Mechatronics 22 (2012) 444–467

Contents lists available at SciVerse ScienceDirect

Mechatronics

journal homepage: www.elsevier.com/locate/mechatronics

A second-order conic optimization-based method for visual servoing

Eissa Nematollahi [a,1], Aleksandar Vakanski [b], Farrokh Janabi-Sharifi [b,*]

[a] School of Administrative Studies, York University, 4700 Keele Street, Toronto, Ontario, Canada M3J 1P3
[b] Department of Mechanical and Industrial Engineering, Ryerson University, 350 Victoria Street, Toronto, Ontario, Canada M5B 2K3

ARTICLE INFO

Article history:
Available online 10 March 2012

Keywords:
Visual servoing
Image feature parameters
Optimization model
Second-order conic optimization

ABSTRACT

This work presents a novel method for the visual servoing control problem based on second-order conic optimization. Special cases of the proposed method provide similar results as those obtained by the position-based and image-based visual servoing methods. The goal in our approach is to minimize both the end-effector trajectory in the Cartesian space and image feature trajectories simultaneously. For this purpose, a series of second-order conic optimization problems is solved. Each problem starts from the current camera pose and finds the camera velocity as well as the next camera pose such that (1) the next camera pose is as close as possible to the line connecting the initial and desired camera poses, and (2) the next feature points are as close as possible to the corresponding lines connecting the initial and desired feature points. To validate our approach, we provide simulations and experimental results for several different camera configurations.

1. Introduction

The robot visual servoing (VS) refers to control of robotic systems using information from a vision system. Visual servoing structures can be categorized into the three main streams of position-based visual servoing (PBVS), image-based visual servoing (IBVS), and hybrid visual servoing (HVS) [1–3].

PBVS systems use information for a target object features extracted from image space to estimate the position and orientation (pose) of an end-effector with respect to the object [4–10]. The control is designed based on the relative error between estimated and desired poses of the end-effector in the Cartesian space. This approach provides good control for large movements of the end-effector. However, PBVS methods do not provide a mechanism for regulation of the feature trajectories in the image space, which can cause features to leave the camera field-of-view. In addition, the pose estimation [4–10] is sensitive to camera and object model errors and is computationally expensive.

IBVS systems employ direct mapping of time changes of the errors between the current and desired image feature parameters to end-effector velocities in the Cartesian space [1–3,9]. The mapping is accomplished by introducing the image Jacobian matrix. For a properly selected set of features, traditional control law based on exponential decrease of the error in the image space ensures local asymptotic stability of the system and demonstrates good performance. IBVS is less sensitive to calibration and modeling errors, does not require full pose estimation, and is not as computationally expensive as PBVS. On the other hand, the control law with exponential error decrease forces the image feature trajectories to follow straight lines, which often causes suboptimal Cartesian end-effector trajectories. That is especially obvious for tasks with rotational camera movements, when the camera retreats along the optical axis and afterward returns to the desired location [11].

HVS techniques have been proposed to alleviate the limitations of IBVS and PBVS systems. In particular, HVS methods offer solutions to the problem of camera retreat through decoupling the rotational and translational degrees of freedom. The hybrid approach of 2-1/2 D visual servoing [12] utilizes extracted partial camera displacement from the Cartesian space to control the rotational motion of the camera, and visual features from the image space to control the translational camera motions. Other hybrid approaches include the partitioned methods [13,14], which partition the control of the camera along the optical axis from the control of the remained degrees of freedom, and the switching methods [10,15], which use a switching algorithm between the IBVS and PBVS methods depending on specified criteria for optimal performance. Hybrid strategies in both control and planning levels have been also proposed [10] to avoid image singularities, local minima, and boundaries. Some of the drawbacks associated with most HVS approaches include the computational expense, possibility of features leaving the image boundaries, and sensitivity to image noise. The problem of designing optimal and more effective HVS systems still remains unsolved.

* Corresponding author. Tel.: +1 416 979 5000x7097; fax: +1 416 979 5265.
E-mail addresses: eissa@yorku.ca, eissa.nematollahi@rotman.utoronto.ca (E. Nematollahi), aleksandar.vakanski@ryerson.ca (A. Vakanski), fsharifi@ryerson.ca (F. Janabi-Sharifi).
[1] Present address: Rotman School of Management, University of Toronto, Canada.

0957-4158/$ - see front matter © 2012 Elsevier Ltd. All rights reserved.
doi:10.1016/j.mechatronics.2012.01.010

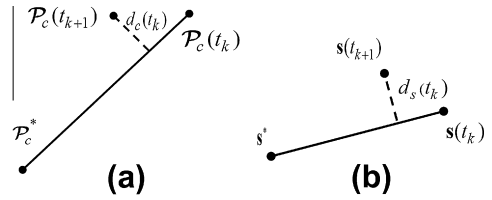

Fig. 1. (a) The distance from the next camera pose $\mathcal{P}_c(t_{k+1})$ to the line connecting $\mathcal{P}_c(t_k)$ and $\mathcal{P}_c^*$; (b) The distance from the next image feature $\mathbf{s}(t_{k+1})$ to the line connecting $\mathbf{s}(t_k)$ and $\mathbf{s}^*$.

Our goal is to design a hybrid controller as an optimization model that optimizes both image feature trajectories and the Cartesian camera trajectory, while satisfying the imposed constraints. For instance, keeping the object features in the field-of-view of the camera is formulated as constraints in the proposed optimization model. The existing optimization efforts have focused on both control and planning levels.

A body of work in the literature, e.g., [16,17], employed optimization techniques in either the Cartesian or the image space to improve the control performance of VS systems. Hashimoto et al. [16] implemented a linear-quadratic (LQ) controller using a linear time-invariant (LTI) model of an IBVS system, obtained by linearization of an image features velocity function at the desired image features. Malis [17] addressed low convergence rates of classical VS methods and proposed two control laws based on second-order minimization techniques with quadratic convergence rates. He

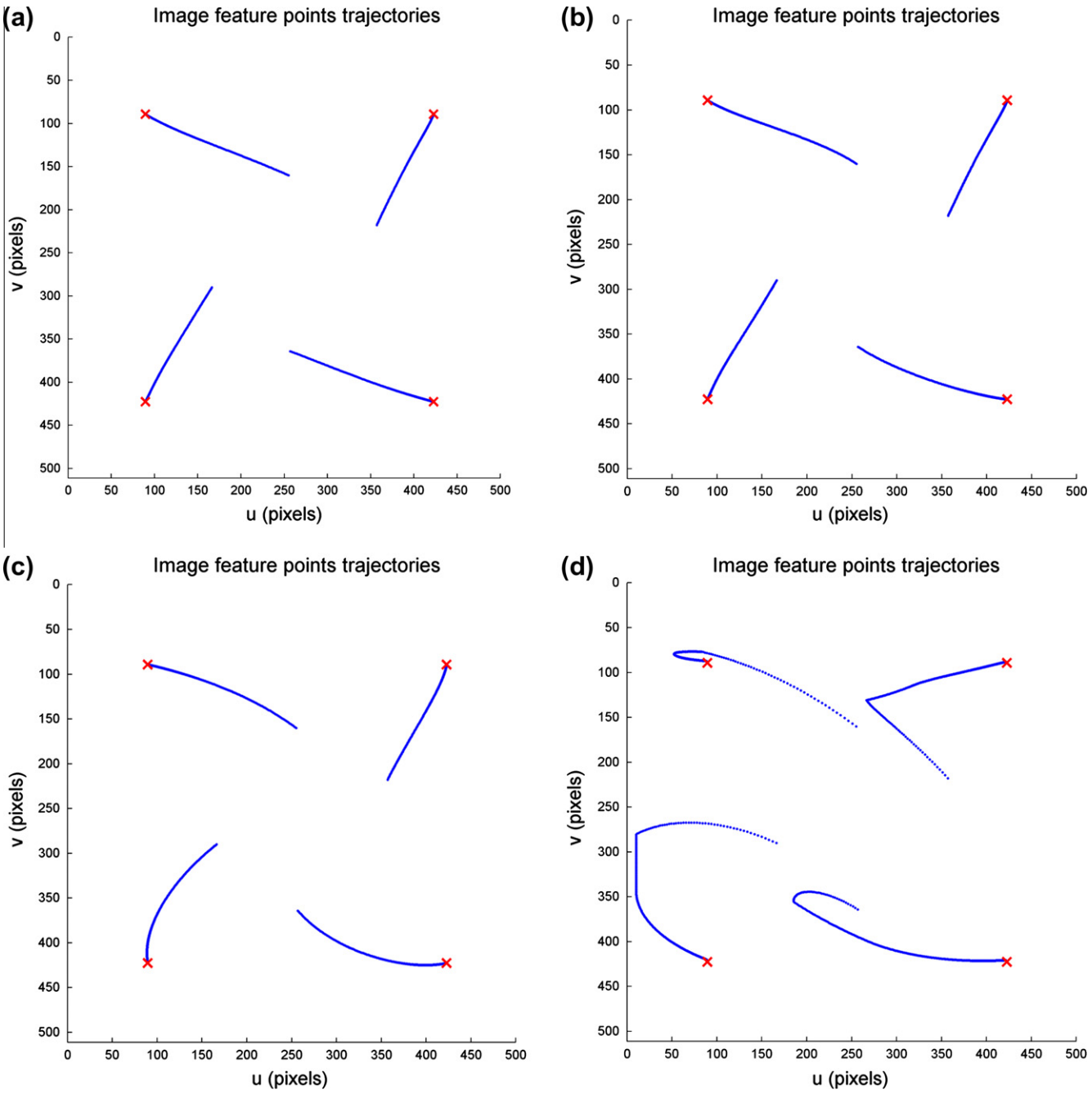

Fig. 2. Effects of varying α on the image feature trajectories obtained by the SOCO method for Case 1. (a) $\alpha = 0.01$, (b) $\alpha = 0.1$, (c) $\alpha = 1$, (d) $\alpha = 10$. The signs ' $\times$ ' in the figures depict the desired positions of the image features.

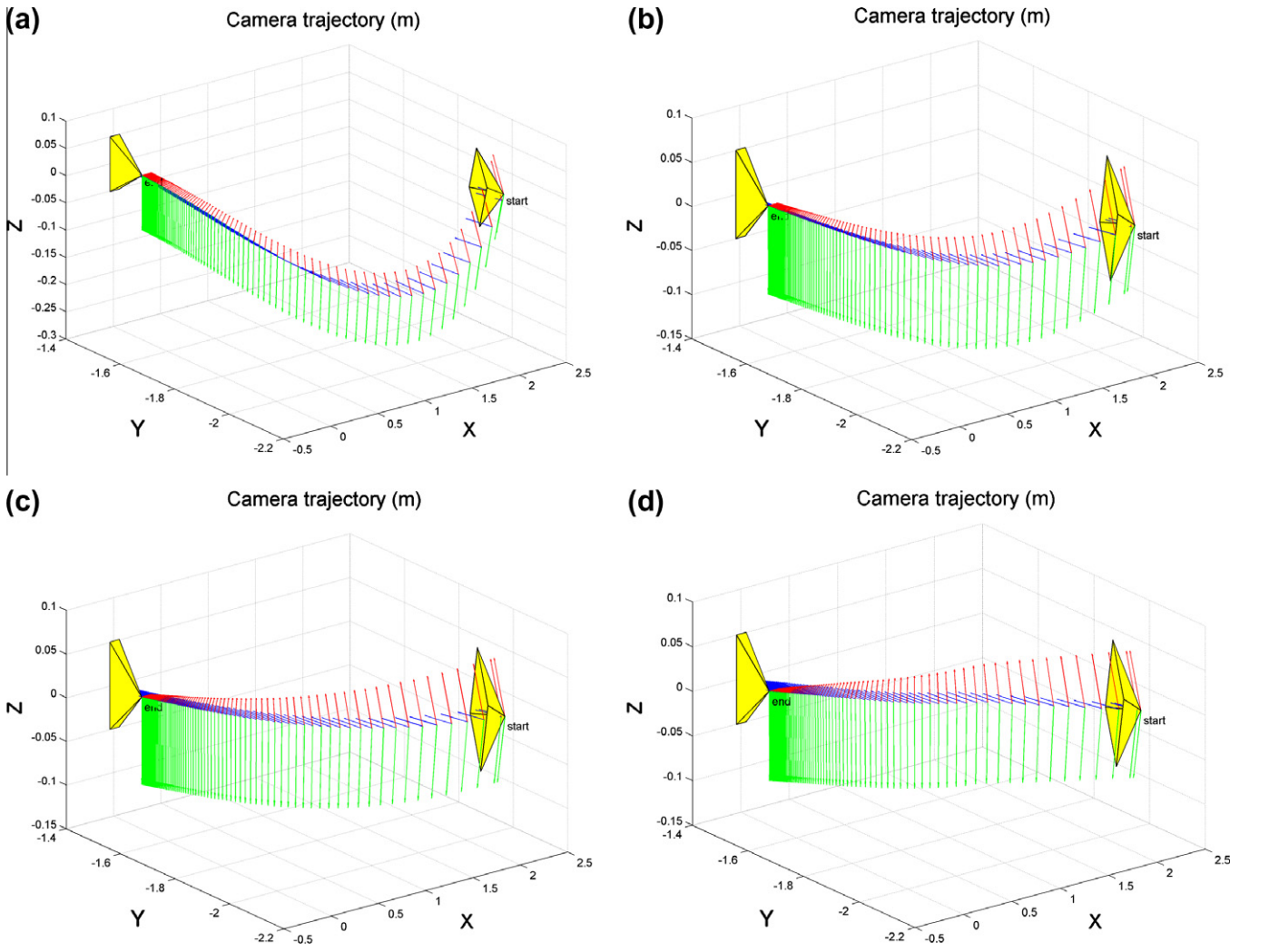

Fig. 3. Effects of varying α on the camera trajectory obtained by the SOCO method for Case 1. (a) $\alpha = 0.01$, (b) $\alpha = 0.1$, (c) $\alpha = 1$, (d) $\alpha = 10$. Note: In the figures that follow, subfigures (a), (b), (c), and (d), depict image feature trajectories, camera trajectory, image feature errors, linear and angular camera velocities, respectively.

applied his methods toward improving the convergence rate of the IBVS method. Similarly, Allibert et al. [18] proposed an approach for improving the performance of the classical IBVS scheme by using models for prediction of the image feature parameters over a finite prediction horizon. The predictive control was formulated as a nonlinear optimization problem in the image space subject to visibility, workspace and joint motors constraints. The approach was presented as an ad hoc algorithm without a stability proof.

Hafez and Jawahar [19] presented an optimization model for visual servoing using a hybrid objective function that minimized both the image feature errors and the camera error. Their approach is based on solving unconstrained nonlinear least square functions. Because the problem was not formulated as a convex optimization, the approach does not warrant global convergence, and due to the lack of constraints integration, it exhibits poor performance compared to other hybrid methods.

A number of researchers applied optimization algorithms in a path planning level [20]. For example, Chesi and Hung [21] used polynomial parameterization of the global path in the formulation of a non-convex optimization model, subject to features visibility, workspace constraints and joint limits constraints. Image projections of the optimized path were mapped into an IBVS controller to achieve the desired camera trajectory. A main shortcoming of this work is the formulation of the problem as a non-convex optimization model, which may lead to the convergence of the algorithm to local minima. Mezouar and Chaumette [22] formulated a problem for minimizing the energy and the acceleration of the trajectories in the image space. However, the approach does not consider the image boundary constraint. Hafez et al. [23] proposed an approach for finding the shortest camera path using a second-order conic optimization with image features visibility constraints. The objective function of their model was to minimize only the Cartesian camera trajectory, whereas the orientation of the camera frame was kept fixed during the positioning task. Their approach yielded satisfactory results only for the cases of a pure translational camera motion, or combinations of translational and small rotational camera motions. Chesi [24] proposed an algorithm for finding a global path of the camera using linear matrix inequalities (LMI) and homogeneous forms for parameterization of both the objective function and the constraints. The path planning problem was formulated as a convex optimization problem, by using a parameter-dependent version of the Rodrigues formula for the axis-angle representation of the camera orientation. The resulting camera path was tracked with an IBVS controller. Also, several authors concentrated on generating optimal paths for differential drive robots in the Cartesian space [25] or in the image space [26].

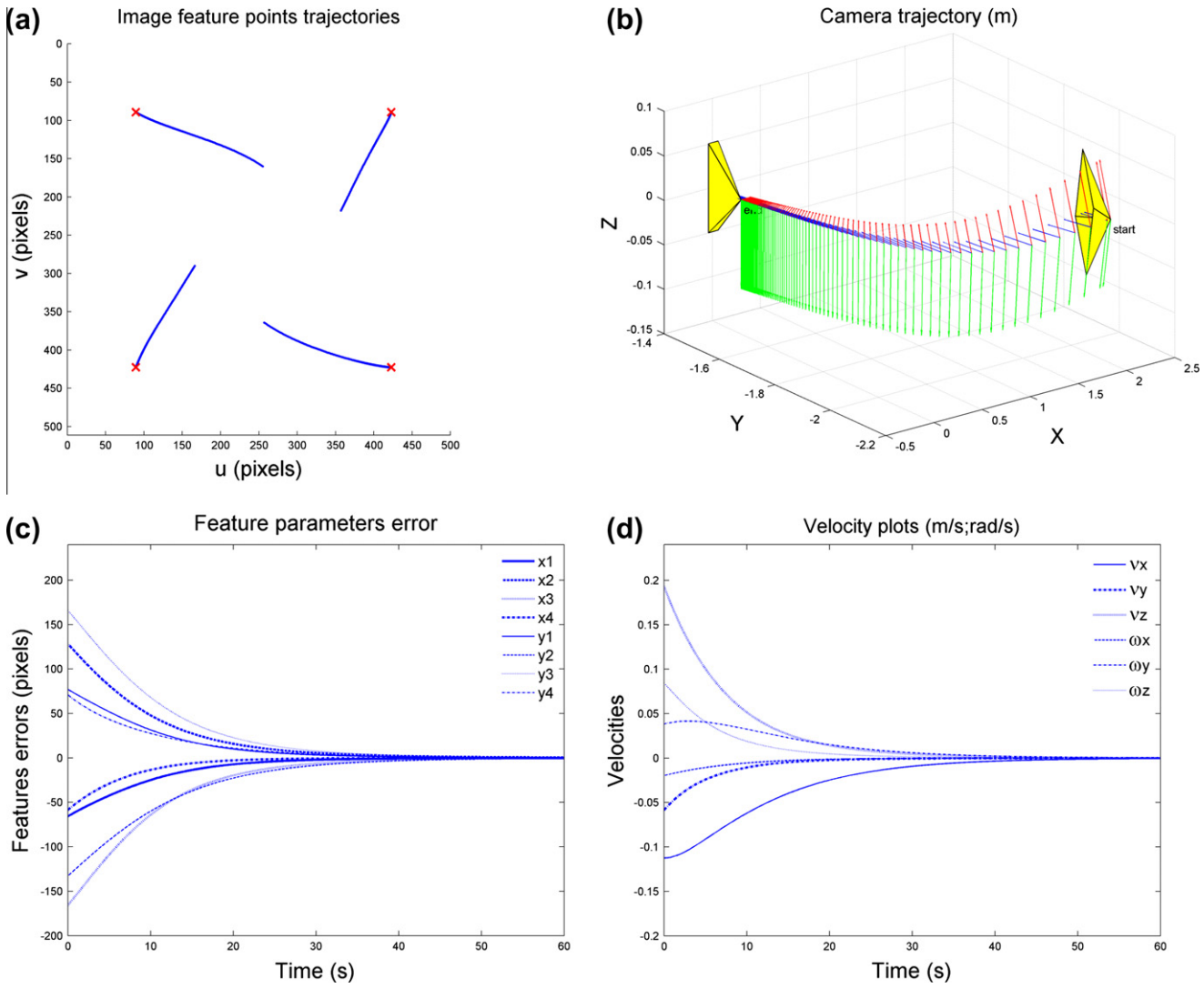

Fig. 4. Simulation results obtained by the SOCO method for Case 1.

In this paper, we propose a method that minimizes simultaneously the lengths of the paths connecting the initial and target feature points in the image plane as well as the length of the path connecting the initial and target camera poses in the Cartesian space subject to some constraints. To achieve this goal, we first discretize the time space. Then, starting from the initial pose, we solve an optimization model at the current time instant to obtain information of the pose at the next time instant. The resulting models are bi-level optimization problems, which can be transformed to parametric optimization problems by assigning a weight parameter to the camera components of the objective functions. Varying this parameter, we obtain similar results to those obtained by PBVS, IBVS, or HVS approaches.

At each time instant, the parameterized model is a second-order conic optimization (SOCO) problem [27], which can efficiently be solved in polynomial time to determine unknown variables consisting of the camera velocity, the next camera pose, and the next image feature parameters. Unlike other existing approaches in VS, which use optimization techniques for camera path planning, e.g., [21–24], our method focuses on solving the control problem, and yields on-line solutions of the optimization task at each sampling period.

The presented simulation and experimental results of our proposed method confirm that both the camera trajectory and image feature trajectories are minimized. In addition, the approach is convergent to the desired poses meaning that the image feature error and the camera error are both driven to zero. The visibility constraint in the model guarantees that the image features are maintained in the field-of-view of the camera. Robustness of the system to image noises, camera calibration errors, and non-planar features was examined through simulations and experiments. The proposed approach also provides a mechanism for a separate regulation of the camera trajectory and image feature trajectories.

The paper is organized as follows. Section 2 presents a short introduction to IBVS and PBVS controls. In Section 3, the SOCO model is introduced, and the objective function and constraints are discussed in details. Convergence of our approach and feasibility of the optimization models are proven in Section 4. Simulation results for different types of camera motions are presented in

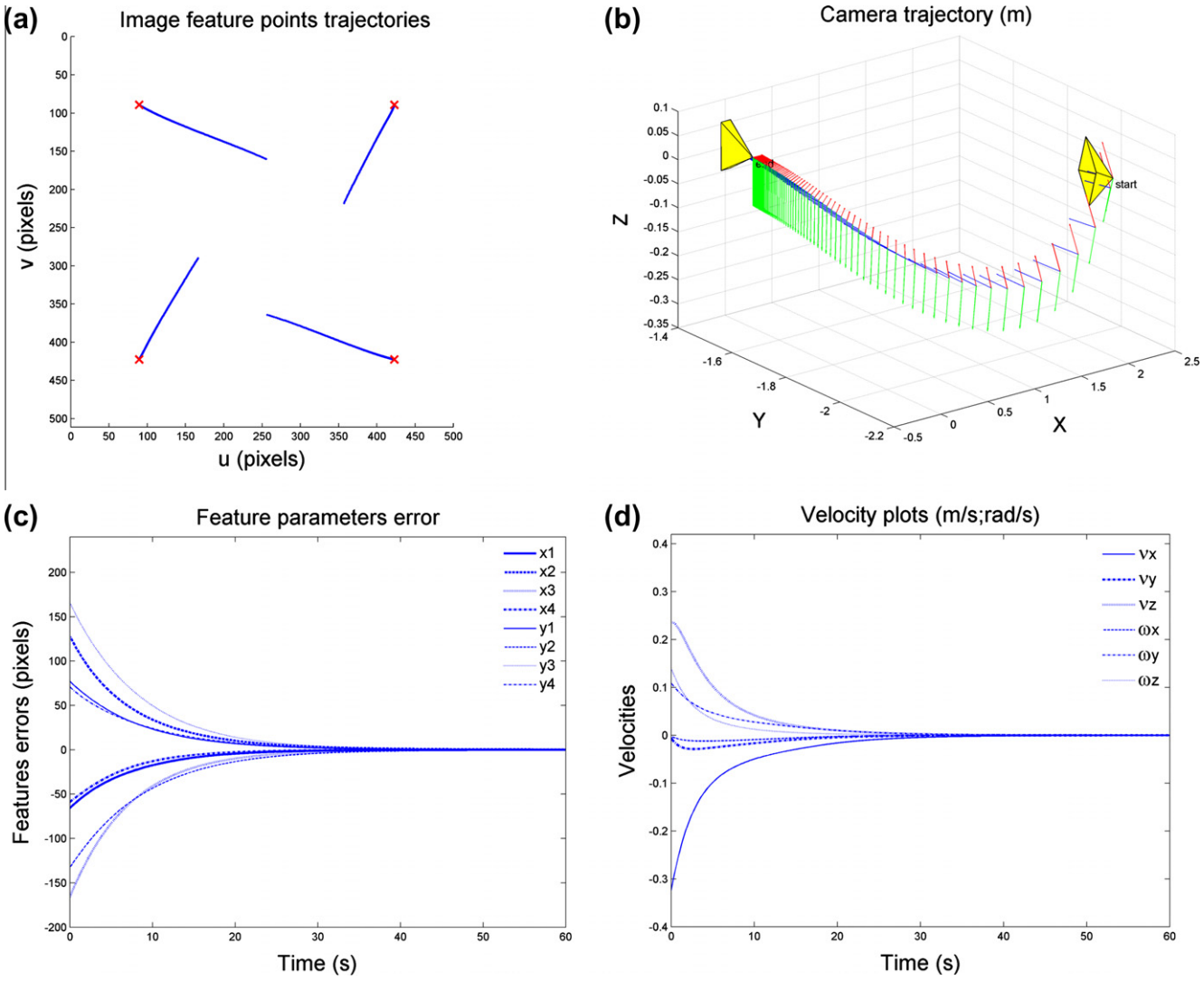

Fig. 5. Simulation results obtained by the IBVS controller for Case 1.

Section 5. Section 6 provides the experimental evaluation of our approach. Finally, concluding remarks and future work directions are given in Section 7.

2. Visual servoing: basic elements

The task of the robot positioning with respect to a target object or tracking a target in VS is based on the projection of target features onto the image plane of the camera. Several factors should be taken into consideration in selecting image features, such as geometric and radiometric properties of the features (e.g., for ease of localization and/or estimation), good visibility from different angles with sufficient resolution, etc. [28,29]. Different image features parameters can be used, such as the coordinates of n feature points in the image plane, the distances between feature points, line parameters, and centroids or other moments of projected surfaces [30]. The presented method considers an eye-in-hand configuration (camera mounted on the robot end-effector), however the results can be extended to eye-off-hand configurations (camera fixed in the workspace) as well.

2.1. Image-based visual servoing

Let $(X_i(t), Y_i(t), Z_i(t))$, for $i = 1, 2, \ldots, n$, denote the 3D coordinates of n feature points of an object in the camera frame at time instant t. The 2D perspective projection of the ith feature point in the image plane of the camera is denoted by $\mathbf{p}_i(t) = [x_i(t)\ y_i(t)]^T \in \mathbb{R}^2$. The set of all image feature vectors $\mathbf{s}(t) = \begin{bmatrix}\mathbf{p}_1^T(t) & \mathbf{p}_2^T(t) & \ldots & \mathbf{p}_n^T(t)\end{bmatrix}^T$ defines the image space $S \subseteq \mathbb{R}^{2n}$.

The relationship between the time change of the image feature parameters and robot end-effector spatial velocity is given by the differential equation

$$\dot{\mathbf{s}}(t) = \mathbf{L}(t, \mathbf{s}(t))\mathbf{v}_e(t), \tag{2.1}$$

where the matrix $\mathbf{L}(t, \mathbf{s}(t))$ is called the image Jacobian matrix, or the interaction matrix, and $\mathbf{v}_e = \begin{bmatrix}\upsilon^T & \boldsymbol{\omega}^T\end{bmatrix}^T$, where υ is the vector of linear velocity of the end-effector frame, and $\boldsymbol{\omega}$ is the vector of angular velocity of the end-effector frame. For simplicity, without loss of generality, we assume that the end-effector coordinate frame coincides with the camera coordinate frame, i.e., $\mathbf{v}_e \equiv \mathbf{v}_c$. For a perspective projection model of the camera, the interaction matrix is a

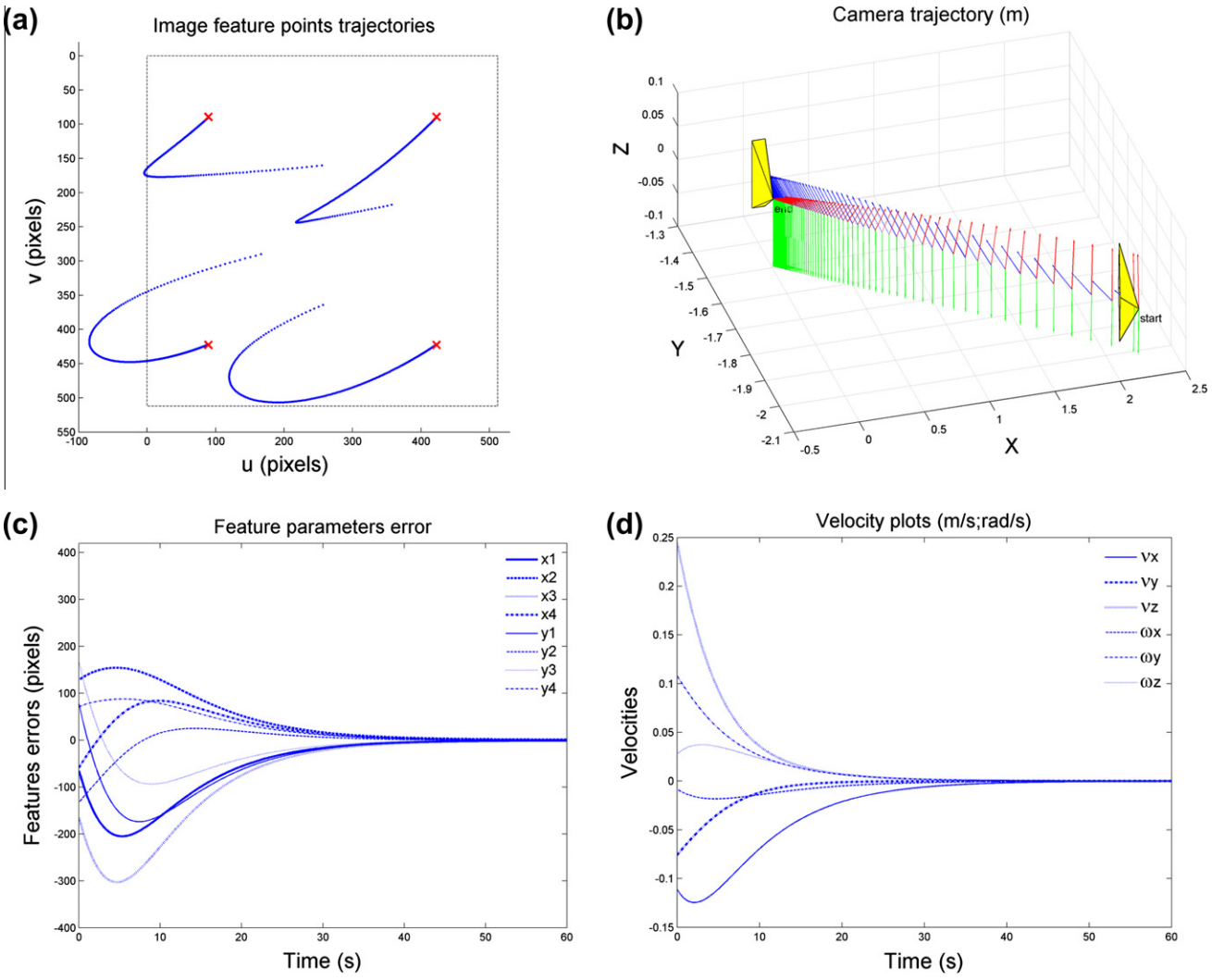

Fig. 6. Simulation results obtained by the PBVS controller for Case 1. Image feature trajectories leave the camera field-of-view.

function of the projections of the features points in the image space $x_i(t)$, $y_i(t)$, and the depth coordinates of the features points with respect to the camera frame $Z_i(t)$, for $i = 1,2,\ldots,n$ [1–3]. Note that the depth coordinates have to be estimated or approximated for computation of the interaction matrix.

For a given vector of desired image feature parameters $\mathbf{s}^*$, the goal in the IBVS method is to generate a control signal such that the error

$$\mathbf{e}_s(t) = \mathbf{s}(t) - \mathbf{s}^*(t) \tag{2.2}$$

is minimized, subject to some constraints. The classical IBVS approach calculates the camera velocity $\mathbf{v}_c(t)$ so that the image feature error $\mathbf{e}_s(t)$ is decreased exponentially with a rate $\lambda \in (0,1)$. Since our approach uses discrete time instances, we first discretize the time space as $t_{k+1} = t_k + \Delta t_k$ for $k = 1,2,\ldots$, then use the Euler's forward discretization rule to approximate (2.1) as

$$\mathbf{s}(t_{k+1}) = \mathbf{s}(t_k) + \mathbf{L}(t_k, \mathbf{s}(t_k))\mathbf{v}_c(t_k)\Delta t_k, \tag{2.3}$$

where the current image feature vector $\mathbf{s}(t_k)$ and the current interaction matrix $\mathbf{L}(t_k, \mathbf{s}(t_k))$ are known, while the next image feature vector $\mathbf{s}(t_{k+1})$ and the current camera velocity $\mathbf{v}_c(t_k)$ are to be determined through solving an optimization problem, which will be introduced in the next section. The sequence $\{\mathbf{s}(t_k)\}_{k=1}^{\infty}$ defines the image features trajectories.

2.2. Position-based visual servoing

In the PBVS approach, the pose of the end-effector with respect to a target object is estimated from the image plane measurements, using approaches such as close-range photogrammetric techniques [4], analytical methods [5], least-square methods [6], Kalman filter estimation [7,8], etc.

The camera error at the current end-effector pose $\mathcal{P}_c(t)$ in the Cartesian space is defined as

$$\mathbf{e}_c(t) = \mathcal{P}_c(t) - \mathcal{P}_c^*, \tag{2.4}$$

where $\mathcal{P}_c^*$ denotes the desired camera pose. The camera pose $\mathcal{P}_c(t)$ at time t consists of position and orientation components. We adopted a minimal representation of the camera orientation by a set of roll-pitch-yaw angles $[\,\varphi(t) \quad \theta(t) \quad \psi(t)\,]^T$.

The relationship between the camera pose and the spatial camera velocity is given by the following differential equation:

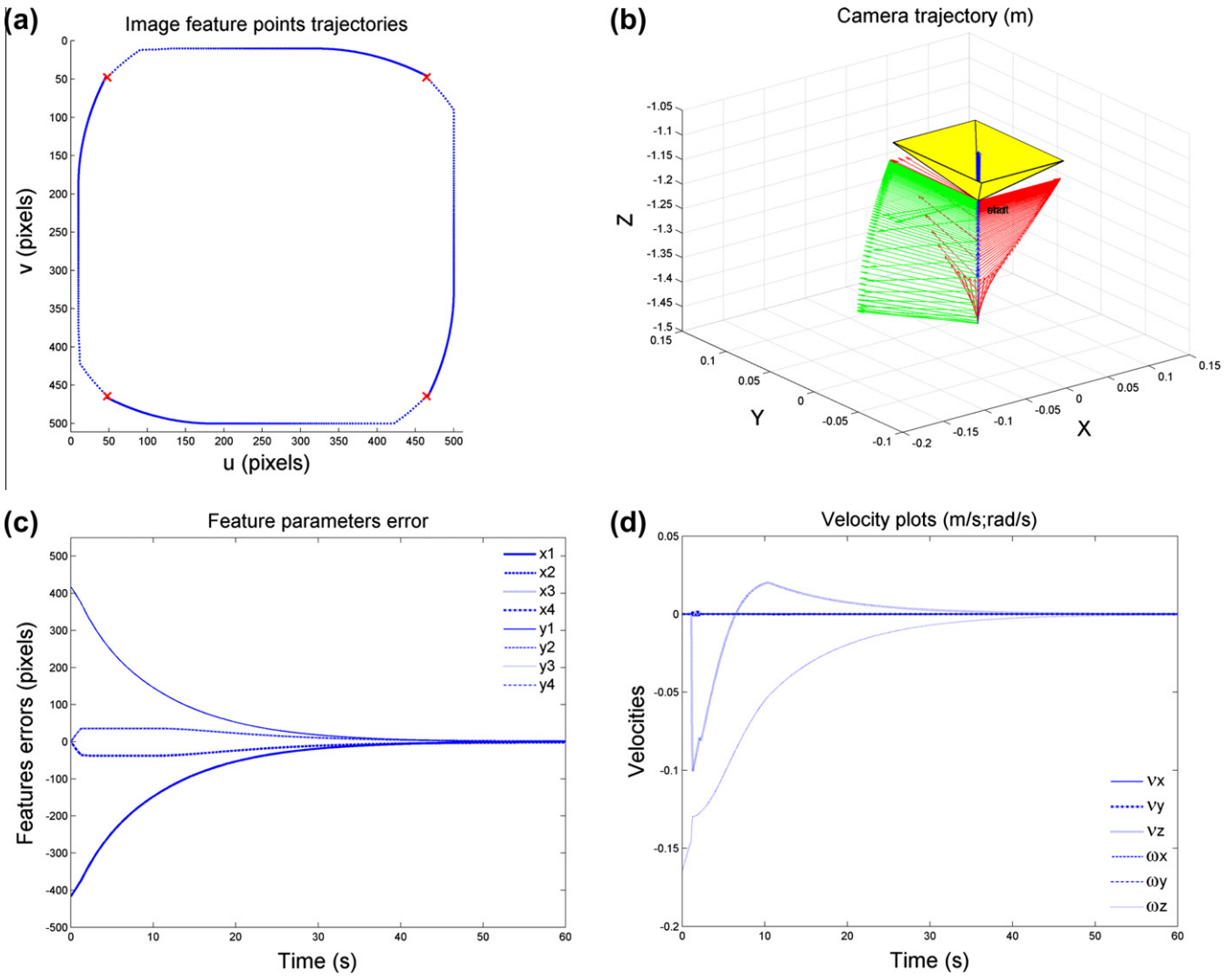

Fig. 7. Simulation results obtained by the SOCO method for Case 2.

$$\begin{bmatrix} \mathbf{I}_{3\times3} & \mathbf{0}_{3\times3} \\ \mathbf{0}_{3\times3} & \mathbf{T}(t) \end{bmatrix} \dot{\mathcal{P}}_c(t) = \begin{bmatrix} \mathbf{R}_c^w(t) & \mathbf{0}_{3\times3} \\ \mathbf{0}_{3\times3} & \mathbf{R}_c^w(t) \end{bmatrix} \mathbf{v}_c(t), \tag{2.5}$$

where the rotation matrix $\mathbf{R}_c^w(t)$ of the camera frame with respect to the world frame is defined as

$$\mathbf{R}_c^w(t) \equiv \begin{bmatrix} c_\varphi c_\theta & c_\varphi s_\theta s_\psi - s_\varphi c_\psi & c_\varphi s_\theta c_\psi + s_\varphi s_\psi \\ s_\varphi c_\theta & s_\varphi s_\theta s_\psi + c_\varphi c_\psi & s_\varphi s_\theta c_\psi - c_\varphi s_\psi \\ -s_\theta & c_\theta s_\psi & c_\theta c_\psi \end{bmatrix}, \tag{2.6}$$

and the transition matrix $\mathbf{T}(t)$ for a set of roll-pitch-yaw angles is defined as

$$\mathbf{T}(t) \equiv \begin{bmatrix} 0 & -s_\varphi & c_\varphi c_\theta \\ 0 & c_\varphi & s_\varphi c_\theta \\ 1 & 0 & -s_\theta \end{bmatrix}. \tag{2.7}$$

The notations c_α and s_α refer to $\cos\alpha$ and $\sin\alpha$, respectively (for an arbitrary angle α). It is easy to show that $\det(\mathbf{T}(t)) = -\cos\theta(t)$, so the matrix $\mathbf{T}(t)$ is nonsingular if $\theta(t) \neq \pm\pi/2$. In addition, we have $\|\mathbf{T}(t)\|_F = \sqrt{3}$ and $\|\mathbf{T}^{-1}(t)\|_F = \sqrt{2+\cos^2\theta}/|\cos\theta|$, where $\|\cdot\|_F$ denotes the Frobenius norm.

Changes in the camera pose can be computed from (2.5) only if the matrix $\mathbf{T}(t)$ is nonsingular. In this case, we have

$$\dot{\mathcal{P}}_c(t) = \mathbf{G}_c(t)\mathbf{v}_c(t), \tag{2.8}$$

where

$$\mathbf{G}_c(t) = \begin{bmatrix} \mathbf{R}_c^w(t) & \mathbf{0}_{3\times3} \\ \mathbf{0}_{3\times3} & \mathbf{T}^{-1}(t)\mathbf{R}_c^w(t) \end{bmatrix}. \tag{2.9}$$

In the PBVS approach, the camera trajectory error is decreased exponentially with a rate $\lambda \in (0,1)$, while the image feature trajectories are not directly controlled, which may result in features leaving the camera's field-of-view.

Using the Euler's forward discretization rule, the discrete approximation of (2.8) is given as

$$\mathcal{P}_c(t_{k+1}) = \mathcal{P}_c(t_k) + \mathbf{G}_c(t_k)\mathbf{v}_c(t_k)\Delta t_k, \tag{2.10}$$

where the current camera pose $\mathcal{P}_c(t_k)$ and the current matrix $\mathbf{G}_c(t_k)$ are known, while the next camera pose $\mathcal{P}_c(t_{k+1})$ and the current camera velocity $\mathbf{v}_c(t_k)$ are to be determined through solving an optimization problem, which will be introduced in the next section. The

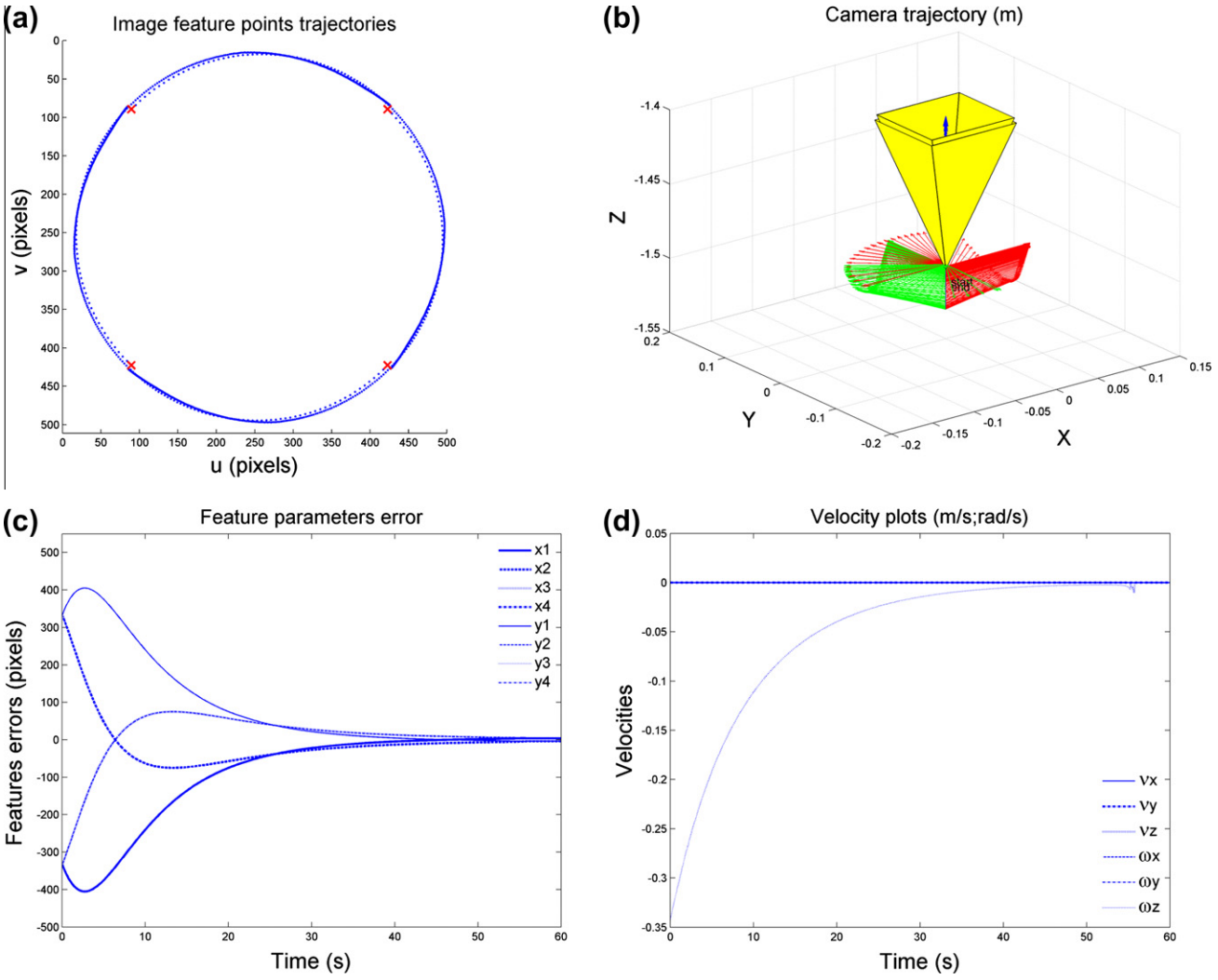

Fig. 8. Simulation results obtained by the SOCO method for Case 3.

sequence $\{\mathcal{P}_c(t_k)\}_{k=1}^{\infty}$ of the camera poses defines the camera pose trajectory.

3. Second-order conic optimization model

3.1. General formulation of a SOCO problem

A SOCO problem is an optimization problem with a linear objective function subject to a set of linear and second–order conic constraints. A general model of a SOCO problem [27,31] has the form

$$\begin{aligned} \min \quad & \mathbf{c}^T\mathbf{z} \\ \text{s.t.} \quad & \mathbf{A}\mathbf{z} = \mathbf{b} \\ & \mathbf{z} \in K \end{aligned} \tag{3.1}$$

where matrix $\mathbf{A} \in \mathbb{R}^{m\times n}$ and vectors $\mathbf{b} \in \mathbb{R}^m, \mathbf{c} \in \mathbb{R}^n$ are the input data, and $\mathbf{z} \in \mathbb{R}^n$ is a variable in a second–order cone K defined as

$$K = \{\mathbf{z} \in \mathbb{R}^n \,|\, \|[z_2, z_3, \ldots, z_n]\| \leqslant z_1\}, \tag{3.2}$$

with the notation $\|\cdot\|$ used for the Euclidean (l_2) norm. This norm notation will be used throughout the text, unless otherwise specified.

The motivation for using a SOCO model in this work is due to appealing properties of the model and fast algorithms that are available for solving such a model. Specifically, the set of feasible solutions of the SOCO problem is convex, which guarantees convergence to a globally optimal solution. There exist efficient polynomial-time algorithms for solving the SOCO problem which have been implemented in commercial software packages, such as MOSEK [32] and CPLEX [33], as well as freely available software packages, such as SeDuMi [34] and SDPT3 [35].

3.2. Application of the SOCO to visual servoing

The formulation of the VS problem in this work differs from the previous works in a sense that the problem is formulated as a constrained second-order conic optimization problem in the hybrid space [12,19]. Our goal is to start from an initial camera pose and find a sequence of camera velocities, camera poses, and image feature points iteratively, so that the induced lengths of the camera trajectory and image feature trajectories are minimized. This is accomplished by forcing the camera trajectory at any time instant to be as close as possible to the line connecting the initial and desired camera poses. In addition, each image feature trajectory

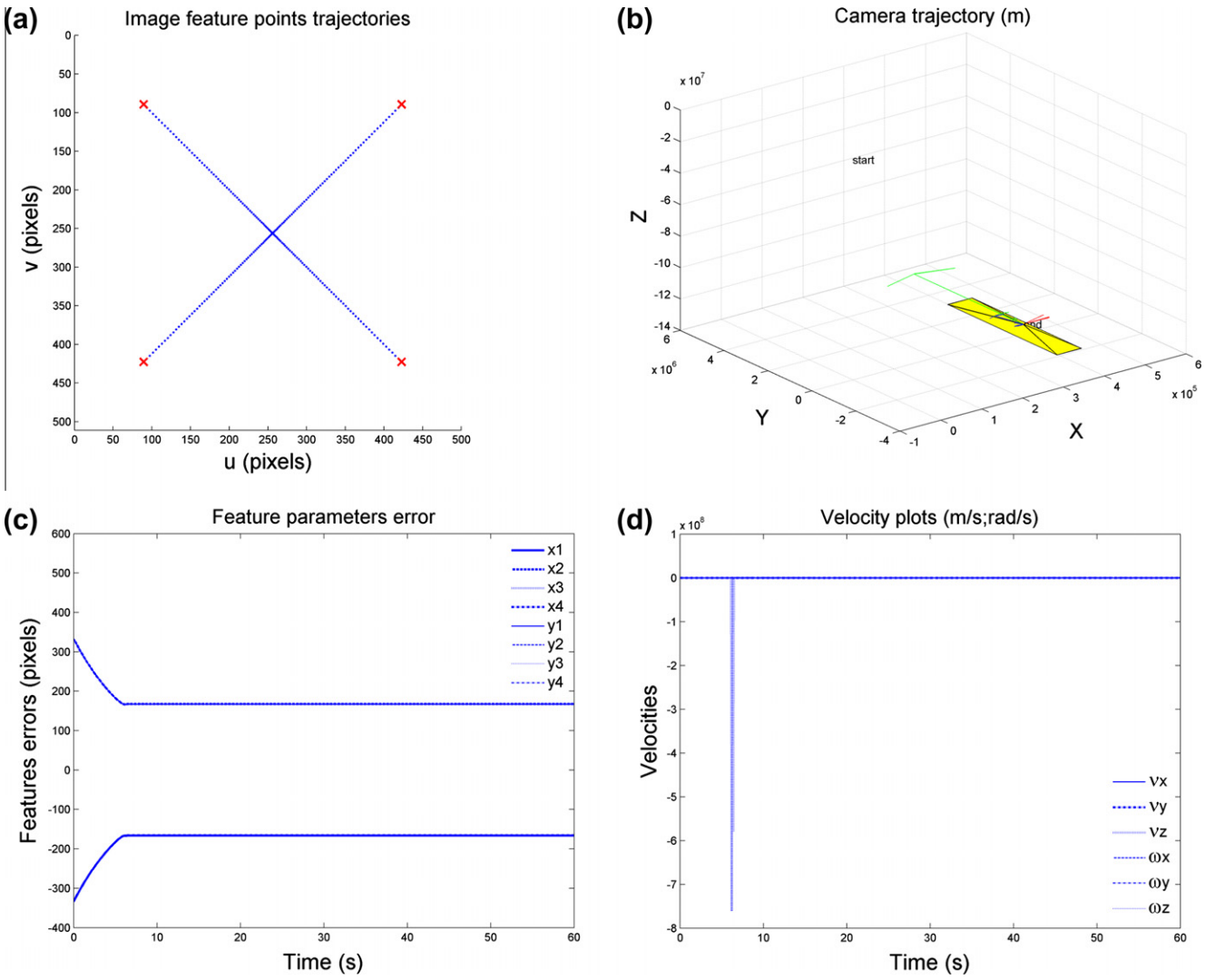

Fig. 9. Simulation results obtained by the IBVS controller for Case 3. This method fails since the camera retreats to infinity.

should be as close as possible to the corresponding line connecting the initial and desired feature points in the image plane.

Assume that the camera intrinsic parameters and the 3-D model of a target object are known. Let t_k be the current time instant. The shortest distance from the next camera pose $\mathcal{P}_c(t_{k+1})$ to the line connecting the current camera pose $\mathcal{P}_c(t_k)$ to the desired camera pose $\mathcal{P}_c^*$ is denoted by $d_c(t_k)$, as illustrated in Fig. 1a, and computed as

$$d_c(t_k) = \frac{\|\mathbf{e}_c(t_{k+1}) \times \mathbf{e}_c(t_k)\|}{\|\mathbf{e}_c(t_k)\|}. \tag{3.3}$$

Here $\mathbf{e}_c(t_k)$ denotes the camera pose error at time t_k with respect to the desired camera pose, as defined in (2.4). Using the Lagrange's identity, (3.3) can be regrouped as

$$\begin{aligned} d_c^2(t_k) &= \frac{\left(\mathbf{e}_c^T(t_{k+1})\mathbf{e}_c(t_{k+1})\right)\left(\mathbf{e}_c^T(t_k)\mathbf{e}_c(t_k)\right) - \left(\mathbf{e}_c^T(t_{k+1})\mathbf{e}_c(t_k)\right)\left(\mathbf{e}_c^T(t_k)\mathbf{e}_c(t_{k+1})\right)}{\mathbf{e}_c^T(t_k)\mathbf{e}_c(t_k)} \\ &= \mathbf{e}_c^T(t_{k+1})\left(\mathbf{I} - \frac{\mathbf{e}_c(t_k)\mathbf{e}_c^T(t_k)}{\mathbf{e}_c^T(t_k)\mathbf{e}_c(t_k)}\right)\mathbf{e}_c(t_{k+1}). \end{aligned} \tag{3.4}$$

Recalling that for a given vector $\mathbf{u}$, the matrix $\mathbf{A_u} = \mathbf{I} - (\mathbf{u}\mathbf{u}^T/\mathbf{u}^T\mathbf{u})$ is symmetric and satisfies $\mathbf{A_u^T A_u} = \mathbf{A_u}$, we have

$$d_c(t_k) = \left(\mathbf{e}_c^T(t_{k+1})\mathbf{A}_{\mathbf{e}_c(t_k)}\mathbf{e}_c(t_{k+1})\right)^{1/2} = \|\mathbf{A}_{\mathbf{e}_c(t_k)}\mathbf{e}_c(t_{k+1})\|. \tag{3.5}$$

Similarly, the shortest distance from the next image feature vector $\mathbf{s}(t_{k+1})$ to the line connecting the current image feature vector $\mathbf{s}(t_k)$ to the desired image feature vector $\mathbf{s}^*$ is denoted by $d_s(t_k)$, as illustrated in Fig. 1(b), and computed as

$$d_s(t_k) = \frac{\|\mathbf{e}_s(t_{k+1}) \times \mathbf{e}_s(t_k)\|}{\|\mathbf{e}_s(t_k)\|}, \tag{3.6}$$

or equivalently

$$d_s(t_k) = \|\mathbf{A}_{\mathbf{e}_s(t_k)}\mathbf{e}_s(t_{k+1})\|. \tag{3.7}$$

The image feature parameters error $\mathbf{e}_s(t_k)$ has been defined in (2.1).

The objective function of our optimization model at time t_k is formulated as

$$\min \alpha d_c(t_k) + d_s(t_k), \tag{3.8}$$

where α is a nonnegative weighting coefficient. If α is zero, the model is expected to perform similar to the IBVS method, i.e., the image features will move on straight lines toward the desired locations. Also, as α increases to infinity, the model is expected to perform

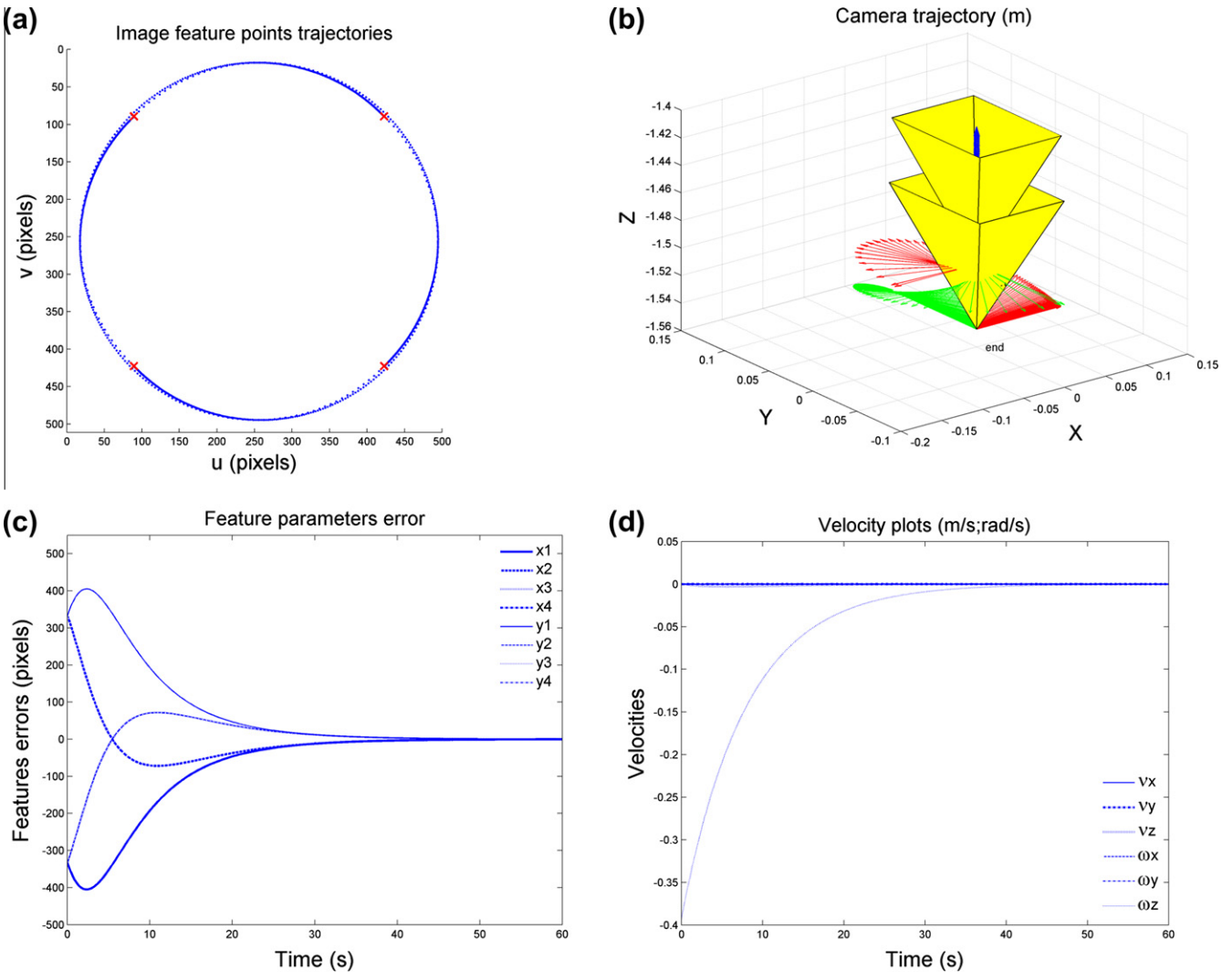

Fig. 10. Simulation results obtained by the PBVS controller for Case 3.

as the PBVS method, i.e., the camera will follow the shortest distance path toward the desired pose.

From (3.5) and (3.7), the objective function (3.8) becomes

$$\min \alpha\|\mathbf{A}_{\mathbf{e}_c(t_k)}\mathbf{e}_c(t_{k+1})\| + \|\mathbf{A}_{\mathbf{e}_s(t_k)}\mathbf{e}_s(t_{k+1})\|. \tag{3.9}$$

The variables $\mathbf{e}_c(t_{k+1})$ and $\mathbf{e}_s(t_{k+1})$ represent the camera and the image feature errors, respectively, at the next time instant, which need to be determined. The matrices $\mathbf{A}_{\mathbf{e}_c(t_k)}$ and $\mathbf{A}_{\mathbf{e}_s(t_k)}$ are computed at the current time instant prior to solving the model and fed to the model as input data.

In the following, we introduce and describe the constraints of the model. The constraints include the relations among the current and next camera poses, the current and next image feature points, and the current camera velocity. The current camera velocity $\mathbf{v}_c(t_k)$ is also a variable in the model. The image features visibility and camera velocity limitations are expressed as box constraints.

1. Error Reduction Constraint: Denoting the combined camera and image feature errors as $\mathbf{e}(t_k) = \begin{bmatrix}\mathbf{e}_c(t_k)\\ \mathbf{e}_s(t_k)\end{bmatrix}$, the constraint

$$\|\mathbf{e}(t_{k+1})\| \leqslant \lambda\|\mathbf{e}(t_k)\| \tag{3.10}$$

ensures that the norm of the combined error is strictly monotonically decreasing to zero with a rate $\lambda \in (0,1)$. This constraint is essential for the convergence of our approach. Note that either the norm of the camera error or the norm of the image feature error might not decrease monotonically, but both have to converge to zero.

2. Image Feature Error Constraint: This constraint relates the camera velocity with the current and next image feature points, as introduced in (2.3). Subtracting the desired image feature vector $\mathbf{s}^*$ from both sides of (2.3), we obtain

$$\mathbf{e}_s(t_{k+1}) - \mathbf{L}(t_k, \mathbf{s}(t_k))\mathbf{v}_c(t_k)\Delta t_k = \mathbf{e}_s(t_k). \tag{3.11}$$

3. Camera Error Constraint: This constraint relates the camera velocity with the current and next camera poses with respect to the world frame, as introduced in (2.10). Subtracting the desired camera pose $\mathcal{P}_c^*$ from both sides of (2.10), we obtain

$$\mathbf{e}_c(t_{k+1}) - \mathbf{G}_c(t_k)\mathbf{v}_c(t_k)\Delta t_k = \mathbf{e}_c(t_k). \tag{3.12}$$

4. Image Features Visibility Constraint: This constraint is to ensure that the image feature points always stay in the field-of-view of the camera. Therefore, each image feature point $\mathbf{p}_i\,(t_{k+1})$ should stay in

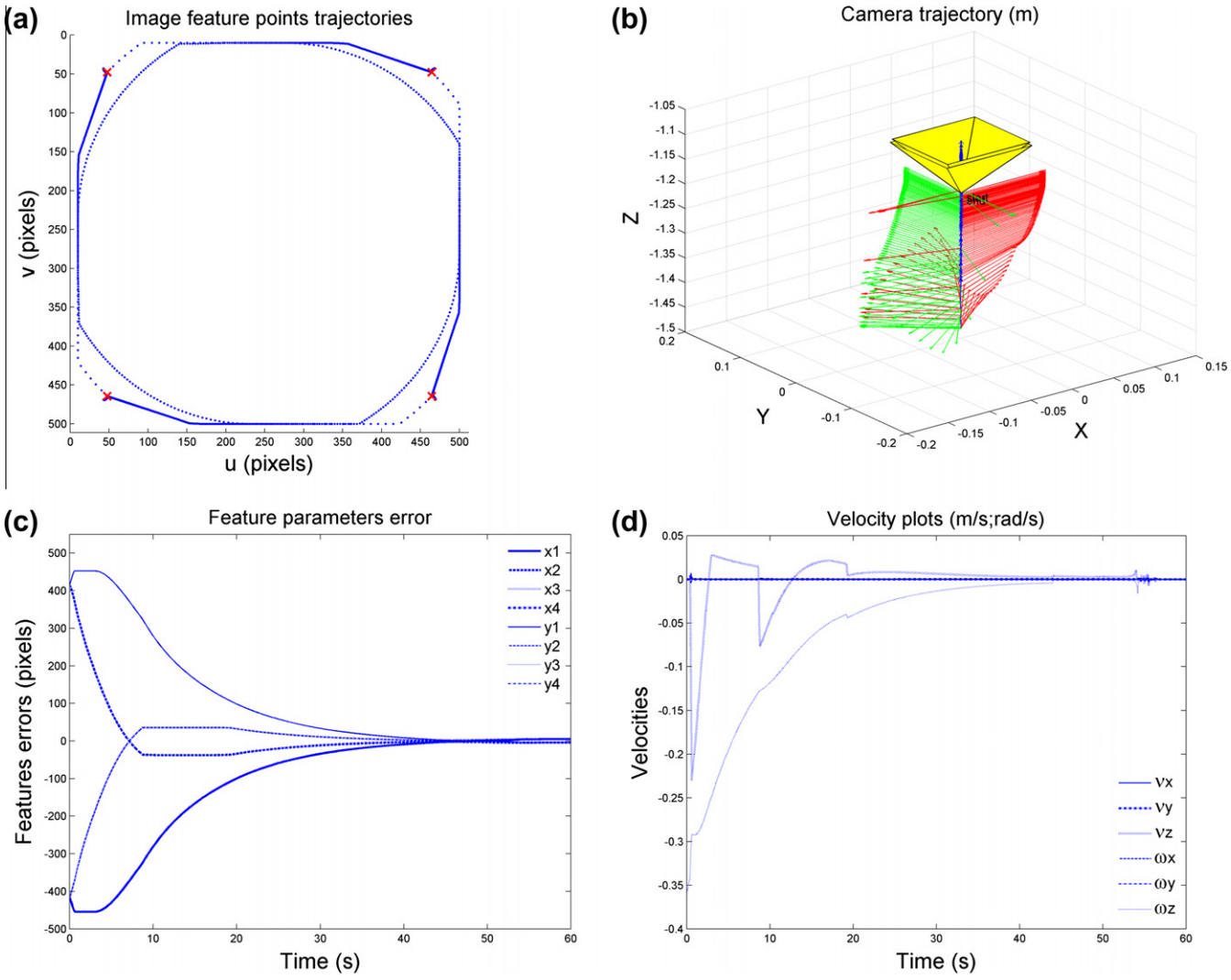

Fig. 11. Simulation results obtained by the SOCO method for Case 4.

a rectangular screen specified by two points $\underline{\mathbf{p}}$ and $\bar{\mathbf{p}}$, i.e., $\underline{\mathbf{p}} \leqslant \mathbf{p}_i(t_{k+1}) \leqslant \bar{\mathbf{p}}$, for $i = 1,2,\ldots,n$. Combining all these constraints into one constraint, we can express the image features visibility constraint as

$$\underline{\mathbf{s}} \leqslant \mathbf{s}(t_{k+1}) \leqslant \bar{\mathbf{s}}, \tag{3.13}$$

where $\underline{\mathbf{s}} = \begin{bmatrix} \underline{\mathbf{p}}^T & \underline{\mathbf{p}}^T & \ldots & \underline{\mathbf{p}}^T \end{bmatrix}^T$ and $\bar{\mathbf{s}} = \begin{bmatrix} \bar{\mathbf{p}}^T & \bar{\mathbf{p}}^T & \ldots & \bar{\mathbf{p}}^T \end{bmatrix}^T$ are two vectors in $\mathbb{R}^{2n}$. Note that for any two vectors $\mathbf{u}$ and $\mathbf{w}$, the inequality $\mathbf{u} \leqslant \mathbf{w}$ used in (3.13) means $u_i \leqslant w_i$, for $i = 1,2,\ldots,2n$. Denoting $\underline{\mathbf{e}}_s = \underline{\mathbf{s}} - \mathbf{s}^*$ and $\bar{\mathbf{e}}_s = \bar{\mathbf{s}} - \mathbf{s}^*$, we can express (3.13) in terms of the variable $\mathbf{e}_s(t_{k+1})$ as

$$\underline{\mathbf{e}}_s \leqslant \mathbf{e}_s(t_{k+1}) \leqslant \bar{\mathbf{e}}_s. \tag{3.14}$$

5. Camera Velocity Limitations Constraint: This constraint, which limits the camera spatial velocity in a box represented by vectors $\underline{\mathbf{v}}_c$ and $\bar{\mathbf{v}}_c$, impeding it from achieving unrealizable values, is given as

$$\underline{\mathbf{v}}_c \leqslant \mathbf{v}_c(t_k) \leqslant \bar{\mathbf{v}}_c. \tag{3.15}$$

All the listed constraints (1)–(5) are either linear or second-order conic. Therefore, the problem can be modeled as the following SOCO model:

$$\min \alpha\tau_c + \tau_s \tag{3.16}$$

subject to

$$\|\mathbf{e}(t_{k+1})\| \leqslant \lambda\|\mathbf{e}(t_k)\|, \tag{3.17}$$
$$\mathbf{e}_s(t_{k+1}) - \mathbf{L}(t_k, \mathbf{s}(t_k))\mathbf{v}_c(t_k)\Delta t_k = \mathbf{e}_s(t_k), \tag{3.18}$$
$$\mathbf{e}_c(t_{k+1}) - \mathbf{G}_c(t_k)\mathbf{v}_c(t_k)\Delta t_k = \mathbf{e}_c(t_k), \tag{3.19}$$
$$\|\mathbf{A}_{\mathbf{e}_c(t_k)}\mathbf{e}_c(t_{k+1})\| \leqslant \tau_c, \tag{3.20}$$
$$\|\mathbf{A}_{\mathbf{e}_s(t_k)}\mathbf{e}_s(t_{k+1})\| \leqslant \tau_s, \tag{3.21}$$
$$\underline{\mathbf{e}}_s \leqslant \mathbf{e}_s(t_{k+1}) \leqslant \bar{\mathbf{e}}_s, \tag{3.22}$$
$$\underline{\mathbf{v}}_c \leqslant \mathbf{v}_c(t_k) \leqslant \bar{\mathbf{v}}_c. \tag{3.23}$$

The objective function (3.16) of the model together with the conic constraints (3.20) and (3.21) is equivalent to (3.9), which minimizes a weighted average length of the camera trajectory and the image feature trajectories.

In contrast to approaches that address path planning problems (e.g., [10,24]), our proposed SOCO model solves the visual servoing control problem in an on-line manner. Starting from an initial pose, we solve a sequence of the SOCO models at time instances $\{t_k\}_{k=1}^{\infty}$ to obtain sequences of the camera velocities $\{\mathbf{v}_c(t_k)\}_{k=1}^{\infty}$,

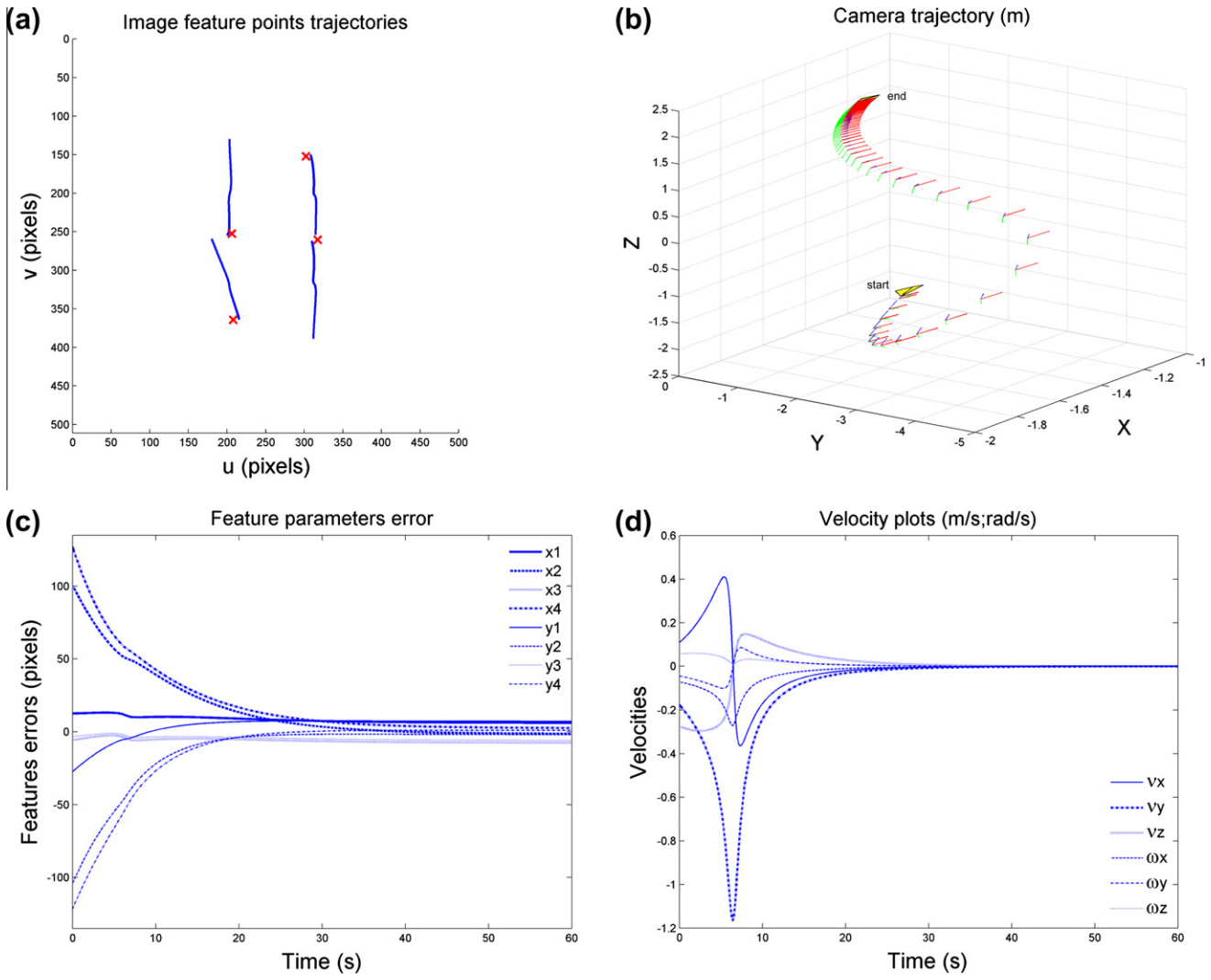

Fig. 12. Simulation results obtained by the IBVS controller for Case 5.

the camera errors $\{\mathbf{e}_c(t_k)\}_{k=1}^{\infty}$, and the image feature errors $\{\mathbf{e}_s(t_k)\}_{k=1}^{\infty}$. As shown in the next section, these sequences converge to zero, implying that both the camera poses and the image feature parameters converge to their corresponding desired counterparts.

Compared to the traditional visual servoing approaches, the proposed approach is computationally more expensive than the classical control methods. The reason is that we solve a sequence of SOCO problems at discrete time instants. However, with the advances in the computer technology, its high-bandwidth implementation should not be a problem, since very efficient polynomial-time algorithms exist for solving SOCO problems.

4. Convergence and feasibility analysis

4.1. Convergence of the camera pose

In this subsection, we provide two assumptions under which the SOCO model is feasible at any time instant. Our approach is convergent if the SOCO model is feasible at any time instant as described next.

Constraint (3.17) implies

$$\|\mathbf{e}(t_k)\| \leqslant \lambda^{k-1}\|\mathbf{e}(t_1)\|. \tag{4.1}$$

Therefore, assuming that $\mathbf{e}(t_1) \neq \mathbf{0}$, we have $\mathbf{e}(t_k) \to \mathbf{0}$ as $k \to \infty$, implying that $\mathcal{P}_c(t_k) \to \mathcal{P}_c^*$ and $\mathbf{s}(t_k) \to \mathbf{s}^*$.

We now present assumptions to guarantee the feasibility of the SOCO model. The first assumption is to ensure that the matrix $\mathbf{T}(t)$ defined in (2.7) does not get close to singularity, in a neighborhood of the desired pose. Therefore, not only does $\mathbf{T}^{-1}(t)$ exist in that neighborhood, but also its norm is bounded above, as will be shown later.

Assumption 1. In a neighborhood of the desired pose, $\theta(t)$ is well separated from $\pm\pi/2$. More precisely, there exists $\varepsilon > 0$ such that $|\cos\theta(t)| \geqslant \varepsilon$ in that neighborhood. Henceforth, any three of the feature points should not be collinear in the image plane.

The second assumption is to ensure that the norm of the interaction matrix $\mathbf{L}(t)$ is bounded above in a neighborhood of the desired pose.

Assumption 2. In a neighborhood of the desired pose, the interaction matrix $\mathbf{L}(t)$ satisfies $\|\mathbf{L}(t)\| \leqslant M$.

Before continuing with the feasibility analysis, we note that equations (3.18) and (3.19) can be combined as

$$\mathbf{e}(t_{k+1}) - \widetilde{\mathbf{L}}(t_k)\mathbf{v}_c(t_k)\Delta t_k = \mathbf{e}(t_k), \tag{4.2}$$

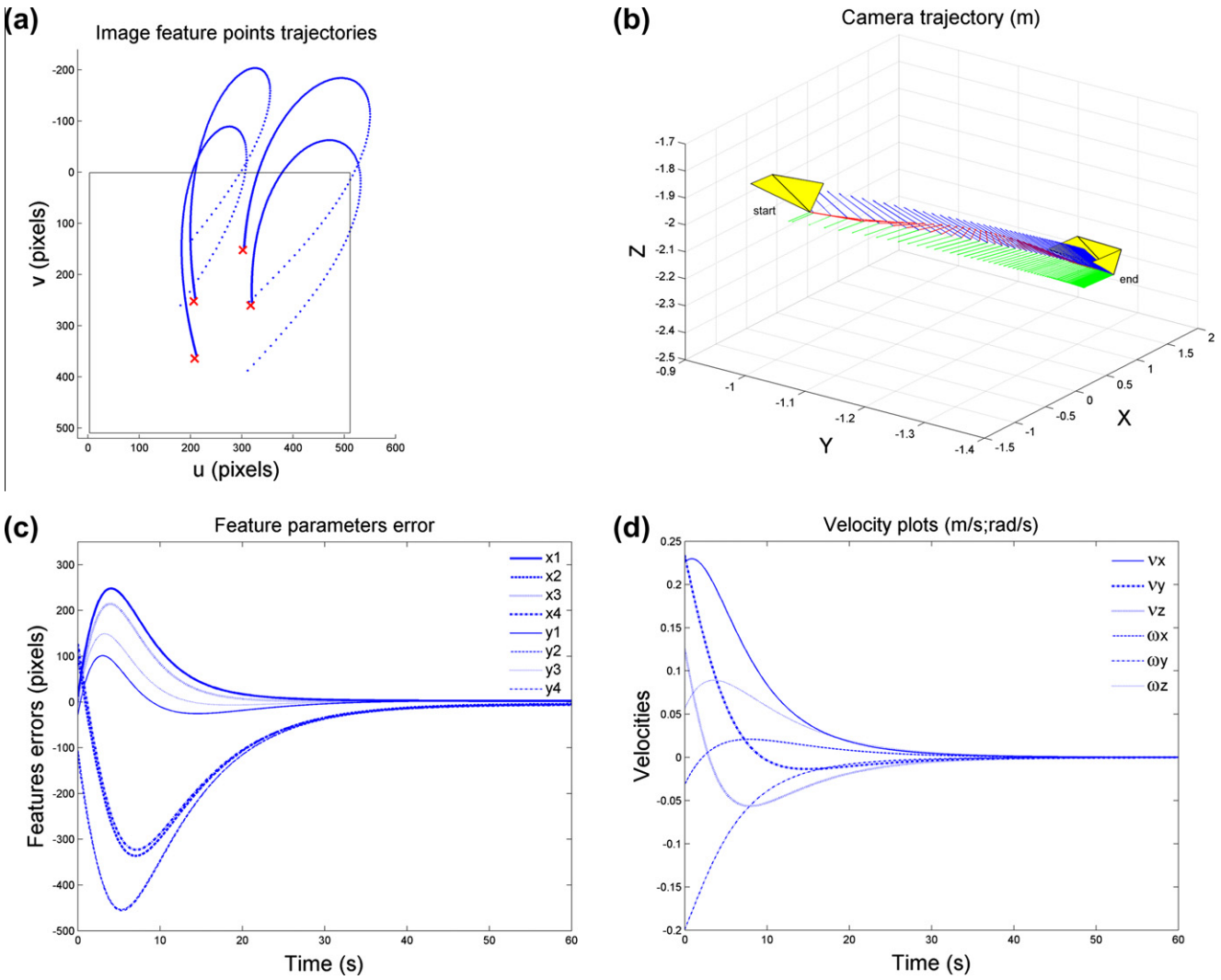

Fig. 13. Simulation results obtained by the PBVS controller for Case 5. Image feature trajectories leave the camera field-of-view.

where $\widetilde{\mathbf{L}}(t_k) = \begin{bmatrix} \mathbf{G}_c(t_k) \\ \mathbf{L}(t_k, \mathbf{s}(t_k)) \end{bmatrix}$ is a combined interaction matrix of size $(2n+6)\times 6$. For convenience, the time sequence indicator t_k will be omitted in the further notation, when no confusions occur.

The following lemma provides some useful inequalities and states that the norm of the matrix $\mathbf{G}_c$ is bounded above if Assumption 1 holds.

Lemma 1. *We have*

$$\text{a)}\ \|\mathbf{G}_c\| \leqslant \|\mathbf{T}^{-1}\|. \tag{4.3}$$

Moreover, if Assumption 1 holds, we get

$$\text{b)}\ \|\mathbf{G}_c\| \leqslant \sqrt{3}\varepsilon^{-1}. \tag{4.4}$$

Proof. For the inequality (a) recall that for any matrix $\mathbf{A}$ of rank r, we have $\|\mathbf{A}\| \leqslant \|\mathbf{A}\|_F \leqslant \sqrt{r}\|\mathbf{A}\|$. Thus, we have

$$\|\mathbf{T}^{-1}\| \geqslant \frac{\|\mathbf{T}^{-1}\|_F}{\sqrt{3}} = \frac{\sqrt{2+\cos^2\theta}}{\sqrt{3}|\cos\theta|} \geqslant 1. \tag{4.5}$$

Now, let $\mathbf{x} = \begin{bmatrix} \mathbf{x}^u \\ \mathbf{x}^l \end{bmatrix}$ be a nonzero vector in $\mathbb{R}^6$, where $\mathbf{x}^u$ and $\mathbf{x}^l$ represent the first three and the last three components of the vector $\mathbf{x}$, respectively. Using (2.9), for any nonzero vector $\mathbf{x}$, we can write

$$\frac{\|\mathbf{G}_c\mathbf{x}\|}{\|\mathbf{x}\|} = \frac{\left\| \begin{bmatrix} \mathbf{R}_c^w\mathbf{x}^u \\ \mathbf{T}^{-1}\mathbf{R}_c^w\mathbf{x}^l \end{bmatrix} \right\|}{\|\mathbf{x}\|} = \frac{\sqrt{\|\mathbf{R}_c^w\mathbf{x}^u\|^2 + \left\|\mathbf{T}^{-1}\mathbf{R}_c^w\mathbf{x}^l\right\|^2}}{\|\mathbf{x}\|} \leqslant \frac{\sqrt{\|\mathbf{R}_c^w\mathbf{x}^u\|^2 + \|\mathbf{T}^{-1}\|^2\|\mathbf{R}_c^w\mathbf{x}^l\|^2}}{\|\mathbf{x}\|}$$
$$= \frac{\sqrt{\|\mathbf{x}^u\|^2 + \|\mathbf{T}^{-1}\|^2\|\mathbf{x}^l\|^2}}{\|\mathbf{x}\|} \leqslant \|\mathbf{T}^{-1}\|. \tag{4.6}$$

Note that (4.6) follows from the orthogonality of the matrix $\mathbf{R}_c^w$, and from (4.5).

To show inequality (b), we write

$$\|\mathbf{T}^{-1}\| \leqslant \|\mathbf{T}^{-1}\|_F = \frac{\sqrt{2+\cos^2\theta}}{|\cos\theta|} \leqslant \frac{\sqrt{3}}{|\cos\theta|}. \tag{4.7}$$

Therefore, the result follows from Assumption 1 and inequality (b). □

Lemma 2. *Let Assumptions 1 and 2 hold. Then, we have*

$$\frac{1}{3}\mathbf{I} \leqslant \widetilde{\mathbf{L}}^T\widetilde{\mathbf{L}} \leqslant M'\mathbf{I}, \tag{4.8}$$

where $M' = M^2 + 3\varepsilon^{-2}$.

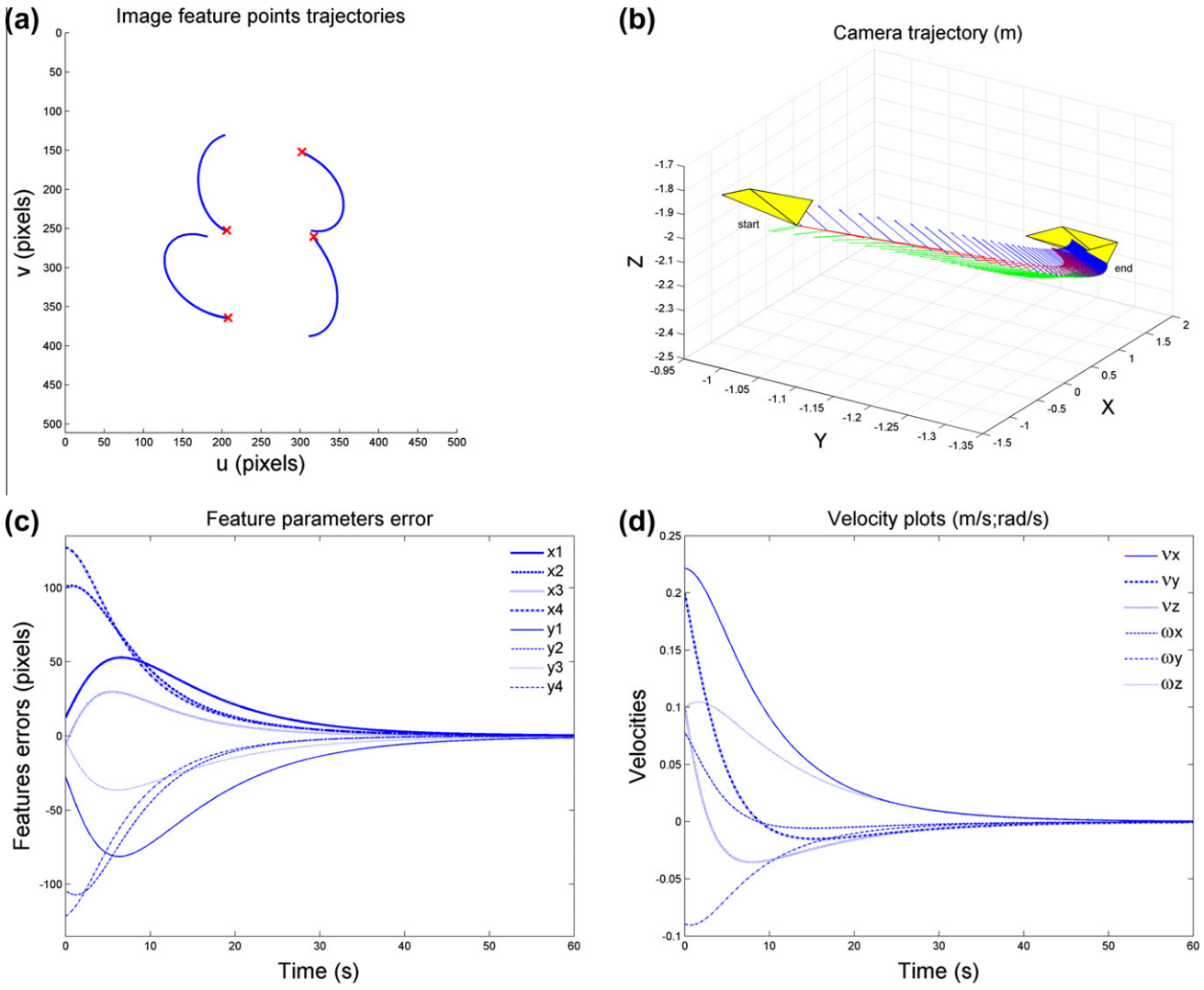

Fig. 14. Simulation results obtained by the SOCO method for Case 5.

Proof. First, we prove the right hand side inequality. From the definition of $\widetilde{\mathbf{L}}$, Assumption 2, and Lemma 1, we can write

$$\widetilde{\mathbf{L}}^T\widetilde{\mathbf{L}} = \mathbf{G}_c^T\mathbf{G}_c + \mathbf{L}^T\mathbf{L} \leqslant \left(\|\mathbf{G}_c\|^2 + \|\mathbf{L}\|^2\right)\mathbf{I} \leqslant \left(3\varepsilon^{-2} + M^2\right)\mathbf{I}. \tag{4.9}$$

To prove the left hand side inequality, we start from the fact that $\widetilde{\mathbf{L}}^T\widetilde{\mathbf{L}} \geqslant \mathbf{G}_c^T\mathbf{G}_c$. Using similar analysis as in the proof of Lemma 1, for any nonzero vector $\mathbf{x}$, we can write

$$\|\mathbf{G}_c\mathbf{x}\| = \sqrt{\|\mathbf{x}^u\|^2 + \left\|\mathbf{T}^{-1}\mathbf{R}_c^w\mathbf{x}^l\right\|^2}. \tag{4.10}$$

Now, letting $\mathbf{y} = \mathbf{T}^{-1}\mathbf{R}_c^w\mathbf{x}^l$, we obtain $\mathbf{T}\mathbf{y} = \mathbf{R}_c^w\mathbf{x}^l$. Taking the norm of the both sides and using the fact that the matrix $\mathbf{R}_c^w$ is orthogonal, we obtain $\|\mathbf{y}\| \geqslant \|\mathbf{T}\|^{-1}\|\mathbf{x}^l\|$. Therefore, we have

$$\|\mathbf{G}_c\mathbf{x}\| \geqslant \sqrt{\|\mathbf{x}^u\|^2 + \|\mathbf{T}\|^{-2}\|\mathbf{x}^l\|^2}. \tag{4.11}$$

Since $\|\mathbf{T}\| \leqslant \|\mathbf{T}\|_F = \sqrt{3}$, we get

$$\|\mathbf{G}_c\mathbf{x}\| \geqslant \sqrt{\|\mathbf{x}^u\|^2 + \frac{1}{3}\|\mathbf{x}^l\|^2} \geqslant \frac{1}{\sqrt{3}}\|\mathbf{x}\| \tag{4.12}$$

which implies that $\mathbf{G}_c^T\mathbf{G}_c \geqslant \frac{1}{3}\mathbf{I}$. □

Corollary 1. *Matrix $\widetilde{\mathbf{L}}$ satisfies the following inequalities:*

1. $$\frac{1}{M'}\mathbf{I} \leqslant (\widetilde{\mathbf{L}}^T\widetilde{\mathbf{L}})^{-1} \leqslant 3\mathbf{I}. \tag{4.13}$$

2. $$\frac{1}{\sqrt{3}} \leqslant \|\widetilde{\mathbf{L}}\| \leqslant \sqrt{M'}. \tag{4.14}$$

Now, we are in a position to prove the feasibility of the SOCO model. Recall that our method starts from an initial camera pose and solves the SOCO model at every iteration to find the next camera pose. As discussed earlier, the method is convergent if the SOCO model at every iteration is feasible.

4.2. Feasibility of the SOCO model

We have shown that our approach is convergent if and only if the SOCO model at every iteration is feasible. In this section, we show that under Assumptions 1 and 2, the SOCO model at every iteration is feasible.

From (4.2) it follows that for a given $\lambda \in (0,1)$, the SOCO model is feasible if and only if there exists a camera velocity $\mathbf{v}_c$ such that $\underline{\mathbf{v}}_c \leqslant \mathbf{v}_c \leqslant \bar{\mathbf{v}}_c$, and

$$\|\mathbf{e} + \widetilde{\mathbf{L}}\mathbf{v}_c\Delta t\| \leqslant \lambda\|\mathbf{e}\|. \tag{4.15}$$

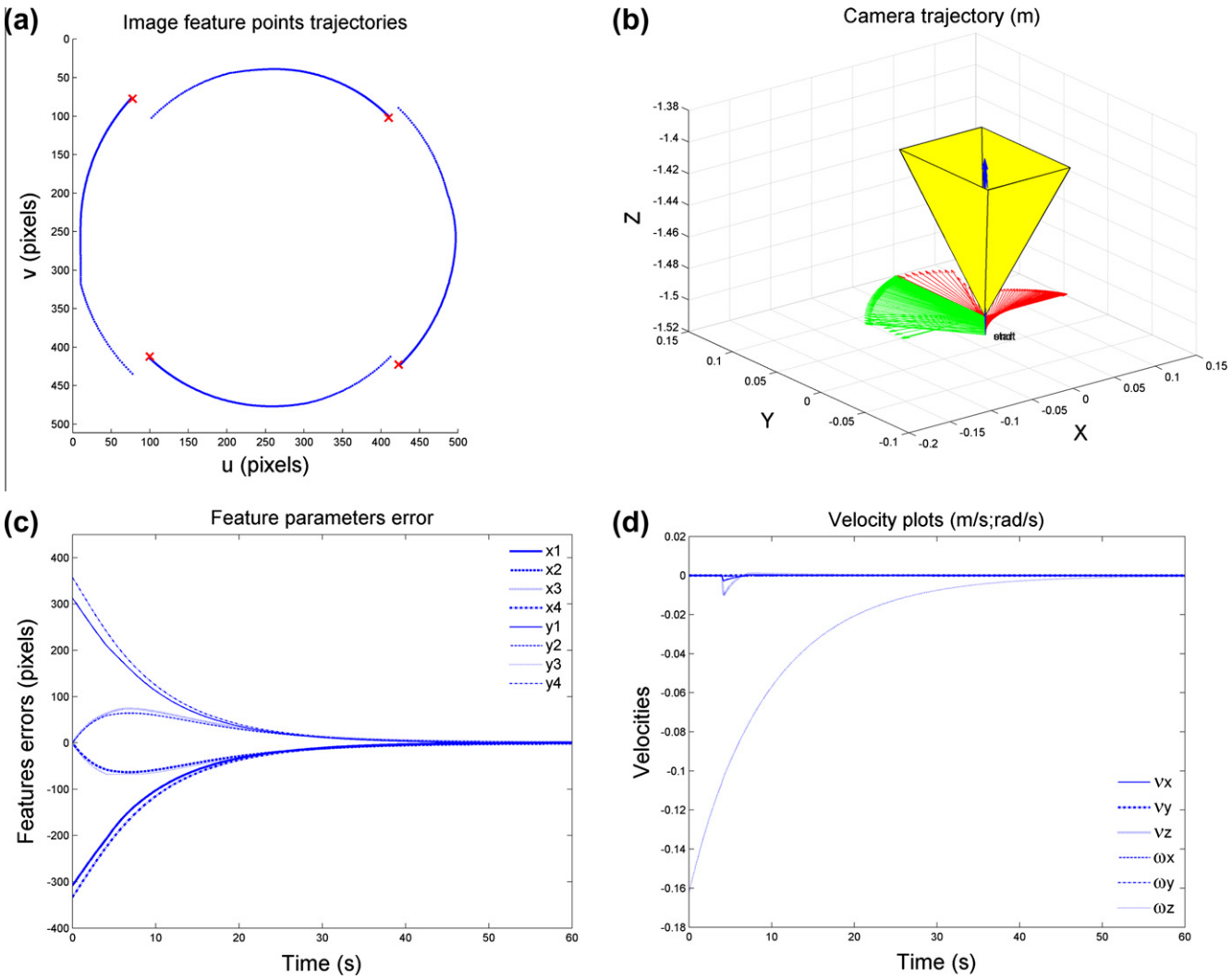

Fig. 15. Simulation results obtained by the SOCO method for Case 6.

To prove the existence of such a velocity, we start by the following lemma.

Lemma 3. *Let Assumption 2 hold. Then, we have*

$$\frac{1}{3\sqrt{M'}}\|\mathbf{e}\| \leqslant \|\widetilde{\mathbf{L}}^T\mathbf{e}\| \leqslant \sqrt{M'}\|\mathbf{e}\|. \tag{4.16}$$

Proof. From Corollary 1, we get

$$\|\widetilde{\mathbf{L}}^T\mathbf{e}\| \leqslant \|\widetilde{\mathbf{L}}^T\|\|\mathbf{e}\| \leqslant \sqrt{M'}\|\mathbf{e}\|. \tag{4.17}$$

Now, we prove the left hand side inequality. If $\mathbf{e} = \mathbf{0}$, then the result obviously follows. For $\mathbf{e} \neq \mathbf{0}$, we can restrict the neighborhood around the desired pose so that there exist a camera velocity $\mathbf{v}_c$ to reach the desired pose from the current pose. This camera velocity would imply a variation of error as $\mathbf{0} - \mathbf{e} = \widetilde{\mathbf{L}}\mathbf{v}_c\Delta t$. Hence $-\widetilde{\mathbf{L}}^T\mathbf{e} = \widetilde{\mathbf{L}}^T\widetilde{\mathbf{L}}\mathbf{v}_c\Delta t$, and, using Lemma 2, we obtain

$$\|\widetilde{\mathbf{L}}^T\mathbf{e}\| = \Delta t\sqrt{\mathbf{v}_c^T(\widetilde{\mathbf{L}}^T\widetilde{\mathbf{L}})^2\mathbf{v}_c} \geqslant \frac{\Delta t}{3}\|\mathbf{v}_c\|. \tag{4.18}$$

Using Lemma 2 again, we obtain

$$\|\mathbf{e}\| = \Delta t\|\widetilde{\mathbf{L}}\mathbf{v}_c\| = \Delta t\sqrt{\mathbf{v}_c^T\widetilde{\mathbf{L}}^T\widetilde{\mathbf{L}}\mathbf{v}_c} \leqslant \Delta t\sqrt{M'}\|\mathbf{v}_c\|. \tag{4.19}$$

From (4.18) and (4.19), we get

$$\|\widetilde{\mathbf{L}}^T\mathbf{e}\| \geqslant \frac{1}{3\sqrt{M'}}\|\mathbf{e}\|, \tag{4.20}$$

which completes the proof. □

The following corollary immediately follows from (4.16).

Corollary 2. *Under Assumption 2, we have*

$$\widetilde{\mathbf{L}}^T\mathbf{e} = \mathbf{0} \Longleftrightarrow \mathbf{e} = \mathbf{0}. \tag{4.21}$$

This corollary states that in a neighborhood of the desired pose, there are no local minima. Next, we provide the main theorem, which states that if the combined camera and image feature error is not zero, then there exists a camera velocity so that the next error is decreased by a rate $\lambda \in (0,1)$.

Theorem 1. *Let Assumption 2 hold. Then, we have either* $\mathbf{e} = \mathbf{0}$ *or there exists a camera velocity* $\mathbf{v}_c$ *such that*

$$\|\mathbf{e} + \widetilde{\mathbf{L}}\mathbf{v}_c\Delta t\| \leqslant \lambda\|\mathbf{e}\|, \text{ for some } \lambda \in (0,1). \tag{4.22}$$

Proof. Assume that $\mathbf{e} \neq \mathbf{0}$. Let $\mathbf{v}_c = -\frac{1}{\Delta t}\widetilde{\mathbf{L}}^{\dagger}\mathbf{e}$, where $\widetilde{\mathbf{L}}^{\dagger}$ denotes the Moore–Penrose pseudo inverse of the matrix $\widetilde{\mathbf{L}}$. Then, using Corollary 1 and Lemma 3, we have

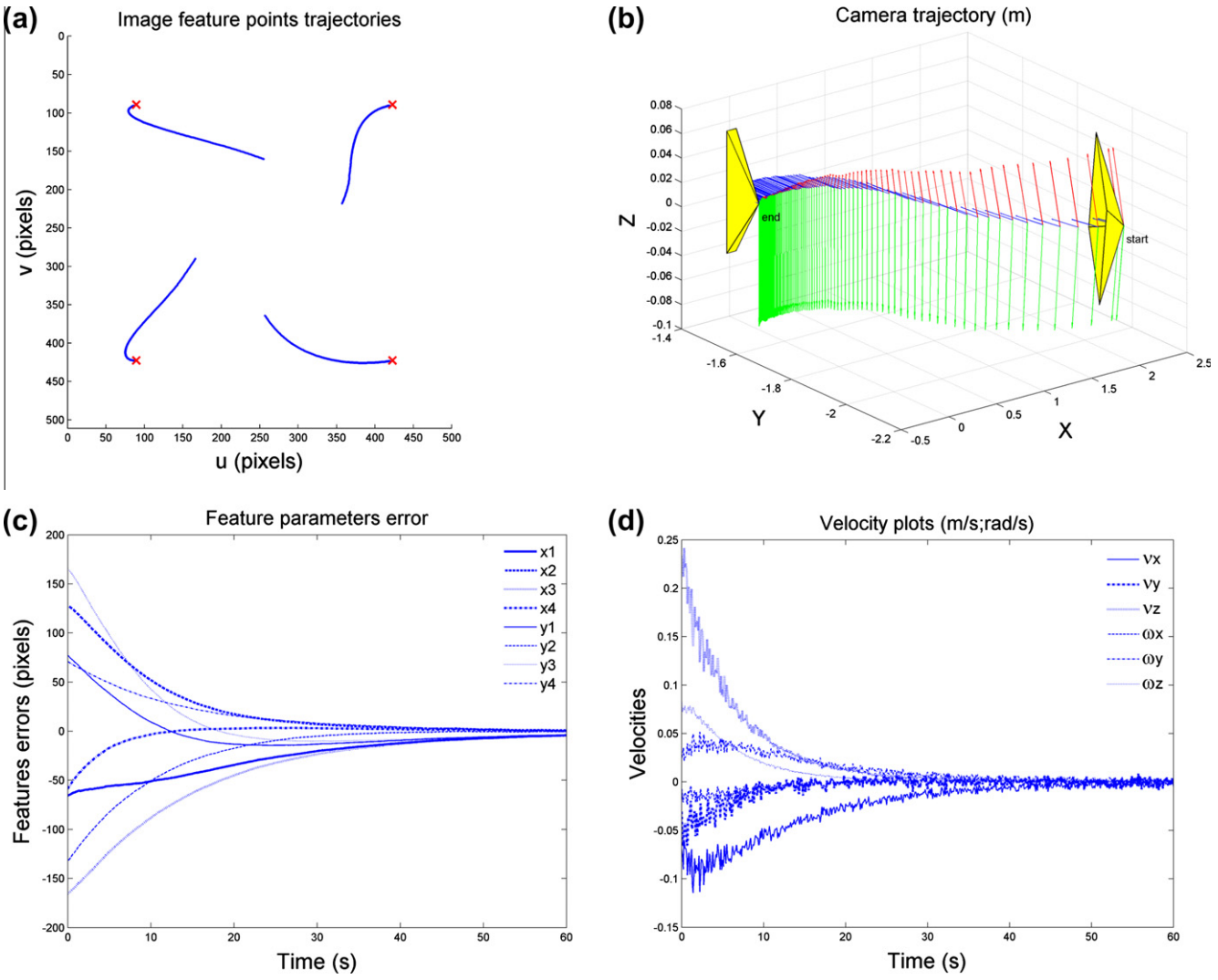

Fig. 16. Simulation results obtained by the SOCO method for Case 7.

$$\|\mathbf{e}+\widetilde{\mathbf{L}}\mathbf{v}_c\Delta t\| = \|\mathbf{e}-\widetilde{\mathbf{L}}\widetilde{\mathbf{L}}^{\dagger}\mathbf{e}\| = \sqrt{\mathbf{e}^T\mathbf{e}-\mathbf{e}^T\widetilde{\mathbf{L}}(\widetilde{\mathbf{L}}^T\widetilde{\mathbf{L}})^{-1}\widetilde{\mathbf{L}}^T\mathbf{e}}$$
$$\leqslant \sqrt{\mathbf{e}^T\mathbf{e}-\frac{1}{M'}\mathbf{e}^T\widetilde{\mathbf{L}}\widetilde{\mathbf{L}}^T\mathbf{e}} \leqslant \sqrt{\|\mathbf{e}\|^2-\frac{1}{M'}\|\widetilde{\mathbf{L}}^T\mathbf{e}\|^2}$$
$$\leqslant \sqrt{\|\mathbf{e}\|^2-\frac{1}{9M'^2}\|\mathbf{e}\|^2} = \sqrt{1-\frac{1}{9M'^2}}\|\mathbf{e}\|. \qquad (4.23)$$

Therefore, choosing $\lambda=\sqrt{1-\frac{1}{9M'^2}}$ completes the proof. □

In summary, we have shown that if Assumptions 1 and 2 hold, the SOCO model at each time instance is feasible, thus, it has an optimal solution. Therefore, in a neighborhood of the desired pose, our approach is guaranteed to converge to the desired pose. In addition, the sequence of velocity vectors, as solutions of the SOCO models, leads the camera to the desired pose.

It is worthwhile mentioning that the convergence of the image feature points to the desired ones does not necessarily imply the convergence of the camera pose to the desired one unless rather strong assumptions are considered. To show this, we first see from (2.1) and (2.8) that

$$\dot{\mathcal{P}}_c(t) = \mathbf{G}_c(t)\mathbf{v}_c(t) = \mathbf{G}_c(t)\mathbf{L}^{\dagger}(t,\mathbf{s}(t))\dot{\mathbf{s}}(t) \qquad (4.24)$$

Thus, in a small neighborhood of the desired pose, the camera error can be written as

$$\mathcal{P}_c(t)-\mathcal{P}_c^* = \int_t^{t^*}\mathbf{G}_c(t)\mathbf{L}^{\dagger}(t,\mathbf{s}(t))\dot{\mathbf{s}}(t)dt, \qquad (4.25)$$

where t is the time at the current pose and t^* is the time at the desired pose $\mathcal{P}_c^*=\mathcal{P}_c(t^*)$. Now, let Assumption 1 hold. Taking the norm from both sides of (4.25), then using (4.4), we obtain

$$\|\mathcal{P}_c(t)-\mathcal{P}_c^*\| \leqslant \frac{\sqrt{3}}{\varepsilon}\|\mathbf{L}^{\dagger}(t,\mathbf{s}(t))\|\|\mathbf{s}(t)-\mathbf{s}^*\|. \qquad (4.26)$$

Therefore, convergence of the image feature points implies the convergence of the camera pose providing that the norm of $\mathbf{L}^{\dagger}(t,\mathbf{s}(t))$ is bounded.

It can then be concluded that under Assumptions 1 and 2, both image features and camera constraints are critical for the convergence of the SOCO model. However, in case the norm of $\mathbf{L}^{\dagger}(t,\mathbf{s}(t))$ is bounded above, the camera constraints are unnecessary, although adding them enhances the performance of the model.

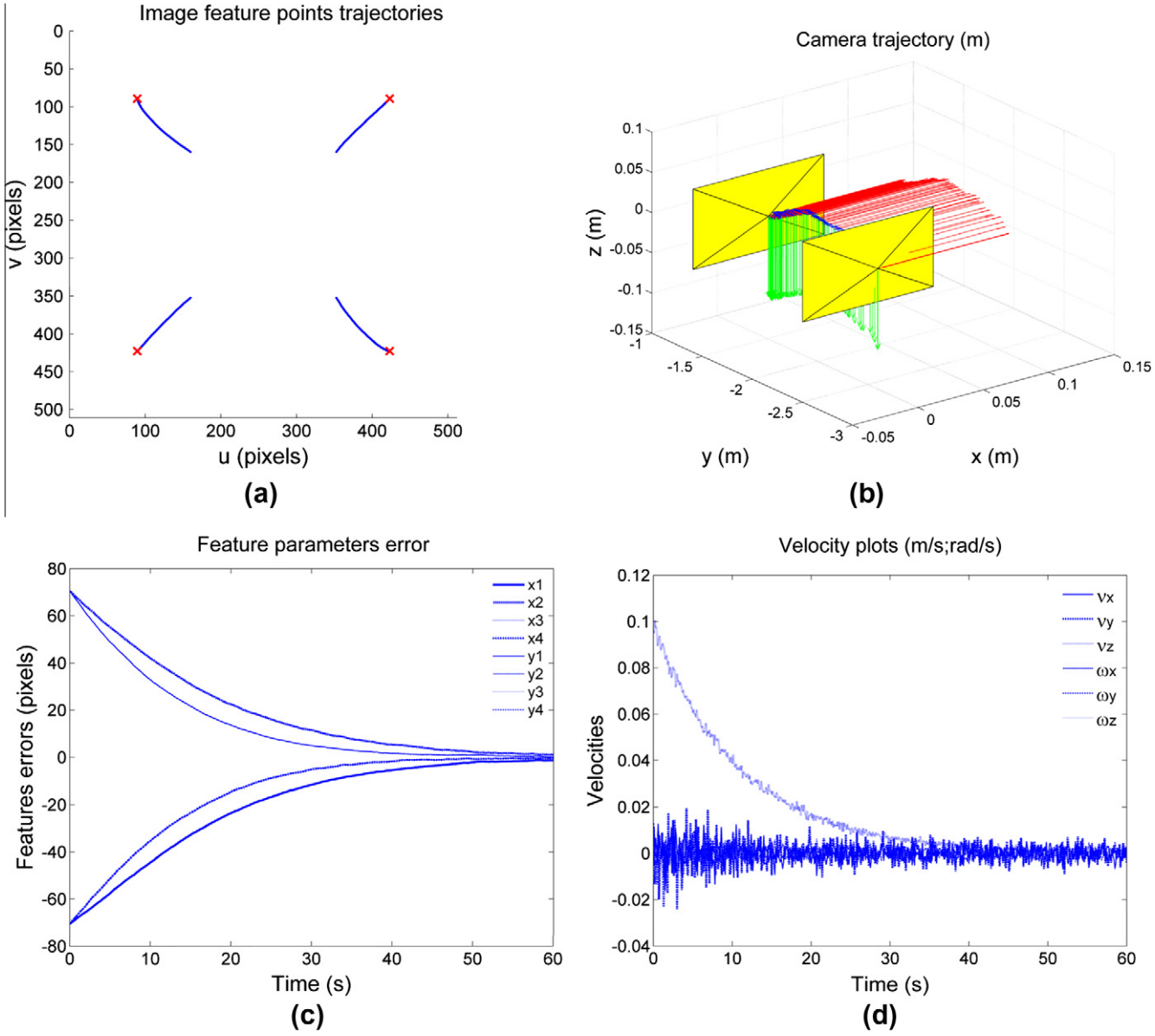

Fig. 17. Simulation results obtained by the SOCO method for Case 8.

5. Simulation results

In this section, we provide simulation results to compare our optimization-based method to the IBVS and PBVS approaches. The simulation results are obtained using the visual servoing toolbox in MATLAB [36], with the SOCO models being solved using SeDuMi [34], a freely available optimization software package.

Simulation of the vision-based controlled system is performed for a square shaped object with its vertices assumed as feature points. Coordinates of the feature points expressed in meters with respect to the object frame are $(0.25,-0.25,0)$, $(0.25,0.25,0)$, $(-0.25,0.25,0)$, and $(-0.25,-0.25,0)$ in meters. The value of the intrinsic camera parameters used in the simulation are chosen as $u_0 = v_0 = 256$, $fk_u = fk_v = 1000$, where u_0 and v_0 are pixel coordinates of the principal point, f is the focal length of the camera, and k_u and k_v are the scaling factors in horizontal and vertical axes directions. The sampling periods are assumed to be identical and $\Delta t_k = 0.1$ s. The parameter λ in (3.17) governs the speed of convergence of the model, with the higher values producing slower system response vs. smaller values of the parameter generating a faster response. For the simulations presented here, the value of the parameter λ was empirically selected as 0.99. The value of α in the cost function can vary in the range $[0,\infty)$. As α tends to zero, the SOCO method acts similar to the IBVS method, while as it tends to infinity, the SOCO method behaves as the PBVS method. Figs. 2 and 3 show the image feature trajectories and the camera trajectory as α varies in the set $\{0.01,0.1,1,10\}$. In the simulations which follow, the parameter α, which is problem dependent, is empirically chosen as

$$\alpha = \frac{\|\mathbf{s}(t_0) - \mathbf{s}^*\|}{\|\mathcal{P}_c(t_0) - \mathcal{P}_c^*\|}, \tag{5.1}$$

where t_0 is the initial time.

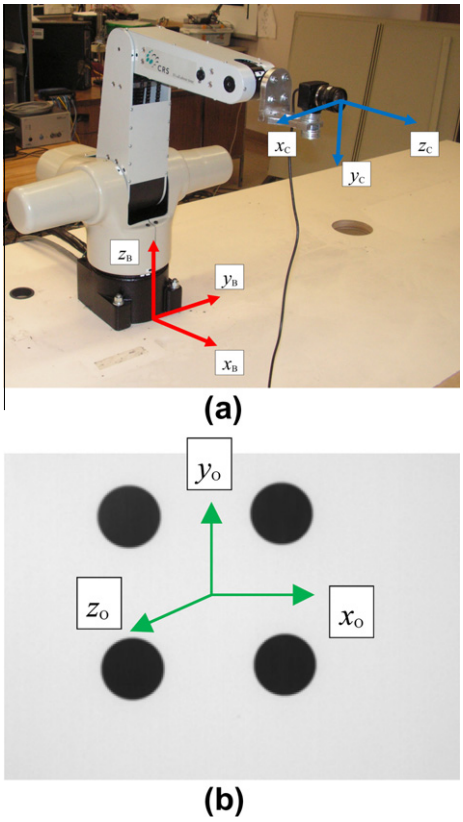

Fig. 18. (a) CRS-A255 robot and Firefly MV camera installed on the end point. The robot base frame and the camera frame are shown; (b) target object and the object frame.

We study the following eight cases.

Case 1. A combination of the translational and rotational movement of the camera.
Case 2. A pure 90° rotation of the camera around the optical axis in which the initial image feature points are close to the boundary of the screen observed by the camera.
Case 3. A pure 180° rotation of the camera around the optical axis.
Case 4. A pure 180° rotation of the camera around the optical axis in which the initial image feature points are close to the boundary of the screen observed by the camera.
Case 5. A camera configuration which causes the IBVS controller to be stuck in a local minimum.
Case 6. A non-coplanar case; the feature points, i.e., vertices of the object, do not belong to a plane.
Case 7. A noisy condition for combination of translational and rotational movement of the camera.
Case 8. A noisy condition for pure translational movement of the camera along the optical axis.

The image feature trajectories in Figs. 4–17 are shown in subfigures (a), where the marks ' × ' identify the desired feature positions. Subfigures (b–d) show the camera trajectory, the image feature errors, and the camera linear and angular velocities, respectively. The value of the weighting coefficient α in the objective function is set according to (5.1). In Cases 2, 3, and 4, the pitch angle θ at the initial camera poses with respect to the world frame is $\pi/2$, thus $\mathbf{T}$ is singular. As a remedy, we use a different orientation of the world frame, where the world frame coincides with the object frame. As a result the matrix $\mathbf{T}$ becomes nonsingular, thus $\mathbf{T}^{-1}$ can be computed.

For Case 1, the image feature points obtained by the SOCO method (Fig. 4a) move on almost straight lines in the image plane similar to those obtained by the IBVS method shown in Fig. 5a. In addition, the camera trajectory in the Cartesian space obtained by the SOCO (Fig. 4b) moves on a seemingly straight line from the initial pose toward the desired pose similar to that obtained by the PBVS method illustrated in Fig. 6b. Therefore, our approach takes the advantages of both the IBVS and PBVS methods and minimizes both the camera trajectory and the image feature trajectories simultaneously.

Note that a sampling period of 0.1 s and a control gain parameter of 0.125 are adopted for both the IBVS and PBVS controllers. For the PBVS method, planar homography is used for pose estimation of the object with respect to the camera, and roll-pitch-yaw angles are used as a representation of the camera orientation [2].

Case 2 is a pure 90° rotation of the camera around the optical axis with the initial and desired image feature points close to the boundary of the camera's field-of-view. As illustrated in Fig. 7, in our approach the image feature points start moving on circular trajectories without any camera retreat until they reach very close to the image boundary. Then, they move along the boundary on almost straight lines causing the camera retreat before converging to the desired pose. Values of 10 and 500 pixels were adopted in the constraint (3.22) for both the horizontal and vertical limits of the camera screen. For the same task, the IBVS method converges on the desired pose as the image feature points are forced by the control scheme to move on straight lines connecting the initial and desired image feature points. The PBVS method, however, fails to converge to the desired pose since the image feature points are forced by the controller to move on circular trajectories, thus, the camera loses the track of the image feature points.

Case 3, a pure 180° rotation of the camera around the optical axis, is a well-known example for which the IBVS method fails [17,37] since the camera retreats to infinity (Fig. 9), due to the singularity of the interaction matrix. In the literature this problem is referred to as Chaumette Conundrum [15]. A solution to this problem using the IBVS control laws is proposed in Nematollahi and Janabi-Sharifi [38] by introducing generalized forms of the interaction matrix. As shown in Fig. 10, the PBVS method converges to the desired pose with a slight camera retreat (Fig. 10b). Our method also converges to the desired pose with no camera retreat, but a slight steady-state error (Fig. 8c). If the value of the parameter λ is increased the steady-states error diminishes, but for the price of a slower response of the system.

Case 4 is a difficult case for both the IBVS and PBVS methods. The IBVS method fails when the camera is required to rotate about 180° as mentioned earlier. In Case 4, the initial and desired image feature points are close to the boundary of the camera's field-of-view. Thus, the PBVS method fails as the camera loses the tracks of the image feature points. However, as shown in Fig. 11, our approach converges to the desired pose with a slight steady-state error.

Case 5 represents an example where the classical IBVS control encounters a local minimum. As seen in Fig. 12c, the steady state behavior of the system is characterized by errors in the image space, while the velocity of the camera is zero (Fig. 12d). As a result, the camera pose did not converge toward the desired pose at the steady state. For the same case, the performance of the PBVS is shown in Fig. 13. This approach does not suffer from the problem of local minima; however the image features leave

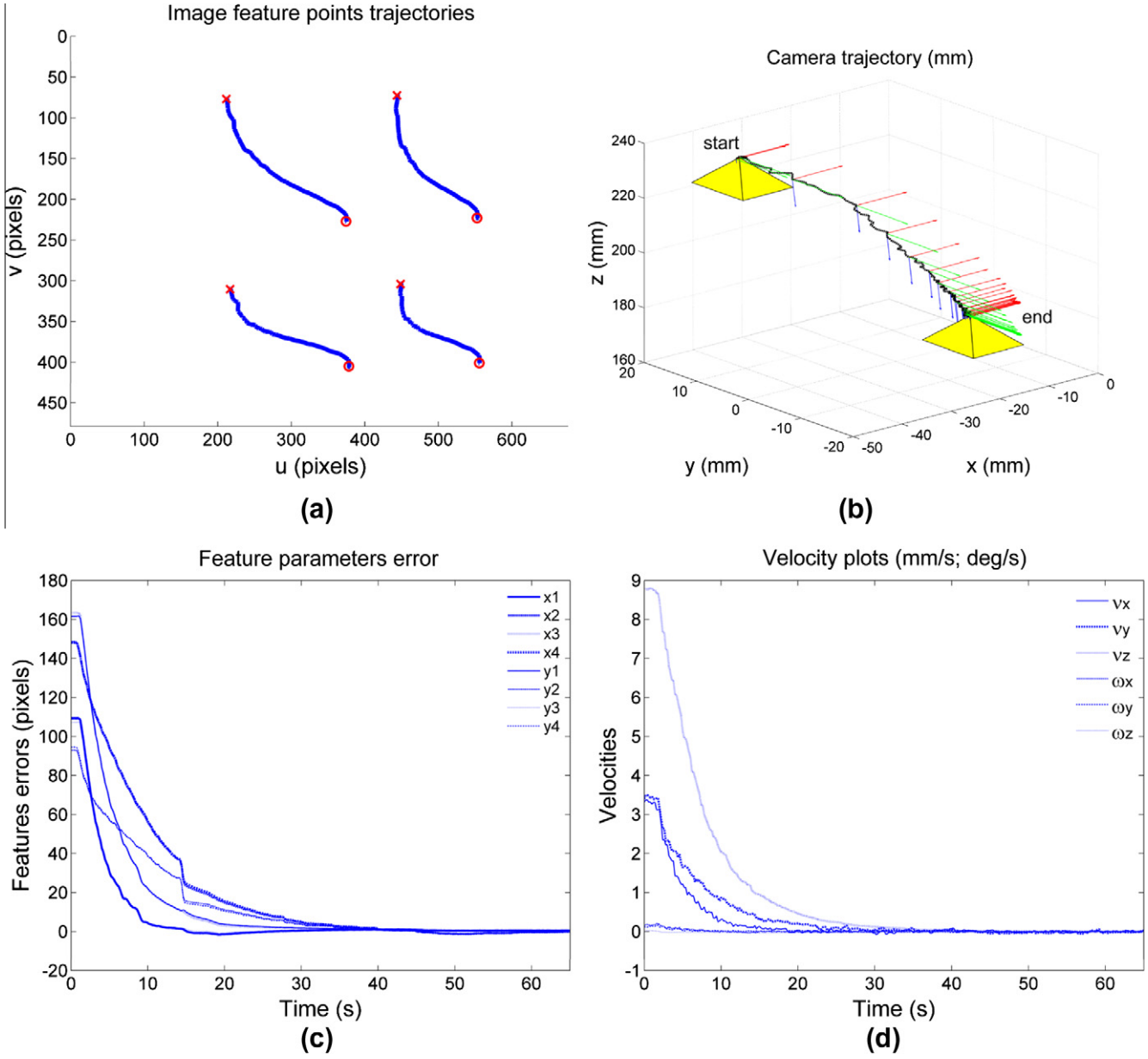

Fig. 19. Experimental results obtained by the SOCO method for Case 1.

the screen boundaries (Fig. 13a) rendering the control solution unsatisfactory. The response from the SOCO model is presented in Fig. 14. Due to the formulation of the error as a combination of image and Cartesian space errors, the problem of local minima has been avoided, and convergence of both the camera pose and the image features toward their desired targets has been achieved.

Case 6 is a pure 90° rotation of the camera around the optical axis with non-coplanar feature points. Coordinates of the feature points with respect to the object frame expressed in meters are chosen as $(0.25,-0.25,0.125)$, $(0.25,0.25,0)$, $(-0.25,0.25,0.1)$, and $(-0.25,-0.25,-0.1)$. Fig. 15 illustrates how our approach works in this case. Similarly, for the other considered cases of camera configurations, non-coplanarity of the image features did not have detrimental effects on the system's behavior.

Case 7 is the same as Case 1, but in a noisy environment. This case is to examine the robustness of the method. At each time instant, a uniformly distributed random noise in the interval $(-2,+2)$ pixels is added to the horizontal and vertical coordinates of all image feature points. Pixel coordinates of the principal point are chosen as $u_0 = 230$ and $v_0 = 276$, and the products of the focal length and scaling factors are assumed as $fk_u = 850$, $fk_v = 1150$. We assume the teaching-by-showing approach, i.e., the desired image features are observed by the camera before positioning, and they are used for the estimation of the desired camera pose. As shown in Fig. 16, the image feature trajectories and the camera trajectory, produced by our approach, are slightly altered by the noise. Nevertheless, the system is not destabilized, and the task goals of reaching the desired position and zeroing the errors are accomplished.

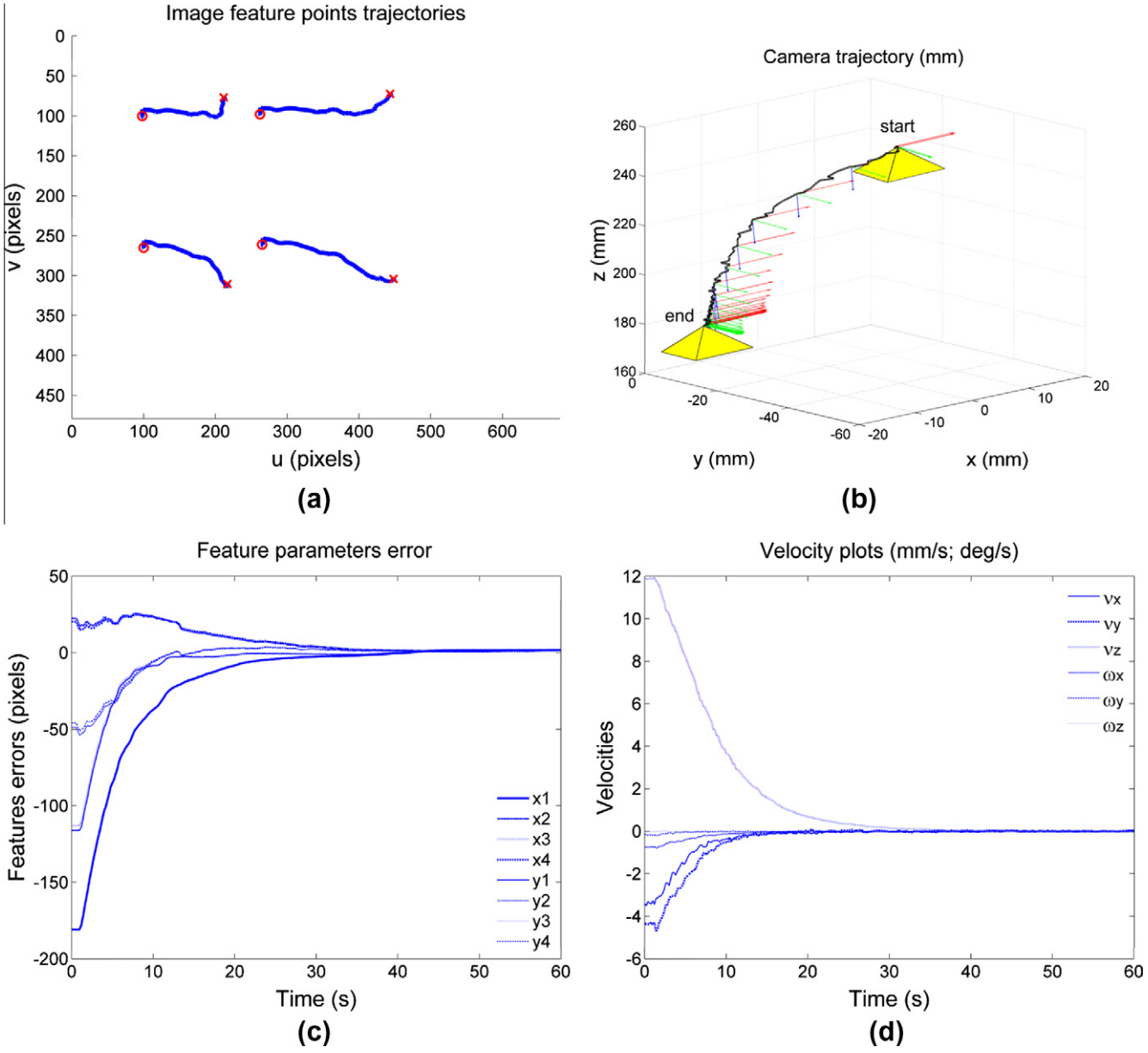

Fig. 20. Experimental results obtained by the SOCO method for Case 2.

Case 8 shows an additional example of the system in noisy conditions for pure translational movement of the camera. The same level of noise as in Case 7 was used. The simulation results are shown in Fig. 17, with the positioning task successfully performed, although some jerky camera motions are present due to the introduced noise and modeling errors.

6. Experiments

A set of experiments were conducted for evaluation of the proposed approach and comparison with the classical VS approaches. The experimental setup is shown in Fig. 18a, consisting of a 5-DOF desktop robot CRS-A255 and a Point Grey's 640 × 480 Firefly MV camera. QuaRC software package [39] in Simulink environment was employed for real-time control of the robot in an open-architecture mode. The Simulink model was employed as a server to communicate with a MATLAB instance as a client, on which the SeDuMi optimization was run. The sampling rate was limited to 200 ms, due to the slow performance of MATLAB codes in performing the optimization procedure at each time step. The intrinsic parameters of the camera obtained by calibration are as follows: principal point coordinates $u_0 = 296.54$, $v_0 = 266.04$, focal lentgh $f = 8.3$ mm, and scaling factors $fk_u = 1395.92$, $fk_v = 1397.88$. A target object with 4 co-planar circular features was employed, with the measurable parameters being the centroids of the dots (Fig. 18b). The coordinates of the features in the object frame are: $(15,15,0)$, $(15,-15,0)$, $(-15,-15,0)$, $(-15,15,0)$ mm. For the parameter λ value of 0.975 was adopted as a trade-off between the speed of convergence and accuracy.

Experiments for the following cases were performed:

Case 1. Pure translational movement of the camera, with the initial camera pose in the desired camera frame expressed in (mm, deg) of $\mathcal{P}_c^{c^*} = (-10,-15,65,0,0,0)$.

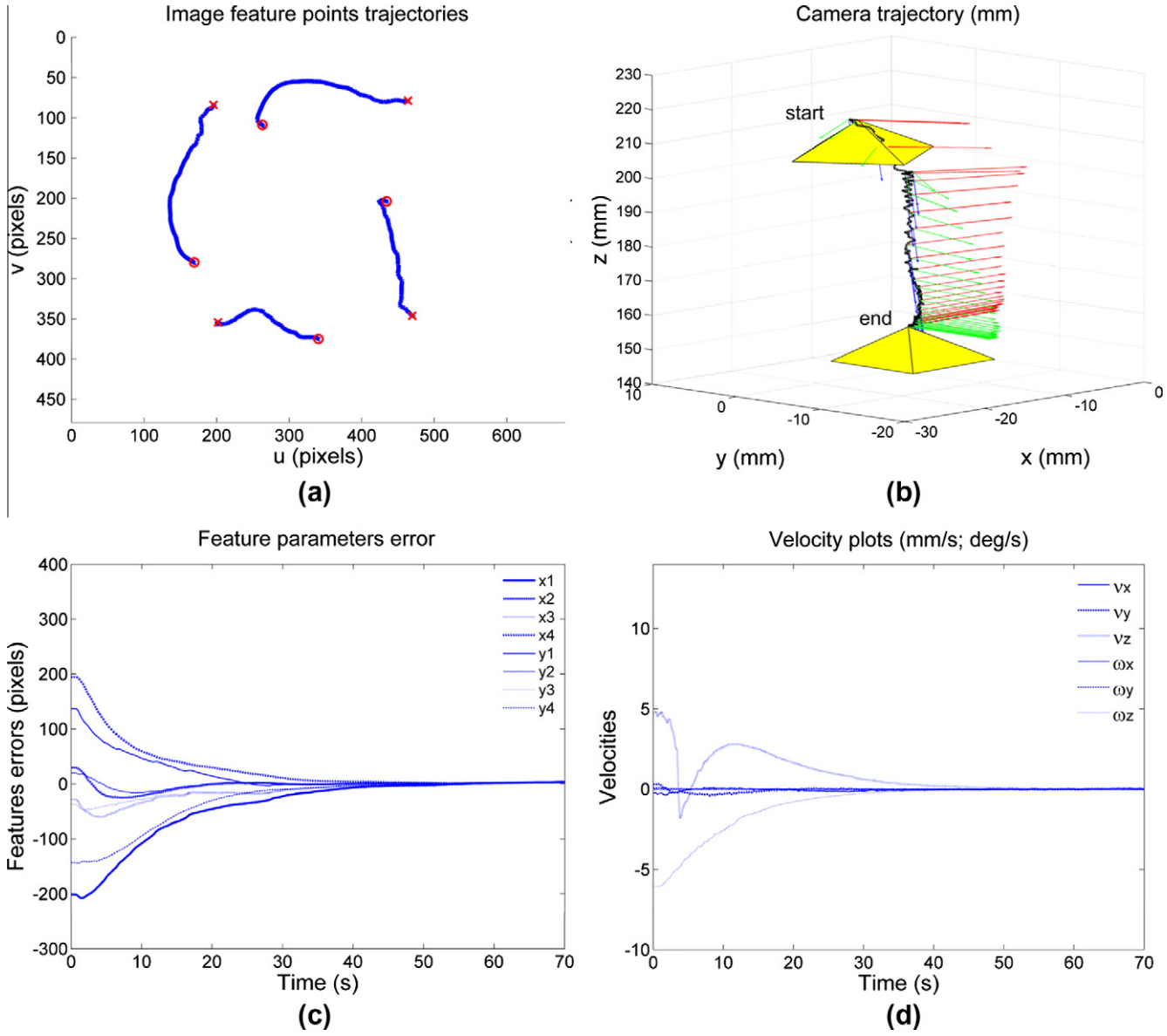

Fig. 21. Experimental results obtained by the SOCO method for Case 3.

Case 2. Combination of translational and rotational movement of the camera, with the initial camera configuration of $\mathcal{P}_c^{c^*} = (10, 40, 75, 0, 5, 0)$.

Case 3. Combination of translational and rotational movement of the camera, with the initial camera configuration of $\mathcal{P}_c^{c^*} = (10, 10, 60, 0, 0, 60)$.

The pose of the desired camera frame with respect to the object frame for the Cases 1 and 2 was set to $\mathcal{P}_{c^*}^{o} = (-14.2, -8.48, 179.82, -1.36, -3.06, 179.73)$, whereas for the Case 3 it was $\mathcal{P}_{c^*}^{o} = (-12.4, -7.41, 155.51, -1.42, -3.02, 179.65)$.

Fig. 19 shows the experimental results by using the SOCO approach proposed in this article for the Case 1. It imposes solely translatory movement of the camera. The image error is minimized with the features converging toward the desired positions. The steady-state errors in the image space are in order of 0.4 pixels.

Fig. 20 shows the experiment plots for Case 2, which involves rotational camera movement of 5° (the small workspace of the robot does not allow using high values of the pitch angle, due to features visibility). The trajectories of the features in the image plane and the trajectory of the camera in the Cartesian space are minimized simultaneously.

The results from implementation of SOCO optimization for the Case 3 are presented in Fig. 21. For comparison, the experimental results for the classical IBVS and PBVS control schemes for Case 3 are given in Figs. 22 and 23, respectively. Sampling rate of 30 ms was utilized for the classical approaches. A value of 0.075 was adopted for the control gain in both IBVS and PBVS methods, as suggested in [40]. It can be noticed that the image feature trajectories resulting from the SOCO approach are very similar to the corresponding IBVS trajectories, i.e., they follow shorter paths when compared to the PBVS results. In addition, the camera trajectory in the Cartesian space is also improved when compared to the IBVS method. As expected, due to the rotational motion of the camera around the optical axis of 60°, the IBVS response include retreat of the camera along the z-axis

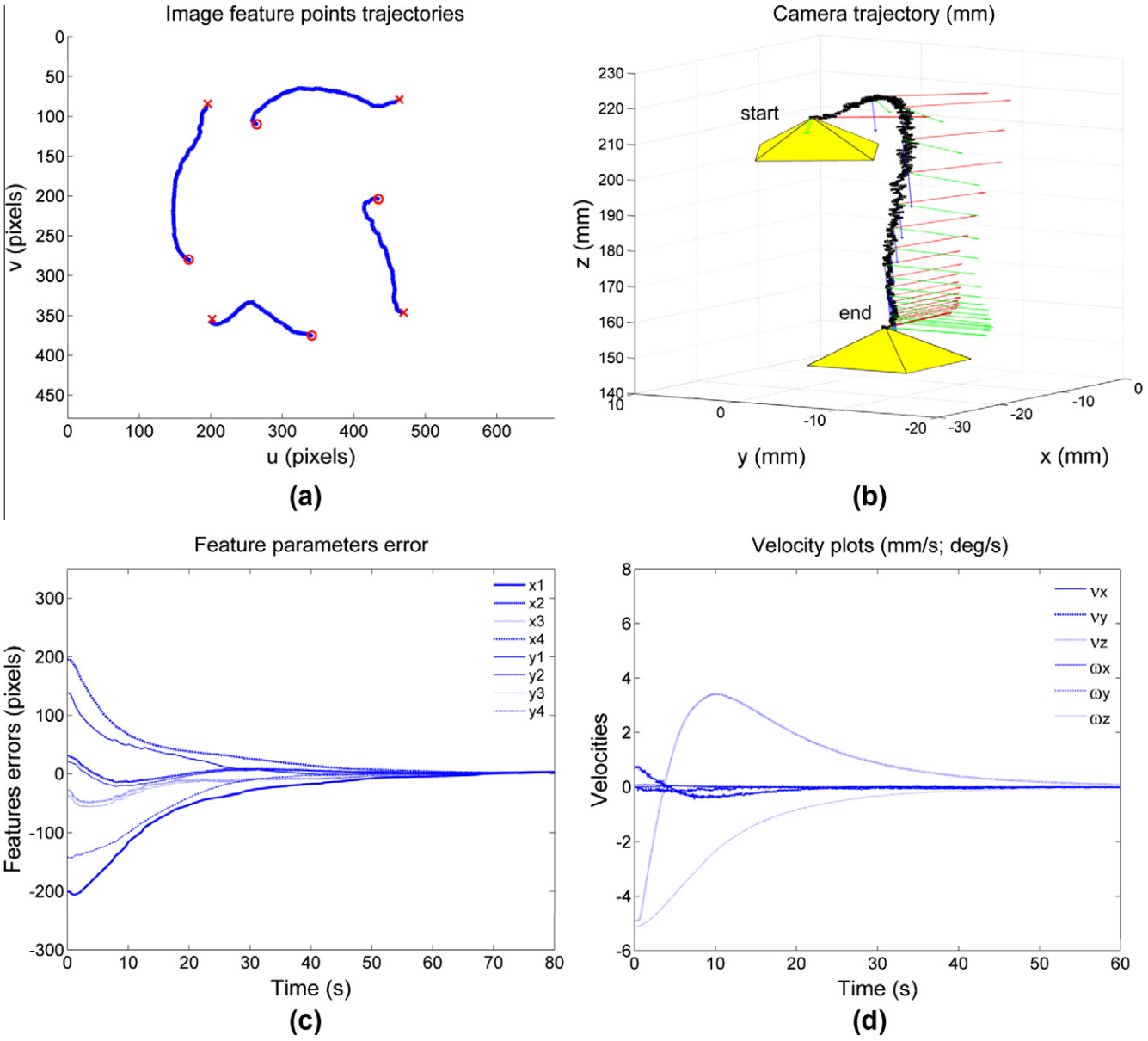

Fig. 22. Experimental results obtained by the IBVS controller for Case 3.

(Fig. 22b and d). Interestingly, the SOCO response is characterized by forward motion of the camera, followed by a slow down, and followed by another forward motion (Fig. 21d), i.e., the task is completed without camera retreat. For the PBVS approach the camera trajectory involves only forward motion, with the image features generating wider trajectories. The proposed SOCO approach takes less iterations than IBVS and PBVS approaches to accomplish the required task, on account of the slower sampling rate.

7. Conclusion

In this paper, a second-order conic optimization (SOCO) model is presented for the problem of visual servo control. The objective function is designed to minimize a weighted average length of the camera and the image feature trajectories. Our motivation for this work initiated from the fact that the IBVS approach does not control the movement of the camera in the Cartesian space, and on the other hand, the PBVS approach does not control the image feature trajectories.

Our approach solves a sequence of SOCO problems at discrete time instances. We have shown that under mild assumptions, these SOCO problems are feasible, thus have optimal solutions. The solutions of these problems yield a sequence of camera velocities, a sequence of camera pose errors, and a sequence of image feature errors. These sequences are shown to be convergent, leading to the convergence of the camera poses and the image feature trajectories to the corresponding desired counterparts.

Simulation results are provided for several different initial and desired camera configurations. Robustness of the model is evaluated for the cases of non-planar target object, image noise, and camera calibration errors. Experiments with CRS-A255 robot arm are conducted as well. The results show the superiority of our approach to both the IBVS and PBVS approaches in a sense that both the camera trajectory and image feature trajectories are minimized.

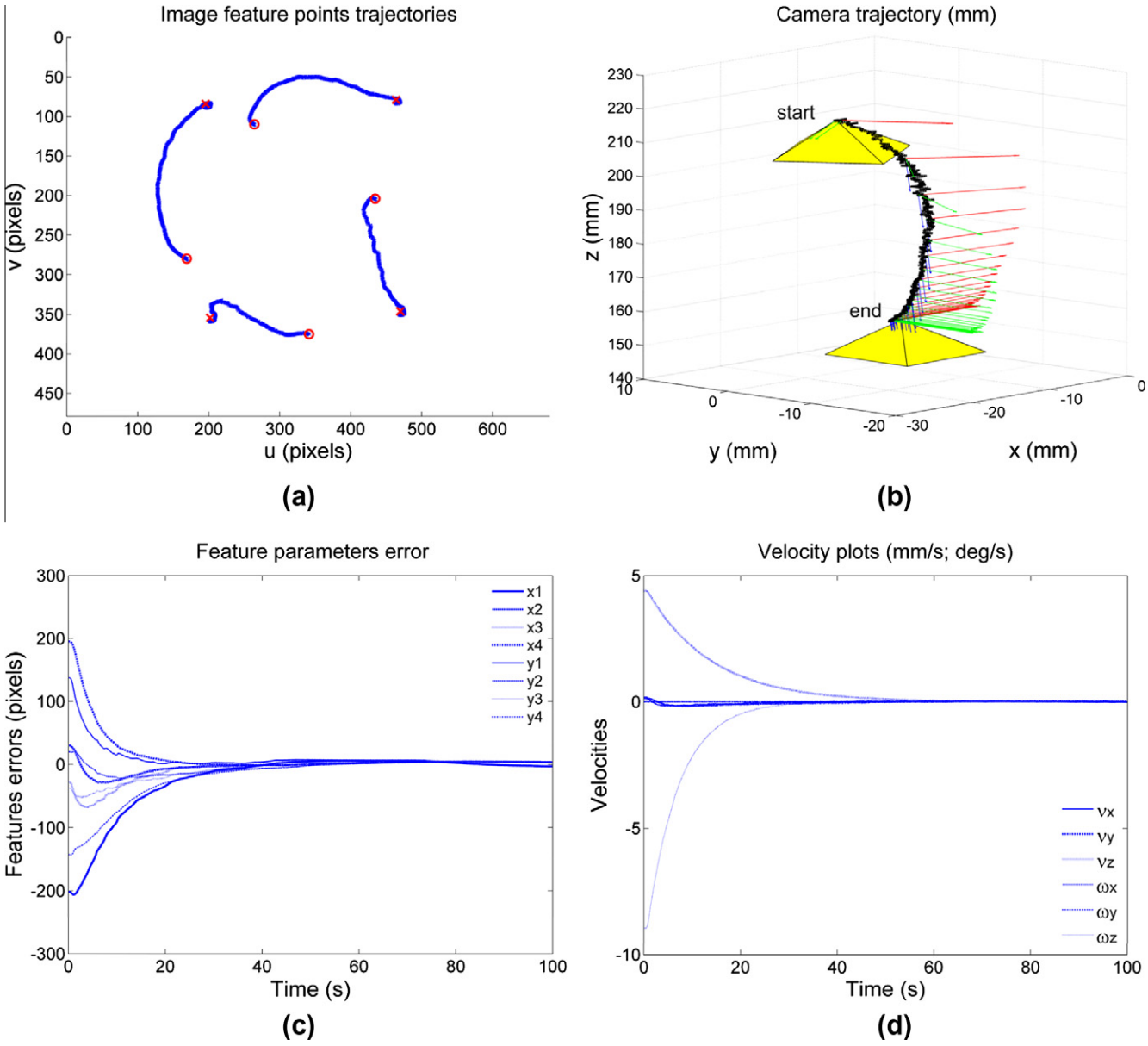

Fig. 23. Experimental results obtained by the PBVS controller for Case 3.

Future work directions include applying robust optimization (RO) techniques to enhance the performance of the model by decreasing its sensitivity to variations in the input data.

Acknowledgment

This work was supported in part by the Natural Sciences and Engineering Research Council of Canada (NSERC) under Discovery Grant 203060-07.

References

[1] Hutchinson S, Hager G, Corke P. A tutorial on visual servo control. IEEE Trans Robot Automat 1996;12(5):651–70.
[2] Chaumette F, Hutchinson S. Visual servo control – Part I: Basic approaches. IEEE Robot Automat Mag 2006;13:82–90.
[3] Janabi-Sharifi F. Visual servoing: theory and applications. In: Cho H, editor. Opto-mechatronics systems handbook. Boca Raton, FL: CRC Press; 2002. p. 15.1–15.24.
[4] Yuan JS-C. A general photogrammetric method for determining object position and orientation. IEEE Trans Robot Automat 1989;5(2):129–42.
[5] Horaud R, Canio B, Leboullenx O, Lacolle B. An analytic solution for the perspective 4-point problem. Comp Vision Graphics Image Process 1989;47:33–44.
[6] Liu Y, Huang TS, Faugeras OD. Determination of camera location from 2-D to 3-D line and point correspondences. IEEE Trans Pattern Anal Mach Intell 1990;12(1):28–37.
[7] Wilson WJ, Hulls CC, Janabi-Sharifi F. Robust image processing and position-based visual servoing. In: Vincze M, Hager GD, editors. Robust vision for vision-based control of motion. New Jersey: IEEE Press; 2000. p. 163–201.
[8] Ficocelli M, Janabi-Sharifi F. Adaptive filtering for pose estimation in visual servoing. In: Proc int conf intelligent robots systems, Maui, HI; 2001. p. 19–24.
[9] Deng L, Janabi-Sharifi F, Wilson WJ. Stability and robustness of visual servoing approaches. In: Proc IEEE int conf robot automat, Washington, DC; 2002. p. 1604–9.
[10] Deng L, Janabi-Sharifi F, Wilson WJ. Hybrid motion control and planning strategy for visual servoing. IEEE Trans Indus Electr 2005;52(4): 1024–40.
[11] Chaumette F. Potential problems of stability and convergence in image-based and position-based visual servo control. In: The confluence of vision and control. Lecture notes in control and information systems. Springer-Verlag; 1998. p. 66–78.

[12] Malis E, Chaumette F, Boudet S. 2-1/2-D visual servoing. IEEE Trans Robot Automat 1999;15(2):238–50.
[13] Corke P, Hutchinson S. A new partitioned approach to image-based visual servo control. IEEE Trans Robot Automat 2001;17(4):507–15.
[14] Kyrki V, Kragic D, Christensen HI. New shortest-path approaches to visual servoing. In: Proc int conf intelligent robots systems, Sendai, Japan; 2004. p. 349–54.
[15] Gans NR, Hutchinson SA. Stable visual servoing through hybrid switched-system control. IEEE Trans Robot 2007;23(3):530–40.
[16] Hashimoto K, Ebine T, Kimura H. Visual servoing with hand-eye manipulator-optimal control approach. IEEE Trans Robot Automat 1996;12(5):766–74.
[17] Malis E. Improving vision-based control using efficient second-order minimization techniques. In: Proc IEEE int conf robot automat, Barcelona, Spain; 2004. p. 1843–8.
[18] Allibert G, Courtial E, Chaumette F. Predictive control of constrained image-based visual servoing. IEEE Trans Robot 2010;26(5):933–9.
[19] Hafez AHA, Jawahar CV. Visual servoing by optimization of a 2D/3D hybrid objective function. In: Proc IEEE int conf robot automat, Roma, Italy; 2007. p. 1691–6.
[20] Kazemi M, Gupta K, Mehrandezh M. Path planning for visual servoing: a review and issues. In: Chesi G, Hashimoto K, editors. Visual servoing via advanced numerical methods. Springer; 2010. p. 189–204.
[21] Chesi G, Hung YS. Global path-planning for constrained and optimal visual servoing. IEEE Trans Robot 2007;23(5):1050–60.
[22] Mezouar Y, Chaumette F. Optimal camera trajectory with image-based control. Int J Robot Res 2003;22(10–11):81–803.
[23] Hafez AHA, Nelakanti AK, Jawahar CV. Path planning approach to visual servoing with feature visibility constraints: a convex optimization based solution. In: Proc int conf intelligent robots systems, San Diego, CA; 2007. p. 1981–6.
[24] Chesi G. Visual servoing path planning via homogeneous forms and LMI optimizations. IEEE Trans Robot 2009;25(2):281–91.
[25] Bhattacharya S, Murrieta-Cid R, Hutchinson S. Optimal paths for landmark-based navigation by differential-drive vehicles with field-of-view constraints. IEEE Trans Robot 2007;23(1):47–59.
[26] Salaris P, Belo F, Fontanelli D, Greco L, Bicchi A. Optimal paths in a constrained image plane for purely image-based parking. In: Proc int conf intelligent robots systems, Nice, France; 2008. p 1673–80.
[27] Alizadeh F, Goldfarb D. Second-order cone programming. Mathemat Program 2003;95(1):3–51.
[28] Janabi-Sharifi F, Wilson WJ. Automatic selection of image features for visual servoing. IEEE Trans Robot Automat 1997;13(6):890–903.
[29] Janabi-Sharifi F, Ficocelli M. Formulation of radiometric feasibility measures for feature selection and planning in visual servoing. IEEE Trans Syst Man Cyber: Part B 2004;34(2):978–87.
[30] Chaumette F. Image moments: a general and useful set of features for visual servoing. IEEE Trans Robot 2004;20(4):713–23.
[31] Boyd S, Vandenberghe L. Convex optimization. Cambridge (England): Cambridge University Press; 2004.
[32] MOSEK Optimization Software; 2010. <http://www.mosek.com/>.
[33] IBM ILOG CPLEX. High Performance Mathematical Programming Engine; 2010. <http://www-01.ibm.com/software/integration/optimization/cplex/>.
[34] Sturm JF. SeDuMi 1.21, a MATLAB Toolbox for Optimization over Symmetric Cones; 1997. <http://sedumi.ie.lehigh.edu/>.
[35] Tutuncu RH, Toh KC, Todd MJ. SDPT3-a MATLAB software package for semidefinite-quadratic-linear programming, version 3.0, 2001. <http://www.math.nus.edu.sg/~mattohkc/sdpt3.html>.
[36] Cervera E. Visual servoing toolbox for MATLAB/Simulink; 2003. <http://vstoolbox.sourceforge.net/>.
[37] Marey M, Chaumette F. Analysis of classical and new visual servoing control laws. In: Proc IEEE int conf robot automat, Nice, France; 2008. p. 3244–9.
[38] Nematollahi E, Janabi-Sharifi F. Generalizations to control laws of image-based visual servoing. Int J Optomech 2009;3(3):167–86.
[39] QuaRC Accelerate Design, Quanser; 2011. <http://www.quanser.com/english/html/faq/fs_quarc_user_info.html>.
[40] Siebel NT, Peters D, Sommer G. Models and control strategies for visual servoing. In: Fung RF, editor. Visual servoing, Intech, Vukovar, Croatia; 2011. p. 21–52.