Constraint matrices are well defined mathematical concepts that have a very large application domain.

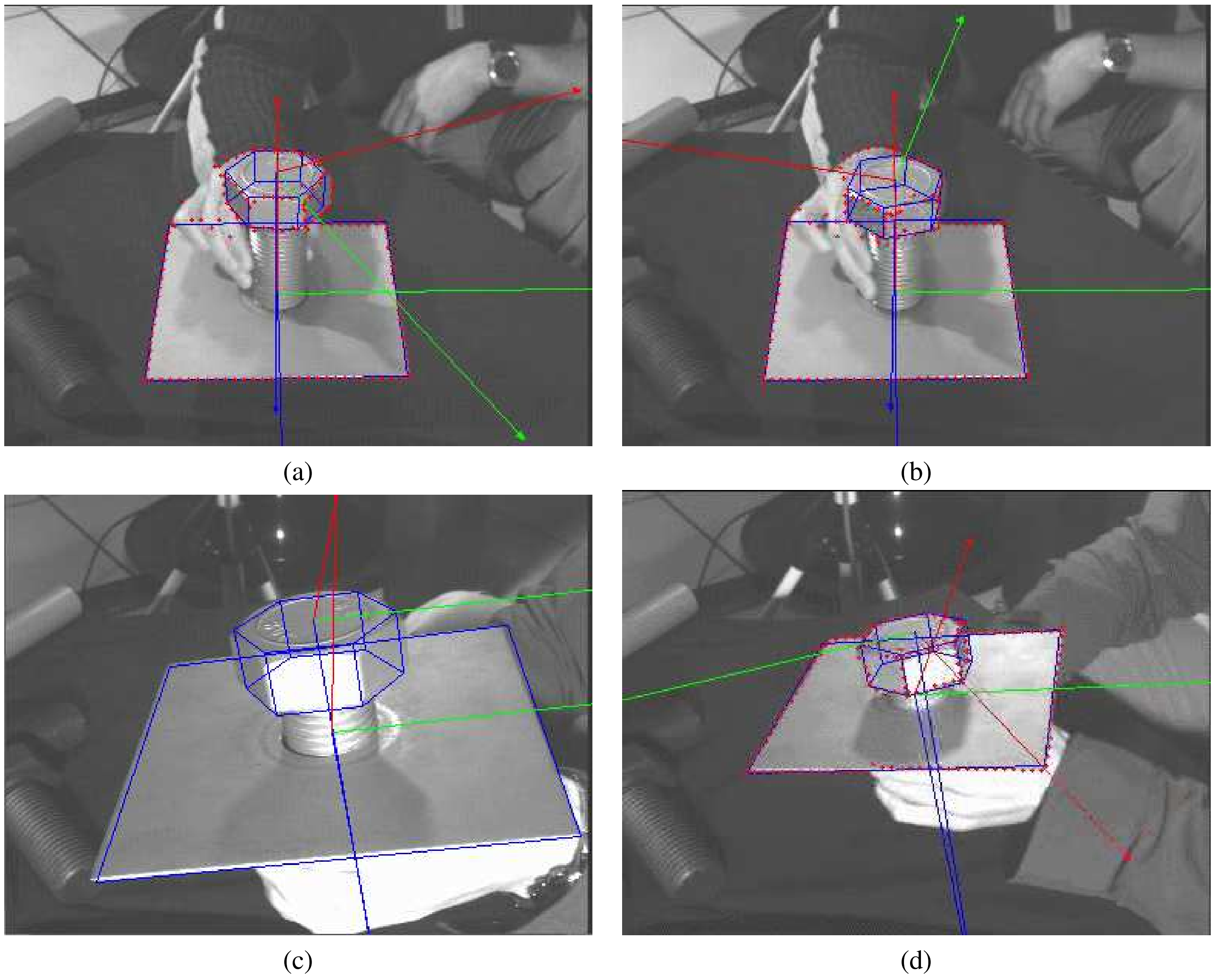
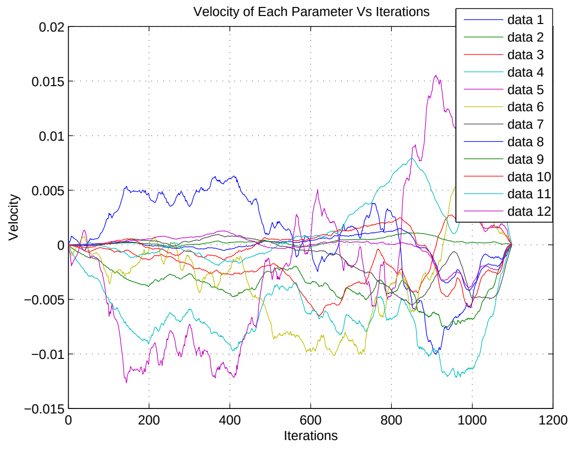
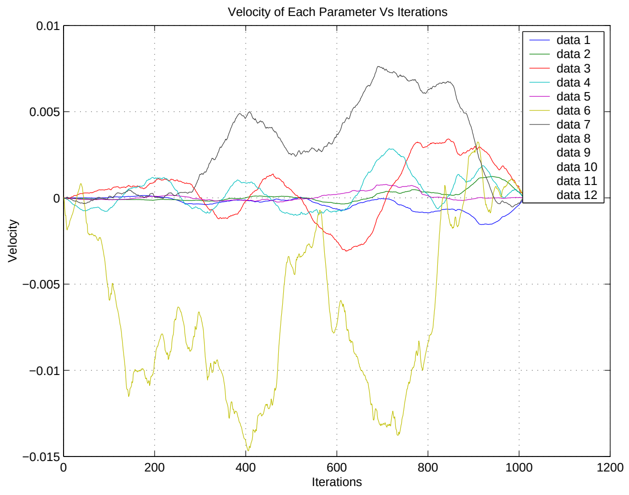


Figure 7.14: Helical movement of a screw while the screw and the platform are simultaneously in movement. Partial occlusion is made by a hand and a wide range of positions and configurations are tested. The object is moved and manipulated by hand and smooth movements of the projected model are observed without the need for filtering.





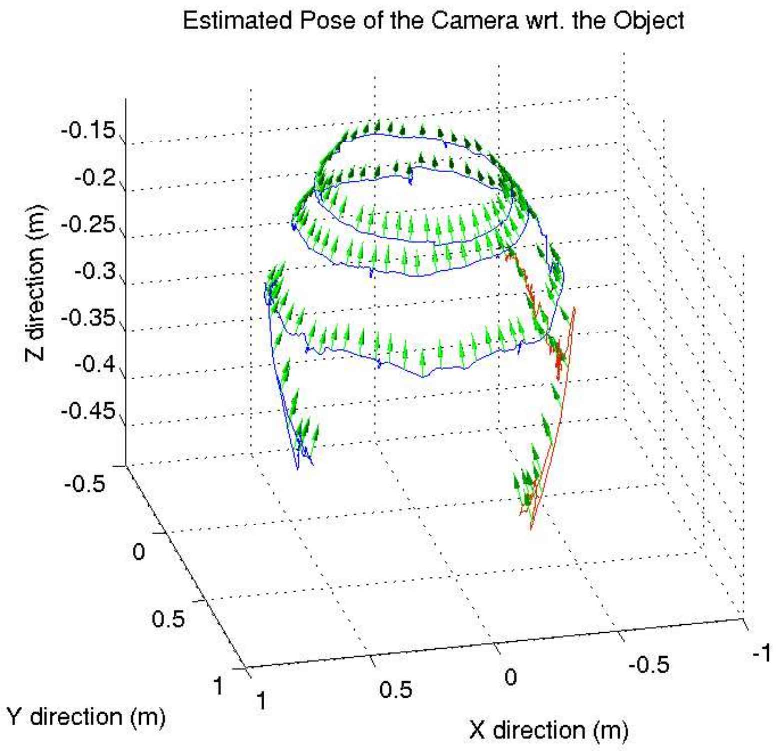


Figure 7.15: The velocity parameters for helical movement of a screw. (a) Objects tracked separately with 12 parameters. (b) Articulated object tracking with 7 parameters. (c) The pose of both the screw (in blue) and the plate (in red) of the components w.r.t. the camera. Note that helical movement of the screw was performed by turning the screw and the object was also moved during these experiments. The helical movement is in green in figure (b). Any jitter observed in the pose can be attributed to hand jitter. It can be seen that only one parameter more than required for a rigid object is needed to estimate the helical movement along and around the z axis.

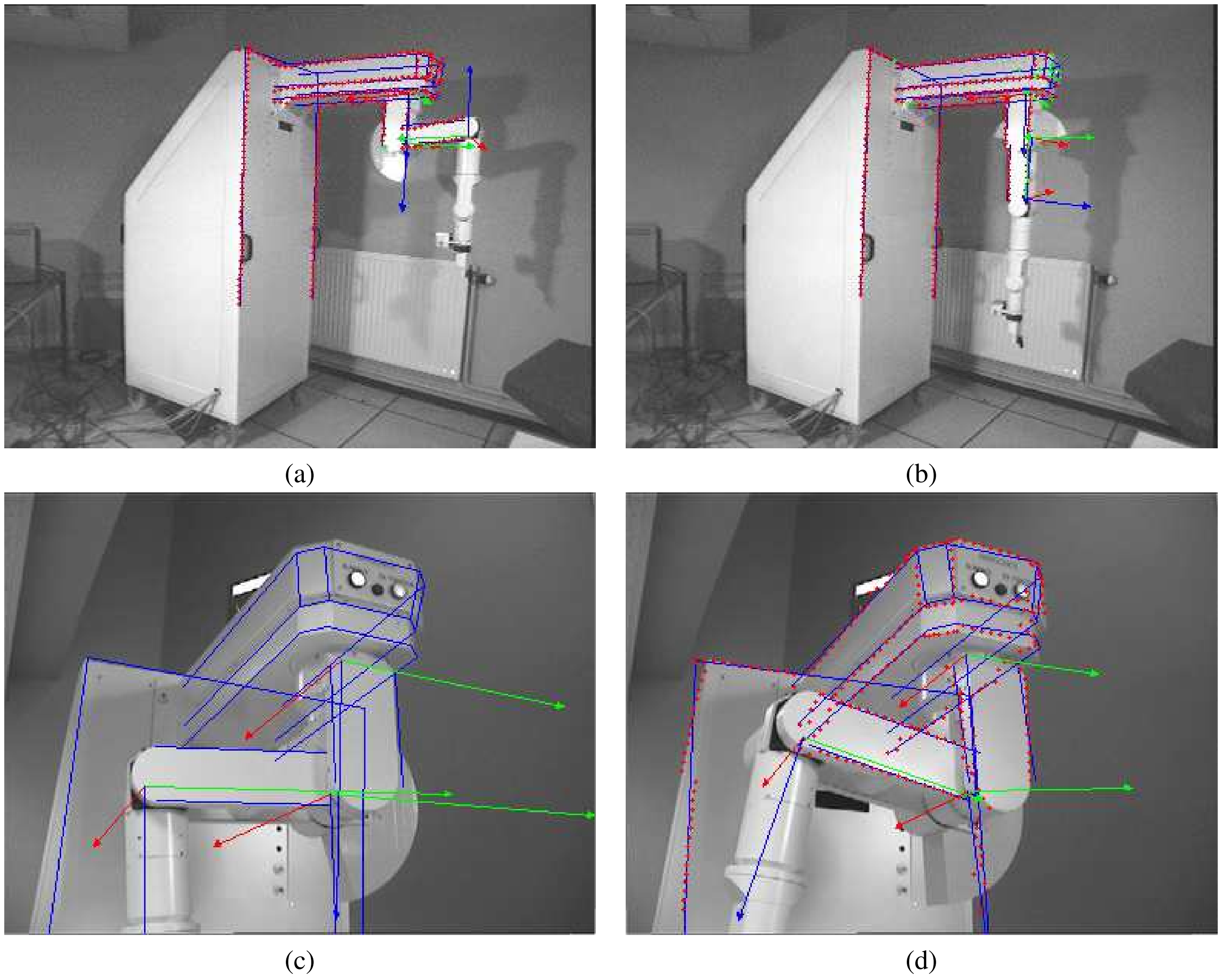


Figure 7.16: Movement of a robotic arm with two degrees of freedom. In this example, it can be seen that this approach generalizes to more than one parameter. In (a) and (b) it can be seen that even when very little visual information is available, the estimation of unknown joint parameters can be achieved by relying on the more stable estimation obtained from the base of the robot. Various positions and configurations have been tested.

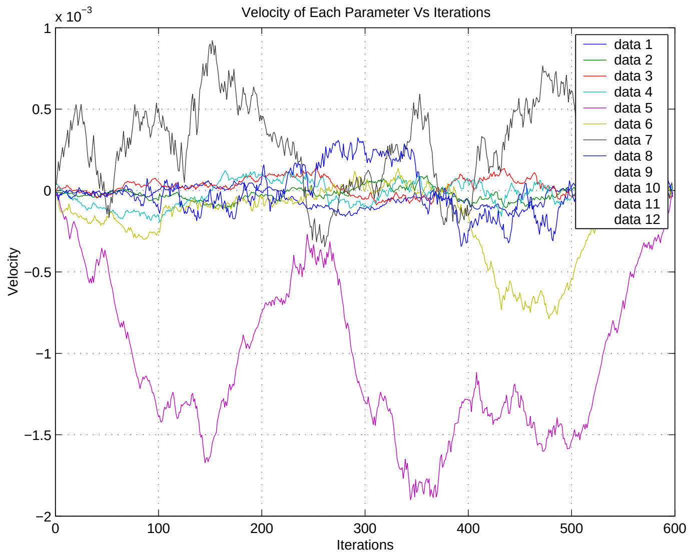


Figure 7.17: The estimated velocity parameters for tracking of a robot arm with 8 parameters. Note that two fore-arm components are rank deficient so it was not possible to estimate their pose individually.

The constraint based formalism presented in this chapter provides the basis for many other interesting applications. Indeed, since visual servoing notations have been used, the velocity constraint based formalism can now be easily applied to visual servoing applications. More particularly, it would be interesting to pursue further the duality between pose estimation techniques and visual servoing techniques based on velocity constraints. This could lead to interesting avenues for constraining a robot along a particular trajectory or even constraining the configuration space between different cameras. This could benefit various applications such as passively constraining medical robot manipulators for surgical tasks. Another important constraint that could be considered is the joint limit constraint [DiFranco99] so that the range of allowable joint angles (or translational displacements) is limited.

Finally, another interesting application would be to track human figures or even hands for numerous applications including human computer interaction and human movement analysis. Tracking of human hands would provide the ideal computer interface for augmented reality based application. In this chapter, manufactured objects were considered with a well defined geometric structure and contours which could be parameterized precisely. As a consequence the kinematic set algorithm provides a much more precise and robust estimation of location and position than that which can be obtained from human tracking. The contour approach, however, remains general and precise by using edge information. The model proposed in this chapter could therefore provide interesting avenues for research in the area of human motion analysis when combined with existing techniques in this area. The fact that human figures are highly articulated structures would make the kinematic set approach an ideal candidate for improving performance over traditional chain structures. This is because decoupling gained with the kinematic set approach would have even greater effect due to the exponential increase in zeros within the articulation matrix in direct relation to the increase in the number of parameters. In order to track a human body it would also be necessary to further relax rigid-body constraints such as introducing some degree of elasticity for modeling clothing and human skin. Indeed, elastic and other sorts of physical parameters could either be introduced into the constraint matrix as known *a priori* information or added as extra parameters to be estimated. It would also be interesting to implement a kinematic set approach with multiple cameras so that any ambiguities in the tracking process may be avoided, rendering the tracking process more robust.

CHAPTER 8

Conclusion

8.1 Overview

In conclusion, a method has been presented for robustly tracking complex objects in an image sequence at a high processing rate (50Hz) using visual servoing techniques. The relationship between a virtual model of the 3D world and the visual perception of this world using a monocular camera sensor has been established. Real-time pose estimation has been achieved by using visual servoing techniques for real-time 3D tracking by performing visual servoing virtually where the Virtual Visual Servoing control law is equivalent to an iterative minimization procedure. This approach has proven to provide precise estimates via a closed-loop control law. Furthermore, different scene models have been investigated including camera-based and object-based movement models showing the respective sensitivity to noise in the object model and image measurements. Hidden surface removal has been shown to allow tracking of complete 3D objects which are never fully visible from a single camera.

Various types of 1D distance information have been considered and corresponding interaction matrices have been derived. These features have included distances to the perspective projection of 3D points, lines, planar contours (ie. circles) and volumetric surfaces (ie. cylinders, spheres), however, any polyhedral model could be considered under this framework. Distance features of different types were effectively combined in a redundant, simultaneous estimation procedure. The results have proven to be accurate and computationally efficient.

The oriented Moving Edges algorithm was used to provide efficient low level edge tracking with convolution efficiency. It was shown how the scale and uncertainty in these low level features could be propagated to the global pose estimation. A robust estimator then allowed for a decision to be made as to which features belong to the object contour and those that do not.

High robustness has been demonstrated with a robust model-based approach using a statistical M-estimator that was integrated into the visual control law via an iteratively re-weighted least squares methodology. This was shown to benefit both tracking and visual servoing applications by remaining robust to various sources of external error including occlusion, changes in illumination and miss-tracking. The Median Absolute Deviation (MAD) was used as an estimate of the standard deviation of the inlier data and Tukey's influence function was used to completely reject the effect of outliers. This combination is advantageous because of its high efficiency, high breakdown point and desirable influence function. The robustness and stability of the control law was shown to be dependent on a subsequent measure of position uncertainty. Furthermore the convergence criteria of the control law were investigated.

In a last part, a new method has been proposed for accurately modeling and tracking articulated objects in real-time. In a first part, multiple independent objects were defined. Velocity constraint matrices were then defined for velocity twist vectors so that the rigid-body formalism could be constrained to include non-rigid objects. An articulation Jacobian was then based on minimal parameter subsets which were derived by applying velocity constraint matrices to rigid components. The model has been shown to track various 3D objects with different joints such as rotational, prismatic and helical. Real-time tracking has been shown to work in extreme situations with limited visual information where the multi-object tracker fails. Furthermore, it has been shown that it is possible to partially decouple the interaction between articulated components using this approach. Subsequent computational efficiency and visual precision have been demonstrated.

A video see through Augmented Reality system has been presented based on the 3D tracking method presented in this thesis. The system has been shown to be capable of treating complex scenes in a realistic live real-time environment without the need to modify the environment. The closed loop tracking scheme has provided accurate and stable estimation of the pose so that virtual objects are rendered onto a real image in a visually acceptable way. The system is based on CAD models of manufactured objects or urban environments.

Robust visual control has been applied to both 3D tracking and visual servoing allowing continuous estimation of the uncertainty in visual features and rejection of outliers during each step of a real-time visual servoing task. Results of 3D tracking applied to an image-based visual servoing control law have also been presented. The 3D tracking approach has introduced advantages such as improved stability and behavior of the system since the depth parameter is estimated. Furthermore, this provides all the essential information for globally convergent methods such as $2.5D$ visual servoing. Most importantly, however, is the fact that by using this method, visual servoing tasks can be carried out even when the final parts of the object are not initially visible, therefore making the visual control more flexible. Furthermore, by minimizing using IBVS, the target can more easily be kept within the image frame.

In what follows is a more global summary of the important points that have been raised in this thesis:

- Motivation - This work has been closely related to applied research in the domains of Augmented Reality and Visual Servoing, which has provided many motivating factors towards the progression and direction of this thesis. Advances in research is therefore easily tied to user-end applications such as: medicine, manufacturing and industrial processes and human computer interfaces. An important motivating factor has been the collaboration with the industrial partner Total Immersion based on Augmented Reality.

- Continuity - It has been seen that the continuity of the estimation procedure has a qualitative impact on its estimation precision. This has been seen at the estimation level where a closed loop control law has been used to estimate the pose simultaneously from multiple distance features. This has also been seen in non-rigid object model where all components have been considered simultaneously.

- Symmetry - It has been seen that real world physical phenomena are more directly modeled via symmetrical approaches which have several advantages including improved precision and efficiency. This has been seen in Chapter 3 where camera-based and object-based estimation models have been considered and related to asymmetric estimation processes. Furthermore in Chapter 7 it has been seen that Kinematic Sets provide an improved articulation Jacobian. It can also be noted from this that mathematics and computer computation are largely based on serial processes which make the definition of symmetric entities difficult.

- Generality - The robust statistics approaches open the door to a class of more powerful and general estimation procedures that are more widely applicable than traditional least squares approaches. Indeed, robust statistics is based on the modification of the distribution of the objective function so as to treat a general class of errors. Therefore, this potentially defines a general class of “everything” except the objective function. In this way, it would be very interesting to use this generality to design methods which are much more widely applicable.

8.2 Perspectives

The different areas studied in this thesis have provided a working technique for tracking rigid and non-rigid objects in real time, however, many problems remain to be solved before a complete working system is available for real world applications. These problems include:

- Visual information - Extracting useful and robust visual information is a central problem. The method presented in this thesis could be improved by: deriving interaction matrices for more complex contours; finding a solution to the problem of multiple contours in a search path; using all important visual information such as color and texture and using different sensors to improve performance.

- 3D CAD Model - The *a priori* model has allowed the tracking procedure to remain robust within a 3D environment. This model could be improved by obtaining it automatically and updating it online. Furthermore, more complex 3D models could be considered and different types of information could be added such as object texture and edge type.

- Estimation - The problem of pose estimation is a central problem that brings together the various aspects of this thesis. In perspective, automatic initialization could be considered, robustness could be improved by developing methods with higher breakdown points that are able to handle multiple object occurrences,

- Non-rigid Motion - the problem of non-rigid motion using velocity constraints and kinematic sets provides a formulation with promising results for different application. Research in this area could be continued by considering robotics applications such as parallel robots or considering other applications such as human motion tracking.

These problems are somewhat overlapping and solving them may be achieved by multiple methods. Indeed treating the more general problems will have an effect on many of the problems mentioned here. Of course the choice of any such a method would necessitate to take into account many performance criteria

- computational efficiency,

- precision,

- robustness,

In the following subsections the various points made here are investigated in more detail.

8.2.1 Visual Information

Extracting visual features with high information content is a central factor in the design of tracking algorithms. In this thesis, contour information has be used as a basis for estimating and tracking the 3D pose of an object because they provide a rich source of geometric information in an efficient manner. There are many real situations that exist where little other information is available which is particularly the case when only homogeneous surfaces are present. In this thesis only certain types of parametric contours have been considered and an extension would be to derive interaction matrices for more complex contours such as has been done in [Pressigout04] where B-Splines or Nurbs were considered. Another approach to treat more complex contours is to extract higher level visual features directly from the image such as in [Tahri05] where moment based features such as the area or different orders of moments were considered to obtain visual features that are invariant to different 3D motions. The advantage of this approach is that it decouples the interaction between the pose and the visual features and provide stable features and near linear globally convergent estimation properties. However, in order to make this approach computationally efficient it would be necessary to treat the complex segmentation step. This could be overcome by developing the relationship between different contours and their corresponding moment representation as was used in the case of an ellipse in Chapter 5.

In the case of contour features, a significant problem occurs when there are multiple edges which are detected within the search path. In this thesis, the inter-frame object movement and subsequently the search path have been assumed small so that this problem rarely arises. Even so, this problem occurs when there is a hidden edge that is appearing at the same position as the occluding edge. In this case it has been necessary to continue considering an appearing edge hidden until it de-passes the search path to the normal of the occluding edge. However, it would be interesting to solve this problem and extend this approach to fast moving objects. Different solutions exist, including using recent particle filtering techniques [Isard98a, Isard98b, Doucet00, Doucet01] at either the low level with the state-space being the search path to the contour normal or at the high level with the state-space being pose of the system. Of course, it would be necessary to further investigate efficiency and real-time performance issues using this sort of methods. An interest of this approach is that it provides a means of maintaining multiple hypotheses simultaneously making the system more robust to ambiguity and perturbation. Another advantage is that this framework lends easily to the fusion of multiple cues. Indeed it would be beneficial to consider the local spatial dependence in the observation of visual contours such as in [MacCormick98]. Indeed there are many sources of information available in an image stream such as color which could also improve robustness. Local spatial consistency constraints could also be introduced into a probabilistic framework at the lower level of the estimation process. Primitive likelihood information could also be used to guide the sampling process such as in [Stewart99a, Armstrong95]. Another alternative would be to consider a purely spatial approach and use Bayesian techniques such as in [Ricquebourg97, Ricquebourg00]. In this case a Markov formulation was used to take into consideration local constraints such as local smoothness linked to an energy function.

In perspective, however, an image has many other sources of important information which must be exploited to a maximum efficiency so as to obtain an optimal performance of tracking procedures. These different sources of information may include illumination, texture, appearance, color, or other measures which can be extracted from an image. In [Comaniciu03] a real-time tracker using color feature information was proposed based on a mean-shift probability density estimation framework. This type of method relates to a Bayesian framework and motion filters and information fusion techniques are also discussed. Many information fusion techniques (ie. [Moreno-Noguer04]) avoid modeling the relationship between different information sources when fusing different sources of information. However, without a precise analytical model describing the relationship between each type of information source and the unknown variables, the behavior of the system becomes unpredictable and estimated parameters remain more approximate and uncertain. Few methods in the literature combine multiple sources of information in a simultaneous estimation procedure. In order to achieve this it is necessary to derive interaction matrices corresponding to each types of visual information such as in [Pressigout04] or [Vacchetti04b] where planar features and texture have been considered. It would therefore be interesting to further extend this optimization function to include even more sources of information.

Finally, it can be mentioned that other types of visual sensor exist which can provide more visual information than in the case of a monocular CCD camera. For example, it could also be interesting to consider an omnidirectional camera which provides a wide angle view of the environment. Another possibility is to use multiple cameras which not only provide hidden depth information but also give information from different views so that features which are usually occluded are visible.

8.2.2 3D Knowledge

Another important factor in knowledge based tracking system is the information which is stored within the off-line CAD model. In this thesis, the *a priori* information has been intimately related to the contour information and wire-frame models have been used. As in the 2D image information case, it would also be interesting to consider more complex type of contours. This could be achieved by using super-quadrics as in [Terzopoulos91, Bardinet94, Leonardis97] or other types of 3D parametric models. As in the 2D visual feature case, it would also be interesting to take advantage of moment based planar models [Tahri05]. Furthermore, it would be interesting to explore the extension of such approached so as to describe complex 3D objects. This would require developing this theory to include full 3D moments and not only planar moments. Apart from more complex contours it would also be interesting to develop the *a priori* model further to include many sources of information such as texture or color information.

Another perspective would be to consider obtaining this 3D model in a completely automatic manner. This would require a reconstruction step which is often performed off-line due to computational costs. The reconstruction could be approached by learning or generating a higher level parametric representation of the world via techniques such as information theory and Bayesian inference. In particular, inference techniques such as the Minimum Message/Description Length principles(MML/MDL) could be used to generate a compressed 3D map of the environment. This type of method would allow the extraction of only the pertinent information by using a Bayesian decision tree which adapts to the best model of the environment. Furthermore, it would be interesting to consider hidden surface removal and Binary Space Partition Trees (BSP-Tree) within this type of framework to account for the full 3D nature of the environment. Another possibility would be to combine more closely the interaction between the *a priori* information and the unknown parameters by continually learning and updating the model. This technique is widely known as Simultaneous Localization and Mapping (SLAM) [Davison02], however, it would be interesting to approach this problem differently by considering a large amount of information already known.

8.2.3 Pose Estimation

Apart from the visual information and the 3D model, considered previously, the problem of estimating the pose can also be improved.

In this thesis a non-linear technique was employed for estimating the pose which depends on a pose initialization procedure. Various techniques have been considered including manually making the correspondence of four point features with the model [Dementhon95], a manual manipulation of the object projected on the image (using the mouse) or an automatic procedure in the case of textured objects using point matching [Lowe04]. In perspective, it would be interesting to use sampling style methods such as LMedS to initialize correspondences between the initial image and the final image at the beginning of a tracking or visual servoing task since it can be used with globally convergent pose estimation methods such as [Dementhon95, Tahri03]. This would correspond more closely to a recognition process rather than a 3D tracking context. In this case, the solutions would then be used to find initial correspondences amongst all the combinations of correspondences between the set of image features and model feature. Furthermore, this could be combined with other quantitative based techniques for obtaining these correspondences [Harris88].

Although low level feature analysis has been the object of much study throughout the past, much more work remains to fully take advantage of forward and backward propagation of information between high and low levels. The method presented in this thesis has been based on two steps, the first involving feature extraction and the second, pose estimation. In perspective, it would be interesting to consider a higher level objective function based purely on image intensity as was done in [Haag99] so that the estimation process is a pure one-step estimation process. It has also been shown in [Kollnig95], that the global CAD model is estimated directly from image intensity values avoiding the need for edge detection and threshold selection. In this way, the scale of the edges would be related directly to the size of the edge on the object. The 3D model could also be exploited further by considering the 3D appearance for different types of object edges. For example, only "step" edges have been considered, however, many different types of surfaces and edges exist in a real image including roof edges or curved edges. This is particularly important if more estimation precision is required.

Another important problem is the problem of robustness. The statistical M-estimation method used in this thesis has proven to work well in many different situations, however, it is highly dependent on a good estimate of scale. In this thesis the preferred automatic scale estimation method was the MAD due to its efficiency and relatively high breakdown point of up to 50% outliers. However, in computer vision applications this breakdown point is unsatisfactory. The ideal would be a method that is not only able to estimate parameters from a large number of outliers, but also to handle multiple instances of the same structure. Therefore it is necessary to further consider the problem of scale estimation. More recently, the mean-shift procedure originally designed for pattern recognition [Fukunaga75] has become an interesting way of analyzing the feature space so as to estimate the scale of the distribution. In recent computer vision literature, this issue had been popular and relevant work can be found in [Suter04, Wang04a, Wang04b, Comaniciu02, Comaniciu03]. The use of these technique in visual control would be of great interest. Apart from the problem of scale estimation is the problem of a non-Gaussian distributions producing sub-optimal estimates. In this case the variance is *heteroscedastic* meaning that it depends on the measurement frame of reference. Sampson proposed a first order approximation of the geometric distance. In [Weng89] Sampson's method was adapted to the estimation of the Fundamental matrix and in [Shapiro95, Torr97] outlier detection was considered for orthogonal regression. It could be interesting to pursue this idea further by designing a multi-modal M-estimator. The robustness could also be improved by modeling the environment better such as tracking objects that occlude the edges. Specular reflections could also be managed by estimating the light source and detecting this type of effect.

8.2.4 Non-rigid Motion

As has already been mentioned in Chapter 7, the proposed kinematic set approach is a fundamental problem that touches upon many domains. Velocity constraint matrices used to form the kinematic sets are well defined mathematical concepts that also have a very large application domain. The kinematic chain approach, is currently used in many domains, could be replaced by a set approach. The various application domains include robotics, computer animation, mechanics and even electrical circuits.

In robotics community velocity constraints are of fundamental importance for controlling the entire space of allowable configurations of a robot in a general manner. The use of these techniques could lead to interesting avenues for constraining a robot along a particular trajectory or even constraining the configuration space between different cameras. The kinematic set approach is also of fundamental importance in robotics applications and could be used to model robotic structures in a more symmetric manner than traditional kinematic chains. In this way many difficulties in parallel systems, such as robotic hands, could be resolved. It could also be used in visual servoing to close the control loop visually via an eye-to-hand system. If the kinematic set approach was used in computer animation for the control of human figures, computational efficiencies could be observed and it would become much easier to control a particular joint without the need of passing along a chain structure. In the mechanics literature the principals of the Lagrange d'Alembert theory have been studied in depth and it would be interesting to study these techniques in more detail and to derive a solid mathematical link with the kinematic set approach proposed here. Furthermore, it can also be noted that there are many parallels between mechanics and circuit theory and similar approaches have been applied to circuit theory. These techniques are known as a Port-Hamiltonian based systems which define the concept of a power port as known in electrical networks [Stramigioli00, Stramigioli02].

APPENDIX A

Appendix

A.1 Virtual Visual Servoing Based Camera Calibration

The basic idea of the camera calibration method presented here is to use the Virtual Visual Servoing principal presented in Chapter 3 to estimate the camera calibration parameters simultaneously with the pose. In this case simple point features that can be tracked accurately will be used. The approach presented here was given in [Marchand01, Marchand02a].

A virtual camera is defined with intrinsic parameters ξ located at a position such that the object frame is related to the camera frame by the pose $\mathbf{r}$. The position of an object feature ${}^c\mathbf{S}$ in the camera frame is defined by a perspective projection onto the image plane. The projection of this point onto the image in pixel coordinates is then given by:

$$\mathbf{s} = pr(\xi, {}^c\mathbf{S}) = pr(\xi, {}^c\mathbf{r}_o, {}^o\mathbf{S}) \tag{A.1}$$

where $pr(.)$ is the projection model according to the intrinsic parameters ξ and the pose parameters $\mathbf{r}$. $\mathbf{s}$ and $\mathbf{S}$ are vectors of 2D and 3D features respectively. The goal of calibration is to minimize the error between the observed data denoted $\mathbf{s}^*$ and the position of the same features computed by back-projection according to the current extrinsic and intrinsic parameters $\mathbf{s}$ (as defined in equation A.1). In order to minimize this function, a virtual camera is displaced (initially in ${}^{ci}\mathbf{r}_o$) along with the intrinsic camera parameters (initially ξ_i) using a visual servoing control law. When the minimization is achieved, the parameters of the virtual camera will be related to the estimated pose ${}^{cf}\mathbf{r}_o$ and the calibration parameters ξ_f.

A.1.1 Calibration

The goal is to minimize the error between a current feature position and a desired feature position, $\|\mathbf{s} - \mathbf{s}^*\|$. The error in the image $\mathbf{e}$ is defined as:

$$\mathbf{e} = \mathbf{s} - \mathbf{s}^* \tag{A.2}$$

The motion of the features in the image is related to the camera velocity $\mathbf{v}$ and the time variation of the intrinsic parameters by:

$$\dot{\mathbf{s}} = \frac{\partial \mathbf{s}}{\partial \mathbf{r}}\frac{\partial \mathbf{r}}{\partial t} + \frac{\partial \mathbf{s}}{\partial \xi}\frac{\partial \xi}{\partial t} \tag{A.3}$$

which can be rewritten as:

$$\dot{\mathbf{s}} = \mathbf{H_s}\mathbf{w} \qquad \text{with} \qquad \mathbf{w} = \begin{bmatrix} \mathbf{v} \\ \\ \dot{\xi} \end{bmatrix} \tag{A.4}$$

Matrix $\mathbf{H_s}$ is the interaction matrix or image Jacobian relating the image feature velocities to the movement of the virtual camera as well as the movement of the calibration parameters. It is given by:

$$\mathbf{H_s} = \begin{bmatrix} \frac{\partial \mathbf{s}}{\partial \mathbf{r}} & \frac{\partial \mathbf{s}}{\partial \xi} \end{bmatrix} \tag{A.5}$$

As usual an exponential decoupled decrease of the error $\mathbf{e}$ is defined as:

$$\dot{\mathbf{e}} = -\lambda \mathbf{e} \tag{A.6}$$

where λ is a proportional coefficient that tunes the decay rate, finally giving:

$$\mathbf{w} = -\lambda \mathbf{H_s}^{+}\mathbf{e} \tag{A.7}$$

A.1.2 Multi-images calibration

In this calibration process only a planar calibration pattern is used, therefore requiring multiple images with different depth disparities to be used so as to calibrate the camera with uniform uncertainty in the estimations. This is due to the fact that the parameter uncertainty in the direction perpendicular to the plane is much larger than the uncertainty in the plane (see the results presented in [Marchand01]).

The calibration approach is therefore to compute the set of calibration parameters which remain constant across multiple images. The aim is to compute these parameters simultaneously using all images. In this case s virtual cameras are defined and their corresponding visual features are related to the variation of p intrinsic parameters. Thus there are $6s+p$ unknown variables since each virtual camera pose is of dimension 6. Furthermore, there are $\sum_{i=1}^{s} s_i$ equations (where s_i is the number of features observed in the i^{th} image).

If $\mathbf{p}^i$ is the set of features extracted from i^{th} image, the interaction matrix used in the calibration process is then given by the relation:

$$\begin{bmatrix} \dot{\mathbf{s}}_1 \\ \dot{\mathbf{s}}_2 \\ \vdots \\ \dot{\mathbf{s}}_s \end{bmatrix} = \mathbf{H} \begin{bmatrix} \mathbf{v}_1 \\ \mathbf{v}_2 \\ \vdots \\ \mathbf{v}_s \\ \\ \dot{\xi} \end{bmatrix} \tag{A.8}$$

with

$$\mathbf{H} = \begin{bmatrix} \frac{\partial \mathbf{s_1}}{\partial \mathbf{r_1}} & 0 & \dots & 0 & \frac{\partial \mathbf{s_1}}{\partial \xi} \\ 0 & \frac{\partial \mathbf{s_2}}{\partial \mathbf{r_2}} & \dots & 0 & \frac{\partial \mathbf{s_2}}{\partial \xi} \\ \vdots & \vdots & & & \vdots \\ 0 & \dots & 0 & \frac{\partial \mathbf{s_s}}{\partial \mathbf{r_s}} & \frac{\partial \mathbf{s_s}}{\partial \xi} \end{bmatrix} \tag{A.9}$$

Minimization is handled using the same methodology:

$$\begin{bmatrix} \mathbf{v}_1 \\ \mathbf{v}_2 \\ \vdots \\ \mathbf{v}_s \\ \dot{\xi} \end{bmatrix} = -\lambda \mathbf{H}^+ \begin{bmatrix} \mathbf{s}_1 - \mathbf{s}_1^* \\ \mathbf{s}_2 - \mathbf{s}_2^* \\ \vdots \\ \mathbf{s}_s - \mathbf{s}_s^* \end{bmatrix} \tag{A.10}$$

A.1.3 Calibration from points

The calibration technique considered here will use simple point features on a planar calibration pattern. These point features will have known object coordinates.

Camera model

Consider the coordinates of a point in the camera frame $\mathbf{P} = (X, Y, Z)$. The coordinates of the perspective projection of this point onto the image plane, in metric units, is given by $\mathbf{p} = (x, y)$ with:

$$\begin{cases} x = \frac{X}{Z}, \\ y = \frac{Y}{Z}, \end{cases} \tag{A.11}$$

From equations (3.8) and (3.12) in Chapter (3) the pixel coordinates (u, v) are related to the coordinates (x, y) by:

$$\begin{cases} u = u_0 + s_x x + k_1 r^2 (u - u_0) \\ v = v_0 + s_y y + k_1 r^2 (v - v_0) \end{cases} \tag{A.12}$$

Furthermore, the term $k_2 r^4$ has been removed so that only the radial distortion is used. This has been shown as usually sufficient for accurate estimation (see [Marchand01]).

The five parameters to be estimated are thus $\xi = \{s_x, s_y, u_0, v_0, k_1\}$.

Deriving the interaction matrix

It remains to compute the interaction matrix $\mathbf{H_p}$ that links the motion $\dot{\mathbf{p}} = (\dot{u}, \dot{v})$ of a point $\mathbf{p} = (u, v)$ in the image to $(\mathbf{v}, \dot{\xi})$. For one point, this Jacobian is given by:

$$\mathbf{H_p} = \begin{bmatrix} \frac{\partial \mathbf{p}}{\partial \mathbf{r}} & \frac{\partial \mathbf{p}}{\partial \xi} \end{bmatrix} \tag{A.13}$$

where $\frac{\partial \mathbf{p}}{\partial \mathbf{r}}$ is a 2×6 matrix and $\frac{\partial \mathbf{p}}{\partial \xi}$ is a 2×5 matrix. Considering a calibration with n points, the full image Jacobian is given by the $2n \times 11$ matrix:

$$\mathbf{H} = (\mathbf{H_{p_1}}, \quad \mathbf{H_{p_2}}, \quad \ldots, \quad \mathbf{H_{p_n}}) \tag{A.14}$$

The interaction matrix $\frac{\partial \mathbf{p}}{\partial \mathbf{r}}$ that relates the motion of a point in the image to the camera motion has already been given in 5.13 leading to the image Jacobian with the intrinsic parameters as:

$$\frac{\partial \mathbf{p}}{\partial \mathbf{r}} = \begin{bmatrix} s_x & 0 \\ 0 & s_y \end{bmatrix} \mathbf{L_p} \tag{A.15}$$

Furthermore, from (A.12), differentiating u and v for ξ leads very easily to:

$$\frac{\partial \mathbf{p}}{\partial \xi} = \begin{bmatrix} x & 0 & 1 - k_1 \left(r^2 + 2\tilde{u}^2\right) & -2\, k_1\, \tilde{u}\tilde{v} & \tilde{u}\, r^2 \\ 0 & y & -2\, k_1\, \tilde{u}\tilde{v} & 1 - k_1\, (r^2 + 2\tilde{v}^2) & \tilde{v}\, r^2 \end{bmatrix}, \tag{A.16}$$

where $\tilde{u} = (u - u_0)$, $\tilde{v} = (v - v_0)$ and $r^2 = \tilde{u}^2 + \tilde{v}^2$.

This completes the relation between the motion of a point in the image and both the camera motion and the variation of the intrinsic camera parameters.

Furthermore, if radial distortion is not considered, $\frac{\partial \mathbf{p}}{\partial \xi}$ becomes simply:

$$\frac{\partial \mathbf{p}}{\partial \xi} = \begin{bmatrix} x & 0 & 1 & 0 \\ 0 & y & 0 & 1 \end{bmatrix} \tag{A.17}$$

with $\xi = (s_x, s_y, u_0, v_0)$

Thus, these parameters are easily estimated using the non-linear iterative approach already given in Chapter 3.

A.1.4 Experimental results for multi-image calibration

The proposed algorithm has been tested on a 2D rectangular rig composed of 24 points.(Figure A.1)

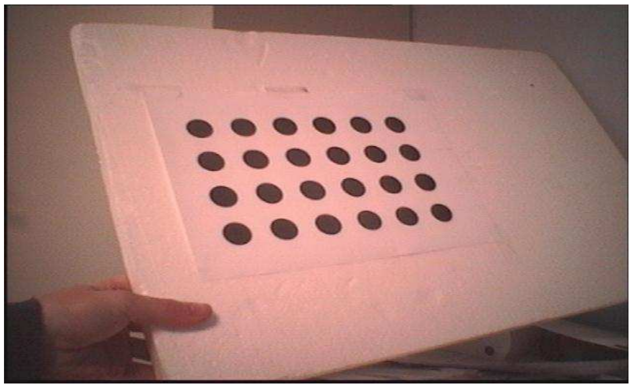


Figure A.1: image of 2D calibration rig with 24 points

Multi-image calibration

For each image, the extrinsic parameters are initialized with the parameters computed by a linear pose calculation and intrinsic parameters are chosen roughly. In a first step, calibration is computed without considering any distortion. Then the results are used for initialization of the calibration with distortion.

Figure (A.3) shows the evolution of the intrinsic parameters over time during the minimization process. The minimization algorithm is fast and converges after about 50 iterations with $\lambda = 0.1$. This method is advantageous because it can be used to calibrate easily any kind of camera using a simple 2D pattern and does not require precision placement of the different planes as in a 3D rig. To obtain good results, 10 images were used since the greater the number of images, the better the calibration precision.

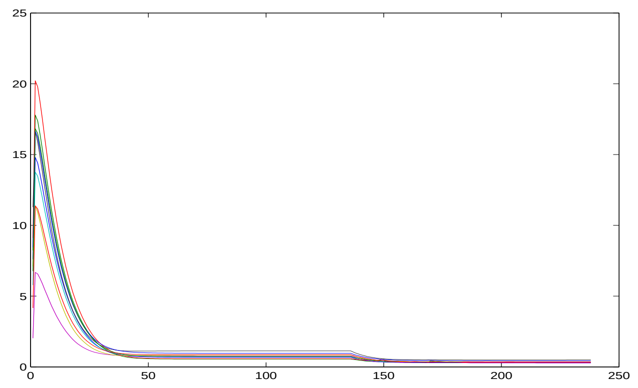


Figure A.2: Evolution of the mean error in pixel for 10 images (radial distortion is introduced after the convergence of the first calibration

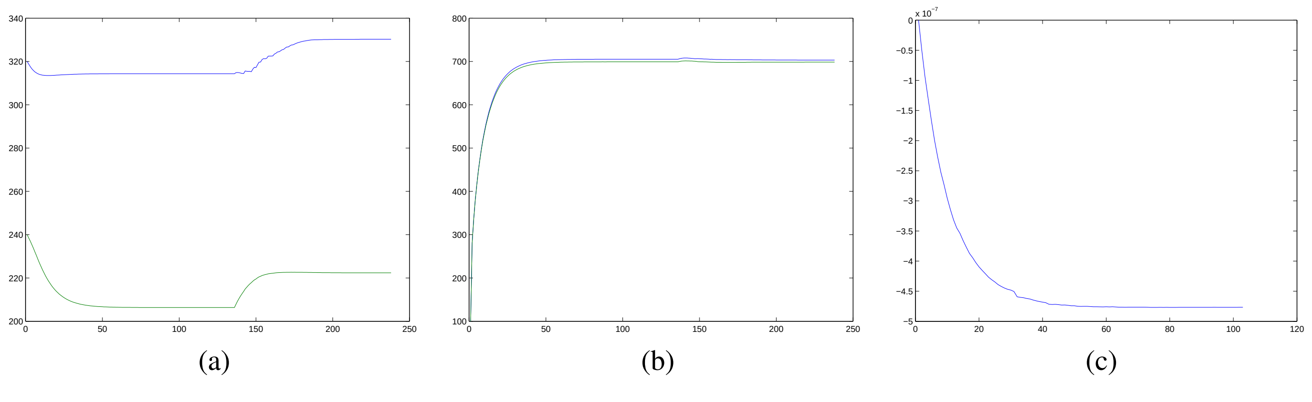


Figure A.3: Evolution of the intrinsic parameters (a)u_0 *and* v_0*, (b)*s_x *and* s_y*, (c)* k_1

Although point features were considered here, any kind of feature can be considered within this control law as soon as it is possible to compute the image Jacobian $\mathbf{H_p}$ [Chaumette90, Espiau92]. Furthermore, $\frac{\partial \mathbf{p}}{\partial \xi}$ is simple to compute. In this way many different types of features can be used simultaneously to calibrate the camera.

APPENDIX B

Appendix

B.1 Hidden Surface Removal

A classical problem in computer graphics is to determine which parts of a 3D object are visible to a 2D viewing plane. Indeed, this has many repercussions on the different computations required within an algorithm. Of course it is computationally inefficient to render the surfaces of a scene which are not visible. The 3D tracking case presented in this thesis differs slightly with computer graphics techniques and it is necessary to determine which technique amongst the various techniques available, best applies. In particular, the tracking algorithm is based on the contours of the object and only the contours need to be taken into consideration. In order to improve the robustness of the algorithm, it is essential to only consider those visual features which are visible to the camera. As will be shown this is achieved by a Hidden Surface Removal Technique.

Due to the fact that the local tracking algorithm is based on the forward-projection of model contours (lines, ellipses, cylinders, etc ...), the result of removing hidden surfaces is equivalent to the display of the wire-frame model in computer graphics using techniques such as ray tracing. To carry out low level tracking, the projected contour of an object is sampled into elements. From these elements, corresponding points are detected using a convolution operator which is applied to the normal of the contour. In order to determine if an element should be taken into account, it is sufficient to determine if it is visible to the camera. To achieve this different methods exist which will be mentioned briefly in the following.

B.1.1 Z-buffer

In this method the visual display is associated with a frame of memory designated to store the intensity of the pixels as well as their depth. The primitives of an object are handled in any order without any prior sorting. During the drawing phase of a primitive, if the depth of a pixel is greater than that which is memorized (calculated for the previous frame), then the pixel is not treated. In the opposite case, the pixel is memorized and the associated intensity is saved. Even if programming a Z-buffer is complicated, most video cards perform this function in their hardware. It is therefore sufficient to access the graphics card to implement this approach. However it would seem that the manufacturers are limiting easy access to this function since for improved bandwidth it is necessary to compress the data making them difficult to access. Furthermore, this approach is often hardware dependent.

B.1.2 BSP Tree

The *BSP (Binary Space Partition) Tree* technique is used for navigating in 3D scenes. It provides a static definition of the visibility between primitives for all primitives in a 3D scene. It consist of building a visibility tree of which each node corresponds to a plane which contains a geometric primitive. From this plane it is possible to class all other primitives into two groups corresponding to those primitives which lie each side of the plane. This classification structured into a tree allows to define the order to treat the visible features according to where the camera viewpoint is located within the tree. This method also requires a significant effort to implement since it is necessary to define mathematically the intersection of a plane with all the different types of features along with a data structure which stores all sub slices of primitives. Furthermore, this method only applies to static scenes and may require adaptation for non-rigid scenes.

B.1.3 Back-face Culling

This method is very simple to implement and is not very computationally expensive. It consists of testing the depth of the normal corresponding to each face of an object. The normal vector $\mathbf{n_o}$ is calculated off-line when the CAD model is first loaded, for each face, in the object reference frame. Thus from two vectors $\mathbf{u} = (u_x, u_y, u_z)$ and $\mathbf{v} = (v_x, v_y, v_z)$ defined by two difference faces, the normal vector is obtained as:

$$\mathbf{n_o} = \mathbf{u} \times \mathbf{v},$$

where $\times$ is the cross product operator.

Next, for each pose of the camera, it is necessary to determine this vector as seen from the camera reference frame. Using the rotation matrix ${}^c\mathbf{R}_o$ between the camera and the object the normal vector as seen from the camera is:

$$\mathbf{n_c} = {}^c\mathbf{R}_o \mathbf{n_o}$$

In Figure (B.1), the vectors in red belong to the visible faces. The visible features are thus those that belong to these faces. The corresponding face is visible if $n_z < 0$.

Even if this approach is computationally efficient, it does not function for objects made up of multiple components. As shown in Figure(B.2), certain contours are displayed using this method even when they should be hidden. In particular certain features of the horizontal block that are behind the vertical block should not be visible. They are omitted since this technique only tests the visibility of an entire face and is not able to handle partial faces. Thus it is not sufficient to test just the orientation of the faces, but it is also necessary to test the visibility of each sample of the contour individually. In this case it is necessary to use the ray tracing method.

B.1.4 Ray Tracing

This method consists in tracing a ray from the optical center of the camera to each point-sample made along the contour (see Figure B.3(a)). If the ray does not intersect any faces then the sample is visible. If this ray intersects a face then the sample is not visible.

In practice, however, a face is represented by an infinite plane with no boundaries. In this case it is necessary to determine if a ray intersecting a plane is *inside* or *outside* the corresponding face. In order to test this case, another ray is traced from the point of intersection with the plane in a direction tangent to the plane. In this case it can be shown that the parity of the number of intersections of this secondary ray indicates whether the sample point is inside or outside the face. For example, consider Figure B.3(b). If the

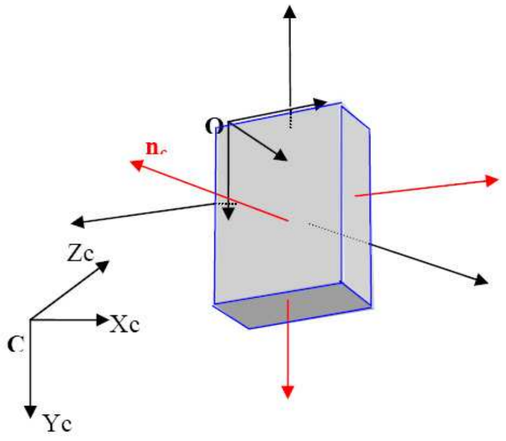


Figure B.1: Visible faces of a parallelepiped with their normals drawn in red. The corresponding face is visible if $n_z < 0$.

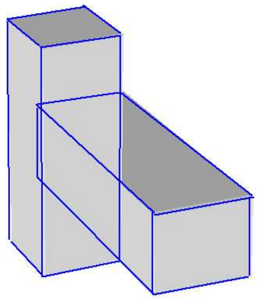


Figure B.2: Limitations of the Back-Face Culling Method. Partial features of the horizontal block that are behind the vertical block should not be visible.

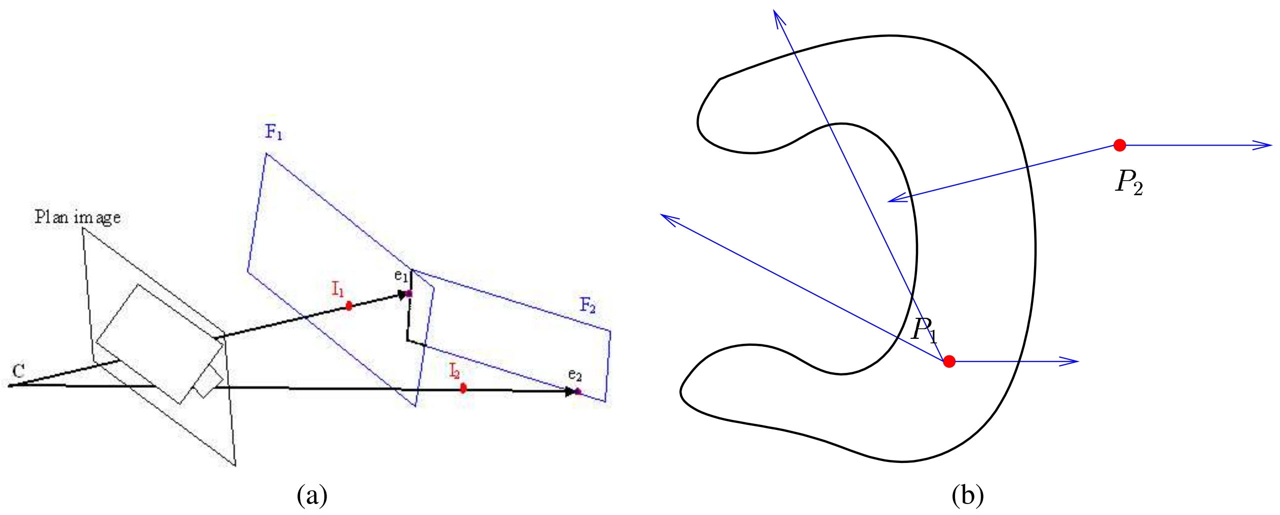


Figure B.3: Ray tracing for visibility testing. (a) Intersections I_1 and I_2 of rays with planes of the object where the faces F_1 and F_2 are represented by infinite planes. The rays run between the optical center C of the camera and points e_1 and e_2 that have been sampled along the object's contour, (b) Testing the intersecting point to determine if it is inside or outside a face of the object. P_1 is inside and any ray running tangent to the face has an odd number of intersections. P_2 is outside the face and only has an even number of intersections (note that the intersection at tangents must be handled explicitly).

number of intersections is odd then the sample point is inside the face. If the number of sample points is even it is outside the face. This method has been used since it takes into account objects that are convex.

In the case of a cylinder, the plane passing through the center of the cylinder and parallel to the image plane is considered. If the intersection with this plan occurs between the two lines of the cylinder then the sample is not visible. The end limits of a cylinder are defined by two circles.

B.1.5 Hybrid Solution

In order to have the most efficient method, the chosen technique was to apply successively the Back-face Culling and the Ray tracing methods. In this way the number of faces for which rays are traced is kept to a minimum. The computation time for computing hidden surface removal using this technique was in the order of 2 to 3 ms for the object shown in Figure B.2 comprising around 100 point samples in total.

Bibliography

[Aggarwal94]	J. Aggarwal, Q. Cai, W. Liao and B. Sabata. Articulated and elastic non-rigid motion: A review. In *IEEE Computer Society Workshop on Motion of Non-Rigid and Articulated Objects*, pages 16–22. Austin, TX, 1994.
[Aggarwal98]	J. Aggarwal, Q. Cai, W. Liao and B. Sabata. Nonrigid Motion Analysis: Articulated and Elastic Motion. *Computer Vision and Image Understanding*, vol. 70(2), 142–156, May 1998.
[Ait-Aider02]	O. Ait-Aider, P. Hoppenot and E. Colle. Adaptation of Lowe's camera pose recovery algorithm to mobile robot self-localisation. *Robotica*, 2002.
[Allen93]	P. Allen, A. Timcenko, B. Yoshimi and P. Michelman. Automated Tracking and Grasping of a Moving Object with a Robotic Hand-Eye System. *IEEE Transactions on Robotics and Automation*, vol. 9(2), 152–165, April 1993.
[Aloimonos87]	Y. Aloimonos, I. Weiss and A. Bandopadhay. Active Vision. *International Journal of Computer Vision*, vol. 1(4), 333–356, January 1987.
[Aloimonos89]	J. Aloimonos and D. Shulman. *Integration of Visual Modules.* Academic, New York, 1989.
[Altmann86]	S. L. Altmann. *Rotations, quaternions, and double groups*. Clarendon, Oxford, 1986.
[Ames52]	A. Ames. *The Ames Demonstrations in Perception.* Hafner Publishing, New York, 1952.
[Amsaleg04]	L. Amsaleg, P. Gros and S.-A. Berrani. Robust Object Recognition in Images and the Related Database Problems. *Special issue of the Journal of Multimedia Tools and Applications*, vol. 23(3), 221–235, 2004.
[Ansar02]	A. Ansar and D. K. Linear Pose Estimation from Points or Lines. In *European Conference on Computer Vision* (edited by A. Heyden et. al.), pages 282–296. Springer-Verlag, Berlin Heidelberg, 2002.

[Arman93] F. Arman and A. J. Model-based object recognition in dense range images. *Computing Surveys*, vol. 25, 5–43, 1993.

[Armstrong95] M. Armstrong and A. Zisserman. Robust Object Tracking. In *Second Asian Conference on Computer Vision*, pages 58–62. Singapore, December 5-8 1995.

[Aron04] M. Aron, G. Simon and M. Berger. Handling uncertain sensor data in vision-based camera tracking. In *IEEE and ACM International Symposium on Mixed and Augmented Reality*, pages 58–67. Arlington, VA, USA, November 2-5 2004.

[Azuma97] R. Azuma. A Survey of Augmented Reality. *Presence: Teleoperators and Virtual Environments*, vol. 6(4), 355–385, August 1997.

[Azuma01] R. Azuma, Y. Baillot, R. Behringer, S. Feiner, S. Julier and B. MacIntyre. Recent Advances in Augmented Reality. *IEEE Computer Graphics and Application*, vol. 21(6), 34–47, November 2001.

[Babaud86] J. Babaud, A. Witkin, M. Baudin and R. O. Duda. Uniqueness of the Gaussian kernel for scale-space filtering. *IEEE Transactions Pattern Analysis and Machine Intelligence*, vol. 8, 26–33, January 1986.

[Badler91] N. Badler, B. Weber, J. Kalita and J. Ekasov. *Making them move: mechanics, control and animation of articulated figures*. Morgan Kaufmann, 1991.

[Bajura92] M. Bajura, H. Fuchs and R. Ohbuchi. Merging Virtual Reality with the Real World: Seeing Ultrasound Imagery Within the Patient. In *International Conference on Computer Graphics and Interactive Techniques, SIGGRAPH*, vol. 26 of *Computer Graphics*, pages 203– 210. Chicago, IL, 26-31 July 1992.

[Bajura95] M. Bajura and U. Neumann. Dynamic Registration Correction in Video-Based Augmented Reality Systems. *IEEE Computer Graphics and Applications*, vol. 15(5), 52– 60, September 1995.

[Ball00] R.-S. Ball. *A treatise on the theory of screws.* Cambridge University Press, 1900.

[Bar-Shalom93] Y. Bar-Shalom and X.-R. Li. *Estimation and Tracking, Principles, Techniques, and Software*. Artech House, Boston, 1993.

[Bardinet94] E. Bardinet, L. D. Cohen and N. Ayache. Fitting 3-D Data Using Superquadrics and Free-form Deformations. In *International Conference on Pattern Recognition*, vol. 1, pages 79–83. Jerusalem, Israel, October 1994.

[Barrow81] H. Barrow and J. Tenenbaum. Interpreting line drawings as three-dimensional surfaces. In *Artificial Intelligence Center*, vol. 17, pages 75–116. Menlo Park, CA, 1981.

[Bascle94] B. Bascle, P. Bouthemy, N. Deriche and F. Meyer. Tracking Complex Primitives in an image sequence. In *International Conference on Pattern Recognition*, pages 426–431. Jerusalem, October 1994.

[Basu02] M. Basu. Gaussian-based edge-detection methods-a survey. *Systems, Man and Cybernetics, Part C, IEEE Transactions on*, vol. 32(3), 252 – 260, 629 – 639, August 2002.

[Beier03] D. Beier, R. Billert, B. Bruderlin, D. Stichling and B. Kleinjohann. Marker-less vision based tracking for mobile augmented reality. In *International Symposium on Mixed and Augmented Reality*, pages 258–259. Tokyo, Japan, October 8 - 10 2003.

[Benhimane04] S. Benhimane and E. Malis. Real-time image-based tracking of planes using efficient second-order minimization. In *IEEE International Conference on Intelligent Robots Systems*. Sendai, Japan, 28 September - 2 October 2004.

[Berchtold98] S. Berchtold, C. Bohm and H. Kriegal. The pyramid-technique: towards breaking the curse of dimensionality. In *International conference on Management of data, SIGMOD*, pages 142–153. ACM Press, Seattle, WA, 2-4 June 1998.

[Berger96] M. Berger, C. Chevrier and G. Simon. Compositing Computer and Video Image Sequences: Robust Algorithms for the Reconstruction of the Camera Parameters. In *Eurographics'96*, vol. 15, pages 23–32. Poitiers, France, August 1996.

[Berger98] M. Berger and S. G. Robust Image Composition Algorithms for Augmented Reality. In *Asian Conference on Computer Vision*, vol. 1352 of *Lecture notes in computer science*, pages 360–367. Hong Kong, China, January 1998.

[Berger99] M. Berger, B. Wrobel-Dautcourt, S. Petitjean and G. Simon. Mixing Synthetic and Video Images of an Outdoor Urban Environment. *Machine Vision and Applications*, vol. 11(3), 145–159, 1999.

[Bergholm87] F. Bergholm. Edge focusing. *IEEE Transactions Pattern Analysis and Machine Intelligence*, vol. 9, 726–741, June 1987.

[Berrani04] S.-A. Berrani. *Recherche approximative de plus proches voisins avec contrôle probabiliste de la précision; application la recherche d'images par le contenu.* Ph.D. thesis, Université de Rennes 1, 2004.

[Besl88] P. J. Besl and R. C. Jain. Segmentation through variable-order surface fitting. *IEEE Transaction on Pattern Analysis and Machine Intelligence*, vol. 10(2), 167–192, March 1988.

[Billinghurst01] M. Billinghurst, H. Kato and I. Poupyrev. The Magic Book: Moving Seamlessly between Reality and Virtuality. *IEEE Computer Graphics and Applications*, vol. 21(3), 6–8, May 2001.

[Bimber02] O. Bimber and B. Frohlich. Occlusion shadows: using projected light to generate realistic occlusion effects for view-dependent optical see-through displays. In *International Symposium on Mixed and Augmented Reality*, page 186. Darmstadt, Germany, October 1st 2002.

[Blake93] A. Blake, R. Curwen and A. Zisserman. A framework for spatiotemporal control in the tracking of visual contours. *International Journal of Computer Vision*, vol. 11(2), 127–145, 1993.

[Blake98] A. Blake and M. Isard. *Active Contours*. Springer Verlag, 1998.

[Bohm01] C. Bohm, S. Berchtold and D. Keim. Searching in high-dimensional spaces: Index structures for improving the performance of multimedia databases. *ACM Computing Surveys*, vol. 33(3), 322–373, September 2001.

[Bookstein79] F. Bookstein. Fitting conic sections to scattered data. *Computer Vision Graphics and Image Processing*, vol. 9, 56 71, 1979.

[Borges96] A. Borges and L. Díbio. 3D Recognition by Parts: A Complete Solution using Parameterized Volumetric Models. In *SIBGRAPI'96*, pages 111–118. 1996.

[Bosse03] M. Bosse, R. Rikoski, J. Leonard and S. Teller. Omni-directional Structure from Motion Using Vanishing Points and 3D Lines. *The Visual Computer Journal*, vol. 19(6), 417–430, October 2003.

[Bosse04] M. Bosse, P. Newman, J. Leonard and S. Teller. SLAM in Large-scale Cyclic Environments using the Atlas Framework. *International Journal of Robotics Research*, vol. 23(12), 1113–1139, December 2004.

[Boukir93] S. Boukir. *Reconstruction 3D d'un environnement statique par vision active*. Université de Rennes 1, IRISA, Rennes, 1993.

[Boukir98] S. Boukir, P. Bouthemy, F. Chaumette and D. Juvin. A local method for contour matching and its parallel implementation. *Machine Vision and Application*, vol. 10(5), 321–330, April 1998.

[Bouthemy89] P. Bouthemy. A Maximum Likelihood Framework for Determining Moving Edges. *IEEE Transactions on Pattern Analysis and Machine intelligence*, vol. 11(5), 499–511, May 1989.

[Boyer97] E. Boyer and M. Berger. 3D Surface Reconstruction Using Occluding Contours. *International Journal of Computer Vision (IJCV)*, vol. 22(3), 219–233, May 1997.

[Bracewell65] R. Bracewell. *The Fourier transform and its applications*. New York: MacGraw-Hill, 1965.

[Braud97] P. Braud, M. Dhome, J. Lapresté and B. Peuchot. Reconnaissance, localisation et suivi d'objets polyhédriques par vision multi-oculaire. *Technique et Science Informatiques*, vol. 16(1), 9–38, January 1997.

[Brautigam98] C. Brautigam, J. Eklundh and H. Christense. A model-free voting approach for integrating multiple cues. In *ECCV* (edited by H. Burkhardt and U. Neumann), pages 734–750. Springer-Verlag Berlin Heidelberg, Frieburg, Germany, June 2-6 1998.

[Bray90] A. Bray. Tracking Object Using Image Disparities. *Image and Vision Computing*, vol. 8(1), 4–9, 1990.

[Bregler98] C. Bregler and J. Malik. Tracking people with twists and exponential maps. In *IEEE Computer Society Conference on Computer Vision and Pattern Recognition*, pages 8–15. Santa Barbara, CA, June 1998. TY - CONF.

[Bregler04] C. Bregler, J. Malik and K. Pullen. Twist Based Acquisition and Tracking of Animal and Human Kinematics. *International Journal of Computer Vision*, vol. 56(3), 179–194, February-March 2004. TY - JOUR.

[Brockett84] R. W. Brockett. Robotic Manipulators and the Product of Exponentials Formula. In *Lecture Notes in Control and Information Sciences*, pages 120–127. Berlin: Springer-Verlag, 1984.

[Brooks81] R. Brooks. Symbolic Reasoning Among 3-D Models and 2-D Images. *Artificial Intelligence*, vol. 17(2), 85–348, 1981.

[Brown71] D. Brown. Close-Range Camera Calibration. *Photogrammetric Engineering*, vol. 4(2), 127–140, March 1971.

[Bulthoff87] H. Bulthoff and H. Mallot. Interaction of different modules in depth perception. In *First International Conference on Computer Vision*, pages 295–305. London, United Kingdom, June 8-11 1987.

[Caillette04] F. Caillette and T. Howard. Real-Time Markerless Human Body Tracking Using Colored Voxels and 3-D Blobs. In *International Symposium on Mixed and Augmented Reality*, pages 266–267. Arlington, VA, USA, 2-5 November 2004.

[Cakmakci04] O. Cakmakci, H. Yonggang and J. P. Rolland. A Compact Optical See-Through Head-Worn Display with Occlusion Support. In *International Symposium on Mixed and Augmented Reality*, page 16. Arlington, VA, USA, 2-5 November 2004.

[Canny83] J. Canny. Finding Edges and Lines in Images. Tech. Rep. 889502, Massachusetts Institute of Technology, 1983.

[Canny86] J. Canny. A computational approach to edge detection. *IEEE Transactions on Pattern Analysis and Machine intelligence*, vol. 8(6), 679–698, November 1986.

[Caprile90] B. Caprile and V. Torre. Using vanishing points for camera calibration. *International Journal of computer Vision*, vol. 4(2), 127–140, March 1990.

[Casanova04] E. Casanova, H. Dejnabadi, B. Jolles, K. Aminian and P. Fua. 3–D Motion Analysis and Synthesis from Inertial Gyroscopes and Image Data. In *European Society for Movement Analysis in Adults and Children Conference*. Warsaw, Poland, November 2004.

[Caudell92] T. Caudell and D. Mizell. Augmented Reality: An Application of Heads-Up Display Technology to Manual Manufacturing Processes. In *Twenty-Fifth Hawaii International Conference on System Sciences*, vol. 2, pages 659–669. Kauai, Hawaii, January 7-10 1992.

[Caudell94] T. Caudell. Introduction to Augmented Reality. In *Telemanipulator and Telepresence Technologie SPIE*, vol. 2351, pages 272– 281. Boston, MA, 31 October – 4 November 1994.

[Cervera03]	E. Cervera, A. P. Del Pobil, F. Berry and P. Martinet. Improving Image-Based Visual Servoing with Three-Dimensional Features. *International Journal of Robotics Research*, vol. 10-11(22), 821–839, October-November 2003.
[Chaumette89]	F. Chaumette and P. Rives. Modélisation et calibration d'une caméra. In *7ème congrès AFCET Reconnaissance des formes et intelligence artificielle, RFIA'89*, vol. 1, pages 527–536. Paris, December 1989.
[Chaumette90]	F. Chaumette. *La relation vision-commande: théorie et application á des tâches robotiques*. Phd thesis, Université de Rennes I, IRISA, 1990. June.
[Chaumette93]	F. Chaumette, P. Rives and B. Espiau. Classification and realization of the different vision-based tasks. In *Visual Servoing* (edited by K. Hashimoto), vol. 7, pages 199–228. World Scientific Series in Robotics and Automated Systems, Singapour, 1993.
[Chaumette96]	F. Chaumette, S. Boukir, P. Bouthemy and D. Juvin. Structure from controlled motion. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, vol. 18(5), 492–504, May 1996.
[Chaumette98]	F. Chaumette. Potential problems of stability and convergence in image-based and position-based visual servoing. In *The Confluence of Vision and Control* (edited by D. Kriegman, G. Hager and A. Morse), pages 66–78. LNCIS Series, No 237, Springer-Verlag, 1998.
[Chaumette00]	F. Chaumette and E. Marchand. A New Redundancy-based Iterative Scheme for Avoiding Joint Limits: Application to visual servoing. In *IEEE International Conference on Robotics and Automation*, vol. 2, pages 1720–1725. San Francisco, CA, April 2000.
[Chaumette02a]	F. Chaumette. Asservissement visuel. In *La commande des robots manipulateurs* (edited by W. Khalil), pages 105–150. Hermés, Rennes, 2002.
[Chaumette02b]	F. Chaumette. A first step toward visual servoing using image moments. In *IEEE/RSJ International Conference on Intelligent Robots and Systems, IROS'02*, vol. 1, pages 378–383. Lausanne, Switzerland, October 2002.
[Chaumette04]	F. Chaumette. Image moments: a general and useful set of features for visual servoing. *IEEE Transactions on Robotics*, vol. 20, 713–723, August 2004.
[Chinthammit03]	W. Chinthammit, E. Seibel and T. Furness. A Shared-Aperture Tracking Display for Augmented Reality. In *Presence: Teleoperators and Virtual Environments*, vol. 1, pages 1–18. 2003.
[Chong99]	K. Chong and L. Kleeman. Feature-based mapping in real, large scale environments using an ultrasonic array. *International Journal of Robotics Research*, vol. 18(2), 3–19, January 1999.
[Cipolla90]	R. Cipolla and A. Blake. The Dynamic Analysis of Apparent Contours. In *3rd International Conference on Computer Vision*, pages 616–623. Osaka, Japan, December 1990.

[Cipolla97] R. Cipolla and N. Hollinghurst. Visually guided grasping in unstructured enviroments. *Robotics and Autonomous Systems*, vol. 19, 337–346, 1997.

[Clark90] J. Clark and A. Yuille. *Data Fusion for Sensory Information Procession Systems.* Kluwer Academic, Norwell, MA, 1990.

[Collewet02] C. Collewet and F. Chaumette. Positioning a camera with respect to planar objects of unknown shape by coupling 2d visual servoing and 3d estimations. *IEEE Transactions on Robotics and Automation*, vol. 18(3), 322–333, June 2002.

[Comaniciu02] D. Comaniciu and P. Meer. Mean shift: a robust approach toward feature space analysis. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, vol. 24(5), 603–619, May 2002. TY - JOUR.

[Comaniciu03] D. Comaniciu, V. Ramesh and P. Meer. Kernel-based object tracking. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, vol. 25(5), 564–577, May 2003.

[Comport03a] A.-I. Comport, E. Marchand and F. Chaumette. A real-time tracker for markerless augmented reality. In *ACM/IEEE International Symposium on Mixed and Augmented Reality, ISMAR'03*, pages 36–45. Tokyo, Japan, 7-10 October 2003.

[Comport03b] A.-I. Comport, M. Pressigout, E. Marchand and F. Chaumette. A Visual Servoing Control Law that is Robust to Image Outliers. In *IEEE International Conference on Intelligent Robots and Systems, IROS'03*, vol. 1, pages 492–497. Las Vegas, Nevada, October 2003.

[Comport04a] A.-I. Comport, E. Marchand and F. Chaumette. Complex articulated object tracking. In *International Workshop on articulated motion and deformable objects, AMDO'04, LNCS* (edited by J. Perales and B. Draper), vol. 3179, pages 189–201. Palma de Mallorca, Spain, September 2004.

[Comport04b] A.-I. Comport, E. Marchand and F. Chaumette. Object-based visual 3D tracking of articulated objects via kinematic sets. In *IEEE Workshop on Articulated and Non-Rigid Motion*. Washington, DC, June 2004.

[Comport04c] A.-I. Comport, E. Marchand and F. Chaumette. Robust model-based tracking for robot vision. In *IEEE/RSJ International Conference on Intelligent Robots and Systems, IROS'04*, vol. 1, pages 692–697. Sendai, Japan, September 2004.

[Corpetti02] T. Corpetti, E. Mémin and P. Pérez. Dense Motion Analysis in Fluid Imagery. In *European Conference on Computer Vision, ECCV*, pages 676–691. Copenhagen, Denmark, May 28-31 2002.

[Csurka99] G. Csurka and P. Bouthemy. Direct Identification of Moving Objects and Background from 2D Motion Models. In *In IEEE International Conference on Computer Vision ICCV'99*, vol. 1, pages 566–571. Corfou, GrÃªce., September 1999.

[Das Peddada96] S. Das Peddada and R. McDevitt. Least average residual algorithm (LARA) for tracking the motion of artic sea ice. *IEEE Transactions on Geosciences and Remote Sensing*, vol. 34(4), 915–926, 1996.

[Daucher93] N. Daucher, M. Dhome, J. Lapresté and G. Rives. Modelled Object Pose Estimation and Tracking by Monocular Vision. In *British Machine Vision Conference, BMVC'93*, pages 249–258. Guildford, UK, September 1993.

[David02] P. David, D. DeMenthon, R. Duraiswami and S. Hanan. SoftPOSIT: Simultaneous Pose and Correspondence Determination. In *Seventh European Conference on Computer Vision*, vol. 2352, pages 698–714. Copenhagen, Denmark, May 2002.

[Davison02] A. J. Davison and D. W. Murray. Simultaneous Localisation and Map-Building using Active Vision. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, July 2002.

[Davison03] A. Davison, W. Mayol and D. Murray. Real-Time Localisation and Mapping with Wearable Active Vision. In *ACM/IEEE International Symposium. on Mixed and Augmented Reality, ISMAR'03*, pages 18–27. Tokyo, Japan, October 2003.

[De Ma93] S. De Ma. Conics-based stereo, motion estimation and pose determination. *International Journal of Computer Vision*, vol. 10(1), 7–25, February 1993.

[Deguchi96] K. Deguchi and T. Noguchi. Visual Servoing Using Eigenspace Method and Dynamic Calculation of Interaction Matrices. In *IEEE International Conference on Pattern Recognition*, vol. 1, page 780. Vienna, Austria, August 25-30 1996.

[Delamarre99] Q. Delamarre and O. Faugeras. 3D Articulated Models and Multi-View Tracking with Silhouettes. In *International Conference on Computer Vision*, vol. 2, page 716. Corfu, Greece, September 20-25 1999.

[Dementhon95] D. Dementhon and L. Davis. Model-Based Object Pose in 25 Lines of Codes. *International Journal of Computer Vision*, vol. 15, 123–141, June 1995.

[Deming43] W. Deming. *Statistical Adjustment of Data*. Wiley, J., New York, 1943.

[Denavit55] J. Denavit and R. S. Hartenberg. A Kinematic Notation for Lower-Pair Mechanisms Based on Matrices. *Journal of Applied Mechanics*, vol. 22, 215–221, June 1955.

[Deng03] L. Deng. *Comparison of image-based and position-based visual servoing methods and improvements*. Phd thesis, University of Waterloo, 2003.

[Deriche87] R. Deriche. Using Canny's criteria to derive a recursively implemented optimal edge detector. *International Journal of Computer Vision*, vol. 1(2), 167–187, April 1987.

[Deriche90] R. Deriche and O. Faugeras. Tracking Lines segments. *Image and Vision Computing*, vol. 8(4), 261–270, November 1990.

[Deutscher05] J. Deutscher and I. Reid. Articulated Body Motion Capture by Stochastic Search. *International Journal of Computer Vision*, vol. 61(2), 185–205, February 2005.

[Dhome89] M. Dhome, M. Richetin, J. Lapresté and G. Rives. Determination of the Attitude of 3-D Objects from a Single Perspective View. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, vol. 11(12), 1265–1278, December 1989.

[Dhome90] M. Dhome, J. Lapresté, G. Rives and M. Richetin. Determination of the attitude of modelled objects of revolution in monocular perspective vision. In *European Conference on Computer Vision, ECCV'90*, vol. LNCS 427, pages 475–485. Antibes, April 1990.

[Dickmanns88] E. Dickmanns and V. Graefe. Dynamic Monocular Machine Vision. *Machine Vision and Applications*, vol. 1, 223–240, 1988.

[Diderot51] D. Diderot. *Encyclopédie ou dictionnaire raisonné des sciences, des arts et des métiers*. Une societe de gens de lettres. Redon CDROM, Dordrecht, 1751.

[DiFranco99] D. DiFranco, T.-J. Cham and J.-M. Rehg. Recovery of 3D Articulated Motion from 2D Correspondences. Tech. rep., Cambridge Research Laboratory, December 1999.

[Dissanayake01] M. Dissanayake, P. Newman, S. Clark, H. Durrant-Whyte and M. Csorba. A solution to the simultaneous localization and map building (SLAM) problem. *IEEE Transactions on Robotics and Automation*, vol. 17(3), 229–241, June 2001.

[Doucet00] A. Doucet, S. Godsill and C. Andrieu. On Sequential Monte Carlo Sampling Methods for Bayesian Filtering. *Statistical Computer*, vol. 10(3), 197–208, 2000.

[Doucet01] A. Doucet, N. de Freitas and N. Gordon. *Sequential Monte Carlo Methods in Practice.* Springer-Verlag, New York, 2001.

[Drascic91] D. Drascic and P. Milgram. Positioning Accuracy of a Virtual Stereographic Pointer in a Real Stereoscopic Video World. In *Proceedings Stereoscopic Displays and Applications II*, vol. 1457, pages 302–313. SPIE Proceedings, San Jose, CA, February 1991.

[Drascic93] D. Drascic. Stereoscopic Vision and Augmented Reality. In *Scientific Computing & Automation*, vol. 9, pages 31–34. June 1993.

[Drummond99] T. Drummond and R. Cipolla. Real-Time Tracking of Complex Structures for Visual Servoing. In *International Workshop on Vision Algorithms*, vol. 1883, pages 69–84. Corfu, Greece, September 21-22 1999.

[Drummond00a] T. Drummond and R. Cipolla. Application of Lie Algebras to Visual Servoing. *IJCV*, vol. 37(1), 21–41, June 2000.

[Drummond00b] T. Drummond and R. Cipolla. Real-Time Tracking of Multiple Articulated Structures in Multiple Views. In *Sixth European Conference on Computer Vision*, vol. 2, pages 20–36. June 2000.

[Drummond01] T. Drummond and R. Cipolla. Real-time tracking of highly articulated structures in the presence of noisy measurements. In *International Conference on Comptuer Vision*, vol. 02, page 315. Vancouver, Canada, February 2001.

[Drummond02] T. Drummond and R. Cipolla. Real-Time Visual Tracking of Complex Structures. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, vol. 27(7), 932–946, July 2002.

[Eck96] M. Eck and H. Hoppe. Automatic reconstruction of B-spline surfaces of arbitrary topological type. In *23rd annual conference on Computer graphics and interactive techniques, SIGGRAPH '96*, pages 325–334. ACM Press, New York, NY, January 1996.

[Edelsbrunner88] H. Edelsbrunner and D. Souvaine. Computing median-of-squares regression lines and guided topological sweep. Tech. Rep. Report UIUCDCS-R-88-1483, Department of Computer Science, University of Illinois, 1988.

[Edwards95] C. Edwards. *Advanced Calculus of Several Variables*. Dover Publications, 1995.

[Ehnes04] J. Ehnes, K. Hirota and M. Hirose. Projected Augmentation - Augmented Reality using Rotatable Video Projectors. In *International Symposium on Mixed and Augmented Reality*, pages 26–35. Arlington, VA, USA, 2-5 November 2004.

[Eklundh01] J. Eklundh and H. Christensen. Computer Vision: Past and Future. *Lecture Notes in Computer Science, Informatics: 10 Years Back, 10 Years Ahead*, vol. 2000, 0–0, 2001.

[Elder98] J. Elder and S. Zucker. Local Scale Control for Edge Detection and Blur Estimation. *Pattern Analysis and Machine Intelligence, IEEE Transactions on*, vol. 20(7), 699 – 716, July 1998.

[Ellis94] S. Ellis and U. J.B. Distance Perception of Stereoscopically Presented Virtual Objects Optically Superimposed on Physical Objects by a Head-Mounted See-Through Display. In *Annual Meeting of the Human Factors and Ergonomics Society*, pages 1300–1305. Nashville, TN, 24-28 October 1994.

[Espiau92] B. Espiau, F. Chaumette and P. Rives. A new approach to visual servoing in robotics. *IEEE Transactions on Robotics and Automation*, vol. 8(3), 313–326, June 1992.

[Espiau93] B. Espiau. Effect of camera calibration errors on visual servoing in robotics. In *International Symposium on Experimental Robotics, ISER'93*. Kyoto, Japan, October 1993.

[Faugeras87] O. Faugeras and G. Toscani. Camera calibration for 3-D computer vision. In *Proc International Workshop on Machine Vision and Machine Intelligence*, pages 240–247. Tokyo, February 1987.

[Faugeras88] O. Faugeras and F. Lustman. Motion and structure from motion in a piecewise planar environment. *International Journal of Pattern Recognition and Artificial Intelligence*, vol. 2(3), 485–508, September 1988.

[Faugeras93] O. Faugeras. *Three-dimensional computer vision: a geometric viewpoint*. MIT Press, Cambridge, Massachusetts, 1993. Book.

[Faugeras98] O. Faugeras and T. Papadopoulo. A nonlinear method for estimating the projective geometry of 3 views. In *Sixth International Conference on Computer Vision*, pages 477–484. January 1998.

[Feddema89] J. Feddema and O. Mitchell. Vision-Guided Servoing with Feature-Based Trajectory Generation. *IEEE Transactions on Robotics and Automation*, vol. 5(5), 691–700, October 1989.

[Feiner93] S. Feiner, B. MacIntyre, M. Haupt and E. Solomon. Windows on the World: 2D Windows for 3D Augmented Reality. In *UIST*, pages 145–155. Atlanta, GA, 3-5 November 1993.

[Ferrin91] F. Ferrin. Survey of Helmet Tracking Technologies. In *Large-Screen Projection, Avionic, and Helmet-Mounted Displays*, vol. 1456, pages 86–94. SPIE, 1991.

[Fischler81] M. Fischler and R. Bolles. Random Sample Consensus: A Paradigm for Model Fitting with Application to Image Analysis and Automated Cartography. *Communication of the ACM*, vol. 24(6), 381–395, June 1981.

[Forsythe57] G. E. Forsythe. Generation and Use of Orthogonal Polynomials for Data-Fitting with a Digital Computer. *Journal of the Society for Industrial Applied Mathematics*, vol. Vol. 5, 195–209, 1957.

[Fua96] P. Fua and Y. G. Leclerc. Taking Advantage of Image-Based and Geometry-Based Constraints to Recover 3–D Surfaces. *Computer Vision and Image Understanding*, vol. 64(1), 111–127, July 1996.

[Fua97] P. Fua and C. Brechbuhler. Imposing Hard Constraints on Deformable Models through Optimization in Orthogonal Subspaces. *Computer Vision and Image Understanding*, vol. 65(2), 148–162, February 1997.

[Fukunaga75] K. Fukunaga and L. Hosteler. The estimation of the gradient of a density function with applications in pattern recognition. *IEEE Transactions on Information Theory*, vol. 21(1), 32–40, january 1975.

[Ganapathy84] S. Ganapathy. Decomposition of Transformation Matrices for Robot Vision. *Pattern Recognition Letter*, vol. 2, 401–412, 1984.

[Gao03] X.-S. Gao, X.-R. Hou, J. Tang and H.-F. Cheng. Complete solution classification for the perspective-three-point problem. *IEEE Transactions onPattern Analysis and Machine Intelligence*, vol. 25(8), 930–943, August 2003. TY - JOUR.

[Gennery81] D. Gennery. A feature-Based Scene Matcher. In *International Joint Conference on Artificial Intelligence*, pages 667–673. Vancouver, Canada, August 1981.

[Gennery82] D. Gennery. Tracking known 3-dimensional object. In *Second National Conference on Artificial Intelligence*, pages 13–17. Pittsburg, PA, August 18-20 1982.

[Gennery92] D. Gennery. Visual tracking of known three-dimensional objects. *International Journal of Computer Vision*, vol. 7(3), 243–270, April 1992.

[Giblin87] P. Giblin and R. Weiss. Reconstruction of Surfaces from Profiles. In *First International Conference on Computer Vision*, pages 136–144. London, UK, June 1987.

[Giordana00]	N. Giordana, P. Bouthemy, F. Chaumette, F. Spindler, J.-C. Bordas and V. Just. Two dimensional model-based tracking of complex shapes for visual servoing tasks. In *Robust vision for vision-based control of motion* (edited by M. Vincze and G. Hager), vol. Ch. 6, pages 67–75. IEEE Press, 2000.
[Goldstein87]	E. B. Goldstein. *Sensation and Perception*. Wadsworth Publishing Co, 2nd edition edn., 1987.
[Goldstein02]	H. Goldstein, C.-P. Poole and J.-L. Safko. *Classical mechanics*. Addison Wesley, San Francisco, 3rd edn., 2002. Herbert Goldstein, Charles Poole, John Safko.
[Gordon93]	N. Gordon, D. Salmond and A. Smith. Novel approach to nonlinear/non-Gaussian Bayesian state estimation. *IEE Proceedings on Radar and Signal Processing*, vol. 140(2), 107–113, April 1993.
[Grimson87]	E. Grimson and L. Pérez. Localizing Overlapping Parts by Searching the Interpretation Tree. *IEEE Transaction on Pattern Analysis and Machine Intelligence*, vol. 9, 469–482, July 1987.
[Grimson95]	W. Grimson, G. Ettinger, S. White, P. Gleason, T. Lozano-Pérez, W. Wells III and R. Kikinis. Evaluating and Validating an Automated Registration System for Enhanced Reality Visualization in Surgery. In *Computer Vision, Virtual Reality, and Robotics in Medicine*, pages 3–12. Nice, France, 3-6 April 1995.
[Grimson96]	E. Grimson, G. Ettinger, S. White, T. Lozano-Pérez, W. Wells III and R. Kikinis. An automatic registration method for frameless stereotaxy, image guided surgery, and enchanced reality visualization. *IEEE Transactions on Medical Imaging*, vol. 15(2), 129–140, April 1996.
[Grosso96]	E. Grosso, G. Metta, A. Oddera and G. Sandini. Robust visual servoing in 3-D reaching tasks. *IEEE Transactions on Robotics and Automation*, vol. 12, 732–742, October 1996.
[Haag98]	M. Haag and H. Nagel. Tracking of Complex Driving Maneuvers in Traffic Image Sequences. *Image and Vision Computing*, vol. 16(8), 517–527, June 1998.
[Haag99]	M. Haag and H. Nagel. Combination of Edge Element and Optical Flow Estimates for 3D-Model-Based Vehicle Tracking in Traffic Image Sequences. *International Journal of Computer Vision*, vol. 35(3), 295–319, December 1999.
[Hager95a]	G. Hager, W. Chang and A. Morse. Robot Feedback Control Based on Stereo Vision: Towards Calibration-Free Hand-Eye Coordination. *IEEE Control Systems Magazine*, vol. 15(1), 30–39, February 1995.
[Hager95b]	G. Hager, G. Grunwald and K. Toyama. Feature-Based Visual Servoing and its Application to Telerobotics. In *Intelligent Robotic Systems* (edited by V. Graefe). Elsevier, Amsterdam, March 1995.
[Hager98a]	G. Hager and P. Belhumeur. Efficient Region Tracking With Parametric Models of Geometry and Illumination. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, vol. 20(10), 1025–1039, October 1998.

[Hager98b]	G. Hager and K. Toyama. The XVision System: A General-Purpose Substrate for Portable Real-Time Vision Applications. *Computer Vision and Image Understanding*, vol. 69(1), 23–37, January 1998.
[Hampel86]	F. Hampel, E. Ronchetti, P. Tousseeuw and W. Stahel. *Robust Statistics: An Approach Based on Influence Functions.* Wiley, J., New York, 1986.
[Haralick81]	R. Haralick. Edge and Region Analysis for Digital Image Data. In *Computer Graphics and Image Processing* (edited by A. Rosenfeld), vol. 12, pages 60–73. January 1981.
[Haralick82]	R. Haralick. Zero-Crossing of Second Directional Derivative Edge Operator. In *Proceedings of SPIE, The International Society for Optical Engineering Technical Symposium*, vol. 336, pages 91–99. East, Arlington, VA, May 6-7 1982.
[Haralick87]	R. Haralick, H. Joo, C. Lee, X. Zhuang, V. Vaidya and M. Kim. Pose Estimation from Corresponding Point Data. In *Workshop on Computer Vision, Representation and Control*, vol. 6, pages 258–263. November-December 1987.
[Haralick88]	R. Haralick and H. Joo. 2D-3D Pose Estimation. In *IAPR International Conference on Pattern Recognition, ICPR'88*, vol. 1, pages 385–391. Rome, Italy, November 1 1988.
[Haralick89]	R. Haralick, H. Joo, C. Lee, X. Zhuang, V. Vaidya and M. Kim. Pose estimation from corresponding point data. *IEEE Trans on Systems, Man and Cybernetics*, vol. 19(6), 1426–1445, November 1989.
[Haralick91]	R. Haralick, C. Lee, K. Ottenberg and M. Nolle. Analysis and Solutions of the Three Point Perspective Pose Estimation Problem. In *CVPR91*, pages 592–598. Hawaii, June3-6 1991.
[Haralick92]	R. Haralick and L. G. Shapiro. *Computer and robot vision*. Addison-Wesley Pub. Co., Reading, Mass., 1992. Robert M. Haralick, Linda G. Shapiro. ill.; 25 cm.
[Harris88]	C. Harris and M. Stephens. A Combined Corner and Edge Detector. In *4th Alvey Vision Conference*, pages 189–192. Manchester University, England, September 1988.
[Hartley94]	R. Hartley. Lines and points in three views - an integrated approach. In *DARPA Image Understanding Workshop* (edited by M. Kaufmann), pages 1009–1016. Monterey, CA, November 13-16 1994.
[Hartley01]	R. Hartley and A. Zisserman. *Multiple View Geometry in computer vision*. Cambridge University Press, 2001. Book.
[Hashimoto93]	K. Hashimoto. *Visual Servoing: Real Time Control of Robot Manipulators Based on Visual Sensory Feedback*, vol. 7 of *Robotics and Automated Systems*. World Scientific Press, Singapore, 1993.

[Henrich98] A. Henrich. The LSDh–Tree: An Access Structure for Feature Vectors. In *International Conference on Data Engineering (ICDE)*, pages 362–369. Orlando, FL, 23-27 February 1998.

[Hoff96] W. Hoff, T. Lyon and K. Nguyen. Computer Vision-based registration techniques for augmented reality. In *Intelligent Robotics and Computer Vision XV*, vol. 2904, pages 538–548. SPIE, Boston, Massachusetts, November 18-22 1996.

[Holland77] P.-W. Holland and R.-E. Welsch. Robust regression using iteratively reweighted least-squares. *Communications in Statistics - Theory and Methods*, vol. A6, 813–827, August 1977.

[Holmgren92] D. Holmgren. Design and Construction of a 30-Degree See-Through Head-Mounted Display. Tech. Rep. 92-030, UNC Chapel Hill Department of Computer Science, July 1992.

[Horaud89] R. Horaud, B. Conio, O. Leboulleux and B. Lacolle. An analytic solution for the perspective 4-points problem. *Computer Vision, Graphics and Image Processing*, vol. 47(1), 33–44, July 1989.

[Horaud98] R. Horaud, F. Dornaika and B. Espiau. Visually guided object grasping. *IEEE Transactions on Robotics and Automation*, vol. 14(4), 525–532, August 1998.

[Horn81] B. K. P. Horn and B. G. Schunck. Determining optical flow. *Artificial Intelligence*, vol. 17, 185–203, 1981.

[Hosoda94] K. Hosoda and M. Asada. Versatile visual servoing without knowledge of true jacobian. In *Proc of IEEE/RSJ International Conf on Intelligent Robots and Systems, IROS'94*, pages 186–193. Munich, Germany, August 1994.

[Hough59] P. Hough. Machine analysis of bubble chamber pictures. In *International Conference on High Energy Accelerators and Instrumentation*. Centre Europèen pour la Recherche Nuclèaire (CERN), 1959.

[Hough62] P. Hough. Method and means for recognizing complex patterns, December 18 1962. Patent.

[Huang90] T. Huang. Modeling, analysis and visualization of nonrigid object motion. In *IEEE Tenth International Conference on Pattern Recognition*, pages 361–364. Atlantic City, NJ, June 16-21 1990.

[Huber81] P.-J. Huber. *Robust Statistics*. Wiler, New York, 1981.

[Huber96] P. Huber. Robust Statistical Procedures. *SIAM Review, second edition*, 1996.

[Hue02a] C. Hue, J.-P. Le Cadre and P. Pérez. Sequential Monte Carlo methods for multiple target tracking and data fusion. *IEEE Transaction on Signal Processing*, vol. 50(2), 309–325, Février 2002.

[Hue02b] C. Hue, J.-P. Le Cadre and P. Pérez. Tracking multiple objects with particle filtering. *IEEE Transaction on Aerospace and Electronic Systems*, vol. 38(3), 791–812, July 2002.

[Hutchinson96] S. Hutchinson, G. Hager and P. Corke. A tutorial on Visual Servo Control. *IEEE Transactions on Robotics and Automation*, vol. 12(5), 651–670, October 1996.

[Ilic03] S. Ilic and P. Fua. Generic Deformable Implicit Mesh Models for Automated Reconstruction. In *ICCV Workshop on Higher-Level Knowledge in 3D Modelling and Motion Analysis*. Nice, France, October 2003.

[Isard96] M. Isard and A. Blake. Contour tracking by stochastic propagation of conditional density. In *European Conference on Computer Vision, ECCV'96, LNCS no. 1064,*, vol. 1, pages 343–356. Springer-Verlag, Cambridge, UK, April 13-14 1996.

[Isard98a] M. Isard and A. Blake. CONDENSATION - conditional density propagation for visual tracking. *International Journal of Computer Vision*, vol. 29(1), 5–28, January 1998.

[Isard98b] M. Isard and A. Blake. ICONDENSATION: Unifying low-level and high-level tracking in a stochastic framework,. In *Fifth European Conference on Computer Vision*, vol. 1, page 893 908. Freiburg, Germany, June 2-6 1998.

[Jaeckel72] L. Jaeckel. Estimating regression coefficients by minimizing the dispersion of residuals. *Annual Mathematics Statistics*, vol. 43, 1449–1458, 1972.

[Jägersand97] M. Jägersand, O. Fuentes and R. Nelson. Experimental evaluation of uncalibrated visual servoing. In *IEEE International Conference on Robotics and Automation*, vol. 3, pages 2874–2880. Albuquerque, NM, April 1997.

[Jain81] J. Jain and A. Jain. Displacement measurement and its application in interframe image coding. *IEEE Transactions on Communications*, vol. 29(12), 1799–1808, December 1981.

[Janin93] A. Janin, D. Mizell and T. Caudell. Calibration of Head-Mounted Displays for Augmented Reality Applications. In *IEEE VRAIS*, pages 246–255. Seattle, WA, 18-22 September 1993.

[Jarvis83] R. Jarvis. A perspective on range finding techniques for computer vision. *IEEE Transactions on Pattern Analysis and Machine intelligence*, vol. 5, 122–139, March 1983.

[Julier99] S. Julier, R. King, B. Colbert, J. Durbin and L. Rosenblum. The Software Architecture of a Real-Time Battlefield Visualization Virtual Environment. In *IEEE Virtual Reality*, pages 29–36. IEEE Computer Society, Houston, TX, 13-17 March 1999.

[Jurie01] F. Jurie and M. Dhome. Real Time 3D Template Matching. In *International Conference on Computer Vision and Pattern Recognition*, vol. 1, pages 791–796. Hawai, December 2001.

[Jurie02] F. Jurie and M. Dhome. Hyperplane Approximation for Template Matching. *IEEE trans on Pattern Analysis and Machine Intelligence*, vol. 24(7), 996–1000, July 2002.

[Kanade81] T. Kanade. Recovery of the three-dimensional shape of an object from a single view. In *Artificial Intelligence*, vol. 17, pages 409–460. 1981.

[Kato99] H. Kato and M. Billinghurst. Marker Tracking and HMD Calibration for a video-based Augmented Reality Conferencing System. In *ACM/IEEE International Workshop on Augmented Reality, IWAR'99*, pages 85–95. San Francisco, CA, October 1999.

[Kato00] H. Kato, M. Billinghurst, I. Poupyrev, K. Imamoto and K. Tachibana. Virtual Object Manipulation on a Table-Top AR Environment. In *Proceedings of International Symposium. on Augmented Reality 2000*, pages 111–119. Munich, Germany, October 2000.

[Kealy04] A. Kealy, S. Scott-Young, P. Collier and I. D. Bishop. Improving Land Vehicle Safety Using Augmented Reality Technologies. In *International Symposium on Mobile Mapping Technology*. Kunming, China, March 29-31 2004.

[Kelly96] R. Kelly. Robust asymptotically stable visual servoing of planar robots. *IEEE Transactions on Robotics and Automation*, vol. 12(5), 759–766, October 1996.

[Kervrann94] C. Kervrann and F. Heitz. A hierarchical statistical framework for the segmentation of deformable objects in image sequences. In *IEEE Conference Computer Vision Pattern Recognition*, pages 724–728. Seattle, USA, June 1994.

[Kervrann98] C. Kervrann and F. Heitz. A Hierarchical Markov Modeling Approach for the Segmentation and Tracking of Deformable Shapes. *Graphical Models and Image Processing*, vol. 60(3), 173–195, May 1998.

[Kim89] D. Kim, J. Kim, P. Meer, D. Mintz and A. Rosenfeld. Robust computer vision: The least median of squares approach. In *Understanding Workshop in DARPA Image Proceedings*, pages 1117–1134. Palo Alto, CA, May 1989.

[Kinoshita94] K. Kinoshita and K. Deguchi. Simultaneous Determination of Camera Pose and Intrinsic Parameters by Visual Servoing. In *IAPR International Conference on Pattern Recognition, ICPR94*, vol. A, pages 285–289. Jerusalem, October 1994.

[Kleeman95] L. Kleeman and R. Kuc. Mobile robot sonar for target localization and classification. *International Journal of Robotics Research*, vol. 14(4), 295–318, August 1995.

[Klein03] G. Klein and T. Drummond. Robust Visual Tracking for Non-Instrumented Augmented Reality. In *ACM/IEEE International Symposium. on Mixed and Augmented Reality, ISMAR'03*, pages 113–122. Tokyo, Japan, October 2003.

[Klein04] G. Klein and T. Drummond. Sensor Fusion and Occlusion Refinement for Tablet-Based AR. In *International Symposium on Mixed and Augmented Reality*, pages 38–47. Arlington, VA, USA, 2-5 November 2004.

[Koenderink84] J. Koenderink. What Does the Occluding Contour Tell us About Solid Shape? *Perception*, vol. 13(3), 321–330, 1984.

[Koller93] D. Koller, K. Daniilidis and H.-H. Nagel. Model-Based Object Tracking in Monocular Image Sequences of Road Traffic Scenes. *International Journal of Computer Vision*, vol. 10(2), 257–281, June 1993.

[Koller94] D. Koller, J. Weber and J. Malik. Robust Multiple Car Tracking with Occlusion Reasoning. In *Third European Conference on Computer Vision*, pages 186–196. Springer-Verlag, LNCS 800, May 2-6 1994.

[Koller97] D. Koller, G. Klinger, E. Rosse, D. Breen, R. Whitaker and M. Tuceryan. Real-time Vision-Based Camera Tracking for augmented reality applications. In *International Symposium. on Virtual Reality Software and Technology, VRST'97*, pages 87–94. Lausanne, Switzerland, September 1997.

[Kollnig95] H. Kollnig and H.-H. Nagel. 3D Pose Estimation by fitting Image Gradients Directly to Polyhedral Models. In *IEEE International Conference on Computer Vision*, pages 569–574. Boston, MA, May 1995.

[Kovesi96] P. Kovesi. *Invariant Measures of Feature Detection.* Ph.d. thesis, The University of Western Australia, 1996.

[Kragic99] D. Kragic and H. Christensen. Integration of visual cues for active tracking of an end–effector. In *IEEE/RSJ International Conference on Intelligent Robots and Systems*, vol. 1, pages 362–268. Kyongju, Korea, October 1999.

[Kragic01] D. Kragic and H. Christensen. Cue integration for visual servoing. *IEEE Transactions on Robotics and Automation*, vol. 17(1), 19–26, February 2001.

[Kumar89] R. Kumar and A. Hanson. Robust estimation of camera location and orientation from noisy data having outliers. In *Workshop on Interpretation of 3D scenes*, pages 52–60. Austin, TX, December 1989.

[Kumar94] R. Kumar and A. Hanson. Robust methods for estimating pose and a sensitivity analysis. *CVGIP: Image Understanding*, vol. 60(3), 313–342, November 1994.

[Kutulakos95] K. Kutulakos. Affine Surface Reconstruction by Purposive Viewpoint Control. In *IEEE International Conference on Computer Vision, ICCV'95*, pages 894–901. Cambridge, Massachusset, June 1995.

[Lapresté04] J. Lapresté, F. Jurie, M. Dhome and F. Chaumette. An Efficient Method to Compute the Inverse Jacobian Matrix in Visual Servoing. In *IEEE International Conference on Robotics and Automation, ICRA'04*, vol. 1, pages 727–732. New Orleans, April 2004.

[Laptev03] I. Laptev and T. Lindeberg. Space-Time Interest Points. In *Nineth IEEE International Conference on Computer Vision*, vol. 1, pages 432–439. Nice, France, 13-16 October 2003.

[Lee89] C. Lee, R. Haralick and X. Zhuang. Recovering 3-D motion parameters from image sequences with gross errors. In *Workshop on Visual Motion*, pages 46–53. Irvine, CA, 20-22 March 1989.

[Leonard91]	J. Leonard and H. Durrant-Whyte. Simultaneous map building and localization for an autonomous mobile robot. In *International Workshop on Intelligent Robots and Systems IROS '91*, vol. 3, pages 1442–1447. Osaka, Japan, 3-5 November 1991.
[Leonardis97]	A. Leonardis, A. Jaklic and F. Solina. Superquadrics for segmenting and modeling range data. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, vol. 19(11), 1289–1295, November 1997. 0162-8828.
[Lepetit00]	V. Lepetit and M. Berger. A Semi Automatic Method for Resolving Occlusions in Augmented Reality. In *IEEE Conference on Computer Vision and Pattern Recognition*, vol. 2, pages 225–230. Hilton Head Island, South Carolina (USA), 13-15 June 2000.
[Lepetit03]	V. Lepetit, L. Vacchetti, T. Thalmann and P. Fua. Fully Automated and Stable Registration for Augmented Reality Applications. In *ACM/IEEE International Symposium. on Mixed and Augmented Reality, ISMAR'03*, pages 93–102. Tokyo, Japan, October 2003.
[Lepetit04]	V. Lepetit, J. Pilet and P. Fua. Point Matching as a Classification Problem for Fast and Robust Object Pose Estimation. In *In Conference on Computer Vision and Pattern Recognition*. Washington, DC, June 2004.
[Li85]	G. Li. *Robust regression.* Exploring Data Tables, Trends and Shapes. Wiley, J., New York, 1985.
[Li04]	P. Li and F. Chaumette. Image Cues Fusion for Contour Tracking Based on Particle Filter. In *In International Workshop on articulated motion and deformable objects, AMDO'04* (edited by J. Perales and B. Draper), vol. 3179 of *Lecture Notes in Computer Science*, pages 99–107. Palma de Majorque, Spain, Septembre 2004.
[Lindeberg98]	T. Lindeberg. Edge Detection and Ridge Detection with Automatic Scale Selection. *International Journal of Computer Vision*, vol. 30(2), 117–156, November 1998. TY - JOUR.
[Lindgren93]	B. Lindgren. *Statistical Theory.* Chapman and Hall, London, 1993.
[Liu90]	Y. Liu, T. Huang and O. Faugeras. Determination of Camera Location from 2-D to 3-D Line and Point Correspondences. *IEEE Transactions On Pattern Analysis and Machine Intelligence*, vol. 12(1), 28–37, January 1990.
[Liu93]	Y. Liu, J. Mundy, D. Forsyth, K. Zissennan and C. Rothwell. Efficient Recognition of Rotationally Symmetric Surfaces and Straight Homogeneous Generalized Cylinders. In *IEEE Computer Vision and Pattern Recognition*, pages 123–128. New York, 15-17 June 1993.
[Longuet-Higgins81]	H. Longuet-Higgins. A Computer Algorithm for Reconstructing a Scene from Two Projections. *Nature*, vol. 293, 133–135, September 1981.
[Lorensen93]	B. Lorensen, W. Cline, C. Nafis, R. Kikinis, D. Altobelli and G. L. Enhancing Reality in the Operating Room. In *IEEE conference on Visualization*, pages 410–415. Los Alamitos, CA, October 1993.

[Lowe80] D. Lowe. Solving for the parameters of object models from image descriptions. In *ARPA Image Understanding Workshop*, pages 121–127. College Park, MD, April 1980.

[Lowe85] D. Lowe. *Perceptual Organization and Visual Recognition*. Kluwer Academic Publishers, Norwell, M.A., 1985.

[Lowe87] D. Lowe. Three-dimensional object recognition from single two-dimensional images. *Artificial Intelligence*, vol. 31(3), 355–394, March 1987.

[Lowe91] D. Lowe. Fitting parameterized three-dimensional models to images. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, vol. 13(5), 441–450, May 1991.

[Lowe92] D. Lowe. Robust Model-based motion tracking through the integration of search and estimation. *International Journal of Computer Vision*, vol. 8(2), 113–122, August 1992.

[Lowe04] D. G. Lowe. Distinctive Image Features from Scale-Invariant Keypoints. *International Journal of Computer Vision*, vol. 60(2), 91–110, November 2004.

[Lu00] C. Lu, G. Hager and E. Mjolsness. Fast and Globally Convergent Pose Estimation from Video Images. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, vol. 22(6), 610–622, June 2000.

[Lucas81] B. Lucas and T. Kanade. An Iterative Image Registration Technique with an Application to Stereo Vision. In *International Joint Conference on Artificial Intelligence*, vol. 2, pages 674–679. Vancouver, Canada, August 1981.

[Luong97] Q.-T. Luong and O. Faugeras. Self Calibration of a moving camera from point correspondances and fundamental matrices. *International Journal of Computer Vision*, vol. 22(3), 261–289, 1997.

[Ma04] Y. Ma, S. Soatto, J. Kosecka and S. Sastry. *An invitation to 3-D vision: from images to geometric models*, vol. 26 of *Interdisciplinary applied mathematics*. Springer, New York, 2004.

[MacCormick98] J. MacCormick and A. Blake. Spatial Dependence in the Observation of Visual Contours. In *Fifth European Conference on Computer Vision (ECCV '98)*, pages 765–781. Freiburg, Germany, 2-6 June 1998.

[Malis98] E. Malis. *Contributions à la modélisation et à la commande en asservissement visuel*. Phdthesis, Université de Rennes 1, Télécommunications et Traitement du Signal, 1998.

[Malis99] E. Malis, F. Chaumette and S. Boudet. 2 1/2 D Visual servoing. *IEEE Transactions on Robotics and Automation*, vol. 15(2), 238–250, April 1999.

[Malis00] E. Malis, F. Chaumette and S. Boudet. 2 1/2 D Visual Servoing with Respect to Unknown Objects Through a New Estimation Scheme of Camera Displacement. *International Journal of Computer Vision*, vol. 37(1), 79–97, June 2000.

[Marchand96] E. Marchand and F. Chaumette. Controlled Camera Motions for Scene Reconstruction and Exploration. In *IEEE International Conference on Computer Vision and Pattern Recognition, CVPR'96*, pages 169–176. San Francisco, CA, June 1996.

[Marchand99a] E. Marchand, P. Bouthemy, F. Chaumette and V. Moreau. Robust real-time visual tracking using a 2D-3D model-based approach. In *IEEE International Conference on Computer Vision, ICCV'99*, vol. 1, pages 262–268. Kerkira, Greece, September 1999.

[Marchand99b] E. Marchand and F. Chaumette. An Autonomous Active Vision System for Complete and Accurate 3D Scene Reconstruction. *International Journal of Computer Vision*, vol. 32(3), 171–194, August 1999.

[Marchand01] E. Marchand and F. Chaumette. *A new formulation for non-linear camera calibration using virtual visual servoing*. IRISA, Rennes, 2001. Report No 1366.

[Marchand02a] E. Marchand and F. Chaumette. Virtual Visual Servoing: a framework for real-time augmented reality. In *EUROGRAPHICS'02 Conference Proceeding*, vol. 21(3) of *Computer Graphics Forum*, pages 289–298. Saarebrücken, Germany, September 2002.

[Marchand02b] E. Marchand, J. Royan and F. Chaumette. Calcul de pose et calibration par asservissement visuel virtuel: application à la réalité augmentée. In *13éme Congrès Francophone AFRIF-AFIA de Reconnaissance des Formes et Intelligence Artificielle, RFIA 2002*, vol. 3, pages 839–848. Angers, France, January 2002.

[Marchand04] E. Marchand, A.-I. Comport and F. Chaumette. Improvements in robust 2D visual servoing. In *IEEE International Conference on Robotics and Automation, ICRA'04*, vol. 1, pages 745–750. New Orleans, April 2004.

[Maronna76] R. Maronna. Robust M-estimators of multivariate location and scatter. *Annual Statistics*, vol. 4, 51–67, 1976.

[Marr78] D. Marr and K. Nishihara. Representation and recognition of the spatial organization of three-dimensional shapes. In *Royal Society London*, vol. B200, pages 269–294. London, 1978.

[Marr80] D. Marr and E. C. Hildreth. Theory of edge detection. In *Proceedings of the Royal Society*, vol. 207B, pages 187–217. London, February 1980.

[Marr81] D. C. Marr and S. Ullman. Directional seletivity and its use in early visual processing. In *Royal Society of London*, vol. B211, pages 151–180. March 1981.

[Marr82] D. Marr. A computational investigation into the human representation and processing of visual information. In *Vision* (edited by H. F. a. Co). San Francisco, CA, 1982.

[Martin02] F. Martin and R. Horaud. Multiple Camera Tracking of Rigid Objects. *International Journal of Robotics Research*, vol. 21(2), 97–113, February 2002.

[Martinet97] P. Martinet, N. Daucher, J. Gallice and M. Dhome. Robot control using monocular pose estimation. In *Workshop on new trends in image-based robot servoing, IROS'97*, pages 1–12. Grenoble, September 1997.

[Masson04] L. Masson, M. Dhome and F. Jurie. Robust Real Time Tracking of 3D Objects. In *International Conference on Pattern Recognition*, vol. 4, pages 252–255. Cambridge, UK, 23-26 August 2004.

[Maybank92] S.-J. Maybank and O. Faugeras. A theory of self calibration of a moving camera. *International Journal of Computer Vision, IJCV*, vol. 8(2), 123–152, August 1992.

[Meer91] P. Meer, D. Mintz, D. Kim and A. Rosenfeld. Robust Regression Methods for Computer Vision: A Review. *International Journal of Computer Vision*, vol. 6(1), 59–70, April 1991.

[Meer00] P. Meer, C. Stewart and D. Tyler. Robust Computer Vision: An Interdisciplinary Challenge. *Computer Vision and Image Understanding: CVIU*, vol. 78(1), 1–7, April 2000.

[Meer04] P. Meer. *Robust techniques for computer vision.* Emerging Topics in Computer Vision. Prentice Hall, 2004.

[Mellor95] J. Mellor. *Enhanced Reality Visualization in a Surgical Environment.* Ph.D. thesis, Massachusetts Institute of Technology, 1995.

[Menegatti04] E. Menegatti, T. Maeda and H. Ishiguro. Image-based memory for robot navigation using properties of omnidirectional images. *Robotics and Autonomous Systems*, vol. 47, 251–267, September 2004.

[Merriam-Webster04] Merriam-Webster. Imagination, 2 December 2004. Online Dictionary, http://www.merriam-webster.com.

[Metaxas00] D. Metaxas and I. Kakadiaris. Model-Based Estimation of 3D Human Motion. *IEEE Transacations on Pattern Analysis and Machine Intelligence*, vol. 22(12), 1453–1459, December 2000.

[Metaxas02] D. Metaxas and I. Kakadiaris. Elastically adaptive deformable models. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, vol. 24(10), 1310–1321, October 2002.

[Meyer92] K. Meyer, H. Applewhite and F. Biocca. A Survey of Position-Trackers. *Presence: Teleoperators and Virtual Environments*, vol. 1(2), 173–200, Spring 1992.

[Mikolajczyk04] K. Mikolajczyk and C. Schmid. Scale and affine invariant interest point detectors. *International Journal of Computer Vision*, vol. 60(1), 63–86, October 2004.

[Milgram93] P. Milgram, Z. Shumin, D. Drascic and J. Grodski. Applications of Augmented Reality for Human-Robot Communication. In *Proceedings of International Conference on Intelligent Robotics and Systems*, pages 1467–1472. Yokohama, Japan, July 1993.

[Milgram94a] P. Milgram, T. Haruo, U. Akira and K. Fumio. Augmented Reality: A Class of Displays on the Reality–Virtuality Continuum. In *Telemanipulator and Telepresence Technologies*, vol. 2351, pages 282–292. SPIE, Boston, MA, 31 October - 4 November 1994.

[Milgram94b] P. Milgram and F. Kishino. A Taxonomy of Mixed Reality Virtual Displays. *IEICE Transactions on Information and Systems*, vol. 12(E77-D), 1321–1329, September 1994.

[Mohring04] M. Mohring, C. Lessig and O. Bimber. Video See-Through AR on Consumer Cell-Phones. In *International Symposium on Mixed and Augmented Reality*, vol. Poster, pages 252–253. Arlington, VA, USA, 2-5 November 2004.

[Monga91] O. Monga, R. Deriche, G. Malandain and J. Cocquerez. Recursive filtering and edge tracking: two primary tools for 3D edge detection. *Image and Vision Computing*, vol. 9(4), 203–214, August 1991.

[Moreno-Noguer04] F. Moreno-Noguer, A. Sanfeliu and D. Samaras. Fusion of a Multiple Hypotheses Color Model and Deformable Contours for Figure Ground Segmentation in Dynamic Environments. In *IEEE Conference on Computer Vision and Pattern Recognition Workshop*, vol. 1, page 13. Washington, DC, June 2004.

[Mosteller77] F. Mosteller and J. Tukey. *Data Analysis and Regression.* Addison Wesley, Reading, MA, 1977.

[Murase93] H. Murase and S. K. Nayar. Learning and recognition of 3D objects from appearance. In *IEEE Workshop on Qualitative Vision*, pages 39–50. New York, USA, 14 June 1993.

[Murase95] H. Murase and S. K. Nayar. Visual Learning and Recognition of 3D Objects from Appearance. *International Journal of Computer Vision*, vol. 14(1), 5–24, 1995.

[Murray94] R. Murray, L. Zexiang and S. Sastry. *A Mathematical introduction to robotic manipulation*. CRC Press, Boca Raton, Florida, 1994.

[Navab93] N. Navab and O. Faugeras. Monocular pose determination from lines: Critical sets and maximum number of solutions. In *IEEE Conference on Computer Vision and Pattern Recognition*, pages 254–260. New York, USA, 15-17 June 1993.

[Navab99] N. Navab, B. Bascle, M. Appel and E. Cubillo. Scene Augmentation via the Fusion of Industrial Drawings and Uncalibrated Images with a View to Markerless Calibration. In *Second International Workshop on Augmented Reality IWAR*, pages 125–133. CS Press, San Francisco, California, 20-21 October 1999.

[Nayar94] S. K. Nayar, H. Murase and S. A. Nene. Learning, positioning, and tracking visual appearance. In *IEEE International Conference on Robotics and Automation*, pages 3237–3246. San Diego, CA, May 1994.

[Nayar96a] S. Nayar, S. Nene and H. Murase. Real-Time 100 Object Recognition System. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, vol. 18(2), 1186–1198, 1996.

[Nayar96b] S. K. Nayar. Subspace Methods for Robot Vision. *IEEE Transactions on Robotics and Automation*, vol. 12(5), 750–758, October 1996.

[Nelson94] B. Nelson and P. Khosla. The Resolvability Ellipsoid for Visual Servoing. In *IEEE International Conference on Computer Vision and Pattern Recognition, CVPR'94*, pages 829–832. Seattle, Washington, June 1994.

[Neumann96] U. Neumann and Y. Cho. A Self-Tracking Augmented Reality System. In *ACM Symposium on Virtual Reality Software and Technology*, pages 109–115. Hong Kong, 1-4July 1996.

[Neumann99] U. Neumann, S. You, Y. Cho, J. Lee and J. Park. Augmented Reality Tracking in Natural Environments. In *International Symposium on Mixed Realities*, pages 101–130. Tokyo, Japan, March 1999.

[Nickels01] K. Nickels and S. Hutchinson. Model-Based Tracking of Complex Articulated Objects. *IEEE Transactions on Robotics and Automation*, vol. 17(1), 28–36, February 2001.

[Nicolau04] S. Nicolau, J. Schmid, X. Pennec, L. Soler and N. Ayache. An Augmented Reality & Virtuality Interface for a Puncture Guidance System: Design and Validation on an Abdominal Phantom. In *Second International Workshop on Medical Imaging and Augmented Reality MIAR* (edited by G. Yang and T. Jiang), vol. 3150 of *LNCS*, pages 302–310. Springer Verlag, Beijing, China, August 2004.

[Nitzan88] D. Nitzan. Three-dimensional vision structure for robot applications. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, vol. 10(3), 291–309, May 1988.

[Nunomaki00] T. Nunomaki, S. Yonemoto, D. Arita and R. Taniguchi. Multipart NoN-Rigid Object Tracking Based on Time Model-Space Gradients. In *First International Workshop onArticulated Motion and Deformable Objects*, pages 78–82. Palma de Mallorca, Spain, September 2000.

[Odobez95] J.-M. Odobez and P. Bouthemy. Robust multiresolution estimation of parametric motion models. *Journal of Visual Communication and Image Representation*, vol. 6(4), 348–365, December 1995.

[Ottenbacher97] A. Ottenbacher, M. Tomasini, K. Holmund and J. Schmetz. Low-level cloud motion winds from Meteosat high-resolution visible imagery. *Weather and Forecasting*, vol. 12(1), 175–184, 1997.

[Papanikolopoulos93a] N. Papanikolopoulos and P. Khosla. Selection of Features and Evaluation of Visual Measurements for 3-D Robotic Visual Tracking. In *International Symposium. on Intelligent Control*, pages 320–325. Chicago, Illinois, 25-27 August 1993.

[Papanikolopoulos93b] N. Papanikolopoulos, P. Khosla and T. Kanade. Visual Tracking of a Moving Target by a Camera Mounted on a Robot: A Combination of Control and Vision. *IEEE Transactions on Robotics and Automation*, vol. 9(1), 14–35, February 1993.

[Papanikolopoulos95] N. Papanikolopoulos, B. Nelson and P. Khosla. Six Degree-of-Freedom Hand/Eye Visual Tracking with Uncertain Parameters. *IEEE Transactions on Robotics and Automation*, vol. 11(5), 725–732, October 1995.

[Pasman03] W. Pasman and C. Woodward. Implementation of an augmented reality system on a PDA. In *International Symposium on Mixed and Augmented Reality*, pages 276–277. Tokyo, Japan, 7-10 October 2003.

[Paterson90] M. Paterson and F. Yao. Efficient Binary Space Partitions for Hidden Surface Removal and Solid Modeling, Discrete and Computational Geometry. In *Fifth annual ACM symposium on computational geometry*, vol. 5, pages 485–503. Springer-Verlag New York, Saarbrucken, Germany, 5-11 June 1990.

[Pentland94] A. Pentland, B. Moghaddam and T. Starner. View-based and modular eigenspaces for face recognition. In *IEEE Conference on Computer Vision and Pattern Recognition*, pages 84–91. Seattle, WA, June 1994.

[Pérez04] P. Pérez, J. Vermaak and A. Blake. Data fusion for visual tracking with particles. In *Proceedings of the IEEE*, vol. 92, pages 495–513. February 2004.

[Perona90] P. Perona and J. Malik. Scale-space and Edge Detection using Anisotropic Diffusion. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, vol. 12(7), 629–639, July 1990.

[Phong95] T.-Q. Phong, R. Horaud, A. Yassine and P.-D. Tao. Object Pose from a 2D to 3D point and line correspondance. *International Journal of Computer Vision*, vol. 15(3), 225–243, July 1995.

[Piekarski02] W. Piekarski and B. Thomas. ARQuake: The Outdoor Augmented Reality Gaming System. In *Communications of the ACM*, vol. 45, pages 36–38. January 2002.

[Pirjanian98] P. Pirjanian, H. Christensen and J. Fayman. Application of Voting to Fusion of Purposive Modules: An Experimental Investigation. *Robotics and Autonomous Systems*, vol. 23(4), 253–266, 1998.

[Plankers03] R. Plankers and P. Fua. Articulated Soft Objects for Multiview Shape and Motion Capture. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, vol. 25(9), 1182–1187, October 2003.

[Press02] W. Press, W. Vetterling, S. Teukolsky and B. Flannery. *Numerical Recipes in C++: the art of scientific computing*. Cambridge University Press, 2002.

[Pressigout04] M. Pressigout and E. Marchand. Model-free augmented reality by virtual visual servoing. In *IAPR International Conference on Pattern Recognition, ICPR'04*, vol. 2, pages 887–891. Cambridge, UK, August 2004.

[Prewitt70] J. Prewitt. Object enhancement and extraction. In *Picture Processing and Psychopictorics*, pages 75–149. Academic Press, 1970.

[Raudys91] S. J. Raudys and A. K. Jain. Small sample size effects in statistical pattern recognition: recommendations for practitioners. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, vol. 13(3), 252–264, March 1991. 0162-8828.

[Remazeilles04] A. Remazeilles, F. Chaumette and P. Gros. Robot Motion Control From a Visual Memory. In *International Conference on Robotics and Automation ICRA'04*, vol. 4, pages 4695–4700. La Nouvelle-Orléans, Louisiane, April 2004.

[Rice45] S. Rice. Mathematical analysis of random Noise. Tech. rep., Bell Laboratories, January 1945.

[Ricquebourg95] Y. Ricquebourg and P. Bouthemy. A statistical regularization framework for estimating normal displacements along contours with subpixel accuracy. In *6th Conference on Computer Analysis of Images and Patterns*, pages 73–81. Prague, Czech Republic, September 1995.

[Ricquebourg97] Y. Ricquebourg. *Analyse de mouvements articulés: mesure et suivi 2D; application à la télésurveillance*. Ph.D. thesis, Université de Rennes 1, Informatique, 1997.

[Ricquebourg00] Y. Ricquebourg and P. Bouthemy. Real-time tracking of moving persons by exploiting spatio-temporal image slices. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, vol. 22(8), 797, 2000.

[Rissanen87] J. Rissanen. Stochastic Complexity. *Journal of the Royal Statistical Society, Series B (Methodological)*, vol. 49(3), 223–239, 1987.

[Roberts65] L. Roberts. Machine perception of threedimensional solids. In *Optical and Electro-Optical Information Processing* (edited by J. Tippett). MIT Press, Cambridge, MA, 1965.

[Rodrigues40] M. O. Rodrigues. Des lois géométriques qui régissent les déplacement d'un systéme solide dans l'espace, et de la variation des coordonnées provenant de ces déplacement considérés indépendamment des causes qui peuvent les produire. *Journal de Mathématiques Pures et Appliquées*, vol. 5(Liouville), 380–440, 1840.

[Roetenberg03] D. Roetenberg, H. Luinge and P. Veltink. Inertial and magnetic sensing of human movement near ferromagnetic materials. In *International Symposium on Mixed and Augmented Reality*, page 268. Tokyo, Japan, 7-10 October 2003.

[Rolland95] J. Rolland, F. Biocca, T. Barlow and A. Kancherla. Quantification of Adaptation to Virtual-Eye Location in See-Thru Head-Mounted Displays. In *IEEE VRAIS*, pages 56–66. Research Triangle Park, NC, 11-15 March 1995.

[Rose95] E. Rose, D. Breen, K. Ahlers, C. Crampton, M. Tuceryan, R. Whitaker and D. Greer. Annotating Real- World Objects Using Augmented Reality. In *Computer Graphics International*, pages 357–370. Leeds, UK, 25-30 June 1995.

[Rosenberg77] R. Rosenberg. *Analytical Dynamics of Discrete Systems*. Plenum Press, New York, 1977.

[Rosenhahn05] B. Rosenhahn, C. Perwass and G. Sommer. Pose Estimation of 3D Free-Form Contours. *International Journal of Computer Vision (IJCV)*, vol. 62(3), 267–289, May 2005.

[Rosenthaler92] L. Rosenthaler, F. Heitger, O. Kubler and R. Von der Heydt. Detection of general edges and key-points. In *European Conference of Computer Vision II*, pages 78–86. Santa Margherita Ligure, Italy, May 1992.

[Rothganger05] F. Rothganger, S. Lazebnik, C. Schmid and J. Ponce. Object modeling and recognition using local affine-invariant image descriptors and multi-view spatial contraints. *International Journal of Computer Vision (IJCV)*, 2005.

[Rousseau02] F. Rousseau, P. Hellier and C. Barillot. A Fully Automatic Calibration Procedure for Freehand 3D Ultrasound. In *IEEE Symposium. on Biomedical Imaging*, pages 985–988. Washington D.C., July 2002.

[Rousseeuw84] P. Rousseeuw. Least Median of Squares Regression. *Amercain Statistical Association Journal*, vol. 79, 871–880, 1984.

[Rousseeuw87] P. Rousseeuw and A. Leroy. *Robust Regression and Outlier Detection*. John Wiley and Sons, New York, 1987.

[Royer04] E. Royer, M. Lhuiller, M. Dhome and T. Chateau. Towards an alternative gps sensor in dense urban environment from visual memory. In *British Machine Vision Conference, BMVC*, pages 197–206. London, UK, 7-9 September 2004.

[Ruf99a] A. Ruf and R. Horaud. Rigid and Articulated Motion Seen with an Uncalibrated Stereo Rig. In *IEEE International Conference on Computer Vision*, vol. 2, pages 789–796. Kerkira, Greece, September 1999.

[Ruf99b] A. Ruf and R. Horaud. Visual servoing of robot manipulators, part "i": Projective kinematics. *International Journal on Robotics Research*, vol. 18(11), 1101–1118, November 1999.

[Ruppert90] D. Ruppert and D. G. Simpson. Comment on Unmasking Multivariate Outliers and Leverage Points. *Journal of American Statistics Association*, vol. 85, 644–646, 1990.

[Safaee-Rad92] R. Safaee-Rad, I. Tchoukanov, B. Benhabib and K. Smith. 3D-pose estimation from a quadratic-curved feature in two perspective views. In *IAPR International Conference on Pattern Recognition, ICPR'92*, vol. 1, pages 341–344. La Haye, Pays Bas, August 1992.

[Sampson82] P. Sampson. Fitting conic sections to very scattered data: An iterative refinement of the Bookstein algorithm. *Computer Graphics and Image Processing*, vol. 18(1), 97–108, 1982.

[Samson88] C. Samson, B. Espiau and M. Le Borgne. Robot redundancy: An automatic control approach. In *NATO Advanced Research Workshop on Robots with Redundancy*. Salo, Italia, June 1988.

[Samson91] C. Samson, M. Le Borgne and B. Espiau. *Robot Control: the Task Function Approach*, vol. 22. Clarendon Press, Oxford, UK, 1991.

[Satoh04] K. Satoh, S. Uchiyama and H. Yamamoto. A Head Tracking Method Using Bird's-Eye View Camera and Gyroscope. In *International Symposium on Mixed and Augmented Reality*, pages 202–211. Arlington, VA, USA, 2-5 November 2004.

[Sattinger86] D. Sattinger and O. Weaver. *Lie groups and algebras with applications to physics, geometry, and mechanics*, vol. 61 of *Applied Mathematical Sciences*. Springer-Verlag, 1986.

[Schmid97] C. Schmid and R. Mohr. Local Greyvalue Invariants for Image Retrieval. *Pattern Analysis and Machine Intelligence*, vol. 19(5), 530–534, 1997.

[Schulz01] D. Schulz, W. Burgard, D. Fox and A. Cremers. Tracking multiple moving targets with a mobile robot using particle filters and statistical data association. In *IEEE International Conference on Robotics Automat*, pages 1665–1670. Seoul, Korea, 21-26 May 2001.

[Schunck86] B. Schunck. The image flow constraint equation. *Computer Vision, Graphics and Image Processing*, vol. 35(1), 20–46, July 1986. 16506.

[Schunck87] B. G. Schunck. Edge detection with Gaussian filters and multiple scales. In *IEEE Comper Society Workshop on Computer Vision*, pages 208–210. 30 November -2 December 1987.

[Shabana89] A. A. Shabana. *Dynamics of multibody systems*. Wiley, New York, 1989.

[Shapiro95] L. Shapiro and J. Brady. Rejecting outliers and estimating errors in an orthogonal regression framework. *Philosopical Transactions, Royal Society London*, vol. 350, 407–439, 1995.

[Shashua95] A. Shashua and M. Werman. On the trilinear tensor of three perspective views and its underlying geometry. In *International Conference on Computer Vision*, page 920. Boston, MA, 20-23 June 1995.

[Shen92] J. Shen and S. Castan. An Optimal Linear Operator for Step Edge Detection. *CVGIP: Graphical Models and Image Processing*, vol. 54(2), 112–133, March 1992.

[Shi94] J. Shi and C. Tomasi. Good Features to Track. In *IEEE International Conference on Computer Vision and Pattern Recognition, CVPR'94*, pages 593–600. Seattle, Washington, June 1994.

[Simon98] G. Simon and M. Berger. A Two-stage Robust Statistical Method for Temporal Registration from Features of Various Type. In *International Conference on Computer Vision*, pages 261–266. Bombay, India, January 1998.

[Simon99] G. Simon and M.-O. Berger. Registration with a Zoom Lens Camera for Augmented Reality Applications. In *ACM/IEEE International Workshop on Augmented Reality, IWAR'99*, pages 103–114. San Francisco, CA, October 1999.

[Simon02a] G. Simon and M.-O. Berger. Pose Estimation for Planar Structures. *IEEE Computer Graphics and Applications*, vol. 22(6), 46–53, November 2002.

[Simon02b] G. Simon, A. Fitzgibbon and A. Zisserman. Markerless Tracking using Planar Structures in the Scene. In *IEEE/ACM International Symposium. on Augmented Reality*, pages 120–128. Munich, Germany, October 2002.

[Sims94] D. Sims. New Realities in Aircraft Design and Manufacture. *IEEE Computer Graphics and Applications*, vol. 14(2), 91, March 1994.

[Smith94] C. Smith, S. Brandt and N. Papanikolopoulos. Controlled Active Exploration of Uncalibrated Environments. In *IEEE International Conference on Computer Vision and Pattern Recognition*, pages 792–795. Seattle, USA, June 1994.

[Sobel70] E. Sobel. Camera Models and Machine Perception. Tech. rep., Stanford University, 1970.

[Song91] S. Song and R. Leahy. Computation of 3D from 3D cine and CT images human imaging. *IEEE Transactions on Medical Imaging*, vol. 10(3), 295–306, September 1991.

[State94] A. State, C. D.T., C. Tector, A. Brandt, H. Chen, R. Ohbuchi, M. Bajura and H. Fuchs. Case Study: Observing a Volume Rendered Fetus within a Pregnant Patient. In *IEEE Visualization*, pages 364–368. Washington DC, 17-21 October 1994.

[State96] A. State, D. Hirota, C. D.T., B. Garret and M. Livingston. Superior Augmented Reality Registration by Integrating Landmark Tracking and Magnetic Tracking. In *SIGGRAPH*, pages 429–438. New Orleans, LA, 4-9 August 1996.

[Stewart99a] C. Stewart, K. Bubna and A. Perera. Estimating Model Parameters and Boundaries by Minimizing a Joint Robust Objective Function. In *IEEE Conference on Computer Vision and Pattern Recognition*, pages 387–393. Fort Collins, CO, 23-25 June 1999.

[Stewart99b] C.-V. Stewart. Robust parameter estimation in computer vision. *SIAM Review*, vol. 41(3), 513–537, September 1999.

[Stewart03] J. Stewart. *Calculus*. Thomson Brooks/Cole, Belmont, CA, 5th edn., 2003.

[Stramigioli00] S. Stramigioli, B. Maschke and C. Bidard. A Hamiltonian Formulation of the Dynamics of Spatial Mechanisms Using Lie Groups and Screw Theory. In *Proceedings of the Symposium Commemorating the Legacy, Works, and Life of Sir Robert Stawell Ball*. Cambridge, UK, 2000.

[Stramigioli02] S. Stramigioli, A. Schaft, B. Mashke and C. Melchiorri. Geometric Scattering in Robotic Telemanipulation. *IEEE Transactions of Robotics and Automation*, vol. 18(4), August 2002.

[Suh93] I. Suh and T. Kim. Visual Servoing of Robot Manipulators by Fuzzy Membership Function Based Neural Networks. In *Visual Servoing* (edited by K. Hashimoto), vol. 7, pages 285–315. World Scientific Series in Robotics and Automated Systems, Singapor, October 1993.

[Sundareswaran98] V. Sundareswaran and R. Behringer. Visual Servoing-based Augmented Reality. In *IEEE International Workshop on Augmented Reality*, pages 193–200. San Francisco, USA, 1 November 1998.

[Suter04] D. Suter and H. Wang. Robust Fitting Using Mean Shift: Applications in Computer Vision. In *Theory and Applications of Recent Robust Methods, Statistics for Industry and Technology* (edited by M. Hubert, G. Pison, A. Struyf and S. Van Aelst), pages 307–318. Birkhauser, Basel, 2004.

[Szczepanski58] W. Szczepanski. Die Lösungsvorschläge für den räumlichen Rückwärtseinschnitt. *Deutsche Geodätische Kommission*, vol. 29, 163–178, 1958.

[Szeliski93] R. Szeliski and R. Weiss. Robust Shape Recovery from Occluding Contours Using a Linear Smoother. In *IEEE Conference on Computer Vision and Pattern Recognition*, pages 666–667. New York, USA, June 1993.

[Tahri03] O. Tahri and F. Chaumette. Application of moment invariants to visual servoing. In *IEEE International Conference on Robotics and Automation, ICRA'03*, vol. 3, pages 4276–4281. Taipeh, Taiwan, May 2003.

[Tahri05] O. Tahri and F. Chaumette. Complex objects pose estimation based on image moment invariants. In *IEEE International Conference on Robotics and Automation, ICRA'05*, pages 438–443. Barcelona, Spain, April 2005.

[Tarel95] J.-P. Tarel and A. Gagalowicz. Calibration de Caméra á Base d'Ellipses. *Traitement du Signal*, vol. 12(2), 177–187, 1995.

[Taylor99] C.-J. Taylor, J.-P. Ostrowski and S.-H. Jung. Robust Visual Servoing based on Relative Orientation. In *IEEE International Conference on Computer Vision and Pattern Recognition*, pages 574–580. Collins, CO, June 1999.

[Taylor03] G. Taylor and L. Kleeman. Fusion of multimodal visual cues for model-based object tracking. In *Australasian Conference on Robotics and Automation (ACRA2003)*. Brisbane, 1-3 December 2003.

[Taylor04] G. Taylor and L. Kleeman. Stereoscopic Light Stripe Scanning: Interference Rejection, Error Minimization and Calibration. *International Journal Robotics Research*, vol. 23(12), 1141–1156, December 2004.

[Terzopoulos91] D. Terzopoulos and D. Metaxas. Dynamic 3D models with local and global deformations: deformable superquadrics. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, vol. 13(7), 703, 1991. 0162-8828.

[Thrun98] S. Thrun, D. Fox and W. Burgard. A probabilistic approach to concurrent mapping and localization for mobile robots. *Machine Learning*, vol. 31, 29–53, 1998.

[Tirumalai88] A. Tirumalai and B. Schunk. Robust surface approximation using least median of squares. Tech. Rep. CSE-TR-13-89, Artificial Intelligence Laboratory, University of Michigan, 1988.

[Tommasini98] T. Tommasini, A. Fusiello, E. Trucco and V. Roberto. Making Good Features Track Better. In *IEEE International Conference on Computer Vision and Pattern Recognition*, pages 178–183. Santa Barbara, USA, June 1998.

[Tonko97] M. Tonko, K. Schafer, F. Heimes and H.-H. Nagel. Towards Visually Servoed Manipulation of Car Engine Parts. In *IEEE International Conference on Robotics and Automation*, vol. 4, pages 3166–3171. Albuquerque/NM, April 1997.

[Tonko00] M. Tonko and H. Nagel. Model-Based Stereo-Tracking of Non-Polyhedral Objects for Automatic Disassembly Experiments. *International Journal of Computer Vision*, vol. 37(1), 99–118, June 2000.

[Torr93] P. Torr and D. Murray. Statistical Detection of Independent Mouvement from a Moving Camera. *Image Vision Computer*, vol. 11, 180–187, 1993.

[Torr97] P. Torr and D. Murray. The Development and Comparison of Robust Methods for Estimating the Fundamental Matrix. *International Journal of Computer Vision*, vol. 24(3), 271–300, 1997.

[Trucco98] E. Trucco and A. Verri. *Introductory Techniques for 3D Computer Vision*. Prentice-Hall, 1998.

[Tsai89] R. Tsai and R. Lenz. A New Technique for Fully Autonomous and Efficient 3D Robotics Hand/Eye Calibration. *IEEE Transactions on Robotics and Automation*, vol. 5(3), 345–358, June 1989.

[Tuceryan95] M. Tuceryan, D. S.G., R. Whitaker, D. Breen, C. Crampton, E. Rose and A. K.H. Calibration Requirements and Procedures for Augmented Reality. *IEEE Transactions on Visualization and Computer Graphics*, vol. 1(3), 255–273, September 1995.

[Uenohara95] M. Uenohara and T. Kanade. Vision-based object registration for real-time image overlay. In *1st International Conference on Computer Vision, Virtual Reality and Robotics in Medicine*, pages 13–22. Nice, France, April 1995.

[Vacchetti03] L. Vacchetti, V. Lepetit and P. Fua. Stable 3-D tracking in real-time using integrated context information. In *IEEE International Conference on Conference on Computer Vision and Pattern Recognition, CVPR'03*, vol. 2, pages 241–248. Madison, WI, June 2003.

[Vacchetti04a] L. Vacchetti, V. Lepetit and P. Fua. Combining edge and texture information for real-time accurate 3D camera tracking. In *International Symposium on Mixed and Augmented Reality*, pages 48 – 56. Arlington, VA, USA, 2-5 November 2004.

[Vacchetti04b] L. Vacchetti, V. Lepetit and P. Fua. Stable Real-Time 3D Tracking Using Online and Offline Information. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, vol. 26(10), 1385–1391, 2004.

[Vaillant92]	R. Vaillant and O. Faugeras. Using Extremal Boundaries for 3-D Object Modeling. *IEEE Transactions on PAMI*, vol. 14(2), 157–173, February 1992.
[Varadarajan74]	V. Varadarajan. *Lie Groups, Lie Algebras and Their Representations*, vol. 102 of *Graduate Texts in Mathematics*. Springer-Verlag, 1974.
[Venkatesh89]	S. Venkatesh and R. Owens. An Energy Feature Detection Scheme. In *IEEE International Conference on Image Processing*, pages 553–557. Singapor, September 1989.
[Venkatesh90]	S. Venkatesh. *A Study of Energy Based Models for the Detection and Classification of Image Features.* Phd. thesis, The University of Western Australia, 1990.
[Viirre98]	E. Viirre, H. Pryor, S. Nagata and T. A. Furness. The Virtual Retinal Display: A New Technology for Virtual Reality and Augmented Vision in Medicine. In *Proceedings of Medicine Meets Virtual Reality* (edited by D. Stredney and S. Weghorst), pages 252–257. IOS Press and Ohmsha, San Diego, CA, 1998.
[Vincze01]	M. Vincze. Robust Tracking of Ellipses at Frame Rate. *Pattern Recognition*, vol. 34(2), 487–498, February 2001.
[Viola95]	P. Viola and W. M. Wells. Alignment by maximization of mutual information. In *International Conference on Computer Vision*, page 16. IEEE Computer Society, Washington, DC, USA, 1995.
[Wallace87]	C. S. Wallace and P. R. Freeman. Estimation and Inference by Compact Coding. *Journal of the Royal Statistical Society - Series B (Methodological)*, vol. 49(3), 240–265, 1987.
[Wang04a]	H. Wang and D. Suter. MDPE: A Very Robust Estimator for Model Fitting and Range Image Segmentation. *International Journal of Computer Vision (IJCV)*, vol. 59(2), 139–166, September 2004.
[Wang04b]	H. Wang and D. Suter. Robust Adaptive-Scale Parametric Model Estimation for Computer Vision. *IEEE Transactions Pattern Analysis and Machine Intelligence*, vol. 26(11), 1459–1479, November 2004.
[Webster96]	A. Webster, S. Feiner, B. MacIntyre, W. Massie and T. Krueger. Augmented reality in architectural construction, inspection, and renovation. *Computing in Civil Engineering*, pages 913–919, 1996.
[Wei94]	G.-Q. Wei and S. Ma. Implicit and Explicit Camera Calibration: Theory and Experiments. *IEEE Transactions on Pattern Analysis and Machine intelligence*, vol. 16(5), 469–480, May 1994.
[Weiss87]	L. Weiss, A. Sanderson and C. Neuman. Dynamic Sensor-Based Control of Robots with Visual Feedback. *IEEE Journal of Robotics and Automation*, vol. 3(5), 404–417, October 1987.
[Wells96]	G. Wells and C. Venaille. Vision-based robot positioning using neural networks. *Image and Vision Computing*, vol. 14(10), 715–732, 1996.

[Weng89] J. Weng, T. Huang and N. Ahuja. Motion and structure from two perspective views: algorithms, error analysis, and error estimation. *IEEE Transactions on Pattern Analysis and Machine intelligence*, vol. 11(5), 451–476, May 1989.

[Weng92] J. Weng, P. Cohen and M. Herniou. Camera Calibration with Distorsion Models and Accuracy Evaluation. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, vol. 14(10), 965–980, October 1992.

[Wilson96] W. Wilson, C. Hulls and G. Bell. Relative end-effector control using cartesian position-based visual servoing. *IEEE Transactions on Robotics and Automation*, vol. 12(5), 684–696, October 1996.

[Witkin83] A. Witkin. Scale-space filtering. In *International Joint Conference on Artificial Intelligence*, vol. 2, pages 1019–1022. Karlsruhe, Germany, August 1983.

[Wolf02] J. Wolf, W. Burgard and H. Burkhardt. Robust vision-based localization for mobile robots using an image retrieval system based on invariant features. In *IEEE International Conference on Robotics and Automation*. Washington, USA, May 2002.

[Woolard03] A. Woolard, V. Laliodati, N. Hedley, N. Carrigan, M. Hammond and J. Julien. Case studies in application of augmented reality in future media production. In *International Symposium on Mixed and Augmented Reality*, pages 294–295. Tokyo, Japan, 7-10 October 2003.

[Wu89] J. Wu, R. Rink, T. Caelli and V. Gourishankar. Recovery of the 3-D Location and Motion of a Rigid Object Through Camera Image (an Extended Kalman Filter Approach). *International Journal of Computer Vision*, vol. 2(4), 373–394, 1989.

[Wu03] Y. Wu, G. Hua and T. Yu. Tracking articulated body by dynamic Markov network. In *Nineth IEEE International Conference on Computer Vision*, vol. 2, pages 1094–1101. Nice, France, 13-16 October 2003.

[Wunsch97] P. Wunsch and G. Hirzinger. Real-Time Visual Tracking of 3-D Objects with Dynamic Handling of Occlusion. In *International Conference on Robotics and Automation*, pages 2868–2873. Albuquerque, NM, April 1997.

[Xie89] M. Xie. *Contribution à la vision dynamique: reconstruction d'objets 3D polyédriques par une caméra mobile*. Phdthesis, Université de Rennes 1, IRISA, 1989. Thesis.

[Xie95] Z. Xie. *Multi-scale Analysis and Texture Segmentation.* D. phil., Oxford University, 1995.

[Yesin04] K. Yesin and B. Nelson. A CAD-Model Based Tracking System for Visually Guided Microassembly. *Robotica*, 2004.

[Zerroug95] M. Zerroug and R. Nevatia. Pose Estimation of Multi-Part Curved Objects. In *International Symposium on Computer Vision*, pages 431–436. Florida, USA, 21-23 November 1995.

[Zhang95] Z. Zhang, R. Deriche, O. Faugeras and Q. Luong. A robust technique for matching two uncalibrated images through the recovery of the unknown epipolar geometry. *Artificial Intelligence*, vol. 78, 87–119, October 1995.

[Zhang97] Z. Zhang. Parameter estimation techniques: a tutorial with application to conic fitting. *Image and Vision Computing*, vol. 15(1), 59–76, January 1997.

[Zhang00] Z. Zhang. A Flexible New Technique for Camera Calibration. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, vol. 22(11), 1330–1334, November 2000.

[Zhang02] X. Zhang, S. Fronz and N. Navab. Visual Marker Detection and Decoding in AR Systems: A Comparative Study. In *IEEE International Symposium. on Mixed and Augmented Reality (ISMAR'02)*, pages 79–106. Darmstadt, Germany, September 2002.

[Zhao94] C. Zhao and R. Mohr. Relative 3D regularized B-Splines surface reconstruction through image sequences. In *3rd European Conference on Computer Vision, ECCV'94*, vol. 2, pages 417–426. Stockholm, Suède, May 1994.

[Zhou01] L. Zhou, C. Kambhamettu, D. Goldgof, K. Palaniappan and A. Hasler. Tracking nonrigid motion and structure from 2D satellite cloud images without correspondences. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, vol. 23(11), 1330–1336, November 2001.

Summary

This thesis presents a real-time 3D tracking method using an *a priori* CAD model of the environment and a monocular vision sensor. Both rigid and non-rigid object models are considered. 3D tracking is essential for many applications and in this thesis it is applied to both Augmented Reality (AR) and Visual Servoing (VS). The tracking is based on the estimation of the pose between the camera and its environment and is estimated via a non-linear relation between visual features extracted from the image and the projection of the 3D object onto the image. The estimation is considered as synonymous to a Virtual Visual Servoing control law.

Local object features are tracked by an efficient Moving Edges (ME) technique. An adaptation of this tracker is proposed to make it invariant to changes in scale of the contour and so that no predetermined threshold is needed. These features, combined with the 3D model, are then related to the pose by point-to-contour type interaction matrices.

In order to take into account a erroneous data a robust closed loop control law is proposed. It is based on the robust statistical technique of M-estimation which allows to consider a general class of errors including : noise in feature extraction, small tracking errors and large correspondence errors.

In a last part, the approach is extended to a class of non-rigid objects in 3D. More particularly, different types of mechanical joint are defined within the CAD model. The configuration of an articulated object is therefore obtained from a set of generalized parameters which represent both the pose of all the components of the object as well as the relative poses between the different components. In order to derive the minimal parameter representation a Kinematic Set approach is proposed. Kinematic subsets a derived in a symmetric manner using velocity constraints associated with the mechanical joints. The subsets are then combined to form the articulation matrix which defines the minimal representation.

Experimental results are presented on several indoor and outdoor sequences. They show the efficiency of the proposed 3D tracking method even with severe outlier contamination including occlusion, changes in illumination and miss-tracking. In the case of articulated motion tracking, experiments have validated the approach for prismatic, rotational and helical type joints.

Key-words : 3D visual tracking, articulated non-rigid motion, real-time, model-based, augmented reality, visual servoing, robust control.