Elements of Geometric Computer Vision

Andrea Fusiello[*]

http://www.sci.univr.it/~fusiello

May 29, 2006

©E. Trucco

1 Introduction

This notes introduces the basic geometric concepts of multiple-view computer vision. The focus is on geometric models of perspective cameras, and the constraints and properties such models generate when multiple cameras observe the same 3-D scene.

Geometric vision is an important and well-studied part of computer vision. A wealth of useful results has been achieved in the last 15 years and has been reported in comprehensive monographies, e.g., [5, 14, 7], a sign of maturity for a research subject.

It is worth reminding the reader that geometry is an important but not the only important aspect of computer vision, and in particular of multi-view vision. The information brought by each image pixel is twofold: its *position* and its *colour* (or brightness, for a monochrome image). Ultimately, each computer vision system must start with brightness values, and, to smaller or greater depth, link such values to the 3-D world.

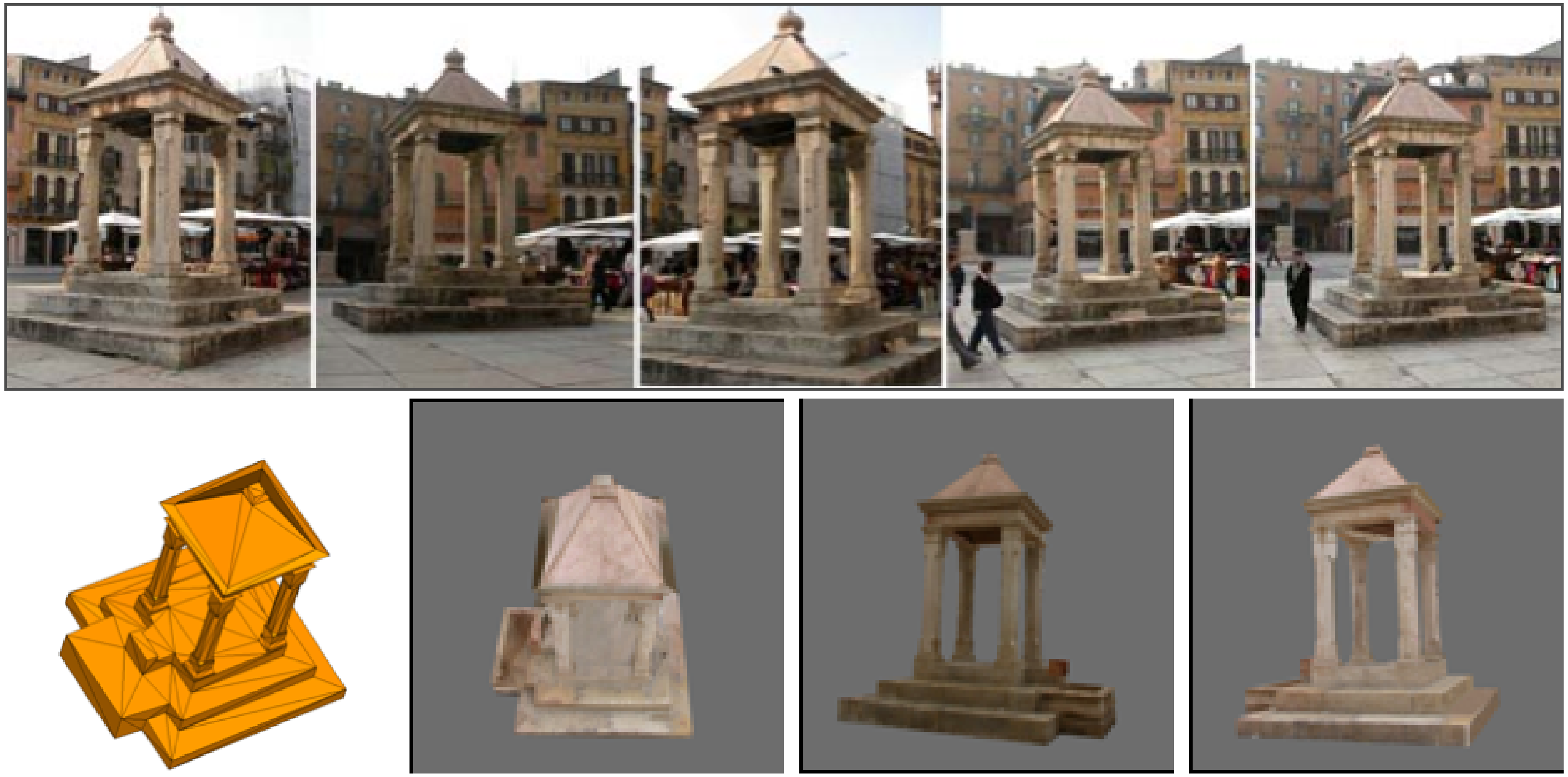


Fig. 1. Example of reconstruction from images. Original images (top row) and reconstructed model (bottom row).

2 Projective Geometry

The ambient space is modellled as a projective 3-D space $\mathbb{P}^3$, obtained by completing the affine space $\mathbb{X}^3$ with a projective plane, known as plane at infinity Π_∞. In this ideal plane lie the intersections of the planes parallel in $\mathbb{X}^3$.

The projective coordinates of a point in $\mathbb{P}^3$ are 4-tuples defined up to a scale factor. We write

$$\mathbf{M} \simeq (x, y, z, t) \tag{1}$$

where $\simeq$ indicates equality to within a multiplicative factor.

Π_∞ is defined by its equation $t = 0$.

The points of the affine space are those of $\mathbb{P}^3$ which do not belong to Π_∞. Their projective coordinates are thus of the form $(x, y, z, 1)$, where (x, y, z) are the usual affine coordinates.

The linear transformations of a projective space into itself are called collineations or homographies. Any collineation of $\mathbb{P}^3$ is represented by a generic 4×4 invertible matrix.

Affine transformations of $\mathbb{X}^3$ are the subgroup of collineations of $\mathbb{P}^3$ that preserves the plane at infinity.

Similarity transformations are the subgroup of affine transformations of $\mathbb{X}^3$ that leave invariant a very special curve, the *absolute conic*, which is in the plane at infinity and whose equation is:

$$x^2 + y^2 + z^2 = 0 = t \tag{2}$$

The space is therefore stratified into more and more specialized structures:

- projective
- affine (knowing the plane at infinity)
- Euclidean (knowing the absolute conic)

The stratification reflects the amount of knowledge that we possess about the scene and the sensor.

3 Pin-hole Camera Geometry

The pin-hole camera is described by its *optical centre* $\mathbf{C}$ (also known as *camera projection centre*) and the *image plane*.

The distance of the image plane from $\mathbf{C}$ is the *focal length* f.

The line from the camera centre perpendicular to the image plane is called the *principal axis* or *optical axis* of the camera.

The plane parallel to the image plane containing the optical centre is called the *principal plane* or *focal plane* of the camera.

The relationship between the 3-D coordinates of a scene point and the coordinates of its projection onto the image plane is described by the *central* or *perspective projection*.

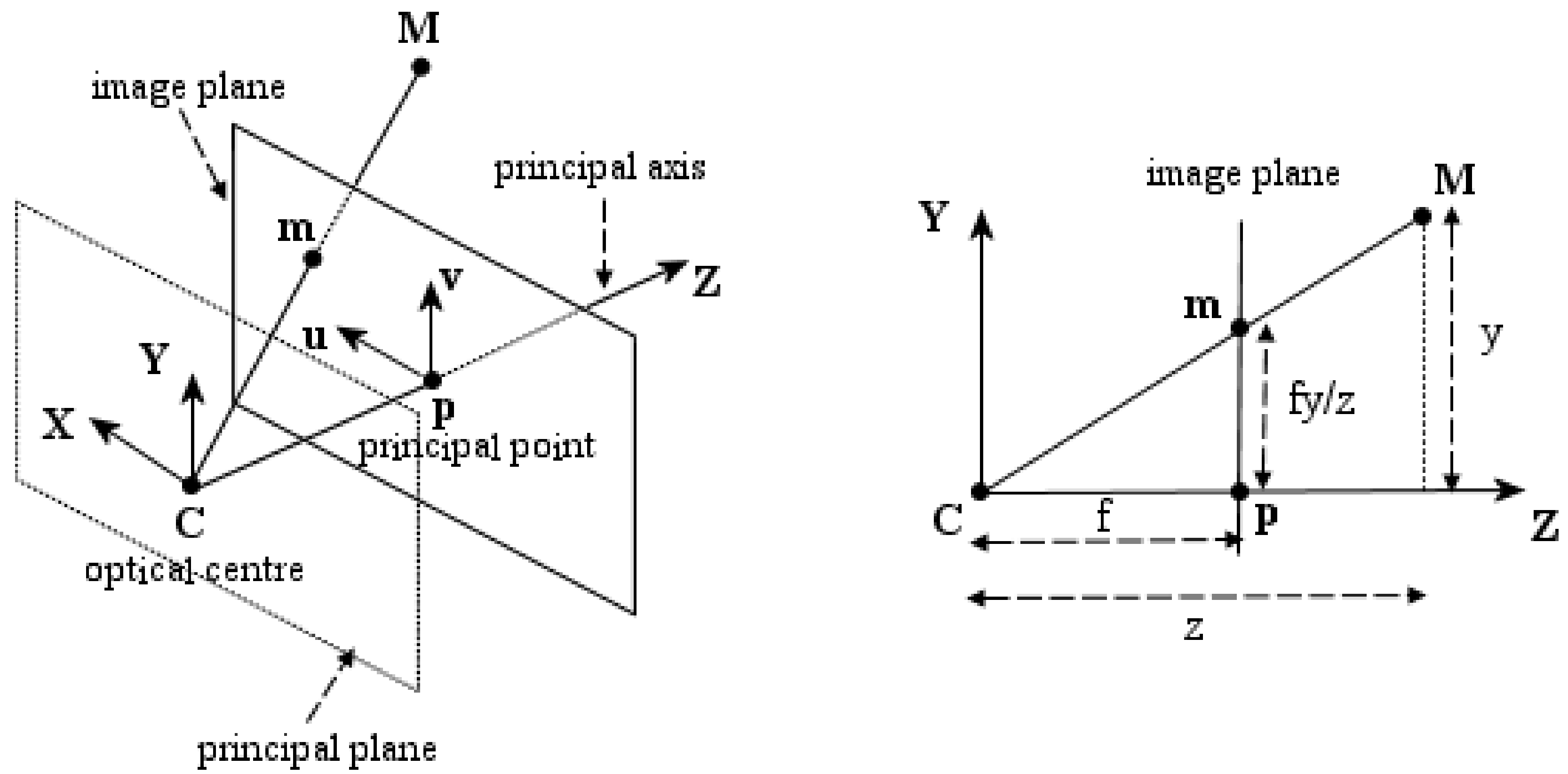


Fig. 2. Pin-hole camera geometry. The left figure illustrates the projection of the point **M** on the image plane by drawing the line through the camera centre **C** and the point to be projected. The right figure illustrates the same situation in the YZ plane, showing the similar triangles used to compute the position of the projected point **m** in the image plane.

A 3-D point is projected onto the image plane with the line containing the point and the optical centre (see Figure 2).

Let the centre of projection be the origin of a Euclidean coordinate system wherein the z-axis is the principal axis.

By similar triangles it is readily seen that the 3-D point $(x, y, z)^T$ is mapped to the point $(fx/z, fy/z)^T$ on the image plane.

3.1 The camera projection matrix

If the world and image points are represented by homogeneous vectors, then perspective projection can be expressed in terms of matrix multiplication as

$$\begin{pmatrix} fx \\ fy \\ z \end{pmatrix} = \begin{bmatrix} f & 0 & 0 & 0 \\ 0 & f & 0 & 0 \\ 0 & 0 & 1 & 0 \end{bmatrix} \begin{pmatrix} x \\ y \\ z \\ 1 \end{pmatrix} \tag{3}$$

The matrix describing the mapping is called the *camera projection matrix* P.

Equation (3) can be written simply as:

$$z\mathbf{m} = P\mathbf{M} \tag{4}$$

where $\mathbf{M} = (x, y, z, 1)^T$ are the homogeneous coordinates of the 3-D point and $\mathbf{m} = (fx/z, fy/z, 1)^T$ are the homogeneous coordinates of the image point.

The projection matrix P in Equation (3) represents the simplest possible case, as it only contains information about the focal distance f.

General camera: bottom up approach

The above formulation assumes a special choice of world coordinate system and image coordinate system. It can be generalized by introducing suitable changes of the coordinates systems.

Changing coordinates in space is equivalent to multiplying the matrix P to the right by a 4×4 matrix:

$$G = \begin{bmatrix} R & \mathbf{t} \\ 0 & 1 \end{bmatrix} \tag{5}$$

G is composed by a rotation matrix R and a translation vector $\mathbf{t}$. It describes the position and orientation of the camera with respect to an external (world) coordinate system. It depends on six parameters, called *external* parameters.

The rows of R are unit vectors that, together with the optical centre, define the *camera reference frame*, expressed in world coordinates.

Changing coordinates in the image plane is equivalent to multiplying the matrix P to the left by a 3×3 matrix:

$$K = \begin{bmatrix} f/s_x & f/s_x \cot\theta & o_x \\ 0 & f/s_y & o_y \\ 0 & 0 & 1 \end{bmatrix} \tag{6}$$

K is the *camera calibration matrix*; it encodes the transformation from image coordinates to pixel coordinates in the image plane.

It depends on the so-called *intrinsic* parameters:

- focal distance f (in mm),
- principal point (or image centre) coordinates o_x, o_y (in pixel),
- width (s_x) and height (s_y) of the pixel footprint on the camera photosensor (in mm),
- angle θ between the axes (usually $\pi/2$).

The ratio s_y/s_x is the aspect ratio (usually close to 1).

Thus the camera matrix, in general, is the product of three matrices:

$$P = K[I|\mathbf{0}]G = K[R|\mathbf{t}] \tag{7}$$

In general, the projection equation writes:

$$\zeta \mathbf{m} = P\mathbf{M} \tag{8}$$

where ζ is the distance of $\mathbf{M}$ from the focal plane of the camera (this will be shown after), and $\mathbf{m} = (u, v, 1)^T$.

Note that, except for a very special choice of the world reference frame, *this "depth" does not coincide with the third coordinate of* $\mathbf{M}$.

General camera: top down approach

If P describes a camera, also λP for any $0 \neq \lambda \in \mathbb{R}$ describes the same camera, since these give the same image point for each scene point.

In this case we can also write:

$$\mathbf{m} \simeq P\mathbf{M} \tag{9}$$

where $\simeq$ means "equal up to a scale factor."

In general, the camera projection matrix is a 3×4 full-rank matrix and, being homogeneous, it has 11 degrees of freedom.

Using QR factorization, it can be shown that any 3×4 full rank matrix P can be factorised as:

$$P = \lambda K[R|\mathbf{t}], \tag{10}$$

(λ is recovered from $K(3,3) = 1$).

3.2 Camera anatomy

Projection centre

The camera projection centre $\mathbf{C}$ is the only point for which the projection is not defined, i.e.:

$$P\mathbf{C} = P\begin{pmatrix}\tilde{\mathbf{C}}\\ 1\end{pmatrix} = \mathbf{0} \tag{11}$$

where $\tilde{\mathbf{C}}$ is a 3-D vector containing the affine (non-homogeneous) coordinates of the optical centre.

After solving for $\tilde{\mathbf{C}}$ we obtain:

$$\tilde{\mathbf{C}} = -P_{1:3}^{-1}P_4 \tag{12}$$

where the matrix P is represented by the block form: $P = [P_{1:3}|P_4]$ (the subscript denotes a range of columns).

Depth of a point

We observe that:

$$\zeta \mathbf{m} = P\mathbf{M} = P\mathbf{M} - P\mathbf{C} = P(\mathbf{M} - \mathbf{C}) = P_{1:3}(\tilde{\mathbf{M}} - \tilde{\mathbf{C}}). \tag{13}$$

In particular, plugging Eq. (10), the third component of this equation is

$$\zeta = \lambda \mathbf{r}_3^T(\tilde{\mathbf{M}} - \tilde{\mathbf{C}})$$

where $\mathbf{r}_3^T$ is the third row of the rotation matrix R, which correspond to the versor of the principal axis.

If $\lambda = 1$, ζ is the projection of the vector $(\tilde{\mathbf{M}} - \tilde{\mathbf{C}})$ onto the principal axis, i.e., the *depth* of $\mathbf{M}$.

If $\lambda = 1$ in Eq. (10), the matrix P is said to be *normalized*.

Optical ray

The projection can be geometrically modelled by a ray through the optical centre and the point in space that is being projected onto the image plane (see Fig. 2).

The *optical ray* of an image point $\mathbf{m}$ is the locus of points in space that projects onto $\mathbf{m}$.

It can be described as a parametric line passing through the camera projection centre $\mathbf{C}$ and a special point (at infinity) that projects onto $\mathbf{m}$:

$$\mathbf{M} = \begin{pmatrix} -P_{1:3}^{-1} P_4 \\ 1 \end{pmatrix} + \zeta \begin{pmatrix} P_{1:3}^{-1} \mathbf{m} \\ 0 \end{pmatrix}, \quad \zeta \in \mathbb{R}. \tag{14}$$

Please note that, provided that P is normalized, the parameter ζ in the equation of the optical ray correspond to the depth of the point $\mathbf{M}$. $\textcircled{\scriptsize 01}$

3.2.1 Image of the absolute conic

We will prove now that the image of the absolute conic depends on the intrinsic parameters only (it is unaffected by camera position and orientation).

The points in the plane at infinity have the form $\mathbf{M} = (\tilde{\mathbf{M}}^T, 0)^T$, hence

$$\mathbf{m} \simeq K[R \mid \mathbf{t}](\tilde{\mathbf{M}}^T, 0)^T = KR\tilde{\mathbf{M}}. \tag{15}$$

The image of points on the plane at infinity does not depend on camera position (it is unaffected by camera translation).

The absolute conic (which is in the plane at infinity) has equation $\tilde{\mathbf{M}}^T\tilde{\mathbf{M}} = 0$, therefore its projection has equation:

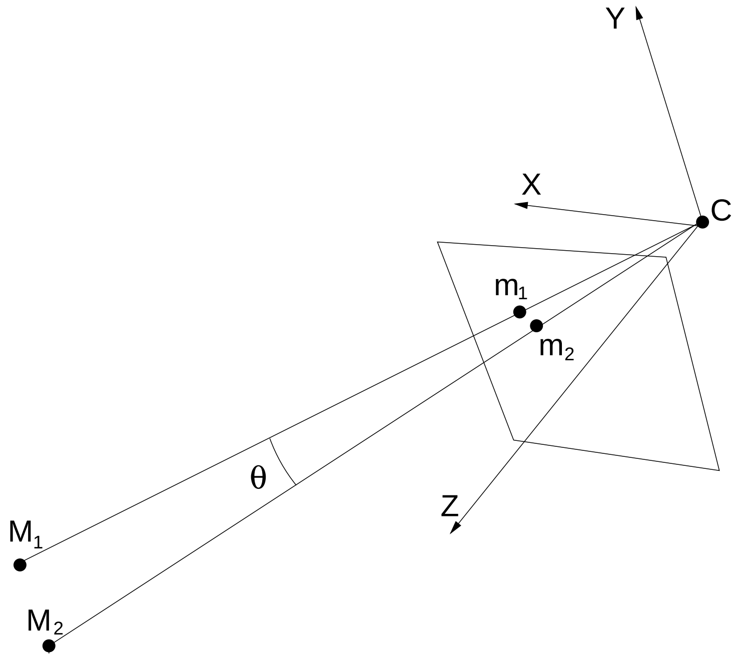
$$\mathbf{m}^T(K^{-T}K^{-1})\mathbf{m} = 0. \tag{16}$$

The conic $\boldsymbol{\omega} = (KK^{-T})^{-1}$ is the image of the absolute conic.

The angle (a metrical property) between two rays is determined by the image of the absolute conic. (02)

Let us consider a camera $P = [K|\mathbf{0}]$, then $\mathbf{m} = \frac{1}{z}K\tilde{\mathbf{M}}$. Let θ be the angle between the rays trough $\mathbf{M}_1$ and $\mathbf{M}_1$, then

$$\cos\theta = \frac{\tilde{\mathbf{M}}_1^T\tilde{\mathbf{M}}_2}{||\tilde{\mathbf{M}}_1||||\tilde{\mathbf{M}}_2||} = \frac{\mathbf{m}_1^T\boldsymbol{\omega}\mathbf{m}_2}{\sqrt{\mathbf{m}_1^T\boldsymbol{\omega}\mathbf{m}_1}\sqrt{\mathbf{m}_2^T\boldsymbol{\omega}\mathbf{m}_2}} \tag{17}$$



3.3 Camera calibration (or resection)

A number of point correspondences $\mathbf{m}_i \leftrightarrow \mathbf{M}_i$ is given, and we are required to find a camera matrix P such that

$$\mathbf{m}_i \simeq P\mathbf{M}_i \quad \text{for all } i. \tag{18}$$

The equation can be rewritten in terms of the cross product as

$$\mathbf{m}_i \times P\mathbf{M}_i = \mathbf{0}. \tag{19}$$

This form will enable a simple a simple linear solution for P to be derived. Using the properties of the Kronecker product ($\otimes$) and the vec operator [35], we derive:

$$\mathbf{m}_i \times P\mathbf{M}_i = \mathbf{0} \iff [\mathbf{m}_i]_\times P\mathbf{M}_i = \mathbf{0} \iff \mathrm{vec}([\mathbf{m}_i]_\times P\mathbf{M}_i) = \mathbf{0} \iff$$
$$\iff (\mathbf{M}_i^T \otimes [\mathbf{m}_i]_\times)\,\mathrm{vec}\,P = \mathbf{0} \iff ([\mathbf{m}_i]_\times \otimes \mathbf{M}_i^T)\,\mathrm{vec}\,P^T = \mathbf{0}$$

After expanding the coefficient matrix, we obtain

$$\begin{bmatrix} \mathbf{0}^T & -\mathbf{M}_i^T & v_i\mathbf{M}_i^T \\ \mathbf{M}_i^T & \mathbf{0}^T & -u_i\mathbf{M}_i^T \\ -v_i\mathbf{M}_i^T & u_i\mathbf{M}_i^T & \mathbf{0}^T \end{bmatrix} \mathrm{vec}\,P^T = \mathbf{0} \tag{20}$$

Although there are three equations, only two of them are linearly independent: we can write the third row (e.g.) as a linear combination of the first two.

From a set of n point correspondences, we obtain a $2n \times 12$ coefficient matrix A by stacking up two equations for each correspondence.

In general A will have rank 11 (provided that the points are not all coplanar) and the solution is the 1-dimensional right null-space of A.

The projection matrix P is computed by solving the resulting linear system of equations, for $n \geq 6$.

If the data are not exact (noise is generally present) the rank of A will be 12 and a least-squares solution is sought.

The least-squares solution for $\mathrm{vec}(P^T)$ is the singular vector corresponding to the smallest singular value of A.

This is called the Direct Linear Transform (DLT) algorithm [14].

4 Two-View Geometry

The two-view geometry is the intrinsic geometry of two different perspective views of the same 3-D scene (see Figure 3). It is usually referred to as *epipolar geometry*.

The two perspective views may be acquired simultaneously, for example in a stereo rig, or sequentially, for example by a moving camera. From the geometric viewpoint, the two situations are equivalent, provided that that the scene do not change between successive snapshots.

Most 3-D scene points must be visible in both views simultaneously. This is not true in case of occlusions, i.e., points visible only in one camera. Any unoccluded 3-D scene point $\mathbf{M} = (x, y, z, 1)^T$ is projected to the left and right view as $\mathbf{m}_\ell = (u_\ell, v_\ell, 1)^T$ and $\mathbf{m}_r = (u_r, v_r, 1)^T$, respectively (see Figure 3).

Image points $\mathbf{m}_\ell$ and $\mathbf{m}_r$ are called *corresponding points* (or conjugate points) as they represent projections of the same 3-D scene point $\mathbf{M}$.

The knowledge of image correspondences enables scene reconstruction from images.

The concept of correspondence is a cornerstone of multiple-view vision. In this notes we assume *known correspondences*, and explore their use in geometric algorithms. Techniques for computing dense correspondences are surveyed in [45, 4].

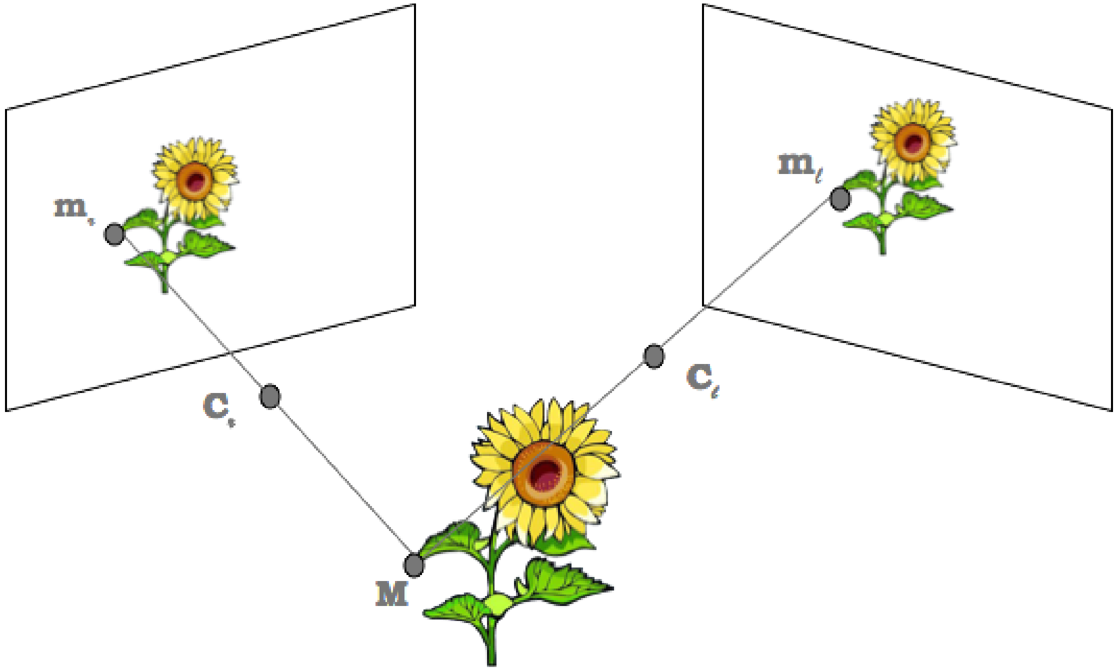


Fig. 3. Two perspective views of the same 3-D scene. $\mathbf{m}_\ell$ and $\mathbf{m}_r$ are corresponding points, as they are the projection of the same 3-D point, $\mathbf{M}$.

We will refer to the camera projection matrix of the left view as P_ℓ and of the right view as P_r. The 3-D point $\mathbf{M}$ is then imaged as (21) in the left view, and (22) in the right view:

$$\zeta_\ell \mathbf{m}_\ell = P_\ell \mathbf{M} \tag{21}$$

$$\zeta_r \mathbf{m}_r = P_r \mathbf{M}. \tag{22}$$

Geometrically, the position of the image point $\mathbf{m}_\ell$ in the left image plane I_ℓ can be found by drawing the optical ray through the left camera projection centre $\mathbf{C}_\ell$ and the scene point $\mathbf{M}$. The ray intersects the left image plane I_ℓ at $\mathbf{m}_\ell$.

Similarly, the optical ray connecting $\mathbf{C}_r$ and $\mathbf{M}$ intersects the right image plane I_r at $\mathbf{m}_r$.

The relationship between image points $\mathbf{m}_\ell$ and $\mathbf{m}_r$ is given by the epipolar geometry, described in Section 4.1.

4.1 Epipolar Geometry

The epipolar geometry describes the geometric relationship between two perspective views of the same 3-D scene.

The key finding, discussed below, is that *corresponding image points must lie on particular image lines*, which can be computed without information on the calibration of the cameras.

This implies that, given a point in one image, one can search the corresponding point in the other along a line and not in a 2-D region, a significant reduction in complexity.

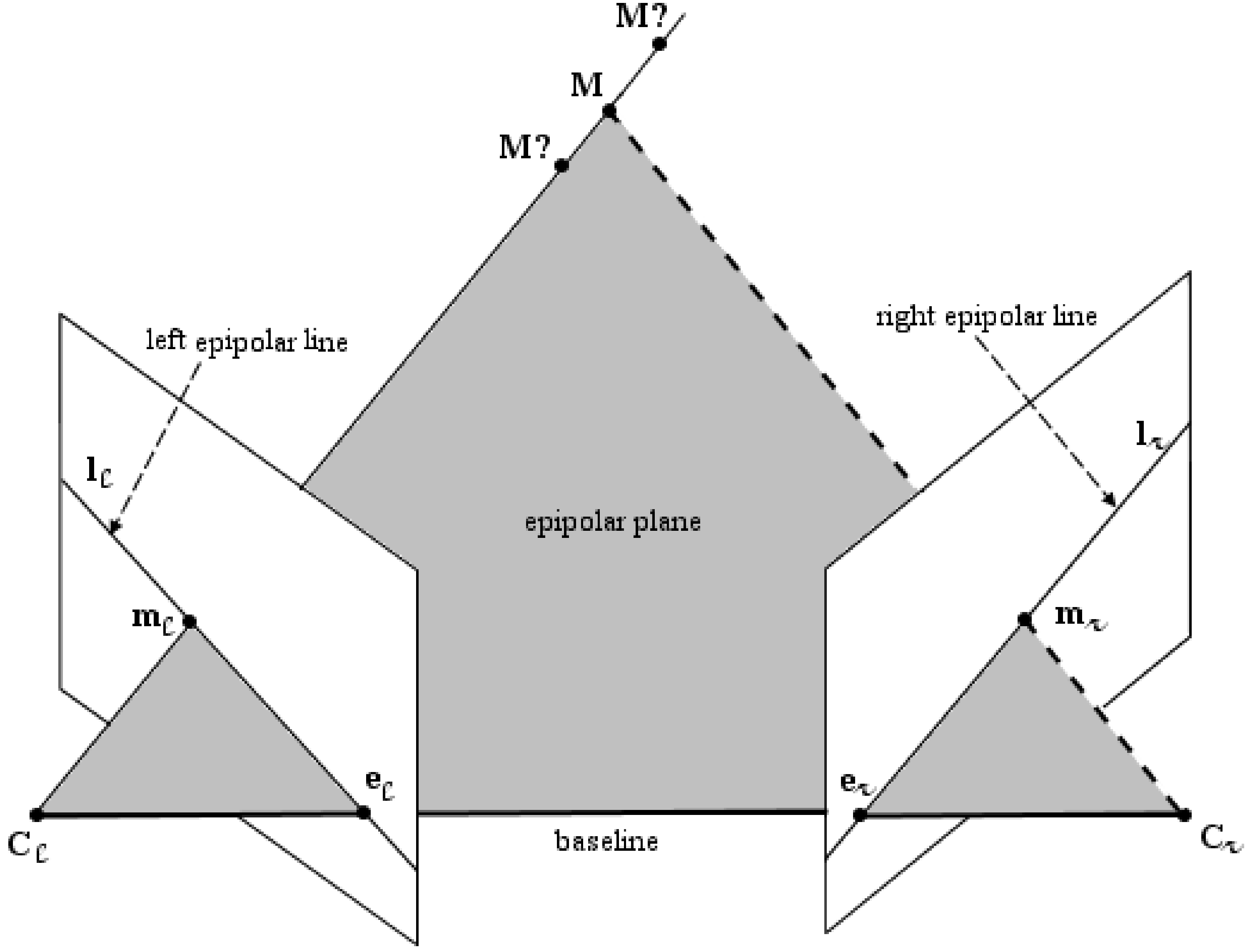


Fig. 4. The epipolar geometry and epipolar constraint.

Any 3-D point $\mathbf{M}$ and the camera projection centres $\mathbf{C}_\ell$ and $\mathbf{C}_r$ define a plane that is called *epipolar plane*.

The projections of the point $\mathbf{M}$, image points $\mathbf{m}_\ell$ and $\mathbf{m}_r$, also lie in the epipolar plane since they lie on the rays connecting the corresponding camera projection centre and point $\mathbf{M}$.

The conjugate epipolar lines, $\mathbf{l}_\ell$ and $\mathbf{l}_r$, are the intersections of the epipolar plane with the image planes. The line connecting the camera projection centres $(\mathbf{C}_\ell, \mathbf{C}_r)$ is called the *baseline*.

The baseline intersects each image plane in a point called *epipole*.

By construction, the left epipole $\mathbf{e}_\ell$ is the image of the right camera projection centre $\mathbf{C}_r$ in the left image plane. Similarly, the right epipole $\mathbf{e}_r$ is the image of the left camera projection centre $\mathbf{C}_\ell$ in the right image plane.

All epipolar lines in the left image go through $\mathbf{e}_\ell$ and all epipolar lines in the right image go through $\mathbf{e}_r$.

The epipolar constraint.

An epipolar plane is completely defined by the camera projection centres and one image point.

Therefore, given a point $\mathbf{m}_\ell$, one can determine the epipolar line in the right image on which the corresponding point, $\mathbf{m}_r$, must lie.

The equation of the epipolar line can be derived from the equation describing the optical ray. As we mentioned before, the right epipolar line corresponding to $\mathbf{m}_\ell$ geometrically represents the projection (Equation (8)) of the optical ray through $\mathbf{m}_\ell$ (Equation (14)) onto the right image plane:

$$\zeta_r \mathbf{m}_r = P_r \mathbf{M} = \underbrace{P_r \begin{pmatrix} -P_{\ell_{1:3}}^{-1} P_{\ell_4} \\ 1 \end{pmatrix}}_{\mathbf{e}_r} + \zeta_\ell P_r \begin{pmatrix} P_{\ell_{1:3}}^{-1} \mathbf{m}_\ell \\ 0 \end{pmatrix} \tag{23}$$

If we now simplify the above equation we obtain the description of the right epipolar line:

$$\zeta_r \mathbf{m}_r = \mathbf{e}_r + \zeta_\ell \underbrace{P_{r_{1:3}} P_{\ell_{1:3}}^{-1} \mathbf{m}_\ell}_{\mathbf{m}'_\ell} \tag{24}$$

This is the equation of a line through the right epipole $\mathbf{e}_r$ and the image point $\mathbf{m}'_\ell$ which represents the projection onto the right image plane of the point at infinity of the optical ray of $\mathbf{m}_\ell$.

The equation for the left epipolar line is obtained in a similar way.

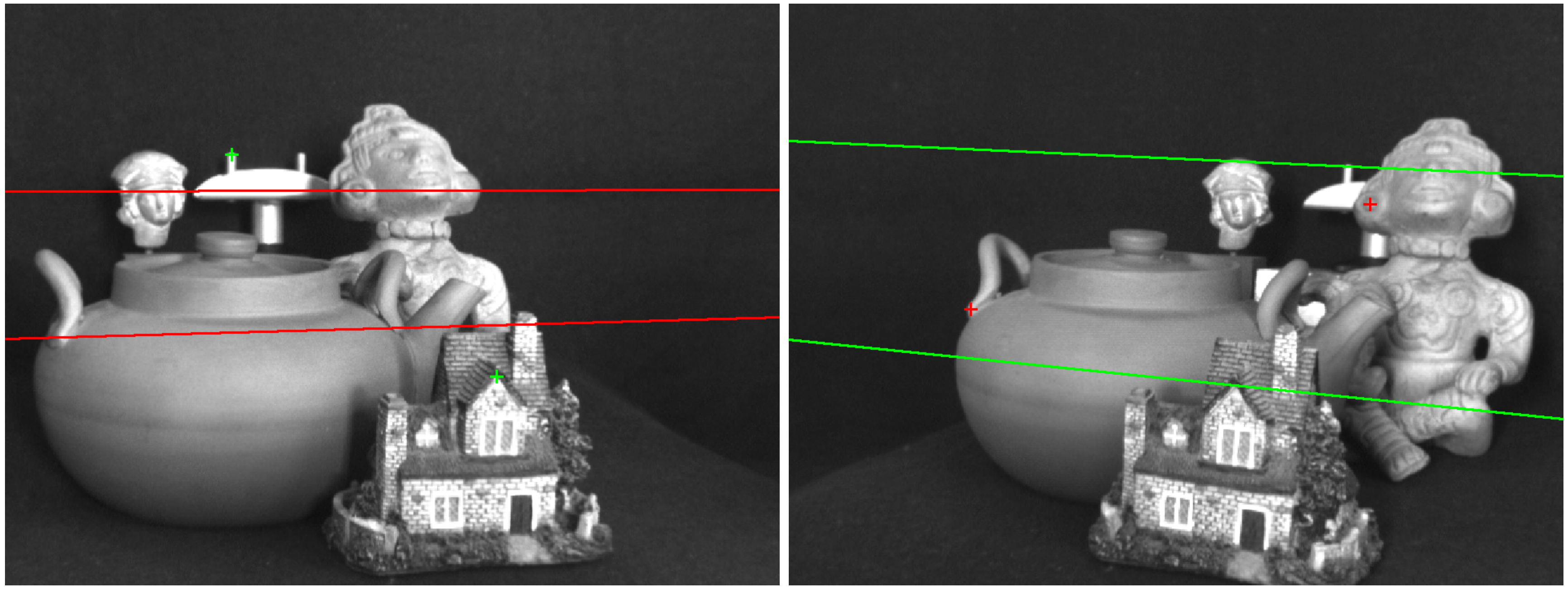


Fig. 5. Left and right images with epipolar lines.

The epipolar geometry can be described analytically in several ways, depending on the amount of the *a priori* knowledge about the stereo system. We can identify three general cases.

If both *intrinsic* and *extrinsic* camera parameters are known, we can describe the epipolar geometry in terms of the projection matrices (Equation (24)).

If only the *intrinsic* parameters are known, we work in normalised coordinates and the epipolar geometry is described by the *essential matrix*.

If neither intrinsic nor extrinsic parameters are known the epipolar geometry is described by the *fundamental matrix*.

4.1.1 The Essential Matrix E

If the intrinsic parameters are known, we can switch to *normalised coordinates*: $\mathbf{m} \leftarrow K^{-1}\mathbf{m}$ (please note that this change of notation will hold throughout this section).

Consider a pair of normalised cameras. Without loss of generality, we can fix the world reference frame onto the first camera, hence:

$$P_\ell = [I|0] \quad \text{and} \quad P_r = [R|\mathbf{t}]. \tag{25}$$

With this choice, the unknown extrinsic parameters have been made explicit.

If we substitute these two particular instances of the camera projection matrices in Equation (24), we get

$$\zeta_r \mathbf{m}_r = \mathbf{t} + \zeta_\ell R \mathbf{m}_\ell; \tag{26}$$

in other words, the point $\mathbf{m}_r$ lies on the line through the points $\mathbf{t}$ and $R\mathbf{m}_\ell$. In homogeneous coordinates, this can be written as follows: (14)

$$\mathbf{m}_r^T(\mathbf{t} \times R\mathbf{m}_\ell) = 0, \tag{27}$$

as the homogeneous line through two points is expressed as their cross product.

Similarly, a dot product of a point and a line is zero if the point lies on the line.

The cross product of two vectors can be written as a product of a skew-symmetric matrix and a vector. Equation (27) can therefore be equivalently written as

$$\mathbf{m}_r^T [\mathbf{t}]_\times R \mathbf{m}_\ell = 0, \tag{28}$$

where $[\mathbf{t}]_\times$ is the skew-symmetric matrix of the vector $\mathbf{t}$. Let us define the *essential matrix* E:

$$E \triangleq [\mathbf{t}]_\times R. \tag{29}$$

In summary, the relationship between the corresponding image points $\mathbf{m}_\ell$ and $\mathbf{m}_r$ in normalised coordinates is the bilinear form:

$$\mathbf{m}_r^T E \mathbf{m}_\ell = 0. \tag{30}$$

E encodes only information on the extrinsic camera parameters. It is singular, since $\det[\mathbf{t}]_\times = 0$. The essential matrix is a homogeneous quantity. It has only five degrees of freedom: a 3-D rotation and a 3-D translation direction.

4.1.2 The Fundamental Matrix F

The fundamental matrix can be derived in a similar way to the essential matrix. All camera parameters are assumed unknown; we write therefore a general version of Equation (25):

$$P_\ell = K_\ell[I|0] \quad \text{and} \quad P_r = K_r[R|\mathbf{t}]. \tag{31}$$

Inserting these two projection matrices into Equation (24), we get

$$\zeta_r \mathbf{m}_r = \mathbf{e}_r + \zeta_\ell K_r R K_\ell^{-1} \mathbf{m}_\ell \quad \text{with} \quad \mathbf{e}_r = K_r \mathbf{t}, \tag{32}$$

which states that point $\mathbf{m}_r$ lies on the line through $\mathbf{e}_r$ and $K_r R K_\ell^{-1} \mathbf{m}_\ell$. As in the case of the essential matrix, this can be written in homogeneous coordinates as:

$$\mathbf{m}_r^T [\mathbf{e}_r]_\times K_r R K_\ell^{-1} \mathbf{m}_\ell = 0. \tag{33}$$

The matrix

$$F = [\mathbf{e}_r]_\times K_r R K_\ell^{-1} \tag{34}$$

is the *fundamental matrix* F, giving the relationship between the corresponding image points in pixel coordinates.

Therefore, the bilinear form that links corresponding points writes:

$$\mathbf{m}_r^T F \mathbf{m}_\ell = 0. \tag{35}$$

F is the algebraic representation of the epipolar geometry in the least information case. It is a 3×3, rank-two homogeneous matrix. It has only seven degrees of freedom since it is defined up to a scale and its determinant is zero. Notice that F is completely defined by pixel correspondences only (the intrinsic parameters are not needed).

For any point $\mathbf{m}_\ell$ in the left image, the corresponding epipolar line $\mathbf{l}_r$ in the right image can be expressed as

$$\mathbf{l}_r = F\mathbf{m}_\ell. \tag{36}$$

Similarly, the epipolar line $\mathbf{l}_\ell$ in the left image for the point $\mathbf{m}_r$ in the right image can be expressed as

$$\mathbf{l}_\ell = F^T\mathbf{m}_r. \tag{37}$$

The left epipole $\mathbf{e}_\ell$ is the right null-vector of the fundamental matrix and the right epipole is the left null-vector of the fundamental matrix:

$$F\mathbf{e}_\ell = 0 \tag{38}$$
$$\mathbf{e}_r^T F = 0 \tag{39}$$

One can see from the derivation that the essential and fundamental matrices are related through the camera calibration matrices K_ℓ and K_r:

$$F = K_r^{-T} E K_\ell^{-1}. \tag{40}$$

Consider a camera pair. Using the fact that if F maps points in the left image to epipolar lines in the right image, then F^T maps points in the right image to epipolar lines in the left image, Equation (32) gives: [10]

$$\zeta_r F^T \mathbf{m}_r = \zeta_\ell (\mathbf{e}_\ell \times \mathbf{m}_\ell). \tag{41}$$

This is another way of writing the epipolar constraint: the epipolar line of $\mathbf{m}_r$ $(F^T\mathbf{m}_r)$ is the line containing its corresponding point $(\mathbf{m}_\ell)$ and the epipole in the left image $(\mathbf{e}_\ell)$.

4.1.3 Estimating F: the eight-point algorithm

If a number of point correspondences $\mathbf{m}_\ell^i \leftrightarrow \mathbf{m}_r^i$ is given, we can use Equation (35) to compute the unknown matrix F.

We need to convert Equation (35) from its bilinear form to a form that matches the null-space problem. To this end we use again the vec operator, as in the DLT algorithm:

$$\mathbf{m}_r^T F \mathbf{m}_\ell = 0 \iff \mathrm{vec}(\mathbf{m}_r^T F \mathbf{m}_\ell) = 0 \iff (\mathbf{m}_r^T \otimes \mathbf{m}_\ell^T)\,\mathrm{vec}(F) = 0.$$

Each point correspondence gives rise to one linear equation in the unknown entries of F. From a set of n point correspondences, we obtain a $n \times 9$ coefficient matrix A by stacking up one equation for each correspondence.

In general A will have rank 8 and the solution is the 1-dimensional right null-space of A.

The fundamental matrix F is computed by solving the resulting linear system of equations, for $n \geq 8$.

If the data are not exact and more than 8 points are used, the rank of A will be 9 and a least-squares solution is sought.

The least-squares solution for $\mathrm{vec}(F)$ is the singular vector corresponding to the smallest singular value of A.

This method does not explicitly enforce F to be singular, so it must be done *a posteriori*.

Replace F by $F^{'}$ such that $\det F^{'} = 0$, by forcing to zero the least singular value.

It can be shown that $F^{'}$ is the closest singular matrix to F in Frobenius norm.

Geometrically, the singularity constraint ensures that the epipolar lines meet in a common epipole.

4.2 Triangulation

Given the camera matrices P_ℓ and P_r, let $\mathbf{m}_\ell$ and $\mathbf{m}_r$ be two corresponding points satisfying the epipolar constraint $\mathbf{m}_r^T F \mathbf{m}_\ell = 0$. It follows that $\mathbf{m}_r$ lies on the epipolar line $F\mathbf{m}_\ell$ and so the two rays back-projected from image points $\mathbf{m}_\ell$ and $\mathbf{m}_r$ lie in a common epipolar plane. Since they lie in the same plane, they will intersect at some point. This point is the reconstructed 3-D scene point $\mathbf{M}$.

Analytically, the reconstructed 3-D point $\mathbf{M}$ can be found by solving for parameter ζ_ℓ or ζ_r in Eq. (24). Let us rewrite it as:

$$\mathbf{e}_r = \zeta_r \mathbf{m}_r - \zeta_\ell \mathbf{m}'_\ell \tag{42}$$

The depth ζ_r and ζ_ℓ are unknown. Both encode the position of $\mathbf{M}$ in space, as ζ_r is the depth of $\mathbf{M}$ wrt the right camera and ζ_ℓ is the depth of $\mathbf{M}$ wrt the left camera.

The three points $\mathbf{m}_r, \mathbf{e}_r$ and $\mathbf{m}'_\ell$ are known and are collinear, so we can solve for ζ_r using the following closed form expressions [44]:

$$\zeta_r = \frac{(\mathbf{e}_r \times \mathbf{m}'_\ell) \cdot (\mathbf{m}_r \times \mathbf{m}'_\ell)}{||\mathbf{m}_r \times \mathbf{m}'_\ell||^2} \tag{43}$$

The reconstructed point $\mathbf{M}$ can then be calculated by inserting the value ζ into Equation (14).

In reality, camera parameters and image locations are known only approximately. The back-projected rays therefore do not actually intersect in space. It can be shown, however, that Formula (43) solve Eq. (42) in a least squares sense [28].

Triangulation can be also cast as a null-space problem, starting from the general projection equation (8).$^{\textcircled{12}}$

Triangulation is addressed in more details in [2, 17, 14, 56].

4.3 Planes and collineations

When observing a plane, we obtain an interesting specialization of the epipolar geometry of two views.

First, let us establish that the map between a world plane and its perspective image is a collineation of $\mathbb{P}_2$. The easiest way to see it is to choose the world coordinate system such that the plane has equation $z = 0$.

Expanding the projection equation gives:

$$\zeta \begin{bmatrix} u \\ v \\ 1 \end{bmatrix} = P \begin{bmatrix} x \\ y \\ 0 \\ 1 \end{bmatrix} = [P_1|P_2|P_4] \begin{bmatrix} x \\ y \\ 1 \end{bmatrix}. \tag{44}$$

Points are mapped from the world plane to the image plane with a 3×3 (non-singular) matrix, which represents a collineation of $\mathbb{P}_2$.

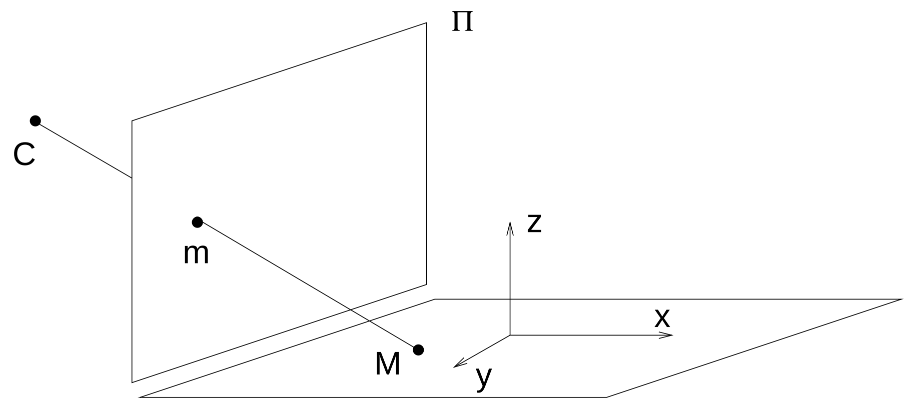


Fig. 6. The map between a world plane Π and a perspective image is a collineation.

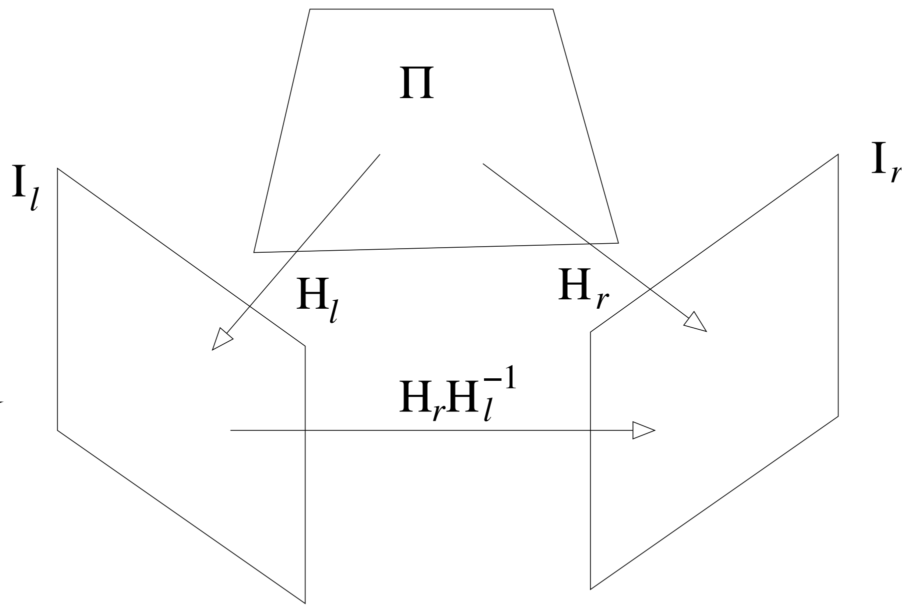


Fig. 7. The plane Π induces a collineration between two views.

Next, we prove that: images of points on a plane are related to corresponding image points in a second view by a *collineation* (or homography) of $\mathbb{P}_2$.

We have one collineation from Π to the left image plane, and another collineation from Π to the right image plane. By composing the inverse of the first with the second, we define a collineation from the image plane of the left camera to the image plane of the right camera.

The plane Π *induces* a collineation H_Π between the views, which transfers points from one view to the other:

$$\mathbf{m}_r \simeq H_\Pi \mathbf{m}_\ell \quad \text{if } \mathbf{M} \in \Pi. \tag{45}$$

where H_Π is a 3×3 non-singular matrix.

Even though a collineation of $\mathbb{P}_2$ depends upon eight parameters, there is no contradiction with the fact that a plane depends upon three parameters. Indeed, the collineation induced by a plane must be compatible with the epipolar geometry, i.e.:

$$(H_\Pi \mathbf{m}_\ell)^T F \mathbf{m}_\ell = 0 \tag{46}$$

for all points $\mathbf{m}_\ell$. This implies that the matrix $H_\Pi^T F$ is antisymmetric:

$$H_\Pi^T F + F^T H_\Pi = \mathbf{0} \tag{47}$$

and this imposes six homogeneous constraints on H_Π.

A collineation H that satisfies Eq. (47) is said to be *compatible* with F.

A collineation H is compatible with F if and only if [03]

$$F \simeq [\mathbf{e}_r]_\times H. \tag{48}$$

From the latter – provided that Π does not contain $\mathbf{C}_r$ – follows that: [04]

$$H_\Pi \mathbf{e}_\ell \simeq \mathbf{e}_r \tag{49}$$

4.3.1 Homography induced by a plane

If the 3-D point $\mathbf{M}$ lies on a plane Π with equation $\mathbf{n}^T\mathbf{M} = d$, Eq. (32) can be specialized, obtaining (after elaborating): [05]

$$\frac{\zeta_r}{\zeta_\ell}\mathbf{m}_r = K_r\left(R + \frac{\mathbf{t}\ \mathbf{n}^T}{d}\right)K_\ell^{-1}\mathbf{m}_\ell. \tag{50}$$

Therefore, the collineation induced by Π is given by:

$$H_\Pi = K_r\left(R + \frac{\mathbf{t}\ \mathbf{n}^T}{d}\right)K_\ell^{-1} \tag{51}$$

This is a three-parameter family of collineations, parameterized by $\mathbf{n}/d$.

4.3.2 Infinite homography

The infinite homography H_∞ is the collineation induced by the plane at infinity; it maps vanishing points to vanishing points (a vanishing point is where all the lines that shares the same direction meet).

It can be derived by letting $d \rightarrow \infty$ in (50), thereby obtaining:

$$H_\infty = K_r R K_\ell^{-1} \tag{52}$$

The infinity homography does not depend on the translation between views.

In other terms, the vanishing points are fixed under camera translation.

4.3.3 Plane induced parallax

In general, when points are not on the plane, the homography induced by a plane generates a virtual parallax. This gives rise to an alternative representation of the epipolar geometry and scene structure [47].

First, let us rewrite Eq. (32), which links two general conjugate points, as:

$$\frac{\zeta_r}{\zeta_\ell}\mathbf{m}_r = H_\infty \mathbf{m}_\ell + \frac{1}{\zeta_\ell}\mathbf{e}_r, \tag{53}$$

The mapping from one point to its conjugate can be seen as composed by a transfer with the infinity homography $(H_\infty \mathbf{m}_\ell)$ plus a parallax correction term $(\frac{1}{\zeta_\ell}\mathbf{e}_r)$.

Note that if $\mathbf{t} = \mathbf{0}$, then the parallax vanishes. Thus H_∞ not only relates points at infinity when the camera describes a general motion, but it also relates image points of any depth if the camera rotates about its centre.

We want to generalize this equation to any plane. To this end we substitute ⓪⑥

$$H_\infty = H_\Pi - K_r \left(\frac{\mathbf{t}\ \mathbf{n}^T}{d} \right) K_\ell^{-1} \tag{54}$$

into Eq. (53), obtaining

$$\frac{\zeta_r}{\zeta_\ell} \mathbf{m}_r = H_\Pi \mathbf{m}_\ell + \gamma \mathbf{e}_r \tag{55}$$

with $\gamma = \left(\frac{a}{d\ \zeta_\ell} \right)$, where a is the distance of $\mathbf{M}$ to the plane Π.

When $\mathbf{M}$ is on the 3-D plane Π, then $\mathbf{m}_r \simeq H_\Pi \mathbf{m}_\ell$. Otherwise there is a residual displacement, called *parallax*, which is proportional to γ and oriented along the epipolar line.

The magnitude parallax depends only on the left view and the plane. It does not depend on the parameters of the right view.

From Eq. (55) we derive $\mathbf{m}_r^T(\mathbf{e}_r \times H_\Pi \mathbf{m}_\ell) = 0$, hence

$$F \simeq [\mathbf{e}_r]_\times H_\Pi \tag{56}$$

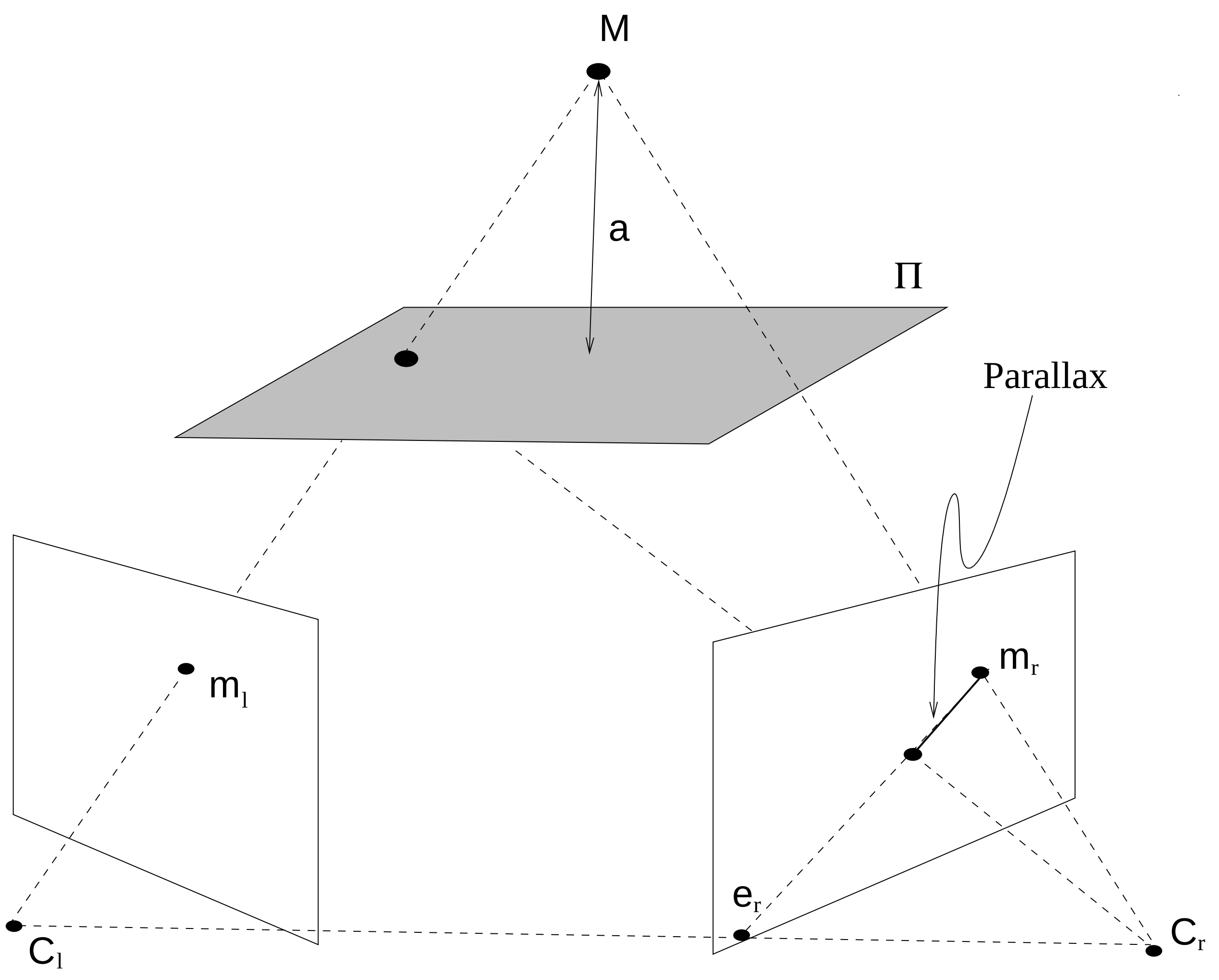


Fig. 8. Plane induced parallax.

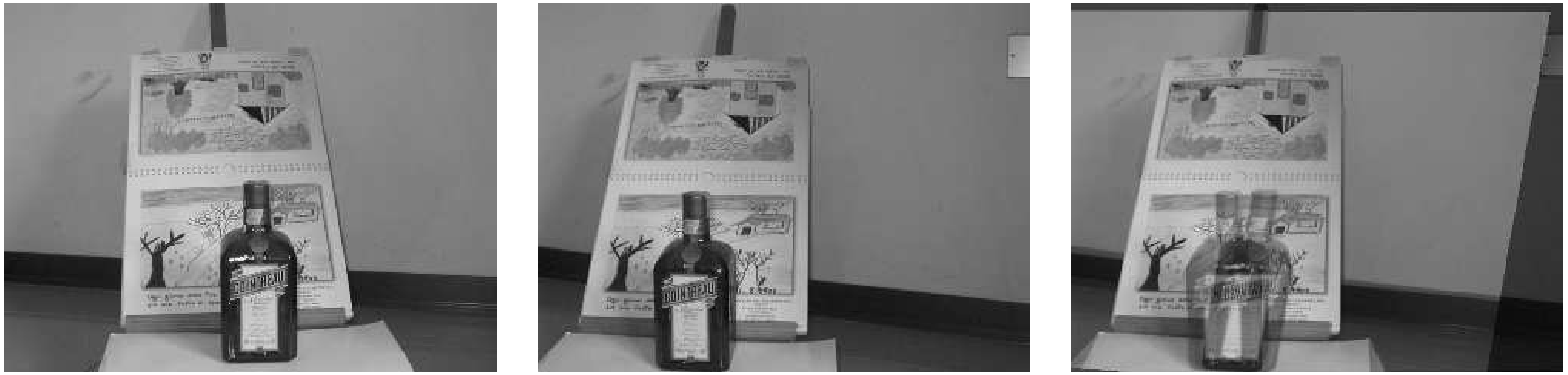


Fig. 9. Left and right images. The leftmost image is a superposition of the warped left image and the right image. The reference plane exactly coincide. However, points off the plane (such as the bottle) do not coincide.

4.3.4 Estimating H

A number of point correspondences $\mathbf{m}_r^i \leftrightarrow \mathbf{m}_\ell^i$ is given, and we are required to find an homography matrix H such that

$$\mathbf{m}_r^i \simeq H\mathbf{m}_\ell^i \quad \text{for all } i \tag{57}$$

The equation (we drop the index i for simplicity) can be rewritten in terms of the cross product as

$$\mathbf{m}_r \times H\mathbf{m}_\ell = \mathbf{0} \tag{58}$$

As we did before, we exploit the properties of the Kronecker product and the vec operator to transform this into a null-space problem and then derive a linear solution:

$$\mathbf{m}_r \times H\mathbf{m}_\ell = \mathbf{0} \iff [\mathbf{m}_r]_\times H\mathbf{m}_\ell = \mathbf{0} \iff \mathrm{vec}([\mathbf{m}_r]_\times H\mathbf{m}_\ell) = \mathbf{0}$$
$$\iff (\mathbf{m}_\ell^T \otimes [\mathbf{m}_r]_\times)\,\mathrm{vec}(H) = \mathbf{0} \iff ([\mathbf{m}_r]_\times \otimes \mathbf{m}_\ell^T)\,\mathrm{vec}(H^T) = \mathbf{0}$$

After expanding the coefficient matrix, we obtain

$$\begin{bmatrix} \mathbf{0}^T & -\mathbf{m}_\ell^T & v\mathbf{m}_\ell^T \\ \mathbf{m}_\ell^T & \mathbf{0}^T & -u\mathbf{m}_\ell^T \\ -v\mathbf{m}_\ell^T & u\mathbf{m}_\ell^T & \mathbf{0}^T \end{bmatrix} \mathrm{vec}(H^T) = \mathbf{0} \tag{59}$$

Although there are three equations, only two of them are linearly independent: we can write the third row (e.g.) as a linear combination of the first two.

From a set of n point correspondences, we obtain a $2n \times 9$ coefficient matrix A by stacking up two equations for each correspondence.

In general A will have rank 8 and the solution is the 1-dimensional right null-space of A.

The projection matrix H is computed by solving the resulting linear system of equations, for $n \geq 4$.

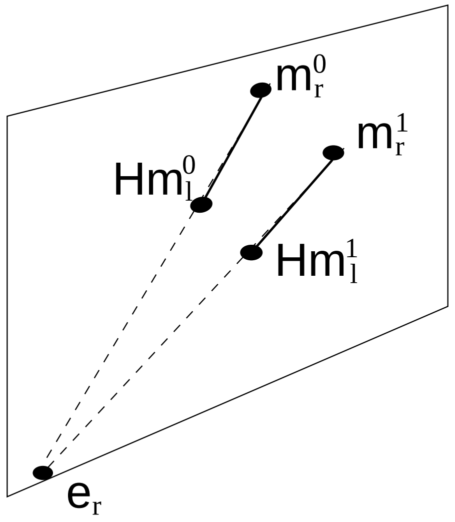
If the data are not exact, and more than 4 points are used, the rank of A will be 9 and a least-squares solution is sought.

The least-squares solution for $\mathrm{vec}(H^T)$ is the singular vector corresponding to the smallest singular value of A.

This is another incarnation of the DLT algorithm.

4.3.5 Estimating the epipole

The epipole can be located [52] given the homography H_Π between two views and two off-plane conjugate pairs $\mathbf{m}_\ell^0 \leftrightarrow \mathbf{m}_r^0$ and $\mathbf{m}_\ell^1 \leftrightarrow \mathbf{m}_r^1$.



m_r^0
m_r^1
Hm_l^0
Hm_l^1
e_r

Following simple geometric consideration, the epipole is computed as the intersection between the line containing $H_\Pi \mathbf{m}_\ell^0, \mathbf{m}_r^0$ and the line containing $H_\Pi \mathbf{m}_\ell^1, \mathbf{m}_r^1$:

$$\mathbf{e}_r \simeq (H_\Pi \mathbf{m}_\ell^0 \times \mathbf{m}_r^0) \times (H_\Pi \mathbf{m}_\ell^1 \times \mathbf{m}_r^1) \tag{60}$$

In the projective plane, the line determined by two points is given by their cross product, as well as the point determined by two lines.

4.3.6 Estimating the parallax

We are required to compute the magnitude of the parallax γ for a point $\mathbf{m}_\ell$ given its corresponding point $\mathbf{m}_r$, the homography H_Π between the two views and the epipole. To this end we rewrite (55) as:

$$H_\Pi \mathbf{m}_\ell = -\gamma \mathbf{e}_r + \frac{\zeta_r}{\zeta_\ell}\mathbf{m}_r \tag{61}$$

and, given that points $\mathbf{e}_r$, $\mathbf{m}_r$ and $H_\Pi \mathbf{m}_\ell$ are collinear, we solve for γ using:

$$\gamma = \frac{(\mathbf{H}_\Pi \mathbf{m}_\ell \times \mathbf{m}_r)^T (\mathbf{m}_r \times \mathbf{e}_r)}{||\mathbf{m}_r \times \mathbf{e}_r||^2} \tag{62}$$

Please note that the epipole and the homography can be computed from images only up to an unknown scale factor. It follows that the magnitude of the parallax as well is known only up to a scale factor.

4.3.7 Applications

Mosaics. Image mosaicing is the automatic alignment (or registration) of multiple images into larger aggregates [50]. There are two types of mosaics. In both cases, it turns out that images are related by homographies, as we discussed previously.

Planar mosaic: result from the registration of different views of a planar scene.

Panoramic mosaic result from the registration of views taken by a camera rotating around its optical centre (typ. panning). In some cases, in order to cope with large rotations (> 180 deg), the images are converted to cylindrical coordinates.

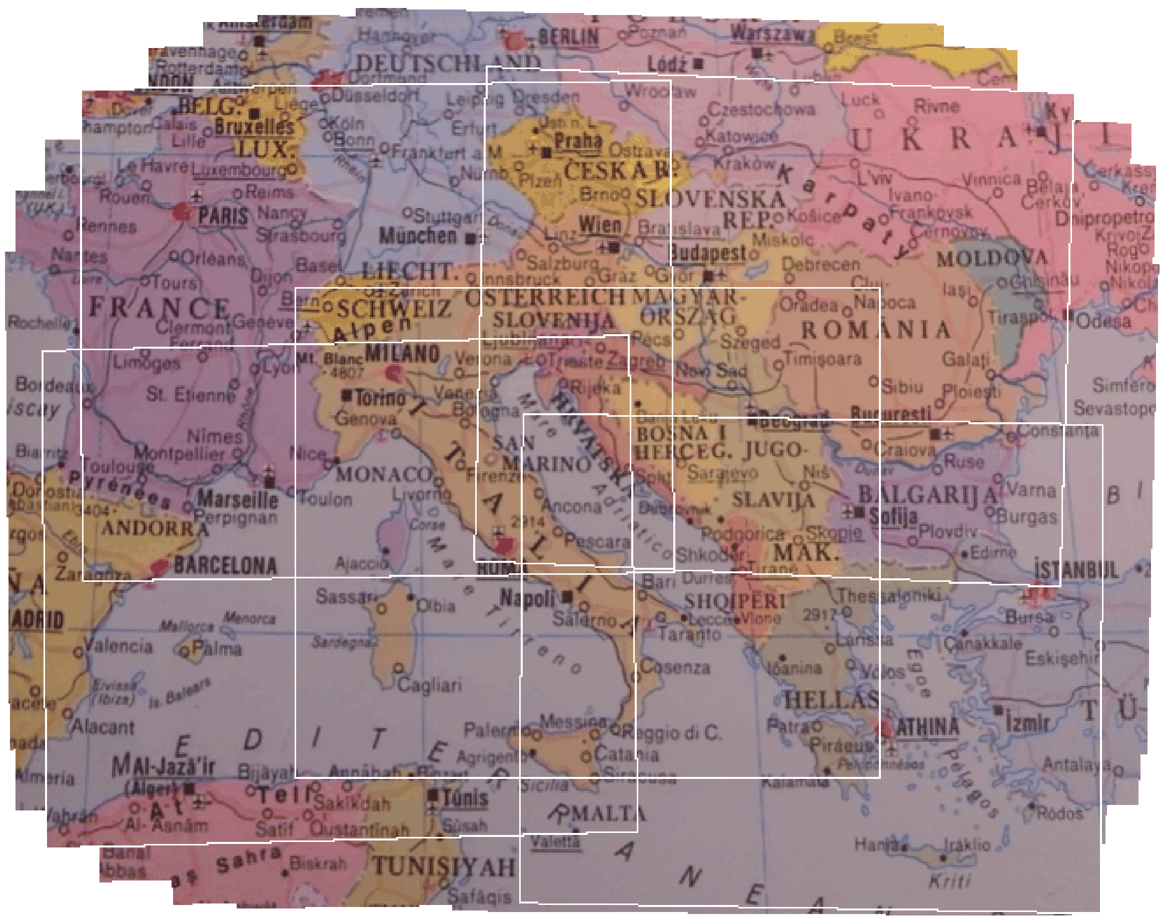


Fig. 10. Planar mosaic with components location shown as white outlines.

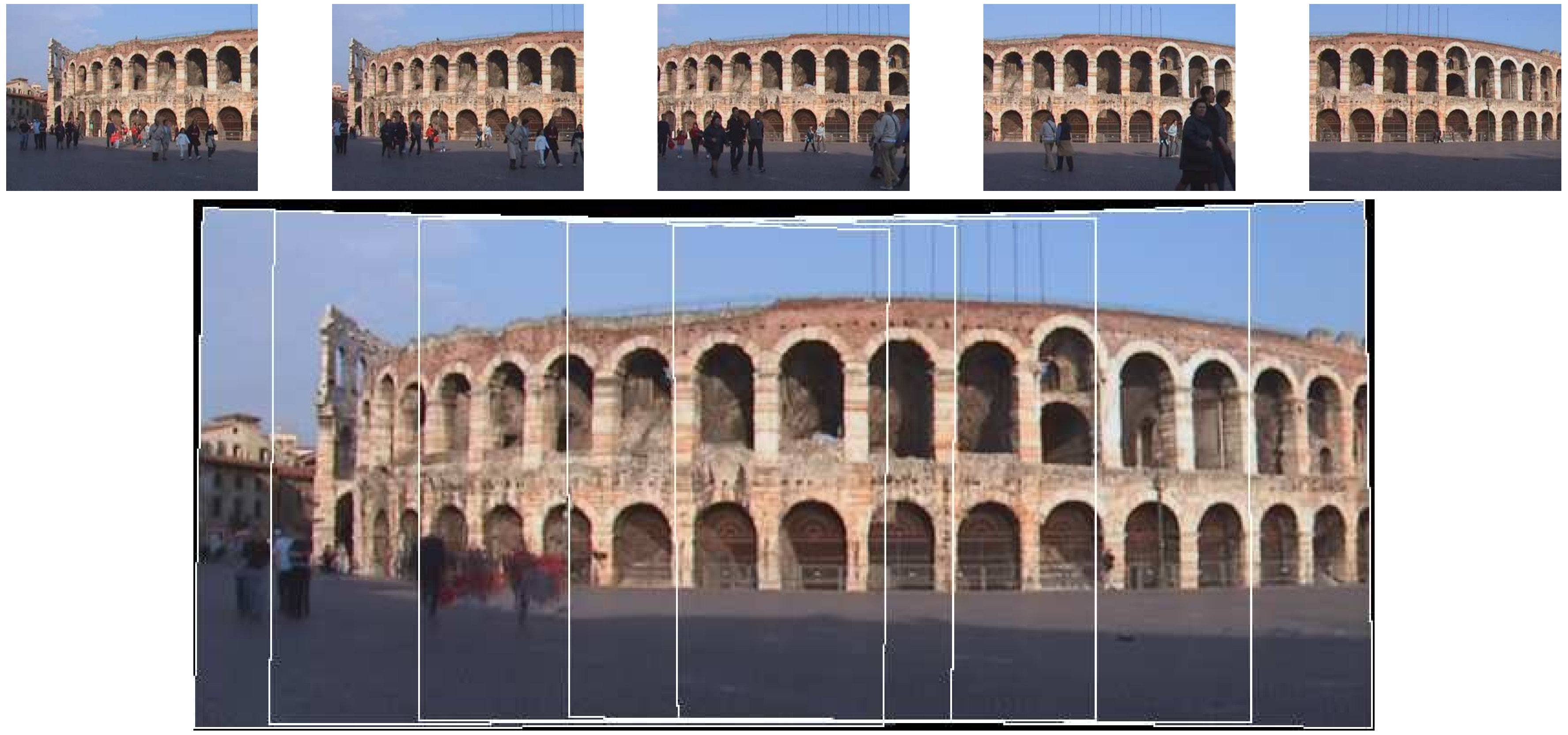


Fig. 11. Selected frames from "Arena" sequence (top) and panoramic mosaic (bottom). Components location shown as white outlines.

Orthogonal rectification. The map between a world plane and its perspective image is an homography. The world-plane to image-plane homography is fully defined by four points of which we know the relative position in the world plane. Once this homography is determined, the image can be back projected (warped) onto the world plane. This is equivalent to synthesize an image as taken from a fronto-parallel view of the plane. This is known as *orthogonal rectification* [31] of a perspective image.

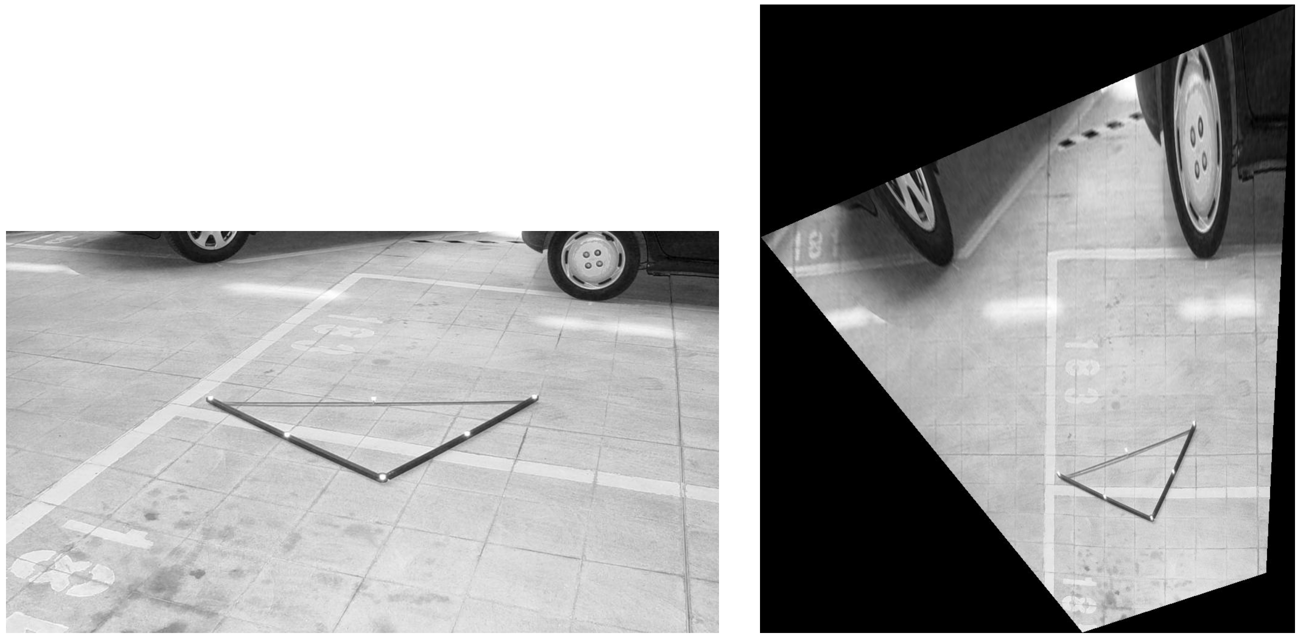


Fig. 12. A perspective image and a ortho-rectified image of the floor plane

4.4 3-D Reconstruction

What can be reconstructed depends on what is known about the scene and the stereo system. We can identify three cases.

(i) *If both the intrinsic and extrinsic camera parameters are known*, we can solve the reconstruction problem unambiguously by triangulation.

(ii) *If only the intrinsic parameters are known*, we can estimate the extrinsic parameters and solve the reconstruction problem up to an unknown scale factor. In other words, R can be estimated completely, and $\mathbf{t}$ up to a scale factor.

(iii) *If neither intrinsic nor extrinsic parameters are known*, i.e., the only information available are pixel correspondences, we can still solve the reconstruction problem but only up to an unknown, global projective transformation of the world.

4.4.1 Reconstruction up to a Similarity

If only intrinsics are known (plus point correspondences between images), the epipolar geometry is described by the essential matrix (Section 4.1.1). We will see that, starting from the essential matrix, only a reconstruction up to a similarity transformation (rigid+ uniform scale) can be achieved. Such a reconstruction is referred to as “Euclidean”.

Unlike the fundamental matrix, the only property of which is to have rank two, the essential matrix is characterised by the following theorem [24].

Theorem 4.1 *A real 3×3 matrix E can be factorised as product of a nonzero skew-symmetric matrix and a rotation matrix if and only if E has two identical singular values and a zero singular value.*

The theorem has a constructive proof (see [15]) that describes how E can be factorised into rotation and translation using its Singular Value Decomposition (SVD).
$\textcircled{07}$

The rotation R and translation $\mathbf{t}$ are then used to instantiate a camera pair as in Equation (25), and this camera pair is subsequently used to reconstruct the structure of the scene by triangulation.

The rigid displacement ambiguity arises from the arbitrary choice of the world reference frame, whereas the scale ambiguity derives from the fact that $\mathbf{t}$ can be scaled arbitrarily in Equation (29) and one would get the same essential matrix (E is defined up to a scale factor).

Therefore translation can be recovered from E only up to an unknown scale factor which is inherited by the reconstruction. This is also known as *depth-speed ambiguity*. $\textcircled{15}$

4.4.2 Reconstruction up to a Projective Transformation

Suppose that a set of image correspondences $\mathbf{m}_\ell^i \leftrightarrow \mathbf{m}_r^i$ are given. It is assumed that these correspondences come from a set of 3-D points $\mathbf{M}_i$, which are unknown. Similarly, the position, orientation and calibration of the cameras are not known. This situation is usually referred to as *weak calibration*, and we will see that the scene may be reconstructed up to a projective ambiguity, which may be reduced if additional information is supplied on the cameras or the scene.

The reconstruction task is to find the camera matrices P_ℓ and P_r, as well as the 3-D points $\mathbf{M}_i$ such that

$$\mathbf{m}_\ell^i = P_\ell \mathbf{M}^i \quad \text{and} \quad \mathbf{m}_r^i = P_r \mathbf{M}^i, \quad \forall i \tag{63}$$

If T is any 4×4 invertible matrix, representing a collineation of $\mathbb{P}_3$, then replacing points $\mathbf{M}^i$ by $T\mathbf{M}^i$ and matrices P_ℓ and P_r by $P_\ell T^{-1}$ and $P_r T^{-1}$ does not change the image points $\mathbf{m}_\ell^i$. This shows that, if nothing is known but the image points, the structure $\mathbf{M}^i$ and the cameras can be determined only up to a projective transformation.

The procedure for reconstruction follows the previous one. Given the weak calibration assumption, the fundamental matrix can be computed (using the algorithm described in Section 4.1.2), and from a (non-unique) factorization of F of the form

$$F = [\mathbf{e}_r]_\times A \tag{64}$$

two camera matrices P_ℓ and P_r:

$$P_\ell = [I|\mathbf{0}] \quad \text{and} \quad P_r = [A|\mathbf{e}_r], \tag{65}$$

can be created in such a way that they yield the fundamental matrix F, as can be easily verified. The position in space of the points $\mathbf{M}^i$ is then obtained by triangulation.

The matrix A in the factorization of F can be set to $A = -[\mathbf{e}_r]_\times F$ (this is called the *epipolar projection matrix* [33]). 08

Unlike the essential matrix, F does not admit a unique factorization, whence the projective ambiguity follows.

Indeed, for any A satisfying Equation (64), also $A + \mathbf{e}_r \mathbf{x}^T$ for any vector $\mathbf{x}$, satisfies Equation (64).

More in general, any homography induced by a plane can be taken as the A matrix (cfr. Eq. (48)).

5 Multiple View Geometry

In this section we study the relationship that links three or more views of the same 3-D scene, known in the three-view case as *trifocal geometry*.

This geometry can be described in terms of fundamental matrices linking pairs of cameras, but in the three-view case a more compact and elegant description is provided by a special algebraic operator, the *trifocal tensor*.

We also discover that three views are all we need, in the sense that additional views do not allow us to compute anything we could not already compute (Section 5.4).

5.1 Trifocal geometry

Denoting the cameras by $1, 2, 3$, there are now three fundamental matrices, $F_{1,2}$, $F_{1,3}$, $F_{2,3}$, and six epipoles, $\mathbf{e}_{i,j}$, as in Figure 13. The three fundamental matrices describe completely the trifocal geometry [8].

The plane containing the three optical centres is called the *trifocal plane*. It intersects each image plane along a line which contains the two epipoles.

Writing Eq. (41) for each camera pair (taking the centre of the third camera as the point $\mathbf{M}$) results in three epipolar constraints:

$$F_{3,1}\mathbf{e}_{3,2} \simeq \mathbf{e}_{1,3} \times \mathbf{e}_{1,2} \quad F_{1,2}\mathbf{e}_{1,3} \simeq \mathbf{e}_{2,1} \times \mathbf{e}_{2,3} \quad F_{2,3}\mathbf{e}_{2,1} \simeq \mathbf{e}_{3,2} \times \mathbf{e}_{3,1} \qquad (66)$$

Three fundamental matrices include 21 free parameters, less the 3 constraints above; the trifocal geometry is therefore determined by 18 parameters.

This description of the trifocal geometry fails when the three cameras are collinear, and the trifocal plane reduces to a line.

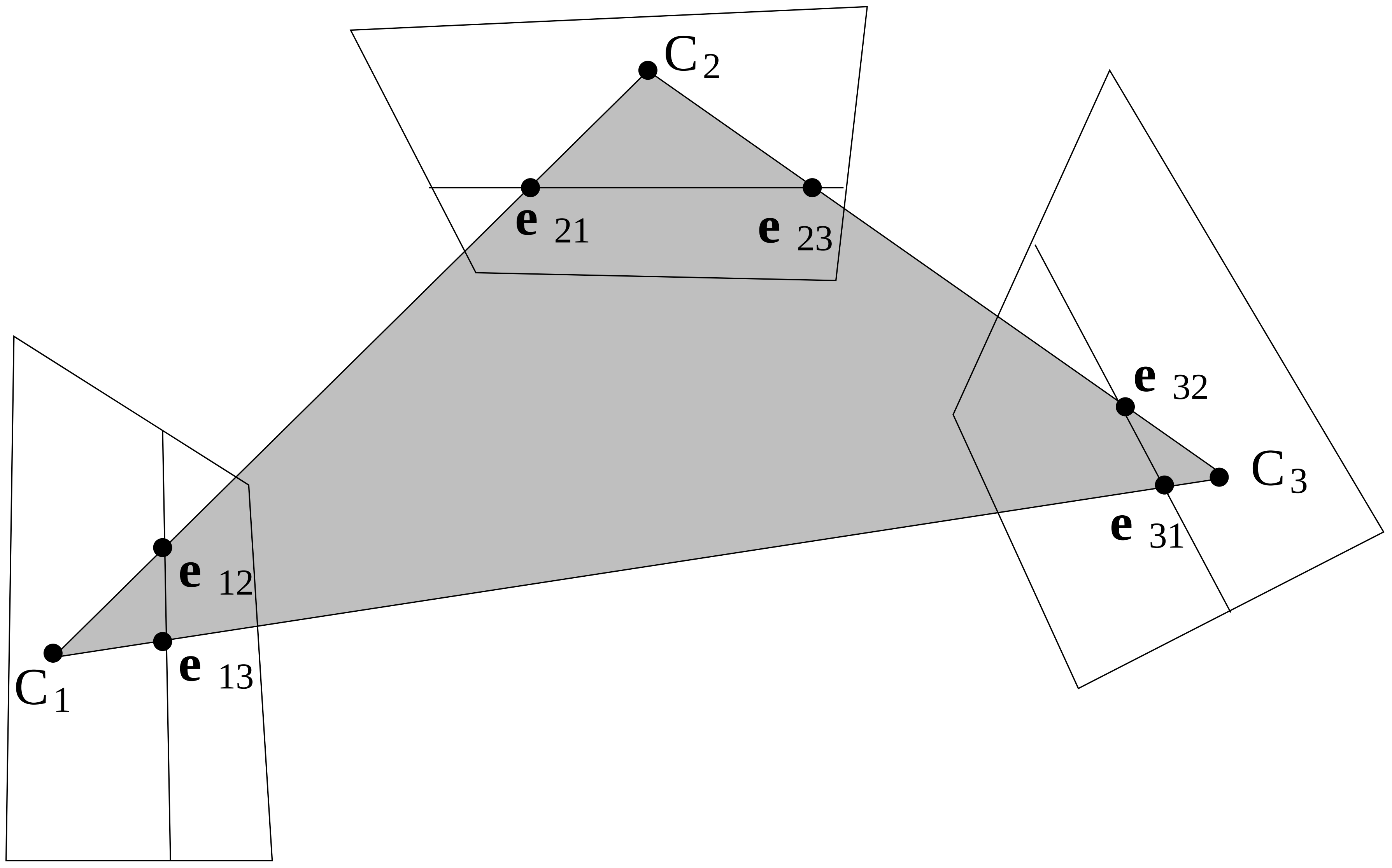


Fig. 13. Trifocal geometry.

Point transfer

If the trifocal geometry is known, given two conjugate points $\mathbf{m}_1$ and $\mathbf{m}_2$ in view 1 and 2 respectively, the position of the conjugate point $\mathbf{m}_3$ in view 3 is completely determined (Figure 14).

This allows for *point transfer* or prediction. Indeed, $\mathbf{m}_3$ belongs simultaneously to the epipolar line of $\mathbf{m}_1$ and to the epipolar line of $\mathbf{m}_2$, hence:

$$\mathbf{m}_3 \simeq F_{1,3}\mathbf{m}_1 \times F_{2,3}\mathbf{m}_2 \tag{67}$$

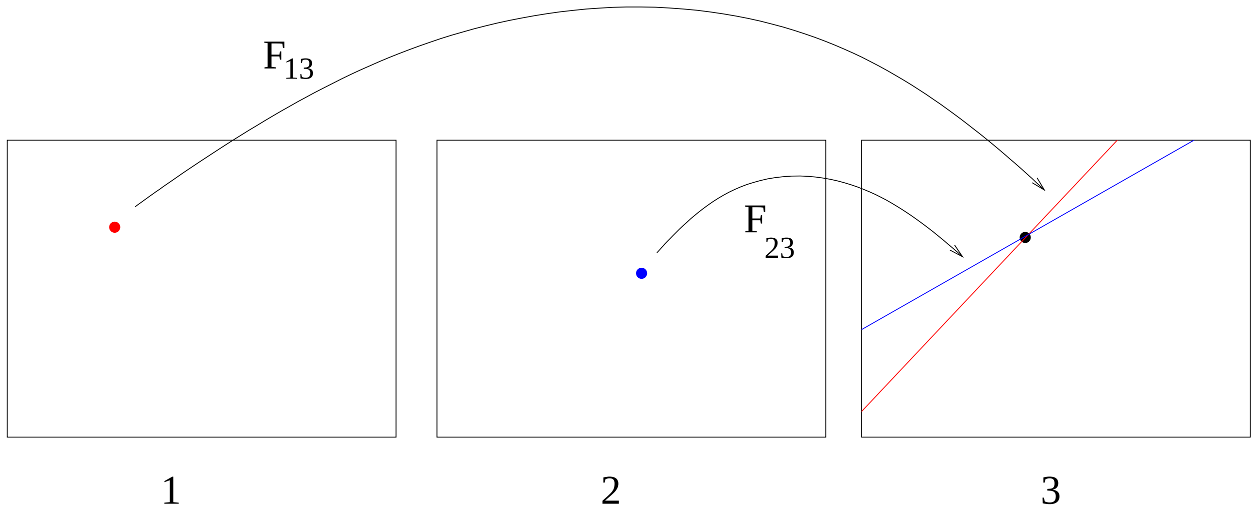


Fig. 14. Point transfer using epipolar constraints between three views.

View synthesis [29, 1, 3], exploit the trifocal geometry to generate novel (synthetic) images starting from two reference views. A related topic is *image-based rendering* [30, 57, 26].

Epipolar transfer fails when the three optical rays are coplanar, because the epipolar lines are coincident. This happens:

- if the 3-D point is on the trifocal plane;
- if the three cameras centres are collinear (independently of the position of 3-D point).

These deficiencies motivate the introduction of an independent trifocal constraint.

In addition, by generalizing the case of two views, one might conjecture that the trifocal geometry should be represented by a trilinear form in the coordinates of three conjugate points.

5.2 The trifocal constraint

Consider a point $\mathbf{M}$ in space projecting to $\mathbf{m}_1$, $\mathbf{m}_2$ and $\mathbf{m}_3$ in three cameras

$$P_1 = [I|0], \quad P_2 = [A_2|\mathbf{e}_{2,1}], \quad \text{and} \quad P_3 = [A_3|\mathbf{e}_{3,1}]. \tag{68}$$

Let us write the epipolar line of $\mathbf{m}_1$ in the other two views (using Equation (24)):

$$\zeta_2\mathbf{m}_2 = \mathbf{e}_{2,1} + \zeta_1 A_2 \mathbf{m}_1 \tag{69}$$

$$\zeta_3\mathbf{m}_3 = \mathbf{e}_{3,1} + \zeta_1 A_3 \mathbf{m}_1. \tag{70}$$

Consider a line through $\mathbf{m}_2$, represented by $\mathbf{s}_2$; we have $\mathbf{s}_2^T\mathbf{m}_2 = 0$, that substituted in (69) gives:

$$0 = \mathbf{s}_2^T\mathbf{e}_{2,1} + \zeta_1\mathbf{s}_2^T A_2\mathbf{m}_1 \tag{71}$$

Likewise, for a line through $\mathbf{m}_3$ represented by $\mathbf{s}_3$ we can write:

$$0 = \mathbf{s}_3^T\mathbf{e}_{3,1} + \zeta_1\mathbf{s}_3^T A_3\mathbf{m}_1 \tag{72}$$

After eliminating ζ_1 from Equation (71) and (72) we obtain:

$$0 = (\mathbf{s}_2^T \mathbf{e}_{2,1})(\mathbf{s}_3^T A_3 \mathbf{m}_1) - (\mathbf{s}_3^T \mathbf{e}_{3,1})(\mathbf{s}_2^T A_2 \mathbf{m}_1) \tag{73}$$

and after some re-writing:

$$0 = \mathbf{s}_2^T \left(\mathbf{e}_{2,1} \mathbf{m}_1^T A_3^T - A_2 \mathbf{m}_1 \mathbf{e}_{3,1}^T \right) \mathbf{s}_3 \tag{74}$$

This is the *fundamental trifocal constraint*, that links (via a trilinear form) $\mathbf{m}_1$, $\mathbf{s}_2$ (any line through $\mathbf{m}_2$) and $\mathbf{s}_3$ (any line through $\mathbf{m}_3$).

Geometrically, the trifocal constraint imposes that the optical rays of $\mathbf{m}_1$ intersect the 3-D line L that projects onto $\mathbf{s}_2$ in the second image and $\mathbf{s}_3$ in the third image.

Please note that given two (arbitrary) lines in two images, they can be always seen as the image of a 3-D line L, because two planes always define a line, in projective space (this is why there is no such thing as the epipolar constraint between lines.)

The trifocal constraint represents the trifocal geometry (nearly) without singularities. It only fails is when the cameras are collinear *and* the 3-D point is on the same line.

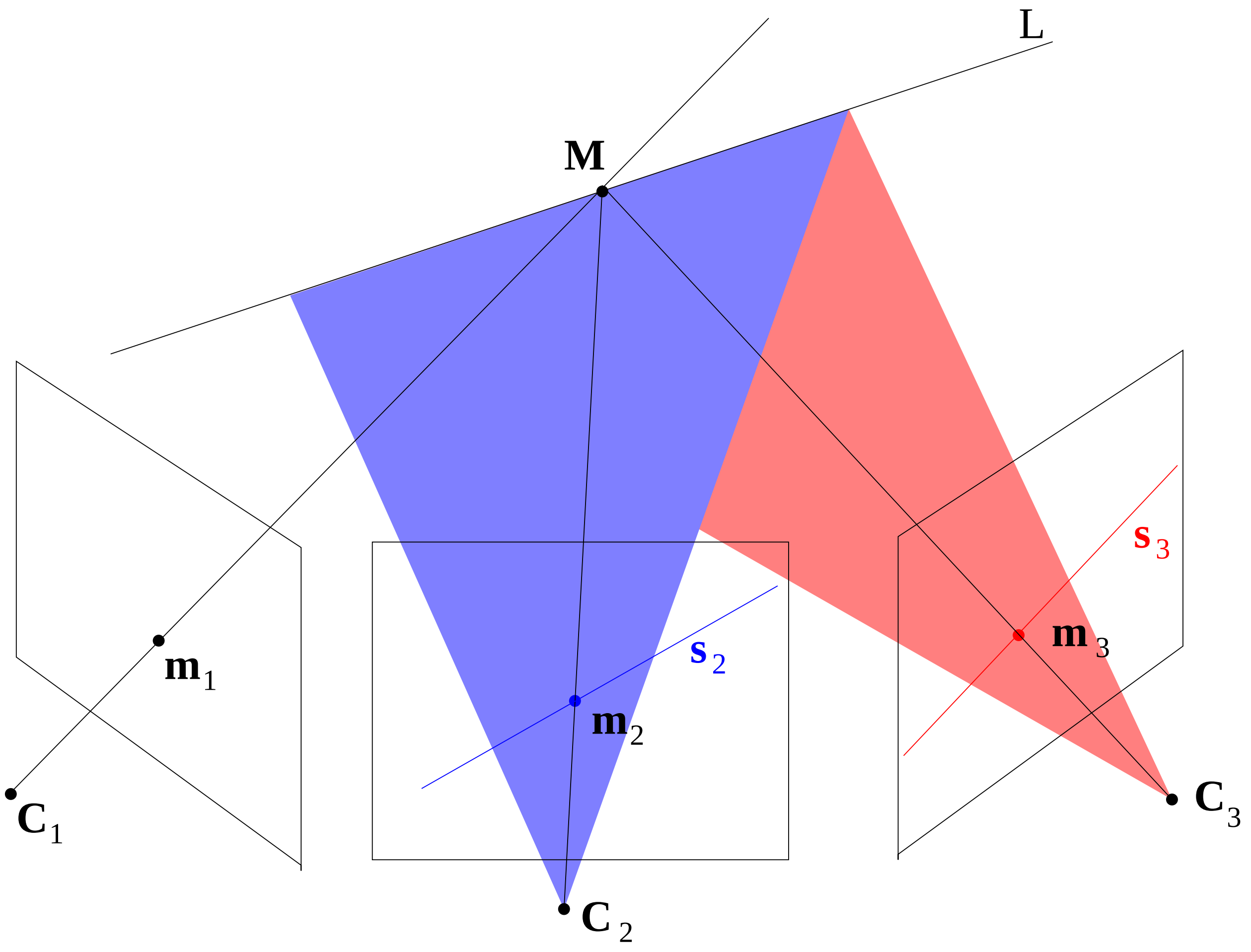


Fig. 15. Two arbitrary lines $\mathbf{s}_2$ and $\mathbf{s}_3$ through corresponding points $\mathbf{m}_2$ and $\mathbf{m}_3$ in the second and third image respectively, define a 3-D line L that must intersect the optical ray of $\mathbf{m}_1$.

Using the properties of the Kronecker product (Cfr. pg. 37), the fundamental trifocal constraint (Eq. (74)) can be written as:

$$0 = (\mathbf{s}_3^T \otimes \mathbf{s}_2^T) T \mathbf{m}_1 = (\mathbf{m}_1^T \otimes \mathbf{s}_3^T \otimes \mathbf{s}_2^T) \operatorname{vec}(T) = \mathbf{s}_2^T (T\mathbf{m}_1)^{(3)} \mathbf{s}_3 \qquad (75)$$

where T is the 9×3 matrix defined by

$$T = (A_3 \otimes \mathbf{e}_{2,1}) - (\mathbf{e}_{3,1} \otimes A_2)$$

and $(T\mathbf{m}_1)^{(3)}$ is a 3×3 matrix such that $\operatorname{vec}(T\mathbf{m}_1)^{(3)} = T\mathbf{m}_1$. The *vector transposition* operator $A^{(p)}$generalizes the transpose of a matrix A by operating on vectors of p entries at a time (see [39]).

The matrix T represents the trilinear form, in the sense that contains its 27 coefficient. It also encodes the trifocal geometry, hence we[1] call it the *trifocal matrix*.

Each of the three equations (75) is an equivalent formulation of the fundamental trifocal constraint.

[1]This is an alternative approach to trifocal geometry, so please be aware that there is no such thing as the trifocal matrix in the literature.

5.2.1 Trifocal constraints for points.

Since a point is determined by two lines, we can write a similar independent constraint for a second line. Such two lines determining a point $\mathbf{m}$ are represented by any two rows of $[\mathbf{m}]_\times$. To keep notation compact let us consider the whole matrix $[\mathbf{m}]_\times$. The trifocal constraints for three points writes:

$$[\mathbf{m}_2]_\times (T\mathbf{m}_1)^{(3)} [\mathbf{m}_3]_\times = 0_{3\times 3}. \tag{76}$$

This is a matrix equation which gives 9 scalar equations, only four of which are independent. Equivalently

$$(\mathbf{m}_1^T \otimes [\mathbf{m}_3]_\times \otimes [\mathbf{m}_2]_\times) \operatorname{vec}(T) = \mathbf{0} \tag{77}$$

This equation can be used to recover T (likewise we did for F). The coefficient matrix is a 9×27 matrix of rank 4 (the rank of the Kronecker product is the product of the ranks), therefore every triplet $\{\mathbf{m}_1, \mathbf{m}_2, \mathbf{m}_3\}$ of corresponding points gives four linear independent equations. Seven triplets determine the 27 entries of T.

Point transfer.

Let $\mathbf{s}_2^T$ be a row of $[\mathbf{m}_2]_\times$, then

$$\left(\mathbf{s}_2^T (T\mathbf{m}_1)^{(3)}\right)[\mathbf{m}_3]_\times = 0 \tag{78}$$

This implies that the transpose of the leftmost term in parentheses (which is a 3-D vector) belongs to the kernel of $[\mathbf{m}_3]_\times$, which is equal to $\mathbf{m}_3$ (up to a scale factor) by construction. Hence

$$\mathbf{m}_3 \simeq (T\mathbf{m}_1)^{(3)^T}\mathbf{s}_2 \tag{79}$$

This is the point transfer equation: if $\mathbf{m}_1$ and $\mathbf{m}_2$ are conjugate points in the first and second view respectively, the position of the conjugate point $\mathbf{m}_3$ in the third view is computed by means of the trifocal matrix.

5.2.2 Trifocal constraint for lines.

Consider a line $\mathbf{L}$ in space projecting to $\mathbf{s}_1$, $\mathbf{s}_2$ and $\mathbf{s}_3$ in the three cameras. The fundamental trifocal constraint must hold for any point $\mathbf{m}_1$ contained in the line $\mathbf{s}_1$:

$$(\mathbf{s}_3^T \otimes \mathbf{s}_2^T) T \mathbf{m}_1 = 0 \quad \forall \mathbf{m}_1 : \mathbf{s}_1^T \mathbf{m}_1 = 0$$

hence

$$\mathbf{s}_1^T = (\mathbf{s}_3^T \otimes \mathbf{s}_2^T) T \tag{80}$$

This is the trifocal constraint for lines, which also allows directly line transfer.

Homography from the trifocal matrix

A line $\mathbf{s}_2$ in the second view defines (by back-projection) a 3-D plane, which induces a homography H between the first and the third view.

Hence, $\mathbf{s}_3 = H^T \mathbf{s}_1$ since $\mathbf{s}_1$ and $\mathbf{s}_3$ are both projection of the same line, that belongs to the plane [2].

On the other hand, Eq. (80) is equivalent to

$$\mathbf{s}_1 = (I_{3\times 3} \otimes \mathbf{s}_2^T) T^{(3)} \mathbf{s}_3.$$

Hence, the homography H can be expressed in terms of the trifocal matrix as:

$$H^T = (I_{3\times 3} \otimes \mathbf{s}_2^T) T^{(3)}$$

[2]The reader can verify that if H is the homography induced by a plane between two views, such that conjugate points are raleted by $\mathbf{m}_2 = H\mathbf{m}_1$, conjugate lines are related by $\mathbf{s}_2 = H^T \mathbf{s}_1$.

The trifocal tensor.

In this section we followed the approach of [40, 34] to avoid the tensorial notation. We will only briefly sketch here the tensorial formulation of the trifocal constraint.

The *trifocal tensor* $\mathcal{T}$ is a a $3 \times 3 \times 3$ array, which can be seen as a stack of three 3×3 matrices. The entries of $\mathcal{T}$ are those of the matrix T (just rearranged).

Indeed, $(T\mathbf{m}_1)^{(3)}$ can be seen as the linear combination of 3 matrices with the components of $\mathbf{m}_1$:

$$(T\mathbf{m}_1)^{(3)} = ([\mathbf{t}_1|\mathbf{t}_2|\mathbf{t}_3]\mathbf{m}_1)^{(3)} = (u\mathbf{t}_1 + v\mathbf{t}_2 + \mathbf{t}_3)^{(3)} = u\mathbf{t}_1^{(3)} + v\mathbf{t}_2^{(3)} + \mathbf{t}_3^{(3)}$$

Therefore, the action of the tensor $\mathcal{T}(\mathbf{m}_1, \mathbf{s}_2, \mathbf{s}_3) = \mathbf{s}_2^T(T\mathbf{m}_1)^{(3)}\mathbf{s}_3$ is to first combine matrices of the stack according to $\mathbf{m}_1$, then combine the columns according to $\mathbf{s}_3$ and finally to combine the elements according to $\mathbf{s}_2$.

A starting point to learn more on the trifocal tensor is [46].

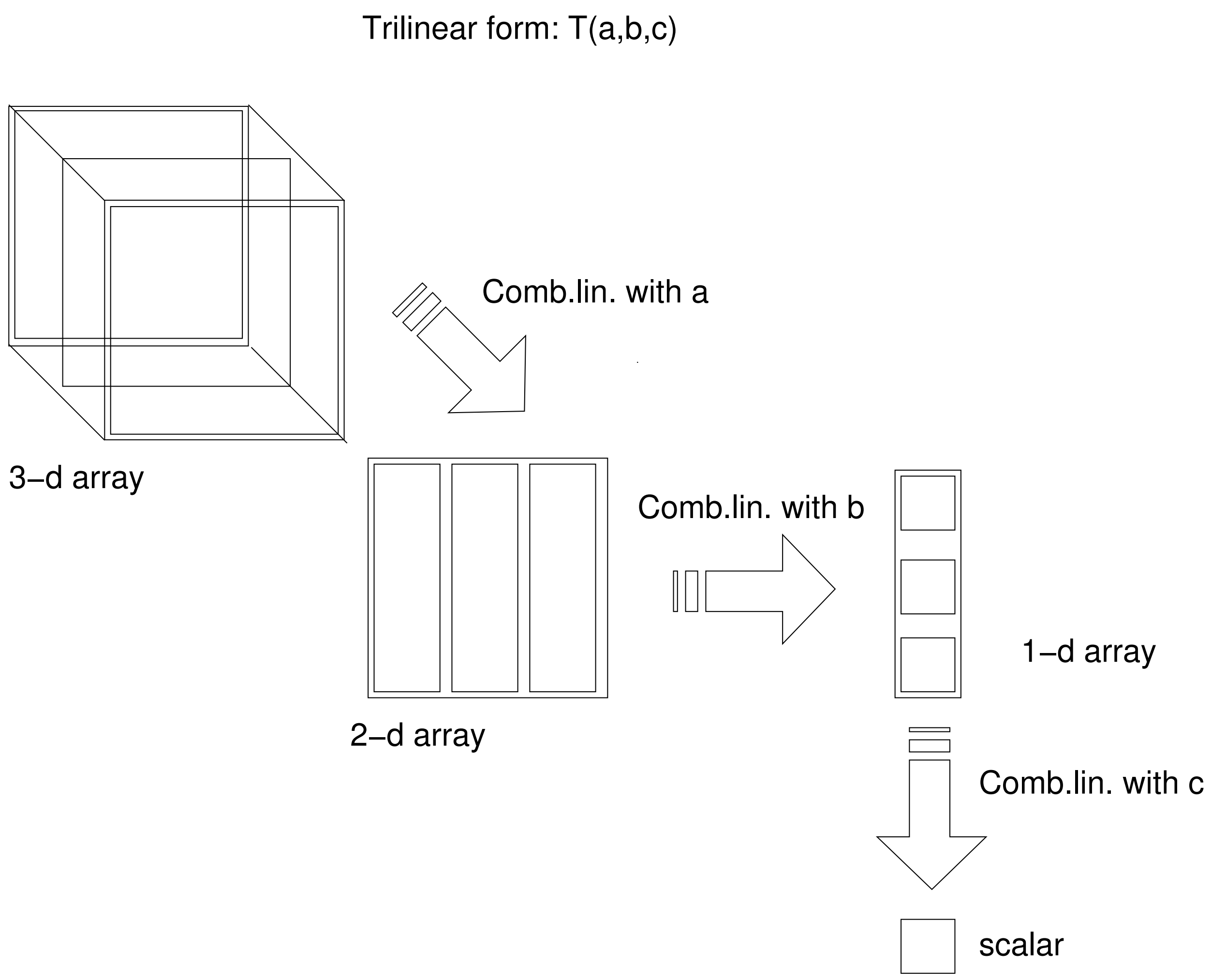


Fig. 16. Action of a trlinear form

5.3 Reconstruction

As in the case of two views, what can be reconstructed depends on what is known about the scene and the cameras.

If the internal parameters of the cameras are known, we can obtain a *Euclidean reconstruction*, that differs from the true reconstruction by a similarity transformation. This is composed by a rigid displacement (due to the arbitrary choice of the world reference frame) plus a a uniform change of scale (due to the well-known depth-speed ambiguity).

In the weakly calibrated case, i.e., when point correspondences are the only information available, a projective reconstruction can be obtained.

In both cases, the solution is not a straightforward generalization of the two view case, as the problem of *global* consistency comes into play (i.e., how to relate each other the *local* reconstructions that can be obtained from view pairs).

5.3.1 Euclidean Reconstruction

Let us consider for simplicity the case of three views, which generalizes straightforward to N views.

If one applies the method of Section 4.4.1 to view pairs 1-2, 1-3 and 2-3 one obtains three displacements $(R_{12}, \hat{\mathbf{t}}_{12}), (R_{13}, \hat{\mathbf{t}}_{13})$ and $(R_{23}, \hat{\mathbf{t}}_{23})$ known up a scale factor, as the norm of translation cannot be recovered, (the symbol $\hat{}$ indicates a unit vector).

The "true" displacements must satisfy the following compositional rule

$$\mathbf{t}_{13} = R_{23}\mathbf{t}_{12} + \mathbf{t}_{23} \tag{81}$$

which can be rewritten as

$$\hat{\mathbf{t}}_{13} = \mu_1 R_{23}\hat{\mathbf{t}}_{12} + \mu_2\hat{\mathbf{t}}_{23} \tag{82}$$

where $\mu_1 = ||\mathbf{t}_{12}||/||\mathbf{t}_{13}||$ and $\mu_2 = ||\mathbf{t}_{23}||/||\mathbf{t}_{13}||$ are unknown.

However, Eq. (81) constraints $\hat{\mathbf{t}}_{13}, R_{23}\hat{\mathbf{t}}_{12}$ and $\hat{\mathbf{t}}_{23}$ to be coplanar, hence the ratios μ_1, μ_2 can be recovered:

$$\frac{||\mathbf{t}_{12}||}{||\mathbf{t}_{13}||} = \mu_1 = \frac{(\hat{\mathbf{t}}_{13} \times \hat{\mathbf{t}}_{23}) \cdot (R_{23}\hat{\mathbf{t}}_{12} \times \hat{\mathbf{t}}_{23})}{||R_{23}\hat{\mathbf{t}}_{12} \times \hat{\mathbf{t}}_{23}||^2} \tag{83}$$

And similarly for μ_2.

In this way three consistent camera matrices can be instantiated.

Note that only ratios of translation norm can be computed, hence the global scale factor remains undetermined.

5.3.2 Projective Reconstruction

If one applies the method of Section 4.4.2 to consecutive pairs of views, she would obtain, in general, a set of reconstructions linked to each other by an unknown projective transformation (because each camera pair defines its own projective frame).

The trifocal geometry could be used to link together consistently triplets of views. In Section 4.4.2 we saw how a camera pair can be extracted from the fundamental matrix. Likewise, a triplet of consistent cameras can extracted from the trifocal matrix (or tensor). The procedure is more tricky, though.

An elegant method for multi-image reconstruction was described in [49], based on the same idea of the factorization method [51].

Consider m cameras $P_1 \dots P_m$ looking at n 3-D points $\mathbf{M}^1 \dots \mathbf{M}^n$. The usual projection equation

$$\zeta_i^j \mathbf{m}_i^j = P_i \mathbf{M}^j \quad i = 1 \dots m, \quad j = 1 \dots n. \tag{84}$$

can be written in matrix form:

$$\underbrace{\begin{bmatrix} \zeta_1^1 \mathbf{m}_1^1 & \zeta_1^2 \mathbf{m}_1^2 & \dots & \zeta_1^n \mathbf{m}_1^n \\ \zeta_2^1 \mathbf{m}_2^1 & \zeta_2^2 \mathbf{m}_2^2 & \dots & \zeta_2^n \mathbf{m}_2^n \\ \vdots & \vdots & \ddots & \vdots \\ \zeta_m^1 \mathbf{m}_m^1 & \zeta_m^2 \mathbf{m}_m^2 & \dots & \zeta_m^n \mathbf{m}_m^n \end{bmatrix}}_{\text{scaled measurements } W} = \underbrace{\begin{bmatrix} P_1 \\ P_2 \\ \vdots \\ P_m \end{bmatrix}}_{P} \underbrace{\left[\mathbf{M}^1, \mathbf{M}^2, \dots \mathbf{M}^n\right]}_{\text{structure } M}. \tag{85}$$

In this formula the $\mathbf{m}_i^j$ are known, but all the other quantities are unknown, including the projective depths ζ_i^j. Equation (85) tells us that W can be factored into the product of a $3m \times 4$ matrix P and a $4 \times n$ matrix M. This also means that W has rank four.

If we assume for a moment that the projective depths ζ_i^j are known, then matrix W is known too and we can compute its singular value decomposition:

$$W = UDV^T. \tag{86}$$

In the noise-free case, $D = \mathrm{diag}(\sigma_1, \sigma_2, \sigma_3, \sigma_4, 0, \ldots 0)$, thus, only the first 4 columns of U (V) contribute to this matrix product. Let $U_{3m\times 4}$ $(V_{n\times 4})$ the matrix of the first 4 columns of U (V). Then:

$$W = U_{3m\times 4}\,\mathrm{diag}(\sigma_1, \sigma_2, \sigma_3, \sigma_4)\,V_{n\times 4}^T. \tag{87}$$

The sought reconstruction is obtained by setting:

$$P = U_{3m\times 4}\,\mathrm{diag}(\sigma_1, \sigma_2, \sigma_3, \sigma_4) \quad \text{and} \quad M = V_{n\times 4}^T \tag{88}$$

This reconstruction is unique up to a (unknown) projective transformation. Indeed, for any non singular projective transformation T, TP and $T^{-1}M$ is an equally valid factorization of the data into projective motion and structure.

Consistently, the choice to subsume $\mathrm{diag}(\sigma_1, \sigma_2, \sigma_3, \sigma_4)$ in P is arbitrary.

In presence of noise, σ_5 will not be zero. By forcing $D = \mathrm{diag}(\sigma_1, \sigma_2, \sigma_3, \sigma_4, 0, \dots 0)$ one computes the solution that minimzes the following error:

$$||W - PM||_F^2 = \sum_{i,j} ||\zeta_i^j \mathbf{m}_i^j - P_i \mathbf{M}^j||^2$$

where $||\cdot||_F$ is the Frobenius norm.

As the depth ζ_i^j are unknown, we are left with the problem of estimating them.

An iterative solution is to alternate estimating ζ_i^j (given P and M) with estimating P and M (givenζ_i^j).

If P and M are known, estimating ζ_i^j is a linear problem. Indeed, for a given point j the projection equation writes:

$$\begin{bmatrix} \zeta_1^j \mathbf{m}_1^j \\ \zeta_2^j \mathbf{m}_2^j \\ \vdots \\ \zeta_m^j \mathbf{m}_m^j \end{bmatrix} = \underbrace{\begin{bmatrix} \mathbf{m}_1^j & 0 & \dots & 0 \\ 0 & \mathbf{m}_2^j & \dots & 0 \\ \vdots & \vdots & \ddots & \vdots \\ 0 & 0 & \dots & \mathbf{m}_m^j \end{bmatrix}}_{Q^j} \underbrace{\begin{bmatrix} \zeta_1^j \\ \zeta_2^j \\ \vdots \\ \zeta_m^j \end{bmatrix}}_{\boldsymbol{\zeta}^j} = PM^j \tag{89}$$

Starting from an initial guess for ζ_i^j (typically $\zeta_i^j = 1$), the following iterative procedure[3] is used:

1. Normalize W such that $||W|||_F = 1$;
2. Factorize W and obtain an estimate of P and M;
3. If $||W - PM||_F^2$ is sufficiently small then stop;
4. Solve for $\boldsymbol{\zeta}^j$ in $Q^j\boldsymbol{\zeta}^j = PM^j$, for all $j = 1 \dots n$;
5. Update W.
6. Goto 1.

Step 1 is necessary to avoid trivial solutions (e.g. $\zeta_i^j = 0$).

This technique is fast, requires no initialization, and gives good results in practice, although there is no guarantee that the iterative process will converge. A provably convergent iterative method has been presented in [36].

[3]Whilst this procedure captures the main idea of Sturm and Triggs, it is not exactly the algorithm proposed in [49]. To start with, the original algorithm [49] was not iterative and used the epipolar constraint (Eq.41) to fix the ratio of the projective depths of one point in successive images. It was [53] who made the scheme iterative. Moreover in [49] the normalization of W is performed by normalizing rows and columns of W. The Frobenius norm was used by [41]. A similar scheme was also proposed by [19].

5.4 Multifocal constraints

We outline here an alternative and elegant way to derive all the meaningful multilinear constraints on N views, based on determinants, described in [20]. Consider one image point viewed by m cameras:

$$\zeta_i \mathbf{m}_i = P_i \mathbf{M} \quad i = 1 \dots m \tag{90}$$

By stacking all these equations we obtain:

$$\begin{bmatrix} P_1 & \mathbf{m}_1 & 0 & \dots & 0 \\ P_2 & 0 & \mathbf{m}_2 & \dots & 0 \\ \vdots & \vdots & \vdots & \ddots & \vdots \\ P_m & 0 & 0 & \dots & \mathbf{m}_m \end{bmatrix} \begin{bmatrix} \mathbf{M} \\ -\zeta_1 \\ -\zeta_2 \\ \vdots \\ -\zeta_m \end{bmatrix} = \begin{bmatrix} 0 \\ 0 \\ \vdots \\ 0 \end{bmatrix} \tag{91}$$

This implies that the $3m \times (m+4)$ matrix (let us call it L) is rank-deficient, i.e., $\operatorname{rank} L < m+4$. In other words, all the $(m+4) \times (m+4)$ minors of L are equal to 0.

It has been proven that there are three different types of such minors that translates into meaningful multi-view constraints, depending on the number of rows taken from each view. $\textcircled{09}$

The minors that does not contain at least one row from each camera are identically zero, since they contain a zero column. Since one row has to be taken from each view and the remaining four can be distributed freely, one can choose:

1. Two rows from one view and two rows from another view. This gives a bilinear two-view constraint, expressed by the bifocal tensor i.e., the fundamental matrix.
2. Two rows from one view, one row from another view and one row from a third view. This gives a trilinear three-view constraint, expressed by the trifocal tensor.
3. One row from each of four different views. This gives a quadrilinear four-view constraint, expressed by the quadrifocal tensor.

All the other type of minors can be factorised as product of the two-, three-, or four-views constraints and point coordinates in the images[4].

[4]Actually, it can be proven that also the quadrifocal constraints are not independent [34].

6 Autocalibration

The aim of *autocalibration* is to compute the internal parameters, starting from weakly calibrated cameras.

More in general, the task is to recover metric properties of camera and/or scene, i.e., to compute a Euclidean reconstruction.

There are two classes of methods:

1. Direct: solve directly for the internal parameters.
2. Stratified: first obtain a projective reconstruction and then transform it to a Euclidean reconstruction (in some cases an affine reconstruction is obtained in between).

The reader is referred to [10] for a review of autocalibration, and to [37, 54, 23, 42, 38, 13] for classical and recent work on the subject.

6.1 Counting argument

Consider m cameras. The difference between the d.o.f. of the multifocal geometry (e.g. 7 for two views) and the d.o.f. of the rigid displacements (e.g. 5 for two views) is the number of independent constraints available for the computation of the intrinsic parameters (e.g. 2 for two views).

The multifocal geometry of m cameras (represented by the m-focal tensor) has $11m - 15$ d.o.f. Proof: a set of m cameras have $11m$ d.o.f., but they determine the m-focal geometry up to a collineation of $\mathbb{P}_3$, which has 15 d.o.f. The net sum is $11m - 15$ d.o.f.

On the other hand, the rigid displacements in m views are described by $6m - 7$ parameters: $3(m-1)$ for rotations, $2(m-1)$ for translations, and $m-2$ ratios of translation norms.

Thus, m *weakly calibrated views give* $5m - 8$ *constraints available for computing the intrinsic parameters*.

Let us suppose that m_k parameters are known and m_c parameters are constant.

The first view introduces $5 - m_k$ unknowns. Every view but the first introduces $5 - m_k - m_c$ unknowns.

Therefore, the unknown intrinsic parameters can be computed provided that

$$5m - 8 \geq (m-1)(5 - m_k - m_c) + 5 - m_k. \tag{92}$$

For example, if the intrinsic parameters are constant, three views are sufficient to recover them.

If one parameter (usually the skew) is known and the other parameters are varying, at least eight views are needed.

6.2 A simple direct method

If we consider two views, two independent constraints are available for the computation of the intrinsic parameters from the fundamental matrix.

Indeed, F has 7 d.o.f, whereas E, which encode the rigid displacement, has only 5 d.o.f. There must be two additional constraint that E must satisfy, with respect to F.

In particular, these constraints stem from the equality of two singular values of the essential matrix (Theorem 4.1) which can be decomposed in two independent polynomial equations.

Let F_{ij} be the (known) fundamental matrix relating views i and j, and let K_i and K_j be the respective (unknown) intrinsic parameter matrices.

The idea of [38] is that the matrix

$$E_{ij} = K_i^T F_{ij} K_j, \tag{93}$$

satisfies the constraints of Theorem 4.1 only if the intrinsic parameters are correct.

Hence, the cost function to be minimized is

$$C(K_i,\ i = 1 \dots n) = \sum_{i=1}^{n} \sum_{j>n}^{n} w_{ij} \frac{{}^1\sigma_{ij} - {}^2\sigma_{ij}}{{}^1\sigma_{ij} + {}^2\sigma_{ij}}, \tag{94}$$

where ${}^1\sigma_{ij} > {}^2\sigma_{ij}$ are the non zero singular values of E_{ij} and w_{ij} are normalized weight factors (linked to the reliability of the fundamental matrix estimate).

The previous counting argument shows that, in the general case of n views, the $n(n-1)/2$ two-view constraints that can be derived are not independent, nevertheless they can be used as they over-determine the solution.

A non-linear least squares solution is obtained with an iterative algorithm (e.g. Gauss-Newton) that uses analytical derivatives of the cost function.

A starting guess is needed, but this cost function is less affected than others by local minima problems. A globally convergent algorithm based on this cost function is described in [11].

6.3 Stratification

We have seen that a projective reconstruction can be computed starting from points correspondences only (weak calibration), without any knowledge of the camera matrices.

Projective reconstruction differs from Euclidean by an unknown projective transformation in the 3-D projective space, which can be seen as a suitable change of basis.

Starting from a projective reconstruction the problem is computing the transformation that "straighten" it, i.e., that upgrades it to an Euclidean reconstruction.

To this purpose the problem is *stratified* [33, 6] into different representations: depending on the amount of information and the constraints available, it can be analyzed at a projective, affine, or Euclidean level.

Let us assume that a projective reconstruction is available, that is a sequence P_i of $m+1$ camera matrices and a set $\mathbf{M}^j$ of $n+1$ 3-D points such that:

$$\mathbf{m}_i^j \simeq P_i \mathbf{M}^j \quad i = 0 \dots m, \quad j = 0 \dots n. \tag{95}$$

Without loss of generality, we can assume that camera matrices writes:

$$P_0 = [I \mid \mathbf{0}]; \qquad P_i = [A_i \mid \mathbf{e}_i] \quad \text{for } i = 1 \dots m \tag{96}$$

We are looking for the a 4×4 nonsingular matrix T that upgrades the projective reconstruction to Euclidean:

$$\mathbf{m}_i^j \simeq \underbrace{P_i T}_{P_i^{\mathrm{E}}} \underbrace{T^{-1} \mathbf{M}^j}_{\text{structure}}, \tag{97}$$

$P_i^{\mathrm{E}} = P_i T$ is the Euclidean camera,

We can choose the first Euclidean-calibrated camera to be $P_0^{\mathrm{E}} = K_0[I \mid \mathbf{0}]$, thereby fixing arbitrarily the world reference frame:

$$P_0^{\mathrm{E}} = K_0[I \mid \mathbf{0}] \qquad P_i^{\mathrm{E}} = K_i[R_i \mid \mathbf{t}_i] \quad \text{for } i = 1 \dots m. \tag{98}$$

With this choice, it is easy to see that $P_0^{\mathrm{E}} = P_0 T$ implies

$$T = \begin{bmatrix} K_0 & \mathbf{0} \\ \mathbf{r}^T & s \end{bmatrix} \tag{99}$$

where $\mathbf{r}^T$ is a 3-D vector and s is a scale factor, which we will arbitrarily set to 1 (the Euclidean reconstruction is up to a scale factor).

Under this parameterization T is clearly non singular, and it depends on eight parameters.

Substituting (99) in $P_i^{\mathrm{E}} \simeq P_i T$ gives

$$P_i^{\mathrm{E}} = [K_i R_i \mid K_i \mathbf{t}_i] \simeq P_i T = [A_i K_0 + \mathbf{e}_i \mathbf{r}^T \mid \mathbf{e}_i] \quad \text{for } i > 0 \tag{100}$$

and, considering only the leftmost 3×3 submatrix, gives

$$K_i R_i \simeq A_i K_0 + \mathbf{e}_i \mathbf{r}^T = P_i \begin{bmatrix} K_0 \\ \mathbf{r}^T \end{bmatrix} \tag{101}$$

Rotation can be eliminated using $RR^T = I$, leaving:

$$K_i K_i^T \simeq P_i \begin{bmatrix} K_0 K_0^T & K_0 \mathbf{r} \\ \mathbf{r}^T K_0^T & \mathbf{r}^T \mathbf{r} \end{bmatrix} P_i^T \tag{102}$$

This is the basic equation for autocalibration (called *absolute quadric constraint*), relating the unknowns K_i $(i = 0 \ldots m)$ and $\mathbf{r}$ to the available data P_i (obtained from weakly calibrated images).

Note that (102) contains five equations, because the matrices of both members are symmetric, and the homogeneity reduces the number of equations with one.

6.3.1 Geometric interpretation

Under camera matrix P the outline of the quadric Q is the conic C given by:

$$C^* \simeq PQ^*P^T \tag{103}$$

where C^* is the dual conic and Q^* is the dual quadric. An expression with Q and C may be derived, but it is quite complicated. C^* (resp. Q^*) is the adjoint matrix of C (resp. Q). If C is non singular, then $C^* = C^{-1}$.

In the beginning we introduced the absolute conic $\mathbf{\Omega}$, which is invariant under similarity transformation, hence deeply linked with the Euclidean stratum.

In a Euclidean frame, its equation is $x_1^2 + x_2^2 + x_3^2 = 0 = x_4$.

The absolute conic may be regarded as a special quadric (a disk quadric), therefore its dual is a quadric, the *dual absolute quadric*, denoted by $\mathbf{\Omega}^*$. Its representation is:

$$\mathbf{\Omega}^* = \mathrm{diag}(1,1,1,0). \tag{104}$$

As we already know, the *image of the absolute conic* under camera matrix P^{E} is given by $\boldsymbol{\omega} = (KK^T)^{-1}$, that is :

$$\boldsymbol{\omega}^* = (KK^T) \simeq P^{\mathrm{E}} \boldsymbol{\Omega}^* P^{\mathrm{E}^T} \tag{105}$$

This property is independent on the choice of the projective basis. What changes is the representation of the dual absolute quadric, which is mapped to

$$\boldsymbol{\Omega}^* = T \mathrm{diag}(1,1,1,0) T^T. \tag{106}$$

under the collineation T.

Substituting T from Eq. (99) into the latter gives:

$$\boldsymbol{\Omega}^* = \begin{bmatrix} K_0 K_0^T & K_0 \mathbf{r} \\ \mathbf{r}^T K_0^T & \mathbf{r}^T \mathbf{r} \end{bmatrix} \tag{107}$$

Recalling that $\boldsymbol{\omega}_i^* = K_i K_i^T$, then Eq. (102) is equivalent to

$$\boldsymbol{\omega}_i^* \simeq P_i \boldsymbol{\Omega}^* P_i^T \tag{108}$$

6.3.2 Solution strategies

Autocalibration requires to solve Eq. (108), with respect to $\mathbf{\Omega}^*$ (and $\boldsymbol{\omega}_i^*$).

If $\mathbf{\Omega}^*$ is known, the collineation T that upgrades cameras from projective to Euclidean is obtained by decomposing $\mathbf{\Omega}^*$ as in Eq. (106), using the eigenvalue decomposition.

$\mathbf{\Omega}^*$ might be parameterized as in Eq. (107) with 8 d.o.f. or parameterized as a generic 4×4 symmetric matrix (10 d.o.f.). The latter is an over-parameterization, as $\mathbf{\Omega}^*$ is also singular and defined up to a scale factor (which gives again 8 d.o.f.).

There are several strategies for dealing with the scale factor.

- Introduce the scale factor explicitly as an additional unknown [22]:

$$\boldsymbol{\omega}_i^* - \lambda_i P_i \mathbf{\Omega}^* P_i^T = \mathbf{0} \tag{109}$$

  This gives 6 equations but introduces one additional unknown (the net sum is 5).

- Eliminate it by using the same idea of the cross product for 3-D vectors [54].

  $$\mathrm{vech}(\boldsymbol{\omega}_i^*) \simeq \mathrm{vech}(P_i \boldsymbol{\Omega}^* P_i^T) \iff \mathrm{rank}\,\underbrace{[\mathrm{vech}(\boldsymbol{\omega}_i^*)|\,\mathrm{vech}(P_i \boldsymbol{\Omega}^* P_i^T)]}_{B} = 1$$

  where vech is the column-wise vectorization with the upper portion excluded, as matrices in Eq. (108) are symmetric.

  This is tantamount to say that every 2×2 minor of B is zero. There are 15 different order-2 minors of a 6×2 matrix, but only 5 equations are independent. $\textcircled{13}$

- Use a matrix norm (namely, Frobenius norm) [42]:

  $$\frac{\boldsymbol{\omega}_i^*}{||\boldsymbol{\omega}_i^*||_F} - \frac{P_i \boldsymbol{\Omega}^* P_i^T}{||P_i \boldsymbol{\Omega}^* P_i^T||_F} = \mathbf{0} \tag{110}$$

In any case, a non-linear least-squares problem has to be solved. Available numerical techniques (based on the Gauss-Newton method) are iterative, and requires an estimate of the solution to start.

This can be obtained by doing an educated guess about skew, principal point and aspect ratio, and solve the linear problem that results [43].

Linear solution

If some of the internal parameters are known, this causes some elements of $\boldsymbol{\omega}_i^*$ to vanish. Linear equations on $\mathbf{\Omega}^*$ are generated from zero-entries of $\boldsymbol{\omega}_i^*$ (because this eliminates the scale factor):

$$\boldsymbol{\omega}_i^*(k,\ell) = 0 \;\Rightarrow\; \mathbf{p}_{i,k}^T \mathbf{\Omega}^* \mathbf{p}_{i,\ell} = 0$$

where $\mathbf{p}_{i,k}^T$ is the k-th row of P_i.

Likewise, linear constraints on $\mathbf{\Omega}^*$ can be obtained from the equality of elements in the the upper (lower) triangular part of $\boldsymbol{\omega}_i^*$ (because $\boldsymbol{\omega}_i^*$ is symmetric).

In order to be able to solve linearly for $\mathbf{\Omega}^*$, at least 10 linear equations must be stacked up, to form a homogeneous linear system, which can be solved as usual (via SVD). Singularity of $\mathbf{\Omega}^*$ can be enforced a-posteriori by forcing the smallest singular value to zero.

If the principal point is known, $\boldsymbol{\omega}_i^*(1,3) = 0 = \boldsymbol{\omega}_i^*(2,3)$ and this gives two linear constraints. If, in addition, skew is zero we have $\boldsymbol{\omega}_i^*(1,2) = 0$. Known aspect ratio r provides a further constraint: $r\boldsymbol{\omega}_i^*(1,1) = \boldsymbol{\omega}_i^*(2,2)$.

Constant internal parameters

If all the cameras has the same internal parameters, so $K_i = K$, then Eq. (102) becomes

$$KK^T \simeq P_i \begin{bmatrix} KK^T & K\mathbf{r} \\ \mathbf{r}^T K^T & \mathbf{r}^T\mathbf{r} \end{bmatrix} P_i^T \qquad (111)$$

The constraints expressed by Eq. (111) are called the Kruppa constraints in [22].

Since each camera matrix, apart from the first one, gives five equations in the eight unknowns, a unique solution is obtained when at least three views are available.

The resulting system of equations is solved with a non-linear least-squares technique (e.g. Gauss-Newton).

7 Getting practical

In this section we will approach estimation problems from a more "practical" point of view.

First, we will discuss how the presence of errors in the data affects our estimates and describe the countermeasures that must be taken to obtain a good estimate.

Second, we will introduce non-linear distortions due to lenses into the pinhole model and we illustrate a practical calibration algorithm that works with a simple planar object.

Finally, we will describe *rectification*, a trasformation of image pairs such that conjugate epipolar lines become collinear and parallel to one of the image axes, usually the horizontal one. In such a way, the correspondence search is reduced to a 1D search along the trivially identified scanline.

7.1 Pre-conditioning

In presence of noise (or errors) on input data, the accuracy of the solution of a linear system depends crucially on the *condition number* of the system. The lower the condition number, the less the input error gets amplified (the system is more stable).

As [16] pointed out, it is crucial for linear algorithms (as the DLT algorithm) that input data is properly pre-conditioned, by a suitable coordinate change (origin and scale): points are translated so that their centroid is at the origin and are scaled so that their average distance from the origin is $\sqrt{2}$.

This improves the condition number of the linear system that is being solved.

Apart from improved accuracy, this procedure also provides invariance under similarity transformations in the image plane.

7.2 Algebraic vs geometric error

Measured data (i.e., image or world point positions) is noisy.

Usually, to counteract the effect of noise, we use more equations than necessary and solve with least-squares.

What is actually being minimized by least squares?

In a typical null-space problem formulation $Ax = 0$ (like the DLT algorithm) the quantity that is being minimized is the square of the residual $||Ax||$.

In general, if $||Ax||$ can be regarded as a distance between the geometrical entities involved (points, lines, planes, etc..), than what is being minimized is a geometric error, otherwise (when the error lacks a good geometrical interpretation) it is called an algebraic error.

All the linear algorithm (DLT and others) we have seen so far minimize an algebraic error. Actually, there is no justification in minimizing an algebraic error apart from the ease of implementation, as it results in a linear problem.

Usually, the minimization of a geometric error is a non-linear problem, that admit only iterative solutions and requires a starting point.

So, why should we prefer to minimize a geometric error? Because:

- The quantity being minimized has a meaning
- The solution is more stable
- The solution is invariant under Euclidean transforms

Often linear solution based on algebraic residuals are used as a starting point for a non-linear minimization of a geometric cost function, which “gives the solution a final polish” [14].

7.2.1 Geometric error for resection

The goal is to estimate the camera matrix, given a number of correspondences $(\mathbf{m}^j, \mathbf{M}^j) \quad j = 1 \dots n$

The geometric error associated to a camera estimate $\hat{P}$ is the distance between the measured image point $\mathbf{m}^j$ and the re-projected point $\hat{P}_i \mathbf{M}^j$:

$$\min_{\hat{P}} \sum_j d(\hat{P}\mathbf{M}^j, \mathbf{m}^j)^2 \tag{112}$$

where $d()$ is the Euclidean distance between the homogeneous points.

The DLT solution is used as a starting point for the iterative minimization (e.g. Gauss-Newton)

7.2.2 Geometric error for triangulation

The goal is to estimate the 3-D coordinates of a point $\mathbf{M}$, given its projection $\mathbf{m}_i$ and the camera matrix $\mathbf{P}_i$ for every view $i = 1 \ldots m$.

The geometric error associated to a point estimate $\hat{\mathbf{M}}$ in the i-th view is the distance between the measured image point $\mathbf{m}_i$ and the re-projected point $P_i\hat{\mathbf{M}}$:

$$\min_{\hat{\mathbf{M}}} \sum_i d(P_i\hat{\mathbf{M}}, \mathbf{m}_i)^2 \tag{113}$$

where $d()$ is the Euclidean distance between the homogeneous points.

The linear solution is used as a starting point for the iterative minimization (e.g. Gauss-Newton).

7.2.3 Geometric error for F

The goal is to estimate F given a a number of point correspondences $\mathbf{m}_\ell^i \leftrightarrow \mathbf{m}_r^i$.

The geometric error associated to an estimate $\hat{F}$ is given by the distance of conjugate points from conjugate lines (note the symmetry):

$$\min_{\hat{F}} \sum_j d(\hat{F}\mathbf{m}_\ell^j, \mathbf{m}_r^j)^2 + d(\hat{F}^T\mathbf{m}_r^j, \mathbf{m}_\ell^j)^2 \tag{114}$$

where $d()$ here is the Euclidean distance between a line and a point (in homogeneous coordinates).

The eight-point solution is used as a starting point for the iterative minimization (e.g. Gauss-Newton).

Note that F must be suitably parameterized, as it has only seven d.o.f. ⑪

7.2.4 Geometric error for H

The goal is to estimate H given a a number of point correspondences $\mathbf{m}_\ell^i \leftrightarrow \mathbf{m}_r^i$.

The geometric error associated to an estimate $\hat{H}$ is given by the symmetric distance between a point and its transformed conjugate:

$$\min_{\hat{H}} \sum_j d(\hat{H}\mathbf{m}_\ell^j, \mathbf{m}_r^j)^2 + d(\hat{H}^{-1}\mathbf{m}_r^j, \mathbf{m}_\ell^j)^2 \tag{115}$$

where $d()$ is the Euclidean distance between the homogeneous points. This also called the *symmetric transfer error*.

The linear solution is used as a starting point for the iterative minimization (e.g. Gauss-Newton).

7.2.5 Bundle adjustment (reconstruction)

If measurements are noisy, the projection equation will not be satisfied exactly by the camera matrices and structure computed in Sec. 5.3.2.

We wish to minimize the image distance between the re-projected point $\hat{P}_i\hat{\mathbf{M}}^j$ and measured image points $\mathbf{m}_i^j$ for every view in which the 3-D point appears:

$$\min_{\hat{P}_i,\hat{\mathbf{M}}^j} \sum_{i,j} d(\hat{P}_i\hat{\mathbf{M}}^j, \mathbf{m}_i^j)^2 \tag{116}$$

where $d()$ is the Euclidean distance between the homogeneous points.

As m and n increase, this becomes a very large minimization problem.

A solution is to alternate minimizing the re-projection error by varying $\hat{P}_i$ with minimizing the re-projection error by varying $\hat{\mathbf{M}}^j$.

See [55] for a review and a more detailed discussion on bundle adjustment.

7.3 Robust estimation

Up to this point, we have assumed that the only source of error affecting correspondences is in the measurements of point's position. This is a small-scale noise that gets averaged out with least-squares.

In practice, we can be presented with *mismatched* points, which are *outliers* to the noise distribution (i.e., wrong measurements following a different, unmodelled, distribution).

These outliers can severely disturb least-squares estimation (even a single outlier can totally offset the least-squares estimation, as demonstrated in Fig. 17.)

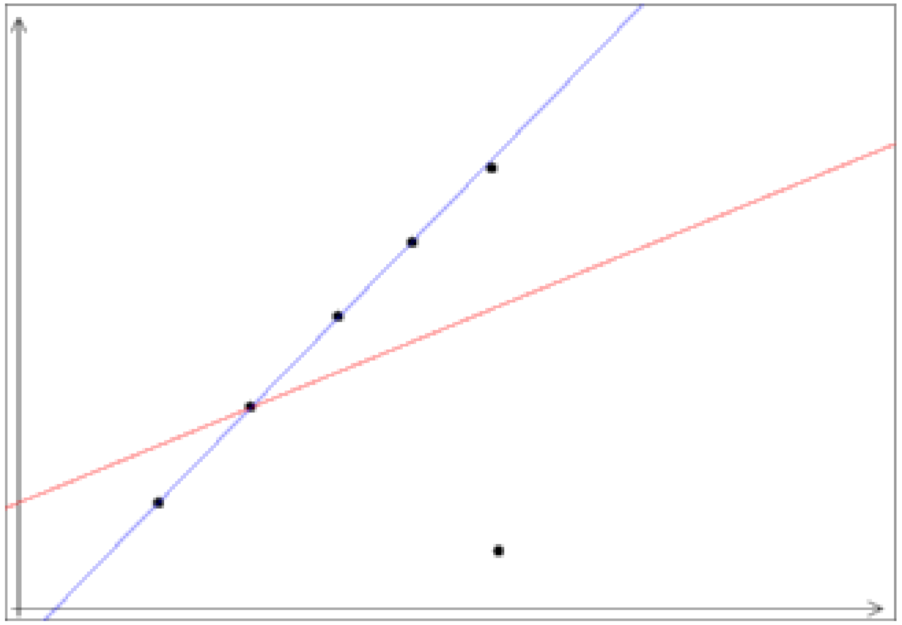


Fig. 17. A single outlier can severely offset the least-squares estimate (red line), whereas the robust estimate (blue line) is unaffected.

The goal of robust estimation is to be insensitive to outliers (or at least to reduce sensitivity).

7.3.1 M-estimators

Least squares:

$$\min_\theta \sum_i (r_i/\sigma_i)^2 \tag{117}$$

where θ are the regression coefficient (what is being estimated) and r_i is the residual. M-estimators are based on the idea of replacing the squared residuals by another function of the residual, yielding

$$\min_\theta \sum_i \rho(r_i/\sigma_i) \tag{118}$$

ρ is a symmetric function with a unique minimum at zero that grows sub-quadratically, called *loss function*.

Differentiating with respect to θ yields:

$$\sum_i \frac{1}{\sigma_i}\rho'(r_i/\sigma_i)\frac{dr_i}{d\theta} = 0 \tag{119}$$

The M-estimate is obtained by solving this system of non-linear equations.

7.3.2 RANSAC

Given a model that requires a minimum of p data points to instantiate its free parameters θ, and a set of data points S containing outliers:

1. Randomly select a subset of p points of S and instantiate the model from this subset
2. Determine the set S_i of data points that are within an error tolerance t of the model. S_i is the consensus set of the sample.
3. If the size of S_i is greater than a threshold T, re-estimate the model (possibly using least-squares) using S_i (the set of inliers) and terminate.
4. If the size of S_i is less than T, repeat from step 1.
5. Terminate after N trials and choose the largest consensus set found so far.

Three parameters need to be specified: t, T and N.

Both T and N are linked to the (unknown) fraction of outliers ϵ.

N should be large enough to have a high probability of selecting at least one sample containing all inliers. The probability to randomly select p inliers in N trials is:

$$P = 1 - (1 - (1 - \epsilon)^p)^N \tag{120}$$

By requiring that P must be near 1, N can be solved for given values of p and ϵ.

T should be equal to the expected number of inliers, which is given (in fraction) by $(1 - \epsilon)$.

At each iteration, the largest consensus set found so fare gives a lower bound on the fraction of inliers, or, equivalently, an upper bound on the number of outliers. This can be used to adaptively adjust the number of trials N.

t is determined empirically, but in some cases it can be related to the probability that a point under the threshold is actually an inlier [14].

As pointed out in [48], RANSAC can be viewed as a particular M-estimator.

The objective function that RANSAC maximizes is the number of data points having absolute residuals smaller that a predefined value t. This may be seen a minimising a binary loss function that is zero for small (absolute) residuals, and 1 for large absolute residuals, with a discontinuity at t.

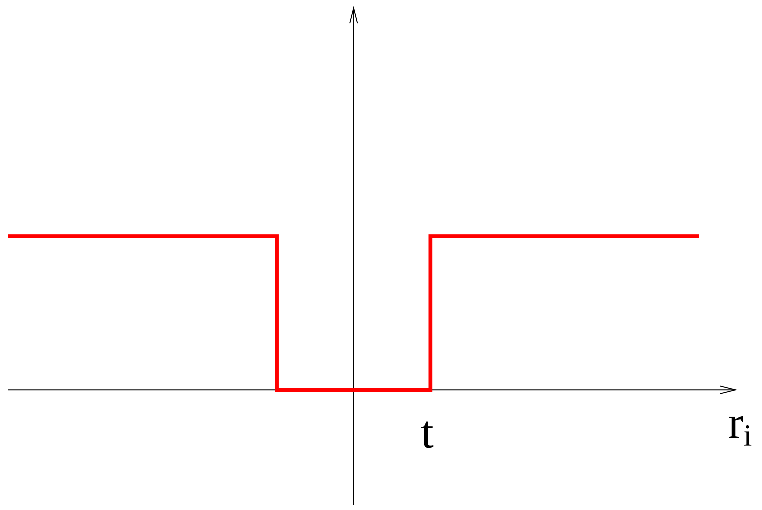


Fig. 18. RANSAC loss function

By virtue of the prespecified inlier band, RANSAC can fit a model to data corrupted by substantially more than half outliers.

7.3.3 LMedS

Another popular robust estimator is the Least Meadian of Squares. It is defined by:

$$\min_{\theta} \mathrm{med}_i r_i \tag{121}$$

It can tolerate up to 50% of outliers, as up to half of the data point can be arbitrarily far from the “true” estimate without changing the objective function value.

Since the median is not differentiable, a random sampling strategy similar to RANSAC is adopted. Instead of using the consensus, each sample of size p is scored by the median of the residuals of all the data points. The model with the least median (lowest score) is chosen.

A final weighted least-squares fitting is used.

With respect to RANSAC, LMedS can tolerate “only” 50% of outliers, but requires no setting of thresholds.

7.4 Practical calibration

Camera calibration (or resection) as described so far, requires a calibration object that consists typically of two or three planes orthogonal to each other. This might be difficult to obtain, without access to a machine tool.

Zhang [58] introduced a calibration technique that requires the camera to observe a planar pattern (much easier to obtain) at a few (at least three) different orientation. Either the camera or the planar pattern can be moved by hand.

Instead of requiring one image of many planes, this method requires many images of one plane.

We will also introduce here a more realistic camera model that takes into account non-linear effects produced by lenses.

In each view, we assume that correspondences between image points and 3-D points on the planar pattern have been established.

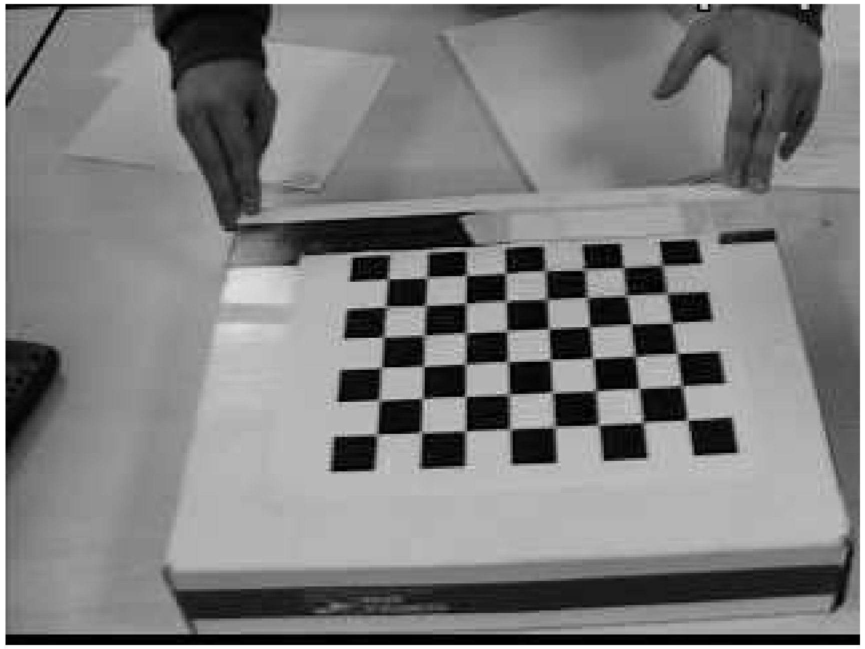


Fig. 19. Image of a planar calibration pattern. The points used for calibration are the corners of the black squares.

7.4.1 Estimating internal parameters

Following the development of Sec. 4.3 we know that for a camera $P = K[R|\mathbf{t}]$ the homography between a world plane at $z = 0$ and the image is

$$H \simeq K[\mathbf{r}_1, \mathbf{r}_2, \mathbf{t}] \tag{122}$$

where $\mathbf{r}_i$ are the column of R.

Suppose that H is computed from correspondences bwtween four or more known world points and their images, then some constraints can be obtained on the intrinsic parameters, thanks to the fact that the columns of R are orthonormal.

Writing $H = [\mathbf{h}_1, \mathbf{h}_2, \mathbf{h}_3]$, from the previous equation we derive:

$$\mathbf{r}_1 = \lambda K^{-1}\mathbf{h}_1 \quad \text{and} \quad \mathbf{r}_2 = \lambda K^{-1}\mathbf{h}_2 \tag{123}$$

where λ is an unknown scale factor.

The orthogonality $\mathbf{r}_1^T\mathbf{r}_2 = 0$ gives

$$\lambda^2\mathbf{h}_1^T(KK^T)^{-1}\mathbf{h}_2 = 0 \tag{124}$$

or, equivalently (remember that $\boldsymbol{\omega} = (KK^T)^{-1}$)

$$\mathbf{h}_1^T \boldsymbol{\omega} \mathbf{h}_2 = 0 \tag{125}$$

Likewise, the condition on the norm $\mathbf{r}_1^T \mathbf{r}_1 = \mathbf{r}_2^T \mathbf{r}_2$ gives

$$\mathbf{h}_1^T \boldsymbol{\omega} \mathbf{h}_1 = \mathbf{h}_2^T \boldsymbol{\omega} \mathbf{h}_2 \tag{126}$$

Introducing the Kronecker product as usual, we rewrite these two equations as:

$$(\mathbf{h}_1^T \otimes \mathbf{h}_2^T) \operatorname{vec} \boldsymbol{\omega} = 0 \tag{127}$$

$$\big((\mathbf{h}_1^T \otimes \mathbf{h}_1^T) - (\mathbf{h}_2^T \otimes \mathbf{h}_2^T)\big) \operatorname{vec} \boldsymbol{\omega} = 0 \tag{128}$$

A single view of the plane gives two equations in six unknowns, hence a solution is achievable with $n \geq 3$ views (in practice, for a good calibration, one should use around 12 views).

7.4.2 Estimating external parameters

K is obtained from the Cholesky factorization of $\boldsymbol{\omega}$, then R and $\mathbf{t}$ are recovered from:

$$[\mathbf{r}_1|\mathbf{r}_2|\mathbf{t}] = \frac{1}{||K^{-1}\mathbf{h}_1||} K^{-1}[\mathbf{h}_1|\mathbf{h}_2|\mathbf{h}_3] \qquad \mathbf{r}_3 = \mathbf{r}_1 \times \mathbf{r}_2 \tag{129}$$

Because of noise, the matrix R is not guaranteed to be orthogonal, hence we need to recover the closest orthogonal matrix.

Let $R = QS$ be the polar decomposition of R. Then Q is the closest possible orthogonal matrix to R in Frobenius norm.

In this way we have obtained the camera matrix P by minimizing an algebraic distance which is not geometrically meaningful.

It is advisable to refine it with a (non-linear) minimization of a geometric error:

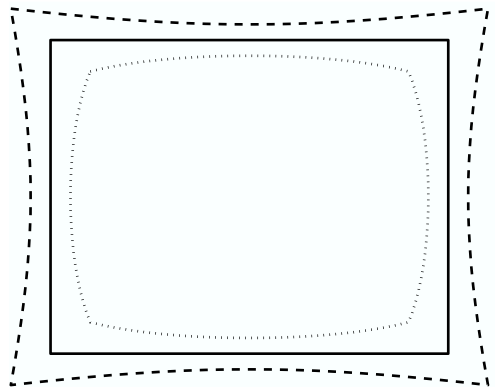
$$\min_{\hat{P}_i} \sum_{i=1}^{n} \sum_{j=1}^{m} d(\hat{P}_i \mathbf{M}^j, \mathbf{m}_i^j)^2 \tag{130}$$

where $\hat{P}_i = \hat{K}[\hat{R}_i | \hat{\mathbf{t}}_i]$ and the rotation has to be suitably parameterized with three parameters (see Rodrigues formula).

The linear solution is used as a starting point for the iterative minimization (e.g. Gauss-Newton).

7.4.3 Radial distortion

A realistic model for a photocamera or a videocamera must take into account non-linear distortions introduced by the lenses, especially when dealing with short focal lengths or low cost devices (e.g. webcams, disposable cameras).



The more relevant effect is the *radial distortion*, which is modeled as a non-linear transformation from ideal (undistorted) coordinates (u, v) to real observable (distorted) coordinates $(\hat{u}, \hat{v})$:

$$\begin{cases} \hat{u} = (u - u_0)(1 + k_1 r_d^2) + u_0 \\ \hat{v} = (v - v_0)(1 + k_1 r_d^2) + v_0 \end{cases} \quad . \qquad (131)$$

where $r_d^2 = \left(\frac{(u-u_0)}{a_u}\right)^2 + \left(\frac{(v-v_0)}{a_v}\right)^2$ and (u_0, v_0) are the coordinates of the image centre.

Estimating k_1

Let us assume that the pinhole model is calibrated. The point $\mathbf{m} = (u, v)$ projected according to the pinhole model (undistorted) do not coincide with the measured points $\hat{\mathbf{m}} = (\hat{u}, \hat{v})$ because of the radial distortion.

We wish to recover k_1 from Eq. (131). Each point gives two equation:

$$\begin{cases} (u - u_0)\left(\left(\frac{(u-u_0)}{a_u}\right)^2 + \left(\frac{(v-v_0)}{a_v}\right)^2\right) k_1 = \hat{u} - u \\ (v - u_0)\left(\left(\frac{(u-u_0)}{a_u}\right)^2 + \left(\frac{(v-v_0)}{a_v}\right)^2\right) k_1 = \hat{v} - v \end{cases} \tag{132}$$

hence a least squares solution for k_1 is readily obtained from $n > 1$ points.

When calibrating a camera we are required to *simultaneously* estimate both the pinhole model's parameters and the radial distortion coefficient.

The pinhole calibration we have described so far assumed no radial distortion, and the radial distortion calibration assumed a calibrated pinhole camera.

The solution (a very common one in similar cases) is to alternate between the two estimation until convergence.

Namely: start assuming $k = 0$, calibrate the pinhole model, then use that model to compute radial distortion. Once k_1 is estimated, refine the pinhole model by solving Eq. (130) with the radial distorion in the projection, and continue until the image error is small enough.

7.5 Rectification

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 two new virtual cameras, obtained by rotating the actual cameras and possibly modifying the intrinsic parameters.

The important advantage of rectification is that computing stereo correspondences is made simpler, because search is done along the horizontal lines of the rectified images.

We assume here that *the stereo pair is calibrated*, i.e., the cameras' internal parameters, mutual position and orientation are known. This assumption is not strictly necessary [18, 32, 25], but leads to a simpler technique and less distorted images.

Specifying virtual cameras.

Given the actual camera matrices P_{or} and $P_{o\ell}$, the idea behind rectification is to define two new *virtual* cameras P_{nr} and $P_{n\ell}$ obtained by rotating the actual ones around their optical centers until focal planes becomes coplanar, thereby containing the baseline (Figure 20). This ensures that epipoles are at infinity, hence epipolar lines are *parallel*.

To have *horizontal* epipolar lines, the baseline must be parallel to the x-axis of both virtual cameras. In addition, to have a proper rectification, conjugate points must have the *same vertical coordinate*.

In summary: positions (i.e, optical centers) of the virtual cameras are the same as the actual cameras, whereas the orientation of both virtual cameras differs from the actual ones by suitable rotations; intrinsic parameters are the same for both cameras.

Therefore, the two resulting virtual cameras 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.

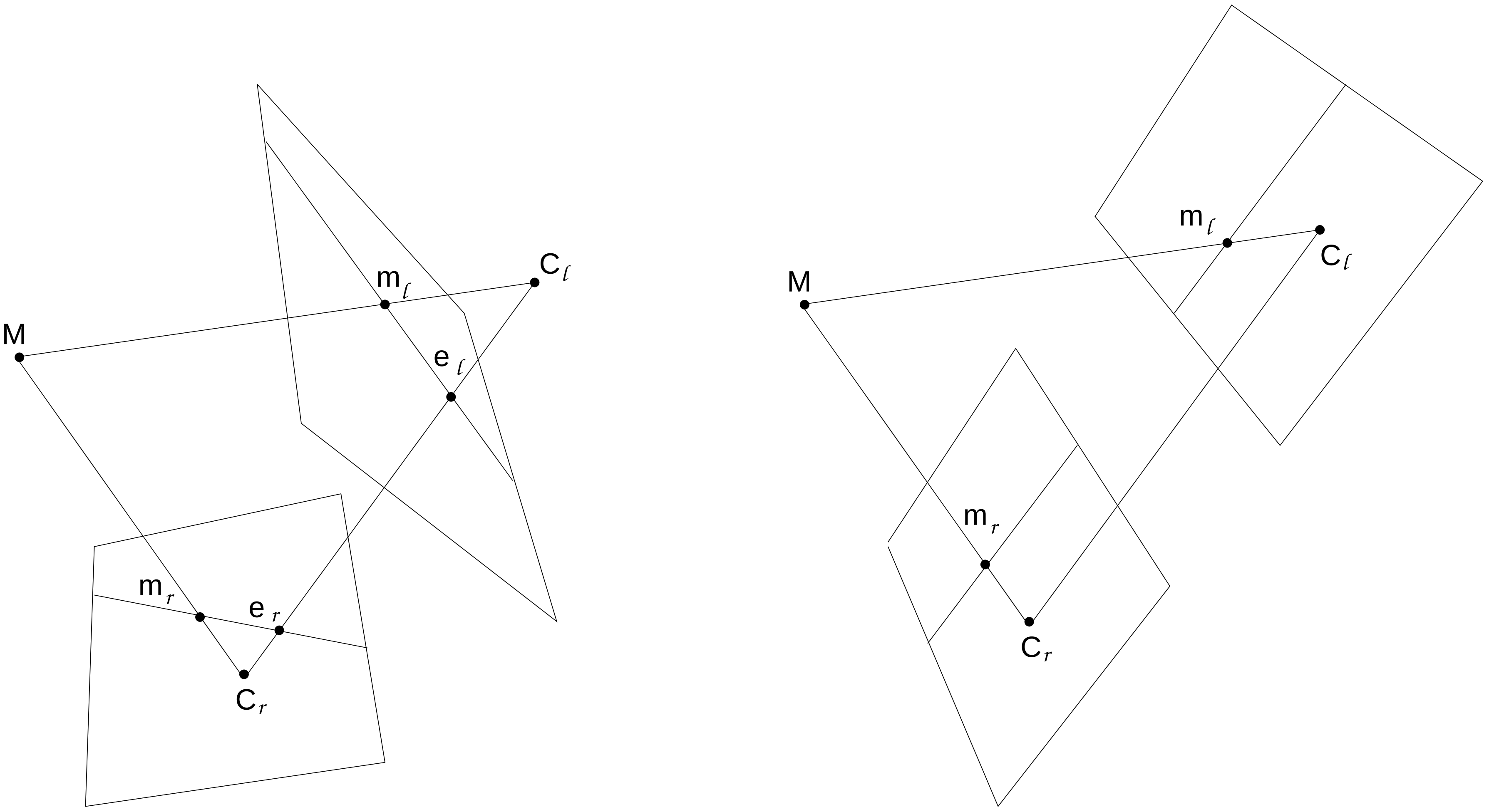


Fig. 20. Epipolar geometry before and after rectification.

Using Eq. (10) and Eq. (12), we can write the virtual cameras matrices as:

$$P_{n\ell} = K[R \mid -R\,\tilde{\mathbf{C}}_\ell], \qquad P_{nr} = K[R \mid -R\,\tilde{\mathbf{C}}_r]. \tag{133}$$

In order to define them, we need to assign $K, R, \tilde{\mathbf{C}}_\ell, \tilde{\mathbf{C}}_r$

The optical centers $\mathbf{C}_\ell$ and $\mathbf{C}_r$ are the same as the actual cameras. The intrinsic parameters matrix K can be chosen arbitrarily. The matrix R, which gives the orientation of both cameras will be specified by means of its row vectors:

$$R = \begin{bmatrix} \mathbf{r}_1^T \\ \mathbf{r}_2^T \\ \mathbf{r}_3^T \end{bmatrix} \tag{134}$$

that are the x, y, and z-axes, respectively, of the virtual camera reference frame, expressed in world coordinates.

According to the previous comments, we take:

(i) The x-axis parallel to the baseline: $\mathbf{r}_1 = (\tilde{\mathbf{C}}_r - \tilde{\mathbf{C}}_\ell)/||\tilde{\mathbf{C}}_r - \tilde{\mathbf{C}}_\ell||$

(ii) The y-axis orthogonal to x (mandatory) and to an arbitrary unit vector $\mathbf{k}$: $\mathbf{r}_2 = \mathbf{k} \times \mathbf{r}_1$

(iii) The z-axis orthogonal to xy (mandatory) : $\mathbf{r}_3 = \mathbf{r}_1 \times \mathbf{r}_2$

In point 2, $\mathbf{k}$ fixes the position of the y-axis in the plane orthogonal to x. In order to ensure that the virtual cameras look in the same direction as the actual ones, $\mathbf{k}$ is set equal to the direction of the optical axis of one of the two actual cameras.

We assumed that both virtual cameras have the same intrinsic parameters. Actually, the horizontal components of the image centre (v_0) can be different, and this degree of freedom might be exploited to “center” the rectified images in the viewport by applying a suitable horizontal translation.

The rectifying transformation.

In order to rectify the images, we need to compute the transformation mapping the image plane of P_o onto the image plane of P_n.

According to the equation of the optical ray, if $\mathbf{M}$ projects to $\mathbf{m}_o$ in the actual image and to $\mathbf{m}_n$ in the rectified image, we have:

$$\begin{cases} \tilde{\mathbf{M}} = \tilde{\mathbf{C}}_{+}\zeta_o P_{o_{1:3}}^{-1}\mathbf{m}_o \\ \tilde{\mathbf{M}} = \tilde{\mathbf{C}}_{+}\zeta_n P_{n_{1:3}}^{-1}\mathbf{m}_n \end{cases} \tag{135}$$

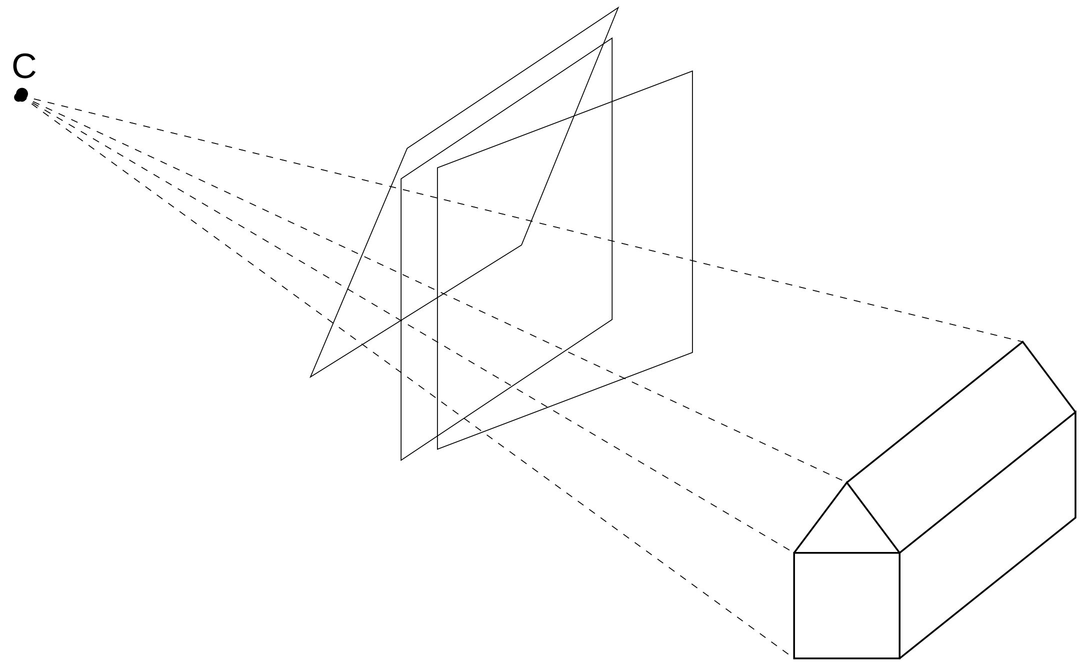
hence

$$\mathbf{m}_n = \frac{\zeta_o}{\zeta_n} \underbrace{P_{n_{1:3}} P_{o_{1:3}}^{-1}}_{H} \mathbf{m}_o \tag{136}$$

The rectifying transformation is a linear transformation of the projective plane (a *collineation*) given by the 3×3 matrix H.

It is understood that this has to be applied to the left and right images.

It is useful to think of an image as the intersection of the image plane with the cone of rays between points in 3-D space and the optical centre. We are moving the image plane while leaving fixed the cone of rays.



Reconstruction of 3-D points by triangulation can be performed from the rectified images directly, using P_{nr} and $P_{n\ell}$.

More details on the rectification algorithm can be found in [12], from which this section has been adapted.

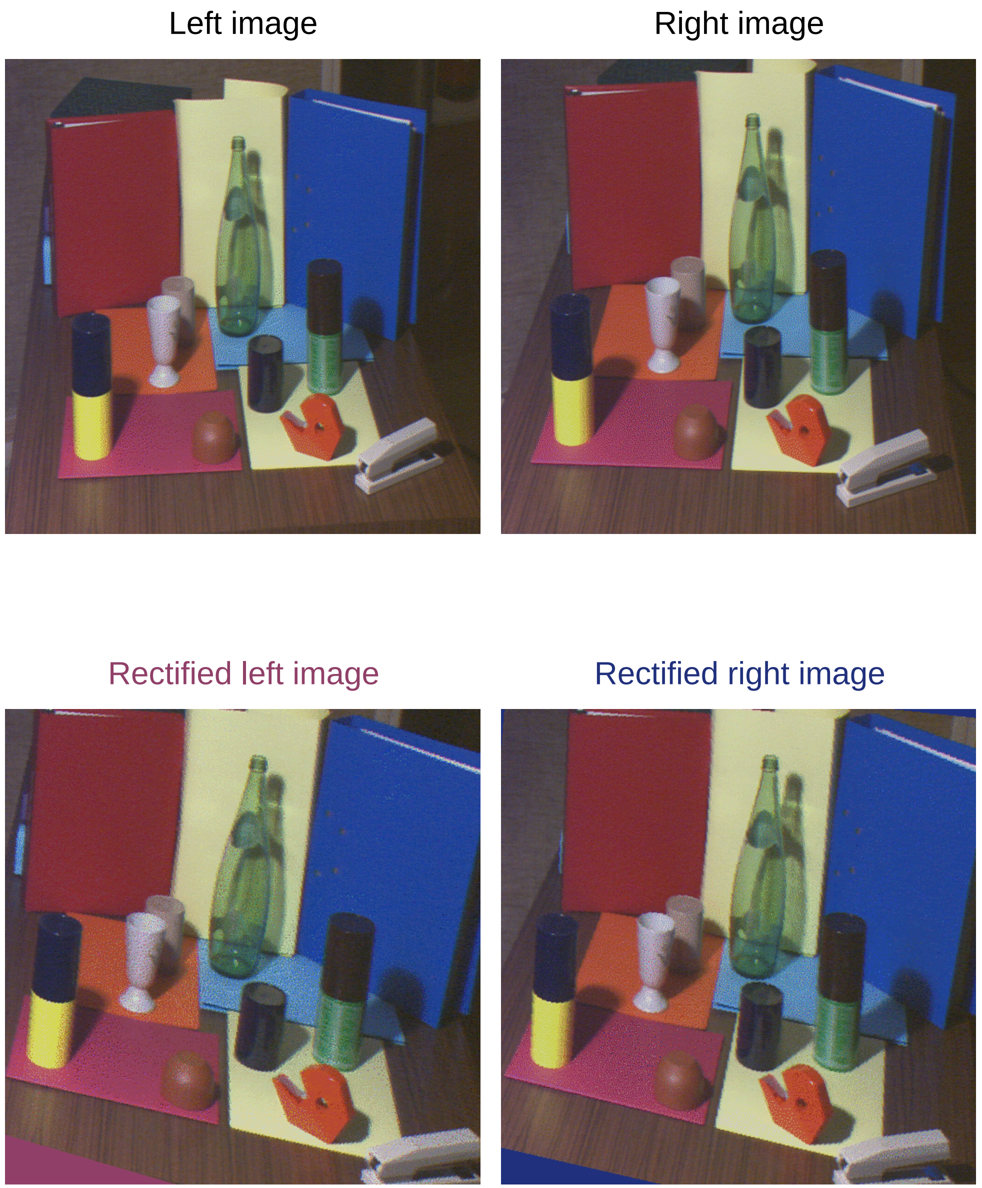


Fig. 21. Original and rectified stereo pair.

7.6 More calibration

Calibration problems can be cast as determining the transformation between two reference frames. Depending on the nature of the available measures and where the reference frames are attached we have four calibration or *orientation* problems[5]:

Relative orientation is the problem of determining the position and attitude of one perspective camera with respect to another camera from correspondences between points in 2-D images (See. Sec. 4.4.1).

Absolute orientation is the problem of aligning two sets of points, whose 3-D locations have been measured (or reconstructed) in two different reference frames.

Exterior orientation is the problem of determining the position and attitude of a perspective camera from correspondences between 3-D points and their 2-D images.

Interior orientation is the problem of determining the (affine) transformation from pixel to normalized image coordinates, i.e., the internal parameters of the camera (See. Sec. 7.4.1).

[5]this terminology comes from Photogrammetry.

7.6.1 Absolute orientation (with scaling)

Given two sets of 3-D points $\mathbf{X}^i$ and $\mathbf{Y}^i$, related by[6]

$$\mathbf{X}^i = s(R\mathbf{Y}^i + t) \quad \text{for all } i = 1 \ldots N \tag{137}$$

we are required to find the rotation matrix R, the vector $\mathbf{t}$ and the scalar s.

Summing these equations for all i and dividing by N shows that the translation is found with:

$$t = \frac{1}{s}\left(\frac{1}{N}\sum_{i=1}^{N}\mathbf{X}^i\right) - R\left(\frac{1}{N}\sum_{i=1}^{N}\mathbf{y}^i\right)$$

Combining this with Eq. (137) gives

$$\bar{\mathbf{X}}^i = sR\bar{\mathbf{Y}}^i$$

where $\bar{\mathbf{X}}^i = \mathbf{X}^i - \frac{1}{N}\sum_{i=1}^{N}\mathbf{X}^i$ and $\bar{\mathbf{Y}}^i = \mathbf{Y}^i - \frac{1}{N}\sum_{i=1}^{N}\mathbf{Y}^i$.

Because the rotation matrix does not change the length of the vectors, we can immediately solve for the scale from $||\bar{\mathbf{X}}^i|| = s||\bar{\mathbf{Y}}^i||$.

[6]Please note the change of notation: points are represented by affine (non-homogeneous) coordinates.

We are left with the problem of estimating the unknown rotation bewteen two sets of points.

Let $\bar{X}$ be the $3 \times N$ matrix formed by stacking the points $\bar{\mathbf{X}}^i$ side by side and $\bar{Y}$ be the matrix formed likewise by stacking the scaled points $s\bar{\mathbf{Y}}^i$.In presence of noise, we would like to minimize the sum of the square of the errors, or

$$\sum_{i=1}^{N} ||\bar{\mathbf{X}}^i - sR\bar{\mathbf{Y}}^i||^2 = ||\bar{X} - R\bar{Y}||_F^2$$

where $||\cdot||_F$ is the Frobenius norm.

This problem is known as the Orthogonal Procustes Problem and the solution is given by [28]

$$R = V \begin{bmatrix} 1 & 0 & 0 \\ 0 & 1 & 0 \\ 0 & 0 & \det(VU^T) \end{bmatrix} U^T$$

where $UDV^T = \bar{Y}\bar{X}^T$ is the SVD of the 3×3 matrix $\bar{Y}\bar{X}^T$.

7.6.2 Exterior orientation

Exterior orientation is a problem that appears repeatedly in computer vision, but in context different from 3-D reconstruction, such as visual servoing and augmented reality.

Given a number of point correspondences $\mathbf{m}^i \leftrightarrow \mathbf{M}^i$ and the internal camera parameters K, we are required to find a rotation matrix R and a translation vector $\mathbf{t}$ (which specify attitude and position of the camera) such that:

$$\zeta^i K^{-1}\mathbf{m}^i = [R|\mathbf{t}]\mathbf{M}^i \quad \text{for all } i. \tag{138}$$

One could immediately solve this problem by doing camera resection with DLT in normalized coordinates. The algorithm is linear, but it does not enforces the orthonormality constraints on the rotation matrix.

Instead, we present here the linear method proposed by Fiore [9]. He first recovers the unknown depths ζ^i, and then observes that what is left is an absolute orientation problem, whose solution yields a rotation matrix which is inherently orthonormal.

In order to recover the depths, let's write Eq. (138) in matrix form:

$$K^{-1}\left[\zeta^1\mathbf{m}^1,\ \zeta^2\mathbf{m}^2,\ \dots\ \zeta^n\mathbf{m}^n\right] = [R|\mathbf{t}]\underbrace{\left[\mathbf{M}^1,\ \mathbf{M}^2,\ \dots\ \mathbf{M}^n\right]}_{S}. \tag{139}$$

Let $r = \operatorname{rank} S$. Take its SVD: $S = UDV^T$ and let V_2 be a matrix composed by the last $n-r$ columns of V, which spans the null-space of S. Then, $SV_2 = 0_{3\times(n-r)}$, and also

$$K^{-1}\left[\zeta^1\mathbf{m}^1,\ \zeta^2\mathbf{m}^2,\ \dots\ \zeta^n\mathbf{m}^n\right]V_2 = 0_{3\times(n-r)} \tag{140}$$

Using vector transposition (see [39]), this equation can be re-written as

$$\left(\underbrace{\begin{bmatrix} K^{-1}\mathbf{m}^1 & 0 & \dots & 0 \\ & & \ddots & \\ 0 & 0 & \dots & K^{-1}\mathbf{m}^n \end{bmatrix}}_{D}\underbrace{\begin{bmatrix}\zeta^1 \\ \vdots \\ \zeta^n\end{bmatrix}}_{\boldsymbol{\zeta}}\right)^{(3)} V_2 = 0_{3\times(n-r)} \quad \text{or} \quad (D\boldsymbol{\zeta})^{(3)}V_2 = 0_{3\times(n-r)} \tag{141}$$

By taking vector transposition on both sides we get:

$$\left((D\boldsymbol{\zeta})^{(3)}V_2\right)^{(3)} = 0^{(3)}_{3\times(n-r)} \iff \left((V_2^T \otimes I_{3\times 3})D\right)\boldsymbol{\zeta} = \mathbf{0}. \tag{142}$$

From the last equation the depths $\boldsymbol{\zeta}$ can be recovered (up to a scale factor) by solving a null-space problem.

The size of the coefficients matrix is $3(n-r)\times 3$, and in order to determine a one-parameter family of solutions, it must have rank $n-1$, hence $3(n-r) \geq n-1$.

Therefore, at least $n \geq (3r-1)/2$ points are needed. If points are in general position, 6 are sufficient, but if they are on a plane, only 4 suffices.

Now that the left side of Eq. (138) is known, up to a scale factor, we are left with an absolute orientation (with scale) problem:

$$\zeta^i K^{-1}\mathbf{m}^i = s(R\tilde{\mathbf{M}}^i + \mathbf{t}) \quad \text{for all } i. \tag{143}$$

which we solve using the algorithm of Sec. 7.6.1. As a result, the rotation matrix estimate is orthonormal by construction.

8 Further readings

General books on (Geometric) Computer Vision are: [5, 56, 7, 14].

Acknowledgements

The backbone of this tutorial has been adapted from [27]. Many ideas are taken from [14]. Most of the material comes from multiple sources, but some sections are based on one or two works. In particular: the section on rectification is adapted from [12]; Sec. 5.4 is based on [21].

Michela Farenzena, Spela Ivekovic, Alessandro Negrente, Sara Ceglie and Roberto Marzotto produced most of the pictures. Some of them were inspired by [59]. The figure at page 127 was taken from Heikkilä.

References

[1] S. Avidan and A. Shashua. Novel view synthesis in tensor space. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, pages 1034–1040, 1997.

[2] P. Beardsley, A. Zisserman, and D. Murray. Sequential update of projective and affine structure from motion. *International Journal of Computer Vision*, 23(3):235–259, 1997.

[3] B. S. Boufama. The use of homographies for view synthesis. In *Proceedings of the International Conference on Pattern Recognition*, pages 563–566, 2000.

[4] Myron Z. Brown, Darius Burschka, and Gregory D. Hager. Advances in computational stereo. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 25(8):933–1008, August 2003.

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

[6] O. Faugeras. Stratification of 3-D vision: projective, affine, and metric representations. *Journal of the Optical Society of America A*, 12(3):465–484, 1994.

[7] O. Faugeras and Q-T Luong. *The geometry of multiple images*. MIT Press, 2001.

[8] O. D. Faugeras and L. Robert. What can two images tell us about a third one? In *Proceedings of the European Conference on Computer Vision*, pages 485–492, Stockholm, 1994.

[9] Paul D. Fiore. Efficient linear solution of exterior orientation. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 23(2):140–148, 2001.

[10] A. Fusiello. Uncalibrated Euclidean reconstruction: A review. *Image and Vision Computing*, 18(6-7):555–563, May 2000.

[11] A. Fusiello, A. Benedetti, M. Farenzena, and A. Busti. Globally convergent autocalibration using interval analysis. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 26(12):1633–1638, December 2004.

[12] A. Fusiello, E. Trucco, and A. Verri. A compact algorithm for rectification of stereo pairs. *Machine Vision and Applications*, 12(1):16–22, 2000.

[13] R. Hartley, E. Hayman, L. de Agapito, and I. Reid. Camera calibration and the search for infinity. In *Proceedings of the IEEE International Conference on Computer Vision*, 1999.

[14] R. Hartley and A. Zisserman. *Multiple view geometry in computer vision*. Cambridge University Press, 2nd edition, 2003.

[15] R. I. Hartley. Estimation of relative camera position for uncalibrated cameras. In *Proceedings of the European Conference on Computer Vision*, pages 579–587, Santa Margherita L., 1992.

[16] R. I. Hartley. In defence of the 8-point algorithm. In *Proceedings of the IEEE International Conference on Computer Vision*, 1995.

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

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

[19] A. Heyden. Projective structure and motion from image sequences using subspace methods. In *Scandinavian Conference on Image Analysis*, 1997.

[20] A. Heyden. A common framework for multiple-view tensors. In *Proceedings of the European Conference on Computer Vision*, Freiburg, Germany,, 1998.

[21] A. Heyden. Tutorial on multiple view geometry. In conjunction with ICPR00, September 2000.

[22] A. Heyden and K. Åström. Euclidean reconstruction from constant intrinsic parameters. In *Proceedings of the International Conference on Pattern Recognition*, pages 339–343, Vienna, 1996.

[23] A. Heyden and K. Åström. Minimal conditions on intrinsic parameters for Euclidean reconstruction. In *Proceedings of the Asian Conference on Computer Vision*, Hong Kong, 1998.

[24] T.S. Huang and O.D. Faugeras. Some properties of the E matrix in two-view motion estimation. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 11(12):1310–1312, December 1989.

[25] F. Isgrò and E. Trucco. Projective rectification without epipolar geometry. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, pages I:94–99, Fort Collins, CO, June 23-25 1999.

[26] F. Isgrò, E. Trucco, P. Kauff, and O. Schreer. 3-D image processing in the future of immersive media. *IEEE Transactions on Circuits and Systems for Video Technology*, 14(3):288–303, 2004.

[27] S. Ivekovic, A. Fusiello, and E. Trucco. Fundamentals of multiple view geometry. In O. Schreer, P. Kauff, and T. Sikora, editors, *3D Videocommunication. Algorithms, concepts and real-time systems in human centered communication*, chapter 6. John Wiley & Sons, 2005. ISBN: 0-470-02271-X.

[28] K. Kanatani. *Geometric Computation for Machine Vision*. Oxford University Press, 1993.

[29] S. Laveau and O. Faugeras. 3-D scene representation as a collection of images and foundamental matrices. Technical Report 2205, INRIA, Institut National de Recherche en Informatique et an Automatique, February 1994.

[30] Jed Lengyel. The convergence of graphics and vision. *IEEE Computer*, 31(7):46–53, July 1998.

[31] D. Liebowitz and A. Zisserman. Metric rectification for perspective images of planes. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, pages 482–488, 1998.

[32] C. Loop and Z. Zhang. Computing rectifying homographies for stereo vision. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, pages I:125–131, Fort Collins, CO, June 23-25 1999.

[33] Q.-T. Luong and T. Viéville. Canonical representations for the geometries of multiple projective views. *Computer Vision and Image Understanding*, 64(2):193–229, 1996.

[34] Yi Ma, Stefano Soatto, Jana Kosecka, and Shankar S. Sastry. *An Invitation to 3-D Vision*. Springer, November 2003.

[35] J. R. Magnus and H. Neudecker. *"Matrix Differential Calculus with Applications in Statistics and Econometrics"*. John Wiley & Sons, revised edition, 1999.

[36] S. Mahamud, M. Hebert, Y. Omori, and J. Ponce. Provably-convergent iterative methods for projective structure from motion. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, pages I:1018–1025, 2001.

[37] S. J. Maybank and O. Faugeras. A theory of self-calibration of a moving camera. *International Journal of Computer Vision*, 8(2):123–151, 1992.

[38] P.R.S. Mendonça and R. Cipolla. A simple techinique for self-calibration. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, pages I:500–505, 1999.

[39] Thomas Minka. Old and new matrix algebra useful for statistics. MIT Media Lab note.

[40] Theo Moons. A guided tour through multiview relations. In *SMILE*, pages 304–346, 1998.

[41] J. Oliensis. Fast and accurate self-calibration. In *Proceedings of the IEEE International Conference on Computer Vision*, 1999.

[42] M. Pollefeys, R. Koch, and L. Van Gool. Self-calibration and metric reconstruction in spite of varying and unknown internal camera parameters. In *Proceedings of the IEEE International Conference on Computer Vision*, pages 90–95, Bombay, 1998.

[43] M. Pollefeys, F. Verbiest, and L. Van Gool. Surviving dominant planes in uncalibrated structure and motion recovery. In *Proceedings of the European Conference on Computer Vision*, pages 837–851, 2002.

[44] 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.

[45] D. Scharstein and R. Szeliski. A taxonomy and evaluation of dense two-frame stereo correspondence algorithms. *International Journal of Computer Vision*, 47(1):7–42, May 2002.

[46] A. Shashua. Trilinear tensor: The fundamental construct of multiple-view geometry and its applications. In *International Workshop on Algebraic Frames For The Perception Action Cycle (AFPAC)*, Kiel Germany, Sep. 8-9 1997.

[47] A. Shashua and N. Navab. Relative affine structure: Canonical model for 3D from 2D geometry and applications. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 18(9):873–883, September 1996.

[48] C. V. Stewart. Robust parameter estimaton in computer vision. *SIAM Review*, 41(3):513–537, 1999.

[49] P. Sturm and B. Triggs. A factorization based algorithm for multi-image projective structure and motion. In *Proceedings of the European Conference on Computer Vision*, pages 709–720, Cambridge, UK, 1996.

[50] R. Szeliski. Video mosaics for virtual environments. *IEEE Computer Graphics and Applications*, 16(2):22–30, March 1996.

[51] C. Tomasi and T. Kanade. Shape and motion from image streams under orthography – a factorization method. *International Journal of Computer Vision*, 9(2):137–154, November 1992.

[52] Philip H. S. Torr and Andrew Zisserman. Robust computation and parametrization of multiple view relations. In *ICCV*, pages 727–732, 1998.

[53] B. Triggs. Factorization methods for projective structure from motion. In *CVPR*, pages 845–851, 1996.

[54] B. Triggs. Autocalibration and the absolute quadric. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, pages 609–614, Puerto Rico, 1997.

[55] Bill Triggs, Philip F. McLauchlan, Richard I. Hartley, and Andrew W. Fitzgibbon. Bundle adjustment - a modern synthesis. In *Proceedings of the International Workshop on Vision Algorithms*, pages 298–372. Springer-Verlag, 2000.

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

[57] Cha Zhang and Tsuhan Chen. A survey on image-based rendering - representation, sampling and compression. Technical Report AMP 03-03, Electrical and Computer Engineering - Carnegie Mellon University, Pittsburgh, PA 15213, June 2003.

[58] Z. Zhang. Determining the epipolar geometry and its uncertainty: A review. *International Journal of Computer Vision*, 27(2):161–195, March/April 1998.

[59] A. Zisserman. Single view and two-view geometry. Handout, EPSRC Summer School on Computer Vision, 1998. available from http://www.dai.ed.ac.uk/CVonline/LOCAL_COPIES/EPSRC_SSAZ/epsrc_ssaz.html.

Mathematical prerequisites

Orthogonal matrices	
Cholesky decomposition	
Null space (kernel)	
Rank	
Laplace expansion	
Eigenvalues and eigenvectors	
OR decomposition	
Singular Value decomposition	
Frobenius norm	
Kronecker product	
Multilinear forms	
Polar decomposition	
Cross product	
Triple product	
Homogeneous coordinates	
Projective space	