Epipolar Rectification

Andrea Fusiello
Dipartimento Scientifico e Tecnologico, Università di Verona
Ca' Vignal 2, Strada Le Grazie, I-37134 Verona, IT
Tel: +39 045 802 7088 Fax: +39 045 802 7928
E-mail: fusiello@sci.univr.it
http://www.sci.univr.it/~fusiello

March 17, 2000

1 Introduction

Given a pair of stereo images, *epipolar rectification* (or simply *rectification*) determines a transformation of each image plane such that pairs of conjugate epipolar lines become collinear and parallel to one of the image axes (usually the horizontal one). The rectified images can be thought of as acquired by a new stereo rig, obtained by rotating the original cameras. The important advantage of rectification is that computing stereo correspondences [3] is made simpler, because search is done along the horizontal lines of the rectified images.

We assume that *the stereo rig is calibrated*, i.e., the cameras' internal parameters, mutual position and orientation are known. This assumption is not strictly necessary, but leads to a simpler technique. On the other hand, when reconstructing 3-D shape of objects from stereo, calibration is mandatory in practice, and can be easily achieved [2, 13].

Rectification is a classical problem of stereo vision. Ayache [1] introduced a rectification algorithm, in which a matrix satisfying a number of constraints is hand-crafted. The distinction between necessary and arbitrary constraints is unclear. Some authors report rectification under restrictive assumptions; for instance, [11] assumes a very restrictive geometry (parallel vertical axes of the camera reference frames). Recently, [6, 14, 8] have introduced algorithms which perform rectification given a *weakly calibrated* stereo rig, i.e., a rig for which only points correspondences between images are given (or, equivalently, for which the fundamental matrix could be computed).

Latest works includes [10, 9, 12]. Some of them concentrates on the issue of minimizing the rectified image distortion. We will not address this problem, also because distortion is less severe than in the weakly calibrated case.

This document have been adapted from [5].

2 Camera model and epipolar geometry

This section recalls briefly the mathematical background on perspective projections necessary for our purposes. For more details see [4, 15].

2.1 Camera model

A pinhole camera is modeled by its *optical center* C and its *retinal plane* (or *image plane*) $\mathcal{R}$. A 3-D point W is projected into an image point M given by the intersection of $\mathcal{R}$ with the line containing C and W. The line containing C and orthogonal to $\mathcal{R}$ is called the *optical axis* and its intersection with $\mathcal{R}$ is the *principal point*. The distance between C and $\mathcal{R}$ is the *focal length*.

Let $\mathbf{w} = [x\ y\ z]^\top$ be the coordinates of W in the world reference frame (fixed arbitrarily) and $\mathbf{m} = [u\ v]^\top$ the coordinates of M in the image plane (pixels). The mapping from 3-D coordinates to 2-D coordinates

is the *perspective projection*, which is represented by a linear transformation in *homogeneous coordinates*. Let $\tilde{\mathbf{m}} = [u\ v\ 1]^\top$ and $\tilde{\mathbf{w}} = [x\ y\ z\ 1]^\top$ be the homogeneous coordinates of M and W respectively; then the perspective transformation is given by the matrix $\tilde{\mathbf{P}}$:

$$\tilde{\mathbf{m}} \simeq \tilde{\mathbf{P}}\tilde{\mathbf{w}}, \tag{1}$$

where $\simeq$ means equal up to an arbitrary scale factor. The camera is therefore modeled by its *perspective projection matrix* (henceforth PPM) $\tilde{\mathbf{P}}$, which can be decomposed, using the QR factorization, into the product

$$\tilde{\mathbf{P}} = \mathbf{A}[\mathbf{R} \mid \mathbf{t}]. \tag{2}$$

The matrix $\mathbf{A}$ depends on the *intrinsic parameters* only, and has the following form:

$$\mathbf{A} = \begin{bmatrix} \alpha_u & \gamma & u_0 \\ 0 & \alpha_v & v_0 \\ 0 & 0 & 1 \end{bmatrix}, \tag{3}$$

where $\alpha_u = -fk_u$, $\alpha_v = -fk_v$ are the focal lengths in horizontal and vertical pixels, respectively (f is the focal length in millimeters, k_u and k_v are the effective number of pixels per millimeter along the u and v axes), (u_0, v_0) are the coordinates of the *principal point*, given by the intersection of the optical axis with the retinal plane, and γ is the *skew* factor that models non-orthogonal $u - v$ axes..

The camera position and orientation (*extrinsic parameters*), are encoded by the 3×3 rotation matrix $\mathbf{R}$ and the translation vector $\mathbf{t}$, representing the rigid transformation that brings the camera reference frame onto the world reference frame.

Let us write the PPM as

$$\tilde{\mathbf{P}} = \left[\begin{array}{c|c} \mathbf{q}_1^\top & q_{14} \\ \mathbf{q}_2^\top & q_{24} \\ \mathbf{q}_3^\top & q_{34} \end{array}\right] = [\mathbf{Q}|\tilde{\mathbf{q}}]. \tag{4}$$

In Cartesian coordinates, the projection (1) writes

$$\begin{cases} u = \dfrac{\mathbf{q}_1^\top \mathbf{w} + q_{14}}{\mathbf{q}_3^\top \mathbf{w} + q_{34}} \\ v = \dfrac{\mathbf{q}_2^\top \mathbf{w} + q_{24}}{\mathbf{q}_3^\top \mathbf{w} + q_{34}}. \end{cases} \tag{5}$$

The *focal plane* is the plane parallel to the retinal plane that contains the optical center C. The coordinates $\mathbf{c}$ of C are given by

$$\mathbf{c} = -\mathbf{Q}^{-1}\tilde{\mathbf{q}}. \tag{6}$$

Therefore $\tilde{\mathbf{P}}$ can be written:

$$\tilde{\mathbf{P}} = [\mathbf{Q}| - \mathbf{Q}\mathbf{c}]. \tag{7}$$

The *optical ray* associated to an image point M is the line $\mathsf{M}\ \mathsf{C}$, i.e. the set of 3-D points $\{\mathbf{w} : \tilde{\mathbf{m}} \simeq \tilde{\mathbf{P}}\tilde{\mathbf{w}}\}$. Its parametric equation in Cartesian coordinates writes:

$$\mathbf{w} = \mathbf{c} + \lambda\mathbf{Q}^{-1}\tilde{\mathbf{m}}, \quad \lambda \in \mathbb{R}. \tag{8}$$

2.2 Epipolar geometry

Let us consider a stereo rig composed by two pinhole cameras (Fig. 1). Let C_1 and C_2 be the optical centers of the left and right cameras respectively. A 3-D point W is projected onto both image planes, to points M_1

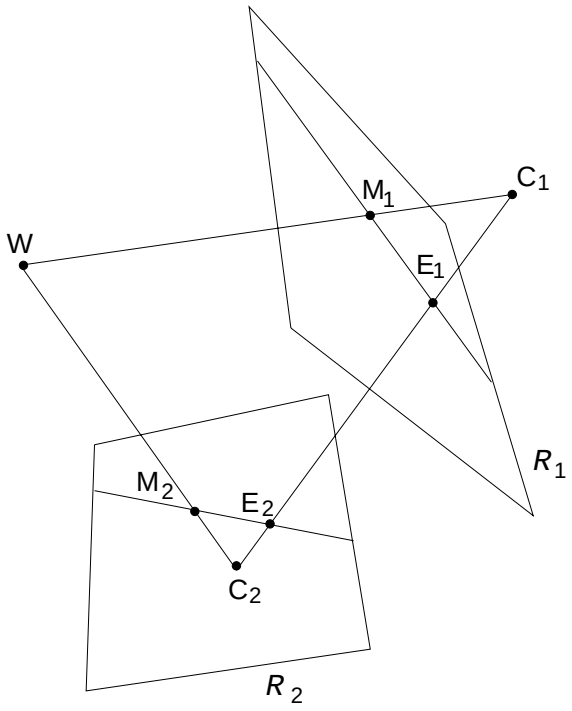


Figure 1: Epipolar geometry.

and M_2, which constitute a conjugate pair. Given a point M_1 in the left image plane, its conjugate point in the right image is constrained to lie on a line called the *epipolar line* (of M_1). Since M_1 may be the projection of an arbitrary point on its optical ray, the epipolar line is the projection through C_2 of the optical ray of M_1. All the epipolar lines in one image plane pass through a common point (E_1 and E_2 respectively) called the *epipole*, which is the projection of the optical center of the other camera.

When C_1 is in the focal plane of the right camera, the right epipole is at infinity, and the epipolar lines form a bundle of parallel lines in the right image. A very special case is when both epipoles are at infinity, that happens when the line $\mathsf{C}_1\mathsf{C}_2$ (the *baseline*) is contained in both focal planes, i.e., the retinal planes are parallel to the baseline. Epipolar lines, then, form a bundle of parallel lines in both images. Any pair of images can be transformed so that epipolar lines are parallel and horizontal in each image. This procedure is called *rectification*.

3 Rectification of camera matrices

We assume that the stereo rig is *calibrated*, i.e., the PPMs $\tilde{\mathbf{P}}_{o1}$ and $\tilde{\mathbf{P}}_{o2}$ are known. The idea behind rectification is to define two new PPMs $\tilde{\mathbf{P}}_{n1}$ and $\tilde{\mathbf{P}}_{n2}$ obtained by rotating the old ones around their optical centers until focal planes becomes coplanar, thereby containing the baseline. This ensures that epipoles are at infinity, hence epipolar lines are *parallel*. To have *horizontal* epipolar lines, the baseline must be parallel to the new X axis of both cameras. In addition, to have a proper rectification, conjugate points must have the *same vertical coordinate*. This is obtained by requiring that the new cameras have the same intrinsic parameters. Note that, being the focal length the same, retinal planes are coplanar too, as in Figure 2.

In summary: positions (i.e, optical centers) of the new PPMs are the same as the old cameras, whereas the new orientation (the same for both cameras) differs from the old ones by suitable rotations; intrinsic parameters are the same for both cameras. Therefore, the two resulting PPMs will differ only in their optical centers, and they can be thought as a single camera translated along the X axis of its reference system.

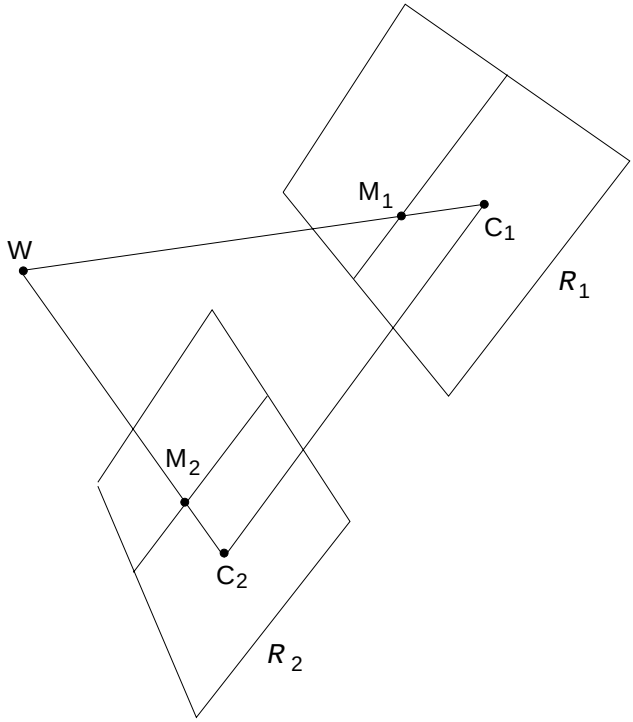


Figure 2: Rectified cameras. Retinal planes are coplanar and parallel to the baseline.

Let us write the new PPMs in terms of their factorization. From (2) and (7):

$$\tilde{\mathbf{P}}_{n1} = \mathbf{A}[\mathbf{R} \mid -\mathbf{R}\,\mathbf{c}_1], \quad \tilde{\mathbf{P}}_{n2} = \mathbf{A}[\mathbf{R} \mid -\mathbf{R}\,\mathbf{c}_2]. \tag{9}$$

The intrinsic parameters matrix $\mathbf{A}$ is the same for both PPMs, and can be chosen arbitrarily (see MATLAB code). The optical centers $\mathbf{c}_1$ and $\mathbf{c}_2$ are given by the old optical centers, computed with (6). The matrix $\mathbf{R}$, which gives the camera's pose, is the same for both PPMs. It will be specified by means of its row vectors

$$\mathbf{R} = \begin{bmatrix} \mathbf{r}_1^\top \\ \mathbf{r}_2^\top \\ \mathbf{r}_3^\top \end{bmatrix} \tag{10}$$

that are the X, Y, and Z axes, respectively, of the camera reference frame, expressed in world coordinates.

According to the previous comments, we take:

1. The new X axis parallel to the baseline: $\mathbf{r}_1 = (\mathbf{c}_1 - \mathbf{c}_2)/||\mathbf{c}_1 - \mathbf{c}_2||$
2. The new Y axis orthogonal to X (mandatory) and to $\mathbf{k}$: $\mathbf{r}_2 = \mathbf{k} \wedge \mathbf{r}_1$
3. The new Z axis orthogonal to XY (mandatory) : $\mathbf{r}_3 = \mathbf{r}_1 \wedge \mathbf{r}_2$

In point 2, $\mathbf{k}$ is an arbitrary unit vector, that fixes the position of the new Y axis in the plane orthogonal to X. We take it equal to the Z unit vector of the old left matrix, thereby constraining the new Y axis to be orthogonal to both the new X and the old left Z.

This algorithm fails when the optical axis is parallel to the baseline, i.e., when there is a pure forward motion.

4 The rectifying transformation

In order to rectify – let's say – the left image, we need to compute the transformation mapping the image plane of $\tilde{\mathbf{P}}_{o1} = [\mathbf{Q}_{o1}|\tilde{\mathbf{q}}_{o1}]$ onto the image plane of $\tilde{\mathbf{P}}_{n1} = [\mathbf{Q}_{n1}|\tilde{\mathbf{q}}_{n1}]$. It is useful to think of an image as the intersection of the image plane with the cone of rays between points in 3D space and the optical centre. We are moving the image plane while leaving fixed the cone of rays. We will see that the sought transformation is the *collinearity* (linear transformatiopn of the projective plane) given by the 3×3 matrix $\mathbf{T}_1 = \mathbf{Q}_{n1}\mathbf{Q}_{o1}^{-1}$. The same result applies to the right image.

For any 3-D point $\mathbf{w}$ we can write

$$\begin{cases} \tilde{\mathbf{m}}_{o1} \simeq \tilde{\mathbf{P}}_{o1}\tilde{\mathbf{w}} \\ \tilde{\mathbf{m}}_{n1} \simeq \tilde{\mathbf{P}}_{n1}\tilde{\mathbf{w}}. \end{cases} \tag{11}$$

According to (8), the equations of the optical rays are the following (since rectification does not move the optical center):

$$\begin{cases} \mathbf{w} = \mathbf{c}_1 + \lambda_o \mathbf{Q}_{o1}^{-1}\tilde{\mathbf{m}}_{o1} \quad \lambda_o \in \mathbb{R} \\ \mathbf{w} = \mathbf{c}_1 + \lambda_n \mathbf{Q}_{n1}^{-1}\tilde{\mathbf{m}}_{n1} \quad \lambda_n \in \mathbb{R}; \end{cases} \tag{12}$$

hence

$$\tilde{\mathbf{m}}_{n1} = \lambda \mathbf{Q}_{n1}\mathbf{Q}_{o1}^{-1}\tilde{\mathbf{m}}_{o1} \quad \lambda \in \mathbb{R} \tag{13}$$

(note that λ absorbes the arbitrary scale factor, hence we use $=$).

The transformation $\mathbf{T}_1$ is then applied to the original left image to produce the rectified image, as in Figure 3. Note that the pixels (integer-coordinate positions) of the rectified image correspond, in general, to non-integer positions on the original image plane. Therefore, the gray levels of the rectified image are computed by bilinear interpolation.

Reconstruction of 3-D points by triangulation can be performed from the rectified images directly, using `Pn1,Pn2`.

5 Summary of the rectification algorithm

Given the high diffusion of stereo in research and applications, we have endeavored to make our algorithm as easily reproducible and usable as possible. To this purpose, we give the working MATLAB code of the algorithm; the code is simple and compact (22 lines), and the comments enclosed make it understandable without knowledge of MATLAB. The usage of the `rectify` function (see MATLAB code) is the following:

- Given a stereo pair of images `I1,I2` and PPMs `Po1,Po2` (obtained by calibration);
- compute `[T1,T2,Pn1,Pn2] = rectify(Po1,Po2)`;
- rectify images by applying `T1` and `T2`.

```
function [A,R,t] = art(P)
% ART: factorize a PPM as  P=A*[R;t]

Q = inv(P(1:3, 1:3));
[U,B] = qr(Q);

R = inv(U);
t = B*P(1:3,4);
A = inv(B);
A = A ./A(3,3);
```

```
function [T1,T2,Pn1,Pn2] = rectify(Po1,Po2)

% RECTIFY: compute rectification matrices

% factorize old PPMs
[A1,R1,t1] = art(Po1);
[A2,R2,t2] = art(Po2);

% optical centers (unchanged)
c1 = - inv(Po1(:,1:3))*Po1(:,4);
c2 = - inv(Po2(:,1:3))*Po2(:,4);

% new x axis (= direction of the baseline)
v1 = (c1-c2);
% new y axes (orthogonal to new x and old z)
v2 = cross(R1(3,:)',v1);
% new z axes (orthogonal to baseline and y)
v3 = cross(v1,v2);

% new extrinsic parameters
R = [v1'/norm(v1)
     v2'/norm(v2)
     v3'/norm(v3)];
% translation is left unchanged

% new intrinsic parameters (arbitrary)
A = (A1 + A2)./2;
A(1,2)=0; % no skew

% new projection matrices
Pn1 = A * [R -R*c1 ];
Pn2 = A * [R -R*c2 ];

% rectifying image transformation
T1 = Pn1(1:3,1:3)* inv(Po1(1:3,1:3));
T2 = Pn2(1:3,1:3)* inv(Po2(1:3,1:3));
```

The MATLAB anc C implementation of the algorithm can be found on line.

6 Experiments

For this experiments we used calibrated stereo pairs from INRIA-Syntim. We show the results obtained with a nearly rectified stereo rig (Figure 3) and with a more general stereo geometry (Figure 4). The pixel coordinates of the rectified images are not constrained to lie in any special part of the image plane, and an arbitrary translation were applied to both images to bring them in a suitable region of the plane, by changing the image center. Then the output images were cropped to the size of the input images.

For example, In the case of the "Sport" stereo pair (image size 768×576), we started from the following camera matrices:

$$\mathbf{P}_{o1} = \begin{bmatrix} 9.765{\cdot}10^{2} & 5.382{\cdot}10^{1} & -2.398{\cdot}10^{2} & 3.875{\cdot}10^{5} \\ 9.849{\cdot}10^{1} & 9.333{\cdot}10^{2} & 1.574{\cdot}10^{2} & 2.428{\cdot}10^{5} \\ 5.790{\cdot}10^{-1} & 1.108{\cdot}10^{-1} & 8.077{\cdot}10^{-1} & 1.118{\cdot}10^{3} \end{bmatrix}$$

$$\mathbf{P}_{o2} = \begin{bmatrix} 9.767{\cdot}10^{2} & 5.376{\cdot}10^{1} & -2.400{\cdot}10^{2} & 4.003{\cdot}10^{4} \\ 9.868{\cdot}10^{1} & 9.310{\cdot}10^{2} & 1.567{\cdot}10^{2} & 2.517{\cdot}10^{5} \\ 5.766{\cdot}10^{-1} & 1.141{\cdot}10^{-1} & 8.089{\cdot}10^{-1} & 1.174{\cdot}10^{3} \end{bmatrix}$$

After adding the statement `A(1,3) = A(1,3) + 160`
to the `rectify` program, to keep the rectified image in the center of the 768×576 window, we obtained the following rectified camera matrices:

$$\mathbf{P}_{n1} = \begin{bmatrix} 1.043{\cdot}10^{3} & 7.452{\cdot}10^{1} & -2.585{\cdot}10^{2} & 4.124{\cdot}10^{5} \\ 1.165{\cdot}10^{2} & 9.338{\cdot}10^{2} & 1.410{\cdot}10^{2} & 2.388{\cdot}10^{5} \\ 6.855{\cdot}10^{-1} & 1.139{\cdot}10^{-1} & 7.190{\cdot}10^{-1} & 1.102{\cdot}10^{3} \end{bmatrix}$$

$$\mathbf{P}_{n2} = \begin{bmatrix} 1.043{\cdot}10^{3} & 7.452{\cdot}10^{1} & -2.585{\cdot}10^{2} & 4.069{\cdot}10^{4} \\ 1.165{\cdot}10^{2} & 9.338{\cdot}10^{2} & 1.410{\cdot}10^{2} & 2.388{\cdot}10^{5} \\ 6.855{\cdot}10^{-1} & 1.139{\cdot}10^{-1} & 7.190{\cdot}10^{-1} & 1.102{\cdot}10^{3} \end{bmatrix}$$



Figure 3: "Sport" stereo pair (top) and rectified pair (bottom). The right pictures plot the epipolar lines corresponding to the points marked in the left pictures.

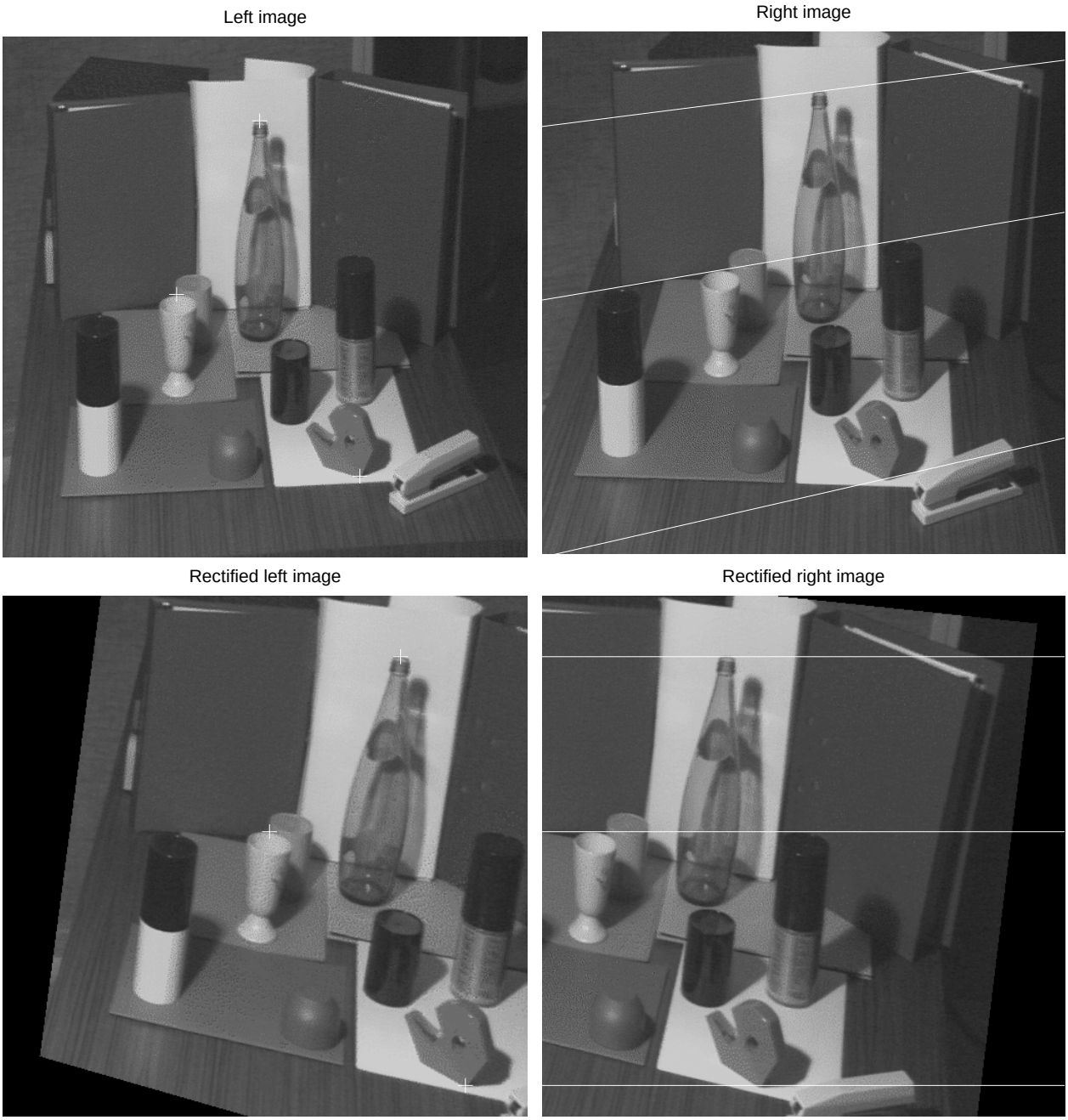


Figure 4: "Color" stereo pair (top) and rectified pair (bottom). The right pictures plot the epipolar lines corresponding to the points marked in the left pictures.

References

[1] N. Ayache and F. Lustman. Trinocular stereo vision for robotics. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 13:73–85, January 1991.

[2] B. Caprile and V. Torre. Using vanishing points for camera calibration. *International Journal of Computer Vision*, 4:127–140, 1990.

[3] U. R. Dhond and J. K. Aggarwal. Structure from stereo – a review. *IEEE Transactions on Systems, Man and Cybernetics*, 19(6):1489–1510, November/December 1989.

[4] O. Faugeras. *Three-Dimensional Computer Vision: A Geometric Viewpoint.* The MIT Press, Cambridge, MA, 1993.

[5] A. Fusiello, E. Trucco, and A. Verri. A compact algorithm for rectification of stereo pairs. *Machine Vision and Applications*, 2000. To appear.

[6] R. Hartley and R. Gupta. Computing matched-epipolar projections. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, pages 549–555, 1993.

[7] R. I. Hartley and P. Sturm. Triangulation. *Computer Vision and Image Understanding*, 68(2):146–157, November 1997.

[8] R.I. Hartley. Theory and practice of projective rectification. *International Journal of Computer Vision*, 35(2):1–16, November 1999.

[9] F. Isgro and E. Trucco. Projective rectification without epipolar geometry. In *CVPR99*, pages I:94–99, 1999.

[10] C. Loop and Z. Zhang. Computing rectifying homographies for stero vision. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, pages I:125–131, 1999.

[11] D. V. Papadimitriou and T. J. Dennis. Epipolar line estimation and rectification for stereo images pairs. *IEEE Transactions on Image Processing*, 3(4):672–676, April 1996.

[12] M. Pollefeys, R. Koch, and L. VanGool. A simple and efficient rectification method for general motion. In *ICCV99*, pages 496–501, 1999.

[13] L Robert. Camera calibration without feature extraction. *Computer Vision, Graphics, and Image Processing*, 63(2):314–325, March 1996.

[14] L. Robert, C. Zeller, O. Faugeras, and M. Hébert. Applications of non-metric vision to some visually-guided robotics tasks. In Y. Aloimonos, editor, *Visual Navigation: From Biological Systems to Unmanned Ground Vehicles*, chapter 5, pages 89–134. Lawrence Erlbaum Associates, 1997.

[15] E. Trucco and A. Verri. *Introductory Techniques for 3-D Computer Vision.* Prentice-Hall, 1998.