Camera Calibration: Problems and Algorithms

Andrea Fusiello*

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

VISMAC 2006, Palermo

1 Introduction

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[1]:

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.

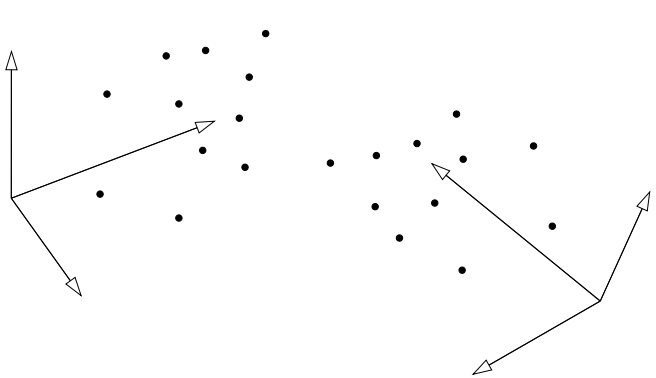
Absolute orientation is the problem of aligning two sets of points, whose 3-D locations have been measured (or reconstructed) in two different reference frames.

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.

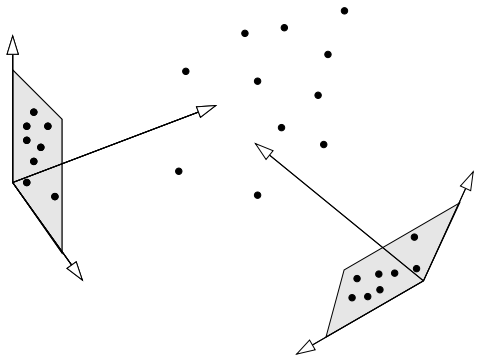
[1] this terminology comes from Photogrammetry.

Interior orientation is the problem of determining the (affine) transformation of the image plane from *normalized camera coordinates* to pixel coordinates.

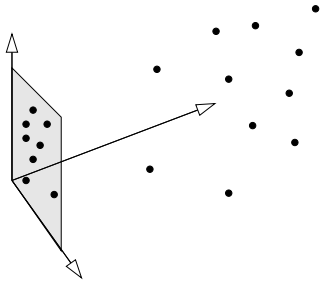
The solution to the interior and exterior orientation problems provides an overall solution to the camera calibration problem (or *resection*) that relates the position of pixels in the image to 3-D points.



Absolute orientation



Relative orientation



Exterior orientation

2 Pin-hole Camera

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

A 3-D point is projected onto the image plane with the line containing the point and the optical centre (see Figure 1).

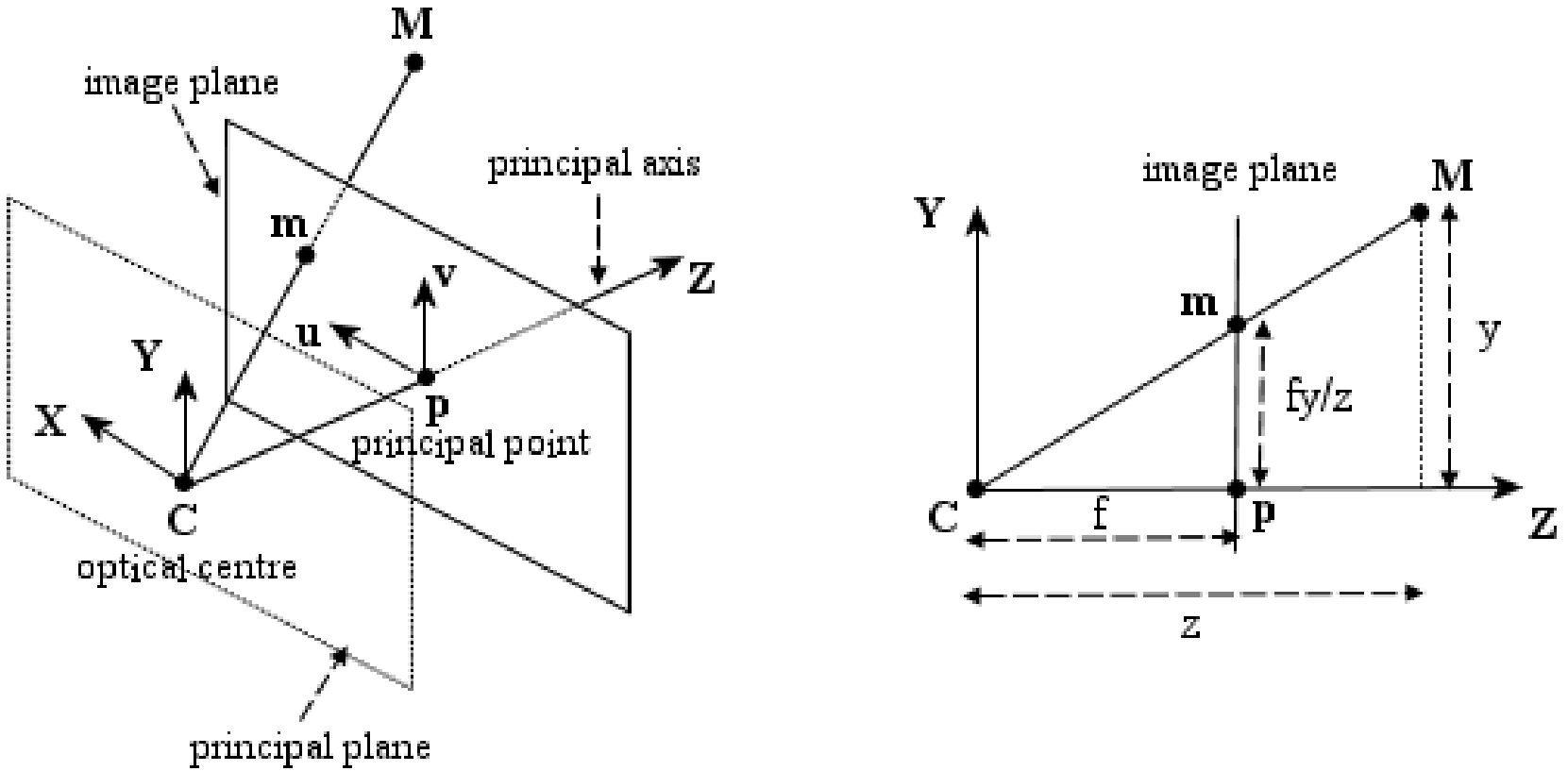


Fig. 1. Pin-hole camera geometry. The left figure illustrates the projection of the point **M** on the image plane by drawing the line through the camera centre **C** and the point to be projected. The right figure illustrates the same situation in the YZ plane, showing the similar triangles used to compute the position of the projected point **m** in the image plane.

2.1 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

$$\zeta \mathbf{m} = P\mathbf{M} \tag{1}$$

where

- $\mathbf{M} = (x, y, z, 1)^T$ are the homogeneous coordinates of the 3-D point,
- $\mathbf{m} = (u, v, 1)^T$ are the homogeneous pixel coordinates of the image point,
- ζ is the distance of $\mathbf{M}$ from the focal plane of the camera and
- P is the matrix describing the mapping, called the *camera projection matrix*.

The camera matrix is the product of two matrices

$$P = K[I|\mathbf{0}]G = K[R|\mathbf{t}] \tag{2}$$

Extrinsic parameters

$$G = \begin{bmatrix} R & \mathbf{t} \\ 0 & 1 \end{bmatrix} \tag{3}$$

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 *extrinsic* parameters.

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

Intrinsic parameters

$$K = \begin{bmatrix} f/s_x & f/s_x \cot\theta & o_x \\ 0 & f/s_y & o_y \\ 0 & 0 & 1 \end{bmatrix} \tag{4}$$

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

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).

General camera

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:

$$\mathbf{m} \simeq P\mathbf{M} \tag{5}$$

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:

$$P = \lambda K[R|\mathbf{t}], \tag{6}$$

(λ is recovered from $K(3,3) = 1$).

2.2 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

$$\mathbf{m}_i \simeq P\mathbf{M}_i \quad \text{for all } i. \tag{7}$$

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

$$\mathbf{m}_i \times P\mathbf{M}_i = \mathbf{0}. \tag{8}$$

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 [10], 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

$$\begin{bmatrix} \mathbf{0}^T & -\mathbf{M}_i^T & v_i\mathbf{M}_i^T \\ \mathbf{M}_i^T & \mathbf{0}^T & -u_i\mathbf{M}_i^T \\ -v_i\mathbf{M}_i^T & u_i\mathbf{M}_i^T & \mathbf{0}^T \end{bmatrix} \mathrm{vec}\,P^T = \mathbf{0} \tag{9}$$

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 [5].

3 Relative Orientation

The *relative orientation* problem requires to determine the rigid motion that links two cameras starting from two perspective views.

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 2).

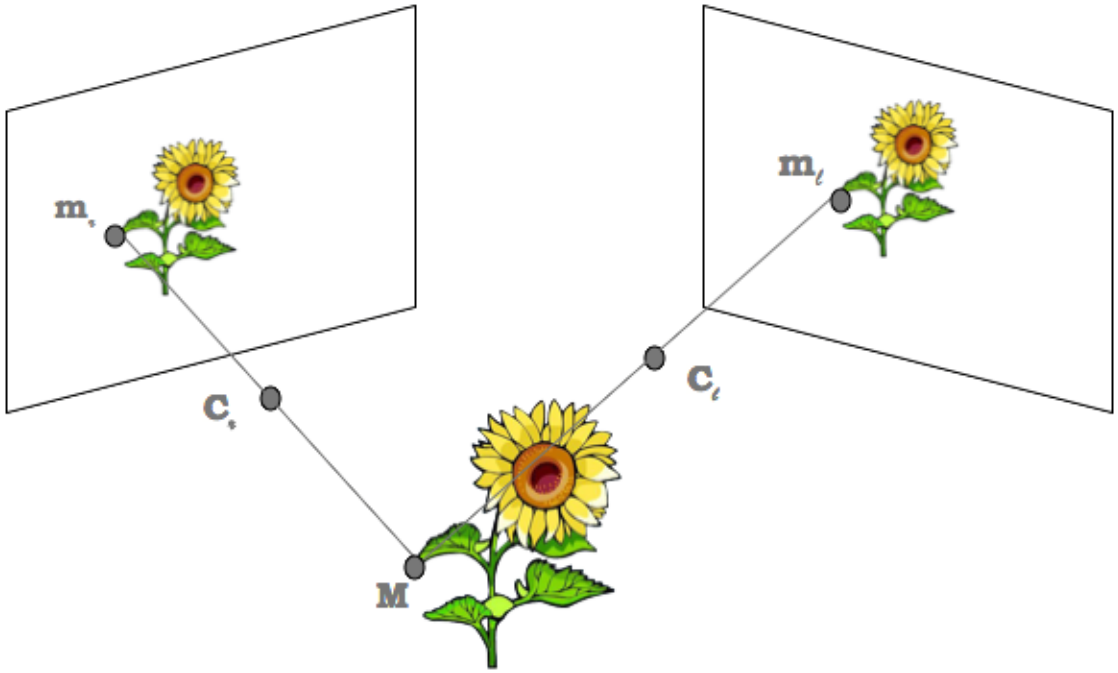


Fig. 2. Two perspective views of the same 3-D scene. $\mathbf{m}_\ell$ and $\mathbf{m}_r$ are corresponding points, as they are the projection of the same 3-D point, $\mathbf{M}$.

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 $\mathbf{M}$.

Given a number of point correspondences $\mathbf{m}_\ell^i \leftrightarrow \mathbf{m}_r^i$ and the intrinsic camera parameters K, we are required to find a rotation matrix R and a translation vector $\mathbf{t}$ that specify attitude and position of the second camera with respect to the first.

Plan:

- recall the essential matrix $E = [\mathbf{t}]_\times R$
- compute E from corresponding points
- extract R and $\mathbf{t}$ from E.

3.1 The Essential Matrix E

As the intrinsic parameters are known, we can switch to *normalized camera coordinates*: $\mathbf{m} \leftarrow K^{-1}\mathbf{m}$ (please note that this change of notation will hold throughout Sec.3).

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

$$P_\ell = [I|0] \quad \text{and} \quad P_r = [R|\mathbf{t}]. \tag{10}$$

With this choice, the unknown R and $\mathbf{t}$ have been made explicit.

The relationship between the corresponding image points $\mathbf{m}_\ell$ and $\mathbf{m}_r$ in normalized camera coordinates is the bilinear form:

$$\mathbf{m}_r^T E \mathbf{m}_\ell = 0. \tag{11}$$

E is the *essential matrix*:

$$E \triangleq [\mathbf{t}]_\times R. \tag{12}$$

where $[\mathbf{t}]_\times$ is the skew-symmetric matrix that act as the cross product with $\mathbf{t}$.

The essential matrix is a homogeneous quantity (i.e., is defined up to scale). This implies that $\mathbf{t}$ can be scaled arbitrarily in Equation (12) and one would get the same essential matrix.

Therefore translation can be recovered from E only up to an unknown scale factor. This is also known as *depth-speed ambiguity*.

E has five degrees of freedom: a 3-D rotation and a 3-D translation direction.

3.2 Factorization of E

The following theorem by [7] allows us to factorize the essential matrix into rotation and translation. It also define the constraints that the essential matrix must satisfy.

Theorem 3.1 *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.*

Proof. Let $E = SR$ where R is a rotation matrix and S is skew-symmetric. Let $S = [\mathbf{t}]_\times$ where $||\mathbf{t}|| = 1$. Then

$$EE^T = SRR^T S^T = SS^T = I - \mathbf{t}\mathbf{t}^T$$

Let U the orthogonal matrix such that $U\mathbf{t} = [0, 0, 1]^T$. Then

$$UEE^T U^T = U(I - \mathbf{t}\mathbf{t}^T)U^T = I - U\ \mathbf{t}\ \mathbf{t}^T U^T = I - [0, 0, 1]^T\ [0, 0, 1] = \begin{bmatrix} 1 & 0 & 0 \\ 0 & 1 & 0 \\ 0 & 0 & 0 \end{bmatrix}.$$

The elements of the diagonal matrix are the eigenvalues of EE^T i.e., the singular values of E. This demonstrate one implication.

Let us now give a constructive proof of the converse. Let $E = UDV^T$ be the SVD of E, with $D = \mathrm{diag}(1, 1, 0)$ (with no loss of generality, since E is defined up to a scale factor) and U and V orthogonal. The key observation is that

$$D = \begin{bmatrix} 1 & 0 & 0 \\ 0 & 1 & 0 \\ 0 & 0 & 0 \end{bmatrix} = \begin{bmatrix} 0 & -1 & 0 \\ 1 & 0 & 0 \\ 0 & 0 & 0 \end{bmatrix} \begin{bmatrix} 0 & 1 & 0 \\ -1 & 0 & 0 \\ 0 & 0 & 1 \end{bmatrix} \stackrel{\Delta}{=} S'R'$$

where S' is skew symmetric and R' a rotation. Hence

$$E{=}UDV^T{=}US'R'V^T{=}(US'U^T)(UR'V^T) = \underbrace{\det(UV^T)(US'U^T)}_{S}\underbrace{\det(UV^T)(UR'V^T)}_{R}.$$

Q.E.D.

This factorization is not unique. Because of homogeneity of E, we can change its sign, either by changing the sign of S' or by taking the transpose of R' (because $S'R'^T = -D$). In total, we have four possible factorizations given by:

$$S = U(\pm S')U^T \tag{13}$$

$$R = \det(UV^T)UR'V^T \text{ or } R = \det(UV^T)UR'^TV^T, \tag{14}$$

The choice between the four displacements is determined by the requirement that the 3-D points must lie in front of both cameras, i.e., their depth must be positive.

3.3 Estimating E: 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 (11) to compute the unknown matrix E.

We need to convert Equation (11) 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:

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

Each point correspondence gives rise to one linear equation in the unknown entries of E. 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 essential matrix E 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 $\mathrm{vec}(E)$ is the singular vector corresponding to the smallest singular value of A.

This method does not explicitly enforce constraints E, so it must be done *a posteriori*.

Let $E = UDV^T$ be the SVD decomposition of E, with $D = \mathrm{diag}(r, s, t)$ and $r \geq s \geq t$.

Replace E by E' such that $E' = U\mathrm{diag}(\frac{r+s}{2}, \frac{r+s}{2}, 0)V^T$

It can be shown that E' is the closest singular matrix to E in Frobenius norm.

4 Absolute orientation (with scaling)

Given two sets of 3-D points $\mathbf{X}^i$ and $\mathbf{Y}^i$, related by[2]

$$\mathbf{X}^i = s(R\mathbf{Y}^i + t) \quad \text{for all } i = 1 \dots N \tag{15}$$

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. (15) 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 $||\bar{\mathbf{X}}^i|| = s||\bar{\mathbf{Y}}^i||$.

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

We are left with the problem of estimating the unknown rotation between 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 Procrustes Problem and the solution is given by [9]

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

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

5 Exterior orientation

Exterior orientation is a problem that appears repeatedly in computer vision, in context such as visual servoing and augmented reality.

Given a number of point correspondences $\mathbf{m}^i \leftrightarrow \mathbf{M}^i$ and the intrinsic 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:

$$\zeta^i K^{-1}\mathbf{m}^i = [R|\mathbf{t}]\mathbf{M}^i \quad \text{for all } i. \tag{16}$$

One could immediately solve this problem by doing camera resection with DLT in normalized camera 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 [3]. 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. (16) in matrix form:

$$K^{-1}\underbrace{\left[\zeta^1\mathbf{m}^1,\ \zeta^2\mathbf{m}^2,\ \dots\ \zeta^n\mathbf{m}^n\right]}_{W} = [R|\mathbf{t}]\underbrace{\left[\mathbf{M}^1,\ \mathbf{M}^2,\ \dots\ \mathbf{M}^n\right]}_{M}. \tag{17}$$

Let $r = \operatorname{rank} M$. Take its SVD: $M = UDV^T$ and let V_2 be a matrix composed by the last $n - r$ columns of V, which spans the null-space of M. Then, $MV_2 = 0_{3\times(n-r)}$, and also

$$K^{-1}WV_2 = 0_{3\times(n-r)} \tag{18}$$

By taking vec on both sides we get:

$$(V_2^T \otimes K^{-1})\operatorname{vec}(W) = \mathbf{0}. \tag{19}$$

Let us observe that:

$$\operatorname{vec}(W) = \begin{bmatrix}\zeta^1\mathbf{m}^1\\ \zeta^1\mathbf{m}^2\\ \vdots\\ \zeta^n\mathbf{m}^n\end{bmatrix} = \underbrace{\begin{bmatrix}\mathbf{m}^1 & 0 & \dots & 0\\ & & \ddots & \\ 0 & 0 & \dots & \mathbf{m}^n\end{bmatrix}}_{D}\underbrace{\begin{bmatrix}\zeta^1\\ \vdots\\ \zeta^n\end{bmatrix}}_{\boldsymbol{\zeta}} \tag{20}$$

Hence

$$\left((V_2^T \otimes K^{-1})D\right)\boldsymbol{\zeta} = \mathbf{0}. \tag{21}$$

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. (16) is known, up to a scale factor, we are left with an absolute orientation (with scale) problem:

$$\zeta^i K^{-1}\mathbf{m}^i = s(R\tilde{\mathbf{M}}^i + \mathbf{t}) \quad \text{for all } i. \tag{22}$$

which we solve using the algorithm of Sec. 4. As a result, the rotation matrix estimate is orthonormal by construction.

6 Calibration with a planar target

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 [14] 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.

First, let us establish that the map between a world plane and its perspective image is given by a 3×3 homogeneous matrix.

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:

$$\zeta \begin{bmatrix} u \\ v \\ 1 \end{bmatrix} = P \begin{bmatrix} x \\ y \\ 0 \\ 1 \end{bmatrix} = [P_1|P_2|P_4] \begin{bmatrix} x \\ y \\ 1 \end{bmatrix}. \tag{23}$$

Points are mapped from the world plane to the image plane with a 3×3 (non-singular) matrix, which represents an *homography*[3] of the projective plane.

[3]Collineations or homographies are linear transformations of a projective space into itself.

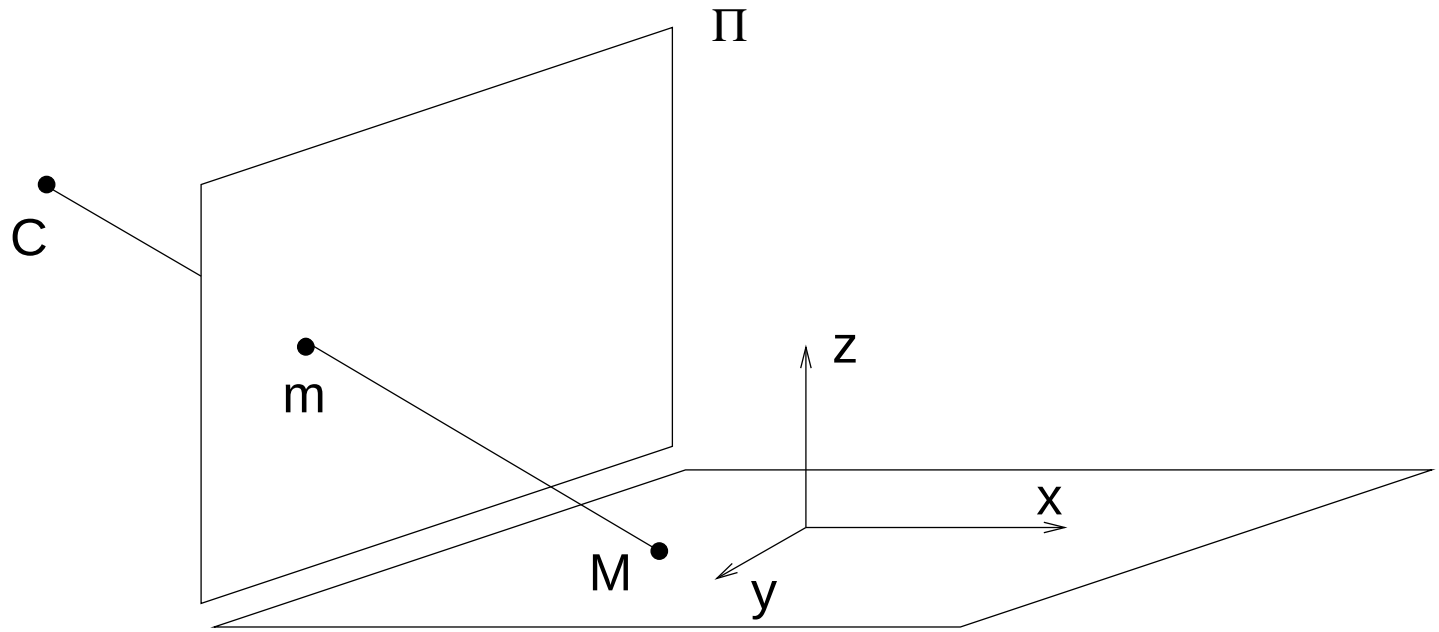


Fig. 3. The map between a world plane Π and a perspective image is a collineation.

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

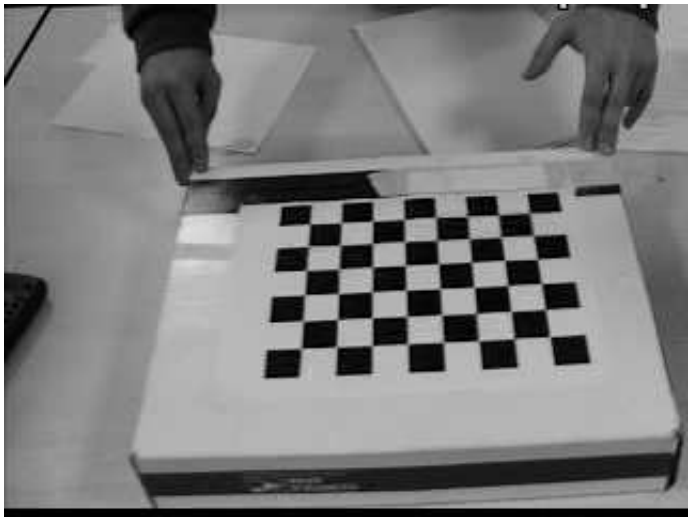


Fig. 4. Planar calibration pattern. Points used for calibration are the corners of the black squares.

Plan:

- compute homography H from point correspondences
- recover intrinsic parameters K from H
- recover extrinsic parameters

6.1 Estimating the homography

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

$$\mathbf{m}_r^i \simeq H\mathbf{m}_\ell^i \quad \text{for all } i \tag{24}$$

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

$$\mathbf{m}_r \times H\mathbf{m}_\ell = \mathbf{0} \tag{25}$$

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

$$\begin{bmatrix} \mathbf{0}^T & -\mathbf{m}_\ell^T & v\mathbf{m}_\ell^T \\ \mathbf{m}_\ell^T & \mathbf{0}^T & -u\mathbf{m}_\ell^T \\ -v\mathbf{m}_\ell^T & u\mathbf{m}_\ell^T & \mathbf{0}^T \end{bmatrix} \mathrm{vec}(H^T) = \mathbf{0} \tag{26}$$

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.

6.2 Estimating intrinsic parameters (Interior orientation)

We know that for a camera $P = K[R|\mathbf{t}]$ the homography between a world plane at $z = 0$ and the image is

$$H \simeq K[\mathbf{r}_1, \mathbf{r}_2, \mathbf{t}] \tag{27}$$

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

Suppose that H is computed from correspondences between 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:

$$\mathbf{r}_1 = \lambda K^{-1}\mathbf{h}_1 \quad \text{and} \quad \mathbf{r}_2 = \lambda K^{-1}\mathbf{h}_2 \tag{28}$$

where λ is an unknown scale factor.

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

$$\lambda^2\mathbf{h}_1^T(KK^T)^{-1}\mathbf{h}_2 = 0 \tag{29}$$

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

$$\mathbf{h}_1^T \boldsymbol{\omega} \mathbf{h}_2 = 0 \tag{30}$$

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

$$\mathbf{h}_1^T \boldsymbol{\omega} \mathbf{h}_1 = \mathbf{h}_2^T \boldsymbol{\omega} \mathbf{h}_2 \tag{31}$$

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

$$(\mathbf{h}_1^T \otimes \mathbf{h}_2^T) \operatorname{vec} \boldsymbol{\omega} = 0 \tag{32}$$

$$\big((\mathbf{h}_1^T \otimes \mathbf{h}_1^T) - (\mathbf{h}_2^T \otimes \mathbf{h}_2^T)\big) \operatorname{vec} \boldsymbol{\omega} = 0 \tag{33}$$

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).

6.3 Estimating extrinsic parameters

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

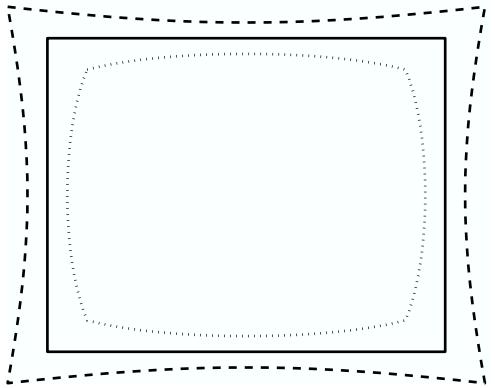
$$[\mathbf{r}_1|\mathbf{r}_2|\mathbf{t}] = \frac{1}{||K^{-1}\mathbf{h}_1||}K^{-1}[\mathbf{h}_1|\mathbf{h}_2|\mathbf{h}_3] \qquad \mathbf{r}_3 = \mathbf{r}_1 \times \mathbf{r}_2 \tag{34}$$

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.

6.4 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) pixel coordinates (u, v) to observed (distorted) pixel coordinates $(\hat{u}, \hat{v})$:

$$\begin{cases} \hat{u} = (u - u_0)(1 + k_1 r_d^2) + u_0 \\ \hat{v} = (v - v_0)(1 + k_1 r_d^2) + v_0 \end{cases} . \tag{35}$$

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. (35). Each point gives two equation:

$$\begin{cases} (u - u_0)\left(\left(\dfrac{(u - u_0)}{a_u}\right)^2 + \left(\dfrac{(v - v_0)}{a_v}\right)^2\right) k_1 = \hat{u} - u \\ (v - u_0)\left(\left(\dfrac{(u - u_0)}{a_u}\right)^2 + \left(\dfrac{(v - v_0)}{a_v}\right)^2\right) k_1 = \hat{v} - v \end{cases} \tag{36}$$

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 with the radial distortion in the projection, and continue until the image error is small enough.

7 Getting practical

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

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.

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 [6] 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" [5].

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 $\hat{P}_i \mathbf{M}^j$:

$$\min_{\hat{P}} \sum_j d(\hat{P}\mathbf{M}^j, \mathbf{m}^j)^2 \tag{37}$$

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 E

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

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

$$\min_{\hat{E}} \sum_j d(\hat{E}\mathbf{m}_\ell^j, \mathbf{m}_r^j)^2 + d(\hat{E}^T\mathbf{m}_r^j, \mathbf{m}_\ell^j)^2 \tag{38}$$

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 E must be suitably parametrized, as it has only five d.o.f.

7.2.3 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:

$$\min_{\hat{H}} \sum_j d(\hat{H}\mathbf{m}_\ell^j, \mathbf{m}_r^j)^2 + d(\hat{H}^{-1}\mathbf{m}_r^j, \mathbf{m}_\ell^j)^2 \tag{39}$$

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).

7.3 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 illustrated in Fig. 5.)

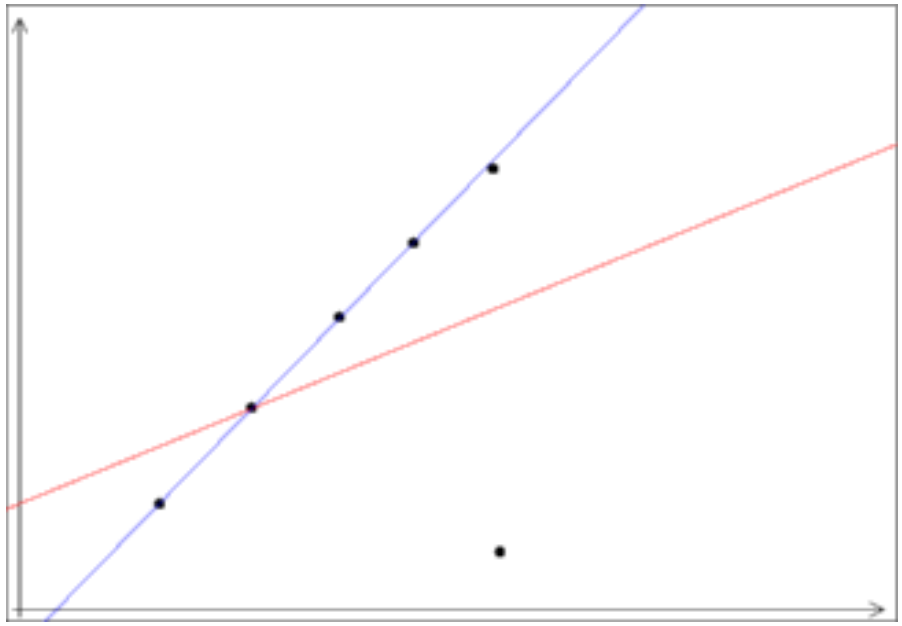


Fig. 5. A single outlier can severely offset the least-squares estimate (red line), whereas the robust estimate (blue line) is unaffected.

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

7.3.1 M-estimators

Least squares:

$$\min_{\theta} \sum_{i} (r_i/\sigma_i)^2 \tag{40}$$

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

$$\min_{\theta} \sum_{i} \rho(r_i/\sigma_i) \tag{41}$$

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

Differentiating with respect to θ yields:

$$\sum_{i} \frac{1}{\sigma_i} \rho'(r_i/\sigma_i) \frac{dr_i}{d\theta} = 0 \tag{42}$$

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:

$$P = 1 - (1 - (1 - \epsilon)^p)^N \tag{43}$$

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 [5].

As pointed out in [11], 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.

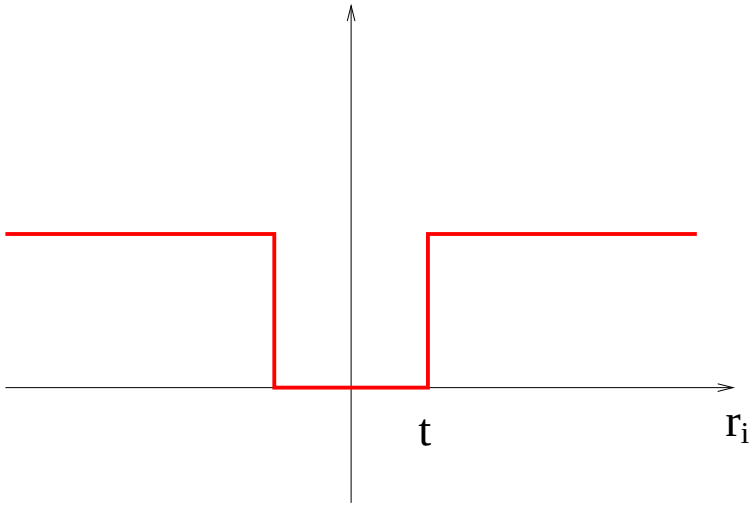


Fig. 6. RANSAC loss function

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 Median of Squares. It is defined by:

$$\min_{\theta} \mathrm{med}_i r_i \tag{44}$$

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.

8 Further readings

General books on (Geometric) Computer Vision are: [1, 2, 5, 13].

Acknowledgements Figure 1 has been created by Spela Ivekovic, Figure 2 by Michela Farenzena, Figure 4 by Alberto Valinetti. Figure 3 has been copied from [15]. The image at page 36 is due to Heikkilä.

References

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

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

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

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

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

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

[7] 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.

[8] 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.

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

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

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

[12] 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.

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

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

[15] 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.