

Visual Servoing Path-Planning with Elliptical Projections

Tiantian Shen^{1(✉)} and Graziano Chesi²

¹ Department of Electronic Information Engineering, College of Polytechnic,
Hunan Normal University, Changsha, China
`tiantianshen@gmail.com`

² Department of Electrical and Electronic Engineering, University of Hong Kong,
Pokfulam, Hong Kong
`chesi@eee.hku.hk`

Abstract. This paper proposes a path planning approach for visual servoing with elliptical projections. 3D primitives like circles and spheres may project onto image plane of a perspective camera as ellipsoids. From these elliptical projections, moment-based features are constructed. Constraints required by the usage of moment-based features will include camera field of view (FOV) limits and occlusion avoidance of all the observed 3D primitives, a straight or an obstacle dodging path, global convergence and etc. We propose to parametrize these constraints into polynomial inequalities in a common path abscise. They share common variables in polynomial coefficients and these variables will be reassigned via a multidimensional nonlinear minimization process until a satisfactory path is obtained. Such a planned path is interpolated into several segments, at the ends of which elliptical projections are tracked by an image-based visual servoing controller. Two synthetic scenarios demonstrate excellent performance of the path-planning algorithm and tracking scheme.

Keywords: Visual servoing · Path-planning · Circles · Spheres

1 Introduction

Visual servoing is a useful, image-based technique for guiding a robot end-effector to a desired location. Established methods include position-based visual servoing (PBVS) [24] and image-based visual servoing (IBVS) [13]. Both these methods give, in practice, satisfactory results in the case of a motionless target, a fixed desired pose, and a six degrees of freedom (6-DoF) robot with an eye-in-hand camera system [4]. They differ in the error inputs of their control schemes. Input errors for PBVS are comprised of 3D parameters estimated from image measurements. Theoretically, this allows the camera to follow an optimal trajectory in Cartesian space but generally not in image space, since small errors in image measurements can significantly reduce the system accuracy [4]. While in IBVS, feature errors are directly expressed in the image space, allowing IBVS to be

remarkably robust to errors in calibration and image noise; however, when the servoed displacement is large, the camera may reach a local minimum or may cross an interaction matrix singularity [4], particularly in the case of large rotations.

Other than the singularity problem, there are many other constraints and limitations aroused in IBVS approaches attract attention of a bunch of researchers. A significant limitation is the transient loss of features caused either by occlusions or by feature departure from the camera field of view (FOV) [10], since vision feedback plays an essential role in IBVS approaches. Other limitations addressed included joint limits and end-effector collision avoidance. To overcome these constraints as many as possible, a great amount of work have been devoted to design VS controllers, or use different sensors, various kinds of features, or explore high-level control strategies such as path-planning techniques. Typically, navigation functions were presented in [10] in order to overcome local convergence in Cartesian space and violation of camera FOV limits. Potential field methods, originally developed for an on-line collision avoidance [14] were adapted in [16] for robust image-based control while taking into account FOV constraints and joint limits. Numerical optimization techniques were used for offline path planning in [6,7] to address joint limits, FOV limits and collision avoidance. Other useful path-planning methods are presented in a review [15].

The most usual feature errors used in these early work are expressed as a set of pixel coordinates. Other useful features that have been explored in VS community include image moments [2] and luminance [9]. Considering the target consists of generalized objects that can be treated as a combination of some geometrical primitives (such as segments, circles, spheres and cylinders), image moments of these primitives are more intuitive features than pixel coordinates of some representational points, and they have been explored as general and useful features in visual servoing [3]. In the work of [23], for example, a set of three-dimensional (3D) features are computed from image moments of a sphere and used in a classical control law that is proved to perform satisfactorily in a VS task. These 3D features are structured through spherical projection of the sphere, and therefore they are applicable to omnidirectional vision systems. High-level control strategies like path-planning techniques, however, have not yet been considered to take into account constraints for VS with moment-based feature errors. This motivates our preliminary work in [19,20] and then formulates the main focus of this chapter.

The approach of path-planning is here explored for solving constraints problem that arise in moment-based VS, particularly with feature errors constructed and extracted from some elliptical projections. Circles and spheres may project on image plane of a perspective camera as ellipsoids. We first propose some new moment-based feature errors and adopt them in the estimation of camera displacement via a virtual VS (VVS) method, where the target model and position are approximated as a prior knowledge. The estimation results serve as two ends of a path in workspace (Cartesian space). No matter which path is taken between these two ends, the whole target (consists of usually more than one primitive)

has to be all the way visible. This requires the maintenance of camera FOV and also occlusion avoidance of all the observed primitives. Besides, limitations in workspace such as target depth, obstacle avoidance, or simply a shortest straight path are also sometimes compulsory considerations. Constraints required by the usage of moment-based features are much more demanding than that of pixel coordinates of some representational points. We propose to transform these constraints into a group of polynomial inequalities in a common path parameter, with a few coefficients of them adjustable to achieve a satisfactory path. Such a planned path is then interpolated to form a set of segments, elliptical projections at the ends of which are then one by one followed by an IBVS controller.

This chapter is organized as follows. Section 2 introduces some background and formulates the problem. Section 3 focuses on pose estimation from moment-based features. Section 4 proposes path-planning strategies concerning to these moment-based features. Section 5 shows some simulation results, respectively, with two circles and three spheres. Section 6 concludes with some final remarks.

2 Background

Let \mathcal{R} denote the real number set, \mathbf{I}_n the $n \times n$ identity matrix, \mathbf{e}_n the n -th column of $n \times n$ identity matrix, $\mathbf{0}_n$ the $n \times 1$ null vector, $\mathbf{u} * \mathbf{v}$ the convolution of vectors \mathbf{u} and \mathbf{v} , $[\mathbf{v}]_{\times}$ the skew-symmetric matrix of $\mathbf{v} \in \mathcal{R}^3$.

2.1 Camera Frame

Given two camera frames $F^\circ = \{\mathbf{R}, \mathbf{t}\}$ and $F^* = \{\mathbf{I}_3, \mathbf{0}_3\}$, the pose transformation from F° to F^* is expressed as $\{\mathbf{R}^\top, -\mathbf{R}^\top \mathbf{t}\}$. Suppose there is a 3D point with its coordinates to be $\mathbf{H} = [x_o, y_o, z_o]^\top$ in frame F^* , then its coordinates in frame F° is computed as $\mathbf{R}^\top(\mathbf{H} - \mathbf{t})$. Image projection of this point is simply obtained by a division:

$$[X, Y] = \left[\frac{x_o}{z_o}, \frac{y_o}{z_o} \right]. \quad (1)$$

More precisely in the case of a camera, we still need to multiply an camera intrinsic parameters matrix $\mathbf{K} \in \mathcal{R}^{3 \times 3}$ in a way of $\mathbf{K}[X, Y, 1]^\top$ with

$$\mathbf{K} = \begin{pmatrix} f_1 & 0 & u \\ 0 & f_2 & v \\ 0 & 0 & 1 \end{pmatrix}. \quad (2)$$

In the above matrix, f_1 and f_2 are approximated values of the camera focal length, u and v are half values of the size of a projected image, respectively along the horizontal and vertical direction. We use $\zeta_x = 2u$ and $\zeta_y = 2v$ to symbolize image boundaries, e.g. an image of size $\zeta_x \times \zeta_y = 800 \times 600$ pixels. Note that in the following development, \mathbf{K} is usually assumed to be an identity matrix \mathbf{I}_3 unless otherwise specified.

2.2 Elliptical Projections

Circle. A circle can be seen as an intersection of a sphere and a plane:

$$\begin{cases} (x - x_o)^2 + (y - y_o)^2 + (z - z_o)^2 = r^2, \\ \alpha(x - x_o) + \beta(y - y_o) + \gamma(z - z_o) = 0, \end{cases} \quad (3)$$

where $\mathbf{o} = [x_o, y_o, z_o]^\top$ is the sphere center, r the radius, $[\alpha, \beta, \gamma]^\top$ normal vector of the plane with $\alpha^2 + \beta^2 + \gamma^2 = 1$. Image projection of this circle, see Fig. 1, is in the form of an ellipse:

$$\frac{(X - \bar{X} + E(Y - \bar{Y}))^2}{A^2(1 + E^2)} + \frac{(Y - \bar{Y} - E(X - \bar{X}))^2}{B^2(1 + E^2)} = 1, \quad (4)$$

where (\bar{X}, \bar{Y}) is the centroid of the ellipse, A and B are half values of the major and minor diameters, and φ describes the angle between the X-axis and the major axis of the ellipse and they are computed as

$$\begin{cases} \bar{X} = (K_1 K_3 - K_2 K_4) / (K_2^2 - K_0 K_1), \\ \bar{Y} = (K_0 K_4 - K_2 K_3) / (K_2^2 - K_0 K_1), \\ A^2 = \frac{2(K_0 \bar{X}^2 + 2K_2 \bar{X} \bar{Y} + K_1 \bar{Y}^2 - K_5)}{K_0 + K_1 + \sqrt{(K_1 - K_0)^2 + 4K_2^2}}, \\ B^2 = \frac{2(K_0 \bar{X}^2 + 2K_2 \bar{X} \bar{Y} + K_1 \bar{Y}^2 - K_5)}{K_0 + K_1 - \sqrt{(K_1 - K_0)^2 + 4K_2^2}}, \\ E = (K_1 - K_0 + \sqrt{(K_1 - K_0)^2 + 4K_2^2}) / 2K_2, \\ \varphi = \arctan(E). \end{cases} \quad (5)$$

$K_i, i = 0, \dots, 5$ in the above function is in the sense that

$$K_0 X^2 + K_1 Y^2 + 2K_2 XY + 2K_3 X + 2K_4 Y + K_5 = 0. \quad (6)$$

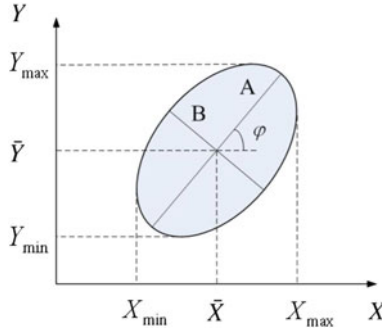


Fig. 1. Elliptical projection of a circle or a sphere [20]

Their values are related to the model parameters in (3):

$$\begin{cases} K_0 = a^2\Delta + 1 - 2ax_0, \\ K_1 = b^2\Delta + 1 - 2by_0, \\ K_2 = ab\Delta - bx_0 - ay_0, \\ K_3 = ac\Delta - cx_0 - az_0, \\ K_4 = bc\Delta - cy_0 - bz_0, \\ K_5 = c^2\Delta + 1 - 2cz_0. \end{cases} \quad \text{and} \quad \begin{cases} a = \alpha/\delta, b = \beta/\delta, c = \gamma/\delta, \\ \Delta = x_0^2 + y_0^2 + z_0^2 - r^2, \\ \delta = x_0\alpha + y_0\beta + z_0\gamma, \\ 1/z = a\bar{X} + b\bar{Y} + c. \end{cases} \quad (7)$$

Sphere. When we consider a sphere

$$(x - x_o)^2 + (y - y_o)^2 + (z - z_o)^2 = r^2 \quad (8)$$

with the same center coordinates and the same radius as in (3), it also project onto image plane as an ellipse and shares in the image plane expressions (4)–(6). The difference lies in the formulation of $K_i, i = 0, \dots, 5$ from model parameters. For a sphere. they are computed [1] as:

$$\begin{cases} K_0 = r^2 - y_o^2 - z_o^2, \\ K_1 = r^2 - x_o^2 - z_o^2, \\ K_2 = x_o y_o, \\ K_3 = x_o z_o, \\ K_4 = y_o z_o, \\ K_5 = r^2 - x_o^2 - y_o^2. \end{cases} \quad (9)$$

Bring (9) into (5), we induce the computation of elliptical parameters in (4) directly from sphere center and its radius [19]:

$$\begin{cases} \bar{X} = \frac{x_o z_o}{z_o^2 - r^2}, \\ \bar{Y} = \frac{y_o z_o}{z_o^2 - r^2}, \\ A = \sqrt{r^2(x_o^2 + y_o^2 + z_o^2 - r^2)/(z_o^2 - r^2)}, \\ B = \sqrt{r^2/(z_o^2 - r^2)}, \\ \varphi = \arccos(y_o/x_o). \end{cases} \quad (10)$$

2.3 Image Moments

Features explored in this chapter are composed of image moments of some solid objects. We assume $I(X, Y)$ are pixel intensities of their projected figures in a grey-scale image, raw image moments m_{ij} and central image moments μ_{ij} of the pertinent primitive are defined as:

$$\begin{aligned} m_{ij} &= \Sigma_X \Sigma_Y X^i Y^j I(X, Y), \\ \mu_{ij} &= \Sigma_X \Sigma_Y (X - \bar{X})^i (Y - \bar{Y})^j I(X, Y), \end{aligned}$$

Table 1. Possible features of geometrical primitives [1]

Primitives 3D	Primitives 2D	Representation possibles
Circle	Ellipse	$(\bar{X}, \bar{Y}, \mu_{20}, \mu_{11}, \mu_{02})$
Sphere	Ellipse	$(\bar{X}, \bar{Y}, (\mu_{20} + \mu_{02})/2)$

where $m_{00} = \pi AB$ calculates the area of the figure, $\bar{X} = m_{10}/m_{00}$ and $\bar{Y} = m_{01}/m_{00}$ are the components of the centroid. The relation between ellipsoid parameters in (4) and a set of second order central moments are given:

$$\begin{aligned}
 \mu_{20} &= (A^2 + B^2 E^2)/(1 + E^2), \\
 \mu_{11} &= E(A^2 - B^2)/(1 + E^2), \\
 \mu_{02} &= (A^2 E^2 + B^2)/(1 + E^2).
 \end{aligned} \tag{11}$$

Possible moment-based representations of geometrical primitives [1], mainly circles and spheres in this chapter, are displayed in Table 1.

Problem. The problem consists of planning trajectories of ellipsoids in order to steer the camera to a location that gives a desired view of some 3D primitives, mainly circles or spheres. How to construct moment-based features to enhance their robustness to errors in image noise? How to maintain target visibility during a VS process, including camera FOV and occlusion avoidance of multiple 3D primitives? How to simultaneously consider constraints in workspace and limitations in image space? How to follow well the planned elliptical trajectories and finally convergence to the desired camera view?

3 Pose Estimation from Image Moments

A satisfactory path concerning many constraints and limitations will be planned in advance of a VS application. We first estimate camera displacement from two perspective target views to form two ends of the path, which will actually serve as a prerequisite and an additional limitation in the proposed path-planning algorithm. There are many existing approaches for pose estimation from two perspective views [8, 12, 17, 25]. The geometric features considered for pose estimation are often points [5], segments [11], contours, conics [18] or image moments [21]. These approaches have the advantage of estimation accuracy; however, they may be subject to local minima and, worse, divergence [22], especially for an eye-in-hand VS application with large camera displacement. Therefore we usually observe at least two primitives to reduce the occurrence of local minima. In order to obtain a large convergence domain and high convergence speed for pose estimation with moment-based features, we estimate the camera displacement [22] via a virtual VS (VVS) based method, similar to moving a virtual camera from one pose to the other with instant camera velocities computed as

$$\mathbf{T} = -\lambda_1 \hat{\mathbf{L}}^+(\mathbf{s}(t) - \mathbf{s}^*), \tag{12}$$

where $\mathbf{T} = [v_x, v_y, v_z, \omega_x, \omega_y, \omega_z]^\top$ describes camera velocities in translation and rotation at time t , which decrease along with the falling trends of $|\mathbf{s}(t) - \mathbf{s}^*|$. λ_1 is a positive gain that controls the iteration progress. $\mathbf{s}(t)$ holds current feature values at time t and \mathbf{s}^* the desired feature values. $\hat{\mathbf{L}}^+$ is the pseudo-inverse of the estimated interaction matrix or image jacobian corresponding to the selected features. This VVS method is equivalent to nonlinear methods that consist in minimizing a cost function using iterative algorithms.

When elliptical projections is formed from a circle, we proposed to use the following features instead of the ones in Table 1:

$$\mathbf{s}_{circle} = [\bar{X}, \bar{Y}, X_{max}, Y_{max}, X_{min}, Y_{min}]^\top, \quad (13)$$

where

$$\begin{cases} X_{max} = \bar{X} + \sqrt{\mu_{20}}, \\ X_{min} = \bar{X} - \sqrt{\mu_{20}}, \\ Y_{max} = \bar{Y} + \sqrt{\mu_{02}}, \\ Y_{min} = \bar{Y} - \sqrt{\mu_{02}} \end{cases} \quad (14)$$

are extreme values of the projected ellipse. The extraction of these features will obviously import less noise than the computation of μ_{20} , μ_{11} and μ_{02} from (11), where noise brought by feature extraction of A and B are squared. Interaction matrices regarding the centroid \bar{X} , \bar{Y} are given below:

$$\begin{aligned} \mathbf{L}_{\bar{X}} &= \begin{bmatrix} -1/z & 0 & \bar{X}/z + a\mu_{20} + b\mu_{11} & \bar{X}\bar{Y} + \mu_{11} & -1 - \bar{X}^2 - \mu_{20} & \bar{Y} \end{bmatrix}, \\ \mathbf{L}_{\bar{Y}} &= \begin{bmatrix} 0 & -1/z & \bar{Y}/z + a\mu_{11} + b\mu_{02} & 1 + \bar{Y}^2 - \mu_{02} & -\bar{X}\bar{Y} - \mu_{11} & -\bar{X} \end{bmatrix}. \end{aligned} \quad (15)$$

Interaction matrices of μ_{20} , μ_{11} and μ_{02} are respectively

$$\begin{aligned} \mathbf{L}_{\mu_{20}} &= \begin{bmatrix} -2(a\mu_{20} + b\mu_{11}) & 0 & 2(1/z + a\bar{X})\mu_{20} + 2b\bar{X}\mu_{11} \\ 2(\bar{Y}\mu_{20} + \bar{X}\mu_{11}) & -4\mu_{20}\bar{X} & 2\mu_{11} \end{bmatrix}, \\ \mathbf{L}_{\mu_{11}} &= \begin{bmatrix} -a\mu_{11} - b\mu_{02} & -a\mu_{20} - b\mu_{11} & a\bar{Y}\mu_{20} + (3/z - c)\mu_{11} + b\bar{X}\mu_{02} \\ 3\bar{Y}\mu_{11} + \bar{X}\mu_{02} & -\bar{Y}\mu_{20} - 3\bar{X}\mu_{11} & \mu_{02} - \mu_{20} \end{bmatrix}, \\ \mathbf{L}_{\mu_{02}} &= \begin{bmatrix} 0 & -2(a\mu_{11} + b\mu_{02}) & 2(1/z + b\bar{Y})\mu_{02} + 2a\bar{Y}\mu_{11} \\ 4\bar{Y}\mu_{02} & -2(\bar{Y}\mu_{11} + \bar{X}\mu_{02}) & -2\mu_{11} \end{bmatrix}. \end{aligned} \quad (16)$$

Therefore interaction matrices corresponding to features in (14) are developed as the following summation:

$$\begin{aligned} \mathbf{L}_{X_{max}} &= \mathbf{L}_{\bar{X}} + \frac{1}{2\sqrt{\mu_{20}}} \mathbf{L}_{\mu_{20}}, \\ \mathbf{L}_{X_{min}} &= \mathbf{L}_{\bar{X}} - \frac{1}{2\sqrt{\mu_{20}}} \mathbf{L}_{\mu_{20}}, \\ \mathbf{L}_{Y_{max}} &= \mathbf{L}_{\bar{Y}} + \frac{1}{2\sqrt{\mu_{02}}} \mathbf{L}_{\mu_{02}}, \\ \mathbf{L}_{Y_{min}} &= \mathbf{L}_{\bar{Y}} - \frac{1}{2\sqrt{\mu_{02}}} \mathbf{L}_{\mu_{02}}. \end{aligned} \quad (17)$$

Interaction matrix regarding feature set in (13) is a stack of matrices in (15) and (17). The computation of a, b, c and $1/z$ in (15)–(17) for a circle, please refer to (7). Afterwards, we refer to (5) and (11) for the estimation of μ_{20} , μ_{11} and μ_{02} in the approximation of these matrices.

When spheres are taken as a target, we consider at least three features for a sphere. Here we use $1/m_{00}$ instead of $(\mu_{02} + \mu_{20})/2$ in the feature set in order to reduce the influence of image noise:

$$\mathbf{s}_{sphere} = [\bar{X}, \bar{Y}, 1/m_{00}]^T. \quad (18)$$

Interaction matrix for \bar{X} and \bar{Y} is the same as (15), while matrix for $1/m_{00}$ is deduced as follows:

$$\mathbf{L}_{1/m_{00}} = \begin{bmatrix} a/m_{00} & b/m_{00} & (c - 3/z)/m_{00} & -3\bar{Y}/m_{00} & 3\bar{X}/m_{00} & 0 \end{bmatrix}. \quad (19)$$

Interaction matrix for a sphere is then a concatenation of matrices (15) and (19). It is noted that sphere is a very special object that projects in the image plane as an ellipse whose centroid may not correspond to the center of the sphere but a point on the spherical surface with its depth z as follows:

$$\begin{cases} z = \frac{1}{a\bar{X} + b\bar{Y} + c} = z_o - r^2/z_o, \\ a = x_o/(x_o^2 + y_o^2 + z_o^2 - r^2), \\ b = y_o/(x_o^2 + y_o^2 + z_o^2 - r^2), \\ c = z_o/(x_o^2 + y_o^2 + z_o^2 - r^2). \end{cases}$$

These computation of z, a, b and c for a sphere will be brought into the approximation of matrices (15) and (19). For further estimation of image moments in them, please refer to (10) and expressions in Sect. 2.3. Ultimately, approximation of the interaction matrix will be brought in (12) to generate instant camera velocities. Given a time interval, camera will move iteration by iteration until feature error $|\mathbf{s}(t) - \mathbf{s}^*|$ is smaller than a threshold, usually 1 pixel. Larger the positive gain, less iterations necessary. The estimation results of camera displacement will serve as pose boundaries for subsequent optimization of a polynomial parametrized camera path.

4 Polynomial Minimization

4.1 Path Parametrization and Polynomial Model

To describe the camera path with boundaries on both sides, we use a path abscise $w \in [0, 1]$ with its value 0 implying one end of the path F^o , and value 1 meaning the other end F^* . Between the above two camera frames, camera path $\{\mathbf{R}(w), \mathbf{t}(w)\}$ is going to be planned according to a lot of servo requirements.

The first requirement is pose boundaries given below:

$$\begin{aligned} F^o : \quad \{\mathbf{R}(0), \mathbf{t}(0)\} &= \{\mathbf{I}_3, \mathbf{0}_3\}, \\ F^* : \quad \{\mathbf{R}(1), \mathbf{t}(1)\} &= \{\mathbf{R}, \mathbf{t}\}, \end{aligned} \quad (20)$$

where \mathbf{R} and \mathbf{t} denote pose transformation from F^o to F^* , which can be derived from the estimation method in Sect. 3. When path abscise w changes from 0 to 1, camera rotation $\mathbf{R}(w)$ alters from \mathbf{I}_3 to \mathbf{R} while camera translation $\mathbf{t}(w)$ changes from $\mathbf{0}_3$ to \mathbf{t} . In order to clearly describe and design the altering process of $\mathbf{R}(w)$, we use quaternion representation of $\mathbf{R}(w)$ to describe rotational path, that is

$$\mathbf{q}(w) = \begin{bmatrix} \sin \frac{\theta(w)}{2} \mathbf{a}(w) \\ \cos \frac{\theta(w)}{2} \end{bmatrix} \quad (21)$$

with $\theta(w) \in (0, \pi)$ and $\mathbf{a}(w) \in \mathcal{R}^3$ are, respectively, rotation angle and axis such that $\mathbf{R}(w) = e^{\theta(w)[\mathbf{a}(w)]_\times}$. Afterwards, we parametrize each coordinate in $\mathbf{q}(w)$ and $\mathbf{t}(w)$ as a polynomial in w :

$$\begin{aligned} \mathbf{q}(w) &= \mathbf{U} \cdot [w^\sigma, \dots, w, 1]^\top, \\ \mathbf{t}(w) &= \mathbf{V} \cdot [w^\tau, \dots, w, 1]^\top. \end{aligned} \quad (22)$$

Polynomial parametrization of quaternions share the same degree of σ , and translational coordinates τ . Matrices $\mathbf{U} \in \mathcal{R}^{4 \times (\sigma+1)}$ and $\mathbf{V} \in \mathcal{R}^{3 \times (\tau+1)}$ contain their polynomial coefficients. If a straight camera path in workspace is demanded, degree τ could be set as 1 leaving only trajectories of quaternion to be adjusted. In general, we take both of quaternion and translational coordinates to be quadratic polynomials with $\sigma = 2$ and $\tau = 2$. In the following development, we assume that the whole camera path is modeled by seven quadratic polynomials unless otherwise indicated.

Both \mathbf{U} and \mathbf{V} for quadratic polynomials have totally three columns. Their last columns are determined by boundaries in (20). Specifically, condition $\mathbf{R}(0) = \mathbf{I}_3$ gives $\mathbf{q}(0) = \mathbf{e}_4$ that constitutes the last column in \mathbf{U} , and condition $\mathbf{t}(0) = \mathbf{0}_3$ generates the last column in \mathbf{V} . Let $\mathbf{b} \in \mathcal{R}^4$ and $\mathbf{d} \in \mathcal{R}^3$ denote the mid column, respectively, in \mathbf{U} and \mathbf{V} . Then their first columns are determined by the other two columns considering conditions $\{\mathbf{R}(1), \mathbf{t}(1)\} = \{\mathbf{R}, \mathbf{t}\}$ with $\mathbf{q}(1) = \mathbf{q}$. Consequently, \mathbf{U} and \mathbf{V} for quadratic polynomials are rewritten as:

$$\begin{aligned} \mathbf{U} &= [\mathbf{q} - \mathbf{b} - \mathbf{e}_4, \mathbf{b}, \mathbf{e}_4], \\ \mathbf{V} &= [\mathbf{t} - \mathbf{d}, \mathbf{d}, \mathbf{0}_3]. \end{aligned} \quad (23)$$

The above limitation to the entries in \mathbf{U} and \mathbf{V} matrices, leaving only their mid columns \mathbf{b} and \mathbf{d} to be variable, guarantees boundaries in (20). Zero assignment of variables in \mathbf{b} and \mathbf{d} implies a straight path in workspace and also four straight quaternion trajectories. Starting from this initial assignment, variables in \mathbf{b} and \mathbf{d} will be reassigned and adjusted according to many other requirements compulsory in a visual servoing application.

For the convenience of subsequent development, we use $\mathbf{u}_i^\top \in \mathcal{R}^{1 \times 3}$, $i = 1, \dots, 4$ to denote the i -th row of matrix \mathbf{U} and $\mathbf{v}_j^\top \in \mathcal{R}^{1 \times 3}$, $j = 1, 2, 3$ the j -th row of matrix \mathbf{V} , then we have

$$\begin{aligned}\mathbf{U} &= [\mathbf{u}_1, \mathbf{u}_2, \mathbf{u}_3, \mathbf{u}_4]^\top, \\ \mathbf{V} &= [\mathbf{v}_1, \mathbf{v}_2, \mathbf{v}_3]^\top.\end{aligned}\tag{24}$$

Each row of \mathbf{U} constitutes polynomial coefficients of the pertinent coordinate in $\mathbf{q}(w)$ in the form of

$$\mathbf{q}_i(w) = \mathbf{u}_i^\top [w^\sigma, \dots, w, 1]^\top, \quad i = 1, \dots, 4.$$

The associated rotation matrix is constructed from the above quaternion:

$$\mathbf{R}(w) = \begin{pmatrix} q_1^2(w) - q_2^2(w) - q_3^2(w) + q_4^2(w) & 2(q_1(w)q_2(w) - q_3(w)q_4(w)) \\ 2(q_1(w)q_2(w) + q_3(w)q_4(w)) & -q_1^2(w) + q_2^2(w) - q_3^2(w) + q_4^2(w) \\ 2(q_1(w)q_3(w) - q_2(w)q_4(w)) & 2(q_2(w)q_3(w) + q_1(w)q_4(w)) \\ 2(q_1(w)q_3(w) + q_2(w)q_4(w)) & 2(q_2(w)q_3(w) - q_1(w)q_4(w)) \\ -q_1^2(w) - q_2^2(w) + q_3^2(w) + q_4^2(w) & \end{pmatrix}.$$

It can be seen that every entry in rotation matrix $\mathbf{R}(w)$ is also a polynomial in w . Their degrees are doubled from (22), that is 2σ . Let $\mathbf{r}_{ij} \in \mathcal{R}^5$ (considering $\sigma = 2$) carry polynomial coefficients of the entry in the i -th row and the j -th column of the rotation matrix. They are related to the rows of matrix \mathbf{U} in the following way [20]:

$$\begin{aligned}\mathbf{r}_{11} &= \mathbf{u}_1 * \mathbf{u}_1 - \mathbf{u}_2 * \mathbf{u}_2 - \mathbf{u}_3 * \mathbf{u}_3 + \mathbf{u}_4 * \mathbf{u}_4, \\ \mathbf{r}_{12} &= 2(\mathbf{u}_1 * \mathbf{u}_2 - \mathbf{u}_3 * \mathbf{u}_4), \\ \mathbf{r}_{13} &= 2(\mathbf{u}_1 * \mathbf{u}_3 + \mathbf{u}_2 * \mathbf{u}_4), \\ \mathbf{r}_{21} &= 2(\mathbf{u}_1 * \mathbf{u}_2 + \mathbf{u}_3 * \mathbf{u}_4), \\ \mathbf{r}_{22} &= -\mathbf{u}_1 * \mathbf{u}_1 + \mathbf{u}_2 * \mathbf{u}_2 - \mathbf{u}_3 * \mathbf{u}_3 + \mathbf{u}_4 * \mathbf{u}_4, \\ \mathbf{r}_{23} &= 2(\mathbf{u}_2 * \mathbf{u}_3 - \mathbf{u}_1 * \mathbf{u}_4), \\ \mathbf{r}_{31} &= 2(\mathbf{u}_1 * \mathbf{u}_3 - \mathbf{u}_2 * \mathbf{u}_4), \\ \mathbf{r}_{32} &= 2(\mathbf{u}_2 * \mathbf{u}_3 + \mathbf{u}_1 * \mathbf{u}_4), \\ \mathbf{r}_{33} &= -\mathbf{u}_1 * \mathbf{u}_1 - \mathbf{u}_2 * \mathbf{u}_2 + \mathbf{u}_3 * \mathbf{u}_3 + \mathbf{u}_4 * \mathbf{u}_4.\end{aligned}\tag{25}$$

As a result, rotation matrix can be rewritten as:

$$\mathbf{R}(w) = \begin{pmatrix} \mathbf{r}_{11}^\top \mathbf{w}_5 & \mathbf{r}_{12}^\top \mathbf{w}_5 & \mathbf{r}_{13}^\top \mathbf{w}_5 \\ \mathbf{r}_{21}^\top \mathbf{w}_5 & \mathbf{r}_{22}^\top \mathbf{w}_5 & \mathbf{r}_{23}^\top \mathbf{w}_5 \\ \mathbf{r}_{31}^\top \mathbf{w}_5 & \mathbf{r}_{32}^\top \mathbf{w}_5 & \mathbf{r}_{33}^\top \mathbf{w}_5 \end{pmatrix},\tag{26}$$

where $\mathbf{w}_5 = [w^4, \dots, w, 1]^\top$.

4.2 Constraints and Limitations

For all possible constraints and limitation, priority will be given to the depth of the target, which is required to be positive all the way along the camera path. Second, collision in workspace will be prevented by adjusting mainly the trajectories of camera translation. Third, camera FOV limit is going to be met by restraining elliptical projections embraced within the image frame. When the target consists of two or more primitives, occlusion among them will also be avoided to maintain their visibility in the course of a VS application.

Target Depth. The target needs to be located in front of the camera with a certain distance. When the target consists of circles or spheres, it is necessary to have the distance larger than their body size (mainly the radius). At least, depth of the target center z_o is meant to be greater than the radius, that is the value of

$$g_1 = z_o - r \quad (27)$$

needs to be positive during a servo process.

Depth of the target center z_o with respect to a camera is determined by camera pose $\{\mathbf{R}(w), \mathbf{t}(w)\}$, taken as the third item in $\mathbf{R}(w)^\top(\mathbf{o} - \mathbf{t}(w))$. In Sect. 2.2, camera frame is assumed to be $\{\mathbf{I}_3, \mathbf{0}_3\}$, coordinates \mathbf{o} could be a circle center or a sphere center. Building on polynomial parametrization of $\{\mathbf{q}(w), \mathbf{t}(w)\}$, we wish to express g_1 into polynomial with some of its coefficients adjustable. Take a circle for example, it is obvious that its center coordinates and normal vector in the w -based camera frame are polynomials in the path abscise. To prove it, we bring (23) into the computation of w -based center coordinates. Firstly, we have

$$\begin{aligned} \mathbf{o} - \mathbf{t}(w) &= \{[\mathbf{0}_3, \mathbf{0}_3, \mathbf{o}] - \mathbf{V}\} \cdot \mathbf{w}_3 \\ &= [\mathbf{d} - \mathbf{t}, -\mathbf{d}, \mathbf{o}] \cdot \mathbf{w}_3. \end{aligned} \quad (28)$$

where $\mathbf{w}_3 = [w^2, w, 1]^\top$. Let $\mathbf{h}_j^\top \in \mathcal{R}^{1 \times 3}$, $j = 1, 2, 3$ denote the j -th row of the associated coefficient matrix for $\mathbf{o} - \mathbf{t}(w)$. Center coordinates in both of (3) and (8) can be computed as:

$$\begin{aligned} \begin{pmatrix} x_o(w) \\ y_o(w) \\ z_o(w) \end{pmatrix} &= \mathbf{R}(w)^\top(\mathbf{o} - \mathbf{t}(w)) \\ &= \begin{pmatrix} \mathbf{r}_{11}^\top \mathbf{w}_5 & \mathbf{r}_{21}^\top \mathbf{w}_5 & \mathbf{r}_{31}^\top \mathbf{w}_5 \\ \mathbf{r}_{12}^\top \mathbf{w}_5 & \mathbf{r}_{22}^\top \mathbf{w}_5 & \mathbf{r}_{32}^\top \mathbf{w}_5 \\ \mathbf{r}_{13}^\top \mathbf{w}_5 & \mathbf{r}_{23}^\top \mathbf{w}_5 & \mathbf{r}_{33}^\top \mathbf{w}_5 \end{pmatrix} \cdot \begin{pmatrix} \mathbf{h}_1^\top \mathbf{w}_3 \\ \mathbf{h}_2^\top \mathbf{w}_3 \\ \mathbf{h}_3^\top \mathbf{w}_3 \end{pmatrix} \\ &= \begin{pmatrix} \mathbf{w}_3^\top \mathbf{h}_1 \mathbf{r}_{11}^\top \mathbf{w}_5 + \mathbf{w}_3^\top \mathbf{h}_2 \mathbf{r}_{21}^\top \mathbf{w}_5 + \mathbf{w}_3^\top \mathbf{h}_3 \mathbf{r}_{31}^\top \mathbf{w}_5 \\ \mathbf{w}_3^\top \mathbf{h}_1 \mathbf{r}_{12}^\top \mathbf{w}_5 + \mathbf{w}_3^\top \mathbf{h}_2 \mathbf{r}_{22}^\top \mathbf{w}_5 + \mathbf{w}_3^\top \mathbf{h}_3 \mathbf{r}_{32}^\top \mathbf{w}_5 \\ \mathbf{w}_3^\top \mathbf{h}_1 \mathbf{r}_{13}^\top \mathbf{w}_5 + \mathbf{w}_3^\top \mathbf{h}_2 \mathbf{r}_{23}^\top \mathbf{w}_5 + \mathbf{w}_3^\top \mathbf{h}_3 \mathbf{r}_{33}^\top \mathbf{w}_5 \end{pmatrix}. \end{aligned} \quad (29)$$

The normal vector in (3) is related to camera rotation:

$$\begin{aligned}
\begin{pmatrix} \alpha(w) \\ \beta(w) \\ \gamma(w) \end{pmatrix} &= \mathbf{R}(w)^\top \begin{pmatrix} \alpha \\ \beta \\ \gamma \end{pmatrix} \\
&= \begin{pmatrix} \mathbf{r}_{11}^\top \mathbf{w}_5 \cdot \alpha + \mathbf{r}_{21}^\top \mathbf{w}_5 \cdot \beta + \mathbf{r}_{31}^\top \mathbf{w}_5 \cdot \gamma \\ \mathbf{r}_{12}^\top \mathbf{w}_5 \cdot \alpha + \mathbf{r}_{22}^\top \mathbf{w}_5 \cdot \beta + \mathbf{r}_{32}^\top \mathbf{w}_5 \cdot \gamma \\ \mathbf{r}_{13}^\top \mathbf{w}_5 \cdot \alpha + \mathbf{r}_{23}^\top \mathbf{w}_5 \cdot \beta + \mathbf{r}_{33}^\top \mathbf{w}_5 \cdot \gamma \end{pmatrix}.
\end{aligned} \tag{30}$$

From (29) and (30), we can see that all of them are polynomials in w , with their polynomial coefficients expressed as [20]:

$$\begin{aligned}
\mathbf{p}_{x_o} &= \mathbf{r}_{11} * \mathbf{h}_1 + \mathbf{r}_{21} * \mathbf{h}_2 + \mathbf{r}_{31} * \mathbf{h}_3, \\
\mathbf{p}_{y_o} &= \mathbf{r}_{12} * \mathbf{h}_1 + \mathbf{r}_{22} * \mathbf{h}_2 + \mathbf{r}_{32} * \mathbf{h}_3, \\
\mathbf{p}_{z_o} &= \mathbf{r}_{13} * \mathbf{h}_1 + \mathbf{r}_{23} * \mathbf{h}_2 + \mathbf{r}_{33} * \mathbf{h}_3, \\
\mathbf{p}_\alpha &= \alpha \mathbf{r}_{11} + \beta \mathbf{r}_{21} + \gamma \mathbf{r}_{31}, \\
\mathbf{p}_\beta &= \alpha \mathbf{r}_{12} + \beta \mathbf{r}_{22} + \gamma \mathbf{r}_{32}, \\
\mathbf{p}_\gamma &= \alpha \mathbf{r}_{13} + \beta \mathbf{r}_{23} + \gamma \mathbf{r}_{33}.
\end{aligned} \tag{31}$$

The size of \mathbf{w}_3 and \mathbf{w}_5 determines the size of the above vectors, that is $\mathbf{p}_{x_o}, \mathbf{p}_{y_o}, \mathbf{p}_{z_o} \in \mathcal{R}^7$ and $\mathbf{p}_\alpha, \mathbf{p}_\beta, \mathbf{p}_\gamma \in \mathcal{R}^5$, therefore $g_1 = z_o - r$ is a six order polynomial with its coefficient to be

$$\mathbf{p}_{g_1} = \begin{pmatrix} \mathbf{0}_6 \\ -r \end{pmatrix} + \mathbf{p}_{z_o}. \tag{32}$$

Local minimum of this six order polynomial within the domain of $w \in (0, 1)$ will be found by taking a differential coefficient of \mathbf{p}_{z_o-r} . Our algorithm aims at positivity of this local minimum, so that the target is always located in front of the camera.

A Straight Line or Collision Avoidance. Either a straight line in Cartesian space or an obstacle dodging path, realization lies in the regulation of $x_o(w)$, $y_o(w)$ and $z_o(w)$. If a straight path is required, it can be fulfilled simply by taking the first order in translational model in (22). Consequently, coefficient matrix \mathbf{V} has only two columns [20] without any variable, letting variables in \mathbf{U} adjusted by other constraints and limitations. In the usual case of quadratic polynomials, we can simply assign \mathbf{d} in (23) to be zeros, which also gives a straight line.

If an obstacle blocks the way of a camera, a dodging path must be achieved. Let $\mathbf{l}_{obs} \in \mathcal{R}^3$ denote the obstacle location in the frame of $\{\mathbf{I}_3, \mathbf{0}_3\}$. Camera path $\{\mathbf{R}(w), \mathbf{t}(w)\}$ will keep a safe distance from the obstacle by the positivity of expression of

$$g_2 = \|\mathbf{R}(w)^\top (\mathbf{l}_{obs} - \mathbf{t}(w))\|^2 - S^2, \tag{33}$$

where S is a tolerated safe distance. $\mathbf{R}(w)^\top (\mathbf{l}_{obs} - \mathbf{t}(w))$ contains obstacle coordinates with respect to camera frame of $\{\mathbf{R}(w), \mathbf{t}(w)\}$. These w -based coordinates

are, respectively, symbolized as x_{obs} , y_{obs} and z_{obs} . They are also polynomials in w with coefficients to be

$$\begin{aligned}\mathbf{p}_{x_{obs}} &= \mathbf{r}_{11} * \mathbf{l}_1 + \mathbf{r}_{21} * \mathbf{l}_2 + \mathbf{r}_{31} * \mathbf{l}_3, \\ \mathbf{p}_{y_{obs}} &= \mathbf{r}_{12} * \mathbf{l}_1 + \mathbf{r}_{22} * \mathbf{l}_2 + \mathbf{r}_{32} * \mathbf{l}_3, \\ \mathbf{p}_{z_{obs}} &= \mathbf{r}_{13} * \mathbf{l}_1 + \mathbf{r}_{23} * \mathbf{l}_2 + \mathbf{r}_{33} * \mathbf{l}_3,\end{aligned}\tag{34}$$

where $\mathbf{l}_j^\top \in \mathcal{R}^{1 \times 3}$, $j = 1, 2, 3$ is the j -th row of matrix $[\mathbf{d} - \mathbf{t}, -\mathbf{d}, \mathbf{l}_{obs}]$. Therefore g_2 is also a polynomial in w , whose coefficients are included in a vector:

$$\mathbf{p}_{g_2} = \begin{pmatrix} \mathbf{0}_{12} \\ -S^2 \end{pmatrix} + \mathbf{p}_{x_{obs}} * \mathbf{p}_{x_{obs}} + \mathbf{p}_{y_{obs}} * \mathbf{p}_{y_{obs}} + \mathbf{p}_{z_{obs}} * \mathbf{p}_{z_{obs}}.\tag{35}$$

This is a twelve degree polynomial. Similar to the case in (32), our algorithm searches for the value of coefficient variables that give positivity to the local minimum of g_2 within the limited range of w .

Field of View Limit. Taking into account of intrinsic parameters of a camera, extreme values of elliptical projections, as shown in Fig. 1, have to be restrained within image size of $\zeta_x \times \zeta_y$. This limitation is realized by four inequalities:

$$X_{max} < \frac{\zeta_x}{2f_1}, \quad X_{min} > -\frac{\zeta_x}{2f_1}, \quad Y_{max} < \frac{\zeta_y}{2f_2}, \quad Y_{min} > -\frac{\zeta_y}{2f_2},\tag{36}$$

where f_1 and f_2 are approximated values of focal length in (2). Bringing (5) and (11) into (14), X_{max} , Y_{max} , X_{min} and Y_{min} are developed and expressed [20] as the function of $K_i, i = 0, \dots, 5$:

$$\begin{aligned}X_{max} &= (K_2K_4 - K_1K_3 + \sqrt{G_3})/(K_0K_1 - K_2^2), \\ X_{min} &= (K_2K_4 - K_1K_3 - \sqrt{G_3})/(K_0K_1 - K_2^2), \\ G_3 &= (K_1K_3 - K_2K_4)^2 - (K_0K_1 - K_2^2)(K_1K_5 - K_4^2), \\ Y_{max} &= (K_2K_3 - K_0K_4 + \sqrt{G_4})/(K_0K_1 - K_2^2), \\ Y_{min} &= (K_2K_3 - K_0K_4 - \sqrt{G_4})/(K_0K_1 - K_2^2), \\ G_4 &= (K_0K_4 - K_2K_3)^2 - (K_0K_1 - K_2^2)(K_0K_5 - K_3^2).\end{aligned}\tag{37}$$

In order to transform inequalities in (36) into the form of polynomials to facilitate an uniform optimization of parameterized variables in (23), let

$$k_i = \delta^2 K_i, i = 1, \dots, 5,\tag{38}$$

and we will produce polynomial coefficients of all of k_i from (7) and (31) as follows:

$$\begin{aligned}
\mathbf{p}_\Delta &= \mathbf{p}_{x_o} * \mathbf{p}_{x_o} + \mathbf{p}_{y_o} * \mathbf{p}_{y_o} + \mathbf{p}_{z_o} * \mathbf{p}_{z_o} + [\mathbf{0}_{12}^\top, -r^2]^\top, \\
\mathbf{p}_\delta &= \mathbf{p}_\alpha * \mathbf{p}_{x_o} + \mathbf{p}_\beta * \mathbf{p}_{y_o} + \mathbf{p}_\gamma * \mathbf{p}_{z_o}, \\
\mathbf{p}_{k_0} &= \mathbf{p}_\alpha * \mathbf{p}_\alpha * \mathbf{p}_\Delta + \mathbf{p}_\delta * \mathbf{p}_\delta - 2\mathbf{p}_\alpha * \mathbf{p}_\delta * \mathbf{p}_{x_o}, \\
\mathbf{p}_{k_1} &= \mathbf{p}_\beta * \mathbf{p}_\beta * \mathbf{p}_\Delta + \mathbf{p}_\delta * \mathbf{p}_\delta - 2\mathbf{p}_\beta * \mathbf{p}_\delta * \mathbf{p}_{y_o}, \\
\mathbf{p}_{k_2} &= \mathbf{p}_\alpha * \mathbf{p}_\beta * \mathbf{p}_\Delta - \mathbf{p}_\beta * \mathbf{p}_\delta * \mathbf{p}_{x_o} - \mathbf{p}_\alpha * \mathbf{p}_\delta * \mathbf{p}_{y_o}, \\
\mathbf{p}_{k_3} &= \mathbf{p}_\alpha * \mathbf{p}_\gamma * \mathbf{p}_\Delta - \mathbf{p}_\gamma * \mathbf{p}_\delta * \mathbf{p}_{x_o} - \mathbf{p}_\alpha * \mathbf{p}_\delta * \mathbf{p}_{z_o}, \\
\mathbf{p}_{k_4} &= \mathbf{p}_\beta * \mathbf{p}_\gamma * \mathbf{p}_\Delta - \mathbf{p}_\gamma * \mathbf{p}_\delta * \mathbf{p}_{y_o} - \mathbf{p}_\beta * \mathbf{p}_\delta * \mathbf{p}_{z_o}, \\
\mathbf{p}_{k_5} &= \mathbf{p}_\gamma * \mathbf{p}_\gamma * \mathbf{p}_\Delta + \mathbf{p}_\delta * \mathbf{p}_\delta - 2\mathbf{p}_\gamma * \mathbf{p}_\delta * \mathbf{p}_{z_o}.
\end{aligned} \tag{39}$$

All of the above $\mathbf{p}_{k_i} \in \mathcal{R}^{19}$ are polynomial coefficients of k_i , $i = 1, \dots, 5$, a eighteen degree polynomial in w . Let

$$g_d = k_0 k_1 - k_2^2 \tag{40}$$

substitutes the denominator in (37). We have first

$$\begin{aligned}
g_3 &= (k_1 k_3 - k_2 k_4)^2 - g_d (k_1 k_5 - k_4^2), \\
g_4 &= (k_0 k_4 - k_2 k_3)^2 - g_d (k_0 k_5 - k_3^2).
\end{aligned} \tag{41}$$

Positivity of g_3 and g_4 ensures that the maximum and minimum values along the same axis are unequal. This will avoid the degenerating case: elliptical projection boils down to a segment. Building on the positivity of g_3 and g_4 , FOV limits are developed as follows:

$$\begin{aligned}
X_{max} : \quad g_5 &= g_d \left[\frac{\zeta_x}{2f_1} g_d - k_2 k_4 + k_1 k_3 \right]^2 - g_d g_3 > 0, \\
X_{min} : \quad g_6 &= g_d \left[\frac{\zeta_x}{2f_1} g_d + k_2 k_4 - k_1 k_3 \right]^2 - g_d g_3 > 0, \\
Y_{max} : \quad g_7 &= g_d \left[\frac{\zeta_y}{2f_2} g_d - k_2 k_3 + k_0 k_4 \right]^2 - g_d g_4 > 0, \\
Y_{min} : \quad g_8 &= g_d \left[\frac{\zeta_y}{2f_2} g_d + k_2 k_3 - k_0 k_4 \right]^2 - g_d g_4 > 0.
\end{aligned} \tag{42}$$

Since k_i , $i = 1, \dots, 5$ are polynomials, therefore g_j , $j = 3, \dots, 8$ are also polynomials in w . Their coefficients are computed from \mathbf{p}_{k_i} , $i = 1, \dots, 5$ in (39). Take g_3 for example, polynomial coefficients of g_3 is computed as:

$$\begin{aligned}
\mathbf{p}_{k_1 k_3 - k_2 k_4} &= \mathbf{p}_{k_1} * \mathbf{p}_{k_3} - \mathbf{p}_{k_2} * \mathbf{p}_{k_4}, \\
\mathbf{p}_{k_1 k_5 - k_4 k_4} &= \mathbf{p}_{k_1} * \mathbf{p}_{k_5} - \mathbf{p}_{k_4} * \mathbf{p}_{k_4}, \\
\mathbf{p}_{g_d} &= \mathbf{p}_{k_0} * \mathbf{p}_{k_1} - \mathbf{p}_{k_2} * \mathbf{p}_{k_2}, \\
\mathbf{p}_{g_3} &= \mathbf{p}_{k_1 k_3 - k_2 k_4} * \mathbf{p}_{k_1 k_3 - k_2 k_4} - \mathbf{p}_{g_d} * \mathbf{p}_{k_1 k_5 - k_4 k_4}.
\end{aligned} \tag{43}$$

Similar computation applies to polynomial coefficients of g_j , $j = 4, \dots, 8$. Provided the initial value of \mathbf{b} and \mathbf{d} in (23), we derive entries of \mathbf{U} and \mathbf{V} and bring them into (22), (24), (25), (31), (32) and (39). As a result, \mathbf{p}_{g_j} , $j = 1, \dots, 8$ can be computed. We take the derivative of each \mathbf{p}_{g_j} and solve for the corresponding $w \in (0, 1)$ that give zero derivatives. If such w exist, then a local minimum of g_j can be found at such w values. We expect these local minimums to be always positive. Take all of these requirements together, we first find local minimums for each g_j , $j = 1, \dots, n$ and again the minimum of all of these local minimums and denote it as g^* :

$$\begin{aligned} g^* &= \min_{j=1}^n (\min_{w \in (0,1)} (g_j)), \\ \begin{pmatrix} \mathbf{b}^* \\ \mathbf{d}^* \end{pmatrix} &= \min_{\mathbf{b}, \mathbf{d}} (-g^*). \end{aligned} \quad (44)$$

No action will be taken if the value of g^* is initially positive, otherwise a minimization of $-g^*$ will be conducted until it converts its sign. We search for appropriate values of \mathbf{b} , \mathbf{d} by minimizing $-g^*$ with MATLAB tool till $g^* > 0$. Once the minimization result is obtained, we denote it as \mathbf{b}^* , \mathbf{d}^* and bring it back into (23) and (22). Till this end, a planned path that converges to the desired camera view while satisfying constraints in both of workspace and image plane is found.

Target Self-occlusion Avoidance. Here we mainly consider occlusion among spheres when more than one sphere are taken as a target. One sphere can not shelter the other from the camera view, otherwise feature extraction will fail to provide some part of the controller input.

We denote (\bar{X}_1, \bar{Y}_1) and (\bar{X}_2, \bar{Y}_2) the center coordinates of two ellipsoids on the image plane, as shown in Fig. 2. Line that passes through these two ellipsoid centers is described as:

$$Y = (X - \bar{X}_1)\alpha + \bar{Y}_1, \quad \alpha = \frac{\bar{Y}_2 - \bar{Y}_1}{\bar{X}_2 - \bar{X}_1}. \quad (45)$$

This line is cut by elliptical contours into three segments, the two of which are embraced within these elliptical contours and depicted as l_1 and l_2 in Fig. 2.

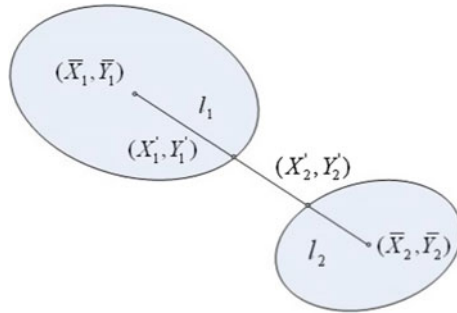


Fig. 2. Two elliptical projections on the image plane [19]

Lengths of l_1 and l_2 are developed [19] as

$$l_1 = \frac{A_1 B_1 \sqrt{\alpha^2 + 1}}{\sqrt{\mu_{02_1} + \alpha^2 \mu_{20_1} - 2\alpha \mu_{11_1}}}, \quad l_2 = \frac{A_2 B_2 \sqrt{\alpha^2 + 1}}{\sqrt{\mu_{02_2} + \alpha^2 \mu_{20_2} - 2\alpha \mu_{11_2}}}. \quad (46)$$

As a result, occlusion between these two spheres will be avoided by imposing the following inequality:

$$\sqrt{(\bar{X}_1 - \bar{X}_2)^2 + (\bar{Y}_1 - \bar{Y}_2)^2} - l_1 - l_2 > 0. \quad (47)$$

In order to realize this inequality with the same strategy proposed in (44), it is necessary to parameterize this condition also in the form of polynomial in the path abscise, as an additional constraint integrated into the minimization problem of (44). We first deduce that

$$\begin{aligned} \mu_{20} &= k_1(k_0 \bar{X}^2 + 2k_2 \bar{X} \bar{Y} + k_1 \bar{Y}^2 - k_5)/g_d, \\ \mu_{11} &= k_2(k_0 \bar{X}^2 + 2k_2 \bar{X} \bar{Y} + k_1 \bar{Y}^2 - k_5)/g_d, \\ \mu_{02} &= k_0(k_0 \bar{X}^2 + 2k_2 \bar{X} \bar{Y} + k_1 \bar{Y}^2 - k_5)/g_d. \end{aligned} \quad (48)$$

where g_d is given in (40). Bring (48) in (46) will result in

$$l_i = \sqrt{\frac{(1 + \alpha^2)v_i}{k_{0i} + \alpha^2 k_{1i} + 2\alpha k_{2i}}}, \quad i = 1, 2. \quad (49)$$

where $v_i = k_{0i} \bar{X}_i^2 + 2k_{2i} \bar{X}_i \bar{Y}_i + k_{1i} \bar{Y}_i^2 - k_{5i}$. Inequality (47) is then transformed into

$$\sqrt{(\bar{X}_1 - \bar{X}_2)^2} - \sqrt{\frac{v_1}{k_{01} + \alpha^2 k_{11} + 2\alpha k_{21}}} - \sqrt{\frac{v_2}{k_{02} + \alpha^2 k_{12} + 2\alpha k_{22}}} > 0. \quad (50)$$

Since $2a^2 + 2b^2 \geq (a + b)^2$, we realize (50) by

$$(\bar{X}_1 - \bar{X}_2)^2 - 2 \frac{v_1}{k_{01} + \alpha^2 k_{11} + 2\alpha k_{21}} - 2 \frac{v_2}{k_{02} + \alpha^2 k_{12} + 2\alpha k_{22}} > 0. \quad (51)$$

The left hand side of (51) can be further developed as function of $k_i, i = 0, \dots, 5$ in (38), therefore it is also a polynomial in w . Since it involves parameters of two elliptical projections, it is quite tedious to derive polynomial coefficients for the above expression. In order to illustrate it can be polynomial parametrized, however, we take a simpler form by taking the larger one of l_1 and l_2 (take l_1 for example) instead of the smaller one, then we will have:

$$(\bar{X}_1 - \bar{X}_2)^2 - 4 \frac{v_1}{k_{01} + \alpha^2 k_{11} + 2\alpha k_{21}} > 0. \quad (52)$$

This will be realized by the positivity of

$$\begin{cases} g_9 = g_{d1}^2 g_{d2}^2 [(\bar{X}_1 - \bar{X}_2)^2 k_{01} + (\bar{Y}_1 - \bar{Y}_2)^2 k_{11} + 2k_{21}(\bar{X}_1 - \bar{X}_2)(\bar{Y}_1 - \bar{Y}_2)], \\ g_{10} = g_9 - 4g_{d1}^2 g_{d2}^2 (k_{01} \bar{X}_1^2 + 2k_{21} \bar{X}_1 \bar{Y}_1 + k_{11} \bar{Y}_1^2 - k_{51}), \end{cases} \quad (53)$$

where $g_{d1} = k_{01}k_{11} - k_{21}^2$ and $g_{d2} = k_{02}k_{12} - k_{22}^2$. Considering $\bar{X}_1 = (k_{21}k_{41} - k_{11}k_{31})/g_{d1}$, $\bar{X}_2 = (k_{22}k_{42} - k_{12}k_{32})/g_{d2}$, $\bar{Y}_1 = (k_{21}k_{31} - k_{01}k_{41})/g_{d1}$ and $\bar{Y}_2 = (k_{22}k_{32} - k_{02}k_{42})/g_{d2}$, we can further develop (53) as:

$$\begin{cases} g_9 = k_{01}\alpha_x^2 + k_{11}\alpha_y^2 + 2k_{21}\alpha_x\alpha_y, \\ g_{10} = g_9 - 4g_{d2}^2(k_{01}(k_{21}k_{41} - k_{11}k_{31})^2 \\ \quad + 2k_{21}(k_{21}k_{41} - k_{11}k_{31})(k_{21}k_{31} - k_{01}k_{41}) \\ \quad + k_{11}(k_{21}k_{31} - k_{01}k_{41})^2 - k_{51}g_{d1}^2), \end{cases} \quad (54)$$

with

$$\begin{cases} \alpha_x = g_{d2}(k_{21}k_{41} - k_{11}k_{31}) - g_{d1}(k_{22}k_{42} - k_{12}k_{32}), \\ \alpha_y = g_{d2}(k_{21}k_{31} - k_{01}k_{41}) - g_{d1}(k_{22}k_{32} - k_{02}k_{42}). \end{cases} \quad (55)$$

It is obvious that g_9 and g_{10} are also polynomials in w based on the expressions in (39) and (43). Therefore they can be brought into the minimization problem in (44) with $n = 10$ to provide additional requirement of occlusion avoidance.

4.3 Tracking the Planned Elliptical Projections

The planned camera path and the associated elliptical projections are going to be tracked by an IBVS controller:

$$\mathbf{T} = -\lambda_1 \hat{\mathbf{L}}^+(\mathbf{s}(t) - \mathbf{s}_i^*), \quad i = 1, \dots, num_seg. \quad (56)$$

Here, λ_1 is a positive gain, usually $\lambda_1 = 0.1$. The desired feature values in \mathbf{s}_i^* are values extracted from the planned elliptical projections. Actually, we equally divide the planned camera path into num_seg segments, and extract feature values from elliptical projections at the ends of these segments and assign sequentially these values into \mathbf{s}_i^* . For every \mathbf{s}_i^* , controller (56) guides the camera motion with updated $\mathbf{s}(t)$ until the largest element in feature error $\mathbf{s}(t) - \mathbf{s}_i^*$ is below a threshold, usually set as 1 pixel. In details, we compute the length of the planned camera path and denoted it as $path_length$, select a segment length to be denoted as $segment_length$, and then derive the value of num_seg in (56) as the nearest integer to the division of $path_length/segment_length$.

Actually, the planned camera path is interpolated with several intermediate values and then produce a few segments, the end of which corresponds to the path abscise value computed as $w_i = i/num_seg$, $i = 1, \dots, num_seg$. At different stages of visual servoing, planned feature values at w_i are computed and treated as \mathbf{s}_i^* . Overall, the tracking performance is mainly dependent on the selection of $segment_length$ and the derived number of segments.

5 Evaluation Examples

Synthetic scenes are generated using MATLAB. One aims to follow a straight line in the Cartesian space while keeping camera FOV of two circles. The other deals with FOV limits and occlusion avoidance of three spheres. In either case, a reference view and an initial view of the target are generated with camera calibration errors and image noises. Image noises conforming to uniform distribution on the interval $[-1, 1]$ are piled onto the selected features. Rotational camera path recorded in iterations is expressed as three trajectories of coordinates in $\mathbf{r} = [r_1, r_2, r_3]^\top$, which is Cayley representation of a rotation matrix \mathbf{R} in the sense that $\mathbf{R} = e^{[\mathbf{r}]^\times}$. Translational camera path is presented along x , y and z coordinates.

5.1 Following a Straight Line with Two Circles

The first example aims to follow a straight line in the Cartesian space while keeping camera FOV of two intersected circles, see Fig. 3a. A desired camera view of these circles is given in Fig. 3b as a reference. There are more than one camera frames, at least two in this case, will produce the desired view. Fig. 3a draws one possibility, that is camera frame F^* . Frame F° labels the one that gives an initial view, as plotted in Fig. 3c. Image noises and camera calibration

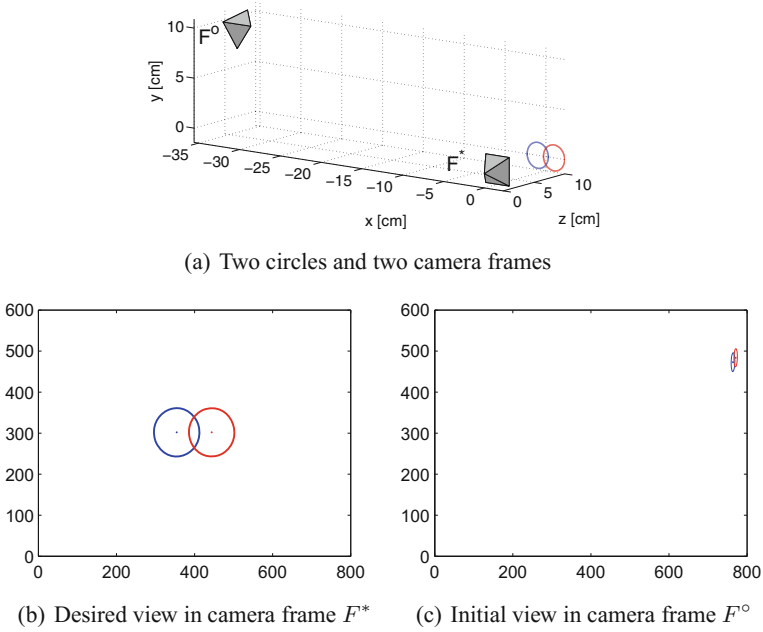


Fig. 3. Scenery with *two circles*

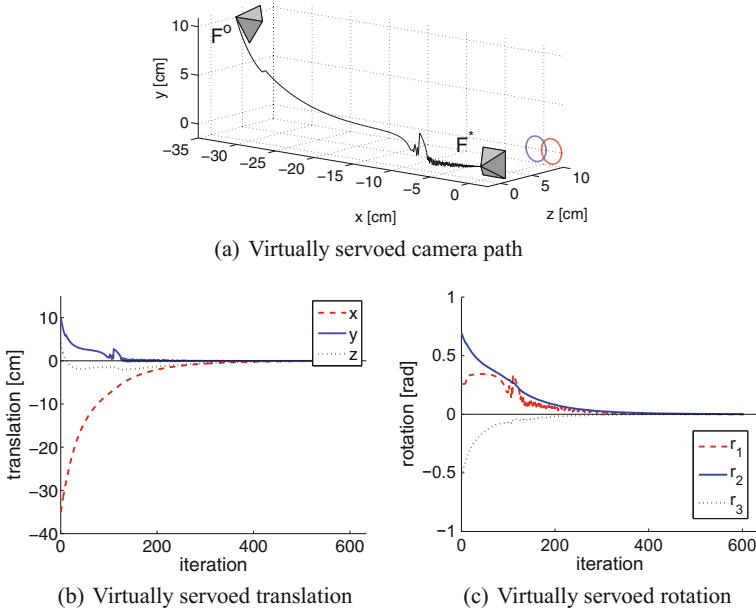


Fig. 4. Virtual visual servoing process with *two circles*

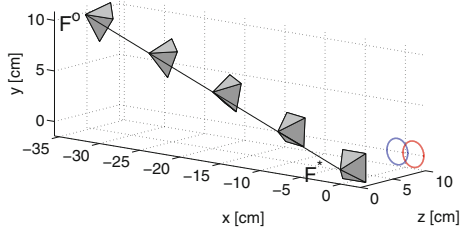
errors are considered with:

$$\mathbf{K} = \begin{pmatrix} 448 & 0 & 399 \\ 0 & 453 & 302 \\ 0 & 0 & 1 \end{pmatrix}.$$

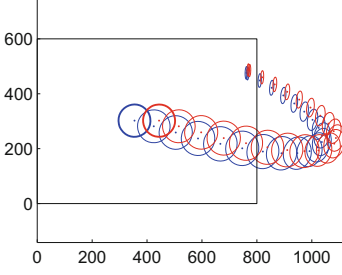
From two views of these circles (position and model are approximated to be known *a priori*), a virtual visual servoing process is performed to move the camera frame from F^o to the one that produces the desired view. Fortunately, the camera converges to the pose of F^* . Figure 4 shows the virtually servoed camera path in the Cartesian space. It fluctuates obviously near the end of the path F^* , mainly along the y -axis and r_1 coordinate, as shown in the iteration records plotted in Fig. 4b, c. The estimated camera displacement is:

$$\begin{aligned} (r_1, r_2, r_3)^\top &= (0.2630, 0.6949, -0.5228)^\top \quad \text{rad}, \\ (x, y, z)^\top &= (-34.9791, 10.0063, 3.9968)^\top \quad \text{cm}. \end{aligned}$$

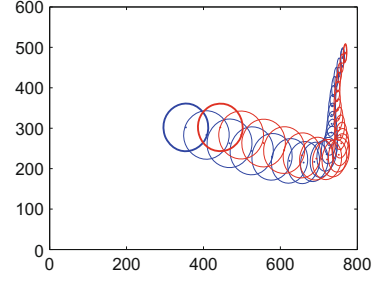
A straight path is planned when variables in (23) are initialized as zeros. The corresponding image trajectory is plotted in Fig. 5b. In this figure, elliptical projections travel from their initial view ultimately to the desired view with their trajectories, however, obviously go across image boundary of $\zeta_x \times \zeta_y = 800 \times 600$. Figure 5b displays this boundary with a rectangle. This severe violation of camera FOV limits motivates polynomial optimization in (44) with \mathbf{d} kept as zeros



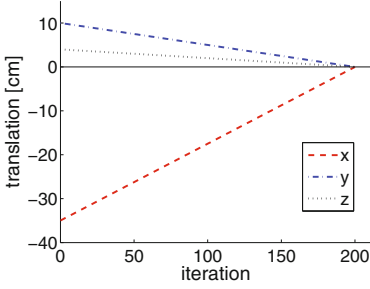
(a) Planned straight path in the Cartesian space



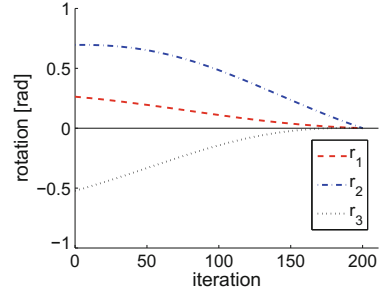
(b) Planned image trajectories satisfying only a straight path



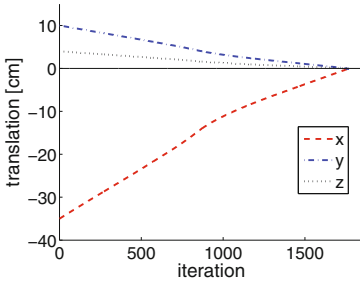
(c) Planned image trajectories satisfying FOV limits and a straight path



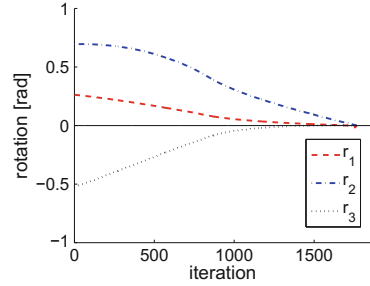
(d) Planned iteration of translation



(e) Planned iteration of rotation



(f) Servoed camera translation



(g) Servoed camera rotation

Fig. 5. Visual servoing path-planning with *two circles*

and \mathbf{b} varying until appropriate \mathbf{b}^* is achieved to give satisfactory trajectories. Figure 5c shows these well planned image trajectories, which still correspond to a straight camera path shown in Fig. 5a. Planned iterations of camera translation and rotation are respectively plotted in Fig. 5d and e. Compared to the virtually servoed iterations in Fig. 4b, c, the planned ones have no oscillation and have all of the translational coordinates converge linearly to their destinations.

The proposed strategy calculates the length of the planned camera path in Fig. 5a, and equally divides it into several segments. Image projections formed at the end of these segments are inserted and treated as the desired feature value in (56). Ultimately, the tracking algorithm in Sect. 4.3 with $path_length = 36.6011$ and $segment_length = 0.2000$ results in servoed paths in Fig. 5f, g. Although it is an image-moment based visual servoing process, the servoed translational path follows the planned one very well.

5.2 Occlusion Avoidance Among Three Spheres

The scenario for visual servoing path-planning with three spheres is illustrated in Fig. 6a. Occlusion avoidance among these spheres and the maintenance of camera view of them are expected to enable a VS application. In Fig. 6a, there are two exemplary camera frames that give, respectively, a reference and an original view. Figure 3b shows the desired image projection of these spheres in F^* , and Fig. 3c the initial one in F° . Image noises are piled and approximated camera intrinsic parameters are given as:

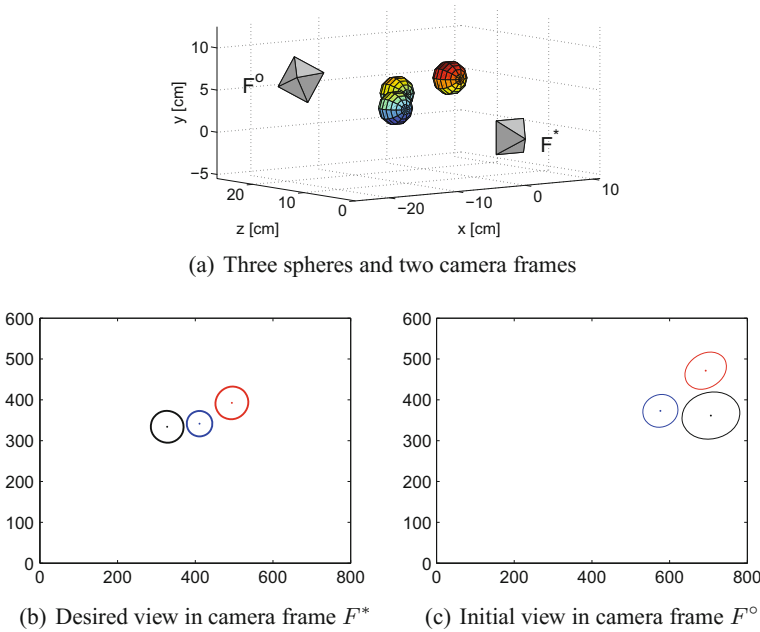


Fig. 6. Scenery with *three spheres*

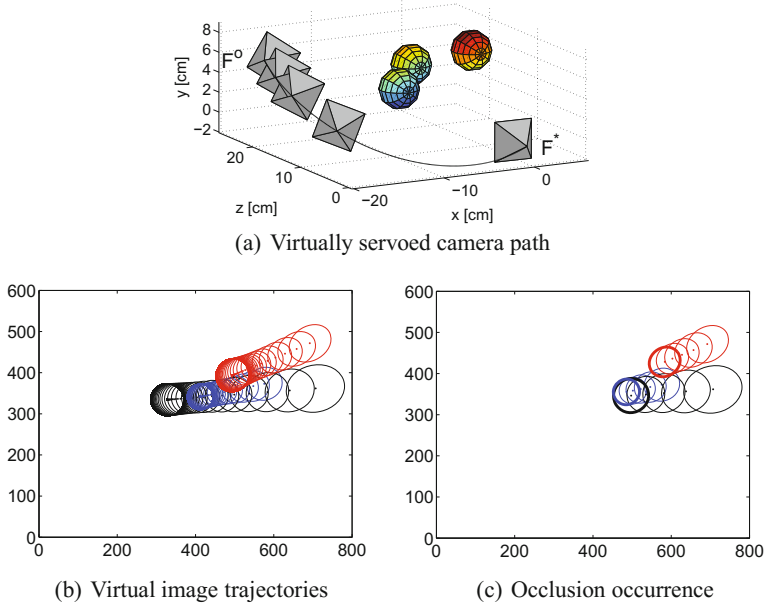


Fig. 7. Virtual visual servoing process with *three spheres*

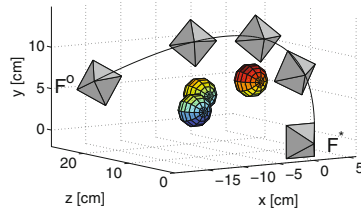
$$\mathbf{K} = \begin{pmatrix} 412 & 0 & 411 \\ 0 & 386 & 295 \\ 0 & 0 & 1 \end{pmatrix}.$$

Given the locations and 3D models of spheres and their two views, camera pose of F^o is estimated using a virtual VS method based on features extracted from two views. The estimated results are:

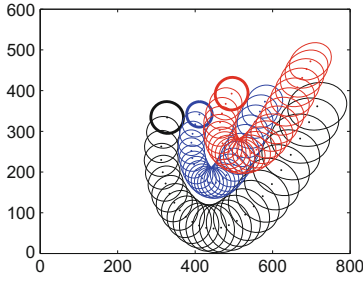
$$\begin{aligned} (r_1, r_2, r_3)^\top &= (0.0087, 0.8442, -0.5209)^\top \quad \text{rad}, \\ (x, y, z)^\top &= (-20.0225, 7.0415, 18.1116)^\top \quad \text{cm}. \end{aligned}$$

In the virtual servo process, F^o moves towards F^* and leaves its track plotted in Fig. 7a. The related elliptical projections have their tracks displayed in Fig. 7b. We cut out a small part of these tracks in Fig. 7c with elliptical projections in a certain step emphasized by their thicker contour lines, it is obvious that, in this step, there are two spheres have their projections overlap with each other. This situation will prevent feature extraction of the occluded sphere and fail immediately a real VS application.

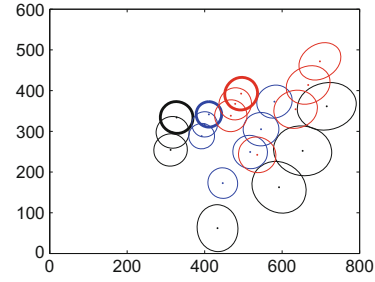
The proposed path-planning strategy aims at a path that avoids image boundary of $\zeta_x \times \zeta_y = 800 \times 600$ as well as occlusion among these spheres. The resulting camera path is displayed in Fig. 8a, with its translational and rotational iterations plotted in Fig. 8d, e. Related elliptical projections in Fig. 8b and a selected few of them in Fig. 8c show perfect occlusion avoidance and satisfaction of camera FOV limits. We use the strategy in Sect. 4.3 with *path.length* =



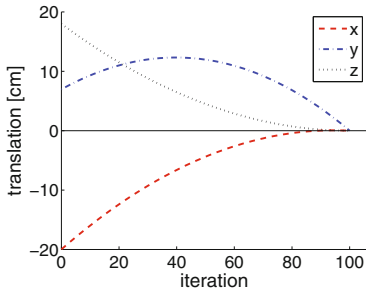
(a) Planned camera path



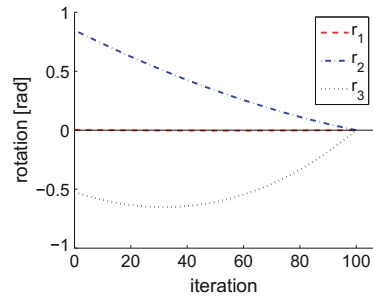
(b) Planned image trajectories satisfying occlusion avoidance and FOV limits



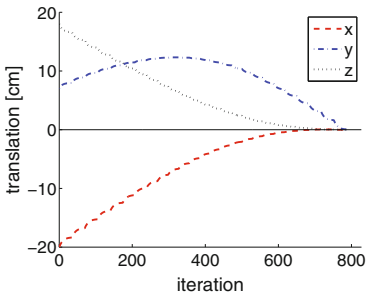
(c) Partial image projections showing perfect constraints satisfaction



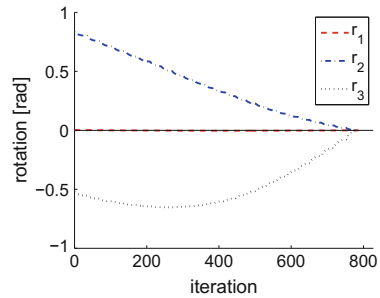
(d) Planned iteration of translation



(e) Planned iteration of rotation



(f) Servoed camera translation



(g) Servoed camera rotation

Fig. 8. Visual servoing path-planning with *three spheres*

36.1924 and *segment_length* = 1.4477 to track the planned trajectories in Fig. 8b and plot the recorded servo data in Fig. 8f, g. Compared to the planned ones in Fig. 8d, e, it can be seen that the servoed camera path follow closely the planned one. Therefore, the servoed image trajectories will also follow up the planned ones in Fig. 8b, satisfying all of the considered constraints.

6 Conclusions

This chapter proposes a path planning approach for visual servoing with elliptical projections. The problem consists of planning a trajectory that ensures the convergence of elliptical projections to their desired/reference view while satisfying target visibility and workspace constraints. Moment-based features are constructed from these elliptical projections. Constraints required by the usage of moment-based features are much more demanding than that of pixel coordinates of some representational points. We propose to parametrize these constraints into polynomial inequalities with some common variable coefficients to be optimized. Based on a well-planned path, we introduce a new scheme to follow closely the planned elliptical trajectories. Simulation results of two main situations validate the proposed approach. In the first situation, two intersected circles are observed. In the second situation, it is supposed that at least three spheres are observed. Both of them demonstrate excellent performance. In the future, it is also worth exploring other primitives such as cylinders. Another possibility is to compare the robustness of different moment-based features to errors in calibration and image noise.

Acknowledgements. This work is supported by Hunan Provincial Natural Science Foundation of China (Grant No. 2017JJ3211).

References

1. Chaumette, F.: De la perception à l'action: l'asservissement visuel; de l'action à la perception: la vision active. Habilitation à diriger les recherches, Habilitation à diriger des recherches Université de Rennes 1 (1998)
2. Chaumette, F.: Potential problems of stability and convergence in image-based and position-based visual servoing. In: Kriegman, D., Hager, G., Morse, A.S. (eds.) *The Confluence of Vision and Control*. LNCIS, vol. 237, pp. 66–78. Springer, Heidelberg (1998)
3. Chaumette, F.: Image moments: a general and useful set of features for visual servoing. *IEEE Trans. Robot.* **20**(4), 713–723 (2004)
4. Chaumette, F., Hutchinson, S.: Visual servo control, part I: basic approaches. *IEEE Robot. Autom. Mag.* **13**(4), 82–90 (2006)
5. Chesi, G., Hashimoto, K.: A simple technique for improving camera displacement estimation in eye-in-hand visual servoing. *IEEE Trans. Pattern Anal. Mach. Intell.* **26**(9), 1239–1242 (2004)
6. Chesi, G., Hashimoto, K. (eds.): *Visual Servoing via Advanced Numerical Methods*. Springer, Heidelberg (2010)

7. Chesi, G., Shen, T.: Conferring robustness to path-planning for image-based control. *IEEE Trans. Control Syst. Technol.* **20**(4), 950–959 (2012)
8. Chum, O., Pajdla, T., Sturm, P.: The geometric error for homographies. *Comput. Vis. Image Underst.* **97**(1), 86–102 (2005)
9. Collewet, C., Marchand, E.: Luminance: a new visual feature for visual servoing. In: Chesi, G., Hashimotos, K. (eds.) *Visual Servoing via Advanced Numerical Methods*. LNCIS, vol. 401, pp. 71–90. Springer, Heidelberg (2010)
10. Cowan, N., Weingarten, J., Koditschek, D.: Visual servoing via navigation functions. *IEEE Trans. Robot. Autom.* **18**(4), 521–533 (2002)
11. Dhome, M., Richetin, M., Lapreste, J.T.: Determination of the attitude of 3d objects from a single perspective view. *IEEE Trans. Pattern Anal. Mach. Intell.* **11**(12), 1265–1278 (1989)
12. Hartley, R.I.: In defense of the eight-point algorithm. *IEEE Trans. Pattern Anal. Mach. Intell.* **19**(6), 580–593 (1997)
13. Hashimoto, K., Kimoto, T., Ebine, T., Kimura, H.: Manipulator control with image-based visual servo. In: *8th IEEE International Conference on Robotics and Automation*, pp. 2267–2272. San Francisco, CA (1991)
14. Latombe, J.C.: *Robot Motion Planning*. Kluwer Academic Publisher (1991)
15. M. Kazemi, K.G., Mehrandezh, M.: Path-planning for visual servoing: A review and issues. In: Chesi, G., Hashimotos, K. (eds.) *Visual Servoing via Advanced Numerical Methods*. LNCIS, vol. 401, pp. 189–207. Springer, Heidelberg (2010)
16. Mezouar, Y., Chaumette, F.: Path planning for robust image-based control. *IEEE Trans. Robot. Autom.* **18**(4), 534–549 (2002)
17. Nister, D.: An efficient solution to the five-point relative pose problem. *IEEE Trans. Pattern Anal. Mach. Intell.* **26**(6), 756–770 (2004)
18. Safae-Rad, R., Tchoukanov, I., Smith, K.C., Benhabib, B.: Three-dimensional location estimation of circular features for machine vision. *IEEE Trans. Robot. Autom.* **8**(5), 624–640 (1992)
19. Shen, T., Chesi, G.: Visual servoing path-planning with spheres. In: *9th International Conference on Informatics in Control. Automation and Robotics*, pp. 22–30. Italy, Rome (2012)
20. Shen, T., Chesi, G.: Following a straight line in visual servoing with elliptical projections. In: *13th International Conference on Informatics in Control, Automation and Robotics*. pp. 47–56. Lisbon, Portugal (2016)
21. Tahri, O., Chaumette, F.: Complex objects pose estimation based on image moment invariants. In: *22th IEEE International Conference on Robotics and Automation*. pp. 438–443. Barcelona, Spain (2005)
22. Tahri, O., Mezouar, Y., Chaumette, F., Araujo, H.: Visual servoing and pose estimation with cameras obeying the unified model. In: Chesi, G., Hashimotos, K. (eds.) *Visual Servoing via Advanced Numerical Methods*. LNCIS, vol. 401, pp. 231–252. Springer, Heidelberg (2010)
23. Tatsambon Fomena, R., Chaumette, F.: Visual servoing from spheres with paracatadioptric cameras. In: *13th International Conference on Advanced Robotics*. Jeju, Korea (2007)
24. Taylor, C., Ostrowski, J.: Robust vision-based pose control. In: *17th IEEE International Conference on Robotics and Automation*. pp. 2734–2740. San Francisco, CA (2000)
25. Triggs, B., McLauchlan, P., Hartley, R., Fitzgibbon, A.: Bundle adjustment—a modern synthesis. In: W. Triggs, A.Z., Szeliski, R. (eds.) *Vision Algorithms: Theory and Practice*, LNCS, vol. 1883, pp. 298–372. Springer, Heidelberg (2000)

Informatics in Control, Automation and Robotics
13th International Conference, ICINCO 2016 Lisbon,
Portugal, 29-31 July, 2016

Madani, K.; Peaucelle, D.; Gusikhin, O. (Eds.)
2018, XVII, 441 p. 206 illus., 164 illus. in color.,
Hardcover

ISBN: 978-3-319-55010-7