The Epipolar Geometry Toolbox (EGT) for MATLAB

Gian Luca Mariottini, Eleonora Alunno, Domenico Prattichizzo
Dipartimento di Ingegneria dell'Informazione
Università di Siena
Via Roma 56, 53100 Siena, Italy
Email: {gmariottini,alunno,prattichizzo}@dii.unisi.it

Technical Report 07-21-3-DII University of Siena, July 2004 Siena, Italy.

***Abstract*— The Epipolar Geometry Toolbox (EGT) was realized to provide a MATLAB user with an extensible framework for the creation and visualization of multiple camera systems and easy manipulation of the geometry relating them. Functions provided, for both pin-hole and panoramic vision sensors, include camera placement and visualization, computation and estimation of epipolar geometry entities and many others. EGT has been used with the Robotics Toolbox [7] for Visual Servoing applications. This article introduces the Toolbox in tutorial form. Examples are provided to show its capabilities. The complete Toolbox and the detailed documentation are freely available on the EGT web site [18].**

I. Introduction

The Epipolar Geometry Toolbox (EGT) is a toolbox designed for MATLAB [25]. MATLAB is a software environment, available for a wide range of platforms, designed around linear algebra principles and graphical presentations also for large datasets. Its core functionalities are extended by the use of many additional toolboxes. Combined with interactive MATLAB environment and advanced graphical functions, EGT provides a wide set of functions to approach computer vision problems especially with multiple views. When used with the Robotics Toolbox by Corke [7], EGT allows to design Visual Servoing simulations.

The increasing interest in robotic visual servoing for both 6DOF kinematic chains and mobile robots equipped with pin-hole or panoramic cameras fixed to the workspace or to the robot, motivated the development of EGT.

Several authors, such as [4], [8], [17], [19], [20], have recently proposed new visual servoing strategies based on the geometry relating multiple views acquired from different camera configurations, i.e. the Epipolar Geometry [13].

In these years we have observed the necessity to develop a software environment that could help researchers to 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 but also completely customizable functions in order to create general multi-camera scenarios and manipulate the visual information between them. A distinguishable remark of EGT is that it can be used to create and manipulate visual data provided by both pin-hole and panoramic cameras. Catadioptric cameras, due to their wide field of view, has been recently applied in visual servoing context [27].

The second motivation lead to the development of EGT was the increasing distribution of "free" software in the latest years, on the basis of the Free Software Foundation [9] principles. In this way users are allowed, and also encouraged, to adapt and improve the program as dictated by their need. Examples of programs that follow these principles include for instance the Robotics Toolbox [7], for the creation of simulations in robotics, and the Intel's OpenCV C++ libraries for the implementation of computer vision algorithms, such as image processing and object recognition [1].

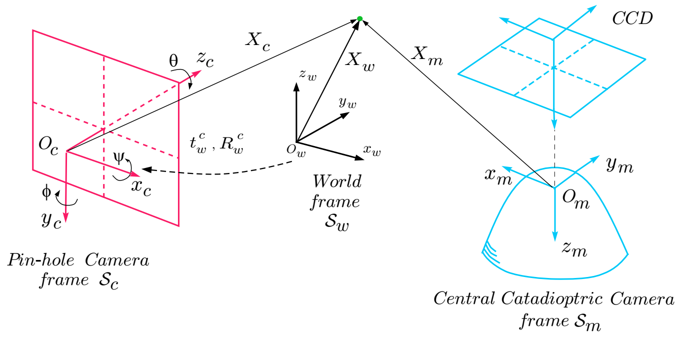
The third important motivation to design of EGT was the availability and increasing sophistication of MATLAB. EGT could have been written in other languages, such as C, C++ and this would have freed it from dependency on other softwares. However these low-level languages are not so conducive to rapid 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. An interesting example is provided at the end of this paper to illustrate the use of EGT in combination with the Robotics Toolbox to tackle involved tasks in visual servoing applications.

Section 2 presents the basic vector notation in EGT, while in Section 3 the pin-hole and omnidirectional camera models together with EGT basic functions are presented. In Section 4 we present the setup for multiple camera geometry (Epipolar Geometry) while in Section 5 an application of EGT to visual servoing is presented together with simulation results. In Section 6 we make a comparison between EGT and other software packages. EGT can be freely downloaded at [18] and requires MATLAB 6.5 or upper. The detailed manual is provided with a large set of examples, figures and source code also for beginners.

II. Basic vector notation

We here present the basic vector notation adopted in Epipolar Geometry Toolbox. In EGT all scene points $\mathbf{X}_w \in \mathbb{R}^3$ are expressed in the *world frame* $\mathcal{S}_w =< O_w x_w y_w z_w >$ (Fig.1). When referred to the *pin-hole camera* frame



CCD
X_c
X_w
X_m
θ
z_c
z_w
y_w
x_w
O_w
O_c
t_w^c, R_w^c
ψ
x_c
ϕ
y_c
World frame $\mathcal{S}_w$
Pin-hole Camera frame $\mathcal{S}_c$
x_m
y_m
O_m
z_m
Central Catadioptric Camera frame $\mathcal{S}_m$

Fig. 1. Main reference frames notation and vector representation in EGT.

$\mathcal{S}_c =< O_c x_c y_c z_c >$ they will be indicated with $\mathbf{X}_c$. Moreover all scene points expressed w.r.t. a *central catadioptric camera* frame $\mathcal{S}_m =< O_m x_m y_m z_m >$ will be indicated with $\mathbf{X}_m$. For the reader convenience we briefly present the basic vector notation and transformation [21]. Refer to Fig. 1 and consider the 3×1 vector $\mathbf{X}_c \in \mathcal{S}_c$. It can be expressed in $\mathcal{S}_w$ as follows:

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

where $\mathbf{t}_w^c$ is the translational vector centered in $\mathcal{S}_w$ and pointing toward the $\mathcal{S}_c$ frame (Fig. 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 $\mathbf{R}_w^c = \mathbf{R}_{roll,pitch,yaw} = \mathbf{R}_{z,\theta}\mathbf{R}_{y,\phi}\mathbf{R}_{x,\psi}$. 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:

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

In analogous way 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) it results that:

$$\begin{aligned} \mathbf{X}_w &= \mathbf{R}_w^d \mathbf{X}_d + \mathbf{t}_w^d & (3) \\ \mathbf{X}_w &= \mathbf{R}_w^a \mathbf{X}_a + \mathbf{t}_w^a & (4) \end{aligned}$$

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.

III. Pin-hole and omnidirectional camera models

EGT provides easy-to-use functions for the placement of pin-hole and central catadioptric (or omnidirectional) cameras. Their imaging model has been here implemented in order to being the user able to manipulate the visual information. In this section the fundamentals of perspective and omnidirectional camera models are quickly reviewed. The reader is referred to [13], [16], [5] for a detailed treatment. According to the purposes of this tutorial, some basic EGT code examples are reported together with the theory.

A. Perspective camera

With reference to Fig. 2, consider a pin-hole camera located at O_c. The full perspective model describes the relationship between a 3D point (in homogeneous coordinates) $\widetilde{\mathbf{X}}_w = \begin{bmatrix} X & Y & Z & 1 \end{bmatrix}^T$ expressed in the world frame and its projection $\widetilde{\mathbf{m}} = \begin{bmatrix} u & v & 1 \end{bmatrix}^T$ onto the image plane according to

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

where $\mathbf{K} \in \mathbb{R}^{3\times3}$ is the camera *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) represent the pixels 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 and γ is the orthogonality factor of the CCD image axes (*skew-factor*).

$\mathbf{\Pi} = [\mathbf{R}\,|\,\mathbf{t}] \in \mathbb{R}^{3\times4}$ is the so-called *external parameters* camera matrix, that contains the rotation $\mathbf{R}$ and the translation $\mathbf{t}$ between the world and the camera frames. According to the most part of the literature, the optical axis z_c of pin-hole cameras has been chosen parallel to the y_w axis of $\mathcal{S}_w$. We then define:

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

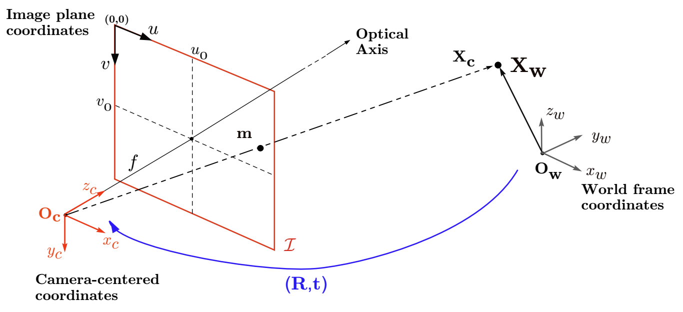


Fig. 2. The pin-hole camera model. The 3D point $\mathbf{X}_w$ is projected onto $\mathbf{m}$ through the optical center O_c. Note that $\mathbf{m}$ is expressed in the image plane coordinates (u, v) (pixels).

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 `t` of a pin-hole camera is specified with respect to the *world frame* while the rotation $\mathbf{R}$ is referred to the *pin-hole camera frame* axes. During the testing phase at the University of Siena this choice seemed to be appreciated from students that addressed it as intuitive.

Example 1 (3D scene and pin-hole camera placement): Consider now a pin-hole camera rotated by `R` $= \mathbf{R}_{y,\pi/4} \in \mathbb{R}^{3\times 3}$ 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 3D 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 pin-hole camera
>> axis equal, grid on, view(12,34)
>> title('3D setup - EGT Tutorial - Ex.1')
```

Plot of 3D view is represented in Fig. 3(a) Here presented functions have additional options. See the EGT Manual [18] for details.

We can also place a set of N 3D 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_perspproj(Xw,H,K)`:

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

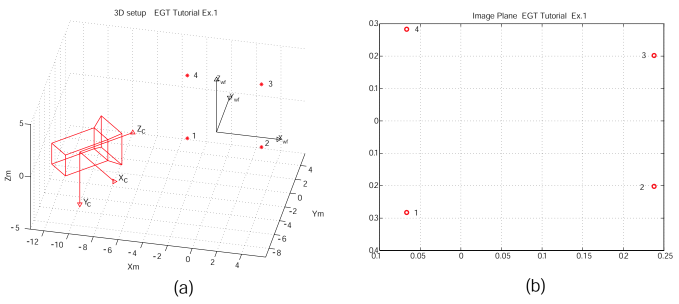


Fig. 3. Example 1. A pin-hole camera is positioned in $t = [-10, -5, 0]$ in the 3D world frame and rotated by $\pi/4$ around the y-axis. The 3D scene points are projected onto the image plane. Note that in this case $\mathbf{K} = \mathbf{I}$ for simplicity.

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

Options of the proposed functions are detailed in the *EGT Manual* [18].

$\square$

Note that while the above example describes the placement of 3D points $\mathbf{X}_w$, EGT is also able to build scenes with 3D objects returning surface points and normals (see function `f_3Dsurface` in [18]).

B. *Omnidirectional Camera Model*

Omnidirectional cameras combine reflective surfaces (mirrors) and lenses. Several types of panoramic cameras can be obtained simply combining cameras (pin-hole or orthographic) and mirrors (hyperbolic, parabolic or elliptical) [5].

Panoramic cameras are classified according to the fact that they satisfy or not the single viewpoint constraint guaranteeing that the visual sensor measures the light through a single point. Note that this constraint is required for the existence of epipolar geometry and for the generation of geometrically correct images [24] [11].

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 pin-hole cameras coupled with hyperbolic mirrors, and orthographic cameras coupled with parabolic mirrors.

In what follows the imaging model for a pin-hole camera with hyperbolic mirror is described.

Consider now the basic scheme in Fig. 4. Note that in this case all frames (for both the camera and the mirror) are aligned with the EGT 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 whose vector is $\mathbf{X}_c$.

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

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

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

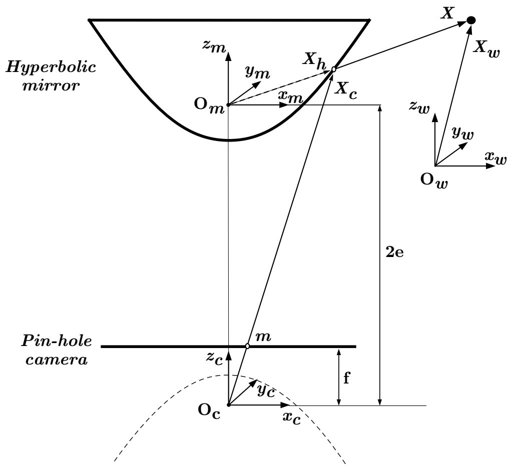


Fig. 4. The panoramic camera model (pin-hole camera and hyperbolic mirror). The 3D point $\mathbf{X}_w$ is projected at $\mathbf{m}$ through the optical center O_c, after being projected through the mirror center O_m.

with eccentricity $e = \sqrt{a^2 + b^2}$, the transformation to obtain the projection $\mathbf{u}$ in the pin-hole camera frame (see Fig. 4) is given by

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

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

In EGT a central catadioptric camera can be placed only specifying the homogeneous transformation matrix between mirror and world frame

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

Example 2 (Panoramic camera placement): In EGT a panoramic camera can be placed and visualized. Let us place the camera in `t=[-5,-5,0]'` with orientation R$\equiv R_{z,\pi/4}$. EGT provides a function to simply visualize the panoramic camera in the 3D world frame as in Fig. 5.

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

Moreover, for assigned camera calibration matrix $\mathbf{K}$:

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

the projection of a 3D 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 =
4.104890614533404e+002
2.400000000000000e+002
1.000000000000000e+000

Xh =
6.031787983781853e-001
3.788185930286120e-017
3.412094548809345e-001
1.000000000000000e+000
```

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

□

IV. Epipolar Geometry

In this section the EGT functions dealing with the Epipolar Geometry are presented.

A. Epipolar geometry for pin-hole cameras

Consider now two perspective views of the same scene taken from distinct viewpoints in O_1 and O_2, as in Fig.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 Fig.6 we define the following set of geometric entities [13], [16]:

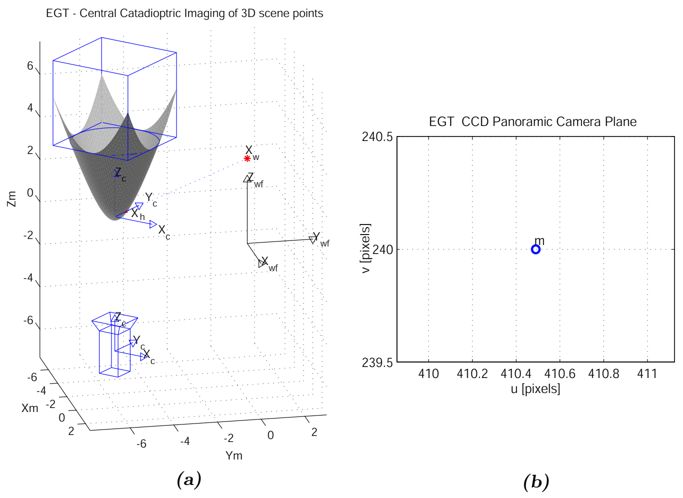


Fig. 5. Example 2. (a) A panoramic camera is positioned in $[-5,-5,0]^T$ in the 3D world frame and (b) the 3D point $\mathbf{X}_w = [0,0,4]^T$ is projected to the pinhole camera after being projected in $\mathbf{X}_h$.

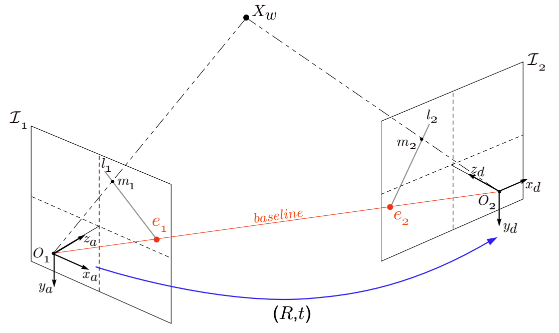


Fig. 6. Basic Epipolar Geometry entities for pinhole cameras.

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 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 onto the image plane of the other camera frame is called *epipole*. The epipole will be expressed in homogeneous coordinates

$$\tilde{\mathbf{e}}_1 = \begin{bmatrix} e_{1x} & e_{1y} & 1 \end{bmatrix}^T \qquad \tilde{\mathbf{e}}_2 = \begin{bmatrix} e_{2x} & e_{2y} & 1 \end{bmatrix}^T$$

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

One of the main parameters of the epipolar geometry is the 3×3 Fundamental Matrix $\mathbf{F}$ that 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**)

Let be given two views of the same 3D 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 $\mathbf{m}_2$ be the two corresponding projections of $\mathbf{X}_w$ in both image planes. The epipolar 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 to $\mathbf{t}$. □

Some important properties of epipolar geometry are reviewed extensively in [13] [16].

Example 3 (Epipolar Geometry for pinhole cameras): The computation of epipolar geometry with EGT is straightforward. First of all suppose the first camera placed at $\mathbf{t}_1 = [-10,\ -10,\ 0]^T$ with orientation $\mathbf{R}_1 = \mathbf{R}_{y,\pi/4}$, while the second one is in the world frame, i.e. $\mathbf{t}_2 = [0,\ 0,\ 0]^T$ with orientation $\mathbf{R}_2 = \mathbf{I}$ (Fig. 7(a)). Pin-hole cameras can be designed and their homogeneous transformation matrices $\mathbf{H}_1$ and $\mathbf{H}_2$ can be obtained with `f_Rt2H` as described in Sec. III-A. 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. The two views setup is reported in Fig. 7. The computation of epipoles and fundamental matrix is obtained from:

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

while the epipolar lines are retrieved with:

```
>> [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 (Fig. 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 $\mathbf{K} = \mathbf{I}$.

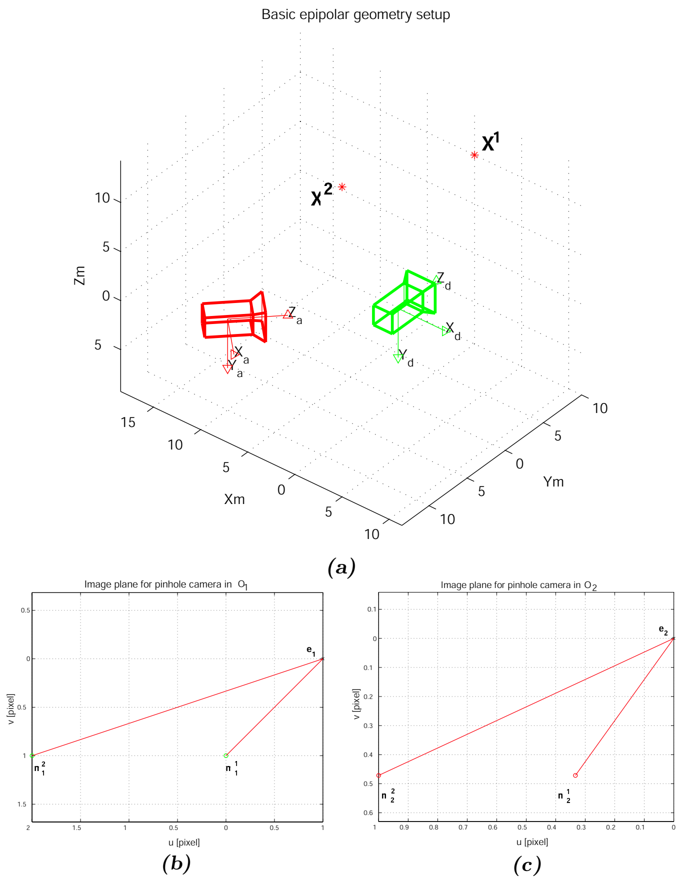


Fig. 7. Example 3. (a) 3D simulation setup for epipolar geometry, with both cameras looking at two scene points $\mathbf{X}_w^1$ and $\mathbf{X}_w^1$; (b)-(c) image planes of both pin-hole cameras with corresponding features and epipoles. Note that all epipolar lines meet at the epipole and pass through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

B. *Epipolar Geometry for panoramic cameras*

As in the pinhole cameras, the epipolar geometry is here defined when a pair of panoramic views is available. Note however that in this case epipolar lines become epipolar conics. The reader is referred to [10], [23] for an exhaustive treatment.

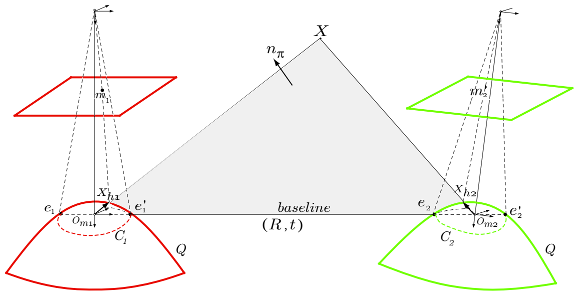


Fig. 8. Epipolar geometry for panoramic camera sensors.

Let now consider two panoramic cameras (with equal hyperbolical mirror $\mathbf{Q}$) with foci O_{m1} and O_{m2} respectively, as represented in Fig. 8. Moreover, let define the rotation $\mathbf{R}$ and the translation $\mathbf{t}$ between the two mirror frames. Let $\mathbf{X}_{h1}$ and $\mathbf{X}_{h2}$ represent the mirror projections of the scene point $\mathbf{X}$ onto both hyperbolas $\mathbf{Q}$.

The coplanarity of $\mathbf{X}_{h1}$, $\mathbf{X}_{h2}$ and $\mathbf{X}$ can be expressed, in the reference coordinate system centered at M_2, as:

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

Equivalently to the pin-hole case, the 3×3 matrix $\mathbf{E}$ is referred to as the *essential matrix* [14].

Vectors $\mathbf{X}_{h1}$, $\mathbf{X}_{h2}$ and $\mathbf{t}$ define the *epipolar plane* with normal vector n_π, that intersects both mirror quadrics $\mathbf{Q}$ in the two *epipolar conics* $\mathbf{C}_1$ and $\mathbf{C}_2$ [22]. For each 3D 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 the intersections of the two mirrors with the baseline $\overline{O_{m1}O_{m2}}$ (see Fig.8).

Example 4 (Epipolar geometry for panoramic cameras): Similar camera and scene configurations as for the Example 3 have been set. Now we placed 8 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 $i = 1, ..., 8$, corresponding to all 3D points in $\mathbf{X}_w$ are computed and visualized by

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

□

Fig. 9 presents the application of EGT to the computation o epipolar geometry between both *actual* (A) and *desired* (D) catadioptric views, respectively. Note that also the epipoles at the mirror surface are visualized. Epipolar conics in both CCD image plane, together with corresponding feature points, are plotted in Fig. 10 The complete MATLAB source code is available in the EGT web-site [18].

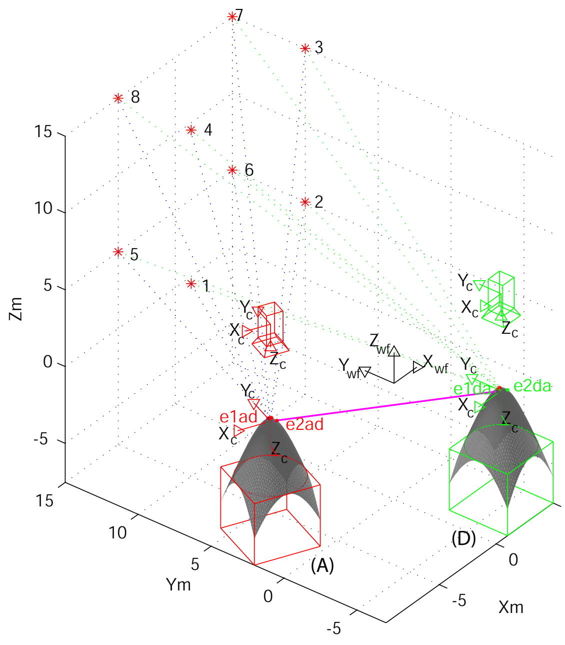


Fig. 9. Example 4: Central catadioptric cameras : epipolar geometry computation and visualization in EGT.

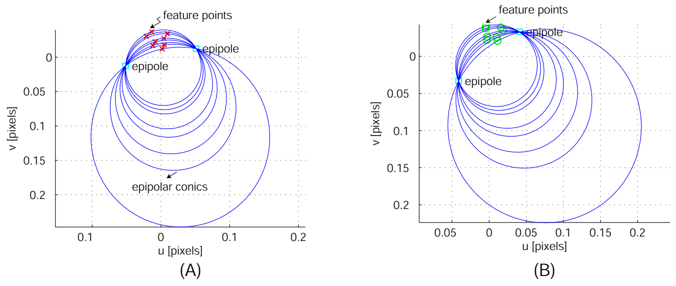


Fig. 10. Example 4: Epipolar conics plot in actual (A) and desired (D) camera planes. All epipolar conics, in each image plane, intersect at the epipoles.

C. Estimation of Epipolar Geometry

Up to now here presented functions for the epipolar geometry computation are 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 geoemtry from the knowledge of corresponding feature points $\mathbf{m}_a^i$ and $\mathbf{m}_d^i$. A very important feature of EGT is that the most important algorithms for the epipolar geometry estimation have been embedded in the toolbox code. In particular we implemented linear methods [15], the well known 8-points algorithm [12], the algorithm based on geometric distance (Sampson) [13] and finally the CFNS - Constrained method [6] as summarized in Table I. A complete comparison and review of these methods has been reported in [13]. 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 the five available reported in Table I.

It is worth noting that these functions allows EGT to estimate the epipolar geometry also in real experiments other than in simulations.

V. Application to Visual Servoing

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

The visual servoing controller proposed by Rives in [20] has been considered as a tutorial example to show the main EGT features in a visual servoing context. In what follows the algorithm proposed in [20] is briefly recalled for the reader convenience.

The main goal of this visual servoing is to drive a 6DOF robot arm (simulated with RT), equipped with a CCD camera, from a starting configuration toward a desired one using only image data provided during the robot motion (Fig. 11). The basic visual servoing idea consists in decoupling the camera/end-effector rotations and translations by the use of the hybrid vision-based task function approach [8]. The servoing task is performed minimizing an error function φ which is divided in both a *priority task* φ_1, that acts rotating the actual camera until the desired orientation is get, and in a *secondary task* φ_2 (to be minimized under the constraint $\varphi_1 = 0$) that deals with the translation of the camera toward the desired position.

The analytical expression of the error function is given by

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

ID	Estimation Algorithm
1	*Unconstrained linear estimation* [15]
2	*Linear estimation with constraint* $(det(F) = 0)$ [15]
3	*The normalized 8-point algorithm* [12]
4	*Algorithm based on Geometric Distance (Sampson)* [13]
5	*CFNS - Constrained method* [6]

TABLE I
THE FIVE EPIPOLAR GEOMETRY ESTIMATION ALGORITHMS IMPLEMENTED IN EGT (WITH BIBLIOGRAPHY).

where $\beta \in \mathbb{R}^+$, while W^+ is the pseudo-inverse of the matrix $W \in \mathbb{R}^{m \times n} : Range(W^T) = Range(J_1^T)$. Moreover, let $\varphi_2 = ({}^kt_1 - {}^2t_1)^2$ be the distance between the current translation and the desired one.

The *primary task* accomplishes to rotate the camera/end-effector using the epipolar geometry property stating that when both the initial and the desired cameras gain the same orientation, then the distances between feature points $\mathbf{m}_i$ and corresponding epipolar lines $\mathbf{l}'_i$ are zero.

Fig. 12 shows the simulation results of the algorithm previously proposed for a scene with 8 points. The dotted points in Fig. 12 shows, in the image plane $(K = I)$, the migration of the features to the epipolar lines during the rotation (first task), while crossed points correspond to the camera translation (second task).

In Fig. 13 the control inputs $(\mathbf{v}_c, \omega_c)$ and the plot of the norm of the error function e are reported.

The MATLAB code of this simulation can be downloaded at the EGT web site [18].

VI. Comparison with other packages

To the best of our knowledge no other software packages exist that deal with multiple view geometry, panoramic cameras, that are designed for the MATLAB environment.

An interesting MATLAB package is the Structure and Motion Toolkit by Torr [26]. 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 with EGT is that the work of Torr, specifically oriented to pin-hole cameras, is not specifically designed to run simulations.

Regarding the visual servoing, it is worthwhile to mention the interesting Visual Servoing Toolbox (VST) by Cervera *et al.* [2] providing a set of functions and blocks for simulation

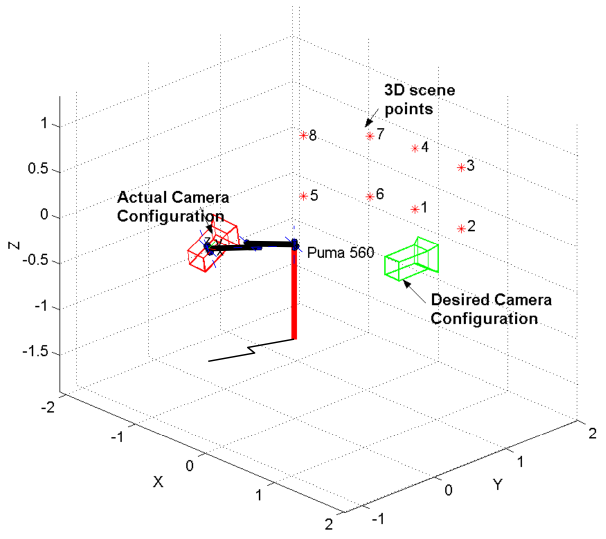


Fig. 11. Simulation setup for EGT application to visual servoing. EGT is compatible also with Robotics Toolbox.

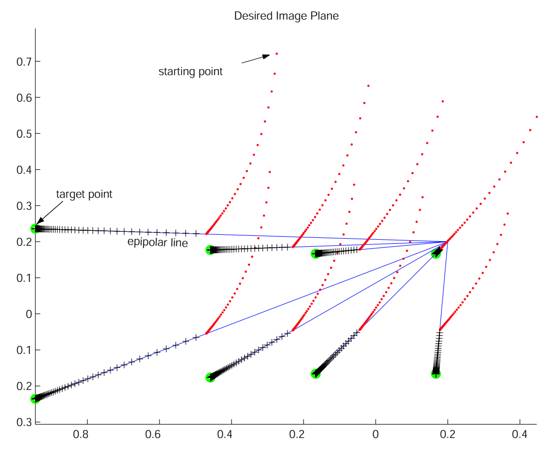


Fig. 12. Migration of feature points from initial position toward desired one. Feature points, from a starting position, reach the epipolar lines at the end of the first task and then move toward desired ones (crossed points), during the second task.

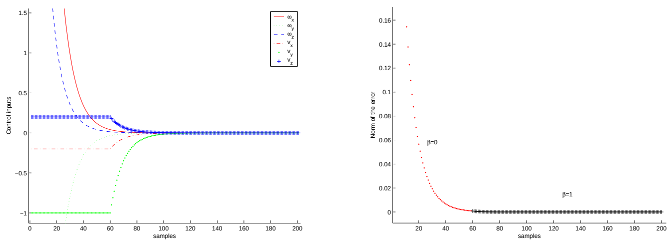


Fig. 13. (a) Control inputs to PUMA 560; (b) Norm of the error when $\beta = 0$ and when $\beta = 1$.

of vision-controlled systems. The main difference is that VST implements some visual servoing control laws but does not take consider neither multiple view geometry nor panoramic cameras that are the distinguishing features of EGT.

VII. Conclusions

The Epipolar Geometry Toolbox for MATLAB is a software package targeted to research and education in Computer Vision and Robotics Visual Servoing. It provides the user a wide set of functions for design multi-camera systems, also when panoramic camera sensors are involved. Several epipolar geometry estimation algorithms have been implemented. EGT is freely available and can be downloaded at the EGT web site together with a detailed manual and code examples. The examples in Sec. V and others are contained in the directory `demos` of the toolbox.

VIII. Acknowledgments

Authors want to thank all the students of the Robotics and Systems Laboratory the University of Siena who contributed to the testing phase of Epipolar Geometry Toolbox. Special thanks go to Jacopo Piazzi, Paola Garosi, Annalisa Bardelli and Fabio Morbidi that helped us with useful comments on the toolbox and on the paper.

References

[1] Intel's OpenCV. http://sourceforge.net/projects/opencvlibrary/.
[2] *The Visual Servoing Toolbox*, 2003. http://vstoolbox.sourceforge.net/.
[3] S. Baker and S.K. Nayar. Catadioptric image formation. In *Proc. of DARPA Image Understanding Workshop*, pages 1431–1437, 1997.
[4] R. Basri, E. Rivlin, and I. Shimshoni. Visual homing: Surfing on the epipoles. In *International Conference on Computer Vision*, pages 863–869, 1998.
[5] R. Benosman and S.B. Kang. *Panoramic Vision: Sensors, Theory and Applications*. Springer Verlag, New York, 2001.
[6] W. Chojnacki, M. J. Brooks, A. van den Hengel, and D. Gawley. A new constrained parameter estimator: experiments in fundamental matrix computation. *Image and Vision Computing, to appear*, 2003.
[7] P.I. Corke. A robotics toolbox for MATLAB. *IEEE Robotics and Automation Magazine*, 3:24–32, 1 1996.
[8] B. Espiau, F. Chaumette, and P. Rives. A new approach to visual servoing in robotics. In *IEEE Trans. on Robotics and Automation*, volume 8(3), 313–326 1992.
[9] Free Software Foundation. The free software definition. http://www.fsf.org/philosophy/free-sw.html.
[10] C. Geyer and K. Daniilidis. A unifying theory for central panoramic systems. In *Proc. of Sixth European Conference on Computer Vision*, pages 445–462, 2000. Dublin, Ireland.
[11] C. Geyer and K. Daniilidis. Mirrors in motion: Epipolar geometry and motion estimation. In *Proc. of the Ninth IEEE International Conference on Computer Vision (ICCV'03)*, 2003.
[12] R. Hartley. In defence of the 8-point algorithm. In *Proc. of IEEE Int. Conference on Computer Vision*, pages 1064–1070, Cambridge MA,USA, June 1995.
[13] R. Hartley and A. Zisserman. *"Multiple view geometry in computer vision"*. Cambridge University Press, September 2000.
[14] H.C. Longuet-Higgins. A computer algorithm for recostructing a scene from two projections. *Nature*, 293:133–135, 1981.
[15] Q.T. Luong and O.D. Faugeras. The fundamental matrix: theory, algorithms and stability analysis. *Int. Journal of Computer Vision*, 17(1):43–76, 1996.
[16] Y. Ma, S. Soatto, J. Košecká, and S. Shankar Sastry. *An invitation to 3-D Vision, From Images to Geometric Models*. Springer Verlag, New York, 2003.
[17] E. Malis and F. Chaumette. 2 1/2 d visual servoing with respect to unknown objects through a new estimation scheme of camera displacement. *International Journal of Computer Vision*, 37(1):79–97, 2000.
[18] G. Mariottini and D. Prattichizzo. *Epipolar Geometry Toolbox for Matlab*. University of Siena, `http://egt.dii.unisi.it`, 2003.
[19] G.L. Mariottini, G.Oriolo, and D.Prattichizzo. Epipole-based visual servoing for nonholonomic mobile robots. In *IEEE International Conference Robotics and Automation*, 2004.
[20] P. Rives. Visual servoing based on epipolar geometry. In *International Conference on Intelligent Robots and Systems*, volume 1, pages 602–607, 2000.
[21] M.W. Spong and M. Vidyasagar. *Robot Dynamics and Control*. John Wiley & Sons, New York, 1989.
[22] T. Svoboda. *Central Panoramic Cameras Design, Geometry, Egomotion*. Phd thesis, Center for Machine Perception, Czech Technical University, Prague, Czech Republic, September 1999.
[23] T. Svoboda and T. Pajdla. Epipolar geometry for central catadioptric cameras. *International Journal of Computer Vision*, 49(1):23–37, 2002. Kluwer Academic Publishers.
[24] T. Svoboda, T. Pajdla, and V. Hlaváč. Motion estimation using central panoramic cameras. In *Proc. IEEE Conf. on Intelligent Vehicles*, pages 335–340, 1998. http://cmp.felk.cvut.cz.
[25] Inc. The MathWorks. Matlab. www.mathworks.com.
[26] Philip Torr. *The Structure and Motion Toolkit for MATLAB*. http://wwwcms.brookes.ac.uk/ philiptorr/.
[27] R. Vidal, O. Shakernia, , and S. Sastry. Omnidirectional vision-based formation control. In *Fortieth Annual Allerton Conference on Communication, Control and Computing,*, page 16251634, 2002.