Elements of Photogrammetric Computer Vision

Andrea Fusiello
University of Udine, IT

Contents

1 Introduction

Goal: recover the geometry of the imaged objects (structure) and the motion of the camera from images.

Photogrammetry was born in 1850.

Analytical Photogrammetry was born in 1950.

Structure from motion in Computer Vision became an active field in the late '70s.

Completely automatic pipeline (Bundler): first decade of this century.

Still an active field.

2 Background

The pin-hole (or *stenopeic*) camera is described by its centre **O** (also known as centre of projection) and the image plane.

The distance of the image plane from **O** is the focal length f (or principal distance).

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

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

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

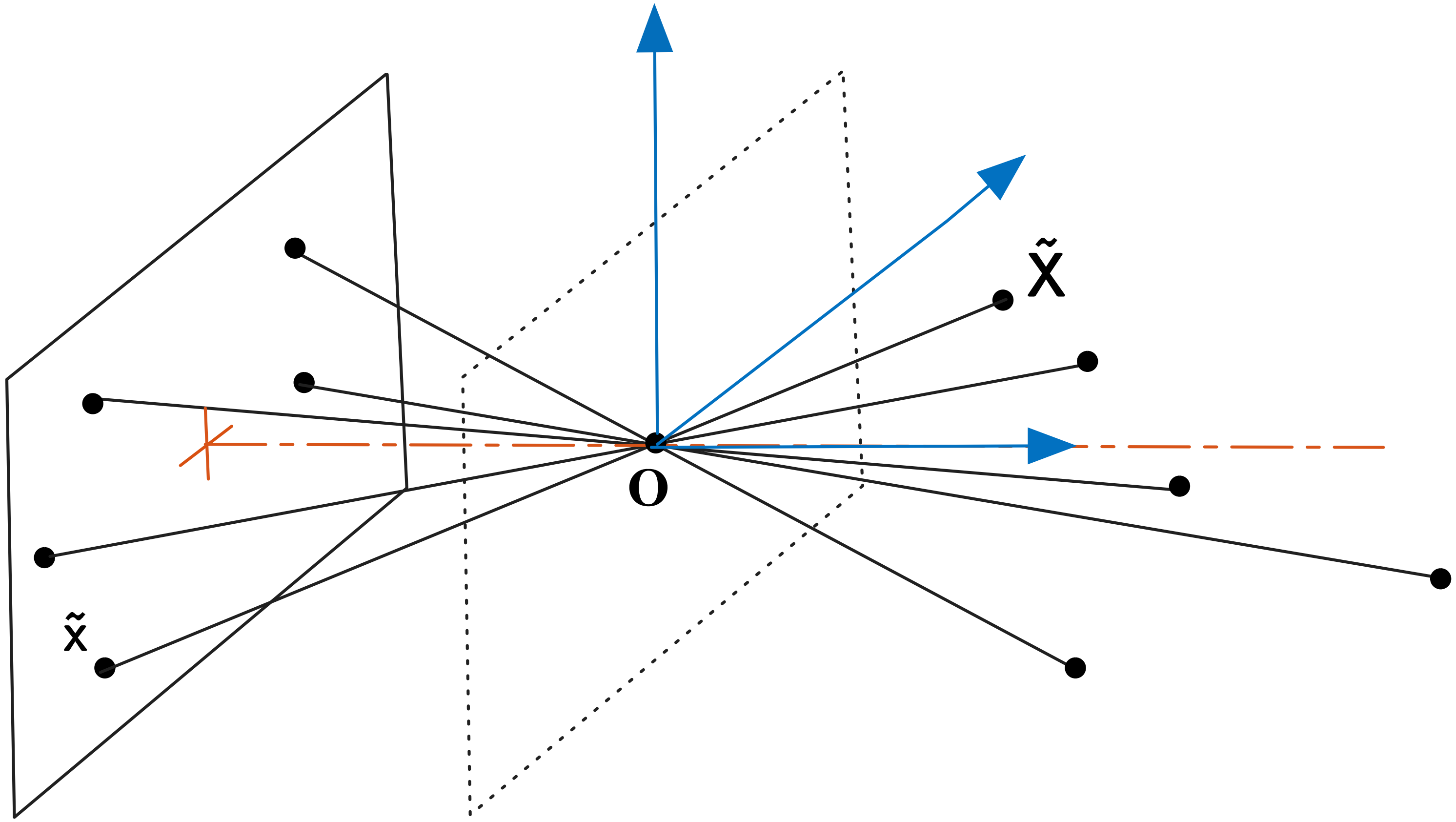


Fig. 1: The pinhole camera

A 3-D point is projected onto the image plane with the line containing the point and the centre of projection.

Let the centre of projection $\mathbf{O}$ be the origin of a Cartesian coordinate system wherein the Z-axis is the principal axis.

By similar triangles it is readily seen that the 3-D point $(X, Y, Z)^T$ is mapped to the point $(fX/Z, fY/Z)^T$ on the image plane.

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

$$\begin{pmatrix} fX \\ fY \\ Z \end{pmatrix} = \begin{bmatrix} f & 0 & 0 & 0 \\ 0 & f & 0 & 0 \\ 0 & 0 & 1 & 0 \end{bmatrix} \begin{pmatrix} X \\ Y \\ Z \\ 1 \end{pmatrix} \tag{1}$$

The matrix describing the mapping is called the *camera projection matrix* P. Equation (1) can be written simply as:

$$Z\tilde{\mathbf{x}} = P\tilde{\mathbf{X}} \tag{2}$$

where $\tilde{\mathbf{X}} = (X, Y, Z, 1)^T$ are the homogeneous coordinates of the 3-D point and $\tilde{\mathbf{x}} = (u, v, 1)^T$ are the homogeneous coordinates of the image point.

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

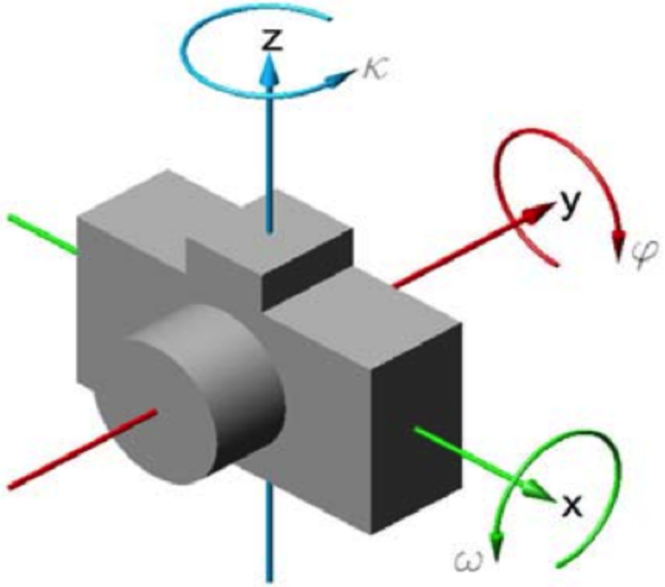
Changing coordinates in space is equivalent to multiplying the matrix P to the right by a 4 × 4 matrix:

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

G is composed by a rotation matrix R and a translation vector $\mathbf{t}$.

It describes the position and attitude of the camera with respect to an external (object) coordinate system.

It depends on six parameters, called *exterior* parameters.



Changing coordinates in the image plane is equivalent to multiplying the matrix P to the left by a 3×3 matrix (representing an affine transform):

$$V = \begin{bmatrix} \frac{1}{\Delta_u} & -\frac{1}{\Delta_u \tan\theta} & u_0 \\ 0 & \frac{1}{\Delta_v \sin\theta} & v_0 \\ 0 & 0 & 1 \end{bmatrix} \tag{4}$$

where

- u_o, v_o are the coordinates (in pixel) of the principal point (or image centre),
- θ is the angle between the coordinate axes of the pixels grid (should be 90°),
- and (Δ_u, Δ_v) are width and height respectively of the pixel footprint on the camera photosensor (or effective dimensions of the pixel).

This transformation account for the fact that the pixel indices have the origin in the upper-left corner of the image, that the photosensor grid can be non-rectangular (or *skewed*), and that pixels have a given physical dimension.

It is customary to include also the focal length f (which act as a uniform scaling) in this transformation, to obtain:

$$K = \begin{bmatrix} \alpha_u & \gamma\alpha_u & u_0 \\ 0 & r\alpha_u & v_0 \\ 0 & 0 & 1 \end{bmatrix} \tag{5}$$

K is the camera calibration matrix; It depends on the so-called interior parameters:

- focal length α_u (in Δ_u units),
- principal point (or image centre) coordinates u_o, v_o (in Δ_u units),
- aspect ratio $r = \Delta_u/(\Delta_v \sin\theta)$ (usually ≈ 1),
- skew $\gamma = -1/\tan\theta$ (usually ≈ 0).

K encodes the (affine) transformation from the so-called normalized image coordinates to image coordinates:

- image coordinates are measured in the digital image, in pixels.
- normalized image coordinates (NIC) would be measured on an ideal image plane at unit distance from **O**. Their unit is the same as the 3D points (e.g., meters).

Normalized image coordinates $\tilde{\mathbf{p}}$ (not accessible) are obtained from the image coordinates (accesible) via the kowledge of K, with

$$\tilde{\mathbf{p}} = K^{-1}\tilde{\mathbf{x}}.$$

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

$$P = K[I|\mathbf{0}]G = K[R|\mathbf{t}] \tag{6}$$

and the projection equation writes:

$$\zeta \tilde{\mathbf{x}} = P\tilde{\mathbf{X}} \tag{7}$$

where ζ is a suitable scale factor, that turns out to be the distance of $\tilde{\mathbf{X}}$ from the focal plane of the camera.

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

$$P\tilde{\mathbf{O}} = P\begin{pmatrix}\mathbf{O}\\1\end{pmatrix} = \mathbf{0} \tag{8}$$

where $\mathbf{O}$ is a 3-D vector containing the Cartesian (non-homogeneous) coordinates of the centre of projection. After solving for $\mathbf{O}$ we obtain:

$$\mathbf{O} = -P_{1:3}^{-1}P_4 \tag{9}$$

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

Optical ray. The *optical ray* of an image point $\tilde{\mathbf{x}}$ is the locus of points in space that projects onto $\tilde{\mathbf{x}}$. It can be described as a parametric line passing through the camera centre $\mathbf{O}$ and a special point (at infinity) that projects onto $\tilde{\mathbf{x}}$:

$$\tilde{\mathbf{X}} = \begin{pmatrix} -P_{1:3}^{-1} P_4 \\ 1 \end{pmatrix} + \zeta \begin{pmatrix} P_{1:3}^{-1} \tilde{\mathbf{x}} \\ 0 \end{pmatrix}, \quad \zeta \in \mathbb{R}. \tag{10}$$

Please note that in order to be able to trace the optical ray of an image point, the interior parameters must be known.

2.1 Collinearity equations

In Photogrammetry the perspective projection is described by the so-called *collinearity equations*, which, in our notation with $\gamma = 0, r = 1$, write:

$$\begin{cases} u = u_0 + \alpha_u \dfrac{\mathbf{r}_1^T(\mathbf{X} - \mathbf{O})}{\mathbf{r}_3^T(\mathbf{X} - \mathbf{O})} \\ v = v_0 + \alpha_v \dfrac{\mathbf{r}_2^T(\mathbf{X} - \mathbf{O})}{\mathbf{r}_3^T(\mathbf{X} - \mathbf{O})} \end{cases}$$

where $\mathbf{r}_i^T$ are the rows of R.

The perspective projection equation (7) is the matrix equivalent of the collinearity equations. To see this, let substitute $P = K[R|\mathbf{t}]$ in (7), obtaining:

$$\tilde{\mathbf{x}} = \zeta^{-1} K(R\mathbf{X} + t)$$

Since (from (9)) $t = -R\mathbf{O}$ we have

$$\tilde{\mathbf{x}} = \zeta^{-1} K(R\mathbf{X} - R\mathbf{O}) = \zeta^{-1} KR(\mathbf{X} - \mathbf{O})$$

The third (homogeneous) coordinate of the lefthand side is 1, the third coordinate of the righthand side is $\zeta^{-1}(\mathbf{r}_3^T(\mathbf{X} - \mathbf{O}))$, hence $\zeta = \mathbf{r}_3^T(\mathbf{X} - \mathbf{O})$.

2.2 Camera resection by DLT

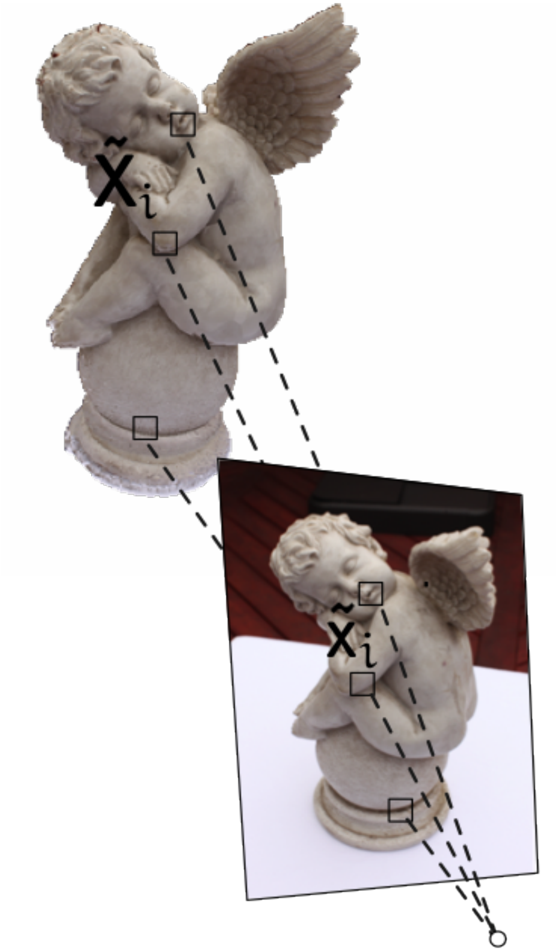
A number of point correspondences $\tilde{\mathbf{x}}_i \leftrightarrow \tilde{\mathbf{X}}_i$ is given, and we are required to find a camera matrix P such that

$$\tilde{\mathbf{x}}_i \simeq P\tilde{\mathbf{X}}_i \quad \text{for all } i. \tag{11}$$

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

$$\tilde{\mathbf{x}}_i \times P\tilde{\mathbf{X}}_i = \mathbf{0}. \tag{12}$$

This form will enable a simple a simple linear solution for P to be derived. Using the properties of the Kronecker product ($\otimes$) and the vec operator (Magnus and Neudecker, 1999), we derive:



$$\tilde{\mathbf{x}}_i \times P\tilde{\mathbf{X}}_i = \mathbf{0} \iff [\tilde{\mathbf{x}}_i]_\times P\tilde{\mathbf{X}}_i = \mathbf{0} \iff \operatorname{vec}([\tilde{\mathbf{x}}_i]_\times P\tilde{\mathbf{X}}_i) = \mathbf{0} \iff (\tilde{\mathbf{X}}_i^T \otimes [\tilde{\mathbf{x}}_i]_\times) \operatorname{vec} P = \mathbf{0}$$

where we used the fact that the cross product of two vectors can be written as a product of a skew-symmetric matrix and one vector: $\mathbf{a} \times \mathbf{b} = [\mathbf{a}]_\times \mathbf{b}$.

These are three equations in 12 unknown, only two of them are linearly independent.

Indeed, the rank of $(\tilde{\mathbf{X}}_i^T \otimes [\tilde{\mathbf{x}}_i]_\times)$ is two because it is the Kronecker product of a rank-1 matrix by a a rank-2 matrix.

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

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

If the data are not exact (noise is generally present) the rank of A will be 12 and a least-squares solution is sought, which can be obtained as the singular vector corresponding to the smallest singular value of A.

This algorithm is known as the Direct Linear Transform (DLT) algorithm (Hartley and Zisserman, 2003; Kraus, 2007).

3 Pairwise processing

Pairwise processing indicates generically all those techniques that are designed to estimate the three-dimensional coordinates of points on an object (a.k.a. stereo-model) employing measurements made in two photographic images.

If the two camera matrices are known the process reduces to intersection.

Thus assuming known interior parameters, the core of the problem is to recover the exterior parameters of the two cameras.

In this section we will study different *orientation*[1] problems:

- Relative orientation
- Absolute orientation
- Exterior orientation

All of them consume point correspondences (of different nature) and produces a 6 d.o.f rigid transformation that represents position and angular attitude.

[1] This terminology comes from Photogrammetry (and from German), where "orientation" means angular attitude and position (Kraus, 2007).

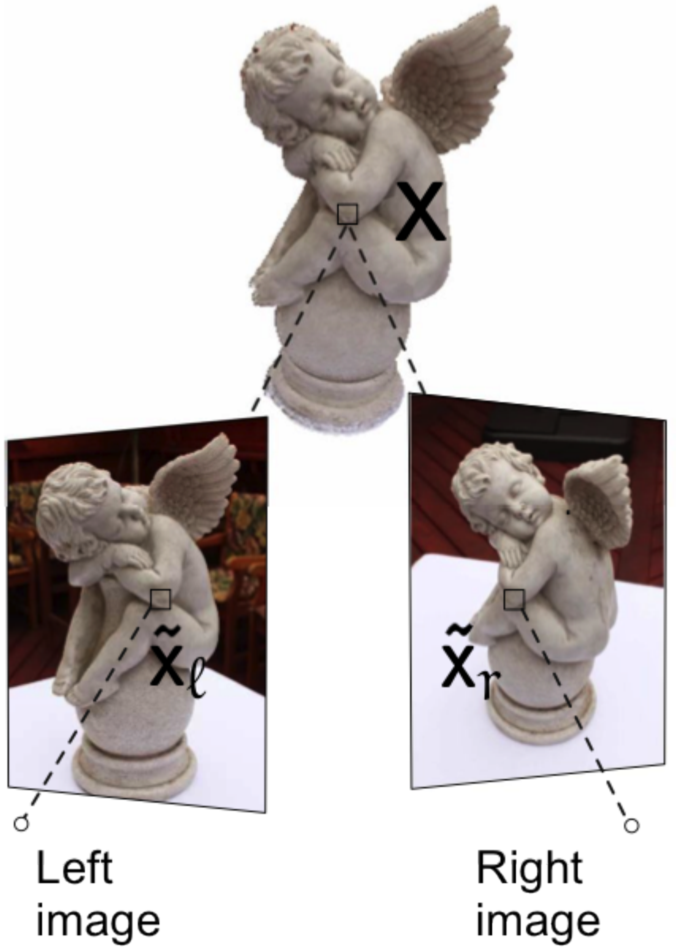
3.1 Intersection (or triangulation)

Given the camera matrices P_ℓ and P_r, let $\tilde{\mathbf{x}}_\ell$ and $\tilde{\mathbf{x}}_r$ be two conjugate points, i.e., they are projections of the same 3-D object point $\tilde{\mathbf{X}}$ on the left and right images respectively.

The goal of intersection is to recover the coordinates of $\tilde{\mathbf{X}}$.

Let us consider $\tilde{\mathbf{x}}_\ell$, the projection of the 3D point M according to the perspective projection matrix P_ℓ. The projection equation (7) can be rewritten using the cross product as

$$\tilde{\mathbf{x}}_\ell \times P_\ell \tilde{\mathbf{X}} = \mathbf{0}. \tag{13}$$



with the effect of eliminating the factor ζ.

Hence, one point in one camera gives three homogeneous equations, two of which are independent.

Let us now consider its conjugate point $\tilde{\mathbf{x}}_r$, and let P_r be the second perspective projection matrix. Likewise we can write:

$$\tilde{\mathbf{x}}_r \times P_r \tilde{\mathbf{X}} = \mathbf{0}. \tag{14}$$

Being both projection of the same 3D point $\tilde{\mathbf{X}}$, the equations provided by $\tilde{\mathbf{x}}_\ell$ and $\tilde{\mathbf{x}}_r$ can be stacked, thereby obtaining a homogeneous linear system of six equations in four unknown (including the last component of $\tilde{\mathbf{X}}$):

$$\begin{bmatrix} [\tilde{\mathbf{x}}_\ell]_\times P_\ell \\ [\tilde{\mathbf{x}}_r]_\times P_r \end{bmatrix} \tilde{\mathbf{X}} = \mathbf{0}. \tag{15}$$

The solution is the null-space of the 6×4 coefficient matrix, which must then have rank three, otherwise only the trivial solution $\tilde{\mathbf{X}} = \mathbf{0}$ would be possible.

In the presence of noise this rank condition cannot be fulfilled exactly, so a least squares solution is sought, typically via Singular Value Decomposition (SVD).

This method generalizes to the case of $m > 2$ cameras: each one gives two equations and one ends up with $2m$ equations in four unknowns.

This topic addressed in more details in (Beardsley et al., 1997; Hartley and Sturm, 1997; Hartley and Zisserman, 2003)).

3.2 Relative orientation and the Essential matrix

Both in CV and Photogrammetry, a pivotal concept in pairwise processing is those of *relative orientation*, i.e., the rigid transformation that represent position and angular attitude of one camera with respect to the other.

The computer vision approach to the problem of relative orientation leads to an encoding of the baseline (translation) and attitude (rotation) in a single 3×3 matrix called the *Essential* matrix.

The essential matrix is defined by $E = [\mathbf{t}]_\times R$, where $[\mathbf{t}]_\times$ is the skew-symmetric matrix that satisfies $[\mathbf{t}]_\times \mathbf{v} = \mathbf{t} \times \mathbf{v}$ for any vector $\mathbf{v}$, with $\mathbf{t}$ being the baseline and R a rotation matrix encoding the attitude.

3.2.1 Epipolar geometry

Any unoccluded object 3-D object point $\tilde{\mathbf{X}} = (X, Y, Z, 1)^T$ is projected to the left and right image as $\tilde{\mathbf{x}}_\ell = (u_\ell, v_\ell, 1)^T$ and $\tilde{\mathbf{x}}_r = (u_r, v_r, 1)^T$, respectively.

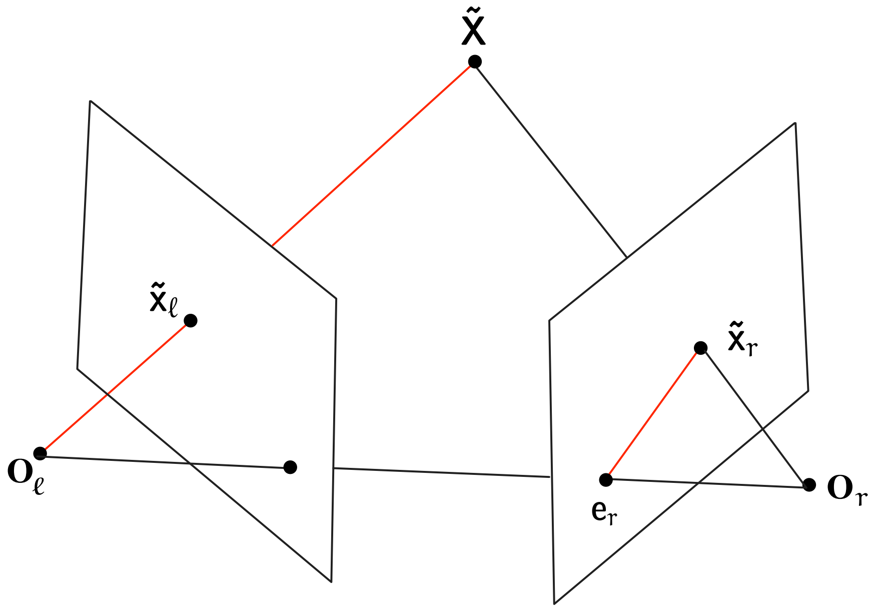
Image points $\tilde{\mathbf{x}}_\ell$ and $\tilde{\mathbf{x}}_r$ are called *corresponding* or *conjugate* points.

We will refer to the camera projection matrix of the left image as P_ℓ and of the right image as P_r.

The 3-D point $\tilde{\mathbf{X}}$ is then imaged as (16) in the left image, and (17) in the right image:

$$\zeta_\ell \tilde{\mathbf{x}}_\ell = P_\ell \tilde{\mathbf{X}} \tag{16}$$

$$\zeta_r \tilde{\mathbf{x}}_r = P_r \tilde{\mathbf{X}}. \tag{17}$$



The relationship between image points $\tilde{\mathbf{x}}_\ell$ and $\tilde{\mathbf{x}}_r$ is given by the *epipolar geometry*.

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

The equation of the epipolar line can be derived from the equation describing the optical ray, for the epipolar line of $\tilde{\mathbf{x}}_\ell$ geometrically represents the projection (Eq. (7)) of the optical ray of $\tilde{\mathbf{x}}_\ell$ (Eq. (10)) onto the right image plane:

$$\zeta_r \tilde{\mathbf{x}}_r = P_r \tilde{\mathbf{X}} = \underbrace{P_r \begin{pmatrix} -P_{\ell_{1:3}}^{-1} P_{\ell_4} \\ 1 \end{pmatrix}}_{\mathbf{e}_r} + \zeta_\ell P_r \begin{pmatrix} P_{\ell_{1:3}}^{-1} \tilde{\mathbf{x}}_\ell \\ 0 \end{pmatrix} \tag{18}$$

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

$$\zeta_r \tilde{\mathbf{x}}_r = \mathbf{e}_r + \zeta_\ell \underbrace{P_{r_{1:3}} P_{\ell_{1:3}}^{-1} \tilde{\mathbf{x}}_\ell}_{\tilde{\mathbf{x}}'_\ell} \tag{19}$$

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

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

3.2.2 The Essential matrix E

Let us now assume that the interior parameters are known, as is customary in Photogrammetry, hence we can assume that points are in NIC.

Using NIC the left and right camera projection matrices write:

$$P_\ell = [I|0] \quad \text{and} \quad P_r = [R|\mathbf{t}]. \tag{20}$$

where the the object reference frame is fixed onto the left camera.

If we substitute these two particular instances of the camera projection matrices in Equation (18), we get

$$\zeta_r \tilde{\mathbf{p}}_r = \mathbf{t} + \zeta_\ell R \tilde{\mathbf{p}}_\ell. \tag{21}$$

In other words, the point $\tilde{\mathbf{p}}_r$ lies on the line through the points $\mathbf{t}$ and $R\tilde{\mathbf{p}}_\ell$. In the projective plane this can be written as follows:

$$\tilde{\mathbf{p}}_r^T (\mathbf{t} \times R \tilde{\mathbf{p}}_\ell) = 0, \tag{22}$$

as the homogeneous line through two points is expressed as their cross product, and a dot product of a point and a line is zero if the point lies on the line.

By introducing the matrix equivalent of the cross product, Equation (22) can be written as

$$\tilde{\mathbf{p}}_r^T [\mathbf{t}]_\times R \tilde{\mathbf{p}}_\ell = 0, \tag{23}$$

In summary, the relationship between the corresponding image points $\tilde{\mathbf{p}}_\ell$ and $\tilde{\mathbf{p}}_r$ in NIC is the bilinear form:

$$\tilde{\mathbf{p}}_r^T E \tilde{\mathbf{p}}_\ell = 0. \tag{24}$$

where we introduced the *essential matrix* E:

$$E = [\mathbf{t}]_\times R. \tag{25}$$

E encodes only information on the rigid displacement between cameras. It has five degrees of freedom: a 3-D rotation and a 3-D translation direction.

E is characterized by the following theorem (Huang and Faugeras, 1989):

Theorem 1 *A real* 3×3 *matrix* E *can be factorized as product of a nonzero skew-symmetric matrix and a rotation matrix if and only if* E *has two identical singular values and a zero singular value.*

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

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

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

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

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

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

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

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

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

Q.E.D.

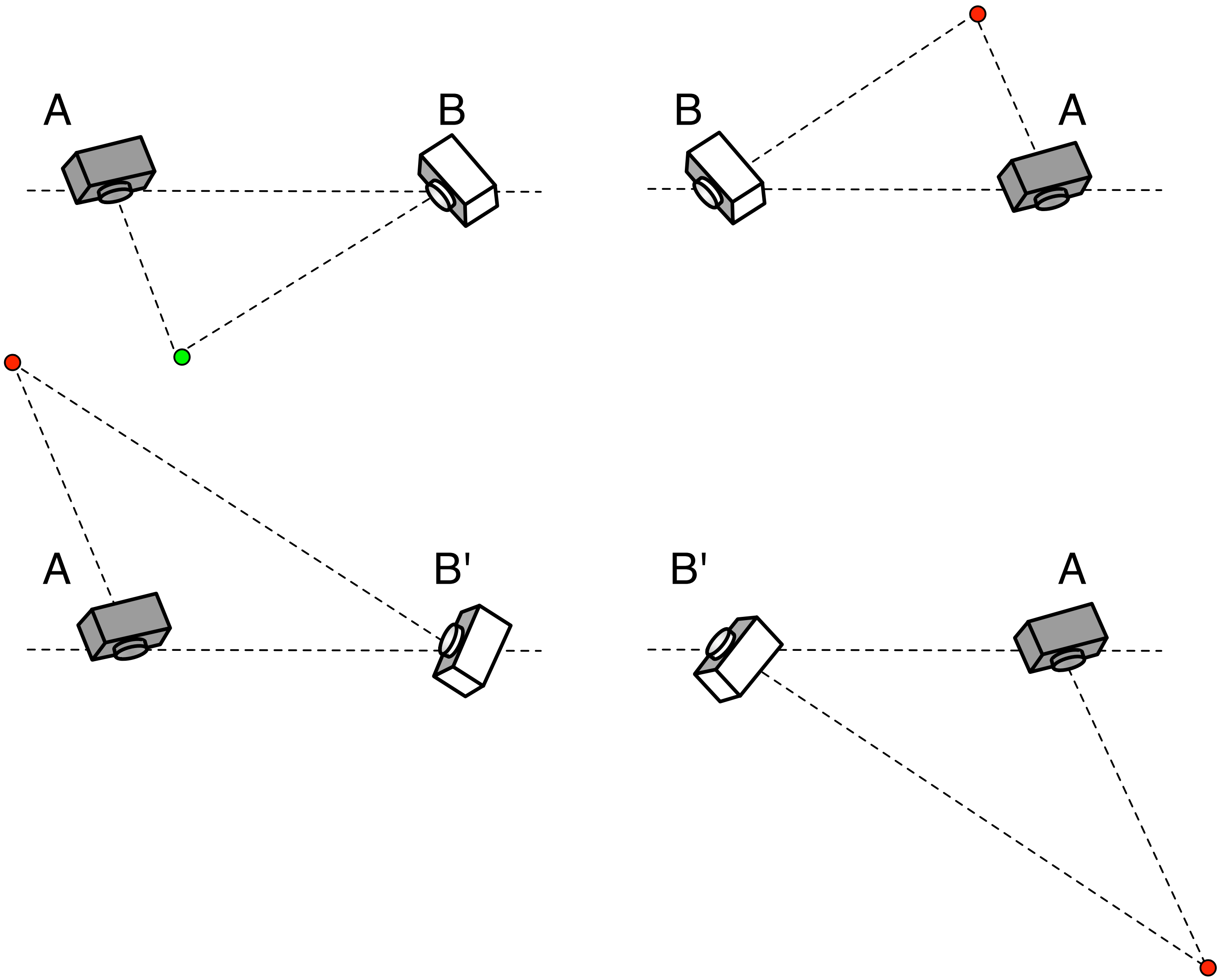
The theorem has a constructive proof that describes how E can be factorized into rotation and translation (skew-symmetric) using its SVD.

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

$$S = U(\pm S')U^T \tag{26}$$

$$R = \det(UV^T)UR'V^T \text{ or } R = \det(UV^T)UR'^TV^T, \tag{27}$$

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


A
B
B
A
A
B'
B'
A

3.2.3 The eight-point algorithm

If a number of point correspondences $\tilde{\mathbf{p}}_\ell^i \leftrightarrow \tilde{\mathbf{p}}_r^i$ is given (in NIC), we can recover the unknown matrix E from Equation (24).

Thanks to the properties of the Kronecker product we can write:

$$\tilde{\mathbf{p}}_r^T E \tilde{\mathbf{p}}_\ell = 0 \iff \operatorname{vec}(\tilde{\mathbf{p}}_r^T E \tilde{\mathbf{p}}_\ell) = 0 \iff (\tilde{\mathbf{p}}_\ell^T \otimes \tilde{\mathbf{p}}_r^T) \operatorname{vec}(E) = 0.$$

Each point correspondence gives rise to one linear equation in the unknown entries of E. From a set of n point correspondences, we obtain a $n \times 9$ coefficient matrix A by stacking up one equation for each correspondence. The least-squares solution for $\operatorname{vec}(E)$ is the singular vector corresponding to the smallest singular value of A.

This simple algorithm provides good results in many situations and can be used to initialize a variety of more accurate, iterative algorithms. Details of these can be found in (Hartley and Zisserman, 2003).

The same algorithm can be used to compute the *fundamental* matrix if image points are expressed in image coordinates.

3.2.4 Closure

If a sufficient number of point correspondences $\tilde{\mathbf{p}}_\ell^i \leftrightarrow \tilde{\mathbf{p}}_r^i$ in NIC is given, the construction of a 3D model from two images proceeds as follows:

- compute E with the eight-points algorithm (Sec. 3.2.3);
- factorize E in $[\mathbf{t}]_\times$ and R using Theorem 1;
- use the rotation R and translation $\mathbf{t}$ to instantiate a camera pair as in Eq. (20);
- compute 3D points coordinates by intersection.

The resulting model has an overall scale ambiguity deriving from the fact that E is defined up to a scale factor: $\mathbf{t}$ can be scaled arbitrarily in Equation (25) and one would get the same essential matrix. Therefore translation can be recovered from E only up to an unknown scale factor, which is inherited by the model.

This is also known as *depth-speed ambiguity* (in a context where points are moving and camera is stationary).

The stereo-model produced by the above procedure is represented in a local, arbitrary reference frame (also called model coordinates)

While in CV this is fine, in Photogrammetry it is mandatory that the model is expressed in object coordinates (frequently a control or global system).

For this reason, the three methods presented below (Kraus, 2007) assume the knowledge of a certain number ground control points (GCP) in the object coordinte system:

- two-step combined orientation (relative + absolute)
- separate exterior orientation
- combined single stage orientation (bundle)

3.3 Two-step combined orientation

This procedure works in two steps:

- Solve relative orientation and compute a stereo-model;
- Align the stereo-model to GCPs via a 3D similarity.

In Photogrammetry specific methods have been conceived to solve relative orientation, but they can be seen as functionally equivalent to the Essential approach described above.

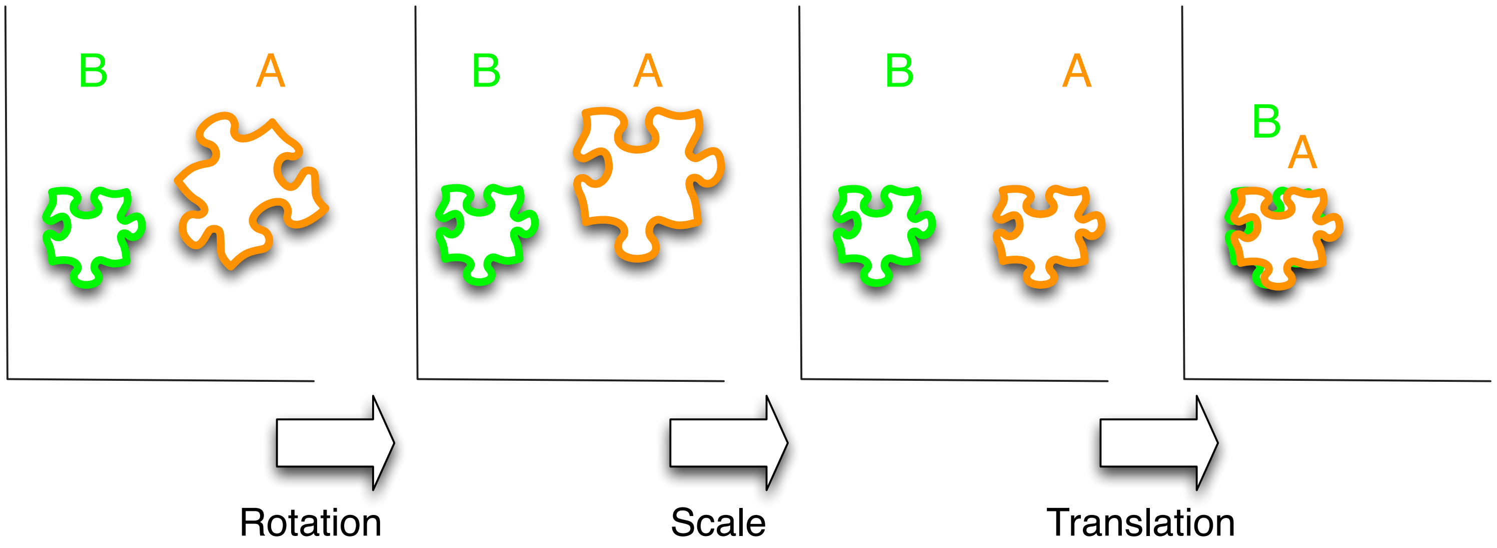
We shall therefore concentrate on the second step, dubbed absolute orientation.

3.3.1 Absolute orientation

Given two sets of 3-D points $\{\mathbf{B}^i\}$ and $\{\mathbf{A}^i\}$, $i = 1 \ldots p$ related by

$$\mathbf{B}^i = \lambda R \mathbf{A}^i + \mathbf{t} \quad \text{for all } i = 1 \ldots p \tag{28}$$

we are required to estimate the unknown rotation R, translation $\mathbf{t}$ and the scale λ from point correspondences.



Assuming homogeneous and isotropic noise, the optimal (ML) estimate can be obtained via (Extended) Orthogonal Procrustes Analysis (next section).

The terms *Procrustes Analysis* (e.g. (Gower and Dijksterhuis, 2004)) is referred to a set of least squares mathematical models used to compute transformations among corresponding points belonging to a generic k-dimensional space, in order to achieve their maximum agreement.

In particular, the Extended Orthogonal Procrustes Analysis (EOPA) model allows to recover the least squares similarity transformation between two point sets.

Let us consider two matrices A and B containing the coordinates of p points of $\mathbb{R}^k$ by rows. EOPA allows to directly estimate the unknown rotation matrix R, a translation vector $\mathbf{t}$ and a global scale factor λ for which the residual:

$$\left\| B - \lambda AR - \mathbf{1}\mathbf{t}^T \right\|_F^2 \tag{29}$$

is minimum, under the orthogonality condition: $R^T R = RR^T = I$.

The minimization proceeds by defining a Lagrangean function and setting the derivatives to zero (details can be found in (Schnemann and Carroll, 1970)).

The rotation is given by

$$R = U\mathrm{diag}\left(1,1,\det(UV^T)\right)V^T \tag{30}$$

where U and V are determined from the SVD decomposition:

$$A^T\left(I - \mathbf{1}\,\mathbf{1}^T/p\right)B = UDV^T \tag{31}$$

The $\det(UV^T)$ normalization guarantees that R is not only orthogonal but has positive determinant (Wahba, 1965).

Then the scale factor can be determined with:

$$\lambda = \frac{\mathrm{tr}\left(R^T A^T\left(I - \mathbf{1}\,\mathbf{1}^T/p\right)B\right)}{\mathrm{tr}\left(A^T\left(I - \mathbf{1}\,\mathbf{1}^T/p\right)A\right)} \tag{32}$$

And finally the translation writes:

$$\mathbf{t} = (B - \lambda AR)^T\,\mathbf{1}/p. \tag{33}$$

To reconcile this notation with the one that is more customary in Computer Vision, it is sufficient to note that:

- points are represented by rows, hence linear operators (e.g., rotations) are represented by post-multiplication with a matrix;
- $A\mathbf{1}/p$ where A is $n \times p$ corresponds to taking the average of the rows;
- $A\left(I - \mathbf{1}\,\mathbf{1}^T/p\right)$ has the effect of subtracting to A its rows average;
- The matrix $\left(I - \mathbf{1}\,\mathbf{1}^T/p\right)$ is symmetric and idempotent.

The EOPA solves also the total least squares formulation of the problem, where both sets are assumed to be corrupted by noise (Arun, 1992)

However, if noise is more realistically considered anisotropic and inhomogeneous, The ML solution becomes a non-linear least-squares, that can be solved with Levenberg-Marquardt. The LM is basically the Gauss-Newton method, to which the gradient descent principle is combined to ensure convergence.

In geodetic science (and Photogrammetry as well), on the other hand, the Gauss-Helmert method is popular for similarity estimation (the similarity transformation is sometimes referred to as the Helmert transformation). The Gauss-Helmert method first linearizes the nonlinear constraint around the current values of the unknowns and expresses the residual as a quadratic function in the increments of the variables. Then, the variables are updated by the increments that minimizes it, and this procedure is iterated.

Gauss-Helmert can be seen as an instance of Gauss-Newton with a specific Hessian approximation (Kanatani and Niitsuma, 2012).

3.4 Separate exterior orientation

In this method the position and attitude of each camera with respect to the object coordinate system (exterior orientation of the camera) is solved independently.

The problem can be solved with the help of the collinearity equations

$$\mathbf{p} = f(\mathbf{O}, \boldsymbol{\omega}, \mathbf{X})$$

that express measured quantities $\mathbf{p}$ as a function of the exterior orientation parameters $\mathbf{O}, \boldsymbol{\omega}$, where the vector $\boldsymbol{\omega}$ collects the three parameters that describe the rotation R.

For every measured point two equations are obtained. If 3 GPS are measured, a total of 6 equations is formed to solve for the 6 parameters of exterior orientation.

The collinearity equations are not linear in the parameters. Therefore, the solution requires approximate values with which the iterative process will start.

3.4.1 Exterior Orientation

The problem of estimating the position and attitude of a perspective camera given its interior parameters and a set of object-to-image correspondences is known as the *Perspective-n-Point* camera pose problem (PnP) in computer vision or *exterior orientation* problem in Photogrammetry

Given a number p of 2D-3D point correspondences $\tilde{\mathbf{p}}_j \leftrightarrow \mathbf{X}_j$ (where $\tilde{\mathbf{p}}_j$ are in NIC) the PnP problem requires to find a rotation matrix R and a translation vector $\mathbf{t}$ (which specify attitude and position of the camera) such that:

$$\zeta_j \tilde{\mathbf{p}}_j = K[R|\mathbf{t}]\tilde{\mathbf{X}}_j \quad \text{for all } j. \tag{34}$$

where ζ_j denotes the depth of $\mathbf{X}_j$.

One could immediately solve this problem by doing camera resection with DLT in NIC istead of image coordinates.

However, this algorithm is sub-optimal, because ut does not enforce the orthonormality constraints on the rotation matrix.

Ad hoc methods for Exterior Orientation should provide an orthogonal matrix by construction (Fiore, 2001; Ansar and Daniilidis, 2003; Lepetit et al., 2009; Gao et al., 2003; Lepetit et al., 2009; Hesch and Roumeliotis, 2011).

In a recent paper (Garro et al., 2012) the image exterior orientation problem have been solved using Procrustean analysis. After some rewriting, (34) becomes:

$$\underbrace{\begin{bmatrix} \zeta_1 & 0 & \dots & 0 \\ \vdots & \vdots & \ddots & \vdots \\ 0 & 0 & \dots & \zeta_p \end{bmatrix}}_{Z} \underbrace{\begin{bmatrix} \tilde{\mathbf{p}}_1^T \\ \vdots \\ \tilde{\mathbf{p}}_p^T \end{bmatrix}}_{P} R + \underbrace{\begin{bmatrix} \mathbf{O}^T \\ \vdots \\ \mathbf{O}^T \end{bmatrix}}_{\mathbf{1}\mathbf{O}^T} = \underbrace{\begin{bmatrix} \mathbf{X}_1^T \\ \vdots \\ \mathbf{X}_p^T \end{bmatrix}}_{S}. \tag{35}$$

where $\mathbf{O} = -R^T\mathbf{t}$, and $\mathbf{1}$ is the unit vector. In matrix form:

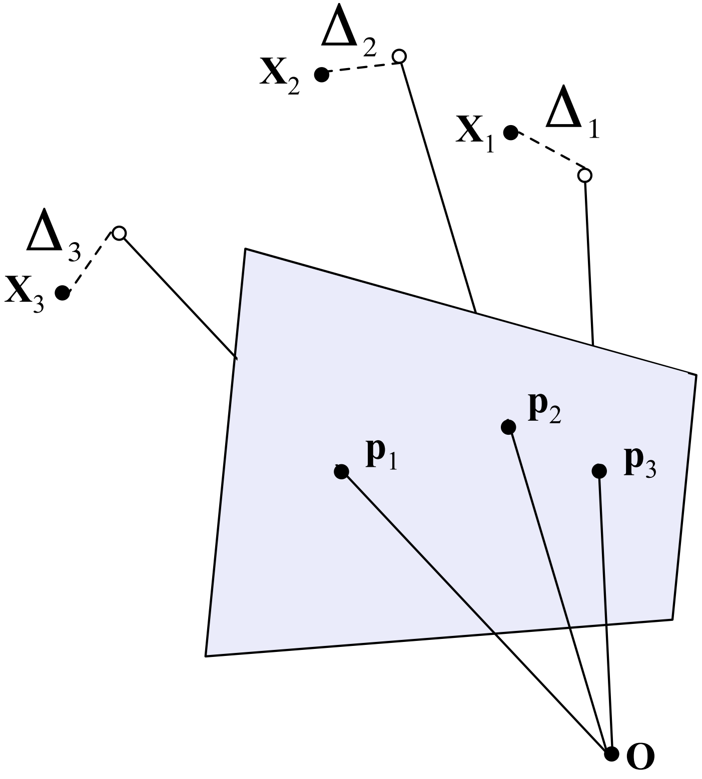
$$S = ZPR + \mathbf{1}\mathbf{O}^T \tag{36}$$

where P is the matrix by rows of (homogeneous) image coordinates defined in the camera frame, S is the matrix by rows of point coordinates defined in the object system, Z is the diagonal (positive) depth matrix, $\mathbf{O}$ is the coordinate vector of the projection centre, and R is the orthogonal rotation matrix.

This is an instance of the EOPA with a diagonal unknown matrix Z of *anisotropic* scales that replaces the uniform scale λ.

The minimization is accomplished with an alternating scheme (also called "block relaxation" (de Leeuw, 1994)), where each variable is alternatively estimated while keeping the others fixed:

- assuming Z is known, use EOPA to find rotation and translation;
- given R and $\mathbf{O}$, solve for Z by finding the position along the (fixed) optical ray that minimizes the distance to the (known) 3D points.



In order to solve this last step, let us rewrite Eq. (36) as:

$$ZP = (S - \mathbf{1}\mathbf{O}^T)R^T \tag{37}$$

or equivalently, with $Y = (S - \mathbf{1}\mathbf{O}^T)R^T$

$$P^T Z = Y^T \tag{38}$$

Since Z is diagonal, the previous equation can be transformed in the followin linear system, thanks to the properties of the Khatri-Rao product ($\odot$):

$$(I \odot P^T)\,\mathrm{diag}^{-1}(Z) = \mathrm{vec}(Y^T) \tag{39}$$

where diag^{-1} returns a vector containing the diagonal elements of its argument.

Non-negativity constraint on Z must be enforced a-posteriori by clipping to zero negative values.

3.5 Combined single stage orientation

This is also called the “bundle” method, and indeed it is equivalent to a bundle adjustment (see ahead) with just two images and GCPs. The basic idea is the following.

Let us rewrite the collinearity equation as

$$\mathbf{p} = f(\mathbf{O}, \boldsymbol{\omega}, \mathbf{X})$$

where the vector $\boldsymbol{\omega}$ collects the three parameters that describe the rotation R; the 6 orientation parameters $(\mathbf{O}, \boldsymbol{\omega})$ are unknown, while some of the 3D points $\mathbf{X}$ are known (GCP) and the others are not (tie-points).

For every GCP seen in one camera two equations can be written in 12 unknown (the orientation of one camera has 6 d.o.f).

For every tie-point (seen in two imgaes) we add four equations and 3 unknowns (its 3D position), therefore the balance is positive: tie-points adds information.

This method can be used *without* GCP, and the model is therefore represented in an arbitrary reference system (*free* solution).

4 Multiple images constraints

The question whether epipolar geometry can be generalized to more than two images arises naturally. Since conjugate points in two images are linked by a bilinear form, one might conjecture that points in tree images are related by a trilinear form, and so on. This is indeed correct, and in this section we shall see how all the meaningful multifocal constraints on N images can be derived in very elegant way, as described in (Heyden, 1998).

Consider one point viewed by m cameras:

$$\zeta_i \tilde{\mathbf{x}}_i = P_i \tilde{\mathbf{X}} \quad i = 1 \dots m \tag{40}$$

By stacking all these equations we obtain:

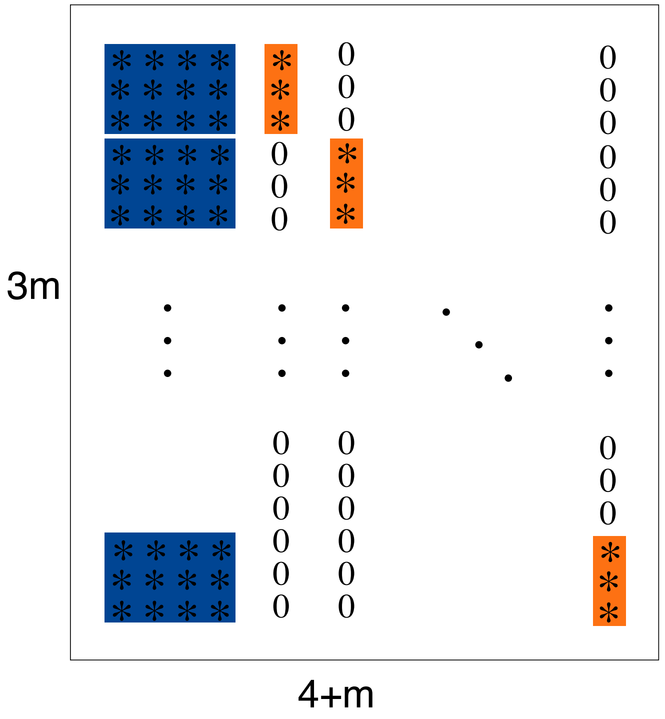
$$\underbrace{\begin{bmatrix} P_1 & \tilde{\mathbf{x}}_1 & 0 & \dots & 0 \\ P_2 & 0 & \tilde{\mathbf{x}}_2 & \dots & 0 \\ \vdots & \vdots & \vdots & \ddots & \vdots \\ P_m & 0 & 0 & \dots & \tilde{\mathbf{x}}_m \end{bmatrix}}_{L} \begin{bmatrix} \tilde{\mathbf{X}} \\ -\zeta_1 \\ -\zeta_2 \\ \vdots \\ -\zeta_m \end{bmatrix} = \begin{bmatrix} 0 \\ 0 \\ \vdots \\ 0 \end{bmatrix} \tag{41}$$

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

The minors that does not contain at least one row from each camera are identically zero, since they contain a zero column.

If a minor contains only one row from some camera, the image coordinate corresponding to this row can be factored out (using Laplace expansion along the corresponding column).

Hence, at least one row has to be taken from each camera to obtain a meaningful constraint, plus another row from each camera to prevent the constraint to be trivially factorized.



Since there are m views, after taking one row from each camera, the remaining four rows can be chosen as follows, depending on the number of cameras:

If $m = 2$ choosing two rows from one image and two rows from another image gives a bifocal (epipolar) constraint.

If $m = 3$, choosing two rows from one image, one row from another image and one row from a third image gives a trifocal constraint.

If $m = 4$, choosing one row from each of four different images gives a quadrifocal constraint.

If $m > 4$, there is no way to avoid that some minors contain only one row from some images.

Hence, constraints involving more than 4 cameras can be factorized as product of the two, three, or four-images constraints and image point coordinates. This indicates that no interesting constraints can be written for more than four images[2].

[2] Actually, it can be proven that also the quadrifocal constraints are not independent (Ma et al., 2003).

In Section 3.2.4 we saw how a camera pair can be extracted from the essential matrix. Likewise, a triplet of consistent cameras can be extracted from the trifocal tensor. The procedure is fairly tricky, though and generalizes only up to four cameras.

As a consequence, there is no direct generalization of the pairwise processing to multiple images, and specific methods have been developed in Photogrammetry and Computer Vision.

5 Block processing

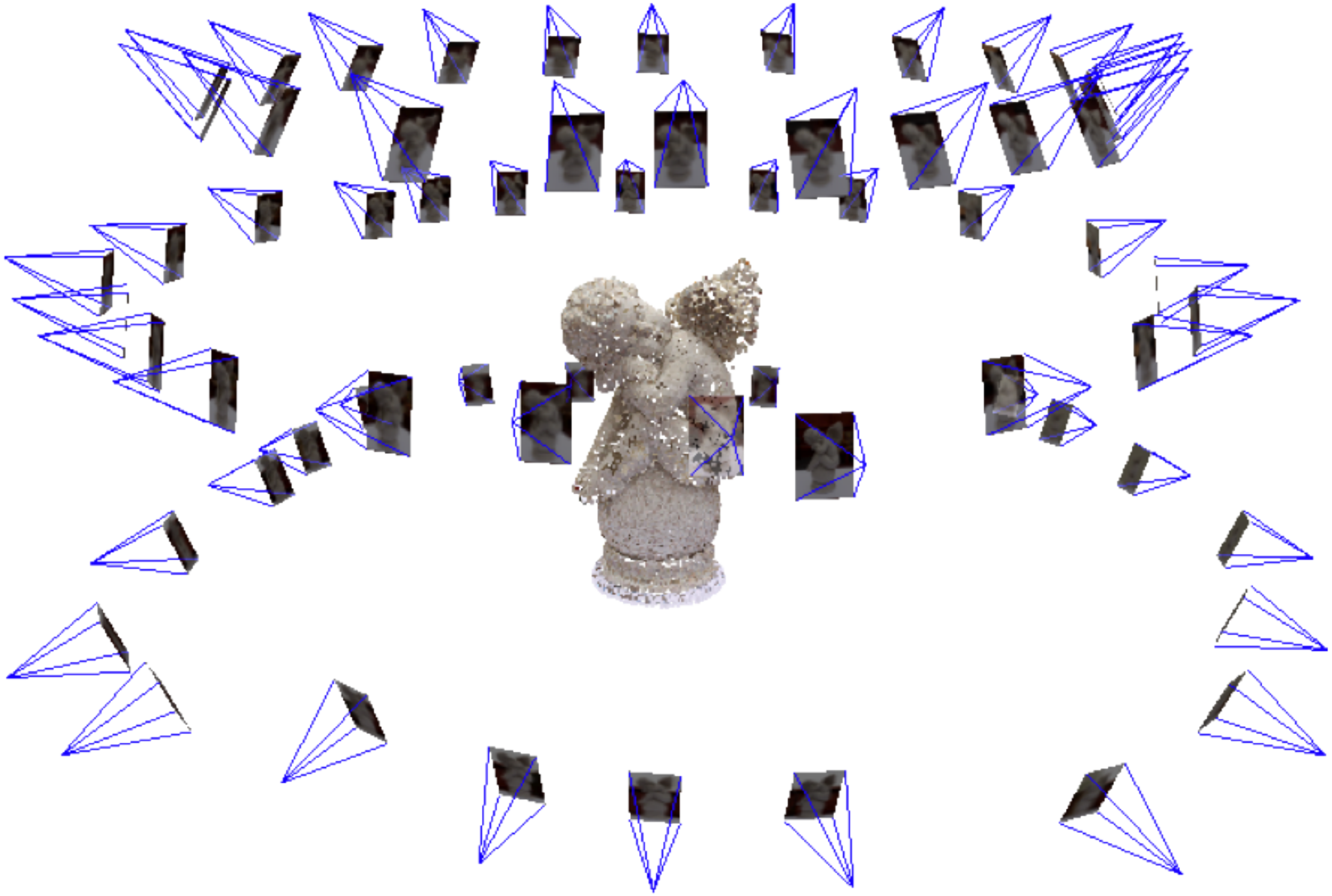
Block processing is the generalization of pairwise processing to multiple overlapping images (a block).

The technique referred to as *Structure from Motion* (SfM) in Computer Vision has a large overlap with the block adjustment problem of Photogrammetry: given multiple images of a stationary scene, the goal is to recover both *structure*, i.e. 3D coordinates of object points, and *motion*, i.e. the exterior orientation (position and attitude) of the photographs.

It is assumed that the interior parameters of the cameras are known, namely the focal length and the coordinates of the principal point.

It is assumed that a certain number of (3-D) tie-points are visible in subsets of images and that they can be identified (via key-point extraction and matching)[3].

[3] Please note that tie-points are 3-D, while key-points or features are 2-D. When two or more key-points are matched they implicitely define a tie-point via intersection.



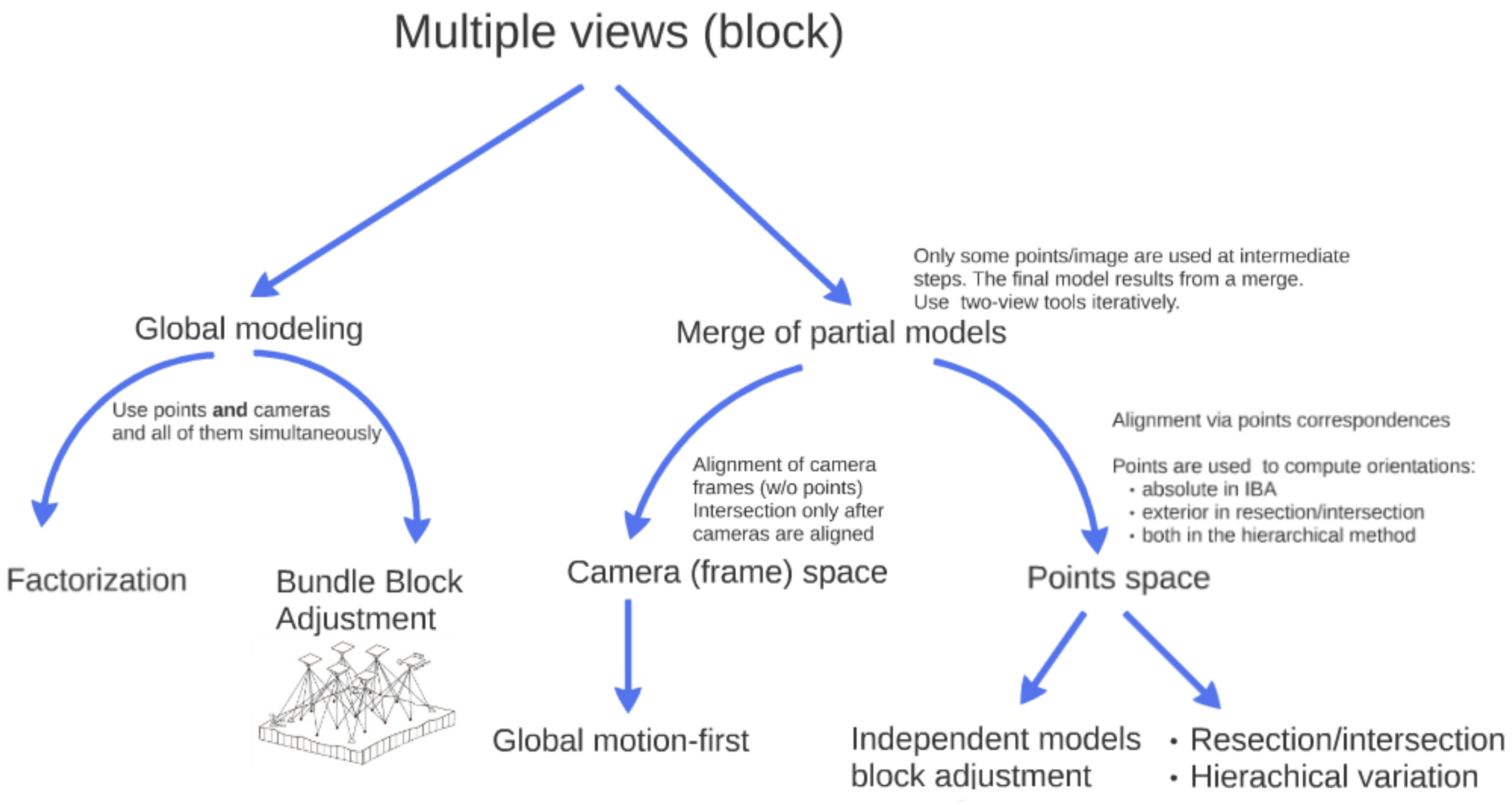

Multiple views (block)
Global modeling
Merge of partial models
Only some points/image are used at intermediate steps. The final model results from a merge. Use two-view tools iteratively.
Use points **and** cameras and all of them simultaneously
Factorization
Bundle Block Adjustment
Alignment of camera frames (w/o points) Intersection only after cameras are aligned
Camera (frame) space
Global motion-first
Alignment via points correspondences
Points are used to compute orientations:
• absolute in IBA
• exterior in resection/intersection
• both in the hierarchical method
Points space
Independent models block adjustment
• Resection/intersection
• Hierachical variation

Fig. 2: The proposed taxonomy of *Structure from Motion* methods

There are many possible taxonomies of the Structure-from-motion methods. We choose to first dichotomize methods that merge partial models vs global methods that uses all points and all cameras simultaneously.

Among the first we single out Bundle Adjustment (e.g. (Triggs et al., 2000)), and factorization-based methods.

The other class is further subdivided according to the the space where the merging occurs: frame space or points space.

In the first case camera frames are aligned before recovering the 3D points (Govindu, 2001; Martinec and Pajdla, 2007; Kahl and Hartley, 2008; Enqvist et al., 2011; Arie-Nachimson et al., 2012; Moulon et al., 2013) (first solve for the "motion" and then recover the "structure"), whereas in the second case 3D points are recovered and then used to guide the alignment.

In the latter group we find the independent models block adjustments (e.g. (Crosilla and Beinat, 2002)), where first stereo-models are built and then co-registered, and structure-*and*-motion methods, such as resection-intersection methods (Brown and Lowe, 2005; Snavely et al., 2006a), hierarchical methods (Gherardi et al., 2010; Ni and Dellaert, 2012)), where "structure" and "motion" are somehow interleaved.

5.1 Bundle block adjustment

Bundle block adjustment minimizes the reprojection error, i.e., the distance in the image plane between the projection of a tie-point $P_i\tilde{\mathbf{X}}^j$ and the corresponding key-points $\tilde{\mathbf{x}}_i^j$ for every image i where they have been detected:

$$\min_{P_i,\tilde{\mathbf{X}}^j} \sum_{i,j} d(P_i\tilde{\mathbf{X}}^j, \tilde{\mathbf{x}}_i^j)^2 \tag{42}$$

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

If the reconstruction is projective P_i is parameterized with its 11 d.o.f. whereas if the reconstruction is Euclidean, one should use $P_i = K_i[R_i|\mathbf{t}_i]$ where the rotation has to be suitably parameterized with 3 d.o.f.

In this case, using the collinearity equations, it can be equivalently written:

$$\min_{\mathbf{O}_i,\boldsymbol{\omega}_i,\mathbf{X}^j} \sum_{i,j} \|\mathbf{p}_i^j - f(\mathbf{O}_i, \boldsymbol{\omega}_i, \mathbf{X}^j)\|^2 \tag{43}$$

with possibly some $\mathbf{X}$ known and fixed (GCPs).

See also (Triggs et al., 2000) for a review and a more detailed discussion on bundle adjustment.

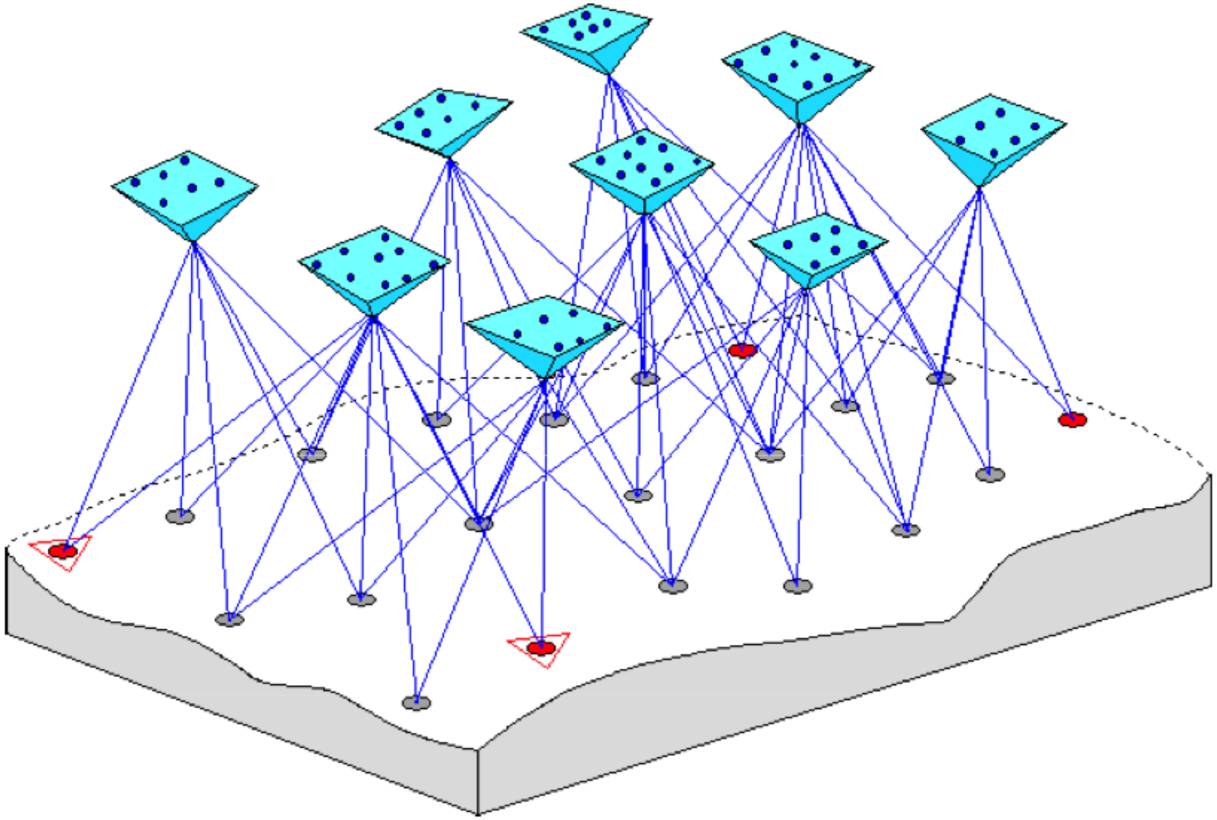
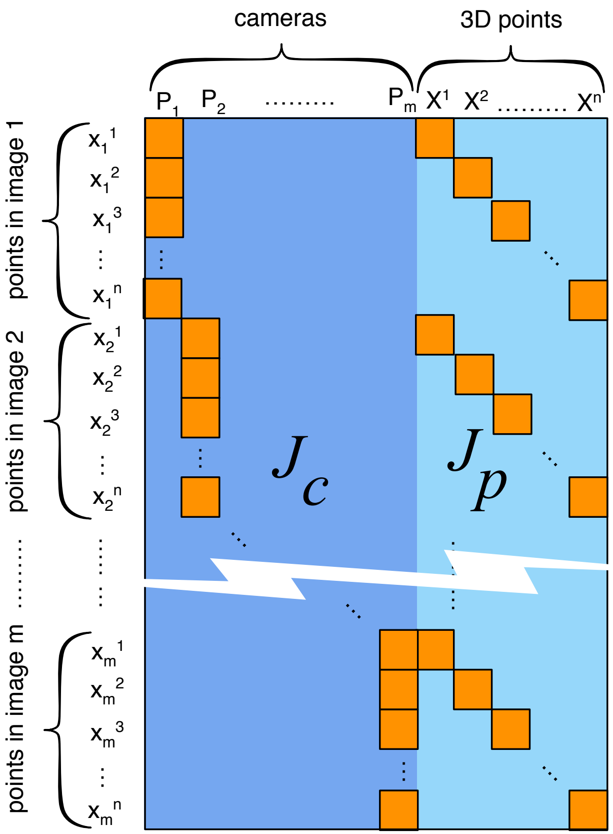


Fig. 3: Bundle adjustment. GCP are in red; tie-points are in gray.

As m and n increase, this becomes a very large minimization problem. However the Jacobian of the residual has a specific structure that can be exploited to gain efficiency.

- Primary structure: on the row corresponding to $\tilde{\mathbf{x}}_i^j$, only the two elements corresponding to camera P_i and to point $\tilde{\mathbf{X}}^j$ are nonzero.
- Secondary structure: not all points are seen in all views (data-dependent).



The primary structure can be exploited to decompose the Jacobian two parts: one relative to cameras and one to points:

$$J = \begin{bmatrix} J_c & J_p \end{bmatrix}. \tag{44}$$

Please note that J_c and J_p are block diagonal. The normal equation

$$\underbrace{J^T J}_{H} \Delta \mathbf{u} = -J^T f(\mathbf{u}) \tag{45}$$

is partitioned accordingly:

$$\begin{bmatrix} J_c^T J_c & J_c^T J_p \\ J_p^T J_c & J_p^T J_p \end{bmatrix} \begin{bmatrix} \Delta \mathbf{u}_c \\ \Delta \mathbf{u}_p \end{bmatrix} = \begin{bmatrix} -J_c^T f(\mathbf{u}_c \,|\, \mathbf{u}_p) \\ -J_p^T f(\mathbf{u}_c \,|\, \mathbf{u}_p) \end{bmatrix} \tag{46}$$

or equivalently:

$$\begin{bmatrix} H_{cc} & H_{cp} \\ H_{pc} & H_{pp} \end{bmatrix} \begin{bmatrix} \Delta \mathbf{u}_c \\ \Delta \mathbf{u}_p \end{bmatrix} = \begin{bmatrix} b_c \\ b_p \end{bmatrix}. \tag{47}$$

where H_{cc} and H_{pp} are block diagonal.

Let us multiply the equation by

$$\begin{bmatrix} I & -H_{cp}H_{pp}^{-1} \\ 0 & I \end{bmatrix} \tag{48}$$

which has the effect of making the lefthand matrix block lower triangular:

$$\begin{bmatrix} H_{cc} - H_{cp}H_{pp}^{-1}H_{pc} & 0 \\ H_{pc} & H_{pp} \end{bmatrix} \begin{bmatrix} \Delta\mathbf{u}_c \\ \Delta\mathbf{u}_p \end{bmatrix} = \begin{bmatrix} b_c - H_{cp}H_{pp}^{-1}b_p \\ b_p \end{bmatrix} \tag{49}$$

Hence the unknown can be recovered as in a blockwise Gaussian elimination

$$(H_{cc} - H_{cp}H_{pp}^{-1}H_{pc})\Delta\mathbf{u}_c = b_c - H_{cp}H_{pp}^{-1}b_p. \tag{50}$$

$$\Delta\mathbf{u}_p = H_{pp}^{-1}(b_p - H_{pc}\Delta\mathbf{u}_c). \tag{51}$$

The linear system is smaller than the original, and the inversion of H_{pp} is made easy by its block structure.

The secondary structure reflects onto the matrix $(H_{cc} - H_{cp}H_{pp}^{-1}H_{pc})$: if each image sees inly a fraction of points it will be sparse. For a sequence of images it has a band structure.

The numerical implementations of BA can differ, but all of them stem from the Gauss-Newton method.

Levemberg-Marquardt is customarily used in Computer Vision, but it is is basically a Gauss-Newton method, to which the gradient descent principle is combined to improve convergence.

If the cost function is weighted by the true measurement covariances, there is no difference between the Gauss-Newton method and the so-called Gauss-Markov adjustment (common in Photogrammetry).

Moreover, all these methods can be seen as instances of a more general class of damped Gauss-Newton methods (Börlin and Grussenmeyer, 2013).

Bundle block adjustment is the optimal (in a ML sense) solution to structure and motion, but it requires to be initialized close to the solution so, it does not solve the problem alone, some other method is needed to bootstrap the reconstruction. Also, it does not deal with the matching stage (how tie-points are obtained).

5.2 Factorization method

When many images are available, an elegant method for multi-image modeling is described in (Sturm and Triggs, 1996), based on the same idea of the factorization method (Tomasi and Kanade, 1992).

Iterative factorization methods (Sturm and Triggs, 1996; Heyden, 1997; Oliensis, 1999; Oliensis and Hartley, 2007) produce a model from multiple images by a two step iteration (a block relaxation, in fact), where in one step a measurement matrix, containing image points coordinates, is factorized with SVD, and in the subsequent step the depths of the points are computed, assuming all the other parameters fixed.

A limitation of these methods is that they work with image coordinates (i.e., uncalibrated images), thereby producing a *projective* model, i.e. a model that differs from the true one by an unknown projectivity of space. The knowledge of the interior parameters of the images (either by calibration or autocalibration) allows to subsequently upgrade the model to a *Euclidean* one, that differs from the true model by a similarity transformation.

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

$$\zeta_i^j \tilde{\mathbf{x}}_i^j = P_i \tilde{\mathbf{X}}^j \quad i = 1 \dots m, \quad j = 1 \dots n. \tag{52}$$

can be written in matrix form:

$$\underbrace{\begin{bmatrix} \zeta_1^1 \tilde{\mathbf{x}}_1^1 & \zeta_1^2 \tilde{\mathbf{x}}_1^2 & \dots & \zeta_1^n \tilde{\mathbf{x}}_1^n \\ \zeta_2^1 \tilde{\mathbf{x}}_2^1 & \zeta_2^2 \tilde{\mathbf{x}}_2^2 & \dots & \zeta_2^n \tilde{\mathbf{x}}_2^n \\ \vdots & \vdots & \ddots & \vdots \\ \zeta_m^1 \tilde{\mathbf{x}}_m^1 & \zeta_m^2 \tilde{\mathbf{x}}_m^2 & \dots & \zeta_m^n \tilde{\mathbf{x}}_m^n \end{bmatrix}}_{\text{scaled measurements } W} = \underbrace{\begin{bmatrix} P_1 \\ P_2 \\ \vdots \\ P_m \end{bmatrix}}_{P} \underbrace{\left[\tilde{\mathbf{X}}^1, \tilde{\mathbf{X}}^2, \dots \tilde{\mathbf{X}}^n\right]}_{\text{structure } M}. \tag{53}$$

In this formula the $\tilde{\mathbf{x}}_i^j$ are known, but all the other quantities are unknown, including the projective depths ζ_i^j. Equation (53) tells us that W can be factored into the product of a $3m \times 4$ matrix P and a $4 \times n$ matrix M. This also means that W has rank four.

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

$$W = UDV^T. \tag{54}$$

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

$$W = U_{3m\times 4} \operatorname{diag}(\sigma_1, \sigma_2, \sigma_3, \sigma_4)\, V_{n\times 4}^T. \tag{55}$$

The sought model is obtained by setting:

$$P = U_{3m\times 4} \operatorname{diag}(\sigma_1, \sigma_2, \sigma_3, \sigma_4) \quad \text{and} \quad M = V_{n\times 4}^T \tag{56}$$

This model is unique up to a (unknown) projective transformation. Indeed, for any non singular projective transformation T, PT and $T^{-1}M$ is an equally valid factorization of the data into projective motion and structure. Consistently, the choice to subsume $\operatorname{diag}(\sigma_1, \sigma_2, \sigma_3, \sigma_4)$ in P is arbitrary.

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

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

where $|| \cdot ||_F$ is the Frobenius norm. As the depth ζ_i^j are unknown, we are left with the problem of estimating them.

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

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

$$\begin{bmatrix} \zeta_1^j \tilde{\mathbf{x}}_1^j \\ \zeta_2^j \tilde{\mathbf{x}}_2^j \\ \vdots \\ \zeta_m^j \tilde{\mathbf{x}}_m^j \end{bmatrix} = \underbrace{\begin{bmatrix} \tilde{\mathbf{x}}_1^j & 0 & \ldots & 0 \\ 0 & \tilde{\mathbf{x}}_2^j & \ldots & 0 \\ \vdots & \vdots & \ddots & \vdots \\ 0 & 0 & \ldots & \tilde{\mathbf{x}}_m^j \end{bmatrix}}_{Q^j} \underbrace{\begin{bmatrix} \zeta_1^j \\ \zeta_2^j \\ \vdots \\ \zeta_m^j \end{bmatrix}}_{\boldsymbol{\zeta}^j} = PM^j \tag{57}$$

The method can be summarized as follows:

1. Start from an initial guess for ζ_i^j (e.g. $\zeta_i^j = 1$)
2. Normalize W such that $||W|||_F = 1$;
3. Factorize W and obtain an estimate of P and M;
4. If $||W - PM||_F^2$ is sufficiently small then stop;
5. Solve for $\boldsymbol{\zeta}^j$ in $Q^j \boldsymbol{\zeta}^j = PM^j$, for all $j = 1 \dots n$;
6. Update W.
7. Repeat from 2. until convergence

Step 2. is necessary to avoid trivial solutions (e.g. $\zeta_i^j = 0$).

Although this technique is fast, requires no initialization, and gives good results in practice, there is no guarantee that the iterative process will converge to a valid solution. A discussion on convergence of this class of methods can be found in (Oliensis and Hartley, 2007).

In this basic version it is of little practical use, though, because it assumes that *all* the point are visible in all the images.

However the issue of visibility in matrix factorization methods can be side-stepped by matrix completion techniques, exploiting the low rank of the measurement matrices (Brand, 2002; Kennedy et al., 2013; Hartley and Schaffalitzky, 2003), or by providing additional information.

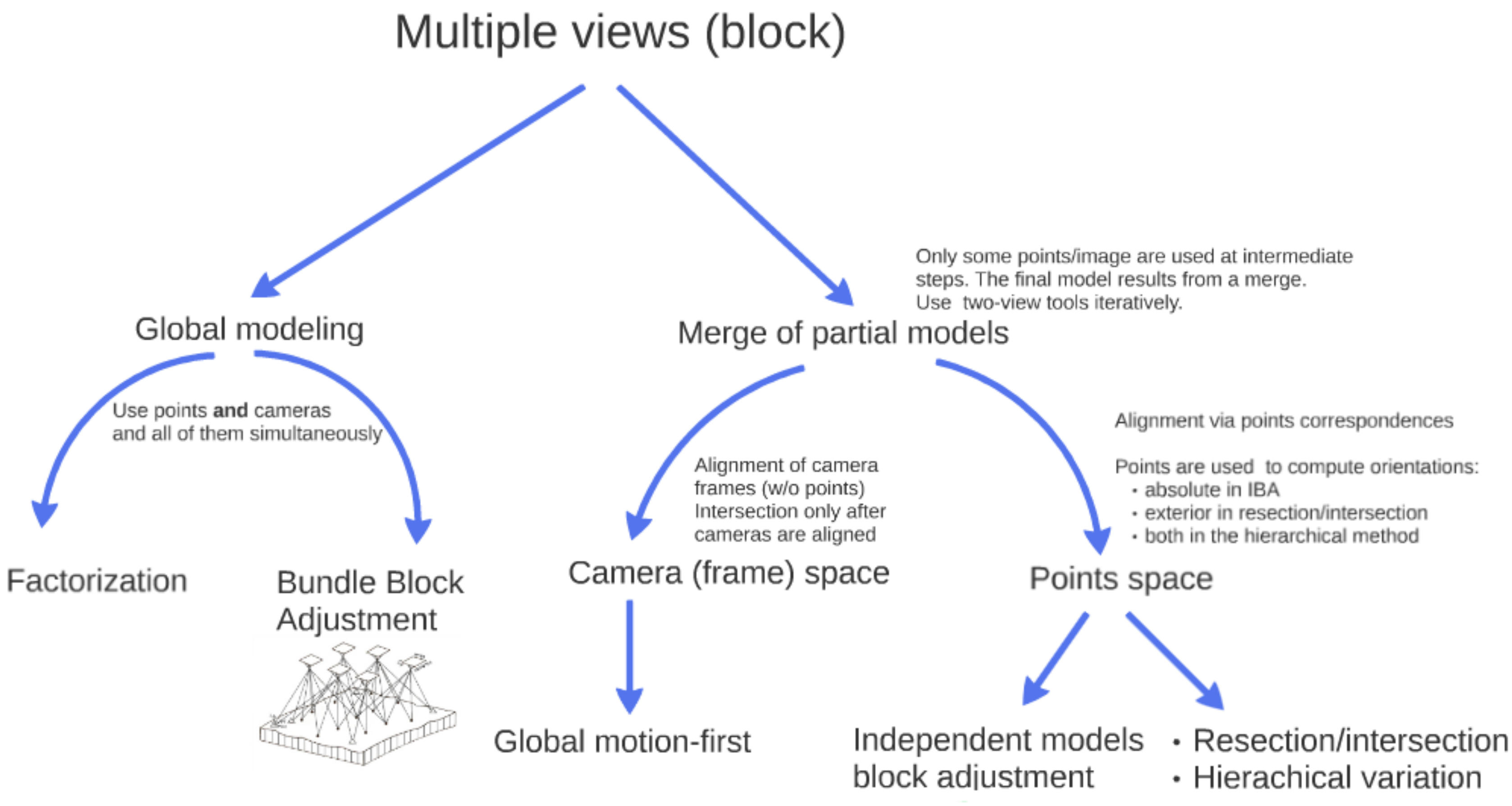

Multiple views (block)
Global modeling
Merge of partial models
Only some points/image are used at intermediate steps. The final model results from a merge. Use two-view tools iteratively.
Use points **and** cameras and all of them simultaneously
Factorization
Bundle Block Adjustment
Alignment of camera frames (w/o points) Intersection only after cameras are aligned
Camera (frame) space
Global motion-first
Alignment via points correspondences
Points are used to compute orientations:
• absolute in IBA
• exterior in resection/intersection
• both in the hierarchical method
Points space
Independent models block adjustment
• Resection/intersection
• Hierachical variation

Fig. 4: The proposed taxonomy of *Structure from Motion* methods

5.3 Independent models block adjustment

This is a classical method in Photogrammetry (Kraus, 2007).

- First stereo-models are created, by relative orientation. Each of these models is described in an local, arbitrary reference frame.
- In the course of the block adjustment the individual models will be amalgamated into a single one and simutaneously transformed into the ground coordinate system.

We will consider here a Procrustean approach proposed in (Crosilla and Beinat, 2002), which by its nature produces a “free” solution (i.e., expressed in an arbitrary reference frame), but can be modified to include GCPs.

5.3.1 Generalized Procrustes Analysis

Generalized Procrustes Analysis (GPA) is a technique that generalizes EOPA and provides a least-squares solution when more than two model points matrices are present (Gower, 1975). It minimize the following least squares objective function:

$$\sum_{i=1}^{m} \sum_{j=i+1}^{m} \|(\lambda_i A_i R_i + \mathbf{1t}_i^T) - (\lambda_j A_j R_j + \mathbf{1t}_j^T)\| \tag{58}$$

where $A_1, A_2, \ldots, A_m$ are m model points matrices, which contain the same set of k-d p points in m different coordinate systems. The GPA objective function has an alternative formulation. Said $B_i = \lambda_i A_i R_i + \mathbf{1t}_i^T$, the following equivalence holds:

$$\sum_{i=1}^{m} \sum_{j=i+1}^{m} ||B_i - B_j||^2 = m \sum_{i=1}^{m} ||B_i - K||^2, \tag{59}$$

where K is the geometrical centroid,

$$K = \frac{1}{m} \sum_{i=1}^{m} B_i. \tag{60}$$

Therefore the righthand term of Eq. 59 can be minimized – instead of Eq. 58 - in order to determine the unknowns $\lambda_i, R_i, \mathbf{t}_i (i = 1 \ldots m)$.

The unknown centroid can be iteratively estimated, according to the following procedure.

1. First the centroid K is initialized.
2. Iterate:
   (a) At each step a direct solution of the transformation parameters of each model points matrix A_i with respect to the centroid K is found by means of a EOPA solution.
   (b) After the update, a new centroid can be estimated.
3. The procedure continues until global convergence, i.e. the stabilization of the centroid K.

The algorithm always converges (Commandeur, 1991), though not necessarily to the global minimum.

GPA can be used also in the global registration of multiple 3D point sets (Beinat and Crosilla, 2001). The difference with IMBA is that that the transformation is a rigid one (6 d.o.f.) instrad of a similarity (7 d.o.f.).

5.3.2 Anisotropic Generalized Procrustes Analysis

We will derive here a procustean solution to the bundle adjustment, on the same line as in (Fusiello and Crosilla, 2015).

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

$$\zeta_i^j \tilde{\mathbf{p}}_i^j = [R_i | \mathbf{t}_i] \tilde{\mathbf{X}}^j \quad i = 1 \dots m, \quad j = 1 \dots n. \tag{61}$$

Working as in Eq. (36), we can write for each camera i:

$$S = Z_i P_i R_i + \mathbf{1 O}_i^T \tag{62}$$

In this formula the P_i are known, but all the other quantities are unknown, including the depths Z_i. We are required to minimize:

$$\sum_{i=1}^{m} \sum_{\ell=\ell+1}^{m} \|(Z_i P_i R_i + \mathbf{1 O}_i^T) - (Z_\ell P_\ell R_\ell + \mathbf{1 O}_\ell^T)\|^2$$

This formulation matches the GPA problem with the difference that the isotropic scale λ_i is substituted by an anisotroipic scaling matrix Z_i (diagonal).

The iterative solution is modelled onto the GPA solution, with the difference that the Z_i matrices are computed by solving:

$$(I \odot P_i^T)\,\mathrm{diag}^{-1}(Z_i) = \mathrm{vec}(Y_i^T) \tag{63}$$

where all the remaining unknowns are contained in Y_i.

The final reconstruction will be referred to an arbitrary reference system used in the generalized extended procustes solution. Georeferencing can be accomplished by a-posteriory by solving an absolute orientation problem.

5.4 Resection-intersection method

As of today, the most succecesful structure-from-motion pipelines in CV (Brown and Lowe, 2005; Snavely et al., 2006b; Vergauwen and Gool, 2006; Irschara et al., 2007; Gherardi et al., 2010) are based on the idea of growing partial models – composed by cameras and points – where new cameras are iteratively added by *resection* and new points by *intersection*. This approach offers the advantage that corresponding features are not required to be visible in all images.

The idea was indeed already known in Photogrammetry (Kraus, 1997, Sec. 4.1), but the CV community coupled it with automated designation of tie-points (SIFT extraction and matching) and resilience to rogue data (RANSAC), achieving the first completely automatic pipeline, from images to 3D models.

We present here an approach to reconstruct the projective (or Euclidean) structure and motion from a *sequence*[4] of images.

We assume that for each pair of consecutive images we are given a set of corresponding keypoints.

The 3D point of which they are projection is called a *tie-point*.

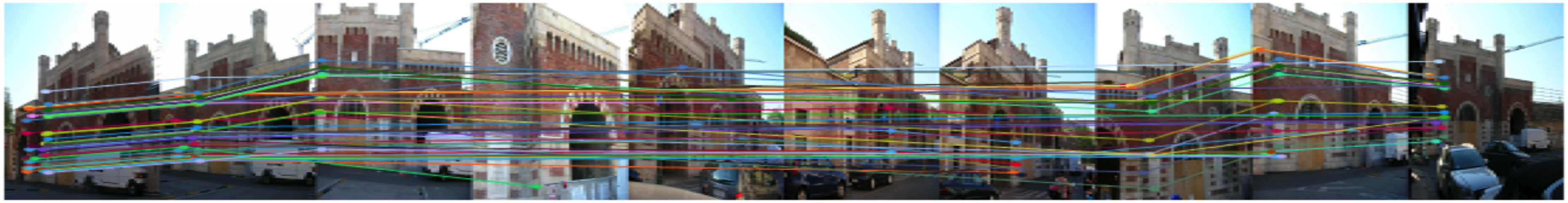


Fig. 5: Keypoints are connected into multiple-view correspondences, called *tracks*.

[4] We assume that even if images are unordered they can be suitably sequenced

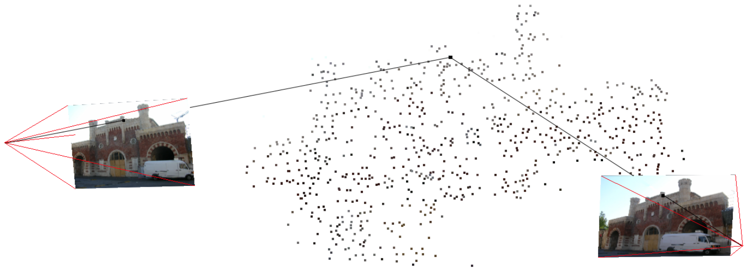
Initializing the model Two images of the sequence are used to initialize the process.

The model reference frame is aligned with the first camera P_1.

The second camera P_2 is chosen so that the epipolar geometry corresponds to the computed *fundamental matrix* F (projective) or *essential matrix* E (Euclidean).

Once P_1 and P_2 have been determined, the coordinates ot the tie-points visible in the two images can be reconstructed through *intersection*.

In other words, the initialization consist in solving a *relative orientation* problem and building a stereo-model.



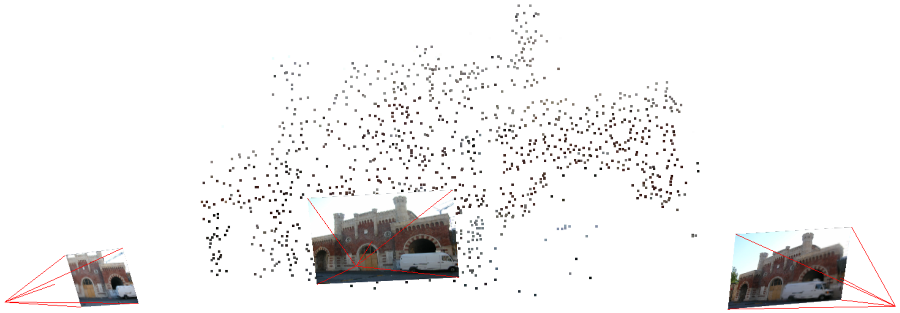
Updating the model After initialization, the following operations are carried out for every additional image $i > 2$.

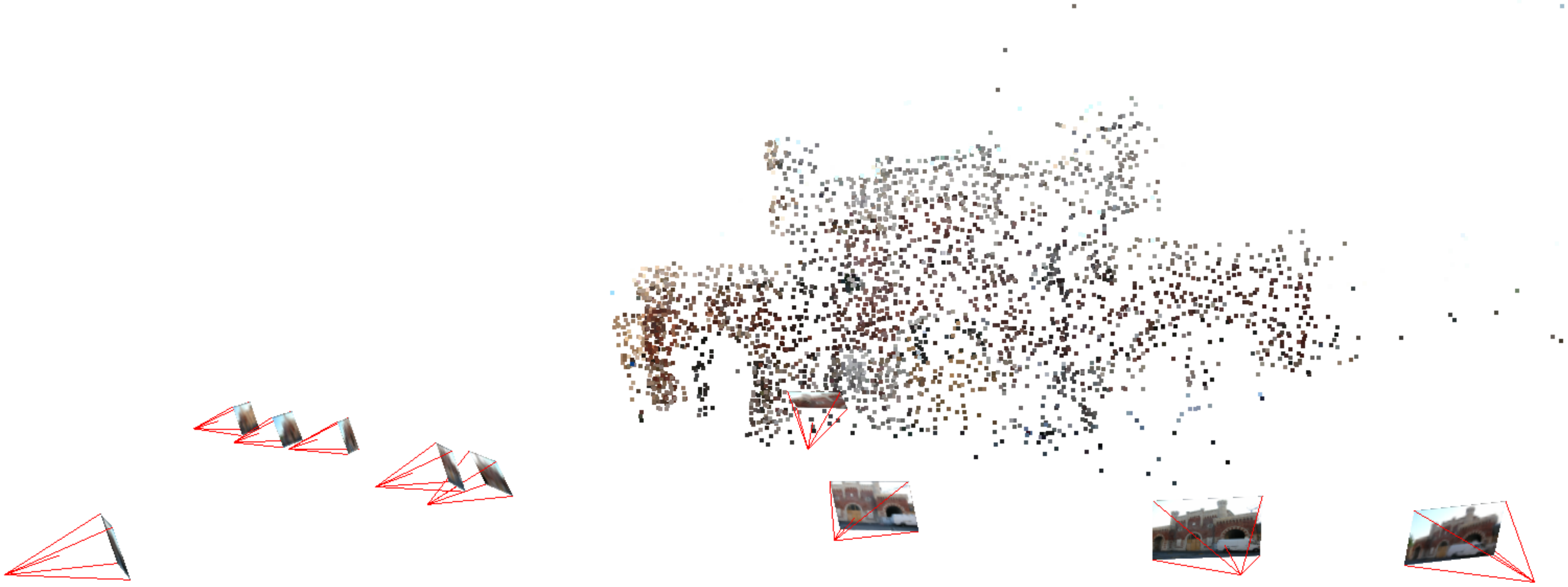
The projection matrix P_i is computed with *exterior orientation* in the Euclidean case or *resection* in the projective case, using tie-points that are visible in image i whose coordinates have been already computed in previous steps.

The model is updated with *intersection*. This entails:

(i) refining the position of tie-points already present in the model and (ii) adding new tie-points thanks to the new matches brought by image i.

Frequent *bundle adjustment* is needed in practice to to contain error accumulation.





5.5 Hierarchical approach

The previous method can be generalized by organizing the photographs on a tree instead of a chain.

The tree is produced by hierarchical clustering the photographs according to their overlap.

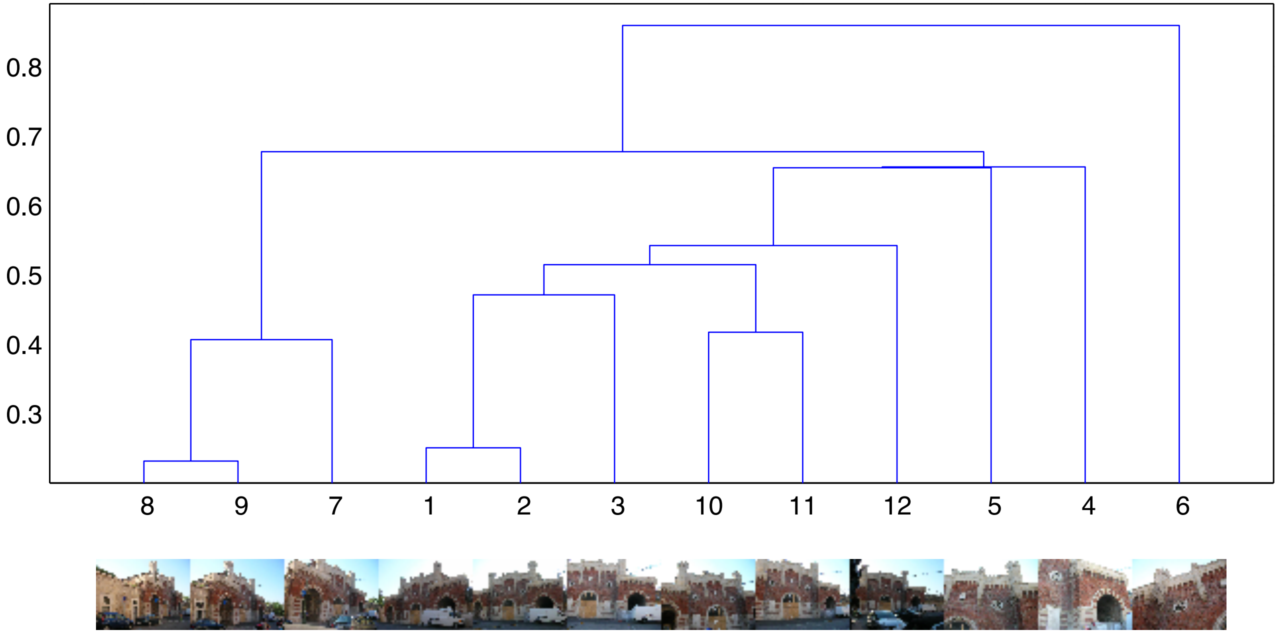


Fig. 6: Dendrogram resulting from hierarchical clustering of images

This hierarchical algorithm, called Samantha in (Gherardi et al., 2010), can be summarized as follows shows a sample model:

1. Solve many independent relative orientation problems at the leaves of the tree, producing many independent stereo-models.
2. Traverse the tree; in each node one of these operations takes place:
   (a) Update one model by adding one image by resection followed by intersection;
   (b) Merge two independent models with absolute orientation.

Steps 1. and 2.(a) are the resection-intersection steps.

Step 2.(b) summons up the photogrammetric Independent Models Block Adjustment.

If the tree reduces to a chain, the algorithm is the resection-intersection method. If the tree is perfectly balanced, only step 2.(b) is taken, and the resulting procedure resembles the IMBA.

5.5.1 Preprocessing (hints)

We assumed images come in a sequence, but usually they are unordered.
The following steps are usually taken:

- Keypoint extraction (usually SIFT or similar)
- Matching – broad phase: select a $O(m)$ pairs to be matched
- Matching – narrow phase: match keypoints between those pairs
- Define the seed pair (critical)
- Define the order of processing of the subsequent views

The two-phases matching avoids the matching of all $O(m^2)$ view pairs.

In the hierarchical approach, the last two steps are substituted by hierarchical clustering.

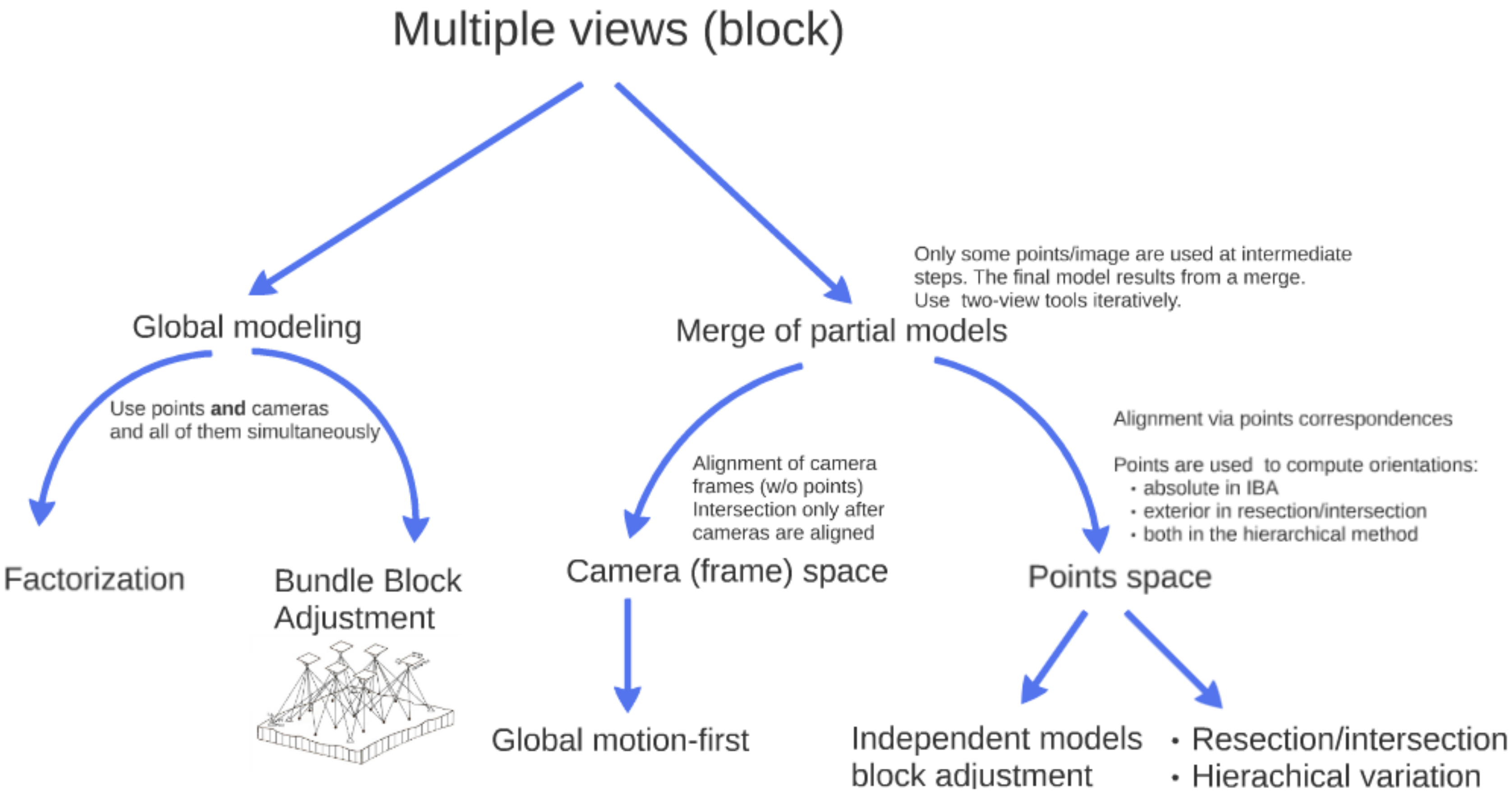

Multiple views (block)
Global modeling
Merge of partial models
Only some points/image are used at intermediate steps. The final model results from a merge.
Use two-view tools iteratively.
Use points **and** cameras and all of them simultaneously
Factorization
Bundle Block Adjustment
Alignment of camera frames (w/o points) Intersection only after cameras are aligned
Camera (frame) space
Alignment via points correspondences
Points are used to compute orientations:
• absolute in IBA
• exterior in resection/intersection
• both in the hierarchical method
Points space
Global motion-first
Independent models block adjustment
• Resection/intersection
• Hierachical variation

Fig. 7: The proposed taxonomy of *Structure from Motion* methods

5.6 Global motion first

Global motion-first methods share a common scheme:

- Solve relative orientation between pair of images, which results in relative rotations and relative translations (up to a scale);
- Solve a motion averaging or syncronzation problem. This is usually broken in a rotation syncronization followed by translation syncronization, but one-step methods have also been proposed.
- The model is computed (by intersection) only at the end.

These global methods are usually faster than the others, while ensuring a fair distribution of the errors among the cameras, being global.

Although the accuracy is worse than those achieved by bundle adjustment, these global methods can be seen as an effective and efficient way of computing approximate orientations to be subsequently refined by bundle adjustment.

Go to:

Synchronization problems in Computer Vision

A Kronecker product

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

$$A \otimes B = \begin{bmatrix} a_{11}B & \dots & a_{1n}B \\ \vdots & & \vdots \\ a_{m1}B & \dots & a_{mn}B \end{bmatrix}. \tag{64}$$

The Kronecker product is defined for any pair of matrices A and B. It is associative and distributive with respect to matrix sum and product, but it is not commutative. The transpose of a Kronecker product is $(A \otimes B)^T = A^T \otimes B^T$.

A very important property concerns the eigenvalues of the Kronecker product: the eigenvalues of $A \otimes B$ are the outer product of the eigenvalues of A and B. This implies that:

$$\operatorname{rank}(A \otimes B) = \operatorname{rank}(A)\operatorname{rank}(B). \tag{65}$$

Vectorization

The *vectorization* of a matrix is a linear transformation which converts the matrix into a column vector. Specifically, the vectorization of the matrix A, denoted by $\mathrm{vec}(A)$, is the vector obtained by stacking the columns of A one underneath the other.

The basic connection between the vec operator and the Kronecker product is

$$\mathrm{vec}(\mathbf{a}\mathbf{b}^T) = \mathbf{b} \otimes \mathbf{a} \tag{66}$$

for any column vectors $\mathbf{a}$ and $\mathbf{b}$. The generalization of this is the following important property:

$$\mathrm{vec}(AXB) = (B^T \otimes A)\,\mathrm{vec}(X) \tag{67}$$

for matrices A, B, X of compatible dimensions.

The *half-vectorization*, $\mathrm{vech}(A)$, of a symmetric $n \times n$ matrix A is the $n(n+1)/2 \times 1$ column vector obtained by vectorizing only the lower triangular part of A.

The *duplication matrix* D_n is the unique $n^2 \times n(n+1)/2$ matrix which, transforms $\mathrm{vech}(A)$ into $\mathrm{vec}(A)$: $D_n\,\mathrm{vech}(A) = \mathrm{vec}(A)$.

B Khatri-Rao product

The Khatri-Rao product (Khatri and Rao, 1968), denoted by $\odot$, is in some sense a partitioned Kronecker product, where by default the column-wise partitioning is considered.

Let us consider two matrices A of order $p \times r$ and B of order $q \times r$ and denote the columns of A by $\mathbf{a}_1 \cdots \mathbf{a}_r$ and the those of B by $\mathbf{b}_1 \cdots \mathbf{b}_r$. The Khatri-Rao product is defined to be the partitioned matrix of order $pq \times r$:

$$A \odot B = [\mathbf{a}_1 \otimes \mathbf{b}_1, \cdots \mathbf{a}_r \otimes \mathbf{b}_r] \tag{68}$$

where $\otimes$ denotes the Kronecker product.

If X is diagonal, then

$$\mathrm{vec}(AXB) = (B^T \odot A)\,\mathrm{diag}^{-1}(X) \tag{69}$$

where diag^{-1} returns a vector containing the diagonal elements of its argument.

With $B = I$ one obtains

$$\mathrm{vec}(AX) = (I \odot A)\,\mathrm{diag}^{-1}(X). \tag{70}$$

It it is easy to see that

$$(I \odot A) = \mathrm{blkdiag}(\mathbf{a}_1 \ldots \mathbf{a}_n) \tag{71}$$

where $\mathbf{a}_1 \ldots \mathbf{a}_n$ are the columns of A and blkdiag is the operator that construct a block diagonal matrix with its arguments as blocks.

References

Ansar, A., Daniilidis, K., 2003. Linear pose estimation from points or lines. IEEE Transactions on Pattern Analysis and Machine Intelligence 25 (5), 578 – 589.

Arie-Nachimson, M., Kovalsky, S. Z., Kemelmacher-Shlizerman, I., Singer, A., Basri, R., 2012. Global motion estimation from point matches. International Conference on 3D Imaging, Modeling, Processing, Visualization and Transmission.

Arun, K. S., 1992. A unitarily constrained total least squares problem in signal processing. SIAM Journal on Matrix Analysis and Applications 13 (3), 729–745.

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

Beinat, A., Crosilla, F., 2001. Generalized procrustes analysis for size and shape 3d object reconstruction. In: Optical 3-D Measurement Techniques. pp. 345–353.

Börlin, N., Grussenmeyer, P., 2013. Bundle adjustment with and without damping. The Photogrammetric Record 28 (144), 396–415.

Brand, M., 2002. Incremental singular value decomposition of uncertain data with missing values. In: Proceedings of the European Conference on Computer Vision. Springer, pp. 707–720.

Brown, M., Lowe, D. G., June 2005. Unsupervised 3D object recognition and reconstruction in unordered datasets. In: Proceedings of the International Conference on 3D Digital Imaging and Modeling.

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 (3), 195–209.

de Leeuw, J., 1994. Block-relaxation algorithms in statistics. In: Information Systems and Data Analysis. Springer-Verlag, p. 308325.

Enqvist, O., Kahl, F., Olsson, C., 2011. Non-sequential structure from motion. In: Eleventh Workshop on Omnidirectional Vision, Camera Networks and Non-classical Camera.

Fiore, P. D., 2001. Efficient linear solution of exterior orientation. IEEE Transactions on Pattern Analysis and Machine Intelligence 23 (2), 140–148.

Fusiello, A., Crosilla, F., April 2015. Solving bundle block adjustment by generalized anisotropic procrustes analysis. ISPRS Journal of Photogrammetry and Remote Sensing 102, 209–221.

Gao, X.-S., Hou, X.-R., Tang, J., Cheng, H.-F., 2003. Complete solution classification for the perspective-three-point problem. IEEE Transactions on Pattern Analysis and Machine Intelligence 25, 930–943.

Garro, V., Crosilla, F., Fusiello, A., 2012. Solving the pnp problem with anisotropic orthogonal procrustes analysis. In: Second Joint 3DIM/3DPVT Conference: 3D Imaging, Modeling, Processing, Visualization and Transmission (3DIMPVT). pp. 262–269.

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).

Govindu, V. M., 2001. Combining two-view constraints for motion estimation. In: Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition.

Gower, J., 1975. Generalized procrustes analysis. Psychometrika 40 (1), 33–51.

Gower, J. C., Dijksterhuis, G. B., January/Winter 2004. Procrustes problems. Vol. 30 of Oxford Statistical Science Series. Oxford University Press, Oxford, UK.

Hartley, R., Schaffalitzky, F., 2003. PowerFactorization: 3D reconstruction with missing or uncertain data. In: Australia-Japan advanced workshop on computer vision. Vol. 74. pp. 76–85.

Hartley, R., Zisserman, A., 2003. Multiple View Geometry in Computer Vision, 2nd Edition. Cambridge University Press.

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

Hesch, J. A., Roumeliotis, S. I., 2011. A direct least-squares (dls) solution for PnP. In: Proc. of the International Conference on Computer Vision.

Heyden, A., 1997. Projective structure and motion from image sequences using subspace methods. In: Scandinavian Conference on Image Analysis. pp. 963–968.

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

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

Irschara, A., Zach, C., Bischof, H., 2007. Towards wiki-based dense city modeling. In: Proceedings of the 11th International Conference on Computer Vision. pp. 1–8.

Kahl, F., Hartley, R. I., 2008. Multiple-view geometry under the l^∞-norm. IEEE Trans. Pattern Anal. Mach. Intell. 30 (9), 1603–1617.

Kanatani, K., Niitsuma, H., 2012. Optimal computation of 3-d similarity: Gaussnewton vs. gausshelmert. Computational Statistics & Data Analysis 56 (12), 4470 – 4483.

Kennedy, R., Balzano, L., Wright, S. J., Taylor, C. J., 2013. Online algorithms for factorization-based structure from motion. CoRR abs/1309.6964.

Khatri, C. G., Rao, C. R., 1968. Solutions to some functional equations and their applications to characterization of probability distributions. Sankhyā: The Indian Journal of Statistics 30 (2), 167–180.

Kraus, K., 1997. Photogrammetry: Advanced methods and applications. Vol. 2. Dümmler.

Kraus, K., 2007. Photogrammetry - Geometry from Images and Laser Scans - 2nd edition. Walter de Gruyter, Berlin.

Lepetit, V., Moreno-Noguer, F., Fua, P., 2009. Ep*n*p: An accurate $o(n)$ solution to the p*n*p problem. International Journal of Computer Vision 81 (2), 155–166.

Ma, Y., Soatto, S., Kosecka, J., Sastry, S. S., November 2003. An Invitation to 3-D Vision. Springer.

Magnus, J. R., Neudecker, H., 1999. "Matrix Differential Calculus with Applications in Statistics and Econometrics", revised Edition. John Wiley & Sons.

Martinec, D., Pajdla, T., 2007. Robust rotation and translation estimation in multiview reconstruction. In: Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition.

Moulon, P., Monasse, P., Marlet, R., December 2013. Global Fusion of Relative Motions for Robust, Accurate and Scalable Structure from Motion. In: Proceedings of the International Conference on Computer Vision. Sydney, Australie, p. to appear.

Ni, K., Dellaert, F., 2012. Hypersfm. 2011 International Conference on 3D Imaging, Modeling, Processing, Visualization and Transmission 0, 144–151.

Oliensis, J., 1999. Fast and accurate self-calibration. In: Proceedings of the International Conference on Computer Vision.

Oliensis, J., Hartley, R., 2007. Iterative extensions of the sturm/triggs algorithm: Convergence and nonconvergence. IEEE Transactions on Pattern Analysis and Machine Intelligence 29 (12), 2217–2233.

Schnemann, P., Carroll, R., 1970. Fitting one matrix to another under choice of a central dilation and a rigid motion. Psychometrika 35 (2), 245–255.

Snavely, N., Seitz, S. M., Szeliski, R., July 2006a. Photo tourism: Exploring photo collections in 3D. ACM Transactions on Graphics 25 (3), 835–846.

Snavely, N., Seitz, S. M., Szeliski, R., 2006b. Photo tourism: exploring photo collections in 3d. In: SIGGRAPH: International Conference on Computer Graphics and Interactive Techniques. New York, NY, USA, pp. 835–846.

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

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

Triggs, B., McLauchlan, P. F., Hartley, R. I., Fitzgibbon, A. W., 2000. Bundle adjustment - a modern synthesis. In: Proceedings of the International Workshop on Vision Algorithms. Springer-Verlag, pp. 298–372.

Vergauwen, M., Gool, L. V., 2006. Web-based 3D reconstruction service. Machine Vision and Applications 17 (6), 411–426.

Wahba, G., July 1965. A Least Squares Estimate of Satellite Attitude. SIAM Review 7 (3).