The Epipolar Geometry Toolbox

Multiple View Geometry and Visual Servoing for MATLAB

The Epipolar Geometry Toolbox (EGT) was created to provide MATLAB users with an extensible framework for the creation and visualization of multicamera scenarios as well as for the manipulation of the visual information and the geometry between them. Functions provided, for both pinhole and panoramic vision sensors, include camera placement and visualization, computation, estimation of epipolar geometry entities, and many others. The compatibility of EGT with the Robotics Toolbox [7] enables users to address general vision-based control issues. Two applications of EGT to visual servoing tasks are examined in this article. We introduce the toolbox in tutorial form, and examples are provided to demonstrate its capabilities. The complete toolbox, detailed manual, and demo examples are freely available on the EGT Web site [21].

EGT

The MATLAB [29] software environment is available for a wide range of platforms and designed around linear algebra principles and graphical presentations for large datasets. Its core functionalities are extended through the use of many additional toolboxes. Combined with the interactive MATLAB environment and advanced graphical functions, EGT provides a wide set of functions for approaching computer vision problems, especially with multiple views.

EGT facilitates the design of vision-based control systems for both pinhole and central catadioptric (or omnidirectional) cameras. EGT is fully compatible with the well-known Robotics Toolbox by Corke [7]. Increasing interest in robotic visual servoing for both six degree of freedom (6-DOF) kinematic chains and mobile robots equipped with pinhole or omnidirectional cameras fixed to the workspace or to the robot motivated the development of EGT.

Several authors, including [4], [9], [18], [20], and [24], have proposed new visual servoing strategies based on the geometry relating multiple views acquired from different camera configurations, i.e., the epipolar geometry [14].

We have observed the necessity of a software environment to help researchers rapidly create a multiple-camera setup, use visual data, and design new visual servoing algorithms. With EGT, we provide a wide set of easy-to-use and completely customizable functions for designing general multicamera scenarios and manipulating the visual information between them.

Let us emphasize that EGT can also be successfully employed in many other contexts where single- and multiple-view geometry is involved, as, for example, in visual odometry and in structure from motion applications [22], [23]. For example, in [23], an interesting "visual odometry" approach for robot simultaneous localization and mapping (SLAM) is proposed in which the multiple-view geometry is used to estimate the camera motion from pairs of images without requiring the knowledge of the observed scene.

EGT, like the Robotics Toolbox, is a simulation environment, yet the EGT functions can be easily embedded by the user in Simulink models. In this way, thanks to the MATLAB Real-Time Workshop, the user can generate and execute stand-alone C code for many off-line and real-time applications.

A distinguishable feature of EGT is that it can be used to create and manipulate visual data provided by both pinhole

BY GIAN LUCA MARIOTTINI AND DOMENICO PRATTICHIZZO

1070-9932/05/$20.00©2005 IEEE

and central catadioptric cameras. These kinds of omnidirectional vision sensors, due to their wide fields of view, have recently been applied in visual servoing [32].

The second motivation that lead to the development of EGT was the increasing distribution of "free" software in recent years, on the basis of the Free Software Foundation [10] principles. In this manner, users are allowed, and also encouraged, to adapt and improve the program as dictated by their specific needs. Examples of programs that follow these principles include the Robotics Toolbox [7] for the creation of simulations in robotics and Intel's OpenCV C++ libraries for the implementation of computer vision algorithms, such as image processing and object recognition [1].

The third important motivation for EGT was the availability and increasing sophistication of MATLAB. EGT could have been written in other languages, such as C or C++, and this would have freed it from dependency on other software. However, these low-level languages are not as conducive to rapid program development as MATLAB.

This tutorial assumes the reader has familiarity with MATLAB and presents the basic EGT functions, after short theory recalls, together with intuitive examples. We also present two applications of EGT to visual servoing.

After exploring the basic vector notation in EGT, we present the pinhole and omnidirectional camera models together with EGT basic functions. We then present the setup for multiple-camera geometry (epipolar geometry) and investigate two applications of EGT to visual servoing together with simulation results. We compare EGT with other software packages. EGT can be freely downloaded at [21]. It can be used under Windows and Linux and requires MATLAB 6.5 or higher. The detailed manual is provided in the EGT Web site, with a large set of examples, figures, and source code for beginners.

Basic Vector Notation

We here present the basic vector notation adopted in EGT. All scene points $\mathbf{X}_w \in \mathbb{R}^3$ are expressed in the world frame $\mathcal{S}_w =< O_w x_w y_w z_w >$ (Figure 1). When referring to the pinhole camera frame $\mathcal{S}_c =< O_c x_c y_c z_c >$, they will be indicated with $\mathbf{X}_c$. Moreover, all scene points expressed with respect to a central catadioptric camera frame $\mathcal{S}_m =< O_m x_m y_m z_m >$ will be indicated by $\mathbf{X}_m$. For the reader's convenience, we briefly present the basic vector notation and transformation [25]. Refer to Figure 1 and consider the 3×1 vector $\mathbf{X}_c \in \mathcal{S}_c$. It can be expressed in $\mathcal{S}_w$ as

$$\mathbf{X}_w = \mathbf{R}_w^c \mathbf{X}_c + \mathbf{t}_w^c, \tag{1}$$

where $\mathbf{t}_w^c$ is the translational vector centered in $\mathcal{S}_w$ and pointing toward the $\mathcal{S}_c$ frame (Figure 1). The matrix $\mathbf{R}_w^c$ is the rotation necessary to align the world frame with the camera frame. For example, we may choose $\mathbf{R}_w^c = \mathbf{R}_{\text{roll,pitch,yaw}} = \mathbf{R}_{z,\theta}\mathbf{R}_{y,\phi}\mathbf{R}_{x,\psi}$. The homogeneous notation aims to express (1) in linear form:

$$\widetilde{\mathbf{X}}_w = \mathbf{H}_w^c \widetilde{\mathbf{X}}_c,$$

where $\widetilde{\mathbf{X}}_w = [\mathbf{X}_w{}^T \ 1]^T$, $\widetilde{\mathbf{X}}_c = [\mathbf{X}_c{}^T \ 1]^T$. The 4×4 matrix $\mathbf{H}_w^c$ is referred to as *homogeneous transformation* matrix:

$$\mathbf{H}_w^c = \begin{bmatrix} \mathbf{R}_w^c & \mathbf{t}_w^c \\ \mathbf{0}^T & 1 \end{bmatrix}.$$

Analogously, a point $\mathbf{X}_w$ can be expressed in the camera frame by the following transformation

$$\widetilde{\mathbf{X}}_c = \begin{bmatrix} \mathbf{R}_w^{c\,T} & -\mathbf{R}_w^{c\,T}\mathbf{t}_w^c \\ \mathbf{0}^T & 1 \end{bmatrix} \widetilde{\mathbf{X}}_w. \tag{2}$$

Consider now the more general case in which two camera frames, referred to as *actual* and *desired*, are observing the same point $\mathbf{X}_w$. From (1)

$$\mathbf{X}_w = \mathbf{R}_w^d \mathbf{X}_d + \mathbf{t}_w^d \tag{3}$$

$$\mathbf{X}_w = \mathbf{R}_w^a \mathbf{X}_a + \mathbf{t}_w^a \tag{4}$$

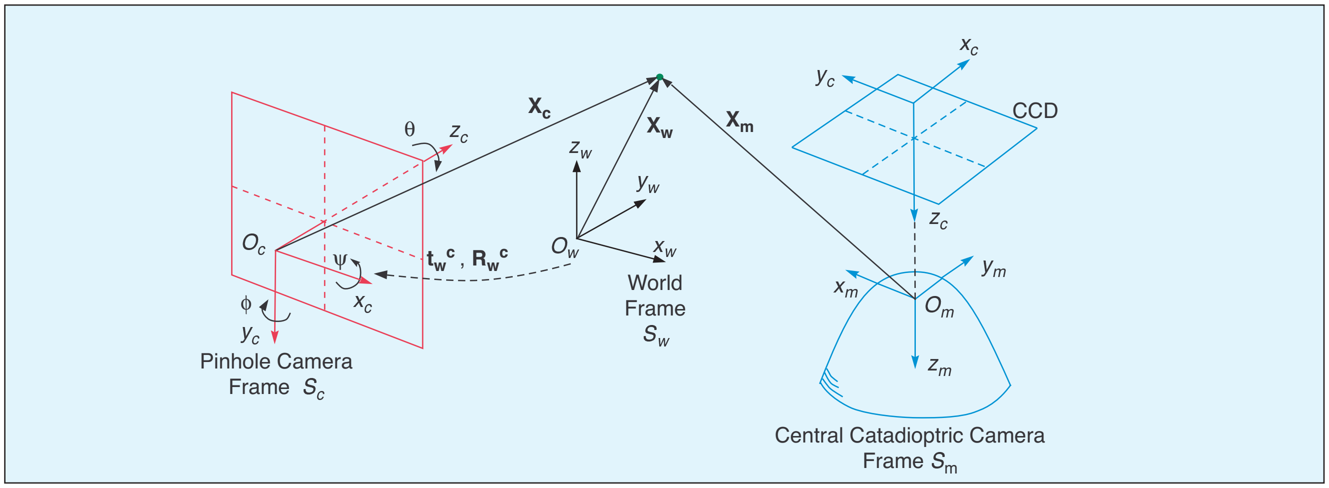

Figure 1. *Main reference frames notation and vector representation in EGT.*

Substituting (4) in (3), it follows

$$\mathbf{X}_d = \underbrace{\mathbf{R}_w^{d\,T}\mathbf{R}_w^a}_{\mathbf{R}_d^a}\mathbf{X}_a + \underbrace{\mathbf{R}_w^{d\,T}\left(\mathbf{t}_w^a - \mathbf{t}_w^d\right)}_{\mathbf{t}_d^a}. \tag{5}$$

Equation (5) will be very useful in EGT for the analytical computation of epipolar geometry, where it is necessary to know the relative displacement $\mathbf{t}_d^a$ and orientation $\mathbf{R}_d^a$ between the two camera frames.

Pinhole and Omnidirectional Camera Models

EGT provides easy-to-use functions for the placement of pinhole and central catadioptric (or omnidirectional) cameras. Their imaging model has been here implemented to allow users to manipulate the visual information. In this section, the fundamentals of perspective and omnidirectional camera models are briefly reviewed. The reader is referred to [5], [14], and [17] for a detailed treatment. To fulfill the purposes of this tutorial, some basic EGT code examples are reported together with the theory.

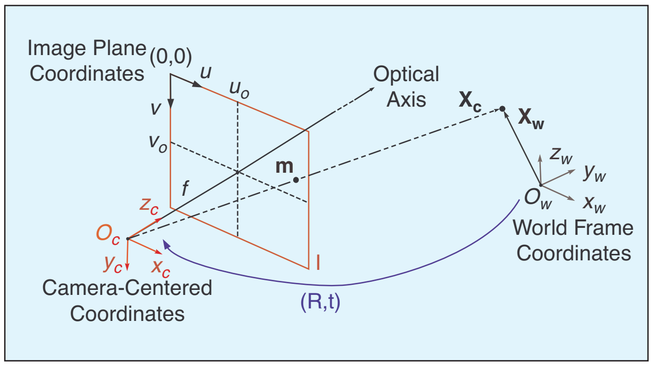

Figure 2. *The pinhole camera model. The 3-D point* $\mathbf{X}_w$ *is projected onto* $\mathbf{m}$ *through the optical center* O_C*. Note that* $\mathbf{m}$ *is expressed in the image plane coordinates* (u, v) *(pixels).*

Perspective Camera

Consider a pinhole camera located at O_c, as in Figure 2. The full-perspective model describes the relationship between a three-dimensional (3-D) point (in homogeneous coordinates) $\widetilde{\mathbf{X}}_w = [X \quad Y \quad Z \quad 1]^T$ expressed in the world frame and its projection $\tilde{\mathbf{m}} = [u \quad v \quad 1]^T$ onto the image plane according to

$$\widetilde{\mathbf{m}} = \mathbf{K}\boldsymbol{\Pi}\widetilde{\mathbf{X}}_w,$$

where $\mathbf{K} \in \mathbb{R}^{3\times3}$ is the camera's intrinsic parameters matrix given by

$$\mathbf{K} = \begin{bmatrix} k_u f & \gamma & u_0 \\ 0 & k_v f & v_0 \\ 0 & 0 & 1 \end{bmatrix}.$$

Here, (u_0, v_0) are the pixel coordinates in the image frame of the principal point (i.e., the intersection point between the image plane and the optical axis z_c), k_u and k_v are the number of pixels per unit distance in image coordinates, f is the focal length (in meters), and γ is the orthogonality factor of the charge-coupled device (CCD) image axes (*skew factor*).

Matrix $\boldsymbol{\Pi} = [\mathbf{R}\,|\,\mathbf{t}] \in \mathbb{R}^{3\times4}$ is the so-called *external parameters* camera matrix, which contains the rotation $\mathbf{R}$ and the translation $\mathbf{t}$ between the world and the camera frames. According to the commonly used notation, in the case of no camera rotation, the optical axis z_c of pinhole cameras is parallel to the y_w axis of $\mathcal{S}_w$. We then define

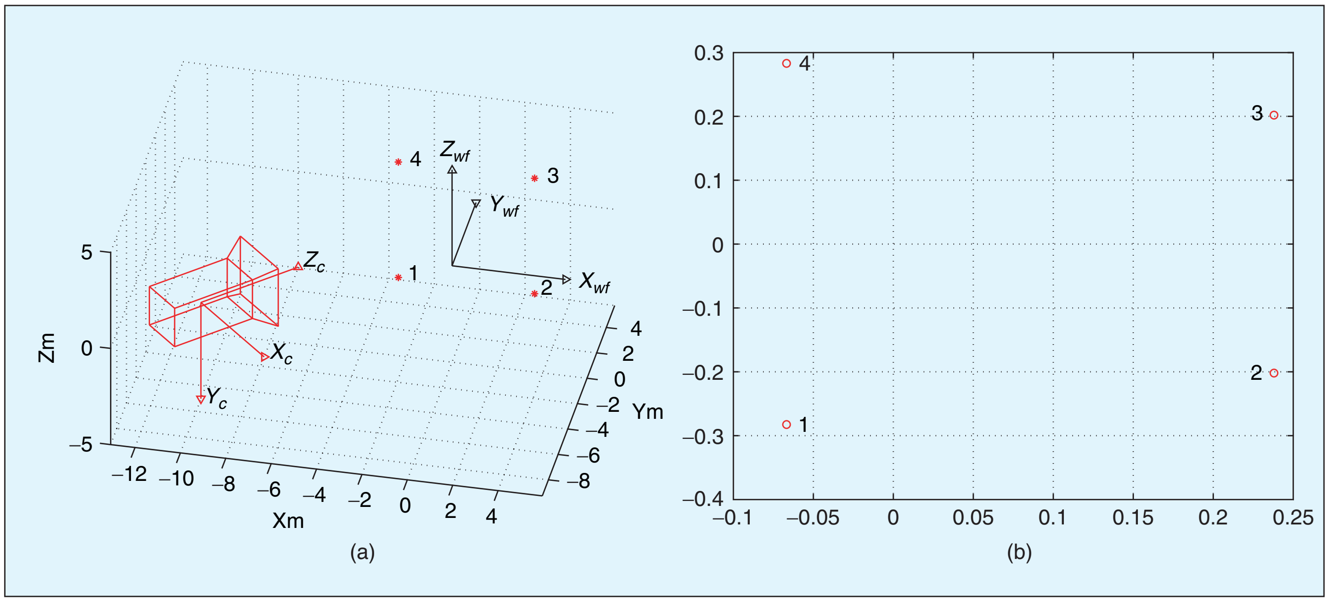

Figure 3. *Example 1. (a) A pinhole camera is positioned in* $t = [-10, -5, 0]$ *in the 3-D world frame and rotated by* $\pi/4$ *around the y axis. (b) The 3-D scene points are projected onto the image plane. Note that in this case* $\mathbf{K} = \mathbf{I}$ *for simplicity.*

$$\mathbf{R} = \left(\mathbf{R}_{x,-\pi/2}\mathbf{R}_{rpy}\right)^T$$
$$\mathbf{t} = -\left(\mathbf{R}_{x,-\pi/2}\mathbf{R}_{rpy}\right)^T \mathbf{t}_w^c.$$

In order to directly obtain the 4×4 homogeneous matrix $\mathbf{H}_c^w$, the function `f_Rt2H` is provided

```
>> H=f_Rt2H(R,t).
```

Note that with the use of `f_Rt2H,` the position $\mathbf{t}$ of a pinhole camera is specified with respect to the *world frame,* while the rotation $\mathbf{R}$ is referred to the *pinhole camera frame* axes. During the testing phase at the University of Siena, this choice was appreciated from students of Robotics and Vision classes that addressed it as very intuitive.

Example 1. 3-D scene and pinhole camera placement

Consider now a pinhole camera rotated by R $= \mathbf{R}_{y,\pi/4} \in \mathbb{R}^{3\times3}$ and translated by t $= [-10, -5, 0]^T$:

```
>> R=rotoy(pi/4);
>> t=[-10,-5,0]';
>> H=f_Rt2H(R,t);
```

In EGT, the camera frame and the associated 3-D camera can be visualized with functions `f_3Dframe(H)` and `f_3Dcamera(H)`, respectively, where `H` is the 4×4 homogeneous transformation describing position and orientation of the camera with respect to $\mathcal{S}_w$

```
>> f_3Dframe(H,1); %camera frame
>> hold on
>> f_3Dcamera(H); %3D pinhole camera
>> axis equal, grid on, view(12,34)
>> title('3D setup - EGT Tutorial - Ex.1')
```

A plot of the 3-D view is given in Figure 3(a). All the functions have further options. See the EGT Manual [21] for details.

We can also place a set of N 3-D points $\mathbf{X}_w^i = [X^i, Y^i, Z^i]$ (e.g., the rectangular panel vertexes) defined as

$$\mathbf{X}_w = \begin{bmatrix} X^1 & X^2 & \dots & X^N \\ Y^1 & Y^2 & \dots & Y^N \\ Z^1 & Z^2 & \dots & Z^N \end{bmatrix} \in \mathbb{R}^{3\times N}$$

by the use of function `f_scenepnt(X)`

```
>> Xi=[-3, 3, 3, -3];
>> Yi=[ 3, 3, 3, 3];
>> Zi=[-3, -3, 3, 3];
>> Xw=[Xi; Yi; Zi];
>> f_scenepnt(Xw);
>> f_3Dwfenum(Xw); %enumerate points
```

The perspective projection $\mathbf{m} = [u, v]^T$ of points $\mathbf{X}_w$ is obtained with `f_persproj(Xw,H,K)`:

```
>> [u,v]=f_perspproj(Xw,H,K);
>> plot(u,v,'rO')
```

Projection of scene points is represented in Figure 3(b).

Note that while the above example describes the placement of 3-D points $\mathbf{X}_w$, EGT is also able to build scenes with more complex 3-D objects returning surface points and normals (see function `f_3Dsurface` in [21]).

Central Catadioptric Camera Model

Catadioptric cameras combine reflective surfaces (mirrors) and lenses. Several types of catadioptric cameras can be obtained simply by combining cameras (pinhole or orthographic) and mirrors (hyperbolic, parabolic, or elliptical) [5].

They are classified according to whether or not they satisfy the single-viewpoint constraint, which guarantees that the visual sensor only measures the light through a single point. Note that this constraint is required for the existence of epipolar geometry and the generation of geometrically correct images [12], [28].

In [3], Baker et al. derive the entire class of catadioptric systems verifying the single-viewpoint constraint. Among these, EGT takes into account catadioptric systems consisting of pinhole cameras coupled with hyperbolic mirrors, and orthographic cameras coupled with parabolic mirrors.

In [11], a unified projection model for central catadioptric camera systems has been proposed. In particular, it was

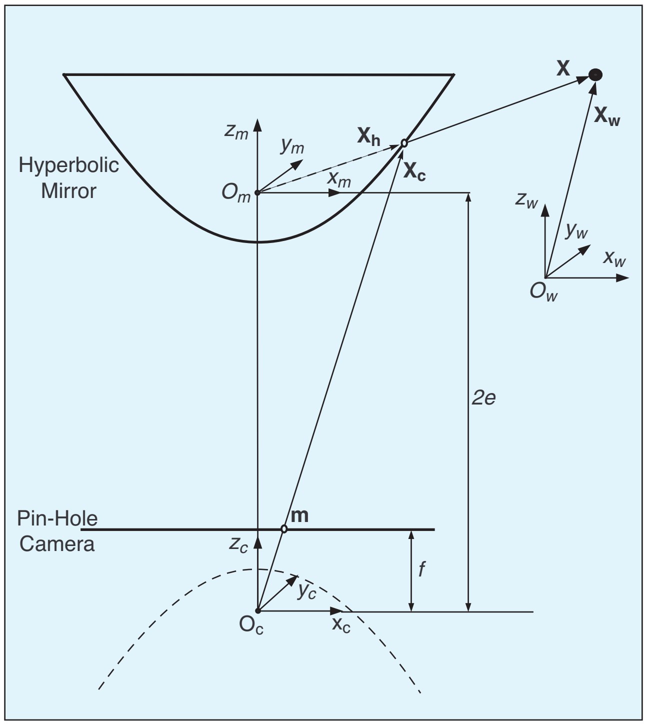

Figure 4. The panoramic camera model (pinhole camera and hyperbolic mirror). The 3-D point $\mathbf{X}_w$ is projected at $\mathbf{m}$ through the optical center O_c, after being projected at $\mathbf{X}_h$ through the mirror center O_m.

shown that all central panoramic cameras can be modeled by a particular mapping on a sphere, followed by a projection from a point on the camera optical axis onto the image plane.

In order to maintain in EGT a physically meaningful graphical representation, we decided, without loosing generality, to represent the central panoramic cameras not as spheres in the space but rather with the pairing of a CCD camera with a parabolic or hyperbolic mirror.

The imaging model for a pinhole camera with hyperbolic mirror is now described. Consider the basic scheme in Figure 4. Note that in this case, all frames (for both the camera and the mirror) are aligned with the world frame. Three important reference frames are defined: 1) the world reference system centered at O_w whose vector is $\mathbf{X}_w$; 2) the mirror coordinate system centered at the focus O_m whose vector is $\mathbf{X} = [X, Y, Z]^T$; and 3) the camera coordinate system centered at O_c, the vector of which is $\mathbf{X}_c$.

Henceforth, all equations will be expressed in the mirror reference frame if not stated otherwise.

Refer to Figure 4 and let a and b be the hyperbolic mirror parameters

$$\frac{(z+e)^2}{a^2} - \frac{x^2+y^2}{b^2} = 1$$

with eccentricity $e = \sqrt{a^2+b^2}$. The transformation to obtain the projection $\mathbf{m}$ in the pinhole camera frame (see Figure 4) is given by

$$\mathbf{m} = \mathbf{K}\frac{1}{2e}\left(\mathbf{R}_c^m\left(\lambda \mathbf{R}_w^{m\,T}\left(\mathbf{X}_w - \mathbf{t}_w^m\right)\right) + \mathbf{t}_c^m\right), \qquad (6)$$

where $\lambda = (b^2(-eZ \pm a\|\mathbf{X}\|))/(b^2Z^2 - a^2X^2 - a^2Y^2)$ is a nonlinear function of $\mathbf{X}$. $\mathbf{K}$ is the internal calibration matrix of the CCD camera looking at the mirror. The mirror center expressed in the camera frame is $\mathbf{t}_c^m$; it corresponds to $[0, 0, 2e]$. $\mathbf{R}_c^m$ is the matrix representing the rotation between camera and mirror frames. Analogously, $\mathbf{t}_w^m$ and $\mathbf{R}_w^m$ represent the mirror configuration (rotation and orientation) with respect to the world frame.

In EGT, a central catadioptric camera is defined by specifying the homogeneous transformation matrix between mirror and world frames

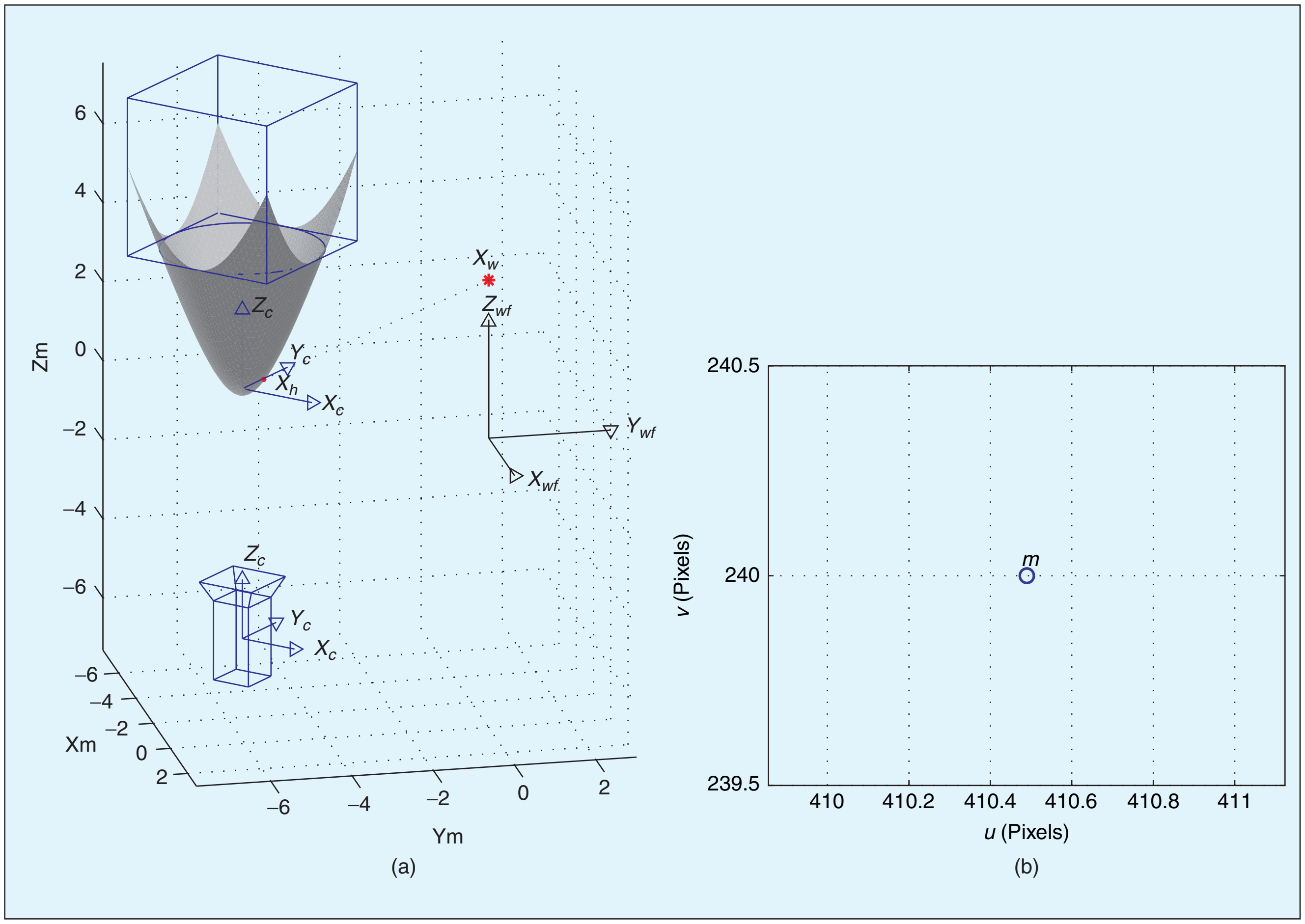

Figure 5. *Example 2. (a) A panoramic camera is positioned in $[-5, -5, 0]^T$ in the 3-D world frame and (b) the 3-D point $\mathbf{X}_w = [0, 0, 4]^T$ is projected to the pinhole camera after being projected in $\mathbf{X}_h$.*

$$\mathbf{H}_w^m = \begin{bmatrix} \mathbf{R}_w^m & \mathbf{t}_w^m \\ \mathbf{0}^T & 1 \end{bmatrix}$$

Example 2. Catadioptric camera placement

In EGT, a panoramic camera can be placed and visualized. Let us place the camera at `t=[-5,-5,0]'` with orientation `R` $\equiv R_{z,\pi/4}$. EGT provides a function to simply visualize the catadioptric camera in the 3-D world frame as in Figure 5.

```
>> H=[rotoz(pi/4), [-5,-5,0];
>>         0 0 0,    1     ];
>> f_3Dpanoramic(H);
```

Moreover, for assigned camera calibration matrix **K**

```
K=[10^3   0   320;
    0   10^3  240;
    0     0    1 ];
```

the projection of a 3-D point `Xw=[0,0,4]'`, in both the camera (`m`) and mirror (`Xh`) frames can be obtained from

```
>> [m,Xh] = f_panproj(Xw,H,K);
>>
m =
4.1048e+002
2.4000e+002
1.0000e+000

Xh =
6.0317e-001
3.7881e-017
3.4120e-001
1.0000e+000
```

Graphical results are reported in Figure 5 and more accurately described in the *EGT Reference Manual*. More examples can be found in the directory `demos`.

Epipolar Geometry

In this section, the EGT functions dealing with the epipolar geometry are presented.

Epipolar Geometry for Pinhole Cameras

Consider two perspective views of the same scene taken from distinct viewpoints O_1 and O_2, as in Figure 6. Let $\mathbf{m}_1$ and $\mathbf{m}_2$ be the corresponding points in two views, i.e., the perspective projection through O_1 and O_2 of the same point $\mathbf{X}_w$, in both image planes $\mathcal{I}_1$ and $\mathcal{I}_2$, respectively. The epipolar geometry defines the geometric relationship between these corresponding points.

Associated with Figure 6, in the following, we define the a set of geometric entities [14], [17].

Definition 1: Epipolar Geometry

1) The plane passing through the optical centers O_1 and O_2 and the scene point $\mathbf{X}_w$ is called an *epipolar plane*. Note that in a N-points scene, a pencil of planes exists (one for each scene point $\mathbf{X}_w^i$).
2) The projection $\mathbf{e}_1$ ($\mathbf{e}_2$) of one camera center O_1 (O_2) onto the image plane of the other camera frame $\mathcal{I}_2$ ($\mathcal{I}_1$) is called *epipole*. The epipole will be expressed in homogeneous coordinates

$$\tilde{\mathbf{e}}_1 = [\,e_{1x} \quad e_{1y} \quad 1\,]^T \qquad \tilde{\mathbf{e}}_2 = [\,e_{2x} \quad e_{2y} \quad 1\,]^T.$$

3) The intersection of the epipolar plane for $\mathbf{X}_w$ with image plane $\mathcal{I}_1$ ($\mathcal{I}_2$) defines the *epipolar line* $\mathbf{l}_1$ ($\mathbf{l}_2$). Note that all epipolar lines pass through the epipole.

One of the main parameters of the epipolar geometry is the 3×3 fundamental matrix $\mathbf{F}$, which conveys most of the information about the relative position and orientation $(\mathbf{t}, \mathbf{R})$ between the two views. Moreover, the fundamental matrix algebraically relates corresponding points in the two images through the epipolar constraint.

Theorem 1: Epipolar constraint

Consider two views of the same 3-D point $\mathbf{X}_w$, both characterized by their relative position and orientation $(\mathbf{t}, \mathbf{R})$ and the internal calibration parameters $\mathbf{K}_1$ and $\mathbf{K}_2$. Moreover, let $\mathbf{m}_1$ and $\mathbf{m}_2$ (homogeneous notation) be the two corresponding projections of $\mathbf{X}_w$ in both image planes. The epipolar

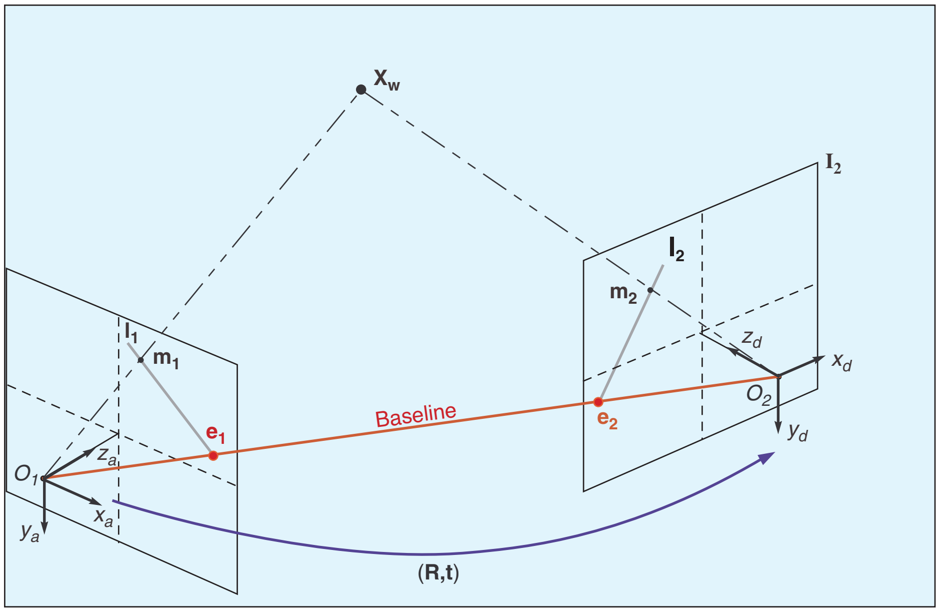

Figure 6. *Basic epipolar geometry entities for pinhole cameras.*

constraint is:

$$\mathbf{m}_2^T \mathbf{F} \mathbf{m}_1 = 0 \qquad \text{where} \qquad \mathbf{F} = \mathbf{K_2}^{-T}[\mathbf{t}]_\times \mathbf{R} \mathbf{K_1}^{-1}$$

and $[\mathbf{t}]_\times$ is the 3×3 skew-symmetric matrix associated with $\mathbf{t}$. The epipolar geometry is reviewed extensively in [14] and [17].

Example 3. Epipolar geometry for pinhole cameras

The computation of epipolar geometry with EGT is straightforward. Consider a two-view scene as in Figure 7(a), where the first camera is placed at $\mathbf{t}_1 = [-10,\ -10,\ 0]^T$ with orientation $\mathbf{R}_1 = \mathbf{R}_{y,\pi/4}$ and the second is coincident with the world frame, i.e., $\mathbf{t}_2 = [0,\ 0,\ 0]^T$ with orientation $\mathbf{R}_2 = \mathbf{I}$. pinhole cameras can be shown, and their homogeneous transformation matrices $\mathbf{H}_1$ and $\mathbf{H}_2$ can be obtained with `f_Rt2H` as described in the "Perspective Camera" section. Two feature points in $\mathbf{X}_w^1 = [0,\ 10,\ 10]^T$ and $\mathbf{X}_w^2 = [-10,\ 5,\ 5]^T$ are observed and projected in $\mathbf{m}_1^1$, $\mathbf{m}_1^2$ and $\mathbf{m}_2^1$, $\mathbf{m}_2^2$ onto the two image planes, respectively.

In EGT, the computation of epipoles and the fundamental matrix is obtained through the function

```
>> [e1,e2,F]=f_epipole(H1,H2,K1,K2);
```

while the epipolar lines are retrieved with

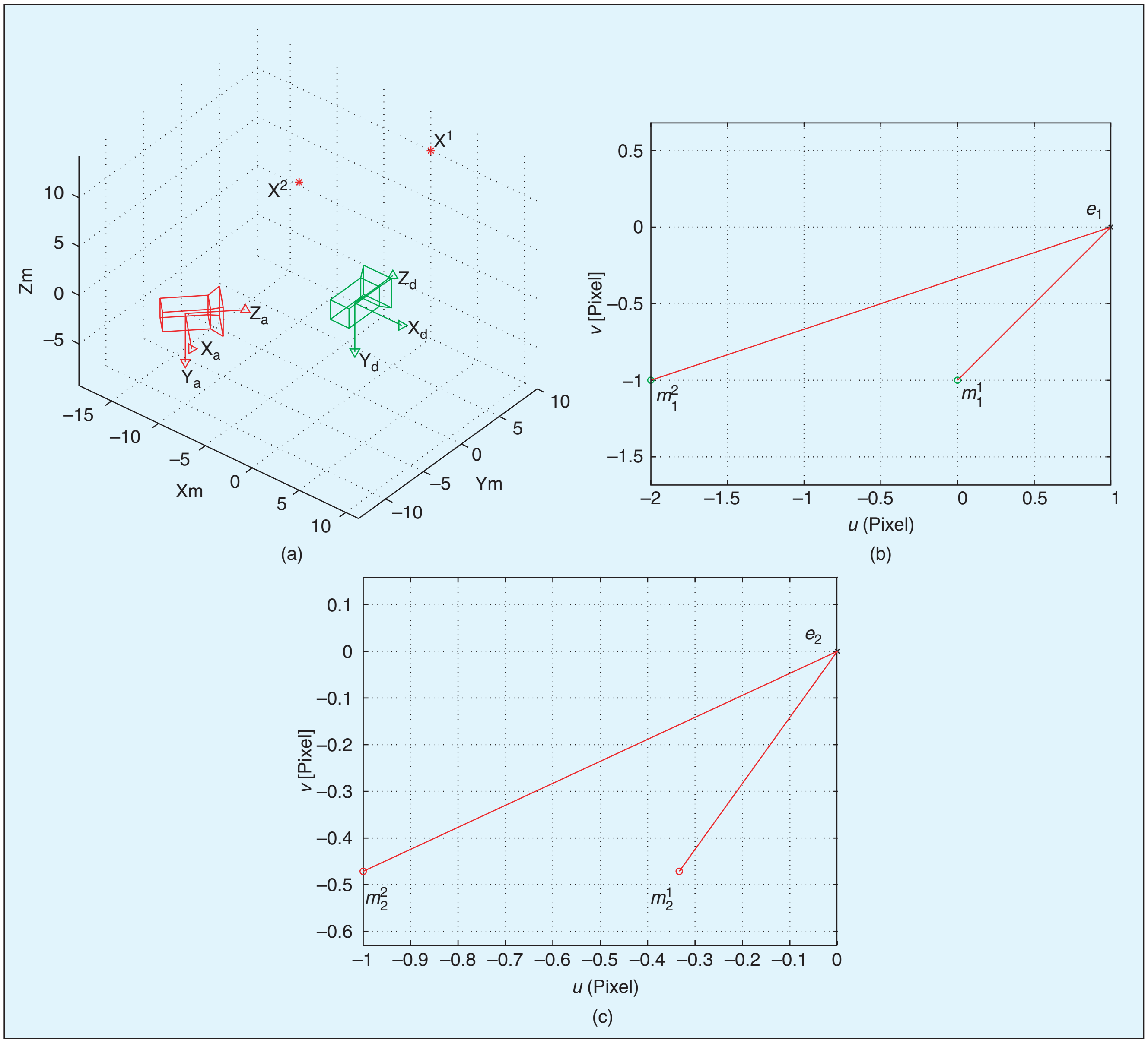

Figure 7. *Example 3. (a) 3-D simulation setup for epipolar geometry, with both cameras looking at two scene points* $\mathbf{X}_W^1$ *and* $\mathbf{X}_W^1$*. (b)-(c) Image planes of both pinhole cameras with corresponding features and epipoles. Note that all epipolar lines meet at the epipole and pass through the corresponding points.*

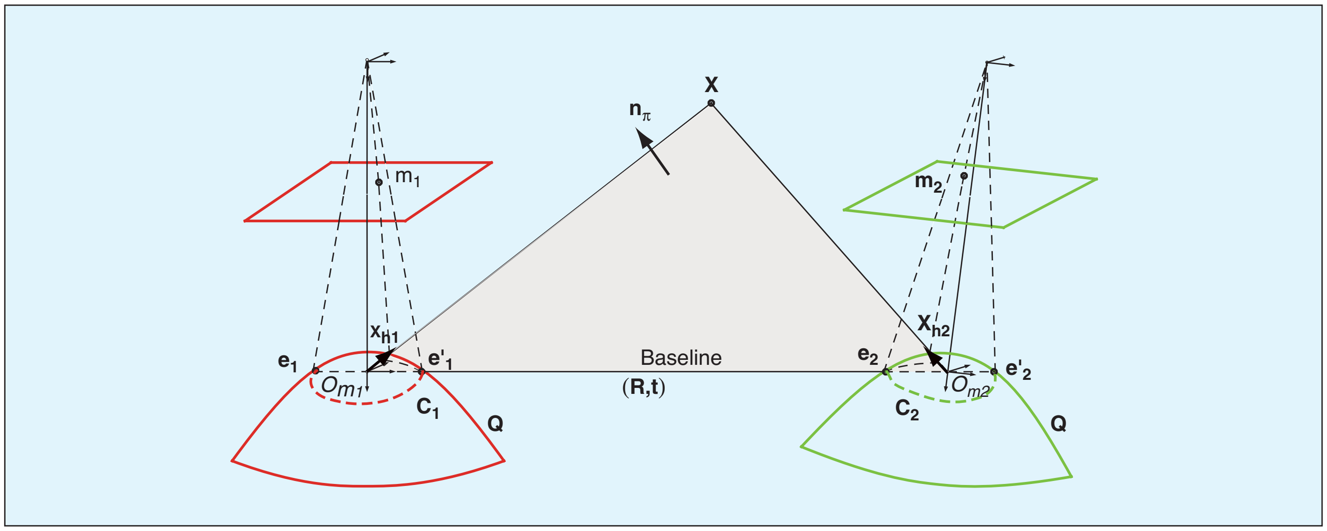

Figure 8. *Epipolar geometry for panoramic camera sensors.*

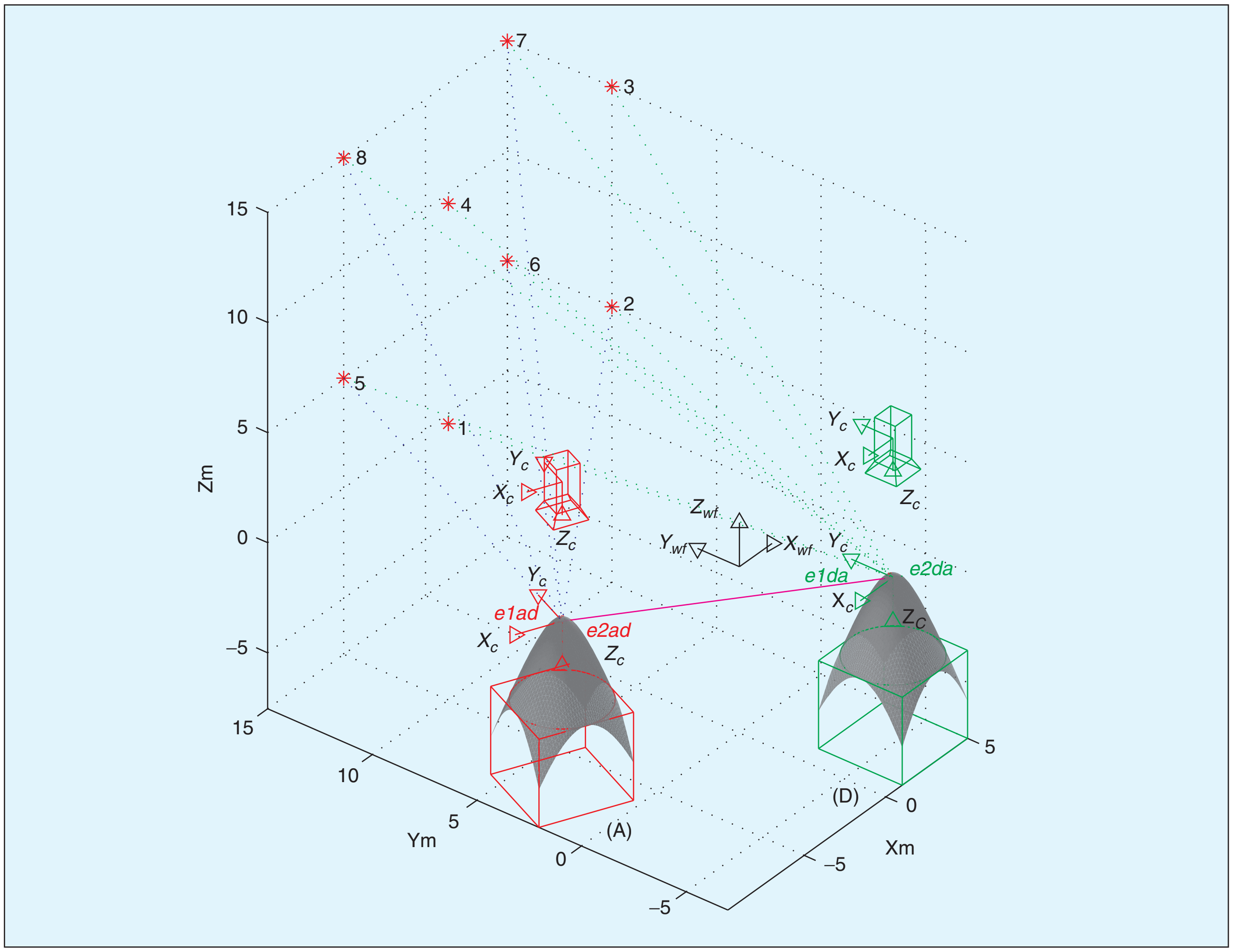

Figure 9. *Example 4: Central catadioptric cameras. Epipolar geometry computation and visualization in EGT.*

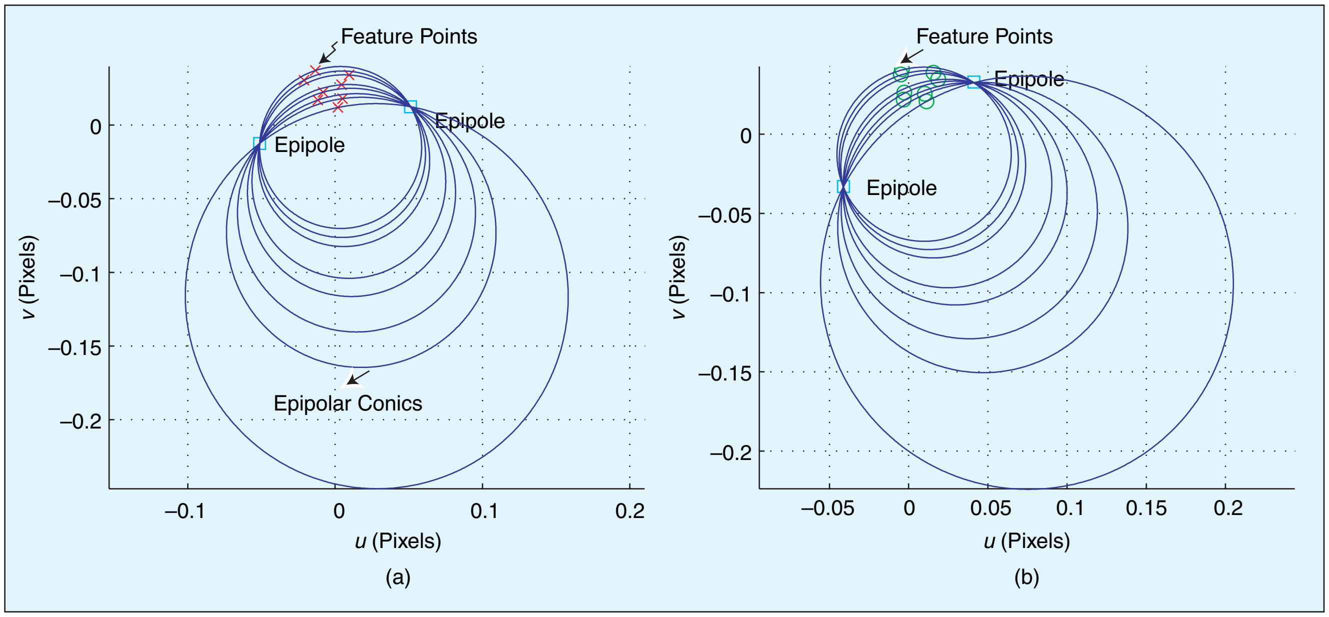

Figure 10. *Example 4: Epipolar conics plot in camera planes (a) and (b). All epipolar conics, in each image plane, intersect at the epipoles.*

```
>> [l1,l2]=f_epipline(m1,m2,F);
```

Note that $\mathbf{l}_1$ ($\mathbf{l}_2$) is a vector in $\mathbb{R}^3$ in homogeneous coordinates. In Figure 7(b), both image planes are represented together with feature points, epipoles, and the corresponding epipolar lines. In this case, for the sake of simplicity, it is assumed that $\mathbf{K} = \mathbf{I}$.

Epipolar Geometry for Catadioptric Cameras

As in the pinhole cameras, the epipolar geometry is here defined when a pair of views is available. Note, however, that in this case epipolar lines become epipolar conics. The reader is referred to [11] and [27] for an exhaustive treatment.

Let us now consider two catadioptric cameras (with equal hyperbolic mirror $\mathbf{Q}$) with foci O_{m1} and O_{m2}, respectively, as represented in Figure 8. Moreover, let $\mathbf{R}$ and $\mathbf{t}$ be the rotation and translation between the two mirror frames, respectively. Let $\mathbf{X}_{h1}$ and $\mathbf{X}_{h2}$ represent the mirror projections of the scene point $\mathbf{X}$ onto both hyperbolic mirrors. The coplanarity of $\mathbf{X}_{h1}$, $\mathbf{X}_{h2}$ and $\mathbf{X}$ can be expressed as

$$\mathbf{X}_{h2}^T \mathbf{E} \mathbf{X}_{h1} = 0 \qquad \text{where} \qquad \mathbf{E} = [\mathbf{t}]_\times \mathbf{R}. \qquad (7)$$

Table 1. The epipolar geometry estimation algorithms implemented in EGT (with bibliography).

ID	Estimation Algorithm
1	*Unconstrained linear estimation* [16]
2	*The normalized 8-point algorithm* [13]
3	*Algorithm based on geometric distance (Sampson)* [14]

Equivalent to the pinhole case, the 3×3 matrix $\mathbf{E}$ is referred to as the *essential matrix* [15].

Vectors $\mathbf{X}_{h1}$, $\mathbf{X}_{h2}$, and $\mathbf{t}$ define the epipolar plane with normal vector $\mathbf{n}_\pi$, which intersects both mirror quadrics $\mathbf{Q}$ in the two epipolar conics $\mathbf{C}_1$ and $\mathbf{C}_2$ [26]. For each 3-D point $\mathbf{X}^i$, a pair of epipolar conics $\mathbf{C}_1^i$ and $\mathbf{C}_2^i$ exist for the two views setup.

All the epipolar conics pass through two pairs of epipoles, $\mathbf{e}_1$ and $\mathbf{e}'_1$ for one mirror and $\mathbf{e}_2$ and $\mathbf{e}'_2$ for the second one, which are defined as the intersections of the two mirrors $\mathbf{Q}$ with the baseline $\overline{O_{m1}O_{m2}}$ (see Figure 8).

Example 4. Epipolar geometry for panoramic cameras

Camera and scene configurations similar to those in Example 3 have been set. Now we place eight feature points in the space. By using the EGT function

```
>> [e1c,e1pc,e2c,e2pc] = f_panepipoles(H1,H2);
```

the CCD coordinates of the epipoles are easily computed and plotted.

The CCD projections of all epipolar conics $\mathbf{C}_1^i$ and $\mathbf{C}_2^i$, where $i = 1, \ldots, 8$, corresponding to all 3-D points in $\mathbf{X}_w$ are computed and visualized by

```
>> [C1m,C2m] = f_epipconics(Xw,H1,H2);
```

Figure 9 presents the application of EGT to the computation of epipolar geometry between central catadioptric cameras (A) and (D). Note that the epipoles onto the mirror surface are also visualized.

Epipolar conics in both the CCD image plane, together with corresponding feature points, are shown in Figure 10(a) and (b). The complete MATLAB source code is available at the EGT Web site [21].

Estimation of Epipolar Geometry

Up to this point, the EGT functions we presented for the epipolar geometry computation were based on the knowledge of the relative position and orientation between the two cameras. However, many algorithms exist in the literature to estimate the epipolar geometry from the knowledge of corresponding feature points $\mathbf{m}_a^i$ and $\mathbf{m}_d^i$. A very important feature of EGT is that some of the most important algorithms for the epipolar geometry estimation have been embedded in the toolbox code. In particular, we implemented the linear method [16], the well-known Hartley's normalized eight-points algorithm [13], and the iterative algorithm based on geometric distance (Sampson) [14], as summarized in Table 1. A complete comparison and review of these methods has been reported in [14]. The epipolar geometry estimation is one of the most relevant features of EGT. The main reason is that users can run simulations where the fundamental matrix is estimated from noisy feature points.

The basic EGT function to estimate the fundamental matrix `F` is

```
>> F=f_Festim(U1,U2,algorithm);
```

where `U1` and `U2` are two $(2 \times N)$ matrices containing all N corresponding features

$$\mathbf{U}_1 = \begin{bmatrix} x_1^1 & x_1^2 & \dots & x_1^N \\ y_1^1 & y_1^2 & \dots & y_1^N \end{bmatrix}.$$

The integer `algorithm` identifies the type of estimation algorithm to be used between those available according to Table 1.

It is worth noting that these functions allow EGT to estimate the epipolar geometry in real experiments as well as in simulations.

Visual Servoing

When used together with a robotics toolbox, such as the Robotics Toolbox (RT) by Corke [7], EGT proves to be an efficient tool for implementing visual servoing techniques based also on multiple-view geometry.

The visual servoing controller proposed by Rives in [24] (for pinhole cameras) and that proposed in [19] (for central panoramic cameras) have been considered as tutorial examples to show the main EGT features in a visual servoing context.

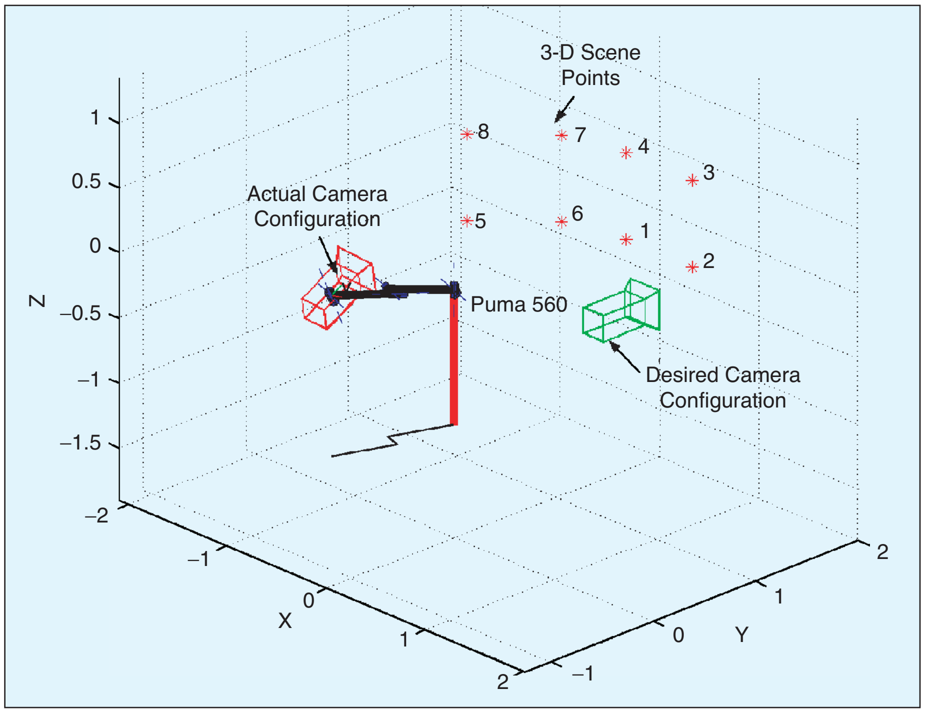

Figure 11. Simulation setup for EGT application to visual servoing. EGT is also compatible with Robotics Toolbox.

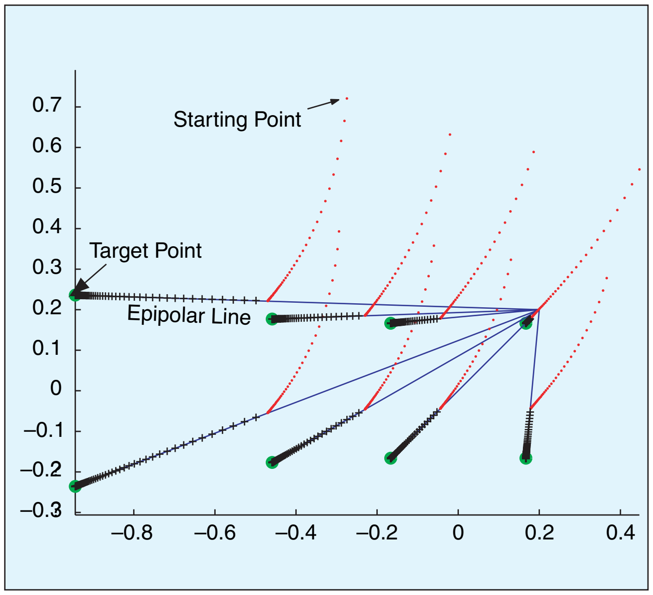

Figure 12. Migration of feature points from initial position toward desired one. Feature points, from a starting position, reach the epipolar lines at the end of the first task (priority task) and then move toward desired ones (crossed points) during the second task.

Visual Servoing with Pinhole Cameras for 6-DOF Robots

In what follows, the algorithm proposed in [24] is briefly recalled for the reader's convenience. The main goal of this visual servoing is to drive a 6-DOF robot arm (simulated with RT), equipped with a CCD camera (simulated with EGT), from a starting configuration toward a desired one using only image data provided during the robot motion (Figure 11). The basic visual servoing idea consists of decoupling the camera/end-effector rotations and translations through the use of the hybrid, vision-based task function approach [9]. The servoing task is performed, minimizing an error function φ that is divided into both a priority task φ_1, which rotates the actual camera until the desired orientation is achieved, and a secondary task φ_2 (to be minimized under the constraint $\varphi_1 = 0$), which has been chosen in [24] to be the camera distance to the target position.

The analytical expression of the error function is given by

$$\varphi = W^+ \varphi_1 + \beta(\mathrm{II}_6 - W^+ W)\frac{\partial \varphi_2}{\partial X}, \qquad (8)$$

where $\beta \in \mathbb{R}^+$; W^+ is the pseudoinverse of the matrix $W \in \mathbb{R}^{m \times n}$: $Range(W^T) = Range(J_1^T)$, where J_1 is the Jacobian matrix of task function $J_1 = \partial\varphi_1/\partial X$.

The priority task rotates the camera/end effector using a well-known epipolar geometry property stating that when both the initial and the desired cameras gain the same orientation, then all the distances between feature points $\mathbf{m}_i$ and corresponding epipolar lines $\mathbf{l}'_i$ are zero.

Figure 12 shows the simulation results of the algorithm previously proposed for a scene with eight points. The dotted points in Figure 12 show, in the image plane ($\mathbf{K} = \mathbf{I}$), the migration of the features to the epipolar lines during the rotation (first task). The crossed points correspond to the camera translation (second task).

Figure 13 shows the control inputs $(\mathbf{v}_c, \omega_c)$ and the plot of the norm of the error function e.

The MATLAB code of this simulation (`SIM_Rives00.m`) is included in the directory `demo_pinhole` of EGT [21].

Visual Servoing with Panoramic Cameras for Mobile Robot

While visual servoing with pinhole cameras has been deeply investigated in the literature, fewer results exist for omnidirectional cameras. Interesting contributions for mobile robots equipped with panoramic cameras can be found in [6], [8], and [31].

We here briefly recall the application of EGT to the design of a visual servoing technique for holonomic mobile robots equipped with a central panoramic camera presented in [19]. The proposed strategy does not assume any knowledge about the 3-D observed scene geometry and position with respect to the camera/robot.

Consider a holonomic mobile robot and let $q = (x, y, \theta)^T$ be its configuration vector with respect to the world frame $< Oxyz >_w$ (Figure 14). Let v_x, v_y be the translational velocities and ω the rotational velocity, respectively

$$\begin{aligned}\dot{x} &= v_x\\ \dot{y} &= v_y\\ \dot{\theta} &= \omega.\end{aligned} \qquad (9)$$

Suppose that a fixed catadioptric camera is mounted on the mobile robot such that its optical axis is perpendicular to the xy-plane. Without loosing generality, suppose that a target panoramic image, referred to as *desired*, has been previously acquired in the desired configuration $q_d = (0, 0, 0)^T$. Moreover, one more panoramic view, the *actual* one, is available at each time instant from the camera robot in the actual position.

As shown in the initial setup of Figure 15(a), the mobile robot disparity between the actual and the desired poses is characterized by rotation $R \in SO(2)$ and translation $t \in \mathbb{R}^2$. The control law will be able to drive the robot disparity between the actual and the desired configuration to zero using only visual information. In particular, we will regulate

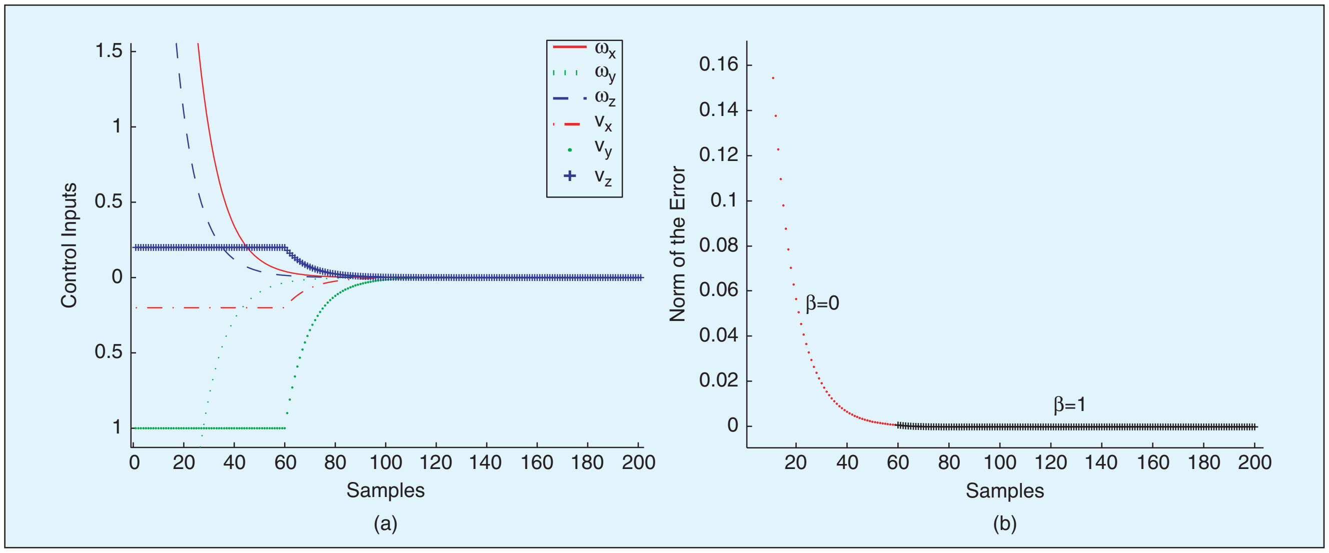

Figure 13. *(a) Control inputs to PUMA 560; (b) Norm of the error function e when $\beta = 0$ and when $\beta = 1$.*

separately the rotational disparity (first step) and the translational displacement (second step), shown in the upper center and upper right of Figure 15(b) and (c), respectively.

In the first step, the rotational disparity is compensated, exploiting the autoepipolar condition [19]. This novel property is based on the concept of bi-osculating conics (or biconics, for short), computed on the image plane from observed image corresponding features. It has been shown [19] that when all biconics intersect in only two points, then the rotational disparity between the actual and the desired view is compensated. Figure 15 (d) shows the image plane biconics when the two cameras are not in autoepipolar configuration. On the other hand, Figure 15(e) shows the image plane biconics when the cameras are in autoepipolar configuration and all biconics intersect in only two points.

Once the rotational disparity is compensated, then the second step is executed. This consists of a translational motion in which the robot is constrained to move along the baseline in order to approach the desired position [Figure 15(c)]. This translational control law can be designed on the image plane constraining all current features to lie on the epipolar conics and also constraining the distance between current and desired features $s(t)$ to decrease as the robot approaches the goal [Figure 15(f)].

Figure 16 gathers some simulation results for the first step. Figure 16(a) and (b) show, respectively, the actual and desired camera locations, together with scene points. In Figure 16(c), we can observe the feature points (crosses) moving from the initial position towards the epipolar conics. In Figure 16(d), it is shown that the angular velocity ω provided by the controller decreases to zero. Figure 17 shows simulation results for the second step. In Figure 17(a), the translational velocities $v_x(t)$ and $v_y(t)$ are reported. In Figure 17(b), the distance $s(t)$ between all feature points goes to zero, as reported in Figure 17(c), where crossed points (actual positions) go to the desired one (circles).

The MATLAB code for the autoepipolar property is the `ex5_autoepipolar.m` and is included in the directory `demo_panoramic` of EGT.

Comparison with Other Packages

To the best of our knowledge, no other software packages for the MATLAB environment exist that deal with the creation of multicamera scenarios for both pinhole and panoramic cameras.

An interesting MATLAB package is the Structure and Motion Toolkit by Torr [30]. This toolkit is specifically designed for features detection and matching, robust estimation of the fundamental matrix, self-calibration, and recovery of the projection matrices. The main difference between EGT and the Structure and Motion Toolkit is that the work of Torr is specifically oriented to pinhole cameras and is not designed to run simulations.

Regarding the visual servoing, it is worthwhile to mention the interesting Visual Servoing Toolbox (VST) by Cervera et al. [2], which provides a set of Simulink blocks for the simulation of vision-controlled systems. The main difference is that the VST implements some well-known visual servoing control laws but does not consider either multiple-view geometry nor panoramic cameras, which are the distinguishing features of EGT.

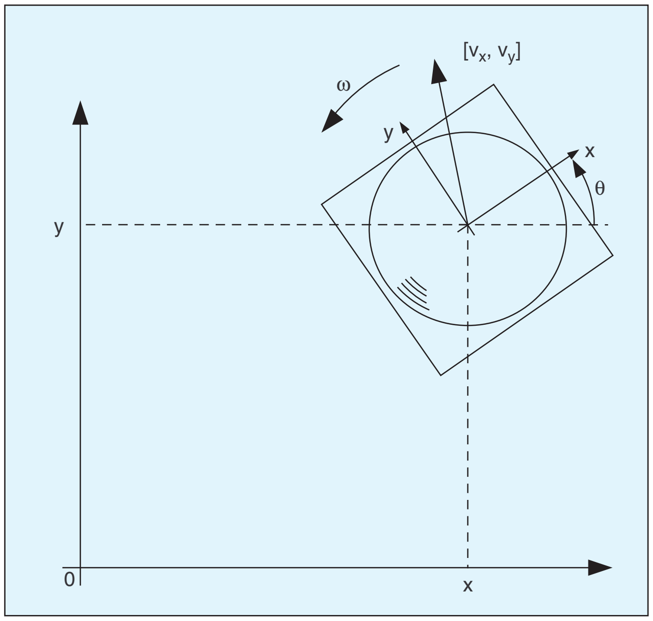

Figure 14. *The holonomic mobile robot with a fixed catadioptric camera moving on a plane.*

Conclusions

EGT for MATLAB is a software package targeted to research and education in computer vision and robotics visual servoing. It provides the user with a wide set of functions for designing multicamera systems for both pinhole and panoramic cameras. Several epipolar geometry estimation algorithms have been implemented. EGT is freely available and can be downloaded at the EGT Web site along with a detailed manual and code examples.

Acknowledgments

The authors would like to thank Eleonora Alunno for her help in developing EGT functions during her master's thesis work, and Jacopo Piazzi for his useful comments. EGT has been widely tested by the students of the Robotics and Vision classes and laboratories at the University of Siena, who contributed to the testing phase of EGT. Special thanks go to Paola Garosi, Annalisa Bardelli, and Fabio Morbidi, who helped us with useful comments on the toolbox and on the article.

Keywords

Epipolar geometry, pinhole cameras, central catadioptric cameras, visual servoing, MATLAB Toolbox

References

[1] Intel's OpenCV [Online]. Available: http://sourceforge.net/projects/opencvlibrary/

[2] *The Visual Servoing Toolbox*, 2003 [Online]. Available: http:// vstoolbox.sourceforge.net/

[3] S. Baker and S.K. Nayar, "Catadioptric image formation," in *Proc.*

DARPA Image Understanding Workshop, 1997, pp. 1431–1437.

[4] R. Basri, E. Rivlin, and I. Shimshoni, "Visual homing: Surfing on the epipoles," in *Proc. International Conf. Computer Vision*, 1998, pp. 863–869.

[5] R. Benosman and S.B. Kang, *Panoramic Vision: Sensors, Theory and Applications*. New York: Springer-Verlag, 2001.

[6] D. Burschka and G. Hager, "Vision-based control of mobile robots," in *Proc. 2001 IEEE Int. Conf. Robotics and Automation*, 2001, pp. 1707–1713.

[7] P.I. Corke, "A robotics toolbox for MATLAB," *IEEE Robot. Automat. Mag.*, vol. 3, no. 1, pp. 24–32, 1996.

[8] N. Cowan, O. Shakernia, R. Vidal, and S. Sastry, "Vision-based follow the leader," in *Proc. 2003 IEEE/RSJ Int. Conf. Intelligent Robots and Systems*, 2003, vol. 2, pp. 1796–1801.

[9] B. Espiau, F. Chaumette, and P. Rives, "A new approach to visual servoing in robotics," *IEEE Trans. Robot. Automat.*, vol. 8, no. 3, pp. 313–326, 1992.

[10] Free Software Foundation, The free software definition [Online]. Available: http://www. fsf.org/philosophy/free-sw.html

[11] C. Geyer and K. Daniilidis, "A unifying theory for central panoramic systems," in *Proc. 6th European Conf. Computer Vision*, 2000, pp. 445–462.

[12] C. Geyer and K. Daniilidis, "Mirrors in motion: Epipolar geometry and motion estimation," in *Proc. 9th IEEE Int.Conf. Computer Vision*, 2003, vol. 2, pp. 766–773.

[13] R. Hartley, "In defence of the 8-point algorithm," in *Proc. IEEE Int. Conf. Computer Vision*, 1995, pp. 1064–1070.

[14] R. Hartley and A. Zisserman, *Multiple View Geometry in Computer Vision*. Cambridge, U.K.: Cambridge Univ. Press, Sep. 2000.

[15] H.C. Longuet-Higgins, "A computer algorithm for recostructing a scene from two projections," *Nature*, vol. 293, pp. 133–135, 1981.

[16] Q.T. Luong and O.D. Faugeras, "The fundamental matrix: theory, Algorithms and stability analysis," *Int. J. Comput. Vision*, vol. 17, no. 1, pp. 43–76, 1996.

[17] Y. Ma, S. Soatto, J. Košecká, and S. Shankar Sastry, *An invitation to 3-D Vision, From Images to Geometric Models*. New York: Springer-Verlag, 2003.

[18] E. Malis and F. Chaumette, "2 1/2 D visual servoing with respect to unknown objects through a new estimation scheme of camera displacement," *Int. J. Comput. Vision*, vol. 37, no. 1, pp. 79–97, 2000.

[19] G.L. Mariottini, E. Alunno, J. Piazzi, and D. Prattichizzo, "Epipole-based visual servoing with central catadioptric camera," in *Proc. 2005 IEEE Int.Conf. Robotics and Automation*, 2005. (in press).

[20] G.L. Mariottini, G. Oriolo, and D. Prattichizzo, "Epipole-based visual servoing for nonholonomic mobile robots," in *Proc. 2004 IEEE Int. Conf. Robotics and Automation*, 2004, pp. 497–503.

[21] G.L. Mariottini and D. Prattichizzo, *The Epipolar Geometry Toolbox for Matlab,* University of Siena, Italy [Online]. Available: http://egt.dii.unisi.it

[22] D. Nister, "An efficient solution to the five-point relative pose problem," *IEEE Trans. Pattern Anal. Machine Intell.*, vol. 26, no. 6, pp. 756–770, 2004.

[23] D. Nister, O. Naroditsky, and J. Bergen, "Visual odometry," in *Proc.*

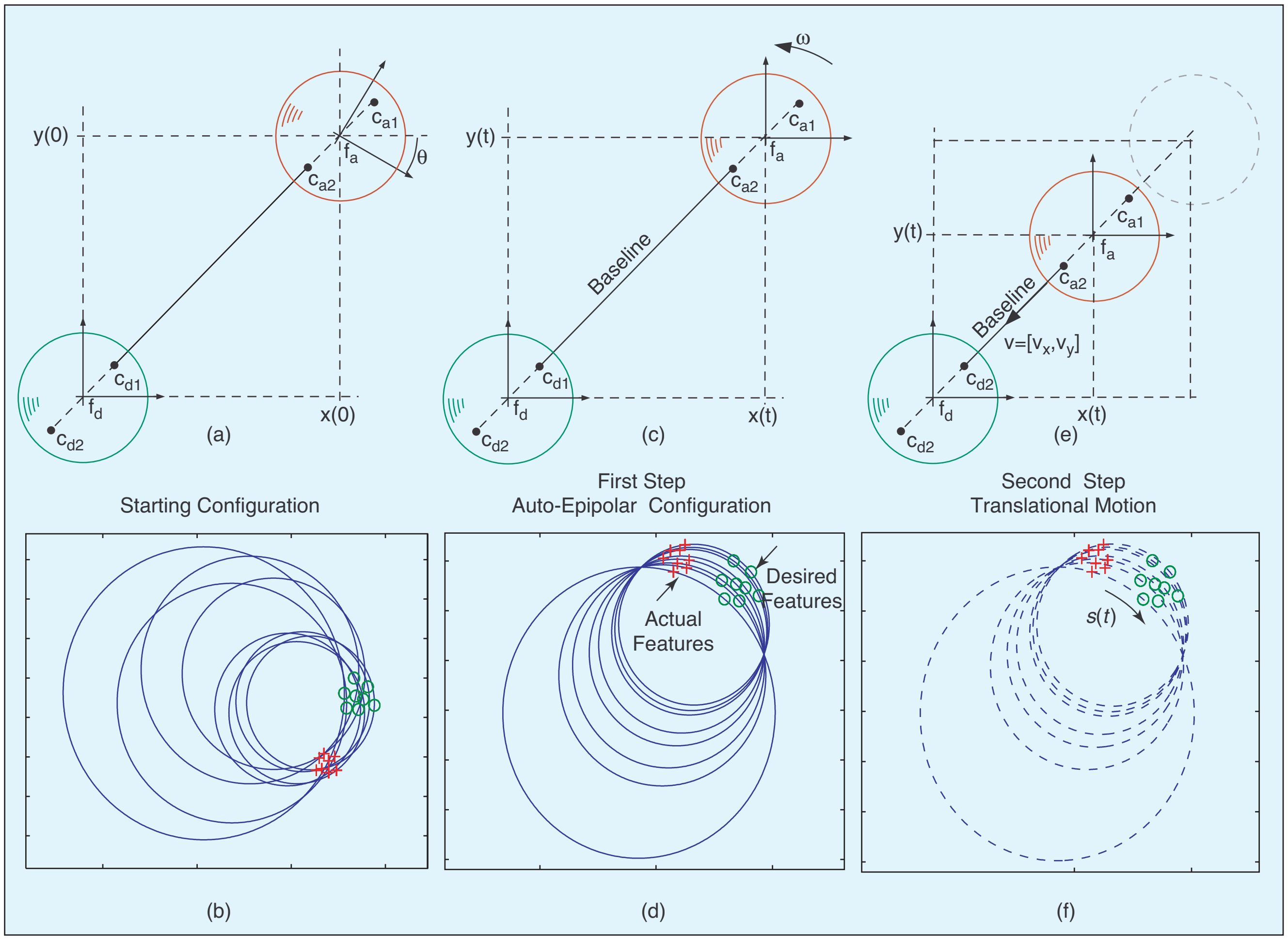

Figure 15. *The visual servoing control law for panoramic cameras drives the holonomic camera robot from (a) the starting configuration centered at f_a [(b) no common intersection between biconics exists], (c) through an intermediate configuration when both views have the same orientation (d) (biconic common intersection), (e) towards the position f_d [(f) until $s(t) = 0$]. Images (b), (d), and (f) have been obtained with EGT.*

2004 IEEE Conf. Computer Vision and Pattern Recognition, 2004, vol. 1, pp. 652–659.

[24] P. Rives, "Visual servoing based on epipolar geometry," in *Proc. Int. Conf. Intelligent Robots and Systems*, 2000, vol. 1, pp. 602–607.

[25] M.W. Spong and M. Vidyasagar, *Robot Dynamics and Control*. New York: Wiley, 1989.

[26] T. Svoboda, "Central panoramic cameras design, geometry, egomotion," Ph.D. thesis, Center for Machine Perception, Czech Technical Univ., 1999.

[27] T. Svoboda and T. Pajdla, "Epipolar geometry for central catadioptric cameras," *Int. J. Comput. Vision*, vol. 49, no. 1, pp. 23–37, 2002.

[28] T. Svoboda, T. Pajdla, and V. Hlaváč, "Motion estimation using central panoramic cameras," in *Proc. IEEE Conf. Intelligent Vehicles*, 1998, pp. 335–340.

[29] The MathWorks Inc., MATLAB [Online]. Available: http://www.mathworks.com

[30] Philip Torr, The structure and motion toolkit for MATLAB [Online]. Available: http://wwwcms.brookes.ac.uk/~philiptorr/

[31] K. Usher, P. Ridley, and P. Corke, "Visual servoing for a car-like vehicle—An application of omnidirectional vision," in *Proc. 2003 IEEE Int. Conf. Robotics and Automation*, 2003, pp. 4288–4293.

[32] R. Vidal, O. Shakernia, and S. Sastry, "Omnidirectional vision-based formation control," in *Proc. Annual Allerton Conf. Communication, Control and Computing*, 2002, pp. 1637–1646.

Gian Luca Mariottini received the Laurea degree (cum laude) in information engineering from the University of Siena, Italy, in 2002. From October 2004 to April 2005, he was a visiting student at the GRASP Lab, University of Pennsylvania, USA. He is currently a Ph.D. degree candidate at the University of Siena and a member of the control and system group at the Department of Information Engineering (Siena). His main research interests are in the area of vision-guided navigation of multiple robots, computer vision, visual servoing, panoramic cameras, and active vision for localization.

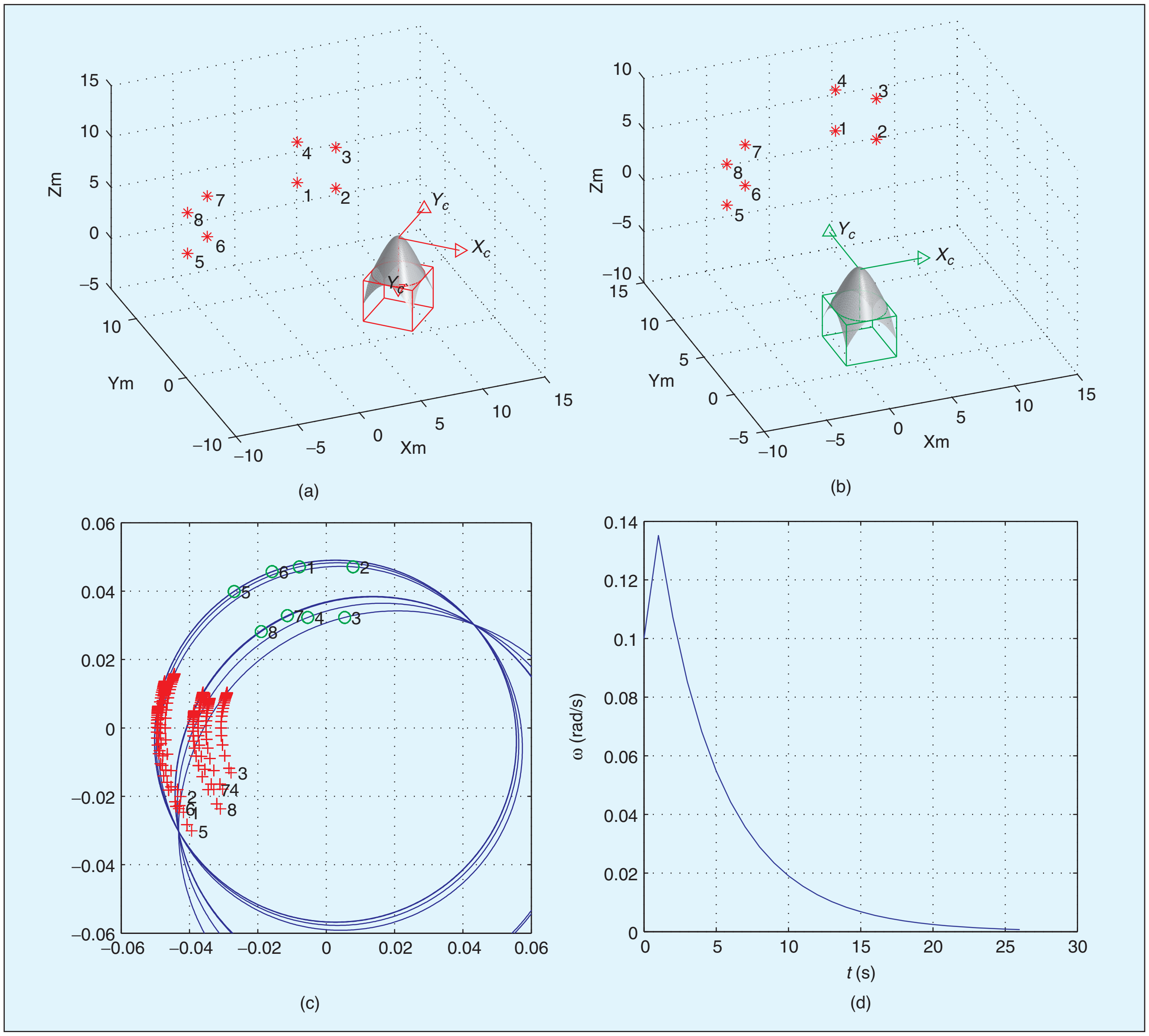

Figure 16. *Simulation results with EGT for the first step. The robot, (a) placed in the actual configuration, (b) rotates until the orientation matches the desired configuration. This leads all feature in migrating to (c) the epipolar conics and (d) the angular velocity to reach zero at the end of the first step, when all biconics intersect.*

Domenico Prattichizzo received his M.S. degree in engineering in 1991. He received the Ph.D. degree in robotics and automation from Pisa University in 1995. He has been an associate professor at Siena University since 2002. In 1994, he was a visiting scientist at the Massachusetts Institute of Technology Artifical Intelligence Lab. He has been a research associate in the Robotics and Automation group at the Centro "E. Piaggio" of Pisa University since 1992 and a research associate in the Center for Complex Systems Studies of Siena University since 2000. He has been a member of the Editorial Board of *IEEE Transactions on Robotics*, *IEEE Transactions on Control Systems Technologies* (since 2003), and the *Journal of Dynamics of Continuous, Discrete and Impulsive Systems (DCDIS) Series B: Application and Algorithms* (since 2001). He has been a member of the Editorial Board and the Conference Editorial Board of the IEEE Control System Society since 2001. He was a member of the Program Committee of IEEE/RSJ 2003 International Conference on Intelligent Robots and Systems 2003. He has functioned as coorganizer and chair of many sessions on robotics and control theory for international conferences. He has coauthored more than 130 papers in the area of robotics and automation.

Address for Correspondence: Gian Luca Mariottini, Dipartimento di Ingegneria dell'Informazione, Università di Siena, Via Roma 56, 53100 Siena, Italy. E-mail: gmariottini@dii.unisi.it. Fax: +39 0577 233602.

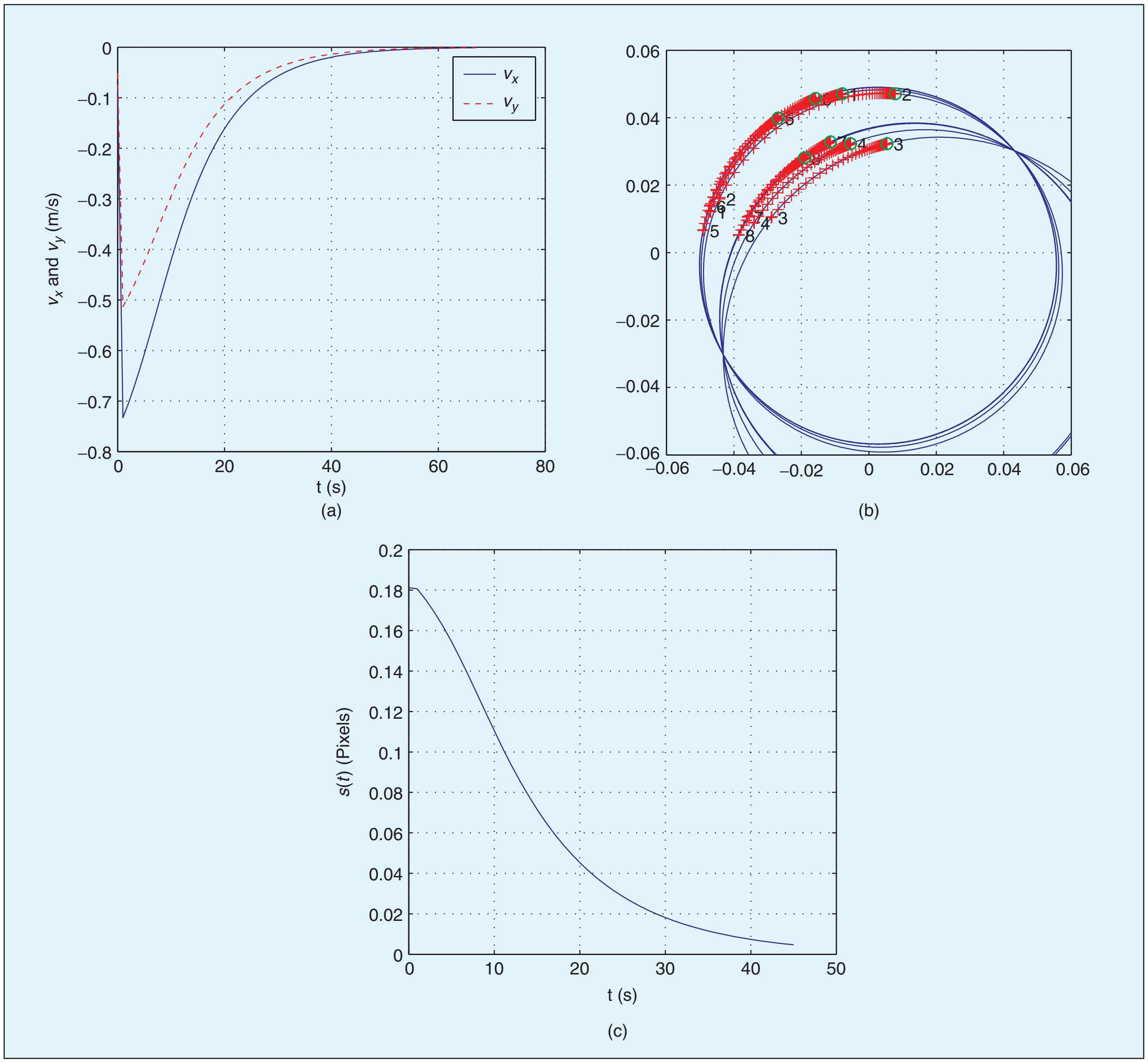

Figure 17. *Simulation results with EGT for the second step: (a) the translational controller inputs guarantee the motion of the robot along the baseline (the feature slide along the biconic); (b) then it stops when the distance $s(t)$ between corresponding features is equal to zero, i.e., (c) the robot is in the desired pose.*