Chapter 20
Visual Servoing via Nonlinear Predictive Control

Guillaume Allibert, Estelle Courtial, and François Chaumette

Abstract In this chapter, image-based visual servoing is addressed via nonlinear model predictive control. The visual servoing task is formulated into a nonlinear optimization problem in the image plane. The proposed approach, named visual predictive control, can easily and explicitly take into account 2D and 3D constraints. Furthermore, the image prediction over a finite prediction horizon plays a crucial role for large displacements. This image prediction is obtained thanks to a model. The choice of this model is discussed. A nonlinear global model and a local model based on the interaction matrix are considered. Advantages and drawbacks of both models are pointed out. Finally, simulations obtained with a 6 degrees of freedom (DOF) free-flying camera highlight the capabilities and the efficiency of the proposed approach by a comparison with the classical image-based visual servoing.

20.1 Introduction

Visual servoing has lead to many fruitful researches over the last decades. In regard to the kind of feedback information considered, one can distinguish three main approaches: image-based visual servoing (IBVS) where the feedback is defined in the image plane, position-based visual servoing (PBVS) where the feedback is composed of 3D data such as the robotic system pose, and the 2-1/2D visual servoing where the feedback combines both 2D and 3D data. Further details about the different approaches can be found in [5, 6, 15]. Here, we focus our interest on IBVS strategy. The IBVS task consists in determining the control input applied to the robotic system so that a set of visual features designed from image measurements

Guillaume Allibert and Estelle Courtial
Institut PRISME, Polytech'Orleans, 8 rue Leonard de Vinci, 45072 Orleans, France, e-mail: {guillaume.allibert,estelle.courtial}@univ-orleans.fr

François Chaumette
INRIA, Campus de Beaulieu, 35042 Rennes, France, e-mail: francois.chaumette@irisa.fr

reaches a desired static reference or follows a desired dynamic reference. Although IBVS approach is robust to modeling errors, several drawbacks can be mentioned when the visual features are not correctly chosen. Besides the classical problem of local minima and singularities in the interaction matrix [4], the constraint handling is a tricky problem in IBVS. For instance, the 2D constraint, also named visibility constraint, has to guarantee that the image measurements stay into the camera field of view. Of course, if the visibility of the target is no longer ensured then the control algorithm is interrupted. The 3D constraints such as workspace limits have to make sure that the robot achieves admissible motions in its workspace all along the task.

Among the numerous works which have investigated this critical issue, three points of view exist. The first one consists in designing adequate visual features. In [17] for instance, the authors have shown that the system behavior explicitly depends on the kind of features. Consequently, lines, spheres, circles, cylinders but also moments may be used and combined to obtain good decoupling and linearizing properties, implicitly ensuring the constraint handling. The control law is generally a decoupled exponential decreasing law. Another way to deal with constraint handling is to combine path-planning and trajectory tracking [7, 16, 19, 24]. When it is successful, this solution allows ensuring both an optimal trajectory of the camera in the Cartesian space and the visibility of the features. Path-planning via linear matrix inequality (LMI) optimization has recently been proposed in [7] to fulfill 2D and 3D constraints. In the third approach, the effort is done on the control law design. The visual features considered are generally basic, namely point-like features. Advanced control laws such as optimal control [14, 22], adaptive control [21], LMIs [9, 10] and predictive control [2, 3, 12, 13, 23] have been reported in the literature. In [12, 13], a predictive controller is used for motion compensation in target tracking applications. The prediction of the target motion is used to reject perturbation in order to cancel tracking errors. In [23], the predictive controller is used from ultrasound images for a medical application.

The strategy proposed in this chapter exploits nonlinear model predictive control for visual servoing tasks. The IBVS objective is formulated as solving on-line a nonlinear optimization problem expressed in the image plane [2, 3]. This strategy, named visual predictive control (VPC), offers two advantages. First, 2D and 3D constraints such as visibility constraints, mechanical constraints and workspace limits can be easily taken into account in the optimization problem. Secondly, the image prediction over a finite horizon plays a crucial role for difficult configurations. The image prediction is based on the knowledge of a model. It can be a nonlinear global model combining the robot model and the camera one. The image prediction can also be obtained thanks to a linear model using the interaction matrix. The choice of the model is addressed and discussed in the sequel. The interest of the image prediction is pointed out through many simulations describing difficult configurations for a free-flying perspective camera.

The chapter is organized as follows. In Section 20.2, the context of the study is stated and the principle of VPC is presented. The control structure and the mathematical formulation are addressed. Then, in Section 20.3, the choice of the image prediction model is discussed. In Section 20.4, numerous simulations on a 6 DOF free-flying camera illustrate the comparison of the different approaches: classical IBVS, predictive control laws with local and global model. Difficult configurations such as large displacements to achieve are tested under constraints. Finally, conclusions are given in the last section.

20.2 Predictive Control for Constrained IBVS

The aim of visual servoing is to regulate to zero an error $e(t)$ between the current features $s(t)$ and the reference features $s^\star$. In IBVS, the features are expressed in the image. The relationship between the camera velocity $\tau(t)$ and the time variation of the visual features $\dot{s}(t)$ is given by the interaction matrix noted L_s. Thus, specifying a decoupled exponential decay law for the error $e(t)$, we obtain the control input to be applied to the camera:

$$\tau(t) = -\lambda \widehat{L_s}^{+} e(t) \ \text{ with } \ \lambda > 0, \tag{20.1}$$

where $\widehat{L_s}^{+}$ is the approximate pseudo-inverse matrix of L_s. The classical IBVS is very easy to implement but its weak points are the constraint handling and its possible bad behavior for large displacements to achieve as already mentioned in Section 20.1. The control objective of IBVS can also be formulated into an optimization problem. The goal is to minimize an image error and to take into account constraints. When a model of the system is available, control predictive strategies are well-adapted to deal with this kind of problem. The extension of predictive strategy to visual servoing tasks is detailed below.

20.2.1 Visual Predictive Control

All predictive strategies are based on four common points: a reference trajectory, a model of the dynamic process, a cost function and a solving optimization method. The keystone of the predictive approach is the model used to predict the process behavior over the future horizon. Its choice will impact on the tracking accuracy and on the computational time. In VPC case, the process considered is generally composed of the robotic system and the camera. For instance, the robotic system can be a nonholonomic mobile robot [2], a drone or a robot arm. The camera system can be a perspective or catadioptric camera [3] whatever its configuration with respect to the robot, that is on board or remote. The model used is then a global model describing the process. The model inputs are the control variables of the robotic system. The model output are the visual features. The model is used to predict the values of the features over a prediction horizon in regard to the control variables and to satisfy the constraint handling. Before discussing the choice of the model, we first introduce the control structure and then state the mathematical formulation of VPC.

20.2.2 Internal Model Control Structure

The control structure considered is the well-known internal model control (IMC) structure [20] (see Fig. 20.1). The process block contains the robotic system and the camera. The input U is the robotic control variable and the output s is the current value of the visual features. For IBVS, the reference s^* is expressed in the image plane, as the visual features, and can be static or dynamic. The error signal ε represents all modeling errors and disturbances between the current features and the values predicted from the model of the system:

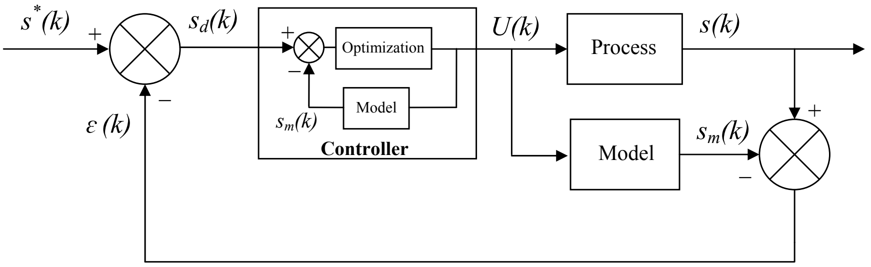

Fig. 20.1 IMC Structure.

$$\varepsilon(k) = s(k) - s_m(k). \tag{20.2}$$

The usual controller is replaced, in the predictive approach, by an optimization algorithm. The latter minimizes the difference between a desired trajectory s_d and the predicted model output s_m. Indeed, according to Fig. 20.1, we can write (k is the current iteration):

$$\begin{aligned} s_d(k) &= s^*(k) - \varepsilon(k), \\ s_d(k) &= s^*(k) - (s(k) - s_m(k)), \\ s_d(k) - s_m(k) &= s^*(k) - s(k). \end{aligned} \tag{20.3}$$

Consequently, the tracking of the reference features s^* by the process output s is thus equivalent to the tracking of the desired features s_d by the model output s_m. The model predicts the behavior of the features over a finite prediction horizon N_p. The difference $s_d(k) - s_m(k)$ between the desired features and the predicted model features is used to define the cost function J to be minimized with respect to a control sequence $\tilde{U}$. Only the first component $U(k)$ of the optimal control sequence is really applied to the process. At the next sampling time, due to disturbances and model mismatches, the measurements are updated, the finite horizon moves one step forward and the procedure starts again.

20.2.3 Mathematical Formulation

The cost function J is defined as a quadratic function of the error to be minimized. Due to the IMC structure, the mathematical formulation of VPC strategy can be written in discrete-time as:

$$\min_{\widetilde{U}\in\mathbb{R}^{m\times Np}} J(U) \tag{20.4}$$

with:

$$J(U) = \sum_{j=k+1}^{k+Np} [s_d(j) - s_m(j)]^T \; Q(j) \; [s_d(j) - s_m(j)] \tag{20.5}$$

subject to:

$$s_d(j) = s^*(j) - \varepsilon(j), \tag{20.6}$$

$$\begin{cases} x(j) = f(x(j-1), U(j-1)) \\ s_m(j) = h(x(j)). \end{cases} \tag{20.7}$$

The variables $x \in \mathbb{R}^n$, $U \in \mathbb{R}^m$ and $s_m \in \mathbb{R}^p$ are respectively the state, the input and the output of the model. We will see, in the next section, that the state can be differently chosen in regard to the prediction model used and in regard to the constraints to be handled. The first nonlinear equation of (20.7) describes the dynamics of the system where $x(j)$ represents the predicted state at time j, $\forall j \in [k+1; k+N_p]$. For $j = k+1$, the predicted state s_m is initialized with the system state s at time k which guarantees the feedback of the IMC structure. Moreover, in case of modeling errors and disturbances, a second feedback is ensured by the error signal $\varepsilon(j)$ which modifies the reference trajectory accordingly. The second equation of (20.7) is the output equation. To compute $s_d(j)$, $\forall j \in [k+1; k+N_p]$, we need to compute the error $\varepsilon(j)$ defined in (20.2). This error depends on $s_m(j)$ that is available but also on $s(j)$ that is unknown over the prediction horizon. Consequently, the error $\varepsilon(j)$ is assumed to be constant over the prediction horizon:

$$\varepsilon(j) = \varepsilon(k) = s(k) - s_m(k), \;\; \forall j \in [k+1; k+N_p]. \tag{20.8}$$

Finally, $\tilde{U} = \{U(k), U(k+1), ..., U(k+N_c), ..., U(k+N_p-1)\}$ is the optimal control sequence. From $U(k+N_c+1)$ to $U(k+N_p-1)$, the control input is constant and equal to $U(k+N_c)$ where N_c is the control horizon. The weighted matrix $Q(j)$ is a symmetric definite positive matrix.

One of the main advantages of VPC is the capability to explicitly handle constraints in the optimization problem. Three kinds of constraints are distinguished:

- constraints on the state of the robotic system. It can typically be a mechanical constraint such as workspace limit when the state represents the camera pose for instance,
$$x_{\min} \leq x(k) \leq x_{\max}; \tag{20.9}$$
- 2D constraints also named visibility constraints to ensure that the visual features stay in the image plane or to represent forbidden areas in the image. The latter

can be very useful to deal with obstacle avoidance or image occlusion,

$$s_{\min} \leq s_m(k) \leq s_{\max}; \tag{20.10}$$

- control constraints such as actuator limitations in amplitude or velocity,

$$U_{\min} \leq U(k) \leq U_{\max}. \tag{20.11}$$

These constraints are added to the problem (20.4) which becomes a nonlinear constrained optimization problem:

$$\min_{\widetilde{U} \in \mathbb{K}} J(U) \tag{20.12}$$

where $\mathbb{K}$ is the constraint domain defined by:

$$\begin{cases} C(U) \leq 0 \\ Ceq(U) = 0. \end{cases} \tag{20.13}$$

The constraints (20.9), (20.10) and (20.11) can be formulated by nonlinear functions $C(U)$ and $Ceq(U)$ [8]. Numerous constrained optimization routines are available in software libraries to solve this kind of problem: projected gradient methods, penalty methods, etc. In our case, a sequential quadratic program (SQP) is used and more precisely, the function *fmincon* from Matlab optimization toolbox.

The setting parameters of the predictive approach are the prediction horizon (N_p), the control horizon (N_c) and the weighted matrix ($Q(j)$):

- the prediction horizon is chosen in order to satisfy a compromise between scheme stability (long horizon) and numerical feasibility in term of computational time requirement (short horizon);
- the control input is usually kept constant over the prediction horizon, which corresponds to a control horizon equal to 1. A $N_c > 1$ can be useful for stabilization task of nonholonomic mobile robot for instance [1];
- the matrix $Q(j)$ is often the identity matrix but it can also be a time-varying matrix useful for stabilizing the system. If $Q(k+1) = I$ and $Q(k+l) = 0\ \forall l \in [2; N_p]$, the cost function J is then similar to the standard criterion of IBVS. It is also equivalent to have a prediction horizon equal to 1.

20.3 Model of Image Prediction

Here we focus on the model used to predict the image evolution. We consider a 6 DOF free-flying perspective camera observing fixed point features. A 3D point with coordinates $P = (X, Y, Z)$ in the camera frame is projected in the image plane as a 2D point with coordinates $s = (u, v)$. The sampling period is T_e and the control input U is the camera velocity noted $\tau = (T_x, T_y, T_z, W_x, W_y, W_z)$.

The role of the model is to predict, over the horizon N_p, the evolution of the visual features in regard to the camera velocity. The principle of the image prediction is depicted in Fig. 20.2. To perform this image prediction, two kinds of model can be considered: a nonlinear global model and a local model based on the interaction matrix. The identification of the model, described above by (20.7), is discussed with respect to both cases in the next section.

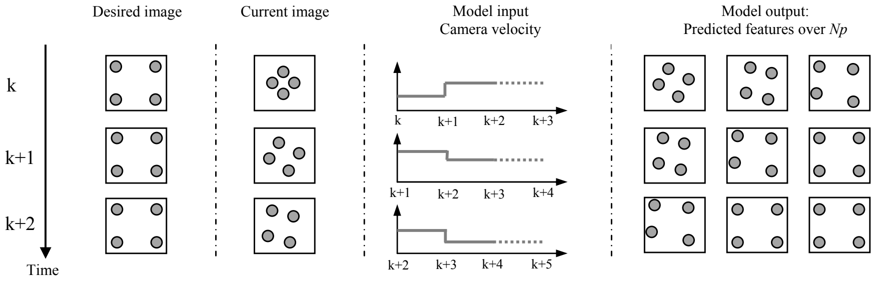

Fig. 20.2 Principle of image prediction ($N_p = 3, N_c = 2$).

20.3.1 Nonlinear Global Model

The control input of the free-flying process is the camera velocity τ applied to the camera. Here, the state of the system can be the camera pose in the target frame: $x = (P_x, P_y, P_z, \Theta_x, \Theta_y, \Theta_z)$. The dynamic equation can be approximated by [1]:

$$x(k+1) = x(k) + Te\ \tau(k) = f(x(k), \tau(k)). \tag{20.14}$$

The output is the visual features expressed in the image plane noted s_m. In the case of a perspective camera, the output equation for one point-like feature in normalized coordinates can be written as:

$$s_m(k) = \begin{pmatrix} u(k) \\ v(k) \end{pmatrix} = \begin{pmatrix} X(k)/Z(k) \\ Y(k)/Z(k) \end{pmatrix} = g(X(k), Y(k), Z(k)), \tag{20.15}$$

where $(X,Y,Z,1)_{R_c}$ are the point coordinates in the camera frame. The rigid transformation between the camera frame and the target frame, noted $l(x)$, can easily be deduced from the knowledge of the camera pose $x(k)$. If the point coordinates are known in the target frame, $(X,Y,Z,1)_{R_t}$, then the point coordinates in the camera frame, $(X,Y,Z,1)_{R_c}$ are given by:

[1] The exponential map could be also used to better describe the camera motion.

$$\begin{pmatrix} X \\ Y \\ Z \\ 1 \end{pmatrix}_{R_c} = \begin{pmatrix} R(x) & T(x) \\ 0_{1\times 3} & 1 \end{pmatrix} \begin{pmatrix} X \\ Y \\ Z \\ 1 \end{pmatrix}_{R_t} = l(x(k)). \tag{20.16}$$

Finally, we obtain:

$$s_m(k) = g \circ l(x(k)) = h(x(k)). \tag{20.17}$$

Equations 20.7 are now completely identified with (20.14) and (20.17). This dynamic model combines 2D and 3D data and so it is appropriate to deal with 2D and/or 3D constraints. The constraints are respectively expressed on the states and/or the outputs of the prediction model and are easily added to the optimization problem. The nonlinear global model has a large validity domain and thus it can be used for large displacements. Nevertheless, the prediction over the prediction horizon can be time consuming. Moreover, this model requires 3D data that are the pose of the target in the initial camera frame, as well as the target model. To reduce the 3D knowledge, a solution can be the linearization of the model based on the interaction matrix.

20.3.2 Local Model Based on the Interaction Matrix

For a point-like feature s expressed in normalized coordinates such that $u = X/Z$ and $v = Y/Z$, the interaction matrix related to s is given by [5]:

$$L_s = \begin{bmatrix} -\frac{1}{Z} & 0 & \frac{u}{Z} & uv & -(1+u^2) & v \\ 0 & -\frac{1}{Z} & \frac{v}{Z} & 1+v^2 & -uv & -u \end{bmatrix}. \tag{20.18}$$

The value Z is the depth of the 3D point expressed in the camera frame. The relationship between the camera velocity τ and the time variation of the visual features $\dot{s}$ is given by:

$$\dot{s}(t) = L_s(t)\tau(t). \tag{20.19}$$

In [11], this dynamic equation is solved to reconstruct the image data in case of occlusion. Here, with a simple first order approximation, we obtain:

$$s(k+1) = s(k) + T_e \, L_s(k)\tau(k). \tag{20.20}$$

To avoid the estimation of the depth parameter at each iteration, its value Z^* given or measured at the reference position can be used. Consequently, the interaction matrix (20.18) becomes $\widehat{L_s}$ and depends only on the current measure of the visual features. By considering here the visual features s as the state x, we obtain the set of equations describing the process dynamics and outputs (20.7):

$$\begin{cases} x(k+1) = x(k) + T_e \, \widehat{L_s}(k)\tau(k) \; = \; f(x(k), \tau(k)) \\ s_m(k) = x(k) \; = \; h(x(k)). \end{cases} \tag{20.21}$$

This approximated local model does not require 3D data but only the approximate value of Z^*. 2D constraints can be taken into account since the model states and outputs are the visual features. On the other hand, no information is available on the camera pose and so 3D constraints can not be directly handled. For doing that, as for the nonlinear model, it would be necessary to reconstruct the initial camera pose by using the knowledge of the 3D model target. That is of course easily possible but has not been considered in this chapter. Finally, for large displacements, a problem can be mentioned as we will see on simulations: the linear and depth approximations may be too coarse and can lead to control law failures.

20.4 Simulation Results

For all presented simulations, the sampling time T_e is equal to 40 ms. This choice allows considering real-time application with an usual camera (25 fps). The control task consists in positioning a perspective free-flying camera with respect to a target composed of four points. These four points form a square of 20 cm in length in Cartesian space. The reference image is obtained when the target pose expressed in the camera frame (R_C) is equal to $P_{T/C} = (0,0,0.5,0,0,0)$ (see Fig. 20.3), where the first three components are the translation expressed in meters and the last three components are the roll, pitch and yaw angles expressed in radians. The coordinates of the four points in the reference image are: $s^* = (u_{d1}, v_{d1}, u_{d2}, v_{d2}, u_{d3}, v_{d3}, u_{d4}, v_{d4}) = (-0.2, 0.2, 0.2, 0.2, 0.2, -0.2, -0.2, -0.2)$ (see Fig. 20.4).

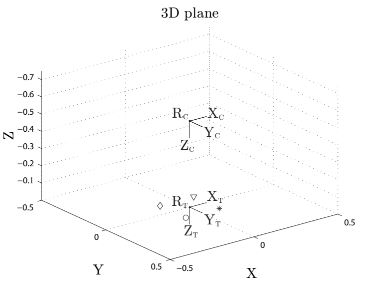

Fig. 20.3 3D desired posture.

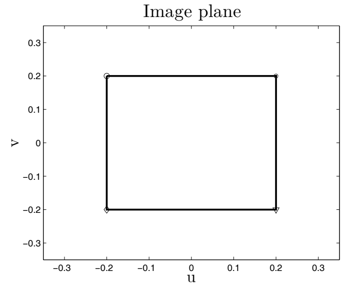

Fig. 20.4 2D reference image.

Different simulations illustrate the performance of the VPC strategy. Besides the choice of the model used to predict the point positions, VPC requires to set three parameters, the prediction horizon N_p, the control horizon N_c and the weighted matrix $Q(j)$:

- the choice of the prediction horizon is crucial. The system behavior and the convergence speed depend on the prediction horizon. The value of N_p is discussed below;

- the control horizon is kept constant and equal to 1 ($N_c = 1$). Only one control is calculated over the prediction horizon;
- the weighted matrix $Q(j)$ is either the identity matrix $Q(j) = I_{8\times 8}\ \forall j$, constant over the prediction horizon, or a time-varying matrix $Q(j) = 2Q(j-1)$ with $Q(1) = I_{8\times 8}$. In this last case, this matrix weights the error at each sampling instant more and more over the prediction horizon and so, stresses the error at the end of the horizon N_p which corresponds to the final objective. In stabilization task, this time-varying matrix can be compared to the terminal constraint used in the classical predictive control strategy. However it is less restrictive for the optimization algorithm. In the sequel, the time-varying matrix is noted Q_{TV}.

The VPC simulation results are compared with the classical IBVS approaches based on the exponential control law (20.1) where $\widehat{L_s}$ can be chosen as:

- $\widehat{L_s}$=$\widehat{L}_{(s(t),Z(t))}$, noted L_c: the interaction matrix is updated at each iteration;
- $\widehat{L_s}$=$\widehat{L}_{(s(t),Z^*)}$, noted L_p: the depth computed or measured at the reference position noted Z^* is used. The interaction matrix varies only through the current measure of the visual features;
- $\widehat{L_s}$=$\widehat{L}_{(s^*,Z^*)}$, noted L_d: the interaction matrix is constant and corresponds to its value at the reference position;
- $\widehat{L_s}$=$\frac{1}{2}(\widehat{L}_{(s^*,Z^*)} + \widehat{L}_{(s(t),Z(t))})$, noted L_m: the interaction matrix is the mean of the constant and current interaction matrices.

In order to compare the VPC approach with the classical IBVS, no constraint on the control input is considered in the first part. Then, mechanical and visibility constraints will be taken in consideration with the VPC approach. In all cases, the control inputs are normalized if needed. The bounds are 0.25 m/s for the translation speed and 0.25 rad/s for the rotation speed.

20.4.1 Case 1: Pure Rotation around the Optical Axis

In case 1, the required camera motion is a pure rotation of $\frac{\pi}{2}$ radians around the optical axis. Due to the lack of space and since it is a classical case, all simulation results are not presented here but all are discussed.

20.4.1.1 Classical IBVS

For the classical IBVS, the following results are obtained:

- with L_c, the trajectories in the image plane are pure straight lines as expected [5]. The camera motion is then a combination of a backward translation and a rotation with respect to the camera optical axis (retreat problem). Due to this undesired retreat, the camera might reach the limit of the workspace;

- with L_p, the results are approximatively the results obtained in the first case (retreat problem), see Fig. 20.5. The visual feature trajectories tend to straight lines;
- with L_d, the camera moves toward the target and simultaneously rotates around the optical axis (advance problem) [5]. Due to this undesired forward motion, some features can go out the camera field of view during the camera motion;
- with L_m, the camera motion is a pure rotation [18]. No additional motion is induced along the optical axis and the visual feature trajectories are circular.

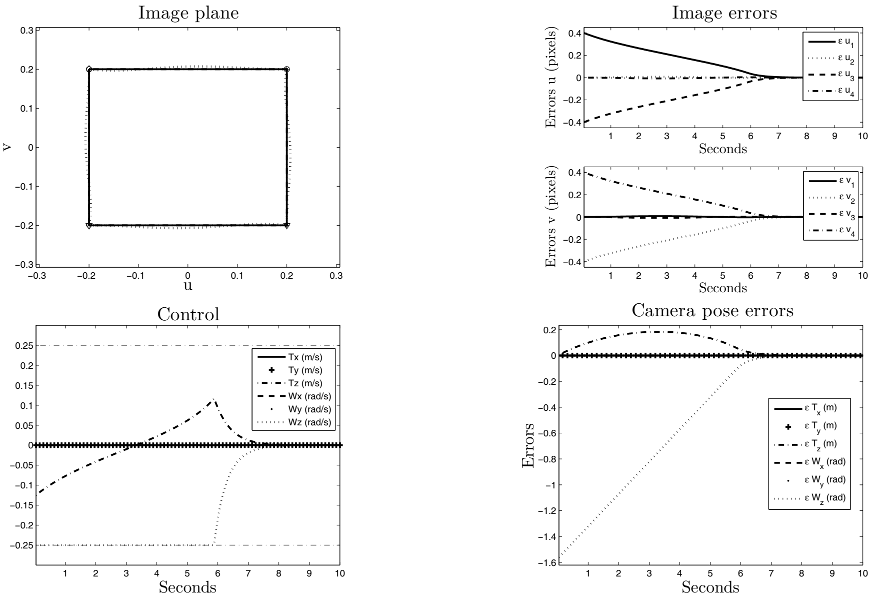

Fig. 20.5 Case 1: Classical IBVS with L_p.

20.4.1.2 VPC with a Local Model ($\mathrm{VPC_{LM}}$)

The following simulations are obtained with the VPC strategy using a local model based on the interaction matrix L_p. The comparison with the classical IBVS is done for different N_p values ($N_p = 1, 10, 20$) and different weighted matrices ($Q(j) = I$ or Q_{TV}). For $N_p = 1$, the results are similar to the classical IBVS with L_p since the model used to predict the next image is exactly the same (see Fig. 20.6). The only difference is the behavior of the control law, decreasing exponentially with IBVS. For $N_p = 10$ (see Fig. 20.7) or $N_p = 20$ (see Fig. 20.8), the trajectories in the image plane become circular. Indeed, the only constant control over N_p which minimizes the cost function is a pure rotation. Thus the translation motion along the optical axis decreases with the increase of N_p value.

The time-varying matrix Q_{TV} accentuates the decoupling control by giving importance at the end of N_p which corresponds to the final objective (see Fig. 20.9). It seems to be equivalent to the behavior obtained with L_m which takes into account the desired position. For a π rotation around the optical axis, the classical IBVS with L_c, L_p or L_d fails as well as $\mathrm{VPC_{LM}}$ with $Q(j) = I$ and whatever N_p. On the other hand, $\mathrm{VPC_{LM}}$ achieves the satisfying motion with Q_{TV} and $N_p \geq 20$ (see Fig. 20.10).

To illustrate the capability of visibility constraint handling, the visual features are constrained to stay in a window defined by the following inequalities:

$$\begin{bmatrix} \mathrm{u_{min=}} - 0.22 \\ \mathrm{v_{min=}} - 0.22 \end{bmatrix} \leq \mathrm{s_m}(j) \leq \begin{bmatrix} \mathrm{u_{max}} = 0.22 \\ \mathrm{v_{max}} = 0.22 \end{bmatrix}. \qquad (20.22)$$

In that case, $\mathrm{VPC_{LM}}$ satisfies both visibility constraint and control task (see Fig. 20.11). A translation along the optical axis is then induced to ensure that the visual features do not get out the camera field of view.

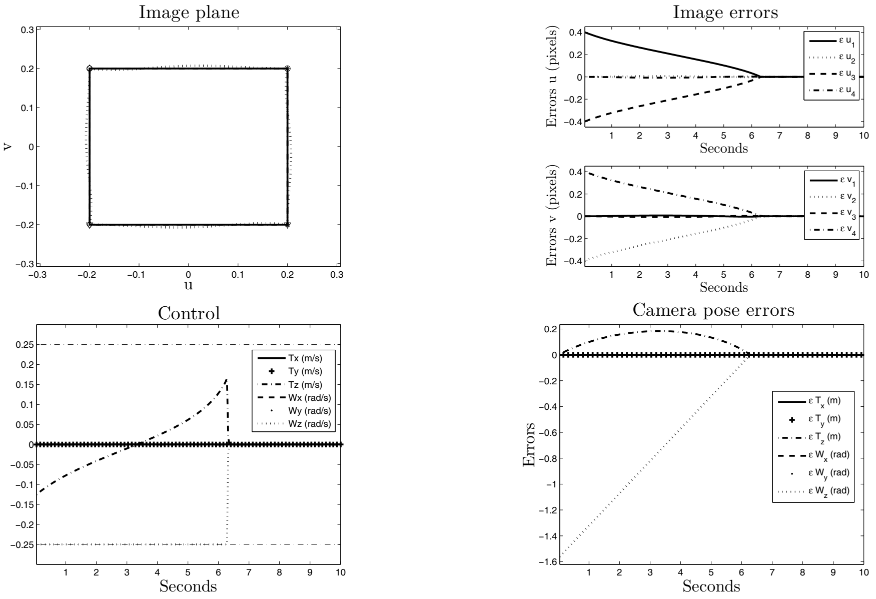

Fig. 20.6 Case 1: $\mathrm{VPC_{LM}}$ with $N_p = 1$, $Q(j) = I$.

20.4.1.3 VPC with a Nonlinear Global Model ($\mathrm{VPC_{GM}}$)

The previous results obtained with $\mathrm{VPC_{LM}}$ are improved with $\mathrm{VPC_{GM}}$ since no linearization is done. For instance, with $\mathrm{VPC_{GM}}$ and $N_p = 1$, the image plane tra-

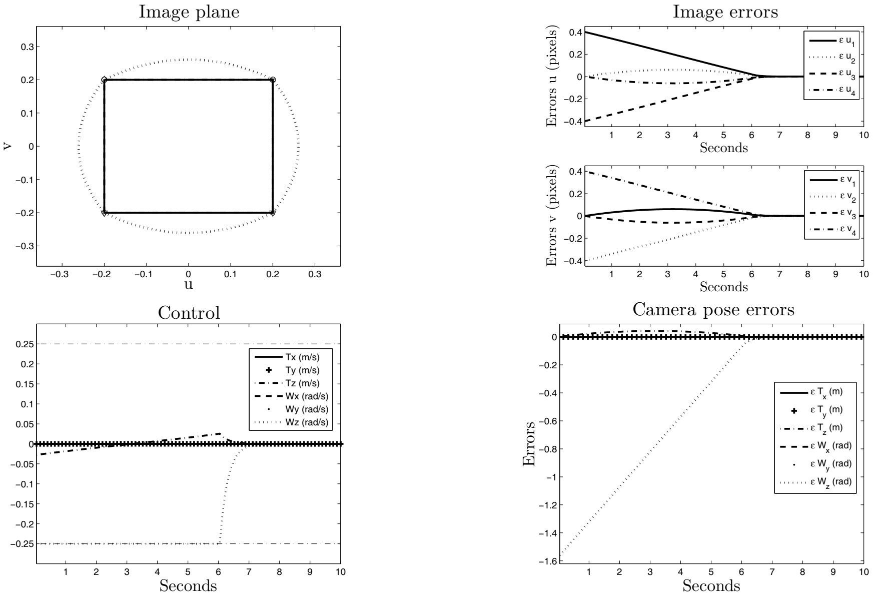

Fig. 20.7 Case 1: $\mathrm{VPC_{LM}}$ with $N_p = 10$, $Q(j) = I$.

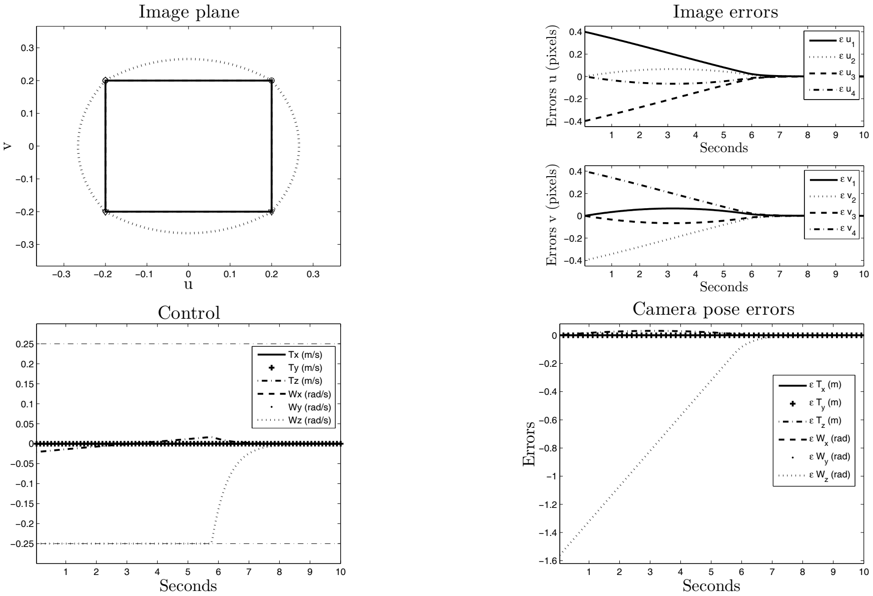

Fig. 20.8 Case 1: $\mathrm{VPC_{LM}}$ with $N_p = 20$, $Q(j) = I$.

jectories are circular as the ones obtained with L_m. Here, we focus on 3D constraint handling. Added to the visibility constraint, we limit the camera workspace along

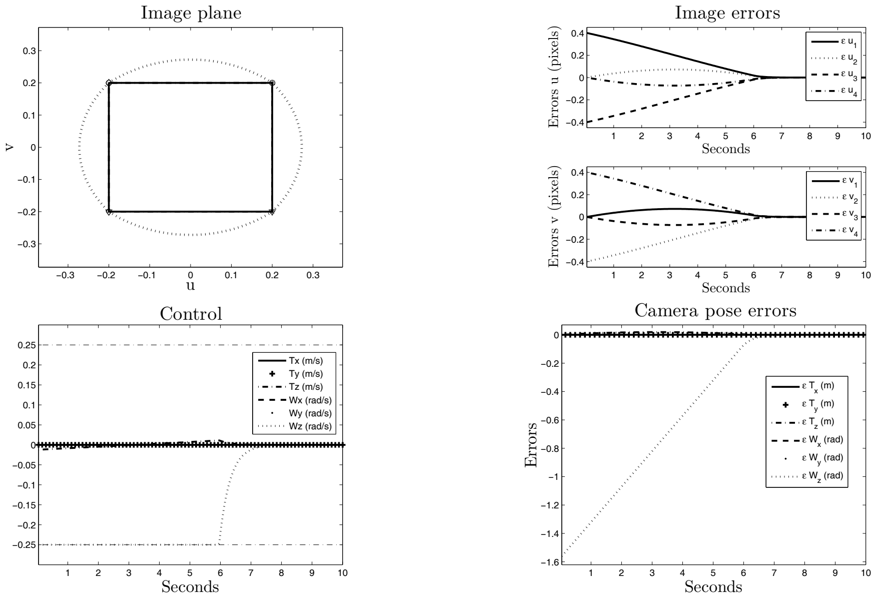
Image plane
Image errors
Control
Camera pose errors

Fig. 20.9 Case 1: VPC_{LM} with $N_p = 10$, $Q(j) = Q_{TV}$.

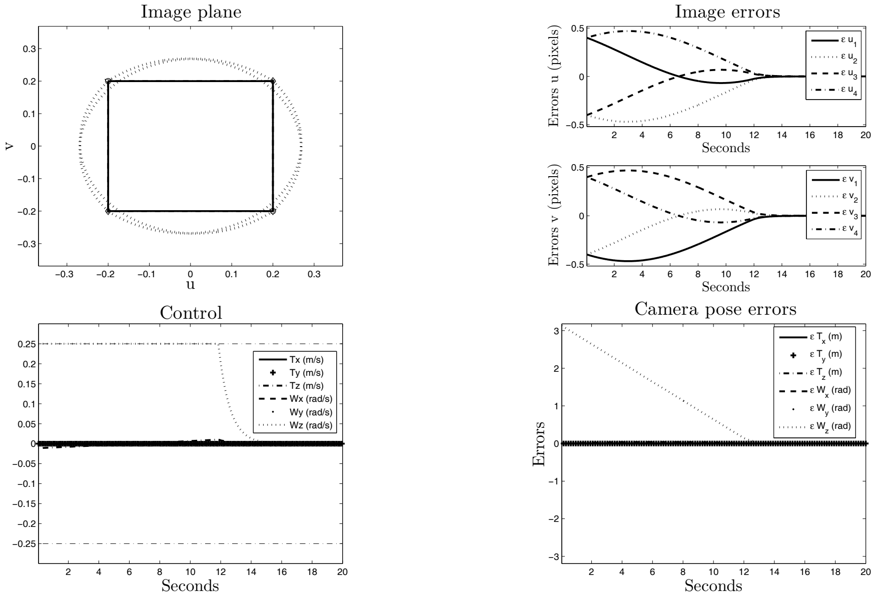
Image plane
Image errors
Control
Camera pose errors

Fig. 20.10 π rotation around optical axis: VPC_{LM} with $N_p = 20$, $Q(j) = Q_{TV}$.

the Z_c axis. As can be seen on Fig. 20.12, VPC_{GM} converges under both constraints by using the other camera DOF. If no admissible trajectory ensuring visibility con-

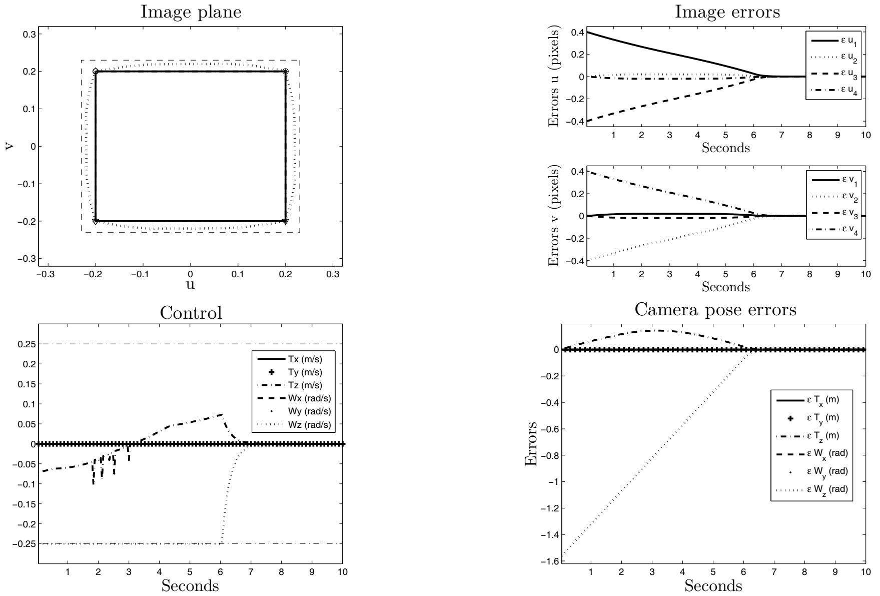

Fig. 20.11 Case 1: VPC$_{\text{LM}}$ with $N_p = 10$, $Q(j) = I$ and visibility constraint.

straints and 3D constraints (such as $-0.05 < X_c < 0.05$, $-0.05 < Y_c < 0.05$, $Z_c < 0.6m$) exists, VPC$_{\text{GM}}$ stops at the position minimizing the constrained cost function (see Fig. 20.13).

20.4.2 Case 2: Large Displacement

In case 2, the initial target pose expressed in the camera frame is given by $P_{T/C} = (0.04, 0.03, 0.35, 1, -0.98, -0.43)$. The classical IBVS with L_p or L_d does not converge for such a large displacement. Indeed, during the motion, the camera reaches the object plane where $Z = 0$. The same result is obtained by the VPC$_{\text{LM}}$ whatever N_p due to the too coarse approximations. However, the convergence is obtained with the weighted time-varying matrix Q_{TV}. VPC$_{\text{GM}}$ always converges even if $Q(j) = I$ (see Fig. 20.15). The trajectories in the image plane are very similar to the ones obtained with the classical IBVS with L_m (see Fig. 20.14). These good results are still kept even if visibility constraints ($-0.29 < u < 0.29$, $-0.4 < v < 0.4$) are considered (see Fig. 20.16).

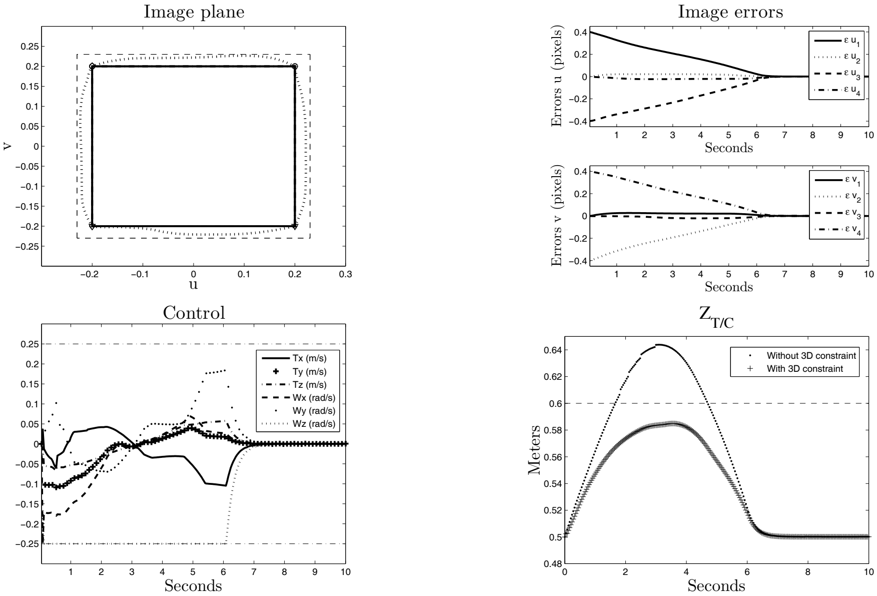

Fig. 20.12 Case 1: VPC$_{\mathrm{GM}}$ with $N_p = 10$, $Q(j) = I$, visibility constraint and $Z_{T/C} < 0.6$ m.

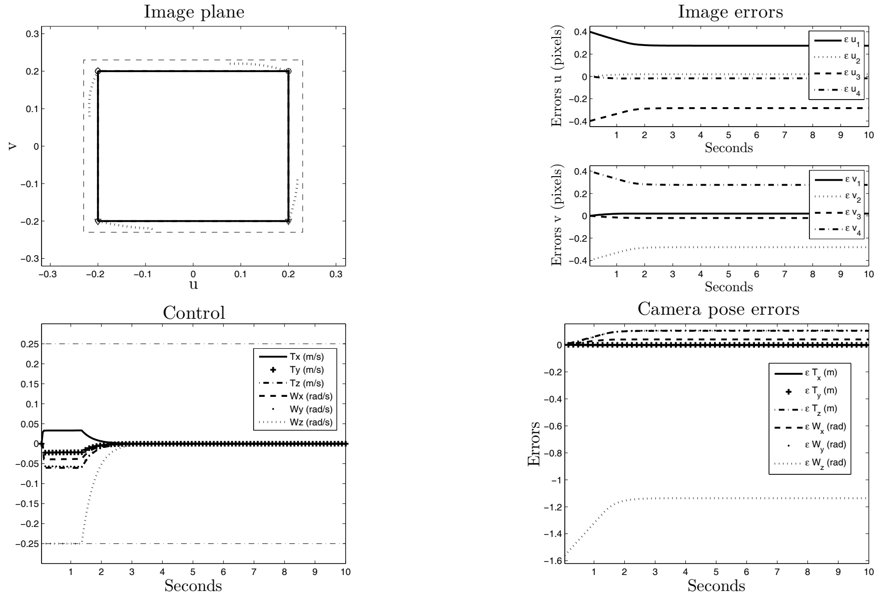

Fig. 20.13 Case 1: VPC$_{\mathrm{GM}}$ with $N_p = 10$, $Q(j) = I$, visibility and strong 3D constraints such that there is no solution to the optimization problem.

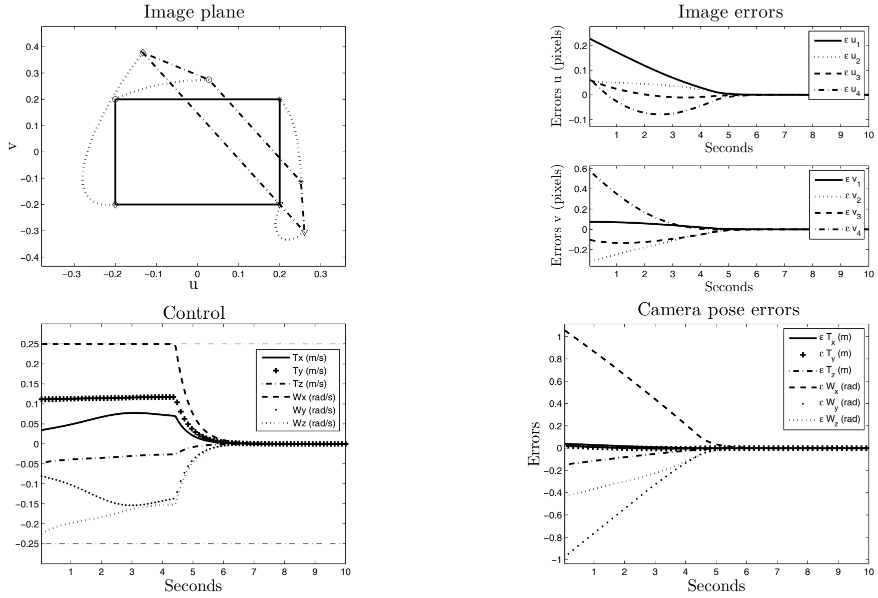

Fig. 20.14 Case 2: Classical IBVS with L_m.

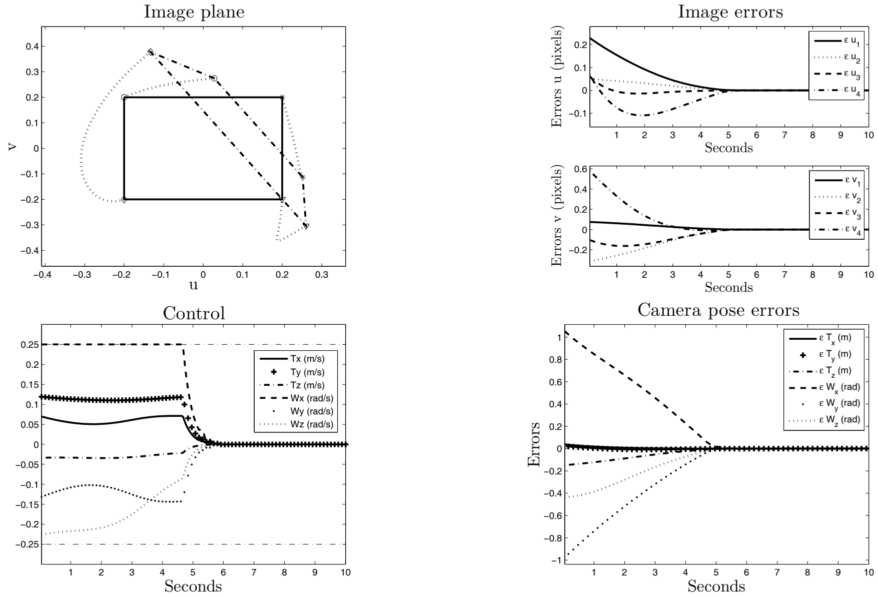

Fig. 20.15 Case 2: VPC$_{\text{GM}}$ with $N_p = 10$, $Q(j) = I$.

20.5 Conclusions

In this chapter, we have shown that an alternative approach of IBVS can be the VPC strategy. The visual servoing task is then formulated into a nonlinear optimiza-

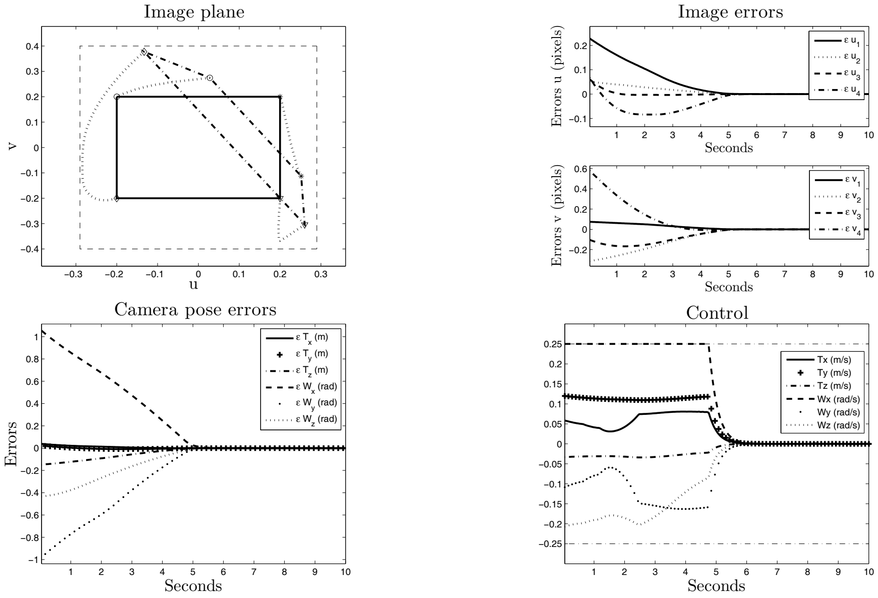

Fig. 20.16 Case 2: VPC$_{\mathrm{GM}}$ with $N_p = 10$, $Q(j) = I$ and visibility constraint.

tion problem over a prediction horizon. The advantage of this formulation is the capability of easily dealing with visibility constraints and 3D constraints. The optimization procedure can be compared to an on-line implicit and optimal constrained path-planning of features in the image plane. The choice of the image prediction model has been discussed. The approximated local model can be less efficient than the global model for difficult configurations but no 3D data are required. On the other hand, if 3D data are available, VPC$_{\mathrm{GM}}$ gives satisfying results for any initial configuration and motion to achieve. The VPC setting parameters, i.e., the prediction horizon and the weighted matrix, play a crucial role in terms of camera and visual feature trajectories. Simulation results highlight the efficiency of VPC. Finally, this strategy is very flexible and can be used whatever the robotic system (mobile robot or robot arm) and the camera (perspective or catadioptric).

References

[1] Allibert G, Courtial E, Touré Y (2006) Visual predictive control. In: IFAC Workshop on Nonlinear Predictive Control for Fast Systems, Grenoble, France

[2] Allibert G, Courtial E, Touré Y (2008) Real-time visual predictive controller for image-based trajectory tracking of mobile robot. In: 17th IFAC World Congress, Seoul, Korea

[3] Allibert G, Courtial E, Touré Y (2008) Visual predictive control for manipu-

lators with catadioptric camera. In: IEEE Int. Conf. on Robotics and Automation, Pasadena, USA

[4] Chaumette F (1998) Potential problems of stability and convergence in image-based and position-based visual servoing. In: Kriegman D, Hager G, Morse A (eds) The Confluence of Vision and Control, LNCIS Series, No 237, Springer-Verlag, pp 66–78

[5] Chaumette F, Hutchinson S (2006) Visual servo control, part i: Basic approaches. IEEE Robotics and Automation Magazine 13(4):82–90

[6] Chaumette F, Hutchinson S (2007) Visual servo control, part ii: Advanced approaches. IEEE Robotics and Automation Magazine 14(1):109–118

[7] Chesi G (2009) Visual servoing path planning via homogeneous forms and lmi optimizations. IEEE Trans Robotics and Automation 25(2):281–291

[8] Chong E, Zak S H (2001) An Introduction to Optimization. John Wiley & Sons Inc, 2nd Edition

[9] Danès P, Bellot D (2006) Towards an lmi approach to multicriteria visual servoing in robotics. European Journal of Control 12(1):86–110

[10] Durola S, Danès P, Coutinho D, Courdesses M (2009) Rational systems and matrix inequalities to the multicriteria analysis of visual servos. In: IEEE Int. Conf. on Robotics and Automation, Kobe, Japan

[11] Folio D, Cadenat V (2008) Dealing with visual features loss during a vision-based task for a mobile robot. International Journal of Optomechatronics 2(3):185–204

[12] Gangloff J, De Mathelin M (2002) Visual servoing of a 6 dof manipulator for unknown 3-d profile following. IEEE Trans Robotics and Automation 18(4):511–520

[13] Ginhoux R, Gangloff J, De Mathelin M, Soler M, Sanchez L (2005) Active filtering of physiological motion in robotized surgery using predictive control. IEEE Trans Robotics and Automation 21(1):67–79

[14] Hashimoto K, Kimura H (1993) Visual Servoing, vol 7, K. Hashimoto, Ed. (Robotics and Automated Systems). Singapore:World Scientific, chap LQ optimal and nonlinear approaches to visual servoing, pp 165–198

[15] Hutchinson S, Hager GD, Corke P (October 1996) A tutorial on visual servo control. IEEE Trans Robotics and Automation 12(5):651–671

[16] Kazemi M, Gupta, K, Mehrandezh M (2009) Global path planning for robust visual servoing in complex environments. In: IEEE Int. Conf. on Robotics and Automation, Kobe, Japan

[17] Mahony R, Corke P, Chaumette, F (2002) Choice of image features for depth-axis control in image-based visual servo control. In: IEEE/RSJ Int. Conf. on Intelligent RObots and Systems, Lausanne, Switzerland

[18] Malis E (2004) Improving vision-based control using efficient second-order minimization techniques. In: IEEE Int. Conf. on Robotics and Automation, New Orleans, LA, USA

[19] Mezouar Y, Chaumette F (2003) Optimal camera trajectory with image-based control. Int Journal of Robotics Research 22(10):781–804

[20] Morari M, Zafiriou E (1983) Robust Control. Dunod

[21] Nasisi O, Carelli R (2003) Adaptive servo visual robot control. Robotics and Autonomous Systems 43(1):51–78

[22] Papanikolopoulos N, Khosla P, Kanade T (February 1993) Visual tracking of a moving target by a camera mounted on a robot: A combination of vision and control. IEEE Trans Robotics and Automation 9(1):14–35

[23] Sauvée M, Poignet P, Dombre E, Courtial E (2006) Image based visual servoing through nonlinear model predictive control. In: 45th IEEE CDC, San Diego, CA, pp 1776–1781

[24] Schramm F, Morel G (2006) Ensuring visibility in calibration-free path planning for image-based visual servoing. IEEE Trans Robotics and Automation 22(4):848–854