# Pose Error Correction For Visual Features Prediction

Nicolas Cazy, Claire Dune, Pierre-Brice Wieber, Paolo Robuffo Giordano, and François Chaumette

***Abstract*— Predicting the behavior of visual features on the image plane over a future time horizon is an important possibility in many different control problems. For example when dealing with occlusions (or other constraints such as joint limits) in a classical visual servoing loop, or also in the more advanced model predictive control schemes recently proposed in the literature. Several possibilities have been proposed to perform the initial correction step for then propagating the visual features by exploiting the measurements currently available by the camera. But the predictions proposed so far are inaccurate in situations where the depths of the tracked points are not correctly estimated. We then propose in this paper a new correction strategy which tries to directly correct the relative pose between the camera and the target instead of only adjusting the error on the image plane. This correction is then analysed and compared by evaluating the corresponding improvements in the feature prediction phase.**

## I. Introduction

A key element in visual servoing is the *prediction* of the visual signal that is going to be observed in the future. This prediction over a long horizon can be very useful in various cases. For instance it can be exploited for computing a standard visual servoing control when one or more features leave the camera field of view during the camera motion [1]. Visual features prediction can also be used in the context of model predictive control with vision feedback, which has been recently proposed for robotic systems, mostly six degrees of freedom manipulators with eye-in-hand configuration [2]–[5] (the camera is mounted on the manipulator), or with eye-to-hand configuration [2] (the camera is looking at the manipulator from a remote pose). Applications to mobile robotics have also been considered [6], [7], [8].

Nonlinear predictive models have been proposed to take into account the whole dynamics of the system, the 6D pose of the camera and the projection model [3]. But to overcome the numerical complexity of such nonlinear models, linear approximations have usually been adopted, e.g., exploiting the relation between the velocity of the camera and the velocity of the visual features in the image [4], [7]. Comparisons of these different options and of their performance concluded that nonlinear models significantly improve the convergence of the control laws if 3D information is available, at the cost of increased numerical complexity [8]. However, we will show in this paper that this conclusion is somehow debatable since the nonlinear and linear models actually yield the same results when the model of the camera/target relative pose is correct enough.

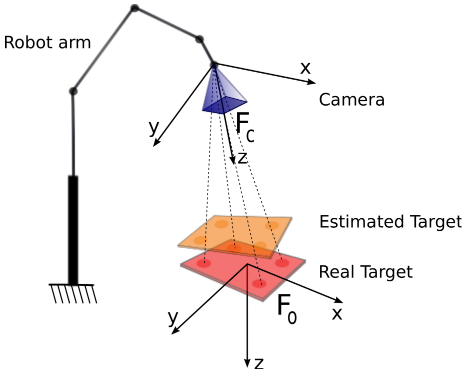


Fig. 1: Robot arm carrying an eye-in-hand camera looking at a target object made of $4$ 3D points

It is then clear that a crucial step for a successful prediction is the *initial model correction* using the measured visual features. A correction based on a first-order linearization of the visual servoing problem is usually considered. Yet, visual servoing is a nonlinear problem, mainly due to the projection on the image plane and the six dimensional motion of the camera. It is then of great importance to develop more sophisticated correction methods able to cope with these nonlinearities. The goal of this paper is then to propose, analyze and critically compare different linear and nonlinear strategies for implementing the *model correction* step. Section II starts reviewing the theoretical framework of the paper and then proposes three different correction methods. Section III then reviews (and discusses) the two strategies for the prediction phase typically considered in the literature, and Sect. IV presents a number of simulation results meant to critically compare the pros/cons of the various methods. Finally, Sect. V concludes the paper and discusses some future directions.

## II. Visual Features Model Error Correction

We consider in this paper a monocular perspective camera mounted on a robot arm, and a target consisting of $N$ 3D points $\boldsymbol{P}_i$ (see Fig 1). Let $\mathcal{F}_C$ and $\mathcal{F}_O$ be Cartesian frames attached respectively to the camera and target. Let $\mathbf{x} = [x \ y \ z \ \theta\boldsymbol{u}^T]^T \in \mathbb{R}^6$ be the pose of $\mathcal{F}_O$ w.r.t. $\mathcal{F}_C$,

N. Cazy is with Inria Rennes Bretagne Atlantique and Irisa, Campus de Beaulieu, 35042 Rennes Cedex, France nicolas.cazy@inria.fr

C. Dune is with the HandiBio laboratory EA 4322 at Université du Sud Toulon-Var La Garde, France claire.dune@univ-tln.fr

P.-B. Wieber is with the Bipop team at Inria Grenoble Rhône-Alpes, France pierre-brice.wieber@inria.fr

P. Robuffo Giordano is with the CNRS at Irisa and Inria Rennes Bretagne Atlantique, Campus de Beaulieu, 35042 Rennes Cedex, France prg@irisa.fr.

F. Chaumette is with Inria Rennes Bretagne Atlantique and Irisa, Campus de Beaulieu, 35042 Rennes Cedex, France francois.chaumette@irisa.fr

with $(\boldsymbol{u},\, \theta) \in \mathbb{S}^2 \times \mathbb{R}$ the axis/angle parameterization of the orientation[1].

Let ${}^{O}\boldsymbol{P}_i = \begin{bmatrix} {}^{O}X_i & {}^{O}Y_i & {}^{O}Z_i & 1 \end{bmatrix}^T$ and ${}^{C}\boldsymbol{P}_i = \begin{bmatrix} {}^{C}X_i & {}^{C}Y_i & {}^{C}Z_i & 1 \end{bmatrix}^T$ be the homogeneous coordinates of the 3D points $\boldsymbol{P}_i$ in the object and camera frames. Assuming a calibrated camera, the normalized perspective 2-D projection of each point $\boldsymbol{P}_i$ on the image plane is

$$\boldsymbol{s}_i = \begin{bmatrix} u_i \\ v_i \end{bmatrix} = \begin{bmatrix} {}^{C}X_i/{}^{C}Z_i \\ {}^{C}Y_i/{}^{C}Z_i \end{bmatrix} = f_{s_i}(\mathbf{x}). \tag{1}$$

The set of all visual features is therefore

$$\boldsymbol{s} = \begin{bmatrix} s_1 \\ \vdots \\ s_N \end{bmatrix} = f_s(\mathbf{x}) \in \mathbb{R}^{2N}. \tag{2}$$

Let $\mathbf{x}_m(t)$ be a model/approximation of the target pose and $\boldsymbol{s}_m(t) = f_s(\mathbf{x}_m(t))$ be the corresponding model of $\boldsymbol{s}(t)$ derived from the internal model $\mathbf{x}_m(t)$ via (2). Since we are assuming a calibrated camera, any error in $\mathbf{x}_m(t)$ can arise from a wrong pose estimation process due to image noise, or to a coarse/incomplete knowledge of the 3D model of the target. The idea pursued in this work is to then exploit the error between the *measured* $\boldsymbol{s}(t_0)$ and *modeled* $\boldsymbol{s}_m(t_0)$ so as to correct the internal model at $t_0$ and, as a consequence, mimimize the future discrepancies between the predicted $\boldsymbol{s}_m(t)$ and real $\boldsymbol{s}(t)$ for $t \geq t_0$.

We now present and discuss several strategies for implementing this correction.

### A. Correction in the image

The method proposed in [8] does not change the model of the pose $\mathbf{x}_m(t_0)$, but it directly affects the model of the visual features. It allows correcting 3D coordinates $X$ and $Y$, but no correction is applied on $Z$. More precisely, we have

$$\begin{cases} \mathbf{x}_{c1}(t_0) &= \mathbf{x}_m(t_0) \\ \boldsymbol{s}_{c1}(t_0) &= \boldsymbol{s}(t_0) \\ {}^{C}X_{c1}(t_0) &= u_{c1}(t_0)\,{}^{C}Z_{c1}(t_0) \\ {}^{C}Y_{c1}(t_0) &= v_{c1}(t_0)\,{}^{C}Z_{c1}(t_0) \\ {}^{C}Z_{c1}(t_0) &= {}^{C}Z_m(t_0) \end{cases} \tag{3}$$

This correction clearly ensures $\boldsymbol{s}_{c1}(t) \approx \boldsymbol{s}(t)$ for $t \approx t_0$, but fails in general in reducing any divergence among $\boldsymbol{s}_{c1}(t)$ and $\boldsymbol{s}(t)$ for $t > t_0$ in case of a non-pure rotational motion (since $\boldsymbol{s}_{c1}(t)$ does not depend on the possibly wrong depths $Z$). This is obviously because (3) only accounts for the error in the image but does not attempt to also correct for the underlying discrepancy between the real pose $\mathbf{x}(t_0)$ and the modeled one $\mathbf{x}_m(t_0)$. Sect. IV will report some examples in this sense.

### B. Correction in the pose

We propose in this paper to exploit the mismatch between the model $\boldsymbol{s}_m(t_0)$ and the measure $\boldsymbol{s}(t_0)$ to correct the pose $\mathbf{x}_m(t_0)$.With $\mathbf{v} = (\boldsymbol{v},\, \boldsymbol{\omega}) \in \mathbb{R}^6$ being the linear and angular velocity of the camera expressed in $\mathcal{F}_C$, standard kinematics gives

$$\dot{\mathbf{x}} = \boldsymbol{L}_x(\mathbf{x})\mathbf{v}, \qquad \boldsymbol{L}_x \in \mathbb{R}^{6\times 6}, \tag{4}$$

and standard visual feature models give [10]

$$\dot{\boldsymbol{s}} = \boldsymbol{L}_s(\boldsymbol{s},\, \mathbf{x})\mathbf{v}, \qquad \boldsymbol{L}_s \in \mathbb{R}^{2N\times 6} \tag{5}$$

directly leading to

$$\dot{\boldsymbol{s}} = \boldsymbol{L}_s \boldsymbol{L}_x^{-1} \dot{\mathbf{x}}. \tag{6}$$

Explicit expressions for both $\boldsymbol{L}_x$ and $\boldsymbol{L}_s$ are given in the Appendix. We can obtain similarly a first-order Taylor expansion

$$s_{c2}(t_0) - \boldsymbol{s}_m(t_0) \approx \boldsymbol{L}_s(\boldsymbol{s}_m(t_0),\, \mathbf{x}_m(t_0))\boldsymbol{L}_x^{-1}(\mathbf{x}_m(t_0))\Delta\mathbf{x}. \tag{7}$$

The following least-squares problem

$$\min_{\Delta\mathbf{x}} \|\boldsymbol{s}_{c2}(t_0) - \boldsymbol{s}(t_0)\|^2 \tag{8}$$

can then be reformulated, exploiting (7), as

$$\min_{\Delta\mathbf{x}} \|\boldsymbol{L}_s \boldsymbol{L}_x^{-1}\Delta\mathbf{x} + \boldsymbol{s}_m(t_0) - \boldsymbol{s}(t_0)\|^2 \tag{9}$$

resulting in

$$\Delta\mathbf{x} = (\boldsymbol{L}_s \boldsymbol{L}_x^{-1})^{\dagger}(\boldsymbol{s}(t_0) - \boldsymbol{s}_m(t_0)) = \boldsymbol{L}_x \boldsymbol{L}_s^{\dagger}(\boldsymbol{s}(t_0) - \boldsymbol{s}_m(t_0)) \tag{10}$$

with $\boldsymbol{L}_s^{\dagger}$ being the Moore-Penrose pseudoinverse of $\boldsymbol{L}_s$.

The increment $\Delta\mathbf{x}$ can then be used to implement the pose correction, which allows to compute visual features and 3D coordinates $X$, $Y$ and $Z$ subsequently used for prediction.

$$\begin{cases} \mathbf{x}_{c2}(t_0) &= \mathbf{x}_m(t_0) + \Delta\mathbf{x} \\ \boldsymbol{s}_{c2}(t_0) &= f_s(\mathbf{x}_{c2}(t_0)) \\ {}^{C}X_{c2}(t_0) &= f_s(\mathbf{x}_{c2}(t_0)) \\ {}^{C}Y_{c2}(t_0) &= f_s(\mathbf{x}_{c2}(t_0)) \\ {}^{C}Z_{c2}(t_0) &= f_s(\mathbf{x}_{c2}(t_0)) \end{cases} \tag{11}$$

Contrary to (3), the correction (11) adjusts the target pose by exploiting the measured $\boldsymbol{s}(t_0)$ but does not guarantee that $\boldsymbol{s}_{c2}(t_0) = \boldsymbol{s}(t_0)$ as (10) only yields a least squares solution to (8). However, as shown in Sect. IV, by applying (11) one can nevertheless obtain better results in future predictions compared to the simpler (3) thanks to the more accurate corrective action of (10) when $\mathbf{x}_m(t_0)$ is particularly incorrect. In fact, the pose correction scheme proposed above corresponds to applying only one iteration of a pose estimation by virtual visual servoing [11]. Iterating could be possible but would be more time consuming. Note finally that in (10), matrices $\boldsymbol{L}_x$ and $\boldsymbol{L}_s$ are evaluated on the modeled $\boldsymbol{s}_m(t_0)$ and $\mathbf{x}_m(t_0)$ and thus represent approximated quantities. However, since a measurement $\boldsymbol{s}(t_0)$ is available, another possibility (considered in the rest of the paper) is to replace $\boldsymbol{s}_m(t_0)$ with

[1] Among all the possible *minimal* representations for the orientation between $\mathcal{F}_O$ in $\mathcal{F}_C$, we chose vector $\theta\boldsymbol{u} \in \mathbb{R}^3$ because its singularities lie at $\theta = 2k\pi$, $k \in \mathbb{Z}^*$, i.e., out of the possible workspace in any normal application (see [9]).

$\boldsymbol{s}(t_0)$ in evaluating $\boldsymbol{L}_s(\boldsymbol{s}(t_0),\, \mathbf{x}_m(t_0))$ in order to reduce the approximation level.

Finally, it is interesting to note the following fact: by left-multiplying (7)–(10) with $\boldsymbol{L}_s^\dagger$, and exploiting the property of the Moore-Penrose pseudoinverse

$$\boldsymbol{L}_s^\dagger \boldsymbol{L}_s \boldsymbol{L}_s^\dagger = \boldsymbol{L}_s^\dagger, \tag{12}$$

one has

$$\boldsymbol{L}_s^\dagger \boldsymbol{s}_{c2}(t_0) \approx \boldsymbol{L}_s^\dagger \boldsymbol{s}(t_0). \tag{13}$$

Therefore, a typical visual servoing control law

$$\mathbf{v} = -\lambda \boldsymbol{L}_s^\dagger(\boldsymbol{s}(t_0) - \boldsymbol{s}^*) = -\lambda \boldsymbol{L}_s^\dagger(\boldsymbol{s}_{c2}(t_0) - \boldsymbol{s}^*)$$

would yield the *same* camera velocity when either using $\boldsymbol{s}(t_0)$ or $\boldsymbol{s}_{c2}(t_0)$ as visual measurement. Classical image-based visual servoing will thus not be perturbed by this pose correction method.

### C. Correction in the image and in the pose

As a final refinement, a third option is to combine (11) with (3) in order to additionally enforce a perfect matching with the measured $\boldsymbol{s}(t_0)$, which allows to correct 3D coordinates $X$, $Y$ using the correction of $Z$. This is simply obtained by implementing the correction

$$\begin{cases} \mathbf{x}_{c3}(t_0) &= \mathbf{x}_m(t_0) + \Delta\mathbf{x} \\ \boldsymbol{s}_{c3}(t_0) &= \boldsymbol{s}(t_0) \\ {}^C X_{c3}(t_0) &= u_{c3}(t_0) {}^C Z_{c3}(t_0) \\ {}^C Y_{c3}(t_0) &= v_{c3}(t_0) {}^C Z_{c3}(t_0) \\ {}^C Z_{c3}(t_0) &= f(\mathbf{x}_{c3}(t_0)) \end{cases} \tag{14}$$

## III. Predictive Model for the Visual Features

The goal of a predictive model is to generate the evolution of the feature location on the image plane in the time period $t \in [t_0,\, t_0 + T]$ (with $0 < T < \infty$ being the prediction horizon) for a given choice of the camera linear/angular velocity $\mathbf{v}(t)$ (seen as the system input). We assume the time interval $[t_0,\, t_0 + T]$ is discretized in $H$ uniform steps of duration $\tau = T/H$, and take $\mathbf{v}(t) = \mathbf{v}(t_k) = const$ for $t \in [t_k, t_{k+1}]$, with $k = 0 \ldots H - 1$. We now discuss two possibilities for implementing the predictive model.

### A. Linear local model

The Linear Local Model, already proposed in [8], is simply a forward integration of $\boldsymbol{s}_m(t)$ via the interaction matrix $\boldsymbol{L}_s$: from an initial $\boldsymbol{s}_m(t_0)$ one can propagate over time

$$\boldsymbol{s}_m(t_{k+1}) = \boldsymbol{s}_m(t_k) + \boldsymbol{L}_s(\boldsymbol{s}_m(t_k),\, \boldsymbol{Z}(t_k))\mathbf{v}(t_k)\tau \tag{15}$$

with $\tau = T/H$ being the sampling time and $\boldsymbol{Z} = [Z_1, \ldots, Z_N]$ the $N$ depths associated to the $N$ feature points. In (15) one can either set $\boldsymbol{Z}(t_k) = \boldsymbol{Z}(t_0) = const$ (keep the initial approximation $\boldsymbol{Z}(t_0)$ from the model $\mathbf{x}_m(t_0)$), or update $\boldsymbol{Z}(t_k)$ from the known $\mathbf{v}(t_k)$ (see (16) in the next Sect. III-B).

### B. Nonlinear global model

The Nonlinear Global Model is based on the target pose $\mathbf{x}(t)$ in $\mathcal{F}_C$. Indeed, as explained, from a model $\mathbf{x}_m(t_0)$ one can compute the location of the 3D points in $\mathcal{F}_C$, that is, ${}^C\boldsymbol{P}_i(t_0) = \begin{bmatrix}{}^C X_i(t_0) & {}^C Y_i(t_0) & {}^C Z_i(t_0) & 1\end{bmatrix}^T$, and then propagate it using standard kinematics as

$${}^C\boldsymbol{P}_i(t_{k+1}) = e^{\widehat{\mathbf{v}}(t_k)}\, {}^C\boldsymbol{P}_i(t_k) \tag{16}$$

where

$$\widehat{\mathbf{v}}(t_k) = \begin{bmatrix} [\boldsymbol{\omega}(t_k)]_\times & \boldsymbol{v}(t_k) \\ \mathbf{0} & 0 \end{bmatrix} \in se(3)$$

and $e^{\widehat{\mathbf{v}}(t_k)}$ represents the canonical exponential map from $se(3)$ to $SE(3)$ [12]. By plugging (16) in (2) one can finally obtain the behavior of the feature position $\boldsymbol{s}_m(t)$ during the interval $[t_0,\, t_0 + T]$.

### C. Discussion

Obviously the accuracy of both predictive models depends on the accuracy of the initial states: $\boldsymbol{s}_m(t_0)$ and $\boldsymbol{Z}(t_0)$ for (15) and ${}^C\boldsymbol{P}_i(t_0)$ for (16), with $\boldsymbol{Z}(t_0)$ and ${}^C\boldsymbol{P}_i(t_0)$ function of the pose $\mathbf{x}_m(t_0)$. In turn, the accuracy of the initial states is affected by the adopted correction: when using (3) no correction of $\mathbf{x}_m(t_0)$ is performed, but an exact match of $\boldsymbol{s}_m(t_0)$ with the measured $\boldsymbol{s}(t_0)$ is ensured. On the other hand, with (11) $\mathbf{x}_m(t_0)$ is corrected but the corresponding $\boldsymbol{s}_m(t_0)$ can be wrong. Finally, the use of correction (14) guarantees a better match of both $\boldsymbol{s}_m(t_0)$ and $\mathbf{x}_m(t_0)$, and is thus preferable in order to minimize the error in the future visual predictions. We finally note that the prediction (15) obviously yields the same result of (16)–(2) when $\boldsymbol{Z}(t_k)$ is updated with (16) (while a worse result is obtained for a translation motion by just setting $\boldsymbol{Z}(t_k) = \boldsymbol{Z}(t_0) = const$ in (15)).

## IV. Simulation results

We consider an eye-in-hand monocular camera moving with a constant velocity over a prediction horizon of $T = 4$ s discretized in $H = 100$ steps (with thus a sampling time of $\tau = 40$ ms). The target is made of four coplanar points forming a square of 20 cm in length with coordinates

$$\begin{cases} {}^O\boldsymbol{P}_0 &= \begin{bmatrix}-0.1 & -0.1 & 0 & 1\end{bmatrix}^T \\ {}^O\boldsymbol{P}_1 &= \begin{bmatrix}0.1 & -0.1 & 0 & 1\end{bmatrix}^T \\ {}^O\boldsymbol{P}_2 &= \begin{bmatrix}0.1 & 0.1 & 0 & 1\end{bmatrix}^T \\ {}^O\boldsymbol{P}_3 &= \begin{bmatrix}-0.1 & 0.1 & 0 & 1\end{bmatrix}^T \end{cases}.$$

The pose of $\mathcal{F}_O$ in $\mathcal{F}_C$ is initialized as

$$\mathbf{x}(t_0) = \begin{bmatrix}0 & 0 & 0.8 & 0 & 0 & 0\end{bmatrix}^T, \tag{17}$$

i.e., with only a displacement of 0.8 m along the $Z$ axis.

In the following simulations, the camera velocity is set to $\mathbf{v}(t) = \begin{bmatrix}-0.02 & 0 & -0.03 & 0.04 & 0.04 & -0.4\end{bmatrix}^T = const$ for all $t \in [t_0,\, t_0 + T]$. Finally, only the prediction (15) is used (with/without the update of $\boldsymbol{Z}(t_k)$, see the end of Sect. III-C).

### A. 2D Translation error

As shown in Fig. 6, we first consider a case where the error in the target pose consists of just a translation along the $X$ and $Y$ axes. The model of the pose is then taken as $\mathbf{x}_m(t_0) = \begin{bmatrix}-0.1 & -0.1 & 0.8 & 0 & 0 & 0\end{bmatrix}^T$ against the real $\mathbf{x}(t_0)$ from (17).

Figure 2a shows the results of the prediction (15) in the image plane without performing any correction and by keeping $\boldsymbol{Z}(t_k) = \boldsymbol{Z}(t_0)$. Red lines represent the real measurements of visual features $\boldsymbol{s}(t)$, while the blue lines represent the approximated visual features $\boldsymbol{s}_m(t)$ from the prediction model. Obviously, the two sets of trajectories are completely off because of the absence of any correction of the (wrong) model of the target pose in the camera frame. For completeness, Fig. 4a also illustrates the error between real and predicted feature point defined as $\boldsymbol{e}_i(t) = \|\boldsymbol{s}(t) - \boldsymbol{s}_m(t)\|$.

Figure 2b shows the prediction of the visual features after the correction (3) has been applied (but still by keeping $\boldsymbol{Z}(t_k) = \boldsymbol{Z}(t_0)$). The prediction obviously performs better when compared to Fig. 2a, but a divergence between $\boldsymbol{s}_m(t)$ and $\boldsymbol{s}(t)$ still occurs over time: this is due to the remaining approximation $\boldsymbol{Z}(t_k) = \boldsymbol{Z}(t_0)$ when evaluating (15). Similarly to Fig. 4a, Fig. 4b reports the prediction errors $\boldsymbol{e}_i(t)$ which are clearly smaller than in the previous case.

The prediction error can be finally perfectly compensated for by correctly updating $\boldsymbol{Z}(t_k)$ via (16). Fig. 2c shows the results of the visual features prediction: one can verify the perfect match between $\boldsymbol{s}(t)$ and $\boldsymbol{s}_m(t)$ in this case, and similarly in Fig. 4c which depicts the feature error behavior. The same results would have also been obtained by implementing either (11) or (14) as correction steps. Indeed, in the absence of any initial error in the depths $\boldsymbol{Z}$ (as in this simulation) the three corrections yield equivalent results.

### B. Full pose error

As shown in Fig. 6, we now consider a more complex case involving a full translation and rotational error in the modeled target pose by taking $\mathbf{x}_m(t_0) = \begin{bmatrix}0 & 0 & 1.5 & 0.05 & 0 & -0.35\end{bmatrix}^T$ (vs. the real $\mathbf{x}(t_0)$ in (17)). In all the following cases the depth vector $\boldsymbol{Z}(t_k)$ was always correctly updated during the model propagation.

As a baseline condition, Fig. 3a reports the results of the prediction (15) in the camera image plane without performing any correction and Fig. 5a shows the corresponding feature errors.

Figures 3b–5b show the results of using the correction in the image plane (3): we can note how $\boldsymbol{s}_m(t_0) = \boldsymbol{s}(t_0)$ (as expected), but also how $\boldsymbol{s}_m(t)$ substantially diverges over time from $\boldsymbol{s}(t)$ because of the non-corrected $\mathbf{x}_m(t_0)$, in particular because of the initial error induced by a wrong $\boldsymbol{Z}(t_0)$. On the other hand, the behavior of the predicted features substantially improves when adopting the more complex correction (11) which attempts to directly adjust the pose $\mathbf{x}_m(t_0)$. Figures 3c–5c report the results: note how, overall, the predicted features $\boldsymbol{s}_m(t)$ match better $\boldsymbol{s}(t)$ as compared to the previous case, although $\boldsymbol{s}_m(t_0) \neq \boldsymbol{s}(t_0)$. Finally, Figs. 3d–5d show the results of using the correction (14): this allows to obtain the best performance (as expected), since now $\boldsymbol{s}_m(t_0) = \boldsymbol{s}(t_0)$ in addition to having corrected the pose $\mathbf{x}_m(t_0)$.

Summarizing: for no depth errors in the camera model $\mathbf{x}_m(t_0)$ (as in Sect. IV-A), the three corrections (3)–(11)–(14) are equivalently able to cope with the initial error, and thus to yield a ‘perfect’ feature prediction $\boldsymbol{s}_m(t)$ over time (when also propagating $\boldsymbol{Z}(t_k)$ via (16)). However, in presence of ‘full’ errors in the target model (Sect. IV-B), the simple correction (3) has a very poor performance, while the novel corrections (11)–(14) yield much better results (thanks to their ability to directly correct the underlying target pose from the visual error $\boldsymbol{s}_m(t_0) - \boldsymbol{s}(t_0)$. This then confirms the analysis proposed in the previous sections and the importance of developing better correction methods for improving the prediction ability in the context of model predictive control.

## V. Conclusion

We introduced in this paper, a new correction scheme for improving the performance of the visual feature prediction step over a future time horizon. This new correction is able to capture much more precisely the nonlinearities of the target model as compared to other existing possibilities in the literature (e.g., the ‘linear’ correction introduced in [8]). The correction scheme only needs an approximation of the object pose in the camera frame, the current interaction matrix and the current measured features. With the reported results, we showed how the new correction allows to match better the future behavior of the measured features during the prediction phase: therefore, any model predictive control scheme can benefit from this improvement as one can, e.g., safely extend the prediction horizon while retaining an adequate level of accuracy in the predicted $\boldsymbol{s}_m(t)$. As a bonus, this new correction scheme also returns an approximation of the relative pose between the camera and the target which matches the measured visual features in the least squares sense.

We are currently working towards an experimental evaluation of these findings by using an eye-in-hand camera mounted on a robot manipulator, as well as exploiting this improved prediction ability for dealing with feature loss and model predictive control.

## Acknowledgments

This work was partly founded by the PSPC Romeo 2 Project.

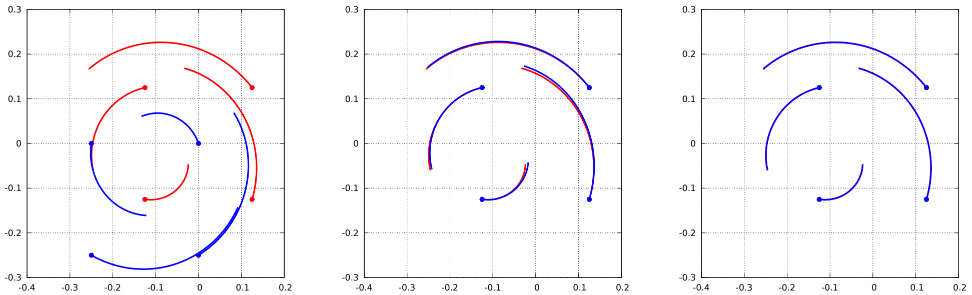
## Appendix

Let ${}^{O}\mathbf{R}_C(\boldsymbol{u},\theta) = \boldsymbol{I}_3 + [\boldsymbol{u}]_\times \sin(\theta) + [\boldsymbol{u}]_\times^2 (1-\cos(\theta))$ be the rotation matrix from $\mathcal{F}_C$ to $\mathcal{F}_O$. One has

$$\dot{\mathbf{x}} = \boldsymbol{L}_x \mathbf{v} = \begin{bmatrix} {}^{O}\mathbf{R}_C(\boldsymbol{u},\theta) & \mathbf{0} \\ \mathbf{0} & \boldsymbol{L}_\omega(\boldsymbol{u},\theta) \end{bmatrix} \mathbf{v}$$

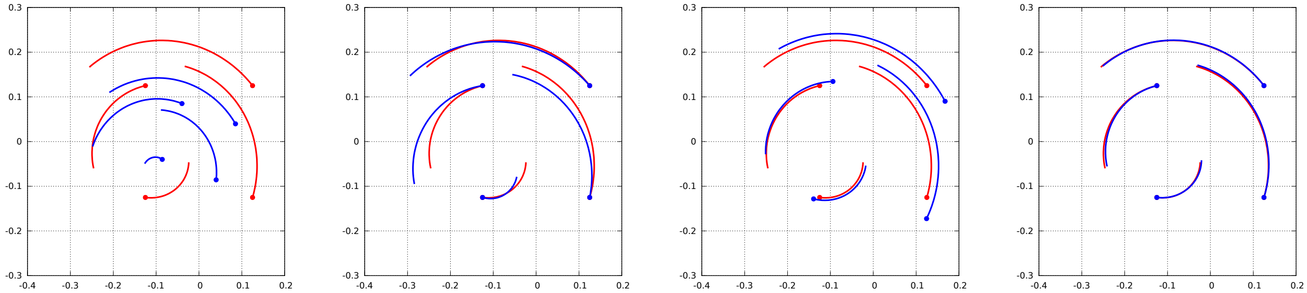
with

$$\boldsymbol{L}_\omega(\boldsymbol{u},\theta) = \boldsymbol{I}_3 - \frac{\theta}{2}[\boldsymbol{u}]_\times + \left(1 - \frac{\operatorname{sinc}(\theta)}{\operatorname{sinc}^2(\theta/2)}\right)[\boldsymbol{u}]_\times^2,$$



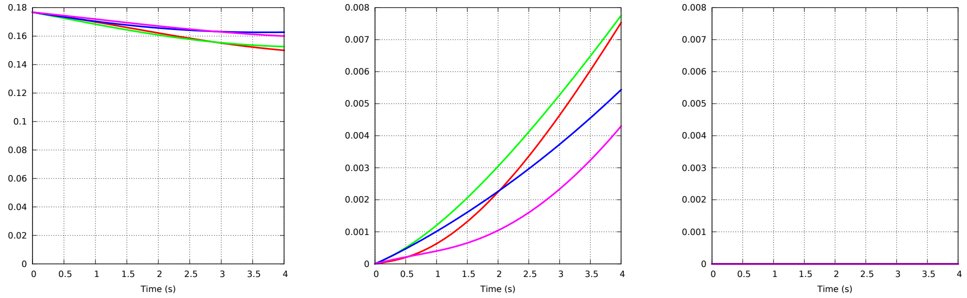
(a) (b) (c)

Fig. 2: First simulation: Visual features in the camera image plane. In red the real features trajectories $\boldsymbol{s}(t)$ and in blue the predicted ones $\boldsymbol{s}_m(t)$. (a): without any initial correction. (b): when using the correction (3) and without updating feature depths , i.e., $\boldsymbol{Z}(t) = \boldsymbol{Z}(t_0) = const$. (c): when using the corrections (11)–(14) and correctly updating $\boldsymbol{Z}(t)$ via (16). Note how the prediction in all the three cases coincides perfectly with the real behavior $\boldsymbol{s}(t)$ (because of the 'simple' 2D initial error in the target pose).



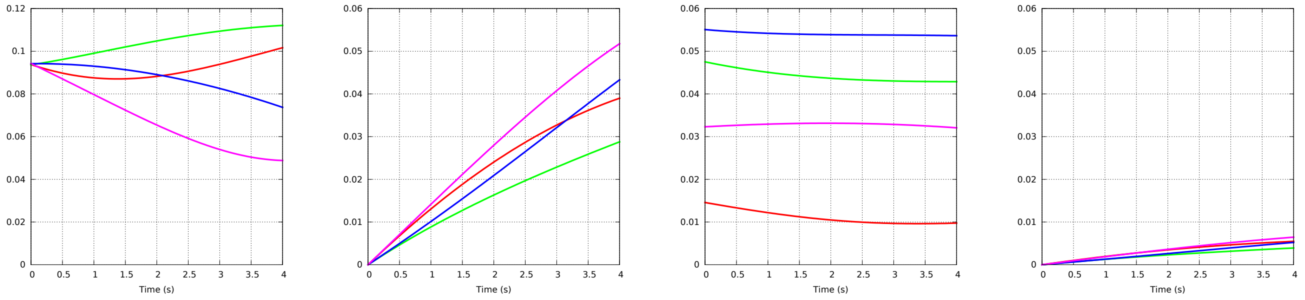
(a) (b) (c) (d)

Fig. 3: Second simulation: Visual features in the camera image plane. In red the real features trajectories $\boldsymbol{s}(t)$ and in blue the predicted ones $\boldsymbol{s}_m(t)$. (a): without any initial correction. (b): when using the correction (3). (c): when using the correction (11). (d): when using the correction (14) which allows to obtain the best performance when the depths $\boldsymbol{Z}$ include initial error.



(a) (b) (c)

Fig. 4: First simulation: Behavior of the prediction error $e_i(t) = \|\boldsymbol{s}(t) - \boldsymbol{s}_m(t)\|$, $i = 1 \ldots 4$. (a): without any initial correction. (b): when using the correction (3) and $\boldsymbol{Z}(t) = \boldsymbol{Z}(t_0) = const$. (c): when using the corrections (11)–(14) and correctly updating $\boldsymbol{Z}(t)$ via (16).



(a)

(b)

(c)

(d)

Fig. 5: Second simulation: Behavior of the prediction error $e_i(t) = \|\boldsymbol{s}(t) - \boldsymbol{s}_m(t)\|$, $i = 1 \ldots 4$. (a): without any initial correction. (b): when using the correction (3). (c): when using the correction (11). (d): when using the correction (14).

and $\mathrm{sinc}(\theta) = \sin(\theta)/\theta$, see [9]. Since $\det(\boldsymbol{L}_\omega(\boldsymbol{u}, \theta)) = 1/\mathrm{sinc}^2(\theta/2)$, matrix $\boldsymbol{L}_x$ is invertible for $\theta \neq 2k\pi$, $k \in \mathbb{Z}^*$, i.e., within the workspace of any normal application.

Furthermore, the relation between camera linear/angular velocity $\mathbf{v}$ and features velocity $\dot{\boldsymbol{s}}$ is

$$\dot{\boldsymbol{s}} = \boldsymbol{L}_s(\boldsymbol{s}, \mathbf{x})\mathbf{v} \tag{18}$$

with [13]

$$\boldsymbol{L}_s = \begin{bmatrix} -\frac{1}{Z} & 0 & \frac{u}{Z} & uv & -(1+u^2) & v \\ 0 & -\frac{1}{Z} & \frac{v}{Z} & 1+v^2 & -uv & -u \end{bmatrix}. \tag{19}$$

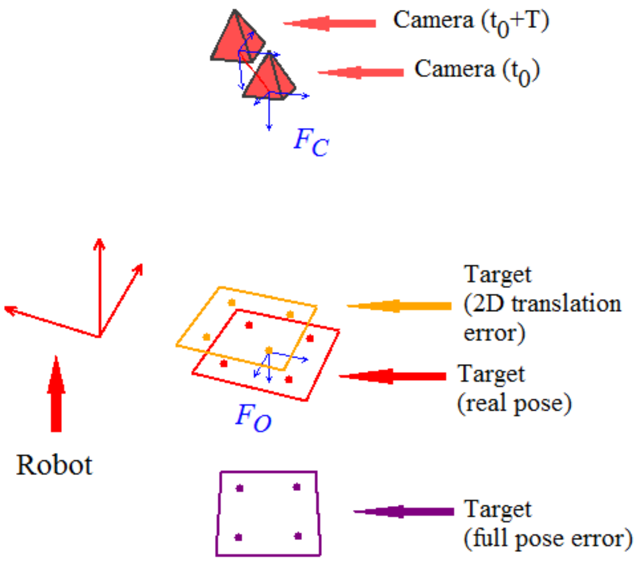


Fig. 6: Robot frame, real target pose and camera trajectory are represented in red. Target 2D translation pose error is represented in orange (Sect. IV-A). Target full pose error is represented in purple (Sect. IV-B).

## References

[1] D. Folio and V. Cadenat, “A controller to avoid both occlusions and obstacles during a vision-based navigation task in a cluttered environment,” in *Decision and Control, 2005 and 2005 European Control Conference. CDC-ECC ’05. 44th IEEE Conference on*, Dec 2005, pp. 3898–3903.

[2] T. Murao, T. Yamada, and M. Fujita, “Predictive visual feedback control with eye-in-hand system via stabilizing receding horizon approach,” in *Decision and Control, 2006 45th IEEE Conference on*, dec. 2006, pp. 1758 –1763.

[3] M. Sauvée, P. Poignet, E. Dombre, and E. Courtial, “Image based visual servoing through nonlinear model predictive control,” in *International Conference on Decision and Control*, San Diego, CA, USA, December 13-15 2006.

[4] C. Lazar and A. Burlacu, “Visual servoing of robot manipulators using model-based predictive control,” in *Industrial Informatics, 2009. INDIN 2009. 7th IEEE International Conference on*, june 2009, pp. 690 –695.

[5] C. Lazar, A. Burlacu, and C. Copot, “Predictive control architecture for visual servoing of robot manipulators,” in *IFAC World Congress*, Milano (Italy), August 28 - September 2 2011, pp. 9464–9469.

[6] G. Allibert, E. Courtial, and Y. Tour, “Real-time visual predictive controller for image-based trajectory tracking of mobile robot,” in *17th IFAC World Congr.*, Seoul, Korea, July 2008.

[7] C. Lazar and A. Burlacu, “Predictive control strategy for image based visual servoing of robot manipulators,” in *International Conference on Automation and Information*, Bucharest, 2008, pp. 91–97.

[8] G. Allibert, E. Courtial, and F. Chaumette, “Predictive control for constrained image-based visual servoing,” *IEEE Transaction on Robotics*, vol. 26, no. 5, pp. 933–939, October 2010.

[9] E. Malis, F. Chaumette, and S. Boudet, “2 1/2 D Visual Servoing,” *IEEE Trans. on Robotics and Automation*, vol. 15, pp. 238–250, 1999.

[10] F. Chaumette and S. Hutchinson, “Visual servo control, Part I: Basic approaches,” *IEEE Robotics and Automation Magazine*, vol. 13, no. 4, pp. 82–90, 2006.

[11] E. Marchand and F. Chaumette, “Virtual visual servoing: a framework for real-time augmented reality,” in *EUROGRAPHICS’02 Conf. Proceeding*, ser. Computer Graphics Forum, G. Drettakis and H.-P. Seidel, Eds., vol. 21(3), no. 3, Saarebrücken, Germany, Sept. 2002, pp. 289–298.

[12] R. M. Murray, Z. Li, and S. S. Sastry, *A Mathematical Introduction to Robotic Manipulation*. CRC Press, 1994.

[13] F. Chaumette and S. Hutchinson, “Visual servo control, part i: Basic approaches,” *IEEE Robotics and Automation Magazine*, vol. 13, no. 4, pp. 82–90, December 2006.