Crash course in geometric Computer Vision

Andrea Fusiello

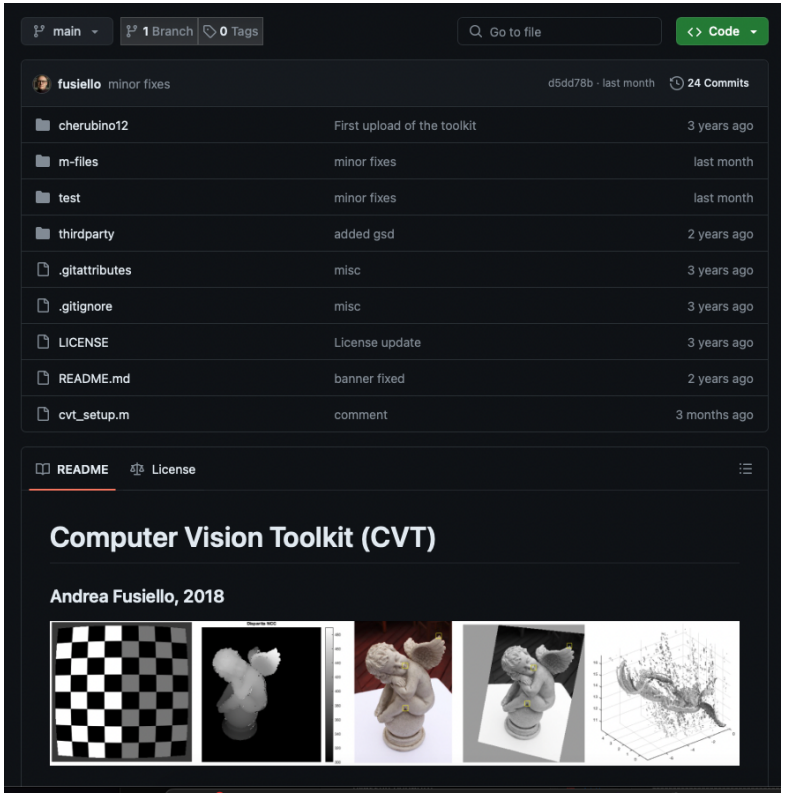
University of Udine, IT

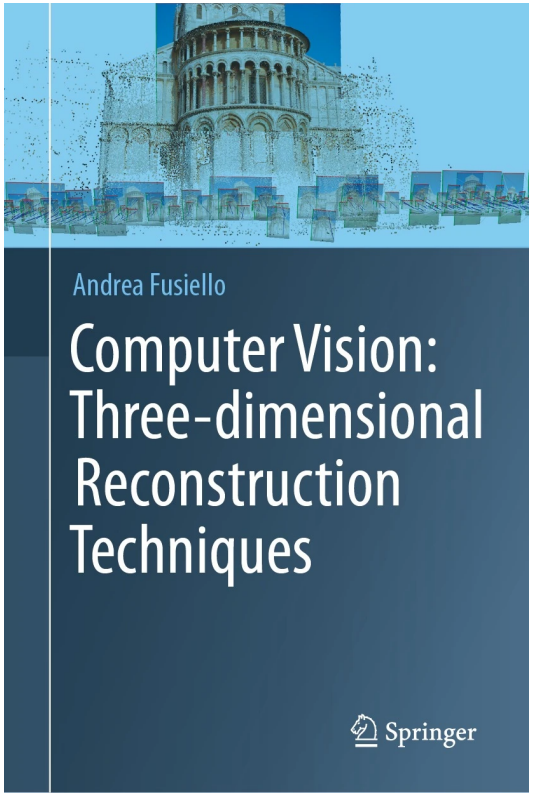
Udine, June 2024

Contents

1 Resources

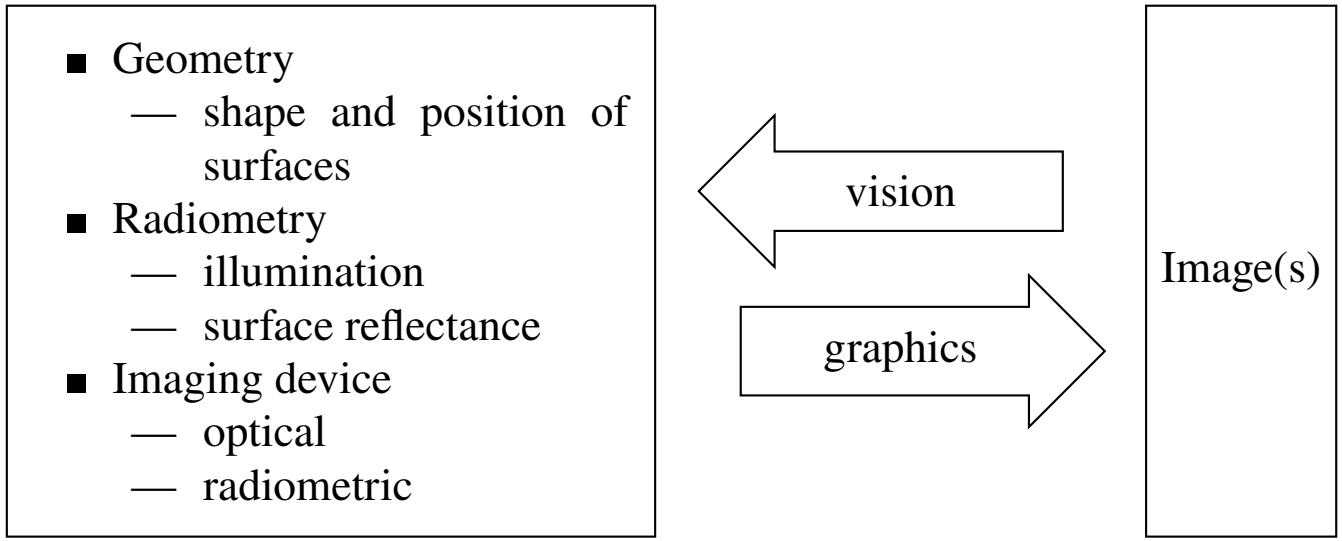
- Matlab listing and the sources are available on Github at https://github.com/fusiello/Computer_Vision_Toolkit.
- A. Fusiello, "Computer Vision: Three-dimensional reconstruction techniques." Springer 2023





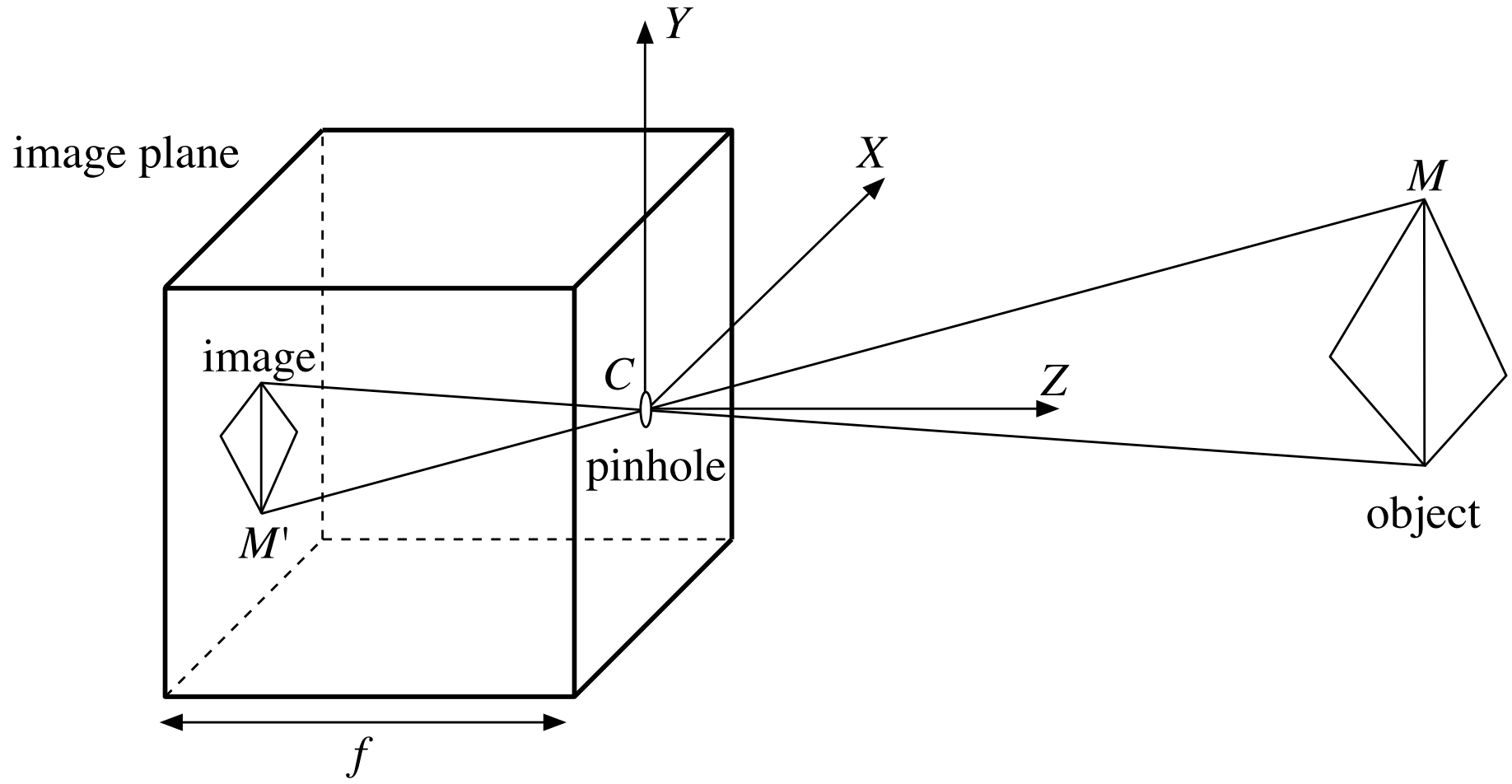
2 Image formation

As seeing agents, we are so used to the benefits of vision, and so unaware of how we actually use it, that it took a long time to appreciate the almost miraculous achievements of our visual system. If one tries to adopt a more objective and detached attitude, by considering the visual system as a device that records a band of electromagnetic radiation as an input, and then uses it to gain knowledge about surrounding objects that emit and reflect it, one cannot help but be struck by the richness of information this system provides. [Ullman]



Geometry of image formation

Pinhole camera:



Let M be a point of the scene, with coordinates (X, Y, Z) and let M' be its projection on the image plane of coordinates (X', Y', Z') through the pinhole C, also called centre of projection (COP). If f is the distance of C from the image plane (focal length), then from the similarity of the triangles we obtain:

$$\frac{-X'}{f} = \frac{X}{Z} \quad \text{e} \quad \frac{-Y'}{f} = \frac{Y}{Z} \tag{1}$$

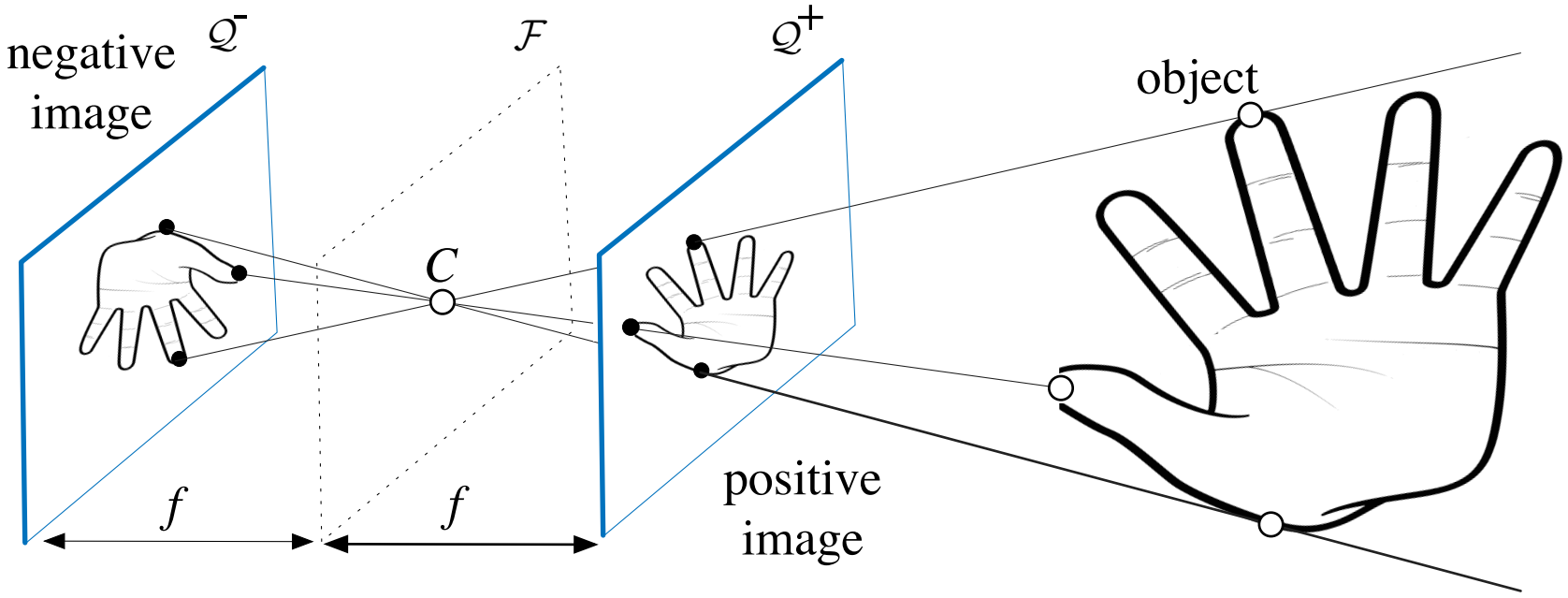
and therefore

$$X' = \frac{-fX}{Z}, \quad Y' = \frac{-fY}{Z}, \quad Z' = -f. \tag{2}$$

Note that the image is inverted with respect to the scene, both right-left and up-down, as indicated by the minus sign.

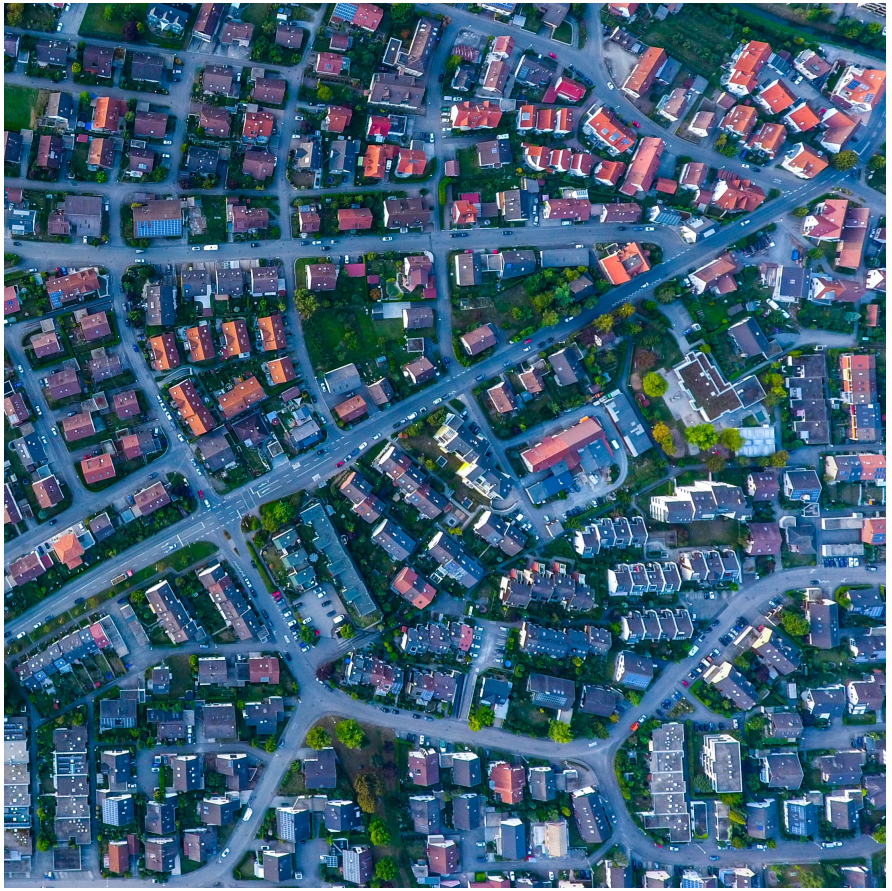
These equations define the image formation process called perspective projection.

Equivalently we can model the perspective projection by placing the image plane in front of the COP, thus eliminating the negative sign.



The division by Z is responsible for the effect of foreshortening, whereby the size of the image of an object varies in proportion of its distance from the observer.

If the framed object is relatively thin, compared with its average distance from the camera, one can approximate the perspective projection with the (scaled) orthographic projection or *weak perspective*.



Digital images

In a digital camera, the picture is made up of a CCD (*Charge-Coupled Device*) or CMOS (*Complementary Metal-Oxide Semiconductor*) matrix.

We can see the CCD (or CMOS) as a matrix of photosensitive rectangular cells, each of which converts the intensity of the incident light radiation into an electric potential.

This matrix is converted into a digital image, i.e., in an array of integer values. The elements of this matrix are called pixels (*picture elements*).

The process is not one-to-one: a pixel corresponds to a rectangular area on the CCD (also called footprint of the pixel), not necessarily equal to one cell of the CCD. Its size (of the order of microns) is the pixel effective size.

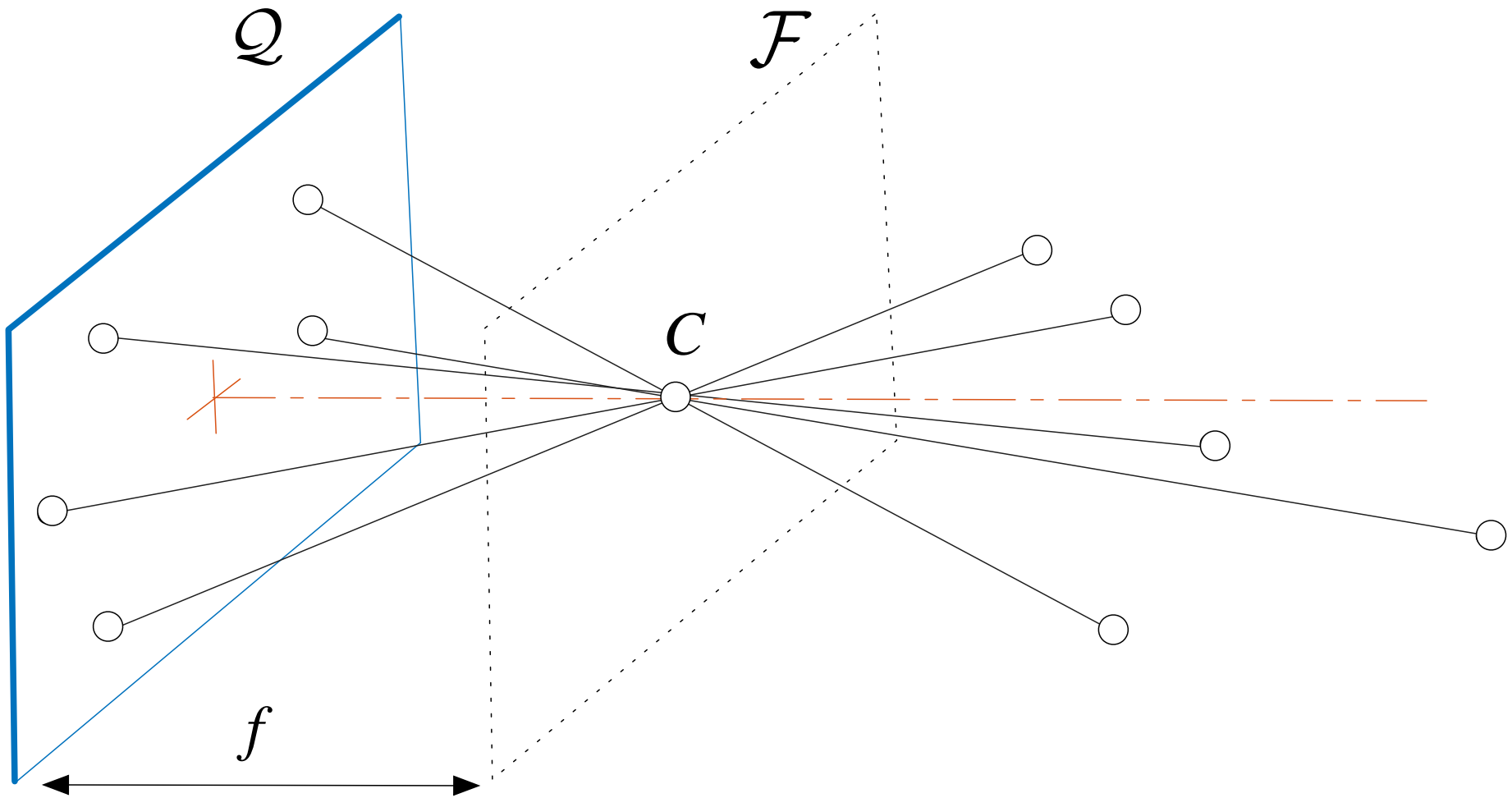
3 Pinhole camera model

The geometric model of the camera we will adopt is the so-called pinhole or perspective model, which consists of an image plane $\mathcal{Q}$ and a centre of projection C, at a distance f (focal length) from $\mathcal{Q}$.

The bundle of straight lines centered at C is also called the projective bundle.

The line passing through C orthogonal to $\mathcal{Q}$ is the optical/principal axis and its intersection with $\mathcal{Q}$ is the principal point.

Plane $\mathcal{F}$ parallel to $\mathcal{Q}$ and containing the center of projection is called focal/principal plane. The points of that plane are projected to infinity on the image.


$\mathcal{Q}$
$\mathcal{F}$
C
f

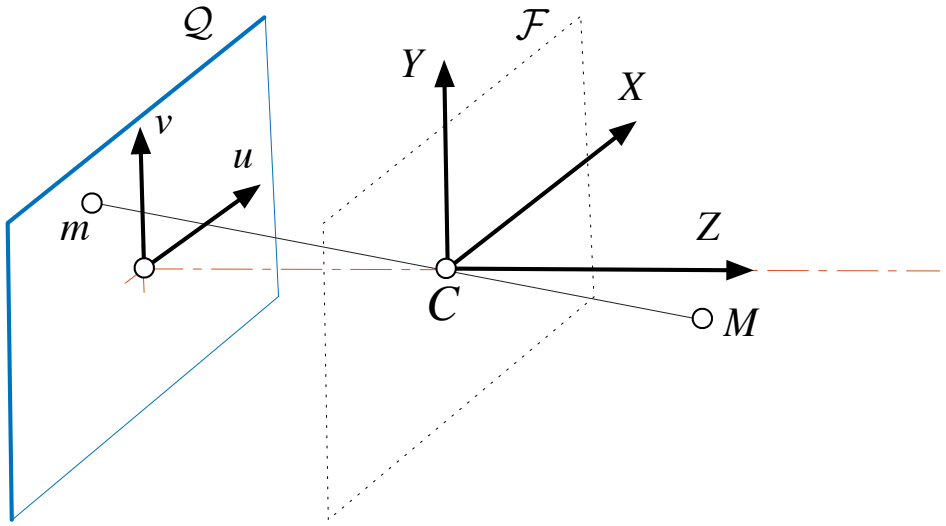
Simplified model

The reference system in which the coordinates of the 3-D points are expressed is called world or object system.

We introduce also the (standard) camera reference system, centered in C and with the axis Z coinciding with optical axis.

Initially, let us chose the world system coincident with the camera system.

We also introduce a reference system (u, v) for the plane $\mathcal{Q}$ centered in the principal point and with the axes u and v oriented as X and Y respectively.



Let us now consider a point M with coordinates $\widetilde{\mathbf{M}} = [X, Y, Z]^{\mathsf{T}}$ in 3D space and its projection m on $\mathcal{Q}$ through C with coordinates $\widetilde{\mathbf{m}} = [u, v]^{\mathsf{T}}$.

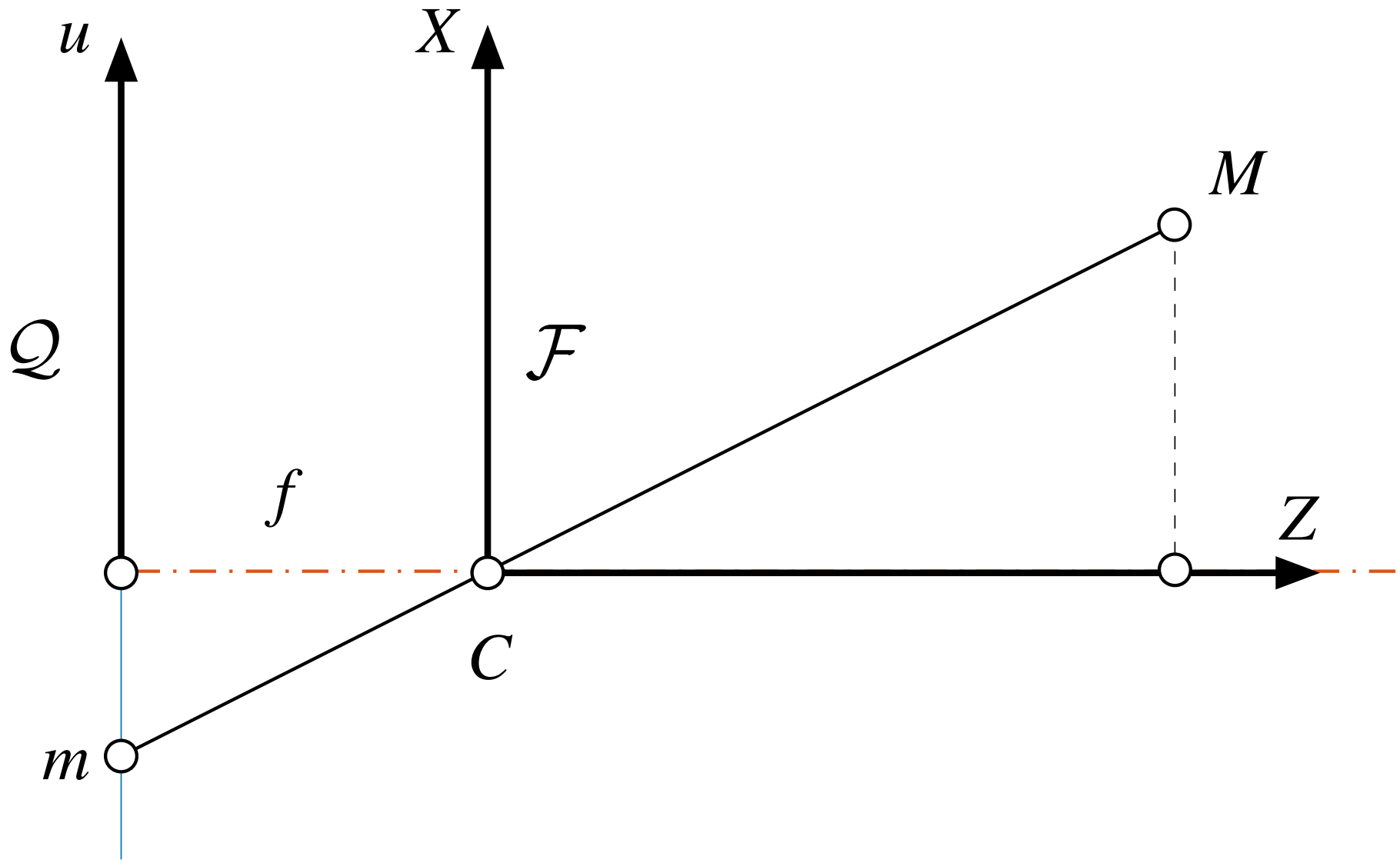
By simple considerations on the similarity of triangles, we arrive at the following relationship:

$$\frac{f}{Z} = \frac{-u}{X} = \frac{-v}{Y}, \tag{3}$$

that is

$$\begin{cases} u = \dfrac{-f}{Z} X \\ v = \dfrac{-f}{Z} Y. \end{cases} \tag{4}$$

This is the perspective projection. The transformation from 3D to 2D coordinates is clearly non-linear (because of the division by Z).


u
X
M
Q
F
f
Z
C
m

Interlude:homogeneous coordinates

Two lines of equations $ax + by + c = 0$ and $a'x + b'y + c' = 0$ meet, if they are not parallel, in point of $\mathbb{A}^2$ of coordinates (Cramer's rule)

$$x = -\frac{\det\left(\begin{bmatrix} c & b \\ c' & b' \end{bmatrix}\right)}{\det\left(\begin{bmatrix} a & b \\ a' & b' \end{bmatrix}\right)} \qquad y = -\frac{\det\left(\begin{bmatrix} a & c \\ a' & c' \end{bmatrix}\right)}{\det\left(\begin{bmatrix} a & b \\ a' & b' \end{bmatrix}\right)}. \tag{5}$$

When the lines are parallel the denominator cancels, so we cannot make sense of the ratio (it becomes "infinite"). We note, however, that of the three quantities that appear in (5), at least one is non-zero (if the lines are distinct). We then propose to keep these three quantity as a representation of the points of $\mathbb{P}^2$.

Instead of representing the points of the plane through pairs of coordinates (x, y), we agree to represent them with triples of homogeneous coordinates (u, v, w), linked to the Cartesian coordinates of $\mathbb{A}^2$ by the relation $x = u/w$ and $y = v/w$.

We note that proportional triples represent the same point - this is why they are called homogeneous - and that the triple $(0, 0, 0)$ is excluded.

When $w \neq 0$ the triple represents a proper point (finite), while each triple with $w = 0$ represents an improper point (at infinity).

Definition 3.1 (Projective plane) *The projective plane $\mathbb{P}^2$ consists of triples of real numbers $(u, v, w) \neq (0, 0, 0)$ modulo the following relation of equivalence: $(u, v, w) \sim \lambda(u, v, w)\ \forall \lambda \neq 0$. The triple (u, v, w) are the homogeneous coordinates of the point of $\mathbb{P}^2$.*

Similarly, the projective space $\mathbb{P}^n$ can be defined .

Example: rigid body transform in homogeneous coordinates.

$$G = \begin{bmatrix} R & \mathbf{t} \\ \mathbf{0} & 1 \end{bmatrix} \tag{6}$$

where R is orthogonal. The action of this matrix in the projective space $\mathbb{P}^3$ is

$$\mathbf{x} \rightarrow G\mathbf{x} \tag{7}$$

which correspond in $\mathbb{R}^3$ to $\tilde{\mathbf{x}} \rightarrow R\tilde{\mathbf{x}} + \mathbf{t}$ where $\mathbf{x} = [\tilde{\mathbf{x}}, 1]^{\mathsf{T}}$.

Using instead homogeneous coordinates it becomes linear.

So let

$$\mathbf{m} = \begin{bmatrix} u \\ v \\ 1 \end{bmatrix} \quad \text{e} \quad \mathbf{M} = \begin{bmatrix} X \\ Y \\ Z \\ 1 \end{bmatrix} \tag{8}$$

be the homogeneous coordinates of m and M respectively.

Hence the equation of perspective projection, in this simplified case, rewrites:

$$Z \begin{bmatrix} u \\ v \\ 1 \end{bmatrix} = Z \begin{bmatrix} -fX/Z \\ -fY/Z \\ 1 \end{bmatrix} = \begin{bmatrix} -fX \\ -fY \\ Z \end{bmatrix} = \underbrace{\begin{bmatrix} -f & 0 & 0 & 0 \\ 0 & -f & 0 & 0 \\ 0 & 0 & 1 & 0 \end{bmatrix}}_{P} \begin{bmatrix} X \\ Y \\ Z \\ 1 \end{bmatrix}. \tag{9}$$

Moving on to matrix notation:

$$\mathbf{m} = \frac{1}{Z} P\mathbf{M} \tag{10}$$

or also:

$$\mathbf{m} \simeq P\mathbf{M} \tag{11}$$

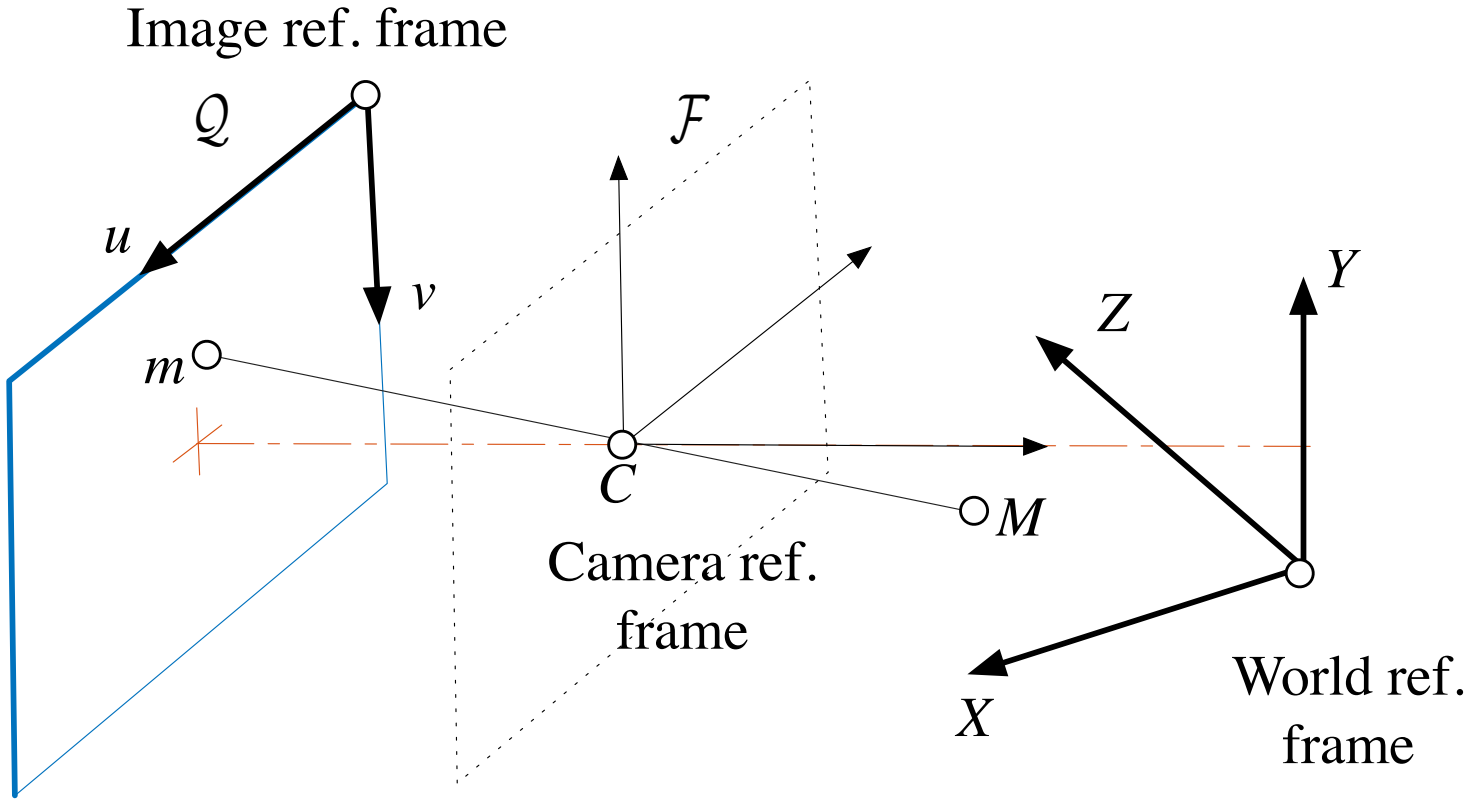
where $\simeq$ means " equal up to a scale factor ".

The matrix P represents the geometric model of the camera and is called perspective projection matrix (PPM) or camera matrix.

General model

A realistic model of a camera must take into account:

- change of coordinates between the camera reference and the absolute reference;
- pixelization, i.e., the change of coordinates from meters in the sensor plane to pixels in the image.



Intrinsic parameters

Pixelization is catered for by means of an affine transformation A which takes into account the translation of the origin in the upper left corner, and the (independent) rescaling of the axes u and v for changing the unit of measurement from meters to pixels (with the inversion of the direction of the axes):

$$A = \begin{bmatrix} -k_u & 0 & u_0 \\ 0 & -k_v & v_0 \\ 0 & 0 & 1 \end{bmatrix} \tag{12}$$

where (u_0, v_0) are the pixel coordinates of the principal point, k_u (k_v) is the inverse of the effective pixel size along the direction u (v), measured in pixel $\cdot$ m^{-1}.

The PPM after this update it becomes:

$$P = \begin{bmatrix} -k_u & 0 & u_0 \\ 0 & -k_v & v_0 \\ 0 & 0 & 1 \end{bmatrix} \begin{bmatrix} -f & 0 & 0 & 0 \\ 0 & -f & 0 & 0 \\ 0 & 0 & 1 & 0 \end{bmatrix} = \begin{bmatrix} fk_u & 0 & u_0 & 0 \\ 0 & fk_v & v_0 & 0 \\ 0 & 0 & 1 & 0 \end{bmatrix} = K[I|\mathbf{0}] \tag{13}$$

where

$$K = \begin{bmatrix} \alpha_u & 0 & u_0 \\ 0 & \alpha_v & v_0 \\ 0 & 0 & 1 \end{bmatrix} \quad \text{e} \quad [I|\mathbf{0}] = \begin{bmatrix} 1 & 0 & 0 & 0 \\ 0 & 1 & 0 & 0 \\ 0 & 0 & 1 & 0 \end{bmatrix} \tag{14}$$

with $\alpha_u = f k_u$ and $\alpha_v = f k_v$ (focal length expressed in horizontal and vertical pixels respectively). The matrix K is called intrinsic parameters matrix. These parameters are therefore the following four: $\alpha_u, \alpha_v, u_0, v_0$.

The matrix $[I|\mathbf{0}]$ encodes the essence of the perspective transformation, without any parameters.

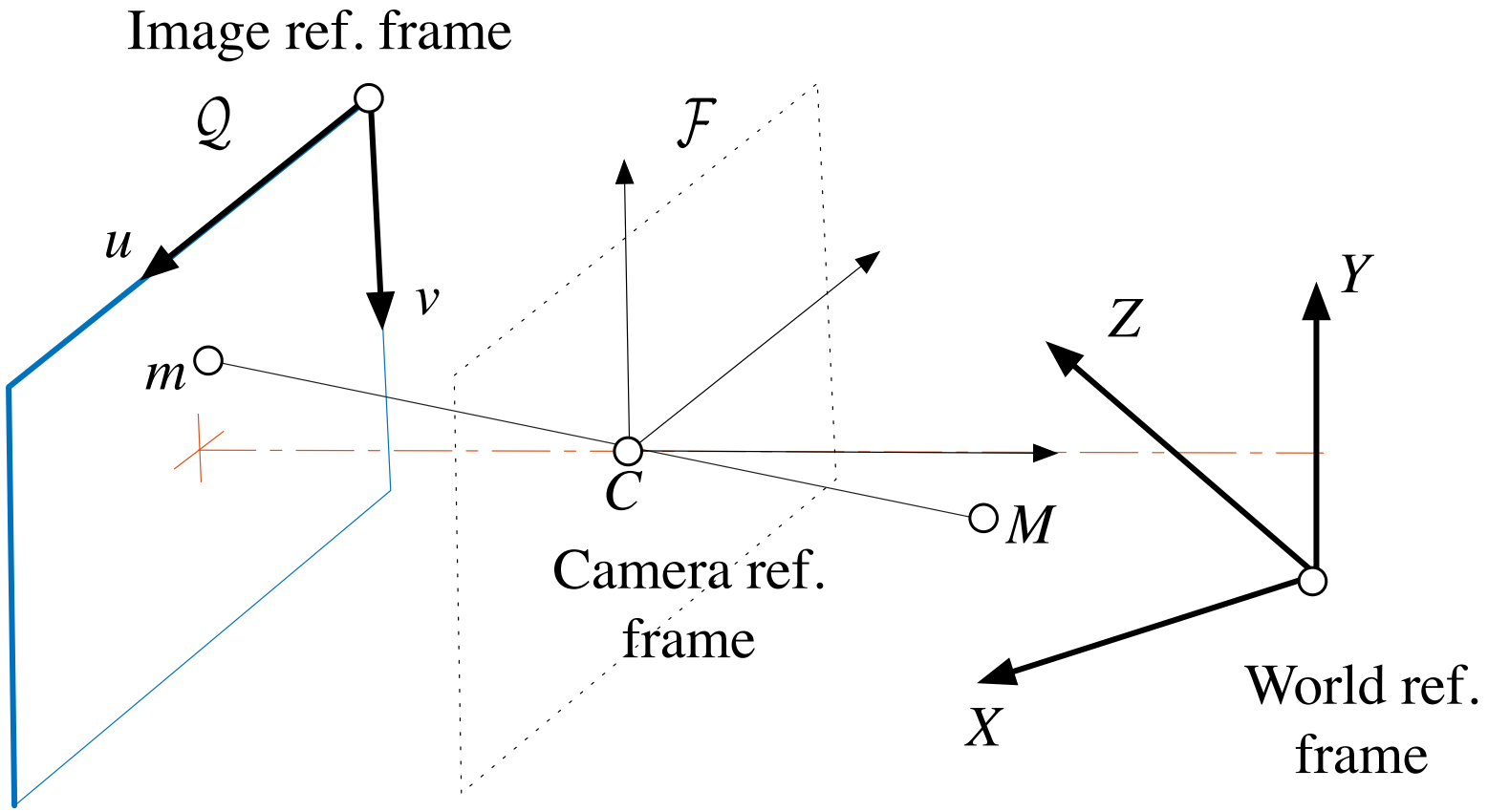
It is obtained when the world reference coincides with the camera reference and image points are expressed in normalized image coordinates, ie

$$\mathbf{q} = K^{-1}\mathbf{m}. \tag{15}$$

Note that while pixel coordinates are always measurable in the image, to step to normalized coordinates one has to know the intrinsic parameters.

Extrinsic parameters

We introduce now a change of coordinate consisting of a direct isometry, i.e., a rotation R followed by a translation $\mathbf{t}$. We denote by $\mathbf{M}_c$ the homogeneous coordinates of a point in the reference system of the camera and with $\mathbf{M}$ the homogeneous coordinates of the same point in the world reference system.



We can therefore write:

$$\mathbf{M}_c = G\mathbf{M} \tag{16}$$

where is it

$$G = \begin{bmatrix} R & \mathbf{t} \\ \mathbf{0} & 1 \end{bmatrix}. \tag{17}$$

The matrix G encodes the exterior orientation (position and angular attitude) of the camera wrt to the world reference. In particular, the rows of R are the X, Y and Z axes of the camera system expressed in the world reference system.

By appropriately parameterizing the rotation R, exterior orientation is coded by six extrinsic parameters: three Euler angles ω, φ, κ to specify the rotation and three components t_1, t_2, t_3 for the translation.

From equation (13) we obtain:

$$\mathbf{m} \simeq K[I|\mathbf{0}]\mathbf{M}_c \tag{18}$$

and replacing (16) in the above equation we have:

$$\mathbf{m} \simeq K[I|\mathbf{0}]G\mathbf{M}, \tag{19}$$

and so

$$P = K[I|\mathbf{0}]G. \tag{20}$$

This is the most general form of the perspective projection matrix:

- the G matrix encodes the exterior orientation of the camera,
- the matrix K encodes its intrinsic parameters (by analogy, we also speak of interior orientation),
- the matrix $[I|\mathbf{0}]$ represents a perspective transformation in normalized coordinates in the camera reference system.

Depending on the convenience, we may also consider the following expression for the PPM:

$$P = K[R|\mathbf{t}] \tag{21}$$

which is obtained by replacing in (20) the block form of G and developing the product with the central matrix.

By letting

$$\mathbf{t} = \begin{bmatrix} t_1 \\ t_2 \\ t_3 \end{bmatrix} \quad \text{e} \quad R = \begin{bmatrix} \mathbf{r}_1^\mathsf{T} \\ \mathbf{r}_2^\mathsf{T} \\ \mathbf{r}_3^\mathsf{T} \end{bmatrix} \tag{22}$$

we obtain the following expression for P as a function of the elements of matrices K and G:

$$P = \begin{bmatrix} \alpha_u \mathbf{r}_1^\mathsf{T} + u_0 \mathbf{r}_3^\mathsf{T} & \alpha_u t_1 + u_0 t_3 \\ \alpha_v \mathbf{r}_2^\mathsf{T} + v_0 \mathbf{r}_3^\mathsf{T} & \alpha_v t_2 + v_0 t_3 \\ \mathbf{r}_3^\mathsf{T} & t_3 \end{bmatrix} . \tag{23}$$

The rows of R could be further exploded in sines and cosines of the angles ω, φ, κ.

An PPM is defined up to a arbitrary scale factor. In fact, if we replace P with λP in (11) the same projection is obtained, for each real λ not null.

So, if one wants to bring a generic PPM into the form of (23) it is necessary to rescale it in so that the vector of three elements corresponding to $\mathbf{r}_3$ have unit length. We will call such a matrix normalized.

The matrix P is made up of 12 elements but it has 11 degrees of freedom because it loses one due to the scale factor: this implies that it must depend on 11 independent parameters.

However, we have highlighted only $6 + 4 = 10$. In fact, one more intrinsic parameter is missing, ϑ, the angle between the axes u and v. The matrix K then becomes:

$$K = \begin{bmatrix} f k_u & -f k_u / \tan\vartheta & u_0 \\ 0 & f k_v / \sin\vartheta & v_0 \\ 0 & 0 & 1 \end{bmatrix}. \tag{24}$$

Equivalently, the skew parameter is often indicated $\gamma = -f k_u / \tan\vartheta$. Normally it is reasonable to assume $\vartheta = \pi/2$, ie $\gamma = 0$.

Listing 1. Projective transformation

```
function Y = htx( T, X )
%HTX Apply homogeneous transform
%   Apply transformation T in homogeneous coordinates
%   T can be any dimension
%   T 3x4 is a projection
%   T 3x3 is a transformation of the projective plane
%   T 4x4 is a transformation of the projective space

Y = T * [X; ones(1, size(X,2))]; % apply tranform

nr = size(Y,1);

Y = Y ./ repmat(Y(nr,:),nr,1); % perspective division

Y(nr,:) = []; % remove trailing ones

end
```

Go to live coding.

Dissection of the PPM

If we write the PPM according to its rows

$$P = \begin{bmatrix} \mathbf{p}_1^\top \\ \mathbf{p}_2^\top \\ \mathbf{p}_3^\top \end{bmatrix} \tag{25}$$

and we insert it in (11) we get:

$$\mathbf{m} \simeq \begin{bmatrix} \mathbf{p}_1^\top \mathbf{M} \\ \mathbf{p}_2^\top \mathbf{M} \\ \mathbf{p}_3^\top \mathbf{M} \end{bmatrix} = \begin{bmatrix} \mathbf{p}_1^\top \\ \mathbf{p}_2^\top \\ \mathbf{p}_3^\top \end{bmatrix} \mathbf{M} \tag{26}$$

and therefore the perspective projection equation becomes:

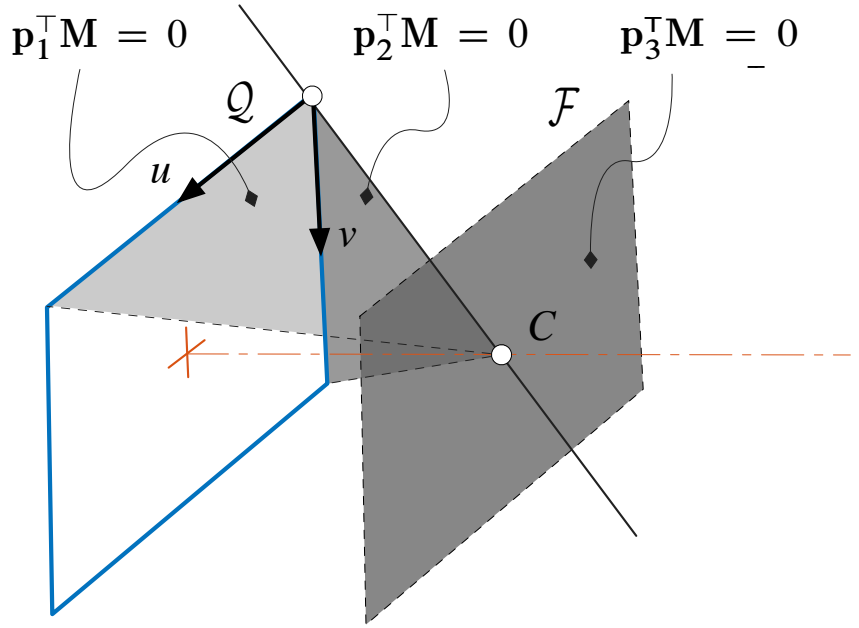
$$\begin{cases} u = \dfrac{\mathbf{p}_1^\top \mathbf{M}}{\mathbf{p}_3^\top \mathbf{M}} \\ v = \dfrac{\mathbf{p}_2^\top \mathbf{M}}{\mathbf{p}_3^\top \mathbf{M}}. \end{cases} \tag{27}$$

This is the generalization of (4).

The focal plane has equation $\mathbf{p}_3^\top \mathbf{M} = 0$ and it contains the points that project to infinity. The planes with equation $\mathbf{p}_1^\top \mathbf{M} = 0$ and $\mathbf{p}_2^\top \mathbf{M} = 0$ project into the image on the axes $u = 0$ and $v = 0$ respectively.

The centre of projection (COP) C belongs to all these three these planes, so it is defined by their intersection:

$$\begin{cases} \mathbf{p}_1^\top \mathbf{C} = 0 \\ \mathbf{p}_2^\top \mathbf{C} = 0 \\ \mathbf{p}_3^\top \mathbf{C} = 0 \end{cases} \tag{28}$$



In other terms:

$$P\mathbf{C} = \mathbf{0}.$$

Hence, the COP is the nullspace of P, which, by the rank-nullity theorem, has size one.

The COP is the only point in the space for which the projection is not defined, and in fact $\mathbf{0}$ is the only triple that does not represent any point of the projective plane.

If we partition P as $P = [\widetilde{P}|\mathbf{p}_4]$ and $\mathbf{C} = [\widetilde{\mathbf{C}}, 1]^\mathsf{T}$ the system of equations (28) is rewritten as:

$$\widetilde{P}\widetilde{\mathbf{C}} + \mathbf{p}_4 = \mathbf{0}. \tag{29}$$

The COP is therefore obtained as:

$$\widetilde{\mathbf{C}} = -\widetilde{P}^{-1}\mathbf{p}_4. \tag{30}$$

Note that setting $P = K[R|\mathbf{t}]$ in the previous equation we obtain the relationship between the COP and the translation component of the extrinsic parameters:

$$\widetilde{\mathbf{C}} = -R^\mathsf{T}\mathbf{t}. \tag{31}$$

The optical ray of point m is the straight line that passes through the projection center C and m itself, i.e., the geometric locus of points $\mathbf{M} : \mathbf{m} \simeq P\mathbf{M}$.

On the optical ray of $\mathbf{m}$ lie all the points of the space of which the point m is projection.

A point belonging to this line is the projection center C, by definition.

Another is the (ideal) point

$$\begin{bmatrix} \widetilde{P}^{-1}\mathbf{m} \\ 0 \end{bmatrix},$$

in fact it is sufficient to project this point to verify that:

$$P \begin{bmatrix} \widetilde{P}^{-1}\mathbf{m} \\ 0 \end{bmatrix} = \widetilde{P}\widetilde{P}^{-1}\mathbf{m} = \mathbf{m}. \tag{32}$$

The parametric equation of the optical ray of $\mathbf{m}$ is therefore the following:

$$\mathbf{M} = \mathbf{C} + \lambda \begin{bmatrix} \widetilde{P}^{-1}\mathbf{m} \\ 0 \end{bmatrix}, \quad \lambda \in \mathbb{R} \cup \{\infty\}. \tag{33}$$

4 Camera Calibration

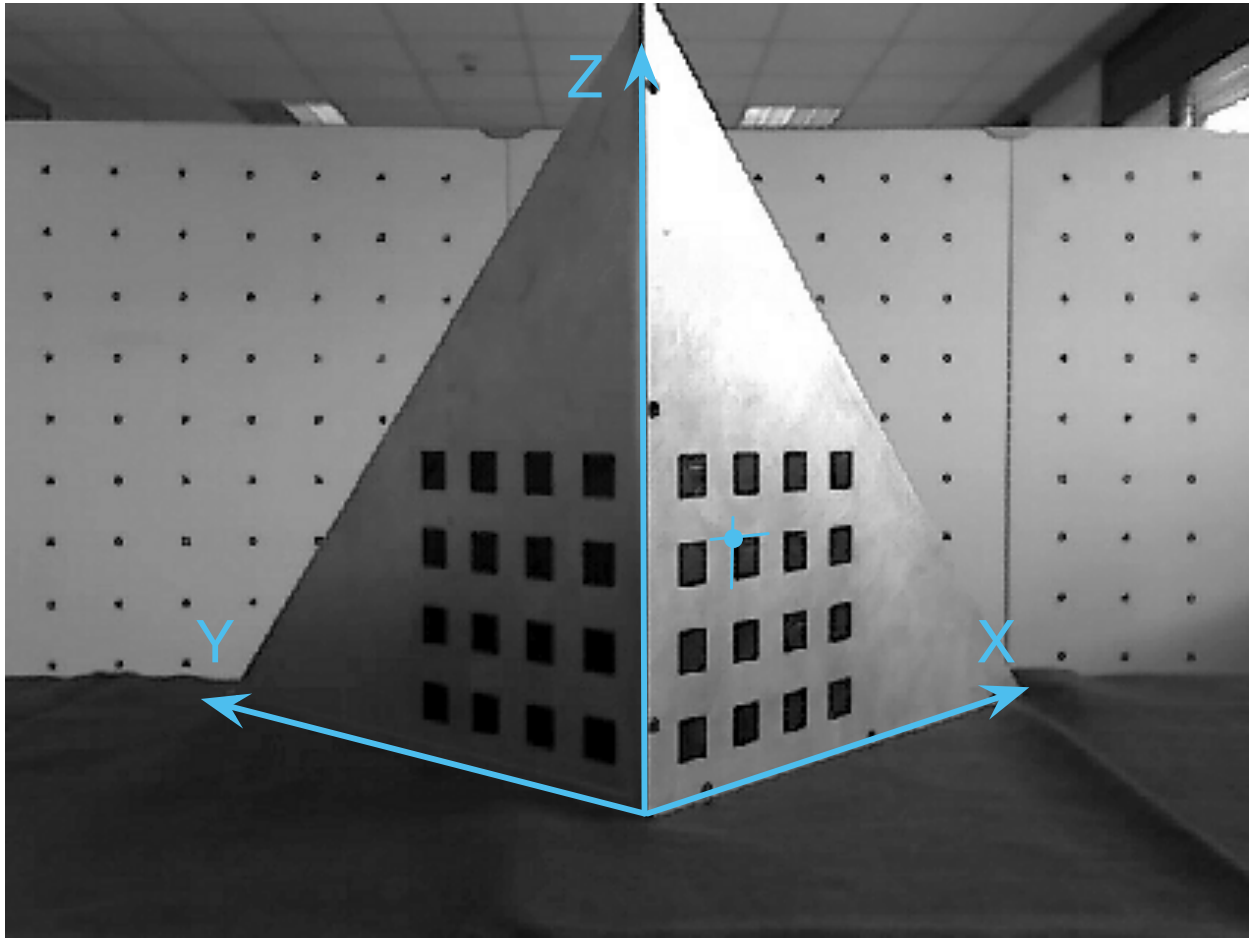
Calibration = compute intrinsic and extrinsic parameters of the camera

There are different calibration techniques, those that use only one image and the knowledge of some 3-D points are known as (spatial) resection.

Calibration by resection

The specific algorithm that we will see is called DLT (Direct Linear Transform).

The points of known coordinates on the object are called (ground) control points.



Given the correspondences between n not coplanar control points $\mathbf{M}_i$ and their projections in the image $\mathbf{m}_i$ it is required to compute the matrix P such that:

$$\mathbf{m}_i \simeq P\mathbf{M}_i \quad i = 1 \ldots n. \tag{34}$$

To eliminate the scale factor, the cross product is used, rewriting the previous equation as:

$$\mathbf{m}_i \times P\mathbf{M}_i = \mathbf{0}. \tag{35}$$

Interlude: cross product, Kronecker product

Definition 4.1 (Cross product) *The cross product of two vectors* $\mathbf{a}, \mathbf{b} \in \mathbb{R}^3$ *is defined as the vector:*

$$\mathbf{a} \times \mathbf{b} = \begin{bmatrix} \det \begin{bmatrix} a_2 & a_3 \\ b_2 & b_3 \end{bmatrix} \\ -\det \begin{bmatrix} a_1 & a_3 \\ b_1 & b_3 \end{bmatrix} \\ \det \begin{bmatrix} a_1 & a_2 \\ b_1 & b_2 \end{bmatrix} \end{bmatrix}. \tag{36}$$

The cross product serves (among other things) to check if two vectors differ only by a multiplicative constant:

Remark 4.2 *Given two vectors* $\mathbf{a}, \mathbf{b} \in \mathbb{R}^3$, $\mathbf{a}, \mathbf{b}, \neq \mathbf{0}$ *we have:* $\mathbf{a} \times \mathbf{b} = \mathbf{0} \iff \mathbf{a} = \lambda \mathbf{b}, \lambda \in \mathbb{R}$.

Indeed, the hypothesis is equivalent to stating that all two order determinants extracted from $[\mathbf{a}, \mathbf{b}]$ are null, therefore it has rank one, from which the thesis follows, and vice versa.

The cross product is naturally associated with a antisymmetric matrix:

Remark 4.3 *Given a vector* $\mathbf{a} \in \mathbb{R}^3$, *the matrix*

$$[\mathbf{a}]_\times = \begin{bmatrix} 0 & -a_3 & a_2 \\ a_3 & 0 & -a_1 \\ -a_2 & a_1 & 0 \end{bmatrix} \tag{37}$$

acts as the cross product by $\mathbf{a}$, *that is:* $[\mathbf{a}]_\times \mathbf{b} = \mathbf{a} \times \mathbf{b}$.

The matrix $[\mathbf{a}]_\times$ is antisymmetric and has rank two, because it must be even.

Also: $\ker([\mathbf{a}]_\times) = \mathbf{a}$, since $[\mathbf{a}]_\times \mathbf{a} = \mathbf{a} \times \mathbf{a} = \mathbf{0}$.

Definition 4.4 (Kronecker product) *Let A be a $m \times n$ matrix and B be a $p \times q$ matrix. of A and B is the $mp \times nq$ matrix defined by*

$$A \otimes B = \begin{bmatrix} a_{11}B & \dots & a_{1n}B \\ \vdots & & \vdots \\ a_{m1}B & \dots & a_{mn}B \end{bmatrix}. \tag{38}$$

Proposition 4.5 (Kronecker Product Rank)

$$\mathrm{r}(A \otimes B) = \mathrm{r}(A)\,\mathrm{r}(B). \tag{39}$$

Definition 4.6 (Vectorization) *The vectorization of a matrix A $m \times n$, denoted by $\mathrm{vec}(A)$, is the vector $mn \times 1$ which is obtained by stacking the columns of A one below the other.*

The vec-trick The connection between Kronecker's product and vectorization is given by the following (important) relation:

$$\mathrm{vec}(AXB) = (B^{\mathsf{T}} \otimes A)\,\mathrm{vec}(X) \tag{40}$$

for matrices A, B, X of compatible size.

Given the correspondences between n not coplanar control points $\mathbf{M}_i$ and their projections in the image $\mathbf{m}_i$ it is required to compute the matrix P such that:

$$\mathbf{m}_i \simeq P\mathbf{M}_i \quad i = 1 \dots n. \tag{41}$$

To eliminate the scale factor, the cross product is used, rewriting the previous equation as:

$$\mathbf{m}_i \times P\mathbf{M}_i = \mathbf{0}. \tag{42}$$

Then, using the "vec-trick":

$$\begin{aligned} \mathbf{m}_i \times P\mathbf{M}_i = \mathbf{0} &\iff [\mathbf{m}_i]_\times P\mathbf{M}_i = \mathbf{0} \iff \\ \mathrm{vec}([\mathbf{m}_i]_\times P\mathbf{M}_i) = \mathbf{0} &\iff (\mathbf{M}_i^\mathsf{T} \otimes [\mathbf{m}_i]_\times)\,\mathrm{vec}(P) = \mathbf{0}. \end{aligned} \tag{43}$$

There are three equations in 12 unknowns, but only two of them are independent, in fact the rank of $(\mathbf{M}_i^\mathsf{T} \otimes [\mathbf{m}_i]_\times)$ is two since it is the Kronecker product of of a rank one matrix with a rank two matrix.

For n points we obtain a system of $2n$ homogeneous linear equations, that we can write as

$$A\,\mathrm{vec}(P) = \mathbf{0} \tag{44}$$

where A is the $2n \times 12$ matrix of the coefficients and depends on the coordinates of the control points, while the vector of unknowns $\mathrm{vec}(P)$ contains the 12 elements of P (read by columns).

In theory six non-coplanar points are sufficient for the computation of P;

In practice many more points are available, and it is advisable use them to compensate for the inevitable measurement errors.

The system (44) is therefore solved in the least squares sense. How?

Interlude: Least-squares solution to a linear system of homogeneous equations

«If you were to be stranded on a deserted island, make sure to bring SVD with you.»

Theorem 4.7 (Singular Values Decomposition) *Let A be a $m \times n$ matrix. There exists a $m \times n$ matrix D with positive diagonal elements $\sigma_1 \geq \sigma_2 \geq \ldots \sigma_n \geq 0$ and null elements elsewhere, an orthogonal $m \times m$ matrix U and an orthogonal $n \times n$ matrix V such that:*

$$U^{\mathsf{T}} A V = D. \tag{45}$$

The diagonal elements of D are called singular values.

Proposition 4.8 (Kernel and image of the matrix) *Let A be a $m \times n$ matrix and let $U^{\mathsf{T}} A V = D$ be its singular values decomposition. The columns of U corresponding to non-null singular values are a basis for* $\mathrm{im}(A)$*, while the columns of V corresponding to the null singular values are a basis for* $\ker(A)$*.*

Proposition 4.1 () *Let A e a $m \times n$ matrix. The solution of the least-squares problem*

$$\min_{\|\mathbf{x}\|=1} \|A\mathbf{x}\|^2 \tag{46}$$

is $\mathbf{x} = \mathbf{v}_n$ where $\mathbf{v}_n$ is the last column of V in the singular values decomposition of A: $U^{\mathsf{T}}AV = D$.

Proof. $\min_{\|\mathbf{x}\|=1}\|A\mathbf{x}\|^2 = \min_{\|\mathbf{x}\|=1}\|UDV^{\mathsf{T}}\mathbf{x}\|^2 = \min_{\|\mathbf{x}\|=1}\|DV^{\mathsf{T}}\mathbf{x}\|^2$. After making the variable change $\mathbf{y} = V^{\mathsf{T}}\mathbf{x}$ (recall that V is orthogonal), we get:

$$\min_{\|\mathbf{x}\|=1} \|A\mathbf{x}\|^2 = \min_{\|\mathbf{y}\|=1} \|D\mathbf{y}\|^2 = \min_{\|\mathbf{y}\|=1} \sum (\sigma_i y_i)^2. \tag{47}$$

The constraint prevent us from choosing $\mathbf{y} = \mathbf{0}$, and since since the singular values in D are ordered, the least-cost solution is $\mathbf{y} = [0, \ldots, 0, 1]^{\mathsf{T}}$, so $\mathbf{x} = \mathbf{v}_n$.

We will refer to $\mathbf{v}_n$ as the least right singular vector of A.

Back to the DLT, the solution of (44) is the least right singular vector in the Singular Value Decomposition (SVD) of A.

Listing 2. DLT

```
function P = dlt( x, X, w)
%DLT Direct Linear Transform

    if nargin <3, w = ones(size(X,2),1); end % weights

     X = [X; ones(1, size(X,2))];
     x = [x; ones(1, size(x,2))];

     L = [];
     for i = 1: size(X,2)
         L = [L; w(i) * kron( X(:,i)', skew(x(:,i)) )];
     end

     [~,~,V] = svd(L);
     P = reshape(V(:,end),size(x,1),[]);
end
```

Go to live coding.

Factorization of PPM

Given a matrix P 3×4 of full rank, we want to decompose it in

$$P \simeq K[R|\mathbf{t}]. \tag{48}$$

Let's focus on submatrix $\widetilde{P}$ in $P = [\widetilde{P}|\mathbf{p}_4]$. By comparison with $P = [KR|K\mathbf{t}]$, we have $\widetilde{P} = KR$ with K upper triangular and R orthogonal (rotation). Let

$$\widetilde{P}^{-1} = QU \tag{49}$$

the QR factorization of $\widetilde{P}^{-1}$, with Q orthogonal and U upper triangular. Being $\widetilde{P}^{-1} = R^{-1}K^{-1}$, it suffices to set

$$R = \det(Q)Q^{\mathsf{T}} \quad \text{and} \quad K = U^{-1} \tag{50}$$

where the product by $\det(Q)$ serves to make the determinant of R positive. We can change sign to R because this translates into the change of sign to P, which can absorb any scale factor.

For the same reason, we can rescale K by imposing $K(3,3) = 1$.

The translation $\mathbf{t}$ is calculated with

$$\mathbf{t} = \det(Q)K^{-1}\mathbf{p}_4 = \det(Q)U\mathbf{p}_4, \tag{51}$$

Listing 3. Factorization of the PPM

```
function [K,R,t] = krt (P )
%KRT Intrinsic and extrinsic parameters from P

[Q,U] = qr(inv(P(1:3, 1:3)));
% enforce negative focal
D = diag(sign(diag(U)).*[-1;-1;1]);
Q = Q*D;  U = D*U;
% fix sign of R
s = det(Q);
R = s*Q';
t = s*U*P(1:3,4);
K =  inv(U./U(3,3));
```

Go to live coding.

Sturm-Maybank-Zhang method for calibration

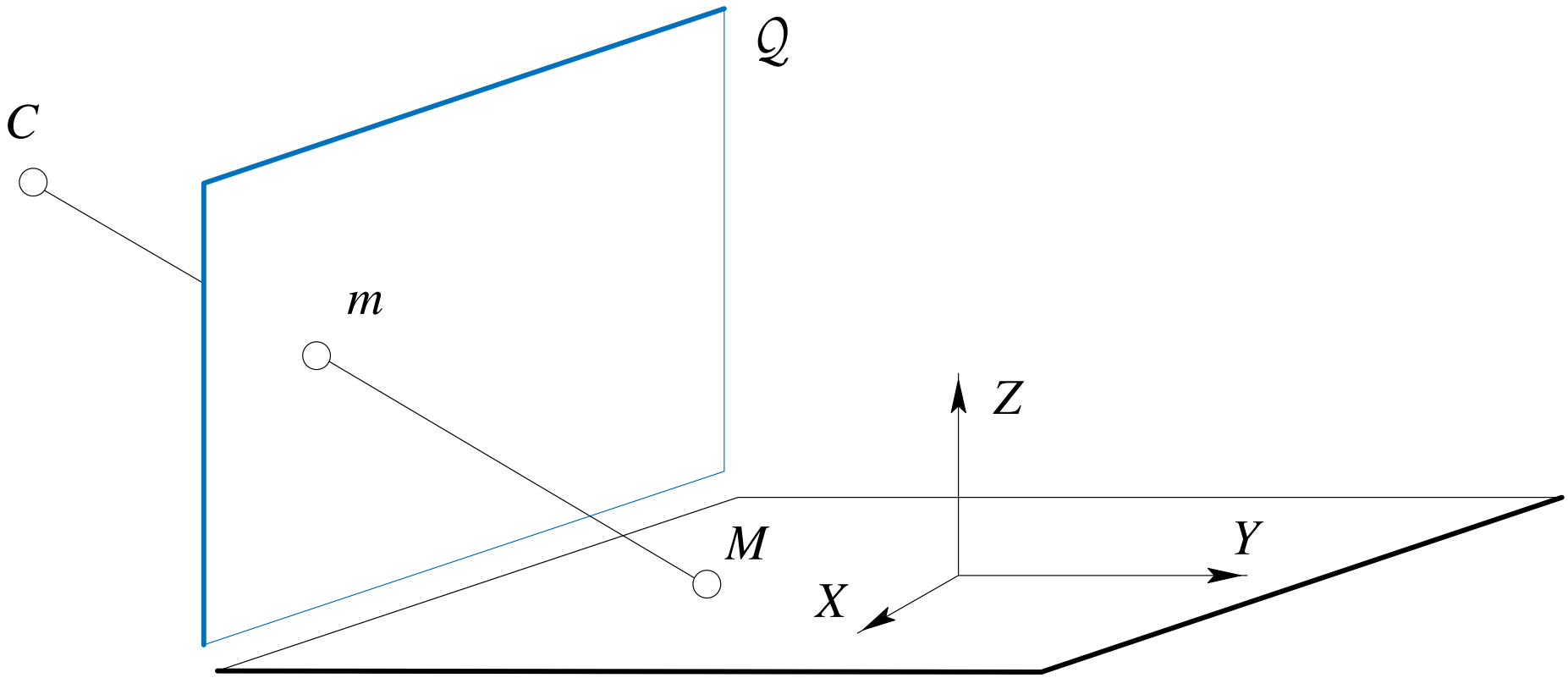
The Sturm-Maybank-Zhang method of calibration (Sturm and Maybank, 1999; Zhang, 2000) is based on many (at least 3) images of a plane, as opposed to resection which relies on one image of many (at least 2) planes.

We begin by establishing that the application between a plane Π in space and its perspective image is a homography.

Writing the perspective projection equation for a point belonging to the plane $Z = 0$, we obtain

$$\mathbf{m} \simeq \underbrace{\left[\mathbf{p}_1, \mathbf{p}_2, \mathbf{p}_3, \mathbf{p}_4\right]}_{P} \begin{bmatrix} X \\ Y \\ 0 \\ 1 \end{bmatrix} = \underbrace{\left[\mathbf{p}_1, \mathbf{p}_2, \mathbf{p}_4\right]}_{H} \begin{bmatrix} X \\ Y \\ 1 \end{bmatrix} \tag{52}$$

where we have represented P with its columns: $P = \left[\mathbf{p}_1, \mathbf{p}_2, \mathbf{p}_3, \mathbf{p}_4\right]$.

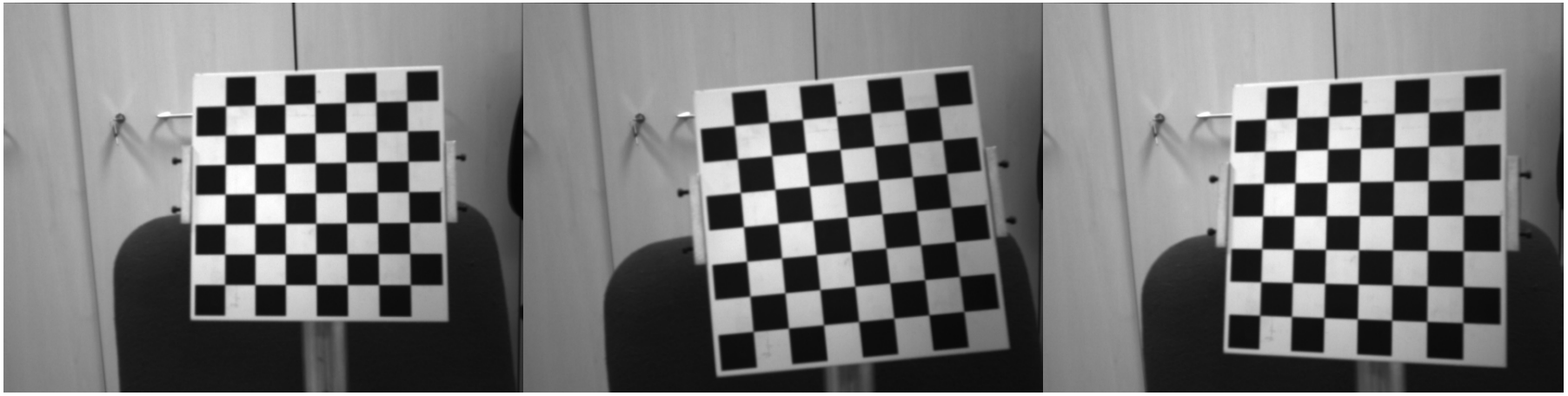

C
Q
m
M
Z
Y
X

The association between the points in the image m and the corresponding points M in the calibration grid are assigned.

If we assume for simplicity that the plane of the chessboard has equation $Z = 0$, the homography H between the image and the plane of the chessboard is:

$$\mathbf{m} \simeq \underbrace{\left[\mathbf{p}_1, \mathbf{p}_2, \mathbf{p}_4\right]}_{H} \begin{bmatrix} X \\ Y \\ 1 \end{bmatrix} \tag{53}$$

and can be computed with DLT.



Considering that $P = K[R, \mathbf{t}]$, and setting $R = [\mathbf{r}_1, \mathbf{r}_2, \mathbf{r}_3]$, we have:

$$H = \lambda K[\mathbf{r}_1, \mathbf{r}_2, \mathbf{t}] \tag{54}$$

where λ is an unknown scalar (remember: the DLT solution is a null-space).

Writing $H = [\mathbf{h}_1, \mathbf{h}_2, \mathbf{h}_3]$ we thus obtain:

$$\mathbf{r}_1 = \lambda K^{-1}\mathbf{h}_1 \quad \text{e} \quad \mathbf{r}_2 = \lambda K^{-1}\mathbf{h}_2. \tag{55}$$

Orthogonality $\mathbf{r}_1^{\mathsf{T}}\mathbf{r}_2 = 0$ yields $\mathbf{h}_1^{\mathsf{T}}(KK^{\mathsf{T}})^{-1}\mathbf{h}_2 = 0$ or, equivalently

$$\mathbf{h}_1^{\mathsf{T}} B\, \mathbf{h}_2 = 0 \tag{56}$$

where $B = (KK^{\mathsf{T}})^{-1}$.

Similarly, the condition on the norm: $\mathbf{r}_1^{\mathsf{T}}\mathbf{r}_1 = \mathbf{r}_2^{\mathsf{T}}\mathbf{r}_2$ translates to

$$\mathbf{h}_1^{\mathsf{T}} B\, \mathbf{h}_1 = \mathbf{h}_2^{\mathsf{T}} B\, \mathbf{h}_2. \tag{57}$$

By introducing the "vec-trick" as usual, the last two equations rewrites:

$$(\mathbf{h}_2^{\mathsf{T}} \otimes \mathbf{h}_1^{\mathsf{T}})\, \mathrm{vec}(B) = 0 \tag{58}$$

$$\big((\mathbf{h}_1^{\mathsf{T}} \otimes \mathbf{h}_1^{\mathsf{T}}) - (\mathbf{h}_2^{\mathsf{T}} \otimes \mathbf{h}_2^{\mathsf{T}})\big)\, \mathrm{vec}(B) = 0. \tag{59}$$

Since B is a symmetric 3×3 matrix, its unique elements are only six. This can formally be taken in consideration by introducing vech and the duplication matrix D_3:

$$(\mathbf{h}_2^\mathsf{T} \otimes \mathbf{h}_1^\mathsf{T}) D_3 \,\mathrm{vech}(B) = 0 \tag{60}$$

$$\left((\mathbf{h}_1^\mathsf{T} \otimes \mathbf{h}_1^\mathsf{T}) - (\mathbf{h}_2^\mathsf{T} \otimes \mathbf{h}_2^\mathsf{T})\right) D_3 \,\mathrm{vech}(B) = 0. \tag{61}$$

In summary, one image provides two equations in six unknowns.

If n images of the plane are observed (differently oriented), we can stack the $2n$ equations resulting in a linear system

$$A \,\mathrm{vech}(B) = \mathbf{0} \tag{62}$$

where A is a $2n \times 6$ matrix. If $n \geq 3$ we have a solution, determined up to a scale factor.

From $\mathrm{vech}(B)$ we obtain B and therefore K (by Cholesky factorization), from which we go back to R and $\mathbf{t}$.

Listing 4. SMZ Calibration

```
function [P,K] = calibSMZ( H )
%CALIBSMZ Camera calibration from object-image homographies
% Sturm-Maybank-Zhang

numV = length(H);
S = duplication(3);

L = [];                 % build linear system
for i =1:numV
    L=[L;(kron(H{i}(:,2)',H{i}(:,1')))*S;
        (kron(H{i}(:,1)',H{i}(:,1'))-kron(H{i}(:,2)',H{i}(:,2')))*S];
end

[~,~,V] = svd(L);  % solve
B = reshape(S*V(:,end),3,[]);

if trace(B)<0
    B=-B; % either B or -B is p.d.
end
iK = diag([-1,-1,1])*chol(B); % force negative focal

K = inv(iK); K = K./K(3,3);
```

Listing 5. SMZ Calibration

```
% compute extrinsic parameters
for i = 1:numV
    A = iK * H{i};
    A = A/norm(A(:,1)); % remove scale

    R = [A(:,1:2) , cross(A(:,1), A(:,2))];
    t = A(:,3);

    %  project onto SO(3)
    [U,~,V] = svd(R);
    R = U*diag([1,1,det(V'*U)])*V';

    if   [0,0,1] * (-R'*t)  < 0
        % change sign if cop is in the negative halfspace
        R(:,1:2) =  - R(:,1:2);
        t = - t;
    end

    P{i} =  K * [R,t];
end
end
```

5 Absolute and Exterior orientation

We define several orientation problems are defined, that require the computation of a 3D direct isometry (rotation + translation) from correspondences between points.

- Absolute orientation (3D-3D). The coordinates of some 3D points points with respect to two different 3D reference frames are given; we are required to determine the transformation between the two reference frames.
- Exterior Orientation (3D-2D). The position of some 3D points and their projection in the image are given; we are required to determine the transformation between the camera reference frame and the world reference frame. The intrinsic parameters are assumed to be known.
- Relative orientation (2D-2D). The projections of some 3D points two distinct images are given; we are required to determine the transformation between the two camera reference frames. The intrinsic parameters are assumed to be known.

Absolute orientation

Suppose we have two sets of n 3D points, which correspond to a single model, but which are expressed in two different systems of reference, that is, they differ for a direct isometry.

We will call one of these sets $\mathcal{D}$ and the other $\mathcal{M}$. We assume that for each point of $\mathcal{M}$ the corresponding point in $\mathcal{D}$ is known.

The absolute orientation (or registration) problem consists in finding the direct isometry (rotation and translation) that once applied to $\mathcal{M}$ leads it to coincide with $\mathcal{D}$, or, in the presence of noise, which minimizes the distance between the two sets of points.

Let us consider the more general problem of absolute orientation with scale, in which the relationship between the two sets is a similarity:

$$\mathbf{D}_i = s(R\mathbf{M}_i + \mathbf{t}) \quad \text{for all } i = 1 \ldots n \geq 3 \tag{63}$$

where R is a 3×3 rotation matrix, $\mathbf{t}$ is a 3×1 vector of translation, s is a scalar and $\mathbf{D}_i, \mathbf{M}_i$ are corresponding points (in Cartesian coordinates [1]) in the sets $\mathcal{D}$ and $\mathcal{M}$.

The goal of registration is to solve:

$$\min_{s,R,\mathbf{t}} \sum_{i=1}^{n} \|\mathbf{D}_i - s(R\mathbf{M}_i + \mathbf{t})\|^2. \tag{64}$$

This method minimizes the sum of the squares of the distances between corresponding points.

[1] Warning: in this paragraph, the Cartesian coordinates of the points are denoted without $\tilde{}$.

Interlude:Orthogonal Procrustian problems

Proposition 5.1 (Orthogonal Procrustian problem) *Given two matrices A and B, the solution of the problem*

$$\min_{Q^\mathsf{T}Q=I} \|A - QB\|_F^2 \tag{65}$$

is $Q = UV^\mathsf{T}$ where UDV^T is theSVD of AB^T: $AB^\mathsf{T} = UDV^\mathsf{T}$.

Proof. $\|A - QB\|_F^2 = \mathrm{tr}(A^\mathsf{T}A - 2AB^\mathsf{T}Q^\mathsf{T} + BB^\mathsf{T})$ therefore we want to maximize $\mathrm{tr}(AB^\mathsf{T}Q^\mathsf{T})$. Let $AB^\mathsf{T} = UDV^\mathsf{T}$ be, therefore

$$\mathrm{tr}(AB^\mathsf{T}Q^\mathsf{T}) = \mathrm{tr}(UDV^\mathsf{T}Q^\mathsf{T}) = \mathrm{tr}(DV^\mathsf{T}Q^\mathsf{T}U) = \mathrm{tr}(DZ) = \sum \sigma_i Z_{ii} \tag{66}$$

Being Z orthogonal we have $|Z_{ii}| \leq 1$ and therefore the maximum is reached with $Z = V^\mathsf{T}Q^\mathsf{T}U = I$, hence the thesis.

If we want Q to be a rotation, i.e., $\det(Q) = 1$, the solution changes as follows:

$$Q = U \begin{bmatrix} 1 & 0 & 0 \\ 0 & 1 & 0 \\ 0 & 0 & \det(V^{\mathsf{T}}U) \end{bmatrix} V^{\mathsf{T}}. \tag{67}$$

It can be seen in fact that $\det(Z) = \det(V^{\mathsf{T}}U)\det(Q) = \det(V^{\mathsf{T}}U)$. Hence, if $\det(V^{\mathsf{T}}U) = 1$ is taken as before, while $Z = 1$ if $\det(V^{\mathsf{T}}U) = -1$ we are forced to take a Z with determinant -1. Since the singular values σ_i are ordered, the cheapest choice is to take $Z = \mathrm{diag}(1, 1, 1, \ldots, -1)$.

A special case of this problem occurs when $B = I$. Then the probem is those of finding the orthogonal matrix closest to a given matrix A, that is:

$$\min_{Q^{\mathsf{T}}Q=I} \|A - Q\|_F^2. \tag{68}$$

The solution is equivalent to replacing the matrix D with the identity in the SVD of A. The procedure is similar in the case that A is a rotation matrix.

Orthogonal Procrustean analysis

We will see a method based on the Orthogonal Procustian Analysis (OPA) (Arun et al., 1987), or, ultimately, on the SVD.

Let D be the $3 \times n$ matrix obtained by juxtaposing side by side the vectors (column) $\mathbf{D}_i$ and M the matrix obtained in the same way with vectors $\mathbf{M}_i$. It can be easily verified that the objective function rewrites:

$$\chi = \sum_{i=1}^{n} \|\mathbf{D}_i - sR(\mathbf{M}_i + \mathbf{t})\|^2 = \|D - s(RM + \mathbf{t}\mathbf{1}^\mathsf{T})\|_F^2. \tag{69}$$

Note that if $\mathbf{t}$ is a vector of d elements and $\mathbf{1}$ is a vector of n ones, then $\mathbf{t}\mathbf{1}^\mathsf{T}$ replicates the vector n times, yielding a $d \times n$ matrix.

If there were no translation, this would be an orthogonal procrustean problem, which we could easily solve with OPA (scale is not a problem).

We rewrite (63) in matrix form:

$$D = s(RM + \mathbf{t}\mathbf{1}^{\mathsf{T}}). \tag{70}$$

Observe that if A is a $n \times n$ matrix, $A\mathbf{1}/n$ is the vector containing the average over the rows of A. Therefore, taking the average on the rows of both members we obtain:

$$D\mathbf{1}/n = s(RM + \mathbf{t}\mathbf{1}^{\mathsf{T}})\mathbf{1}/n = sRM\mathbf{1}/n + s\mathbf{t}\mathbf{1}^{\mathsf{T}}\mathbf{1}/n \tag{71}$$

which immediately gives rise to an expression for translation:

$$\mathbf{t} = (\frac{1}{s}D - RM)\mathbf{1}/n. \tag{72}$$

To get rid of the translation, we substitute this expression in (70):

$$D = s(RM + (\frac{1}{s}D - RM)\mathbf{1}\mathbf{1}^{\mathsf{T}}/n) = s(RM + \frac{1}{s}D\mathbf{1}\mathbf{1}^{\mathsf{T}}/n - RM\mathbf{1}\mathbf{1}^{\mathsf{T}}/n) \tag{73}$$

thus obtaining

$$D\underbrace{(I - \mathbf{1}\mathbf{1}^{\mathsf{T}}/n)}_{J_n} = sRM\underbrace{(I - \mathbf{1}\mathbf{1}^{\mathsf{T}}/n)}_{J_n}. \tag{74}$$

This problem is equivalent to the original one, but the translation has been eliminated. It will be computed at the end after recovering R and s.

Note that the matrix $J_n = I - \mathbf{1}\mathbf{1}^\mathsf{T}/n$, which is symmetric and idempotent, has the effect of subtracting the mean of the columns from each column of the matrix to which it is applied:

$$D - (D\mathbf{1}/n)\mathbf{1}^\mathsf{T} = D(I - \mathbf{1}\mathbf{1}^\mathsf{T}/n). \tag{75}$$

Therefore, if the columns are points, as in our case, it subtracts the centroid of the set from the coordinates of each one.

If it were not for the scale, this could be solved by applying the OPA between the two matrices DJ_n and MJ_n.

Note that in OPA the scale is irrelevant in determining the rotation, for it is absorbed by the diagonal matrix of the SVD, the which does not contribute to determining R.

Then the rotation is given from the SVD of $DJ_nJ_n^\mathsf{T}M^\mathsf{T} = DJ_nM^\mathsf{T}$, being J_n symmetrical and idempotent:

$$R = U \begin{bmatrix} 1 & 0 & 0 \\ 0 & 1 & 0 \\ 0 & 0 & \det(V^\mathsf{T}U) \end{bmatrix} V^\mathsf{T} \tag{76}$$

where $USV^\mathsf{T} = DJ_nM^\mathsf{T}$.

The scale is calculated as the solution of a one-dimensional least squares problem. Given two matrices A and B the problem $A = sB$ is rewritten as: $\mathrm{vec}(A) = s\,\mathrm{vec}(B)$, whose least squares solution is:

$$s = \frac{\mathrm{vec}(B)^{\mathsf{T}}\,\mathrm{vec}(A)}{\mathrm{vec}(B)^{\mathsf{T}}\,\mathrm{vec}(B)} = \frac{\mathrm{tr}(B^{\mathsf{T}}A)}{\mathrm{tr}(B^{\mathsf{T}}B)} \tag{77}$$

due to properties of the trace. In our case we get:

$$s = \frac{\mathrm{tr}(DJ_nM^{\mathsf{T}}R^{\mathsf{T}})}{\mathrm{tr}(MJ_nM^{\mathsf{T}})}. \tag{78}$$

Listing 6. OPA

```
function [R,t,s] = opa(D,M,w)
    %OPA Orthogonal Procrustes analysis
    % Solves the othogonal procrustes problem D=s*(R*M+t)

    if nargin==3  % Weights  provided
        w=w(:); % make sure it is a column
        % same as  D = D*W;  M = M*W;  but faster
        D = bsxfun(@times, w', D);
        M = bsxfun(@times, w', M);

    else  % otherwise no weights
        w = ones(size(M,2),1);
    end

    J = eye(size(M,2)) - (w*w')/(w'*w);  % centering matrix

    [U,S,V] = svd(D*J*M'); % solve for rotation
    R = U*diag([1,1,det(V'*U)])*V';

    if nargout==2  % no scale
        s = 1;
    else   % solve for scale
        s = trace(S) / trace(M*J*M');
        % trace(D*J*M'*R') == trace(S)
    end

    t =(D/s -R*M)*w/(w'*w); % solve for translation
end
```

Go to live coding.

Exterior orientation

Suppose we know the intrinsic parameters, the coordinates of some 3D control points and their projections in the image.

The problem of exterior orientation (or also perspective pose) consists in determining the transformation between the camera reference system and the world reference system in which 3D points are expressed, that is the exterior orientation (position and angular structure) of the camera wrt to the world reference.

Basically this is a problem similar to the calibration of the camera, in which, however, only the extrinsic parameters $\omega, \varphi, \kappa, t_1, t_2, t_3$ are required.

As with calibration, the techniques based on control points and a single image take the name of resection, term that generically indicates the estimate of the orientation (exterior or exterior + interior) of the image from 3D control points and their images.

So let $\mathbf{M}_1 \dots \mathbf{M}_n$ be n control points, expressed in the 3D reference system of the object itself and are $\mathbf{m}_1 \dots \mathbf{m}_n$ the respective points projected on the imagee. The relationship between a control point and its own projection on the image is given by the perspective projection:

$$\mathbf{q}_i \simeq [R|\mathbf{t}]\mathbf{M}_i \tag{79}$$

where $\mathbf{q}_i = [x_i, y_i, 1]^\mathsf{T} = K^{-1}\mathbf{m}_i$ are the normalized coordinates.

If one is given at least six points, one can proceed with calibration using the DLT method, thereby obtaining a PPM identifiable with $[R|\mathbf{t}]$.

This approach is not satisfactory because the matrix R thus obtained is not necessarily a rotation (due to the noise present in the data), and therefore this property must be forced a posteriori.

Fiore's linear method

We introduce a method proposed by Fiore (2001) which cast the problem to that of absolute orientation, thus producing a rotation matrix by construction.

We rewrite (79) by making the scale factors explicit ζ_i:

$$\zeta_i \mathbf{q}_i = [R|\mathbf{t}]\mathbf{M}_i = R\widetilde{\mathbf{M}}_i + \mathbf{t} \quad i = 1 \dots n. \tag{80}$$

It is clearly seen that if the factors ζ_i were known, we would be faced with a problem of absolute orientation. Let us therefore concentrate on the computation of ζ_i. To this end, we rewrite (80) in matrix form:

$$\underbrace{[\mathbf{q}_1, \mathbf{q}_2, \dots \mathbf{q}_n]}_{Q} \operatorname{diag}(\boldsymbol{\zeta}) = [R|\mathbf{t}]\underbrace{[\mathbf{M}_1, \mathbf{M}_2, \dots \mathbf{M}_n]}_{M} \tag{81}$$

where $\boldsymbol{\zeta} = [\zeta_1, \zeta_2, \dots \zeta_n]^{\mathsf{T}}$ and diag constructs the diagonal matrix given a vector. Let $r = \mathrm{r}(M)$. Calculate the SVD of M: $M = UDV^{\mathsf{T}}$ and let V_r be the matrix composed of the last $n - r$ columns of V, which generate the kernel of M.

Multiplying both sides of (81) by V_r has the effect of canceling the right member:

$$Q \operatorname{diag}(\boldsymbol{\zeta}) V_r = M V_r = 0_{4\times(n-r)}. \tag{82}$$

Taking the vec of both members and exploiting the "vec trick" yields:

$$\operatorname{vec}(Q \operatorname{diag}(\boldsymbol{\zeta}) V_r) = \mathbf{0} \iff \left(V_r^{\mathsf{T}} \otimes Q\right) \operatorname{vec}(\operatorname{diag}(\boldsymbol{\zeta})) = \mathbf{0}. \tag{83}$$

The vector $\operatorname{vec}(\operatorname{diag}(\boldsymbol{\zeta}))$ does not only contain the unknowns $\boldsymbol{\zeta}$ but also zeros from the off-diagonal elements of $\operatorname{diag}(\boldsymbol{\zeta})$. To fix this, similarly to what is done with the vectorization of symmetric matrices, we introduce an appropriate matrix D such that $\operatorname{vec}(\operatorname{diag}(\boldsymbol{\zeta})) = D\boldsymbol{\zeta}$ and then substitute in the previous equation, obtaining:

$$\left(V_r^{\mathsf{T}} \otimes Q\right) D\boldsymbol{\zeta} = \mathbf{0}. \tag{84}$$

By stacking enough equations like this we can get $\boldsymbol{\zeta}$ - up to a multiplicative constant - as the kernel of the coefficients matrix.

How many points are needed? The size of the coefficient matrix in (82) is $3(n-r) \times n$ and to determine a one-dimensional family of solutions it must have rank $n-1$, so it must be: $3(n-r) \geq n-1$. It follows that $n \geq (3r-1)/2$ points are needed. For example, six points in general position are needed ($r = 4$).

Now that the right side of (80) is known up to a scale factor, all that remains is to solve the absolute orientation with scale problem given by (80) with the OPA method.

Fiore's method returns a rotation matrix, but needs more points than necessary.

Indeed, expanding (79) we can see that each point match generates two equations:

$$\begin{cases} x_i = \dfrac{\mathbf{r}_1^{\mathsf{T}}\widetilde{\mathbf{M}}_i + t_1}{\mathbf{r}_3^{\mathsf{T}}\widetilde{\mathbf{M}}_i + t_3} \\ y_i = \dfrac{\mathbf{r}_2^{\mathsf{T}}\widetilde{\mathbf{M}}_i + t_2}{\mathbf{r}_3^{\mathsf{T}}\widetilde{\mathbf{M}}_i + t_3} \end{cases} \tag{85}$$

where $R = [\mathbf{r}_1, \mathbf{r}_2, \mathbf{r}_3]^{\mathsf{T}}$ and $\mathbf{t} = [t_1, t_2, t_3]^{\mathsf{T}}$. If the rotation matrix R is expressed as a function of three parameters (for example with the three Euler angles $\omega, \varphi, \ k$), in the end we will have a non linear system in six unknowns, which can be determined if at least three correspondence of points are known.

Go to live coding.

Listing 7. Exterior orientation

```
function [R,t] = exterior_lin(m,M,K)
%EXTERIOR_LIN Exterior orientation with Fiore's algorithm

    Q = K\ensure_homogeneous(m);   % homogeneous NIC
    M = [M; ones(1,size(M,2))  ];  % homogeneous

    [~, D, V] = svd (M);
    r = sum(diag(D)>max(size(M))*eps(max(D(:)))); % = rank(M)
    Vr = V(:,r+1:end);


    % solve null space
    [~, ~, V] = svd(kron(Vr',ones(3,1)) .* kron(ones(size(Vr,2),1), Q));
    z = V(:,end);

    %  fix sign of scale
    z = z * sign(z(1));

    [R,t,~] = opa(Q * diag(z), M(1:3,:)); % with scale

end
```

6 Two-view geometry

Epipolar geometry is the relationship that links two images that frame the same scene from two different points of view. In particular, it concerns the relationship that exists between conjugate (or homologous) points, which are defined as the points in the two images that are projection of the same point in space.

Given a point m in the first image, its conjugate point m' must lie on a straight line in the second image, called epipolar line of m.

Point m' can be the projection of any point in the space lying on the optical ray of m.

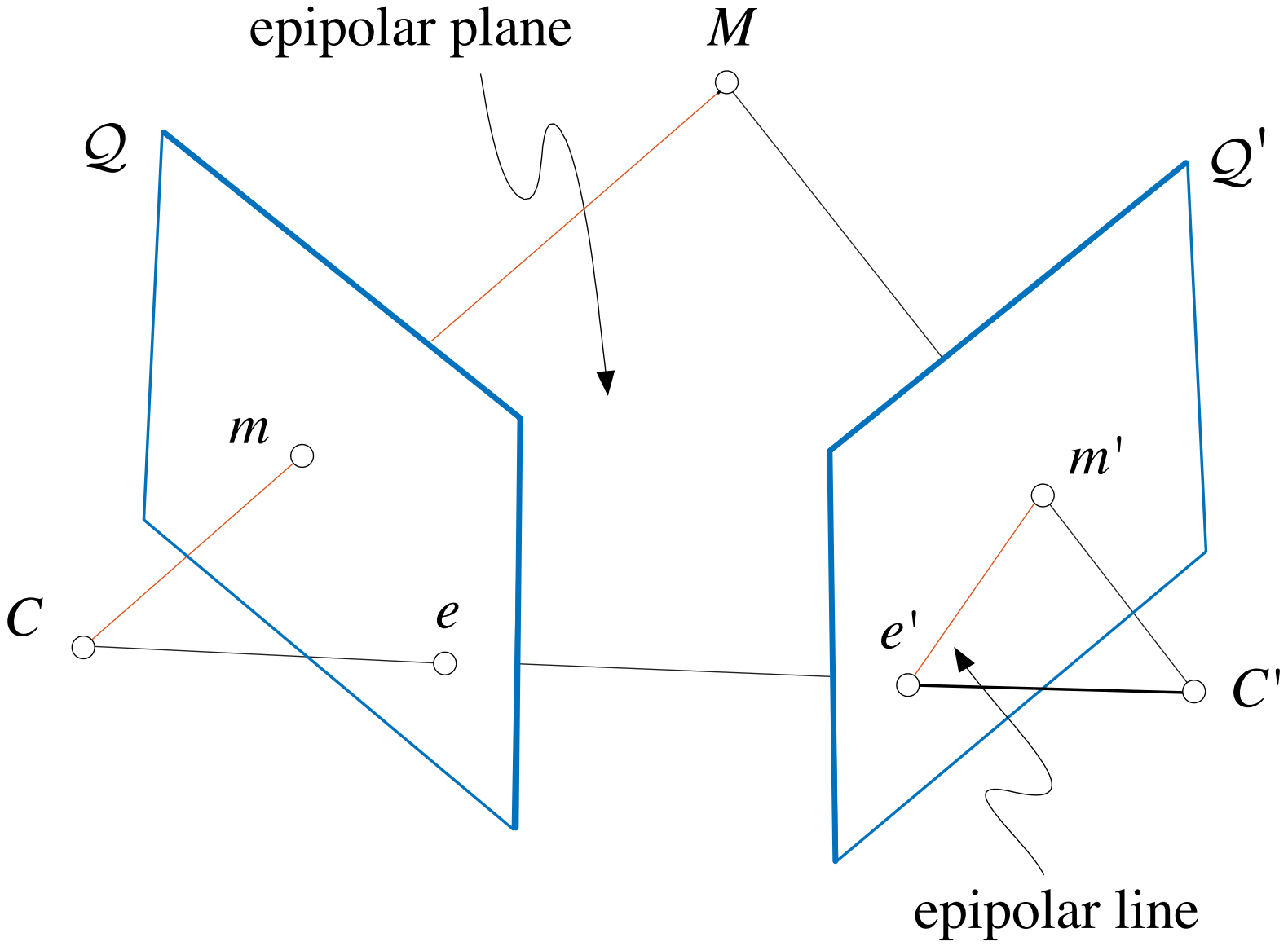
Therefore m' is constrained to lie on the intersection of the image plane with the plane determined by m, C and C', called epipolar plane.

All epipolar lines of an image pass through the same point, called epipole.

The epipolar planes constitute a bundle of planes that have in common the straight line passing through the COPs C and C'.

The line CC' is called *baseline*.

https://www.ipb.uni-bonn.de/html/staff/WolfgangFoerstner/collectednotes_v2/Cinderella/epipolar-geometrie-with-text.html


epipolar plane
M
Q
Q'
m
m'
C
e
e'
C'
epipolar line

Given two PPM $P = [\widetilde{P}|\mathbf{p}_4]$ and $P' = [\widetilde{P}'|\mathbf{p}'_4]$, we know that:

$$\begin{cases} \mathbf{m} \simeq P\mathbf{M} \\ \mathbf{m}' \simeq P'\mathbf{M}. \end{cases} \tag{86}$$

The epipolar line corresponding to m is the projection according to P' of the optical ray of m, which has the equation:

$$\mathbf{M} = \mathbf{C} + \lambda \begin{bmatrix} \widetilde{P}^{-1}\mathbf{m} \\ 0 \end{bmatrix}. \tag{87}$$

Since:

$$P'\mathbf{C} = P' \begin{bmatrix} -\widetilde{P}^{-1}\mathbf{p}_4 \\ 1 \end{bmatrix} = \mathbf{p}'_4 - \widetilde{P}'\widetilde{P}^{-1}\mathbf{p}_4 = \mathbf{e}' \tag{88}$$

and

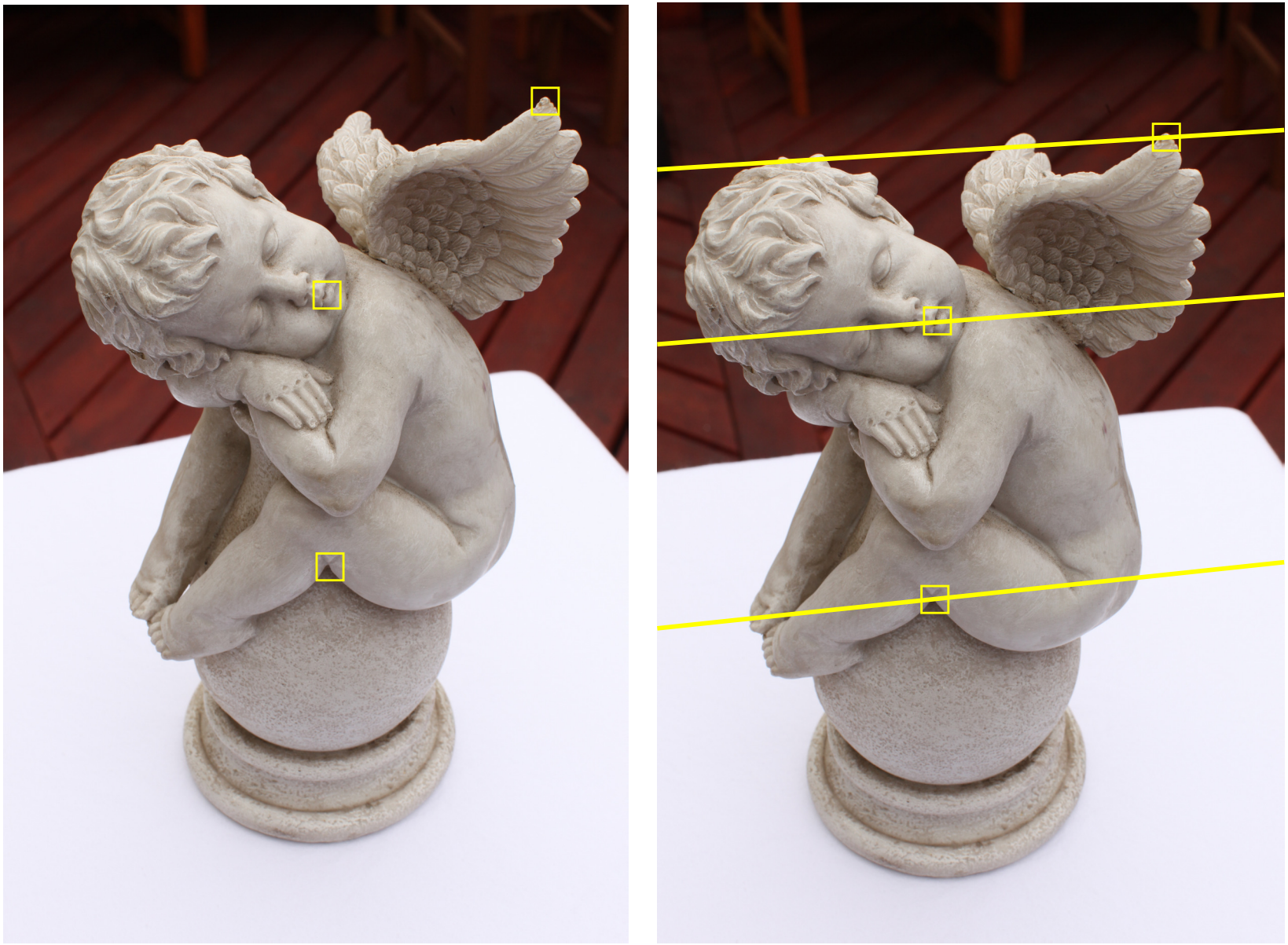
$$P' \begin{bmatrix} \widetilde{P}^{-1}\mathbf{m} \\ 0 \end{bmatrix} = \widetilde{P}'\widetilde{P}^{-1}\mathbf{m}, \tag{89}$$

the epipolar line of m has equation:

$$\mathbf{m}' \simeq \lambda\widetilde{P}'\widetilde{P}^{-1}\mathbf{m} + \mathbf{e}'. \tag{90}$$

This is the equation (in homogeneous coordinates) of the straight line for points $\mathbf{e}'$ (epipole) and the projection of the point at infinity $\widetilde{P}'\widetilde{P}^{-1}\mathbf{m}$.

Example of epipolar lines tracing in a pair of images.



Normal case

When the COP is in the focal plane of the conjugate camera, the epipole $\mathbf{e}'$ lies at infinity and the epipolar lines form a bundle of parallel lines.

A very special case is when both epipoles are to infinity, which happens when the baseline CC' is contained in both focal planes, i.e. the two image planes are parallel to the baseline (we will call this configuration normal).

In this case the epipolar lines form a bundle of parallel lines in both Images.

Fundamental matrix

To see that there is a bilinear relatyionship between the conjugate points we have to elaborate further equation (90).

We multiply left and right for $[\mathbf{e}']_\times$, (remember that $[\mathbf{a}]_\times \mathbf{a} = \mathbf{0}$):

$$[\mathbf{e}']_\times \mathbf{m}' \simeq \lambda [\mathbf{e}']_\times \widetilde{P}' \widetilde{P}^{-1} \mathbf{m}. \tag{91}$$

The left side is a vector orthogonal to $\mathbf{m}'$, so if we multiply left and right by $\mathbf{m}'^{\mathsf{T}}$, we obtain:

$$0 = \mathbf{m}'^{\mathsf{T}} [\mathbf{e}']_\times \widetilde{P}' \widetilde{P}^{-1} \mathbf{m}. \tag{92}$$

This equation, named after Longuet-Higgins, represents a bilinear form in $\mathbf{m}$ and $\mathbf{m}'$.

The matrix

$$F = [\mathbf{e}']_\times \widetilde{P}' \widetilde{P}^{-1} \tag{93}$$

which contains the coefficients of the bilinear form is called fondamental matrix.

The Longuet-Higgins equation therefore rewrites:

$$\mathbf{m}'^{\mathsf{T}} F \mathbf{m} = 0. \tag{94}$$

It states that the corresponding point of $\mathbf{m}$ in the second image must lie on its epipolar line $F\mathbf{m}$; in fact the line defined by $F\mathbf{m}$ is the epipolar line of m.

Given that $\det([\mathbf{e}']_\times) = 0$, we have that $\det(F) = 0$. In in particular we see that the rank of F is equal to 2.

Furthermore, the matrix F is defined up to a scale factor, since if it is multiplied by an arbitrary scalar, the equation (94) is however satisfied.

Hence a fundamental matrix has 7 degrees of freedom.

Observe that, by transposing (94) we obtain the symmetrical relationship:

$$\mathbf{m}^{\mathsf{T}} F^{\mathsf{T}} \mathbf{m}' = 0. \tag{95}$$

The epipole $\mathbf{e}'$ is easily obtained from F as the kernel of F^{T}.

Indeed, since the epipole belongs to all epipolar lines, $\mathbf{e}'^{\mathsf{T}} F\mathbf{m} = 0$ must hold for every $\mathbf{m}$, and therefore it must be that $\mathbf{e}'^{\mathsf{T}} F = \mathbf{0} = F^{\mathsf{T}}\mathbf{e}'$.

One could also have derived the latter using the fact that $[\mathbf{e}']_{\times}\mathbf{e}' = \mathbf{0}$.

Similarly $\mathbf{e}^{\mathsf{T}} F^{\mathsf{T}} = \mathbf{0}$.

Computing the fundamental matrix

Given a set of point correspondences (at least eight, as we shall see) $\{(\mathbf{m}_i, \mathbf{m}'_i) \mid i = 1, \ldots, n\}$, one wants to determine the fundamental matrix F that connects the points in the bilinear relation:

$$\mathbf{m}_i'^{\mathsf{T}} F \mathbf{m}_i = 0. \tag{96}$$

The unknown matrix can be easily obtained thanks to the use of the "vec-trick". In fact we derive:

$$\mathbf{m}_i'^{\mathsf{T}} F \mathbf{m}_i = 0 \iff \mathrm{vec}(\mathbf{m}_i'^{\mathsf{T}} F \mathbf{m}_i) = 0 \iff (\mathbf{m}_i^{\mathsf{T}} \otimes \mathbf{m}_i'^{\mathsf{T}})\,\mathrm{vec}(F) = 0.$$

So each correspondence of points generates a linear homogeneous equation in the nine unknown elements of the matrix F (read by columns).

From n corresponding points we obtain a linear system of n equations:

$$\underbrace{\begin{bmatrix} \mathbf{m}_1^{\mathsf{T}} \otimes \mathbf{m}_1'^{\mathsf{T}} \\ \mathbf{m}_2^{\mathsf{T}} \otimes \mathbf{m}_2'^{\mathsf{T}} \\ \vdots \\ \mathbf{m}_n^{\mathsf{T}} \otimes \mathbf{m}_n'^{\mathsf{T}} \end{bmatrix}}_{A_n} \mathrm{vec}(F) = 0. \tag{97}$$

The solution to the homogeneous linear system is the kernel of A_n. With $n = 8$ the kernel of the matrix has dimension one, so the solution is determined up to a multiplicative constant. Therefore this method is called eight-points algorithm.

In practice, more than eight point correspondences are available and we can determine F by solving a linear least squares problem. The solution is the least right singular vector in the SVD of A_n.

The matrix F found as the solution of the system (97) will not, in general, have rank two.

This can be forced a posteriori by following the Eckart-Young theorem.

Listing 8. Eight-points algorithm

```
function E = eight_points(m2, m1, w)
%EIGHT_POINTS 8-points algorithm for F/E

    if nargin <3, w = ones(size(m1,2),1); end % weights

    m1 = ensure_homogeneous(m1);
    m2 = ensure_homogeneous(m2);

    L = [];
    for i = 1: size(m1,2)
        L = [L; w(i) * kron( m1(:,i)' , m2(:,i)') ];
    end

    [~,~,V] = svd(L);
    E = reshape(V(:,end),3,3);

end
```

Go to live coding.

Seven-point algorithm

Alternatively, it is possible to easily derive a " seven-points algorithm" which produces a F of rank two by construction. Indeed since F has only 7 degrees of freedom, 7 matches must be sufficient to compute it, with the addition of the rank constraint.

The coefficient matrix A_7 which is generated by 7 correspondences has dimension 7×9 and a kernel of dimension 2, so the generic solution is written as a linear combination of two particular solutions (once fixed the scale, only one coefficient is needed):

$$F = F_8 + \mu F_9 \tag{98}$$

where F_8 and F_9 are the two solutions corresponding to the penultimate and last right singular vector in the SVD of A: $A = UDV^{\mathsf{T}}$.

Now one have to determine the specific μ which makes F singular, solving:

$$\det(F_8 + \mu F_9) = 0 \tag{99}$$

which is a third degree polynomial in μ and therefore can be solved analytically, obtaining 1 or 3 real solutions.

Preconditioning

The eight-point algorithm has been criticized for being too sensitive to noise, and therefore not very useful in practical applications.

Hartley (1995) proved that the instability of the method is mainly due to a problem of ill-conditioning.

In a typical image the coordinates are $[\approx 10^3, \approx 10^3, 1]$, so the matrix A_n has entries from 1 to $\approx 10^6$, and consequently a high conditioning number, which makes the system solution unstable.

By applying a simple pre-conditioning technique – that consists of transforming the points so that the three coordinates have the same orders of magnitude – the conditioning number is reduced and the results are more stable.

The points are translated so that their centroid coincides with the origin and then they are scaled so that the average distance from the origin is equal to $\sqrt{2}$.

The latter choice derives from the observation that the best conditioning occurs when the average coordinates are $[\approx 1, \approx 1, 1]$ and this point has a distance $\sqrt{2}$ from the origin.

Let T and T' be the resulting transformations in the two images and $\bar{\mathbf{m}} = T\mathbf{m}$, $\bar{\mathbf{m}}' = T'\mathbf{m}'$ points transformed. Using $\bar{\mathbf{m}}$ and $\bar{\mathbf{m}}'$ in the eight-points algorithm, we obtain a fundamental matrix $\bar{F}$ that can be related to the original one by means of $F = T'^{\mathsf{T}}\bar{F}T$, as can be easily demonstrated.

Listing 9. Preconditioning

```
function [T,m] = precond(m)
%PRECOND Normalize the coordinates of kD points

    avg = mean(m,2) ;
    m = m - avg;
    scale = mean(sqrt(sum(m.^2,1)))/sqrt(size(m,1));
    m = m./scale;

    T = [eye(size(m,1)), -avg]./scale;
    T(end+1,end)=1;

end
```

Listing 10. Linear computation of F

```
function F = fund_lin(m2,m1,w)
%FUND_LIN  Fundamental matrix with 8-points algorithm

if nargin < 3 || isempty(w)
    w = ones(size(m1,2),1);  % weights
end

% pre-conditioning
[T1,m1] = precond(m1);
[T2,m2] = precond(m2);

F = eight_points(m2, m1, w);

% enforce singularity of  F
[U,D,V] = svd(F);
D(3,3) = 0; F = U*D*V';

% apply the inverse scaling
F = T2' * F * T1;

end
```

Planar Homography

Let us return to epipolar geometry.

While in the more general case the conjugate points in two images are linked by a fondamental matrix, when the observed points lie on a plane in space, the correspondences of points between two images are encoded by a projectivity of $\mathbb{P}^2$, or homography or collineation.

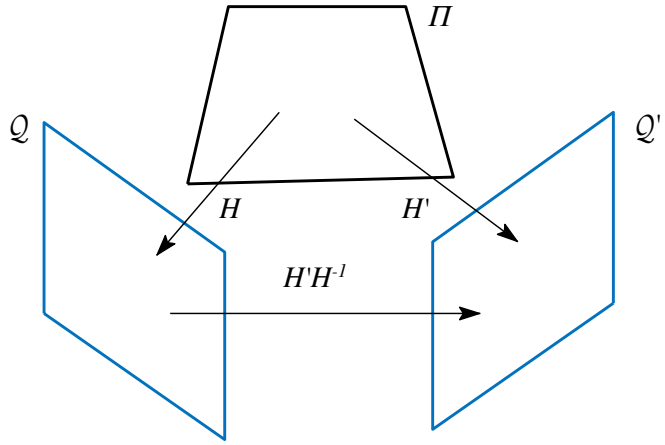
Homography induced by a plane

We already established that the application between a plane Π in space and its perspective image is a homography.

Then we observe that the projections of the points of Π in two images they are linked by a homography.

We will therefore say that Π induce a homography between the two images, represented by the matrix H_Π, in the sense that corresponding points in the two images are mapped via H_Π. In formulas:

$$\mathbf{m}' \simeq H_\Pi \mathbf{m} \quad \text{if} \quad \mathbf{m}' \simeq P'\mathbf{M}, \ \mathbf{m} \simeq P\mathbf{M}, \ \text{and} \ \ \mathbf{M} \in \Pi. \tag{100}$$



The matrix H_Π is 3×3 non-singular, and being defined up to a scale factor it has 8 degrees of freedom. Three of these are due to the choice of the plane Π, the other five derive from the compatibility with the epipolar geometry, represented by F. Indeed all points related through H_Π must also satisfy the epipolar constraint, that is:

$$\begin{cases} \mathbf{m}'^{\mathsf{T}} F \mathbf{m} = 0 \\ \mathbf{m}' = H_\Pi \mathbf{m} \end{cases} \tag{101}$$

which implies:

$$(H_\Pi \mathbf{m})^{\mathsf{T}} F \mathbf{m} = 0. \tag{102}$$

This is equivalent to saying that $H_\Pi^{\mathsf{T}} F$ must be antisymmetric, that is: $H_\Pi^{\mathsf{T}} F + F^{\mathsf{T}} H_\Pi = 0_{3\times3}$. For the homogeneity of the equation and the symmetry of the matrix to the left of the equal sign, this translates into 5 constraints.

To find an expression of H_Π we have to start from epipolar geometry and specialize it. If we take the system of reference of the first camera as the world reference system, we can write the following two PPMs:

$$P = K[I|\mathbf{0}] = [K|\mathbf{0}] \quad \text{and} \quad P' = K'[R|\mathbf{t}]. \tag{103}$$

Substituting these PPMs in the equation of the epipolar line of $\mathbf{m}$ (90) we obtain

$$\mathbf{m}' \simeq \lambda K' R K^{-1} \mathbf{m} + K'\mathbf{t} \tag{104}$$

with

$$\mathbf{e}' = K'[R|\mathbf{t}] \begin{bmatrix} \mathbf{0} \\ 1 \end{bmatrix} = K'\mathbf{t}. \tag{105}$$

If the 3D point $\mathbf{M}$ lies on a plane Π of equation $\mathbf{n}^\mathsf{T}\widetilde{\mathbf{M}} = d$ then equation (104) can be specialized. After a few steps one gets:

$$\mathbf{m}' \simeq \underbrace{K' \left(R + \frac{\mathbf{t}\,\mathbf{n}^\mathsf{T}}{d} \right) K^{-1}}_{H_\Pi} \mathbf{m}. \tag{106}$$

Homography of the plane at infinity If $d \to \infty$ in the limit we obtain $H_\infty = K'RK^{-1}$, the matrix of the homography induced by the plane at infinity.

There is another special case where two images are linked by the homography of the plane at infinity: when the baseline (i.e. the relative translation) is null.

To see it, consider again (104): with $\mathbf{t} = \mathbf{0}$:

$$\mathbf{m}' \simeq K'RK^{-1}\mathbf{m}. \tag{107}$$

H_∞ does not depend from the 3D structure.

H_∞ plays a double role:

- like all homographies induced by a plane it maps projections of the points of that plane between the two images. In particular H_∞ associates the projections of the points that lie on the plane at infinity, for example the vanishing points;
- furthermore H_∞ associates the projections of all the points of the scene (no matter which plane they belong to) between two images when the camera undergoes a purely rotational motion.

Computing of homography with DLT

n corresponding points are given $(\mathbf{m}_i; \mathbf{m}'_i)$, and one is reqiured to determine the homography matrix H such that:

$$\mathbf{m}'_i \simeq H\mathbf{m}_i \quad i = 1 \ldots n. \tag{108}$$

Leveraging the cross product to eliminate the multiplicative factor, the equation is rewritten as:

$$\mathbf{m}'_i \times H\mathbf{m}_i = \mathbf{0}. \tag{109}$$

Along the lines of the derivation made for the DLT one gets:

$$\mathbf{m}'_i \times H\mathbf{m}_i = \mathbf{0} \iff [\mathbf{m}'_i]_\times H\mathbf{m}_i = \mathbf{0} \iff$$
$$\mathrm{vec}([\mathbf{m}'_i]_\times H\mathbf{m}_i) = \mathbf{0} \iff (\mathbf{m}_i^\mathsf{T} \otimes [\mathbf{m}'_i]_\times)\,\mathrm{vec}(H) = \mathbf{0}$$

and similarly we conclude that the matrix $(\mathbf{m}_i^\mathsf{T} \otimes [\mathbf{m}'_i]_\times)$ has rank two, therefore only two equations out of three are linearly independent.

For n points we obtain a system of $2n$ homogeneous linear equations, that we can write as

$$A\,\mathrm{vec}(H) = \mathbf{0} \tag{110}$$

where A is the $2n \times 9$ matrix of the coefficients obtined by stacking two equations for each match, while the vector of the unknowns $\mathrm{vec}(H)$ contains the nine elements of H read by columns.

Therefore four points in general position determine a matrix of coefficients A of rank eight whose one-dimensional kernel is the solution sought, up to a scale factor.

For $n > 4$ points, the solution is the last right singular vector in the SVD of A.

As in other cases, also in the computation of H it can be useful to transform the data so that the problem becomes better conditioned.

Let $\bar{\mathbf{m}} = T\mathbf{m}$ and $\bar{\mathbf{m}}' = T'\mathbf{m}'$.

Using the correspondences between the transformed points $\bar{\mathbf{m}}$ and $\bar{\mathbf{m}}'$ we find a homography of plane $\bar{H}$ which is related to the sought one by $\bar{H} = T'HT^{-1}$, as can be easily seen.

Listing 11. Linear computation of H

```
function H = hom_lin(m2,m1,w)
%HOM_LIN  Homography with DLT algorithm

if nargin < 3 || isempty(w)
    w = ones(size(m1,2),1); % weights
end

% pre-conditioning
[T1,m1] = precond(m1);
[T2,m2] = precond(m2);

H = dlt(m2, m1, w);

% post-conditioning
H = T2\H * T1;
end
```

Planar parallax

Let's go back to (104):

$$\mathbf{m}' \simeq \lambda H_\infty \mathbf{m} + K'\mathbf{t}. \tag{111}$$

One way to interpret this is that a point m is associated with its own conjugate in two steps: first it is mapped throgh H_∞ and then a correction is added, called parallax, along the epipolar line.

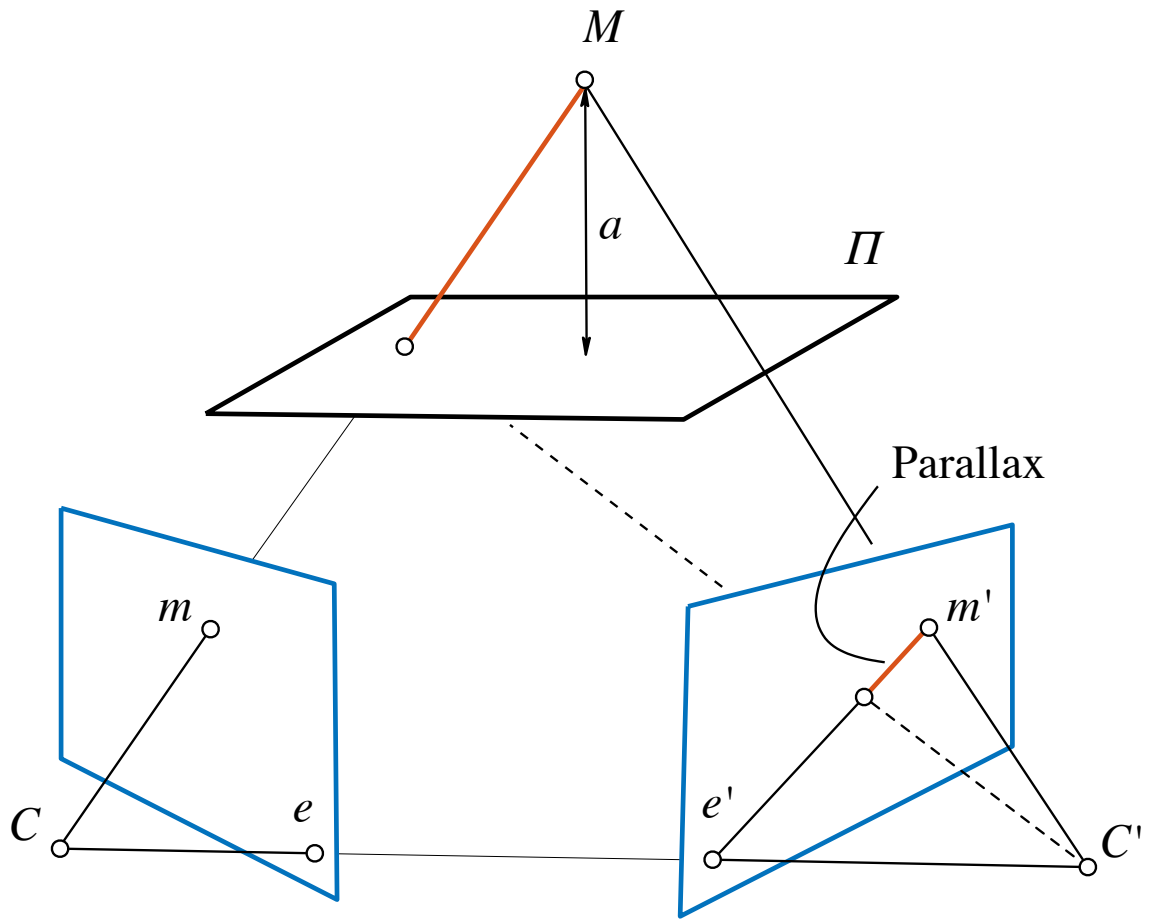
This observation is not only valid for H_∞, but is generalized to any plane. In fact, after a few steps in the wake of the procedure used to derive (106), we obtain:

$$\mathbf{m}' \simeq H_\Pi \mathbf{m} + \left(\frac{a}{d\,\zeta}\right)\mathbf{e}' \tag{112}$$

where a is the distance of the point $\mathbf{M}$ (of which $\mathbf{m}$ and $\mathbf{m}'$ are the projections) from the plane Π and ζ is its depth wrt the first camera.

When $\mathbf{M}$ belongs to the plane Π, then $\mathbf{m}' \simeq H_\Pi \mathbf{m}$, otherwise there is a residual term $\gamma = \dfrac{a}{d\ \zeta}$, called parallax.

The parallax is the projection on the plane of the right camera of the optical ray segment between M and the intersection with Π.



Eq. (112) is an alternative representation of the epipolar geometry, taking a particular plane as a reference. Note that:

- γ is independent of the choice of the second image;
- γ is proportional to the inverse of the depth ζ;
- when the reference plane is the infinite plane $\gamma = \frac{1}{\zeta}$;
- the parallax field is radial and centered in the epipole.



7 Relative orientation

Let's consider the situation where some points of a scene - called in this context tie points - are projected in two different cameras, and for each point in one camera is given the corresponding point in the other camera.

The problem of relative orientation consists in determining orientation (which includes position and angular attitude) of one camera with respect to the other, assuming known intrinsic parameters.

We also say that relative orientation recovers the motion of the camera, implicitly assuming the equivalent scenario in which a static scene is framed by a moving camera with known intrinsic parameters.

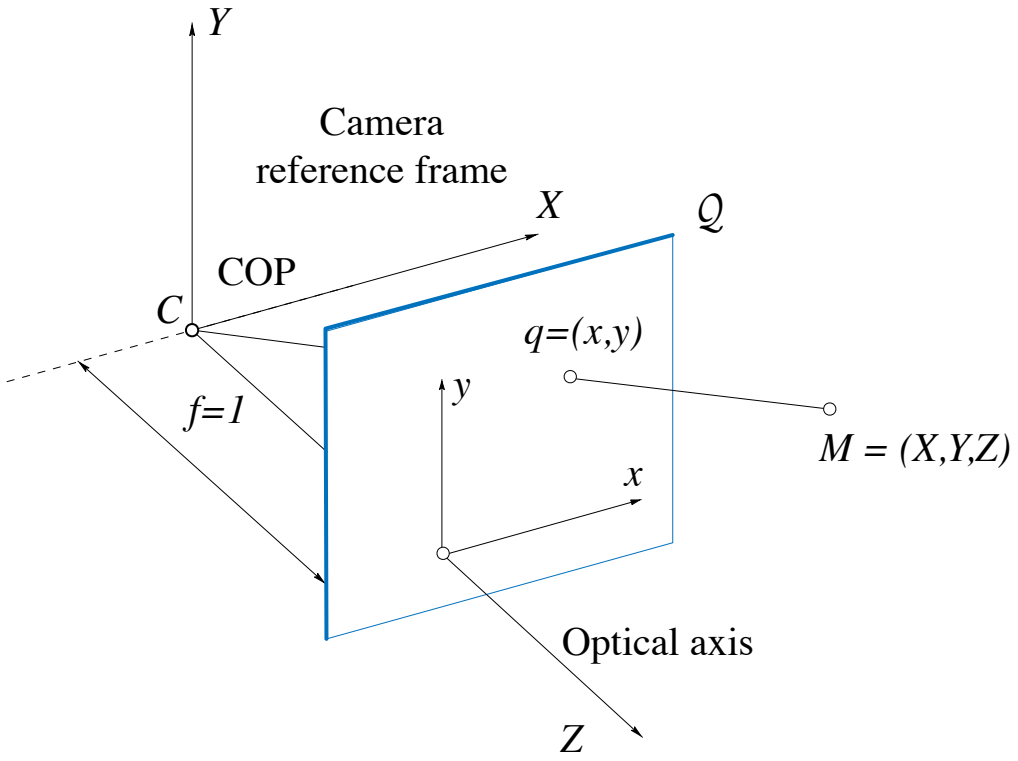
In any case, both orientation and rigid motion are represented by a direct isometry.

The solution to the problem we will describe is based on the essential matrix, which describes the epipolar geometry of two perspective images with known intrinsic parameters and from which the rigid motion of the camera can be obtained with a matrix factorization.

Note that the translational component of the displacement can only be calculated up to a scale factor, because it is impossible to determine (without additional information) if the motion observed in the image is of a close object with the camera that moves slowly or of a distant object with the camera moving faster. This fact is known as depth-speed ambiguity.

Essential matrix

Suppose we have a camera, with known intrinsic parameters, that is moving in a static environment following an unknown trajectory. Let's consider two images taken by the camera in two different instants of time and we assume that we are given a certain number of corresponding points between the two images, in normalized coordinates. The corresponding points are the projections of the tie-points in the two images.



Let P and P' be the PPMs of the cameras that correspond to the two instants of time and $\mathbf{q} = K^{-1}\mathbf{m}$, $\mathbf{q}' = K'^{-1}\mathbf{m}'$ are the normalized coordinates of the corresponding points in the images.

Without loss of generality we represent the 3D points in the reference of the first camera and we therefore write the following two PPMs:

$$P = [I|\mathbf{0}] \quad \text{e} \quad P' = [I|\mathbf{0}]G = [R|\mathbf{t}]. \tag{113}$$

Substituting these two particular PPMs in the Longuet-Higgins equation (92) we obtain the bilinear form which links conjugate points in normalized coordinates:

$$\mathbf{q}'^{\mathsf{T}}[\mathbf{t}]_\times R\mathbf{q} = 0. \tag{114}$$

The matrix

$$E = [\mathbf{t}]_\times R \tag{115}$$

which contains the coefficients of the form is called essential matrix.

Equation (114) is homogeneous with respect to $\mathbf{t}$, ie the modulus of the vector does not count.

This reflects depth-speed ambiguity. We cannot derive the absolute scale of the scene without an additional piece of information, such as the distance between two points.

Degrees of freedom

The essential matrix depends on three parameters for rotation and on two parameters for translation up to a scale factor.

Therefore an essential matrix has five degrees of freedom.

In terms of constraints, we can observe that the essential matrix is defined up to a scale factor and is singular, since $\det[\mathbf{t}]_\times = 0$.

To arrive at the five degrees of freedom obtained from the parameterization we must add other two constraints, which are those described in the Huang and Faugeras (1989) theorem.

Factorization of the essential matrix

Theorem 7.1 *A real 3×3 matrix E can be factored as product of a non-zero antisymmetric matrix and of a rotation matrix if and only if E has two equal, not null singular values and one null singular value.*

Proof. Let $E = UDV^{\mathsf{T}}$ be theSVD 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:

$$D = \begin{bmatrix} 1 & 0 & 0 \\ 0 & 1 & 0 \\ 0 & 0 & 0 \end{bmatrix} = \underbrace{\begin{bmatrix} 0 & 1 & 0 \\ -1 & 0 & 0 \\ 0 & 0 & 0 \end{bmatrix}}_{S'} \underbrace{\begin{bmatrix} 0 & -1 & 0 \\ 1 & 0 & 0 \\ 0 & 0 & 1 \end{bmatrix}}_{R'} \tag{116}$$

where S' is antisymmetric and R' is a rotation matrix (of $-\pi/2$ on the axis z). Therefore

$$E = US'R'V^{\mathsf{T}} = (US'U^{\mathsf{T}})(UR'V^{\mathsf{T}}) \simeq \underbrace{(US'U^{\mathsf{T}})}_{S} \underbrace{\det(UV^{\mathsf{T}})(UR'V^{\mathsf{T}})}_{R}. \tag{117}$$

Taking $S = US'U^{\mathsf{T}}$ and $R = \det(UV^{\mathsf{T}})UR'V^{\mathsf{T}}$, the sought factorization is: $E = SR$.

In fact, the matrix $US'U^{\mathsf{T}}$ is antisymmetric and the matrix $\det(UV^{\mathsf{T}})UR'V^{\mathsf{T}}$ is of rotation. This demonstrates one implication, let's see now the viceversa.

Let $E = SR$ where R is a rotation matrix and S is antisymmetric.

By virtue of proposition A.34, we have $S \simeq US'U^{\mathsf{T}}$. Furthermore from equation (116) we obtain: $S' = \operatorname{diag}(1, 1, 0)R'^{\mathsf{T}}$. So:

$$E \simeq U \operatorname{diag}(1, 1, 0)(R'^{\mathsf{T}}U^{\mathsf{T}}R). \tag{118}$$

Being $(R'^{\mathsf{T}}U^{\mathsf{T}}R)$ orthogonal, this is a decomposition to the singular values of E with two equal singular values and one null, as we wanted to prove.

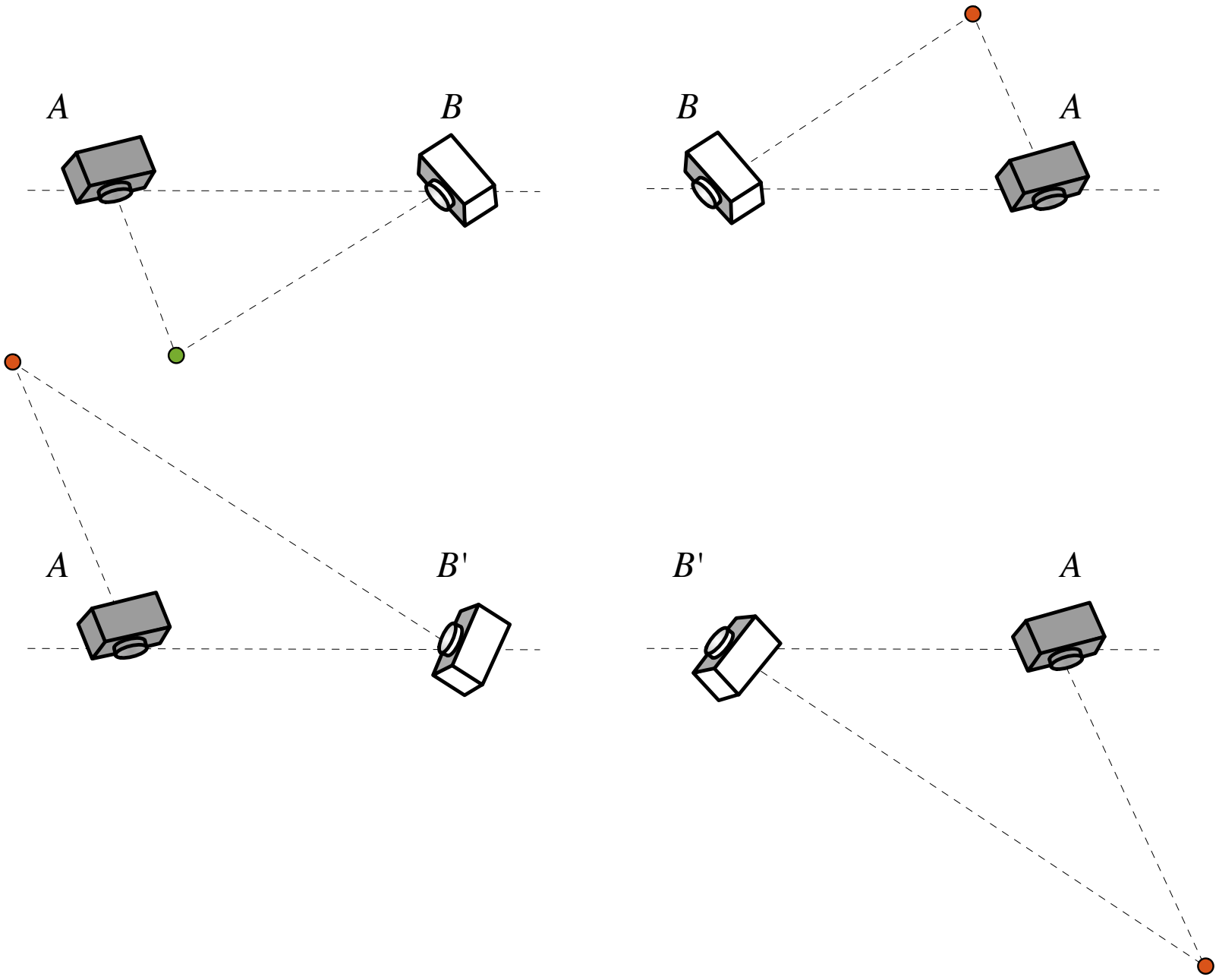
The factorization of E constructed in the first part of the proof is not unique. It is indeed possible to transpose both S' and R' obtaining in all four combinations, of which two change the sign of E (irrelevant, since it is defined up to a multiplication by a scalar). In fact we see that $S'R'^{\mathsf{T}} = -D$, while for the definition of antisymmetric: $S'^{\mathsf{T}} = -S'$.

In summary, therefore, the four possible factorizations are given by:

$$S = \pm U S' U^{\mathsf{T}} \quad \text{e} \quad R = \det(UV^{\mathsf{T}}) U \left\{ \begin{matrix} R' \\ R'^{\mathsf{T}} \end{matrix} \right\} V^{\mathsf{T}}. \tag{119}$$

The geometric interpretation of the four solutions is as follows: the sign change of S swaps the right and left camera, while the choice on R determines a rotation of the camera of 180° around the baseline (that is, the line connecting the COPs).

In all cases it is possible to triangulate the tie-points, but only in one case the tie-points lie in front of both cameras.


A
B
B
A
A
B'
B'
A

Essential and fundamental matrix

The essential matrix and the fundamental matrix are in relationship, since both encode the rigid motion between two cameras. The first relates the normalized coordinates of the conjugate points, while the second maps the pixel coordinates of the conjugate points.

It is easy to verify that:

$$F = K'^{-\mathsf{T}} E K^{-1} \text{ and also } E = K'^{\mathsf{T}} F K. \tag{120}$$

Computing of the essential matrix

The essential matrix can be calculated from at least eight matching points, using the same method described for the fundamental matrix.

It is only a matter of replacing pixel coordinates with normalized coordinates.

Note that due to the inevitable errors that plague the measures, in general, the matrix E found by solving this set of linear equations will not satisfy the requirements of the Huang and Faugeras (1989) theorem, ie it will not have two equal singular values and one null singular value. However this can be forced a posterior via the SVD.

Let E be a matrix 3×3 and $E = UDV^{\mathsf{T}}$ its SVD with $D = \text{diag}(\sigma_1, \sigma_2, \sigma_3)$ and $\sigma_1 \geq \sigma_2 \geq \sigma_3$. It can be shown that

$$\widehat{E} = U \,\text{diag}\left(\frac{\sigma_1 + \sigma_2}{2}, \frac{\sigma_1 + \sigma_2}{2}, 0\right) V^{\mathsf{T}} \tag{121}$$

is the matrix closest to E, in the Frobenius norm, that satisfies the requirements of the theorem.

Although the linear algorithm just described needs at least eight tie-points to compute E, since the matrix depends on five parameters, it is possible, in principle, to compute it with five matches plus the polynomial constraints given by the theorem (7.1).

In fact, Faugeras and Maybank (1990) have shown that this is feasible and that there are ten distinct solutions.

Subsequently, algorithms for the computation of E with five points have been proposed by Nister (2003) and Li and Hartley (2006).

Listing 12. Linear computation of E

```
function E = essential_lin(q2,q1,K1,K2,w)
%ESSENTIAL_LIN  Essential matrix with 8-points algorithm

if nargin < 5 || isempty(w)
    w = ones(size(q1,2),1);  % weights
end

    q1 = K1\ensure_homogeneous(q1);
    q2 = K2\ensure_homogeneous(q2);
    % now q1 and q2 are homogeneous NIC

    E = eight_points(q2(1:2,:), q1(1:2,:),w);

    % enforce constraints on E
    [U,D,V] = svd(E);
    s = (D(1,1)+D(2,2))/2;
    E = U * diag([s,s,0]) * V';

end
```

Listing 13. Relative orientation

```
function [R,t] = relative_lin(q2, q1, K2, K1)
%RELATIVE_LIN Relative orientation

q1 = K1\ensure_homogeneous(q1);
q2 = K2\ensure_homogeneous(q2);
% now q1 and q2 are homogeneous NIC

E = eight_points(q2(1:2,:), q1(1:2,:)) ;

[U,~,V] = svd(E);

S1 = [0 1 0; -1 0 0; 0 0 0];
R1 = [0 -1 0; 1 0 0; 0 0 1];

for j = 1:4
    % 4 combinations
    S = (-1)^j * U*S1*U';

    if j==3   R1 = R1'; end

    t=[S(3,2) S(1,3) S(2,1)]';
    R = det(U*V')*U*R1*V';

    % solve t = z2*q2 - z1*R*q1 for z1 and z2
    z = [ q2(:,1), -R*q1(:,1)] \t;
    % 3D points must be in front of both cameras
    if z(1)>0 && z(2)>0  break,  end

end
```

Go to live coding.

8 Reconstruction from two images

We will deal in this chapter with triangulation, which allows to get the 3D coordinates of the points that have been placed in correspondence in the images.

The set of points obtained is also called a model.

We will see that the model differs from the true one by transformations that reflect our degree of knowledge about the sensor and/or the scene.

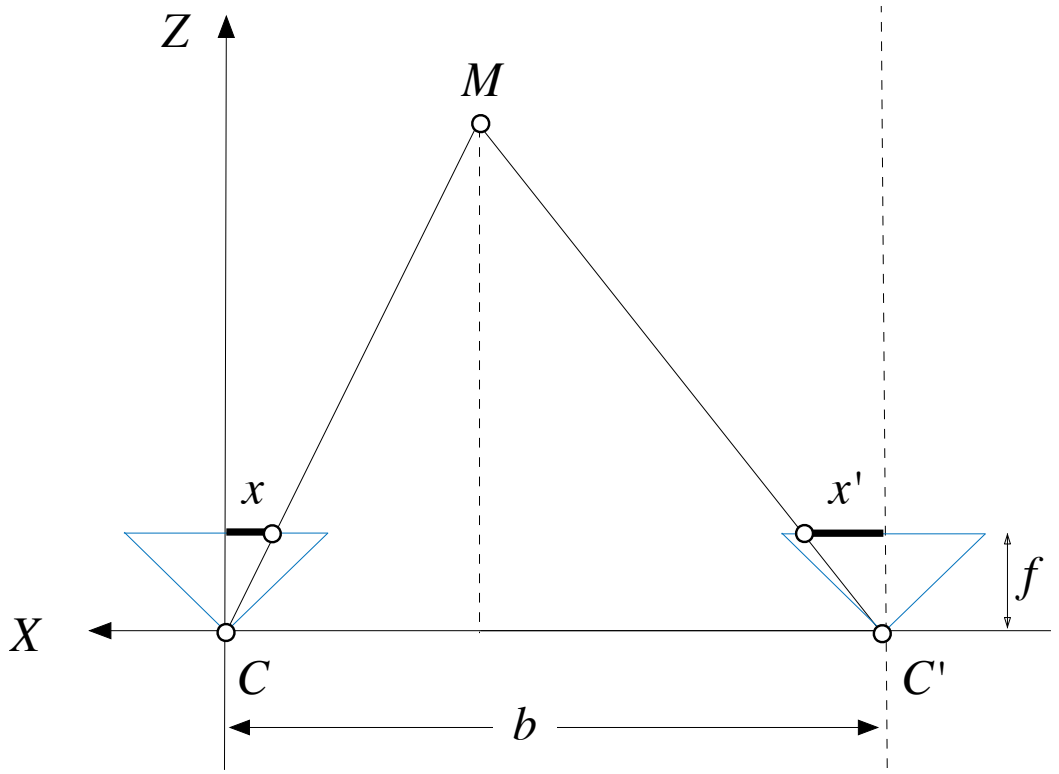
Triangulation

Given the matching between the conjugate points in the two images and given the two PPM, the position in space of the points that are projected on the two images can be reconstructed by triangulation (known also as intersection)

The difference in the position of the conjugate points in the respective images is called binocular disparity.

Normal case

Before dealing with triangulation in the more general case, let us see what can be learned from a simplified case like the following, also called normal case. Consider two parallel and aligned cameras (coincident image planes); it is easy to verify that we have a purely horizontal disparity and therefore a two-dimensional construction is justified.



Let f be the focal length and b the distance between the COP, called baseline. Let us arbitrarily set the world reference coinciding with that of the right camera, we can write the following perspective projection equations:

$$\begin{cases} \frac{1}{Z} = \frac{-x}{X} \\ \frac{1}{Z} = \frac{-x'}{X-b} \end{cases} \tag{122}$$

from which, elaborating, it is obtained

$$Z = \frac{b}{x' - x}. \tag{123}$$

This suggests that it is possible to derive the third coordinate Z, given (i) the parameters of the system (b in this case) and (ii) the binocular disparity ($x' - x$).

Also note that if parameter b is unknown, the reconstruction of the three-dimensional model us up to a scale.

Linear-eigen method

Let us now see how triangulation is achieved in the general case. Given

- the coordinates (in pixels) of two conjugate points and
- the two PPM of the two cameras,

one can reconstruct the position in world coordinates of the point of which the two conjugate points are the projection.

Let us consider $\mathbf{m} = [u, v, 1]^{\mathsf{T}}$, the projection of the point M on the camera whose PPM is P. From the equation of perspective projection is obtained:

$$[\mathbf{m}]_{\times} P\mathbf{M} = \mathbf{0}. \tag{124}$$

There are three equations in the four unknowns of $\mathbf{M}$ (the fourth component of $\mathbf{M}$ is left free and appear as unknown), but only two of them are linearly independent, being $\mathrm{r}([\mathbf{m}_i]_{\times}) = 2$.

Let us now consider also $\mathbf{m}' = [u', v', 1]^{\mathsf{T}}$, the conjugate point of m in the second image, and let P' be the second PPM. Being m and m' projection of the same point M, the equations given by both can be stacked, obtaining a homogeneous linear system of six equations (of which no more than four are independent) in four unknowns:

$$\underbrace{\begin{bmatrix} [\mathbf{m}]_\times P \\ [\mathbf{m}']_\times P' \end{bmatrix}}_{A} \mathbf{M} = \mathbf{0}_{4\times 1}. \tag{125}$$

The solution is the kernel of the 6×4 A matrix, which therefore must have rank three, otherwise there would be only the trivial solution $\mathbf{M} = \mathbf{0}$.

In the presence of noise, however, the condition on the rank does not come exactly satisfied and therefore a LS solution is sought, obtained as the least right singular vector of the SVD of A. Hartley and Sturm (1997) call this method " *linear-eigen* ".

The above is generalized to the case of $m > 2$ images: each image adds two equations and one gets a homogeneous system of $2m$ equations in four unknowns.

Listing 14. Triangulation

```
function M = triang_lin(P, m)
%TRIANG_LIN Triangulation for one point in multiple images

    L=[]; % stack equations for every camera
    for j=1:length(P)
        L = [L; skew([m{j};1])*P{j} ]  ;
    end
    [~,~,V] = svd(L);
    M =  V(:,end)./V(end,end);
    M = M(1:3);
end
```

Go to live coding.

Ambiguity of the reconstruction

Let us consider a set of 3D points, seen by 2 cameras with matrices P and P'. Let $\mathbf{m}_j$ be the (homogeneous) coordinates of the projection of the j -th point in the first image and $\mathbf{m}'_j$ the (homogeneous) coordinates of the projection of the j -th point in the second image.

The reconstruction problem ofcan be states as follows: given the coordinates (in pixels) of the conjugate points $\mathbf{m}_j$ and $\mathbf{m}'_j$ find the PPM $\{P, P'\}$ and the model $\{\mathbf{M}_j\}$ s.t .:

$$\mathbf{m}_j \simeq P\mathbf{M}_j \text{ and } \mathbf{m}'_j \simeq P'\mathbf{M}_j. \tag{126}$$

Please note that if the PPM are known, the model can be obtained by triangulation.

We will also denote as reconstruction the pair formed by the set of PPM and the 3D model.

If $\{P, P'\}$ and $\{\mathbf{M}^j\}$ are a reconstruction, that is they satisfy (126), also $\{P, P'\}T$ and $\{T^{-1}\mathbf{M}^j\}$ satisfy (126) for any non-singular matrix T 4×4.

The matrix T specifies a linear transformation of 3D projective space.

However, if PPM P and P' are non-generic, T must not alter their properties, and therefore any constraints on the PPM reflects in constraints on T.

If the PPM are fully known (up to the usual irrelevant scale factor), T can only be identity. In fact, every other T would change the PPM. The resulting reconstruction is identical to the true one, expressed in the same reference system in which the orientation of the cameras is given (eg in the reference in which control points were expressed during calibration).

If the PPM are known up to the same isometry, it means that the exterior orientation is unknown but the relative one is given. In this case T is constrained to be a direct isometry, which leaves the relative orientation unchanged.

Correspondingly, the points are pre-multiplied by T^{-1}, which is itself a direct isometry. The resulting reconstruction differs from the real one for a (unknown) direct isometry and is called metric reconstruction.

Note that in this case the interior orientation is known, so the PPM must have the structure $K[R, t]$ with $R \in SO(3)$, properties that direct isometries indeed preserve.

In practice, relative orientation with known translation module only occurs in special cases such as eg a pair of cameras rigidly mounted on a beam. Much more often one have to deal with a moving camera, for which we proceed to the recovery of the relative orientation, up to the module of translation.

In this case T is a similarity and it also includes a global scaling, which affects only the translation component of PPM (whose module is unknown). The reconstruction that is obtained differs from the true one by a similarity and is called Euclidean reconstruction.

When we do not know anything about the PPM, T can be any non-singular matrix, and a reconstruction is obtained defined up to an arbitrary projectivity, and for this reason we call it projective reconstruction.

Euclidean reconstruction

Let us consider a single camera in motion; intrinsic parameters (ie, K) are known, but the motion of the camera is unknown (ie, extrinsic parameters are missing).

In this case a Euclidean reconstruction can be obtained, as we have already discussed.

The procedure for obtaining this reconstruction is also called Structure from Motion (SFM) in the literature.

It is based on retrieving the relative orientation $R, \widehat{\mathbf{t}}$ ($\mathbf{t}$ is normalized to unit) and on the observation that the pair of PPM

$$P = K[I \mid \mathbf{0}] \quad \text{e} \quad P' = K[R \mid \mathbf{t}] \tag{127}$$

yields the given Essential matrix: $E = [\widehat{\mathbf{t}}]_\times R$.

Once the two PPM are instantiated, the model is obtained by triangulation.

This pair is not unique: we can obtain the same E multiplying the two PPM by the same similarity T.

The Euclidean reconstruction that is obtained is defined up to a similarity, which is arbitrarily fixed by choosing (eg) $P = K[I \mid \mathbf{0}]$ and $\|\mathbf{t}\| = 1$.

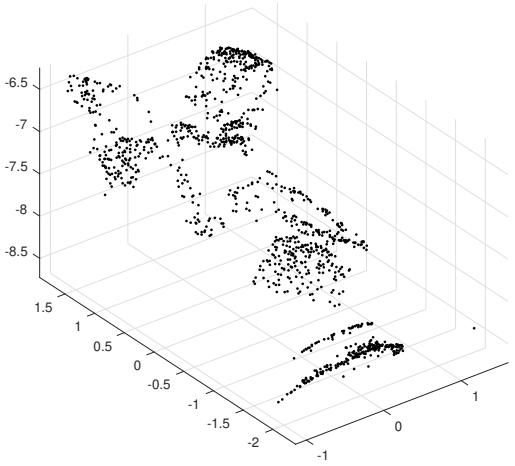
The upgrade from Euclidean to metric reconstruction can be achieved as soon as the baseline is known, or the distance between two 3D points, which fix the unknown scale of the model.

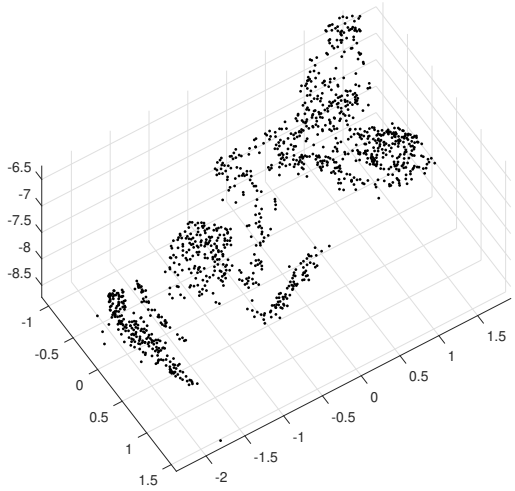
To step up to the identical reconstruction one must also determine the unknown isometry, through the knowledge of at least three control points.

Listing 15. Two views reconstruction

```
%% Two views reconstruction

% Compute (load) homologous points  m1 and m2
load matches

% Intrinsic parameters from calibration
K =  [2864.8, 0, 636.7; 0, 2864.8, 931.9; 0,0,1];

% Relative orientation
[R,t] = relative_lin(m2, m1, K, K);
[R,t] = relative_nonlin(R, t ,m2, m1, K, K);

% Canonical camera pair
PPM{1} = K*[eye(3),zeros(3,1)];
PPM{2} = K*[R,t];

% Triangulation
M_out = triang_lin_batch(PPM, {m1,m2});
M_out = triang_nonlin_batch(M_out, PPM, {m1,m2});
% Done
```

Go to live coding.


-6.5
-7
-7.5
-8
-8.5
1.5
1
0.5
0
-0.5
-1
-1.5
-2
-1
0
1


-6.5
-7
-7.5
-8
-8.5
-1
-0.5
0
0.5
1
1.5
-2
-1.5
-1
-0.5
0
0.5
1
1.5

Projective reconstruction

When the intrinsic parameters are unknown the only information that can be extracted from a pair of images are the correspondences of the points and from these it is possible to compute the fundamental matrix. Following *mutatis mutandis* the procedure for the essential matrix, we arrive at a reconstruction of the scene, defined however up to a projectivity of the space.

A fundamental matrix can be factored into

$$F = [\mathbf{e}']_\times A \tag{128}$$

with a suitable matrix A, which is said compatible with F. There is a similarity with factoring $E = [\mathbf{t}]_\times R$. Unfortunately, this factoring is not unique. Indeed, if a matrix A is compatible with F then $A + \mathbf{e}'\mathbf{v}^\mathsf{T}$ is also compatible, for each vector $\mathbf{v} \in \mathbb{R}^3$, since

$$[\mathbf{e}']_\times (A + \mathbf{e}'\mathbf{v}^\mathsf{T}) = [\mathbf{e}']_\times A + [\mathbf{e}']_\times \mathbf{e}'\mathbf{v}^\mathsf{T} = [\mathbf{e}']_\times A. \tag{129}$$

If A is any matrix compatible with F, it is easily shown that the following pair of PPM:

$$P = [I \mid \mathbf{0}] \quad \text{e} \quad P' = [A \mid \mathbf{e}'] \tag{130}$$

yield the given fundamental matrix. Once the two PPM have been instantiated, the (projective) model is obtained by means of triangulation.

The problem remains of how to get any matrix compatible with F. We will see that any plane-induced homography, including H_∞, is compatible with F. Another matrix compatible with F but which is not a homography (being singular) is the following:

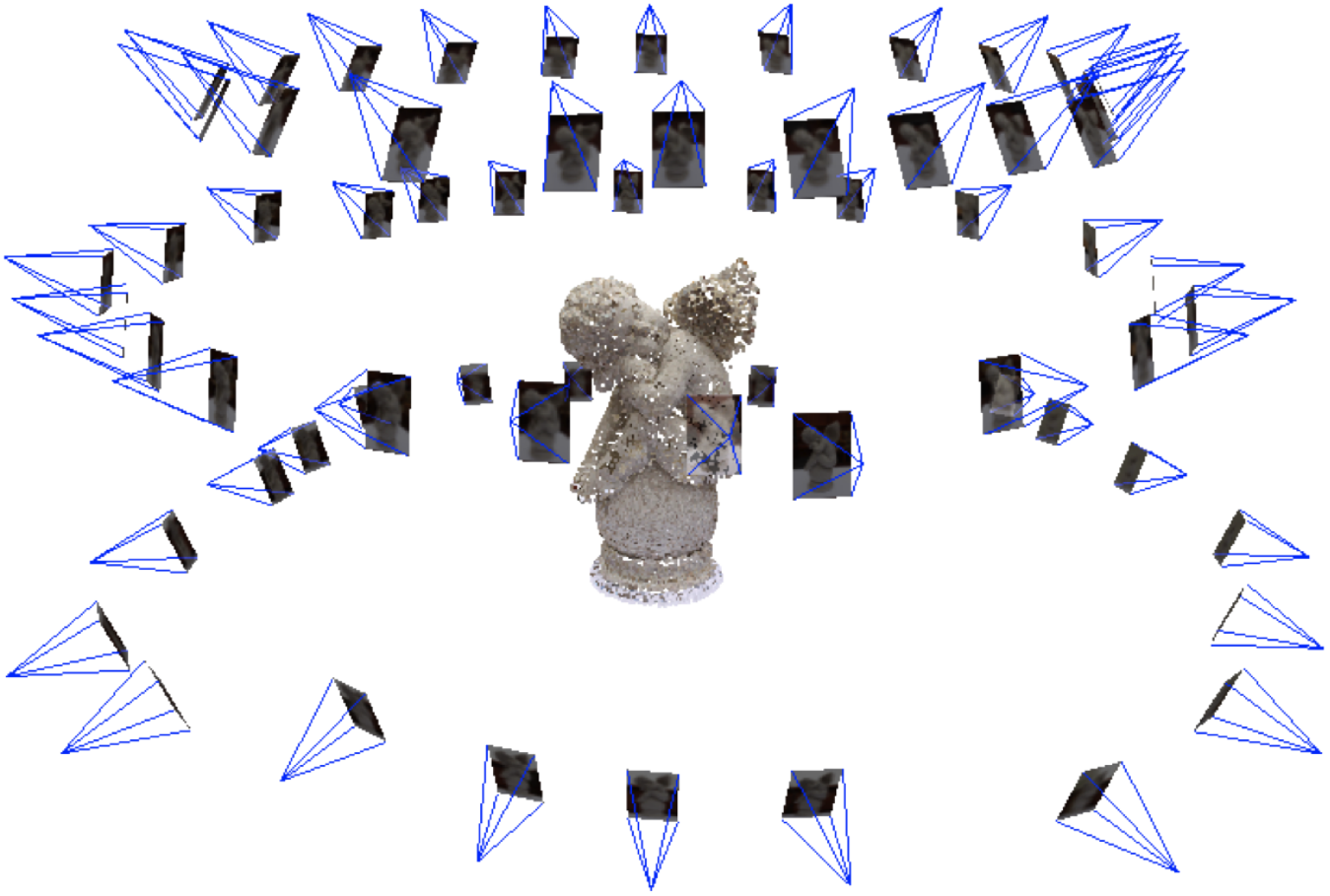
$$S = -\frac{1}{\|\mathbf{e}'\|}[\mathbf{e}']_\times F. \tag{131}$$

With $A = S$ in (130) we obtain the so-called canonical representation, introduced by Luong and Viéville (1996).

The upgrade from projective reconstruction to Euclidean can be obtained from the knowledge of intrinsic parameters, or through a process known as self-calibration.

One can go directly to the identical reconstruction through the knowledge of at least five control points, which allow to compute a projectivity of the space, eg with the DLT algorithm.

9 Multi-view reconstruction



Let us now tackle the problem of reconstruction in the case of $N > 2$ (calibrated) images.

Also known as recovery of structure and motion or *structure from motion*), where "motion" means the extrinsic parameters of a set of cameras.

In the case of 2 (calibrated) images we know how to achieve a reconstruction through the computation of E, its factorization and triangulation.

How this process generalizes to the case $N > 2$?

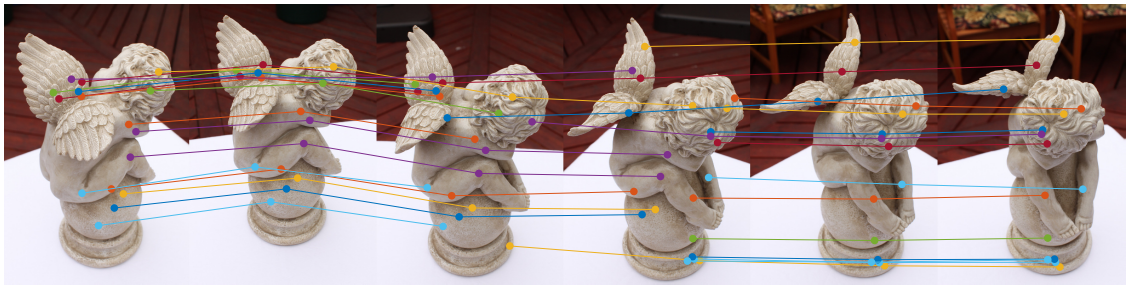
Epipolar graph

Given an (unordered) set of $m > 2$ images, we assume that for a number of image pairs (ideally whenever possible) a set of corresponding points perturbed by noise and containing a significant percentage of outliers is available. Those points are typically produced by a detector and are matched automatically.

The preliminary problem arises of establishing which of them form epipolar pairs and to compute the essential or fundamental matrix (depending on whether the interior orientation are known or not).

Matches in image pairs are then propagated to the whole set, obtaining the so-called tracks, that span multiple images.

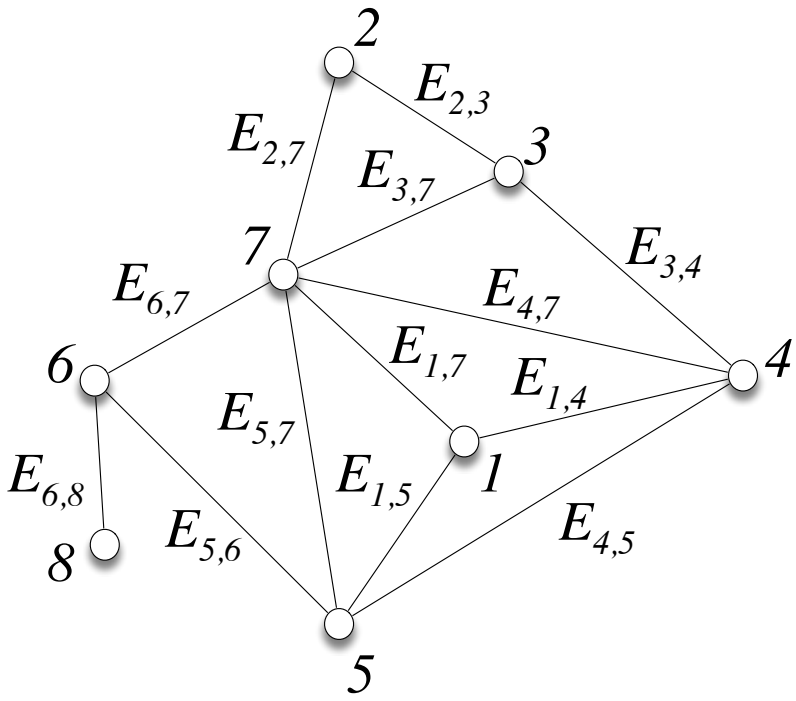
The corresponding 3D points are called tie-points: they are those that will constitute the model or structure.



We represent the information available through the epipolar graph $G = (V, E)$, whose nodes V are the m images and the ℓ edges E correspond to the calculated essential/fundamental matrices.

The adjacency matrix A of the graph contains 1 in the entry (i, j) if E_{ij} is available, 0 otherwise.

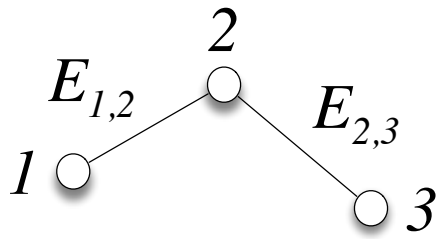
We can associate a weight in $[0, 1]$ to the edges proportional to number of corresponding points between the two images, or, inversely, a cost.



The case of three images

It is useful to start by considering the case of $m = 3$ images.

We assume to know the essential matrices relating to the pairs of images (1,2) and (2,3).

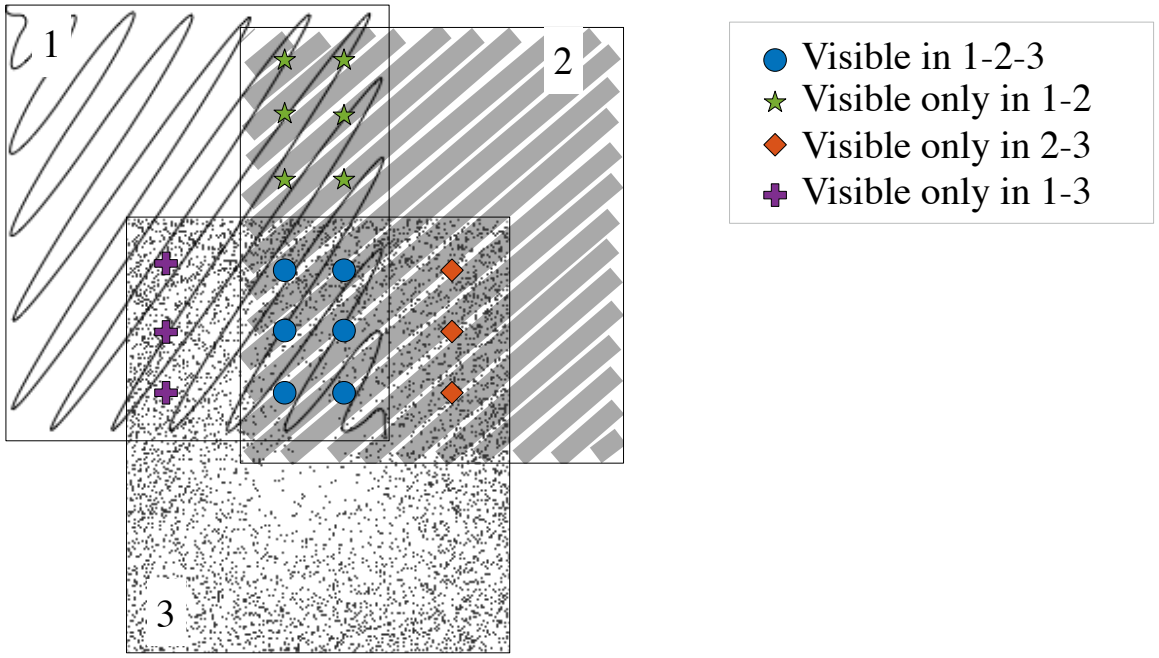


Idea 1. We proceed in an incremental way.

We apply the reconstruction process from two images to the pair (I_1, I_2), obtaining a model.

Assuming that some points of the model are visible in I_3, they can be used to orient I_3 by solving an exterior orientation (with resection).

The model is then updated and augmented (by intersection) thanks to the contribution of I_3.



Idea 2. Independent models adjustment.

Let us independently apply the procedure for the reconstruction from two images to each of the two pairs (I_1, I_2) and (I_2, I_3)

We get two independent models separated by a similitude (both fix the world reference on the first camera of the pair) with some points in common.

We align the two models (and therefore the reconstructions) by solving an absolute orientation and merging them into one.

Idea 3. Synchronization

We aim to instantiate three PPM consistent with each other and proceed to the final triangulation.

From the factorization of the essential matrices relating pairs 1-2, and 2-3 we obtain two rigid motions $(R_{21}, \hat{\mathbf{t}}_{21})$, and $(R_{32}, \hat{\mathbf{t}}_{32})$ in which each translation is known up to the magnitude, and for this we consider the versor, denoted by $\widehat{\ }$.

If the modules of the translations were known, we could concatenate the isometries

$$R_{31} = R_{32}R_{21}, \tag{132}$$

$$\mathbf{t}_{31} = \mathbf{t}_{32} - R_{32}\mathbf{t}_{21} \tag{133}$$

and then instantiate three PPM: $P_1 = [I, \mathbf{0}], P_2 = [R_{21}, \mathbf{t}_{21}]$ and $P_3 = [R_{31}, \mathbf{t}_{31}]$ consistent with each other and reconstruct the model by triangulation.

This concatenation will later evolve into the concept of synchronization.

Unfortunately, however, the composition does not work for versors.

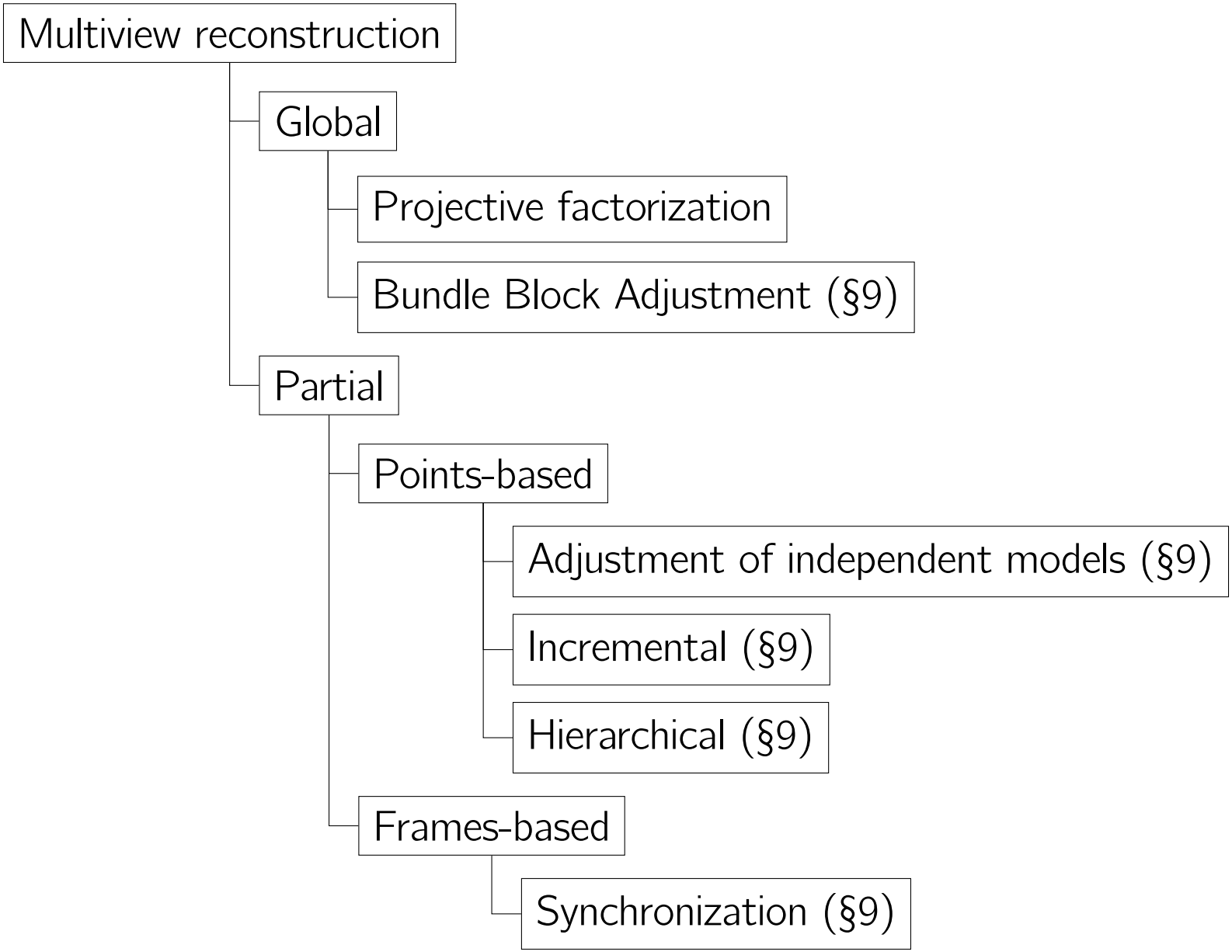
However, if we have R_{31} and $\hat{\mathbf{t}}_{31}$ available, we can exploit (133) to derive the ratio between the modules, rewriting it as

$$\lambda \hat{\mathbf{t}}_{31} = \mu \hat{\mathbf{t}}_{32} - R_{32} \hat{\mathbf{t}}_{21} \tag{134}$$

where $\lambda = \|\mathbf{t}_{31}\| / \|\mathbf{t}_{21}\|$ and $\mu = \|\mathbf{t}_{32}\| / \|\mathbf{t}_{21}\|$, and obtaining the unknowns λ and μ:

$$\mu = \frac{(\hat{\mathbf{t}}_{13} \times \hat{\mathbf{t}}_{23})^{\mathsf{T}} (R_{23} \hat{\mathbf{t}}_{12} \times \hat{\mathbf{t}}_{23})}{\|R_{23} \hat{\mathbf{t}}_{12} \times \hat{\mathbf{t}}_{23}\|^2}. \tag{135}$$

A global scale factor remains indeterminate, and is arbitrarily fixed through the norm of $\mathbf{t}_{12}$, as in the case of two images.


Multiview reconstruction
Global
Projective factorization
Bundle Block Adjustment (§9)
Partial
Points-based
Adjustment of independent models (§9)
Incremental (§9)
Hierarchical (§9)
Frames-based
Synchronization (§9)

Point-based approaches

Point-based approaches employ points for computing the alignment, i.e. they use them to solve problems of orientation (relative, exterior, absolute),

Adjustment of independent models

This method, also used in photogrammetry (Crosilla and Beinat, 2002), is based on the construction and subsequent integration of many models independently built from images pairs.

- The images are grouped in pairs (with overlaps) and many independent models are calculated by reconstructing from two frames;
- These models differ by a similarity: shared 3D points are used to bring them back to a single reference system (with scale).

The pairs of images correspond to a subset of the edges the epipolar graph. For instance, take the edges of a minimum cost spanning tree.

With the tools we have seen so far, the last step could be implemented with a series of cascaded OPA; this method however it is sub-optimal.

We will see how the alignment of two point clouds with OPA can be generalized to the simultaneous alignment of many point clouds, with the GPA.

Generalized Procrustian analysis

If we have correspondences between the different clouds of points, we can merge them in a single operation, applying to each the similarity that brings it into a common reference by generalizing the absolute orientation through OPA we have seen before.

This technique is in fact referred to as Generalized Procrustian Analysis (GPA).

Given the matrices $M_1, M_2, \ldots M_m$ representing the m sets of points $\mathcal{M}_1, \mathcal{M}_2, \ldots \mathcal{M}_m$, the function to be minimized to bring them into alignment is the following

$$\sum_{i=1}^{m} \sum_{j=i+1}^{m} \|s_i(R_i M_i + \mathbf{t}_i \mathbf{1}^\mathsf{T}) - s_j(R_j M_j + \mathbf{t}_j \mathbf{1}^\mathsf{T})\|_F^2 \tag{136}$$

where the unknowns are the parameters of the similarity: $s_i, R_i, \mathbf{t}_i$.

The GPA is used for registering *range* images (with direct isometry) or for independent model adjustment (with similarity). The latter are in fact comparable to clouds of points obtained from *range* sensors, with the difference of the scale.

At the heart of this technique lies the following observation of Commandeur (1991):

$$\sum_{i=1}^{m}\sum_{j=i+1}^{m}\|B_i - B_j\|_F^2 = m\sum_{i=1}^{m}\|B_i - C\|_F^2, \tag{137}$$

where C is the mean or centroid

$$C = \frac{1}{m}\sum_{i=1}^{m} B_i. \tag{138}$$

The minimization of the residue rewritten as in (137) results in an iterative process, in which alternatively

- we compute C
- align the sets $\mathcal{M}_i$ on C, each separately with OPA.

The convergence of the algorithm is demonstrated in (Commandeur, 1991), although not necessarily at the global minimum.

In the realistic case that not all matches are available in all views, multiply each M_i on the right with a diagonal matrix W_i which contains on the diagonal 1 where the corresponding column of M_i is specified and 0 otherwise (missing data).

The formulae for the solution of absolute orientation are modified *mutatis mutandis*, with the trick that also the vector $\mathbf{1}$ must be replaced with $W\mathbf{1}$ and that the formula for the centroid becomes (Crosilla and Beinat, 2002):

$$C = \left(\sum_{i=1}^{m} B_i W_i \right) \left(\sum_{i=1}^{m} W_i \right)^{-1} . \tag{139}$$

Listing 16. GPA

```
function  [G,D] = gpa(M,w)
%GPA Generalized Procrustes analysis
% with similarity (7 parameters)

m = length(M);

% constants
MaxIterations = 50;
FunctionTol = 1e-3;
StepTol = 1e-16;

G = mat2cell(repmat(eye(4),1,m),4,4*ones(1,m));
res_n = cell(1,m); res = Inf;

for iter = 1: MaxIterations

    % compute centroid
    D = compute_centroid(M, w);

    % align each M{i} onto centroid
    for i = 1:m
        [R,t,s] = opa(D,M{i},w{i});
        G{i} = [R, t; 0 0 0 1/s] * G{i};
        M{i} = s*(R*M{i} + t);
        res_n{i} = sum(sum((bsxfun(@times,w{i}',(M{i}-D))).^2));
    end

    prevres=res;
    res = sum([res_n{:}])/m; % norm(res)
    if norm(res-prevres)<StepTol || norm(res)<FunctionTol
        break; end

end
end
```

Incremental reconstruction

Starting from an initial reconstruction from two images, the incremental approach grows it by iterating between the following two steps:

- orient one image relative to the current model via resection;
- update the model through triangulation (or intersection).

For this reason this method is also called resection-intersection.

In the algorithm initialization step two images are chosen with a criterion that must reconcile a high number of conjugated points with a sufficient separation of the two cameras (if too close the triangulation results badly conditioned).

Reconstruction is performed from two frames.

The sequential order of frames to process can be obtained, for example, by a visit (depth-first) of the minimum cost support tree of the epipolar graph (starting from the second image of the initial pair).

After initialization, the algorithm enters the main loop of resection-intersection, in which the next image $i > 2$ is added to the reconstruction.

Resection. The correspondences relative to the points already reconstructed are used to obtain 2D-3D matches. On the base of these, the exterior orientation of camera i wrt to the current model is estimated;

Intersection. The current model is then updated in two ways:

- the position of the existing 3D points that have been observed in the current image is re-triangulated (adding one equation triangulation becomes more precise);
- new 3D points are triangulated thanks to the correspondences that become available after the image is added.

The cycle ends when all images have been considered.

We have described the calibrated case, but the same scheme applies, *mutatis mutandis*, also to non-calibrated images.

Hierarchical reconstruction

A variant of the incremental reconstruction was proposed by Gherardi et al. (2010).

Instead of being sorted in a sequence, images are grouped hierarchically, giving rise to a tree (or dendrogram) in which the leaves are the images and the nodes represent groupings (*cluster*) .

The distances used in *clustering* derive from the weighted adjacency matrix of the epipolar graph, so the images come grouped by the number of corresponding points.

We proceed starting from the leaves of the tree:

1. in the nodes where two leaves join (pairs of images) we proceeds with the reconstruction;
2. in the nodes where a leave joins an internal node, the reconstruction is incremented by adding a single frame by resection and updating the model via intersection (as in the sequential method);
3. when two internal nodes joins, two independent reconstructions are merged by solving a absolute orientation (with OPA).

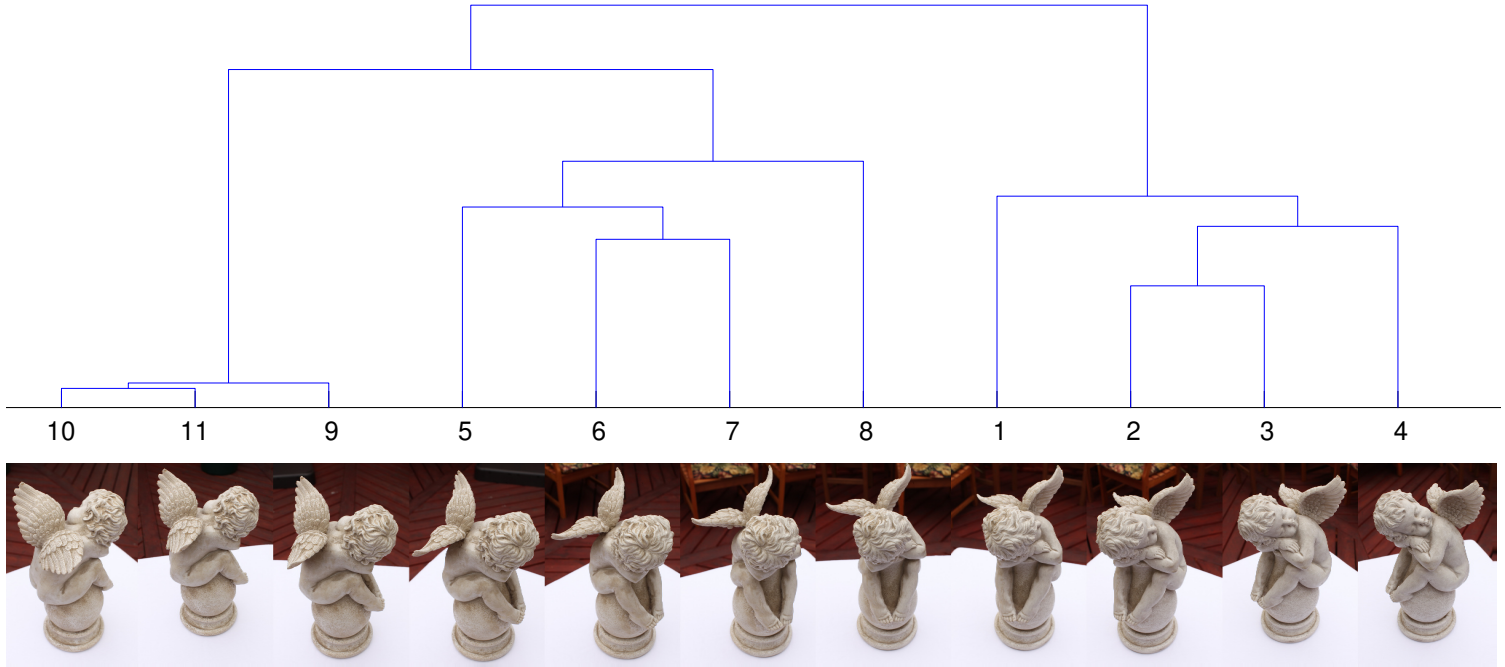
Step 1 is identical to the initialization the sequential method, but it is performed on many pairs instead of just one;

Step 2 coincides with the iteration of the sequential method.

Step 3, on the other hand is reminiscent of the adjustment of independent models.

If the tree is perfectly balanced we get a cascade of absolute orientations (like an adjustment of independent models).

If the tree is totally unbalanced we get the sequential method.



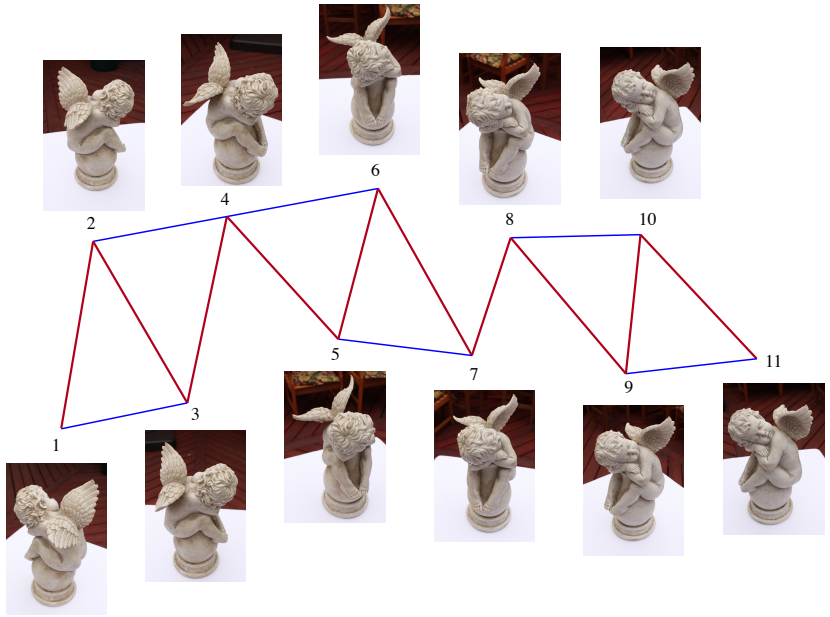
Approach based on Synchronization

The approach in this section is based on the network of relative transformations between reference systems, and computes the model only at the end.

It aims to instantiate the PPM:

$$P_i = K_i[R_i|\mathbf{t}_i] \quad i = 1 \dots m > 3 \tag{140}$$

given the (error-prone) estimates of a number of relative orientations $\{R_{ij}, \hat{\mathbf{t}}_{ij}\}$, which correspond to edges of an epipolar graph.



Let's assume for now that the modules of the translations are known. We will deal with the problem later.

If we take image I_1 as a reference, the $R_{i1} = R_i$ and $\mathbf{t}_{i1} = \mathbf{t}_i$ which are used to instantiate the PPMs are not directly available, since, in general, image I_1 has no overlap with all the other $m - 1$.

They must be calculated by composing the relative orientations, e.g. along a spanning tree for the epipolar graph with root in I_1.

In this way each $R_{i1}, \mathbf{t}_{i1}$ is uniquely determined (in a tree there exists only one path between two nodes) but redundant measures are not taken into account, resulting in the relinquishment to compensate the error.

The procedure of synchronization, on the other hand, implements a globally adjusted solution taking into account all the orientations available as edges of the epipolar graph.

Before proceeding, let's fix the notation and some relationships.

We associate to each node i of the epipolar graph the exterior orientation of the corresponding camera, expressed as position and angular attitude with respect to an world reference. Such orientation can be assigned by a matrix M_i that represents a direct isometry. It is the inverse of the usual matrix G_i that we find in the definition of PPM $P_i = K[I, \mathbf{0}]G_i$, that is:

$$M_i = G_i^{-1} = \begin{bmatrix} R_i & \mathbf{t}_i \\ \mathbf{0} & 1 \end{bmatrix}^{-1} = \begin{bmatrix} R_i^\mathsf{T} & -R_i^\mathsf{T}\mathbf{t}_i \\ \mathbf{0} & 1 \end{bmatrix} = \begin{bmatrix} R_i^\mathsf{T} & \widetilde{\mathbf{C}}_i \\ \mathbf{0} & 1 \end{bmatrix} \tag{141}$$

Note that in M_i it appears the center of the camera i, $\widetilde{\mathbf{C}}_i$.

The edge (i, j) will be labeled with the relative orientation deduced from the essential matrix $E_{ij} = [\mathbf{t}_{ij}]_\times R_{ij}$, defined from $\mathbf{p}_i^\mathsf{T} E_{ij} \mathbf{p}_j = 0$:

$$M_{ij} = \begin{bmatrix} R_{ij} & \mathbf{t}_{ij} \\ \mathbf{0} & 1 \end{bmatrix}. \tag{142}$$

The following compatibility relationship exists:

$$M_{ij} = M_i^{-1} M_j = G_i G_j^{-1} \tag{143}$$

which, considering rotation and translation separately, becomes:

$$R_{ij} = R_i R_j^{\mathsf{T}} \tag{144}$$

$$\mathbf{t}_{ij} = -R_{ij}\mathbf{t}_j + \mathbf{t}_i = R_i \widetilde{\mathbf{C}}_j - R_i \widetilde{\mathbf{C}}_i = R_i(\widetilde{\mathbf{C}}_j - \widetilde{\mathbf{C}}_i). \tag{145}$$

In the following paragraphs we will focus on rotations first and then on translations. As we have observed in the case of three images, composing isometries works only if the translations are known with their magnitude, but this condition is false for the relative orientation derived from essential matrices.

We will ignore this issue by now.

Rotation synchronization

Let's initially focus on the recovery of the angular attitude of the cameras: the aim is to compute the rotations $R_i \in SO(3)$ that satisfy the following compatibility constraint with the measures:

$$R_{ij} = R_i R_j^{\mathsf{T}} \tag{146}$$

for all available pairs (i, j). The problem is known in literature as rotation averaging (Hartley et al., 2013) or rotation synchronization (Singer, 2011).

We immediately notice that the solution is defined up to an arbitrary rotation of all R_i. To be consistent with the solution for two images we can fix it so that $R_1 = I$.

In the presence of noise, we want to solve a minimum problem:

$$\min_{R_1,\dots,R_m \in SO(3)} \sum_{(i,j)} \| R_{ij} - R_i R_j^{\mathsf{T}} \|_F^2. \tag{147}$$

We will now see a solution based on the spectral decomposition of one matrix that solves a relaxation of the problem (147).

For this purpose, we introduce the following matrix $3m \times 3m$ that contains all the relative rotations (we initially assume that all the $\binom{m}{2}$ pairs of images allow the computation of relative orientation):

$$Z = \begin{bmatrix} I & R_{12} & \dots & R_{1m} \\ R_{21} & I & \dots & R_{2m} \\ \dots & & & \dots \\ R_{m1} & R_{m2} & \dots & I \end{bmatrix}. \tag{148}$$

Note that $R_{ij} = R_{ji}^{\mathsf{T}}$, therefore Z is symmetric. X is the matrix $3m \times 3$ constructed by concatenating the unknowns matrices:

$$X = \begin{bmatrix} R_1 \\ R_2 \\ \dots \\ R_m \end{bmatrix}. \tag{149}$$

From the compatibility constraint (146) it follows that Z can be decomposed into:

$$Z = XX^{\mathsf{T}}. \tag{150}$$

Hence $\text{rank}(Z) = \text{rank}(X) = 3$ and Z possesses three non-zero eigenvalues.

We multiply both sides by X to get:

$$ZX = XX^{\mathsf{T}}X = mX \tag{151}$$

as $X^{\mathsf{T}}X = mI$.

The relation (151) tells us that the columns of X are i three eigenvectors corresponding to the non-zero eigenvalues of Z, which coincide with m.

In a practical scenario hardly all relative rotations are available; in this case the blocks of Z corresponding to unknown rotations are set to zero

The i-th block of ZX writes

$$\begin{bmatrix} R_{i1} & R_{i2} & \dots & R_{im} \end{bmatrix} \begin{bmatrix} R_1 \\ R_2 \\ \dots \\ R_m \end{bmatrix} = R_{i1}R_1 + R_{i2}R_2 + \cdots + R_{im}R_m = d_i R_i \qquad (152)$$

where d_i is the number of nonzero blocks in block-row i of Z, which is equal to m if all the relative orientations are available. Let us collect these d_i into a diagonal matrix $D = \text{diag}(d_1, d_2, \cdots, d_n)$.

Then (151) becomes:

$$ZX = (D \otimes I_3)X \qquad (153)$$

where D is degree matrix of the epipolar graph, that contains on the diagonal the degree of the nodes of the graph.

Hence the columns of X are the three eigenvectors of $(D \otimes I)^{-1}Z$ corresponding to the eigenvalue 1, the others being null eigenvalues.

In the presence of noise, the three dominant eigenvectors of $(D \otimes I)^{-1} Z$ are taken.

This represents the solution to one version of problem (147), where the constraint $R_i \in SO(3)$ is relaxed.

The 3×3 blocks of X corresponding to $R_1, R_2, \ldots, R_m$ are therefore projected onto $SO(3)$ only at the end by calculating the nearest rotation matrix.

Listing 17. Rotation Synchronization

```
function R = rotation_synch(Z,A)
%ROTATION_SYNCH Rotation syncronization

    n = size(A,1);

    iD = diag(1./sum(A,2));              % inverse degree matrix
    [Q,~] = eigs( kron(iD,eye(3))*Z,3,'lr'); % top 3 eigenvectors

    Q = real(Q/(Q(1:3, 1:3))); % normalize first rotation to I
    % and guard against spurious complex valuess from roundoff

    R=cell(1,n);
    for i=1:n  % Projection onto SO(3)
        [U,~,V] = svd(Q(3*i-2:3*i,:));
        R{i} = U*diag([1,1,det(U*V')])*V';
    end
end
```

Synchronization of translations

We now deal with the recovery of the translation components of the PPM $\mathbf{t}_i \in \mathbb{R}^3$, on the assumption that the relative translations are known with the module.

The compatibility equation for the translations (145), is the following:

$$\mathbf{t}_{ij} = R_i(\widetilde{\mathbf{C}}_j - \widetilde{\mathbf{C}}_i) \tag{154}$$

which we rewrite as:

$$R_i^{\mathsf{T}}\mathbf{t}_{ij} = \widetilde{\mathbf{C}}_j - \widetilde{\mathbf{C}}_i := \mathbf{u}_{ij}, \tag{155}$$

where $\widetilde{\mathbf{C}}_i$ is the center of the i -th camera (in Cartesian coordinates) and $\mathbf{u}_{ij}$ represents the relative translation expressed in the global reference, the one integral with the first PPM, as we have fixed it.

Let us now consider the matrix X $3 \times m$ obtained from juxtaposition of the centers $\widetilde{\mathbf{C}}_i$: $X = [\widetilde{\mathbf{C}}_1, \widetilde{\mathbf{C}}_2, \ldots \widetilde{\mathbf{C}}_m]$. Then the equation (155) is written:

$$X\mathbf{b}_{ij} = \mathbf{u}_{ij} \tag{156}$$

where the indicator vector

$$\mathbf{b}_{ij} = (0, \ldots, \underset{\uparrow}{-1}_{i}, \ldots, \underset{\uparrow}{1}_{j}, \ldots, 0)^{\mathsf{T}} \tag{157}$$

select columns i and j from X. If we reconsider the graph epipolar G we see that $\mathbf{b}_{ij}$ represents an arc, in fact it is one of the ℓ columns of incidence matrix B, therefore the ℓ equations, one for each arc, which we can write, are expressed in matrix form as:

$$XB = U \tag{158}$$

where U $3 \times \ell$ contains the $\mathbf{u}_{ij}$ as columns. The matrix B is $m \times \ell$, but its rank is at most $m - 1$, if the epipolar graph is connected.

The solution is clearly defined up to a translation of all centers, which is why we can arbitrarily fix $\widetilde{\mathbf{C}}_1 = \mathbf{0}$ (to remain consistent with the solution for two images) and remove it from the unknowns, thus leaving a matrix B_1 $m - 1 \times \ell$ of full rank.

Thanks to the "vec-trick", we can write:

$$(B_1^{\mathsf{T}} \otimes I) \operatorname{vec} X_1 = \operatorname{vec} U. \tag{159}$$

If we consider the noise that inevitably afflicts the measurements, then we will have to look for an approximate solution.

It is proved that the least squares solution of (159) among all those that satisfy the compatibility constraints is the one closest to the measures, in Euclidean norm.

Listing 18. Translation Synchronization

```
function T = translation_synch(U,B)
%TRANSLATION_SYNCH  Translation synchronization

    B(1,:) = []; % remove node 1

    X=kron(B',eye(3)) \ U(:);
    X=[0;0;0;X]; % add node 1
    X=reshape(X,3,[]);

    T = num2cell(X,[1,size(X,2)]);

end
```

Localization from bearings

In the context of image reconstruction, the synchronization of translations is not enough, since the modules of the translations are unknowns.

In other words, of the translation $\mathbf{u}_{ij}$ we only know its versor:

$$\widehat{\mathbf{u}}_{ij} = R_i^{\mathsf{T}} \hat{\mathbf{t}}_{ij} \tag{160}$$

which also takes the name of bearing, and represents the direction under which the camera i sees the camera j, expressed in the global reference, while $\hat{\mathbf{t}}_{ij}$ represents the same direction in the i -th camera reference.

Instead of retrieving the modules, as in the three view example, we proceed directly to recover the COPs (and therefore the modules are implicitly determined).

Let's start with the translation synchronization equation (159) and multiply both sides by the diagonal block matrix $\widehat{S}$ $3\ell \times 3\ell$:

$$\widehat{S} = \mathsf{blkdiag}\left(\left\{[\widehat{\mathbf{u}}_{ij}]_\times\right\}_{(i,j)\in E}\right) \tag{161}$$

getting

$$\widehat{S}(B_1^\mathsf{T} \otimes I)\,\mathrm{vec}\,X_1 = \widehat{S}\,\mathrm{vec}\,U = 0. \tag{162}$$

The right member cancels each $\mathbf{u}_{ij}$ (column of U) for it is multiplied on the left by the corresponding matrix $[\widehat{\mathbf{u}}_{ij}]_\times$, which is equivalent to the cross product of the vector by itself.

Since equation (162) is homogeneous, the solution is defined up to a scale, which potentially includes also a sign, which introduces a reflection of the solution (that is, if X is a solution, $-X$ is a solution as well). The latter can be taken into account a-posteriori by choosing between X and $-X$ the solution that agrees with the direction of the bearings.

In the case of measurements affected by noise $\widehat{S}(B_1^{\mathsf{T}} \otimes I)$ will generally have full rank and a solution is sought that minimizes the residuals:

$$\min_{\|X\|=1} \|\widehat{S}(B_1^{\mathsf{T}} \otimes I)\,\mathrm{vec}\,X_1\|^2 \tag{163}$$

which is obtained from the right singular vector corresponding to minimum singular value, as usual.

Once the camera centers have been calculated, the translational components of the PPM are obtained from the relation:

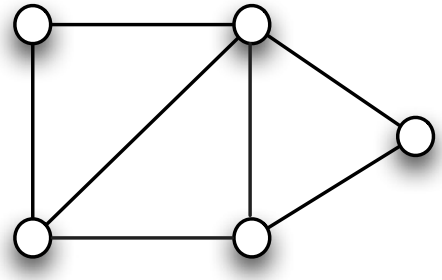
$$\mathbf{t}_i = -R_i\widetilde{\mathbf{C}}_i.$$

This, together with the previous synchronization of rotations, allows one to instantiate the PPMs $P_1 = [I|0], P_2 = [R_2|\mathbf{t}_2], \ldots P_m = [R_m|\mathbf{t}_m]$ and then proceed to the triangulation of points.

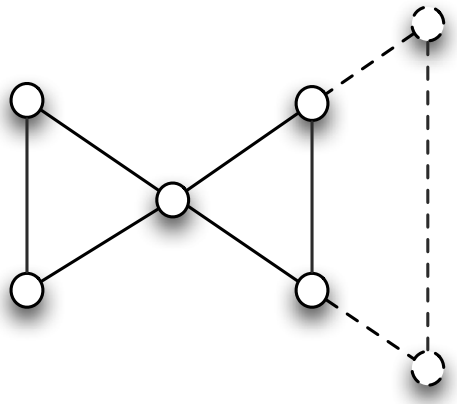
Listing 19. Localization from bearings

```
function C = cop_from_bearings(U,B)
%COP_FROM_BEARINGS  Camera locations from bearings

    B(1,:) = []; % remove node 1

    S=kron(eye(size(B,2)),ones(3,3));
    for i=1:size(U,2)
        S(3*i-2:3*i,3*i-2:3*i) = skew(U(:,i)) ;
    end

    [~,~,V] = svd(S*kron(B',eye(3)));
    X =  reshape(V(:,end),3,[]);
    s = sign(U(:,1)'*X*B(:,1)); % sign
    X = s * [[0;0;0],X]; % add node 1 and fix sign

    C = num2cell(X,[1,size(X,2)]);
end
```

While the synchronization of the translations has solution as soon as the epipolar graph is connected, the conditions under which (162) is solvable are more complex and involve the concept of graph rigidity [2].



Parallel Rigid



Flexible

[2] In the example with three views we also needed edge 1-3 for solve the problem

Bundle Adjustment

Both point-based and camera-based methods are not optimal from a statistical point of view, the former because they produce the final model incrementally, the latter as they are global only in the compensation of the relative orientations, but neglect points, which only played a role in determining the relative orientation between image pairs.

A genuinely global method should optimize a cost that simultaneously encompasses all images and all the tie points.

Bundle (Block) Adjustment: minimize the overall reprojection error of the tie points in the images in which they are visible, with respect to their 3D coordinates and orientation parameters (interior and exterior) of the images.

Geometric interpretation of the problem: each image with its optical rays corresponding to the tie points can be seen as a bundle of rays, and the " compensation " refers to the process of orienting bundles in space until they intersect at best the homologous optical rays, ie corresponding to the same tie point.

Analytically, it is a matter of adjusting both the m cameras and the n tie points so that the sum of the squared distances between the j-th tie point reprojected by the i-th camera $P_i\mathbf{M}^j$ and the measured point $\mathbf{m}_i^j$ is as small as possible, in each image where the point appears:

$$\chi(\widetilde{\mathbf{M}}^j, P_i) = \sum_{i=1}^{m}\sum_{j=1}^{n} \|\widetilde{\eta}(P_i\mathbf{M}^j) - \widetilde{\mathbf{m}}_i^j\|^2 \tag{164}$$

where the function η which performs the so-called "perspective division":

$$\widetilde{\eta} : \mathbb{R}^3 \longrightarrow \mathbb{R}^2 \quad \widetilde{\eta}([x, y, z]^\mathsf{T}) = \left[\frac{x}{z}, \frac{y}{z}\right]^\mathsf{T}. \tag{165}$$

This is called the reprojection error.

The matrix P is represented with its intrinsic and extrinsic parameters:
$[\omega, \varphi, \kappa, t_1, t_2, t_3, \alpha_u, \alpha_v, u_o, v_o]^\mathsf{T}$.

If, on the other hand, we are in a context in which the intrinsic parameters are known and one want to determine only the exterior orientation of the images, the objective function becomes (in normalized image coordinates):

$$\chi(\widetilde{\mathbf{M}}^j, R_i, \mathbf{t}_i) = \sum_{i=1}^{m} \sum_{j=1}^{n} \|\widetilde{\eta}(R_i \widetilde{\mathbf{M}}^j + \mathbf{t}_i) - \widetilde{\mathbf{q}}^j\|^2 \tag{166}$$

The exterior orientation $[R|\mathbf{t}]$ is represented with six parameters $[\omega, \varphi, \kappa, t_1, t_2, t_3]^{\mathsf{T}}$.

Equation (164) (or (166)) is the cost function of a nonlinear least squares problem, for which there are no solutions in closed form, but only iterative techniques (such as Levenberg-Marquardt) that need a starting point close enough to the global minimum.

Jacobian matrix

Two groups of unknowns appear in the BBA: the extrinsic parameters and the coordinates of the 3D point.

The derivatives with respect to each of these two groups of unknowns are the blocks that make up the Jacobian matrix of the LS problem:

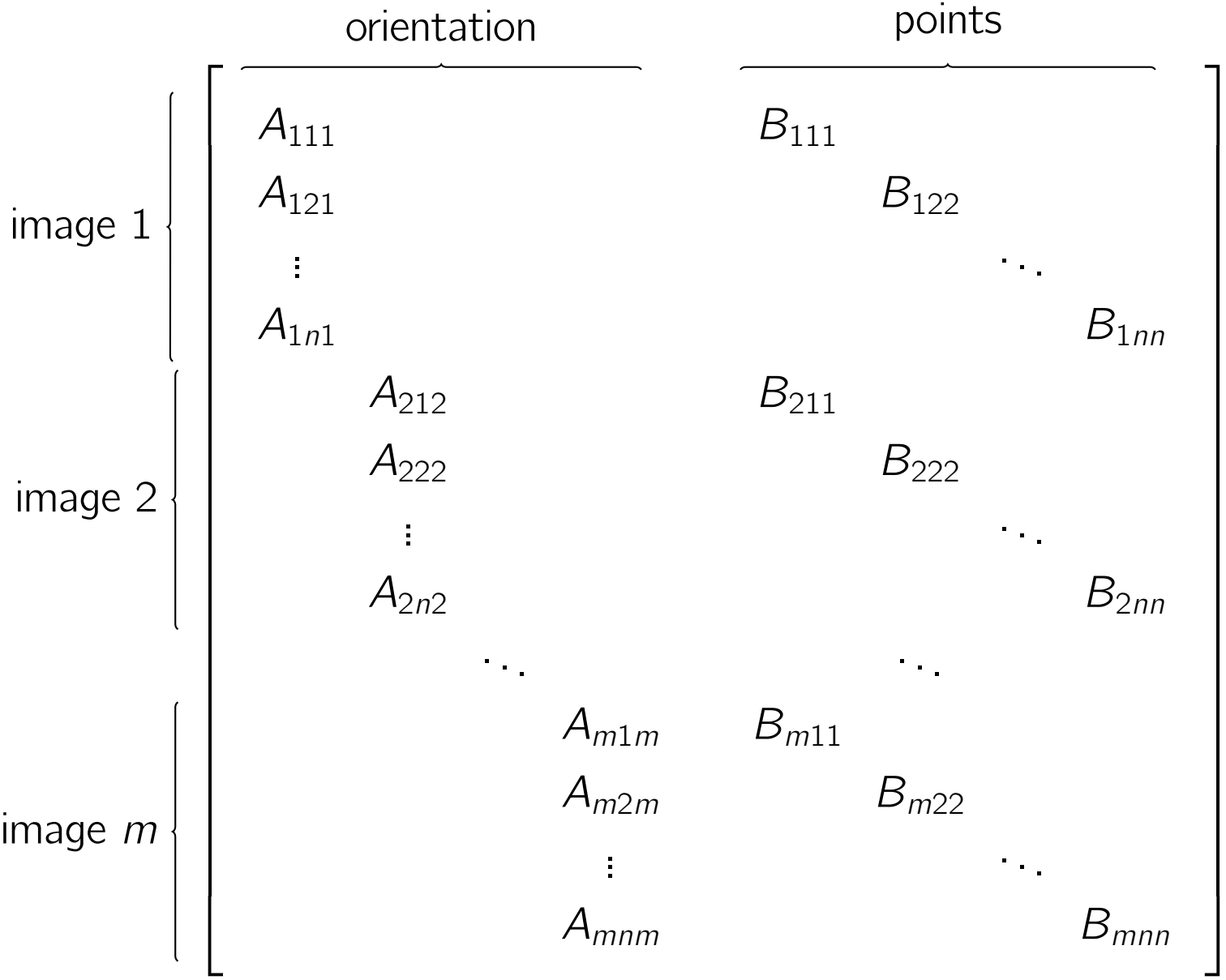
$$A_{ijk} = \frac{\partial \widetilde{\eta}(P_i \mathbf{M}^j)}{\partial \mathbf{g}_k^{\mathsf{T}}} \tag{167}$$

is the matrix of the partial derivatives of the residual of the point j in frame i versus frame orientation k. Similarly:

$$B_{ijk} = \frac{\partial \widetilde{\eta}(P_i \mathbf{M}^j)}{\partial \widetilde{\mathbf{M}}_k^{\mathsf{T}}} \tag{168}$$

is the matrix of the partial derivatives of the residual of the point j in the frame i with respect to the coordinates of the point k.

It is easy to see that $A_{ijk}{=}0 \quad \forall i{\neq}k \quad$ and $B_{ijk}{=}0 \quad \forall j{\neq}k$. Hence the Jacobian matrix has a sparse structure a blocks, called primary structure.


orientation
points
image 1
image 2
image m
A_{111}
A_{121}
A_{1n1}
B_{111}
B_{122}
B_{1nn}
A_{212}
A_{222}
A_{2n2}
B_{211}
B_{222}
B_{2nn}
A_{m1m}
A_{m2m}
A_{mnm}
B_{m11}
B_{m22}
B_{mnn}

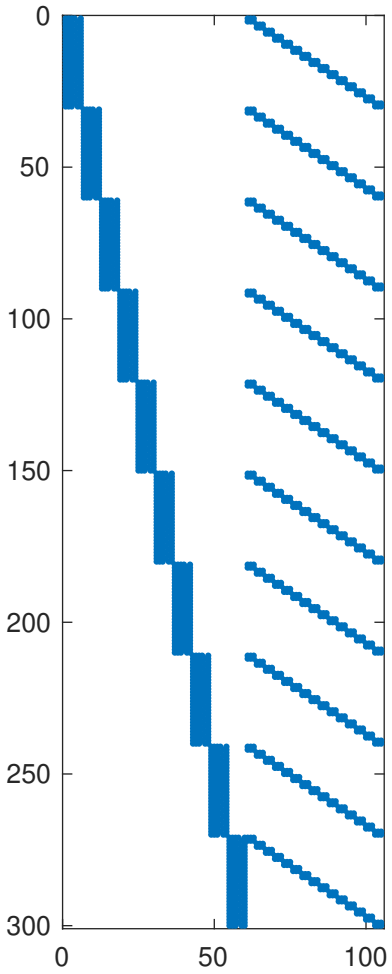
The BBA is nothing more than a simultaneous regression of the PPMs and of the tie points.

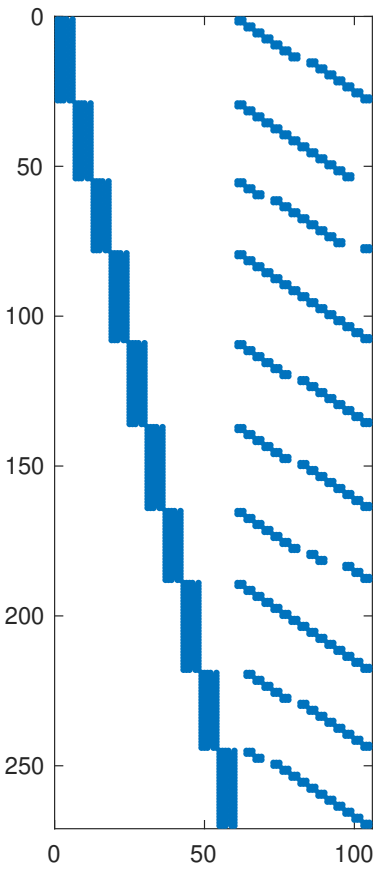
If we consider a single frame and control points instead of tie points, only the blocks A_{111} to A_{1n1} remain, which correspond to the exterior orientation of one frame (n is the number of control points).

If instead we consider the exterior orientation constant, only the blocks B remains, which correspond to the triangulation.

If one also wants to consider the intrinsic parameters among the unknowns, their derivatives would also appear.

It should also be noted that in practice the structure of the Jacobian matrix reflects also the visibility of the points as the two lines corresponding to $\mathbf{m}_i^j$ are present only if the point $\mathbf{M}^j$ is visible in camera P_i. The shape of the matrix that derives from this observation is called secondary structure.





The Jacobian, and consequently the normal equation resulting in the Gauss-Newton method, has a rank degeneracy corresponding to the degrees of freedom of the solution.

In this specific case the solution is defined up to a similarity, which has seven degrees of freedom.

One can solve this problem in different ways:

- remove the degrees of freedom by fixing the coordinates of some points (which become control points) and/or COPs;
- the normal equations are solved with the pseudo-inverse, which in the under-determined case returns the solution of minimum norm, thus implicitly imposing a constraint on the solution.
- the normal equations are solved by adding a damping diagonal term, which remedies the rank degeneracy, as it does the Levenberg-Marquardt method.

The first solution leads to an identical reconstruction, while the the last two to a Euclidean reconstruction.

Reduced system

The primary structure of the Jacobian matrix that we have observed in the previous paragraph can be used to reduce the computational cost of solving the normal equations, through the formulation of a reduced system of equations.

The normal equation that is solved at each Gauss-Newton step to compute $\Delta\mathbf{x}$ is the following:

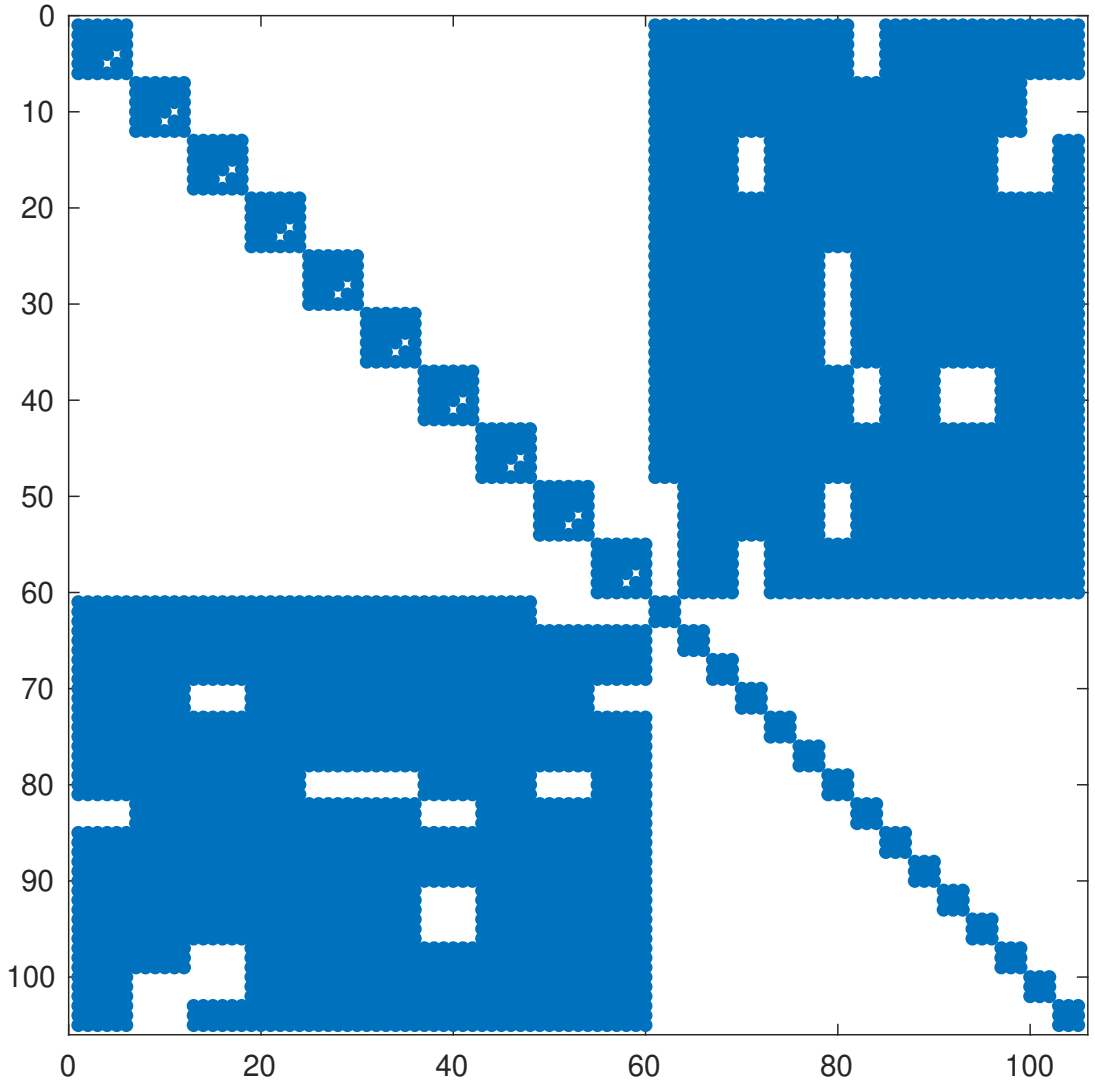
$$\underbrace{J^{\mathsf{T}}J}_{H}\Delta\mathbf{x} = -J^{\mathsf{T}}f(\mathbf{x}) \tag{169}$$

where the components of the vector $f(\mathbf{x})$ are the single terms of the summation in (164), ie the residuals, and $\mathbf{x}$ is the vector of the variables containing $3\times n$ parameters for points $\mathbf{M}^j$ and $6\times m$ parameters for cameras ($11\times m$ if we include the intrinsic parameters or in the projective case).

The size of H is dominated by $n >> m$.

It is therefore a matter of solving a linear system that can be very large.

We can take advantage of the structure of the Hessian matrix.



The primary structure can be exploited to decompose the problem: we partition the Jacobian into two parts, the one relating to the cameras and that relating to points:

$$J = \begin{bmatrix} J_c J_p \end{bmatrix}. \tag{170}$$

Note that both are block diagonals. Then also the matrix H and the normal equation are partitioned as follows:

$$\begin{bmatrix} J_c^\mathsf{T} J_c & J_c^\mathsf{T} J_p \\ J_p^\mathsf{T} J_c & J_p^\mathsf{T} J_p \end{bmatrix} \begin{bmatrix} \Delta \mathbf{x}_c \\ \Delta \mathbf{x}_p \end{bmatrix} = \begin{bmatrix} -J_c^\mathsf{T} f(\mathbf{x}_c \,|\, \mathbf{x}_p) \\ -J_p^\mathsf{T} f(\mathbf{x}_c \,|\, \mathbf{x}_p) \end{bmatrix}. \tag{171}$$

To simplify the notation we rewrite it as:

$$\begin{bmatrix} H_{cc} & H_{cp} \\ H_{pc} & H_{pp} \end{bmatrix} \begin{bmatrix} \Delta \mathbf{x}_c \\ \Delta \mathbf{x}_p \end{bmatrix} = \begin{bmatrix} b_c \\ b_p \end{bmatrix}. \tag{172}$$

Note that H_{cc} and H_{pp} are block diagonals. We multiply to the left both members

$$\begin{bmatrix} I & -H_{cp} H_{pp}^{-1} \\ 0 & I \end{bmatrix} \tag{173}$$

obtaining the effect of making the coefficients matrix lower block triangular:

$$\begin{bmatrix} H_{cc} - H_{cp}H_{pp}^{-1}H_{pc} & 0 \\ H_{pc} & H_{pp} \end{bmatrix} \begin{bmatrix} \Delta\mathbf{x}_c \\ \Delta\mathbf{x}_p \end{bmatrix} = \begin{bmatrix} b_c - H_{cp}H_{pp}^{-1}b_p \\ b_p \end{bmatrix} \tag{174}$$

and therefore solve for one of the two blocks of unknowns:

$$(H_{cc} - H_{cp}H_{pp}^{-1}H_{pc})\Delta\mathbf{x}_c = b_c - H_{cp}H_{pp}^{-1}b_p. \tag{175}$$

This system is smaller than the original and the inversion of H_{pp} is simple thanks to its diagonal block structure. The other block of unknowns is obtained with:

$$\Delta\mathbf{x}_p = H_{pp}^{-1}(b_p - H_{pc}\Delta\mathbf{x}_c). \tag{176}$$

The secondary structure of J is reflected in the reduced system matrix: $H_{cc} - H_{cp}H_{pp}^{-1}H_{pc}$. If every camera sees only a fraction of the points, it will be sparse; in particular if the images are sequenced the reduced system matrix has a band structure, and this can be usefully exploited to reduce the computational cost further.

What was left out?

- radial distortion
- optimal non-linear regression
- robust methods (dealing with outiers)
- feature extraction and matching (provide conjugate points)
- stereo matching (dense conjugate points)
- autocalibration (unknown K)

References

Arun, K.S., Huang, T.S., Blostein, S.D., 1987. Least-squares fitting of two 3-D point sets. IEEE Transactions on Pattern Analysis and Machine Intelligence 9, 698–700.

Commandeur, J.J.F., 1991. Matching configurations. DSWO Press, Leiden.

Crosilla, F., Beinat, A., 2002. Use of generalised procrustes analysis for the photogrammetric block adjustment by independent models. ISPRS Journal of Photogrammetry & Remote Sensing 56, 195–209.

Faugeras, O., Maybank, S., 1990. Motion from point matches: multiplicity of solutions. International Journal of Computer Vision 4, 225–246.

Fiore, P.D., 2001. Efficient linear solution of exterior orientation. IEEE Transactions on Pattern Analysis and Machine Intelligence 23, 140–148.

Gherardi, R., Farenzena, M., Fusiello, A., 2010. Improving the efficiency of hierarchical structure-and-motion, in: Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition (CVPR 2010).

Hartley, R.I., 1995. In defence of the 8-point algorithm, in: Proceedings of the

International Conference on Computer Vision, IEEE Computer Society, Washington, DC, USA. pp. 1064–1071.

Hartley, R.I., Sturm, P., 1997. Triangulation. Computer Vision and Image Understanding 68, 146–157.

Hartley, R.I., Trumpf, J., Dai, Y., Li, H., 2013. Rotation averaging. International Journal of Computer Vision .

Huang, T., Faugeras, O., 1989. Some properties of the E matrix in two-view motion estimation. IEEE Transactions on Pattern Analysis and Machine Intelligence 11, 1310–1312.

Li, H., Hartley, R., 2006. Five-point motion estimation made easy, in: Proceedings of the International Conference on Pattern Recognition, IEEE Computer Society, Washington, DC, USA. pp. 630–633.

Luong, Q.T., Viéville, T., 1996. Canonical representations for the geometries of multiple projective views. Computer Vision and Image Understanding 64, 193–229.

Nister, D., 2003. An efficient solution to the five-point relative pose problem, in: Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition, IEEE Computer Society, Los Alamitos, CA, USA. p. 195.

Singer, A., 2011. Angular synchronization by eigenvectors and semidefinite programming. Applied and Computational Harmonic Analysis 30, 20 – 36.

Sturm, P.F., Maybank, S.J., 1999. On plane-based camera calibration: A general algorithm, singularities, applications., in: CVPR, IEEE Computer Society. pp. 1432–1437.

Zhang, Z., 2000. A flexible new technique for camera calibration. IEEE Transactions on Pattern Analysis and Machine Intelligence 22, 1330–1334.