Chapter 3

Structure from Motion

MRGCV Computer Vision

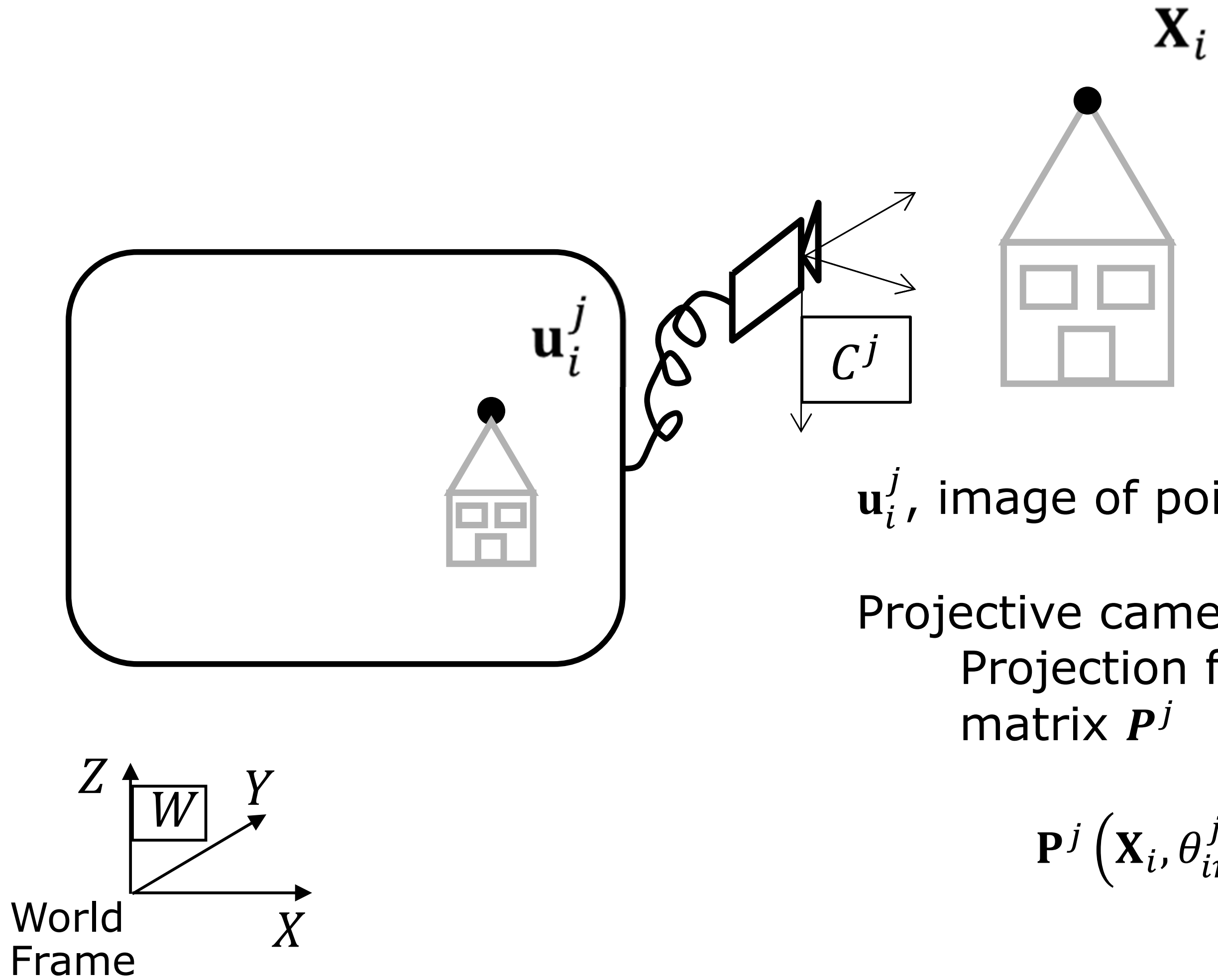
JM Martínez Montiel

Universidad Zaragoza
1542

Structure from Motion

1. Point projection
2. Triangulation
3. Motion from E
4. Structure from Motion
5. Camera Pose
6. Bundle Adjustment
    6.1 Intro
    6.2 Two view BA
    6.3 Non-Linear optimization
    6.4 Initial guess
    6.5 Same goal function. Different estimates.
    6.6 Guage freedoms
    6.7 Phototurism example
    6.8 Typical BA sizes

1. Point projection



$$\mathbf{u}_i^j = \mathbf{P}^j\left(\mathbf{X}_i, \theta_{int}^j, \theta_{ext}^j\right)$$

$$\theta_{int}^j = \left(f_x, f_y, c_x, c_y, \kappa_1, \kappa_2\right)$$

$$\theta_{ext}^j = \left(\mathbf{R}_{cw}^j, \mathbf{t}_{cw}^j\right)$$

$\mathbf{u}_i^j$, image of point $\mathbf{X}_i$ in the camera C^j

Projective cameras, $\kappa_1 = 0, \kappa_2 = 0$

Projection function defined by the camera matrix $\boldsymbol{P}^j$

$$\mathbf{P}^j\left(\mathbf{X}_i, \theta_{int}^j, \theta_{ext}^j\right) = \boldsymbol{P}^j\mathbf{X}_i = \boldsymbol{K}^j\left[\mathbf{R}_{cw}^j \middle| \mathbf{t}_{cw}^j\right]\mathbf{X}_i$$

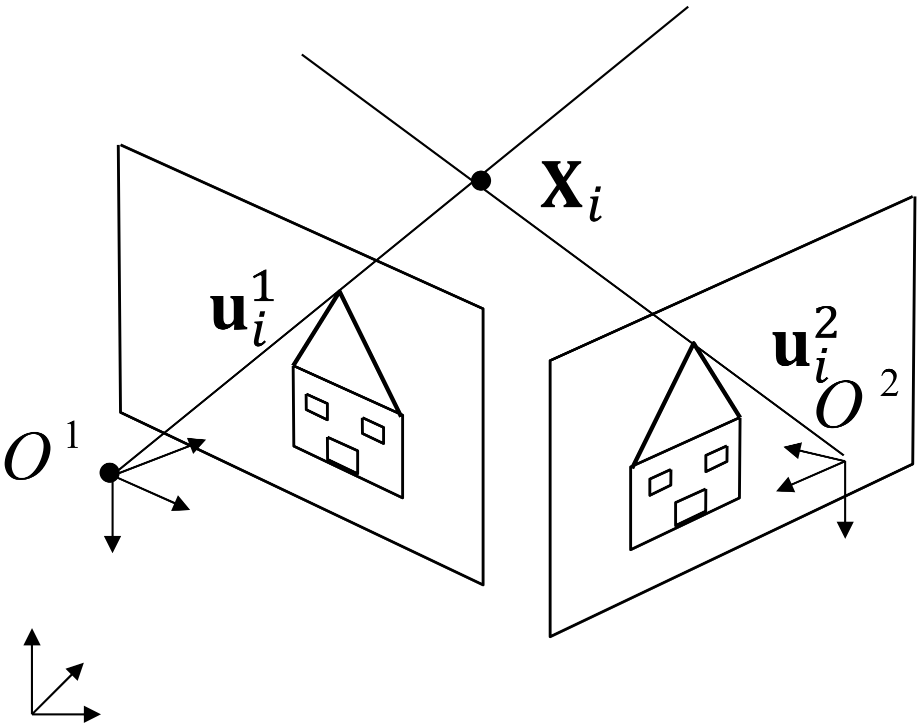
Cameras with optical distortion, $\kappa_1 \neq 0, \kappa_2 \neq 0$

$\mathbf{P}^j\left(\mathbf{X}_i, \theta_{int}^j, \theta_{ext}^j\right)$ projection function includes both perspective projection and distortion.

Universidad Zaragoza

2. Triangulation

If a point is observed in two calibrated projective cameras, with known camera poses, its 3D pose can be recovered



$$\mathbf{u}_i^1, \mathbf{u}_i^2, \theta_{int}^1, \theta_{ext}^1, \theta_{int}^2, \theta_{ext}^2$$

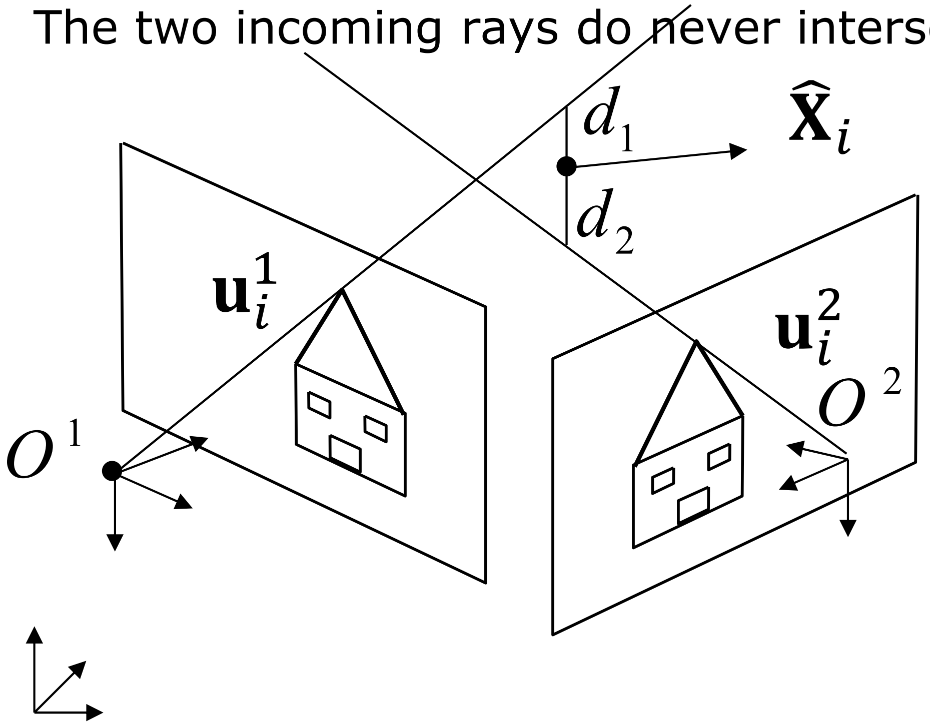
$$\Downarrow$$

$$\mathbf{X}_i$$

Intersection of two incoming rays

2. Triangulation

The two incoming rays do never intersect!



Approximate intersection

$$\mathbf{X}_i \cong \hat{\mathbf{X}}_i = \underset{\hat{\mathbf{X}}_i}{\operatorname{argmin}} \left(d_1^2 + d_2^2\right)$$

2. SVD Triangulation

$$\begin{bmatrix} s_i^j x_i^j \\ s_i^j y_i^j \\ s_i^j \end{bmatrix} = \begin{bmatrix} p_{00}^j & p_{01}^j & p_{02}^j & p_{03}^j \\ p_{10}^j & p_{11}^j & p_{12}^j & p_{13}^j \\ p_{20}^j & p_{21}^j & p_{22}^j & p_{23}^j \end{bmatrix} \begin{bmatrix} X_i \\ Y_i \\ Z_i \\ W_i \end{bmatrix} \qquad s_i^j = p_{20}^j X_i + p_{21}^j Y_i + p_{22}^j Z_i + p_{23}^j W_i$$

Substituting s_i^j in the first and second rows

$$p_{20}^j x_i^j X_i + p_{21}^j x_i^j Y_i + p_{22}^j x_i^j Z_i + p_{23}^j x_i^j W_i = p_{00}^j X_i + p_{01}^j Y_i + p_{02}^j Z_i + p_{03}^j W_i$$
$$p_{20}^j y_i^j X_i + p_{21}^j y_i^j Y_i + p_{22}^j y_i^j Z_i + p_{23}^j y_i^j W_i = p_{10}^j X_i + p_{11}^j Y_i + p_{12}^j Z_i + p_{13}^j W_i$$

Each observation of a point in a camera yields two equations

$$\begin{bmatrix} p_{20}^j x_i^j - p_{00}^j & p_{21}^j x_i^j - p_{01}^j & p_{22}^j x_i^j - p_{02}^j & p_{23}^j x_i^j - p_{03}^j \\ p_{20}^j y_i^j - p_{10}^j & p_{21}^j y_i^j - p_{11}^j & p_{22}^j y_i^j - p_{12}^j & p_{23}^j y_i^j - p_{13}^j \end{bmatrix} \begin{bmatrix} X_i \\ Y_i \\ Z_i \\ W_i \end{bmatrix} = \mathbf{0}$$

Triangulation

- 1 cameras, 2 equations 4 unknowns, underconstrained
- 2 cameras, 4 equations 4 unknowns, homogeneous only trivial solution

2. SVD Triangulation, in homogeneous

$$\begin{bmatrix} x_i^j \\ y_i^j \\ w_i^j \end{bmatrix} = \begin{bmatrix} p_{00}^j & p_{01}^j & p_{02}^j & p_{03}^j \\ p_{10}^j & p_{11}^j & p_{12}^j & p_{13}^j \\ p_{20}^j & p_{21}^j & p_{22}^j & p_{23}^j \end{bmatrix} \begin{bmatrix} X_i \\ Y_i \\ Z_i \\ W_i \end{bmatrix} = \begin{bmatrix} \mathbf{p}_0^j \mathbf{X}_i \\ \mathbf{p}_1^j \mathbf{X}_i \\ \mathbf{p}_2^j \mathbf{X}_i \end{bmatrix}$$

Right cross product $\begin{bmatrix} x_i^j \\ y_i^j \\ w_i^j \end{bmatrix} \times$ yields 3 eq, only 2 independent $\begin{bmatrix} 0 \\ 0 \\ 0 \end{bmatrix} = \begin{bmatrix} y_i^j \mathbf{p}_2^j \mathbf{X}_i - w_i^j \mathbf{p}_1^j \mathbf{X}_i \\ -x_i^j \mathbf{p}_2^j \mathbf{X}_i + w_i^j \mathbf{p}_0^j \mathbf{X}_i \\ x_i^j \mathbf{p}_1^j \mathbf{X}_i - y_i^j \mathbf{p}_0^j \mathbf{X}_i \end{bmatrix}$

$$x_i^j p_{20}^j X_i + x_i^j p_{21}^j Y_i + x_i^j p_{22}^j Z_i + x_i^j p_{23}^j W_i - \; w_i^j p_{00}^j X_i + w_i^j p_{01}^j Y_i + w_i^j p_{02}^j Z_i + w_i^j p_{03}^j W_i = 0$$

$$y_i^j p_{20}^j X_i + y_i^j p_{21}^j Y_i + y_i^j p_{22}^j Z_i + y_i^j p_{23}^j W_i - \; w_i^j p_{00}^j X_i + w_i^j p_{01}^j Y_i + w_i^j p_{02}^j Z_i + w_i^j p_{03}^j W_i = 0$$

$$\begin{bmatrix} p_{20}^j x_i^j - p_{00}^j w_i^j & p_{21}^j x_i^j - p_{01}^j w_i^j & p_{22}^j x_i^j - p_{02}^j w_i^j & p_{23}^j x_i^j - p_{03}^j w_i^j \\ p_{20}^j y_i^j - p_{10}^j w_i^j & p_{21}^j y_i^j - p_{11}^j w_i^j & p_{22}^j y_i^j - p_{12}^j w_i^j & p_{23}^j y_i^j - p_{13}^j w_i^j \end{bmatrix} \begin{bmatrix} X_i \\ Y_i \\ Z_i \\ W_i \end{bmatrix} = \mathbf{0}$$

Triangulation

1 cameras, 2 equations 4 unknowns, underconstrained

2 cameras, 4 equations 4 unknowns, homogeneous only trivial solution

2. SVD Triangulation

Assembling a matrix with the $2m$ equations resulting from m observations:

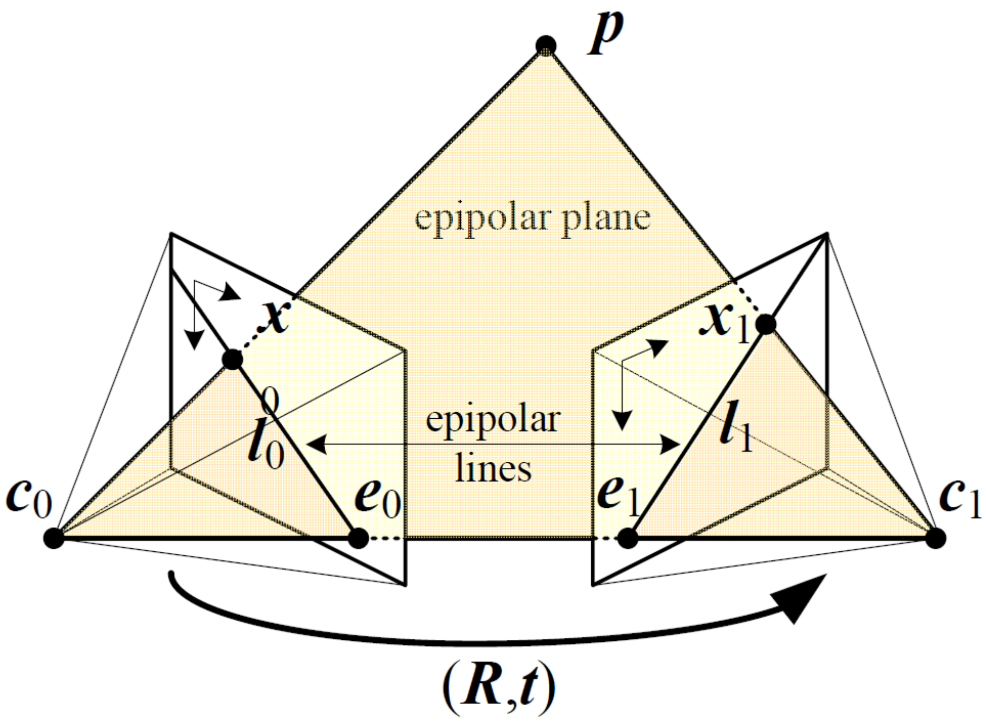
$$\mathbf{A}_{2m\times 4}\begin{bmatrix} X_i \\ Y_i \\ Z_i \\ W_i \end{bmatrix} = \mathbf{0}$$

$\operatorname{rank}(\mathbf{A}) \cong 3$ otherwise the points does not fulfill the epipolar constraint!

$$\begin{bmatrix} X_i \\ Y_i \\ Z_i \\ W_i \end{bmatrix} = \mathbf{v}_4 \,, \quad \mathbf{V} = \begin{bmatrix} \mathbf{v}_1 & \mathbf{v}_2 & \mathbf{v}_3 & \mathbf{v}_4 \end{bmatrix}$$

$$[\mathbf{U}, \mathbf{S}, \mathbf{V}] = \operatorname{svd}(\mathbf{A})$$

3 Motion from E

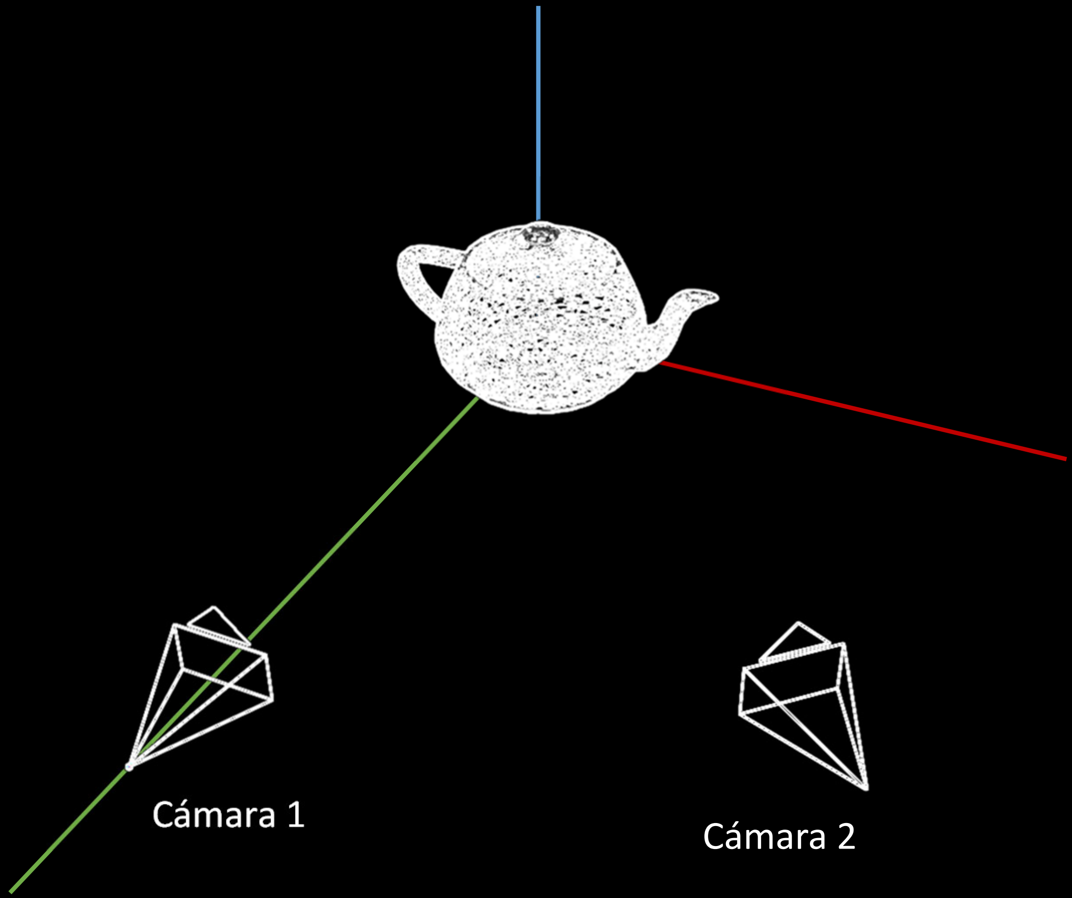


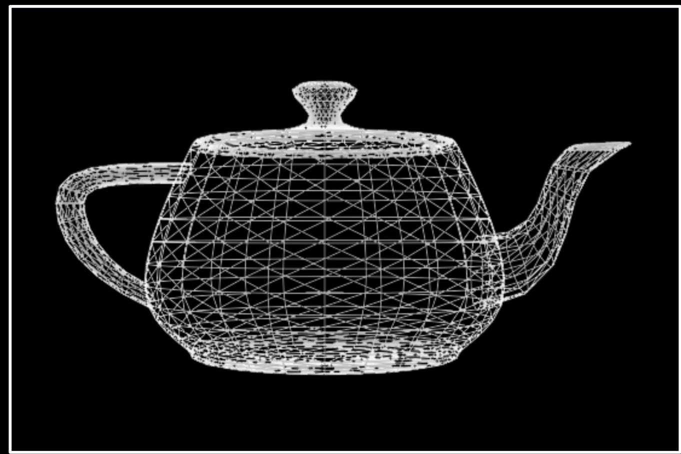
$$\boldsymbol{R} = \boldsymbol{R}_{c1c0}$$
$$\boldsymbol{t} = \boldsymbol{t}_{c1c0}$$

$$\boldsymbol{E} = [\boldsymbol{t}]_{\times}\boldsymbol{R} = \begin{bmatrix} 0 & -t_z & t_y \\ t_z & 0 & -t_x \\ -t_y & t_x & 0 \end{bmatrix}\boldsymbol{R}$$

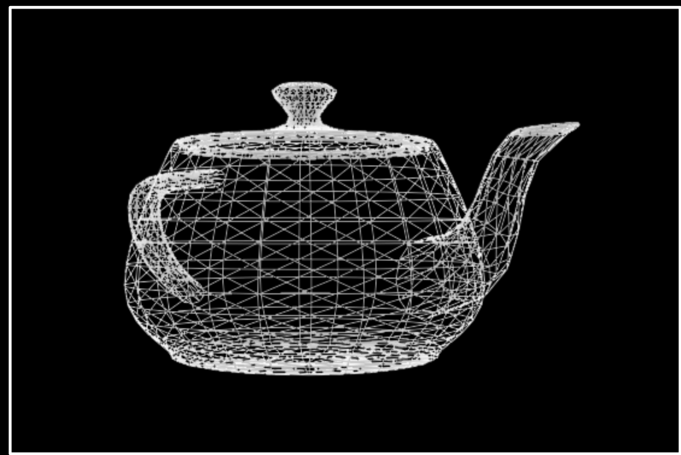
Given $\boldsymbol{E} \overset{??}{\Rightarrow} \boldsymbol{R}, \boldsymbol{t}$ Yes!

Efecto de la escala en la información geométrica





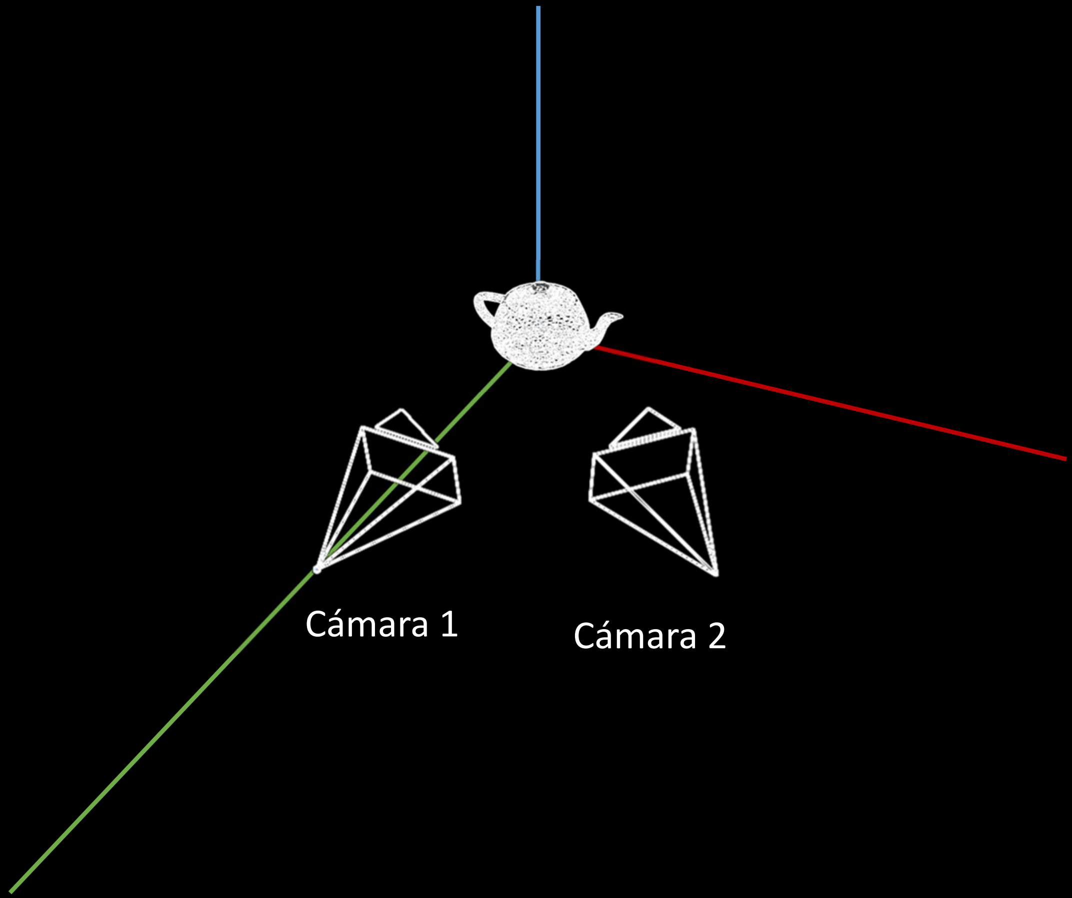
Cámara 1

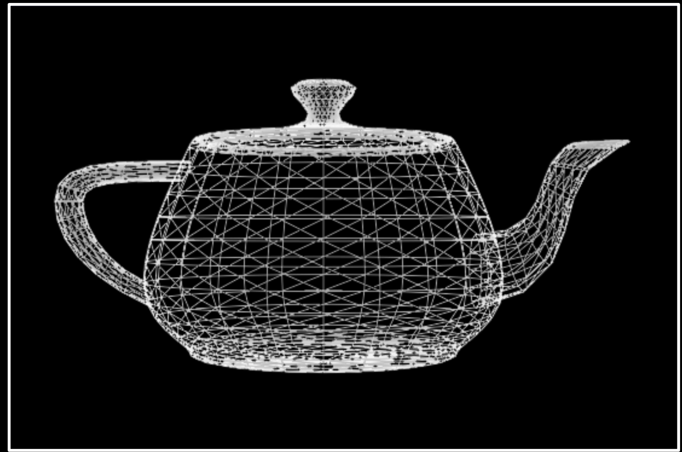


Cámara 2

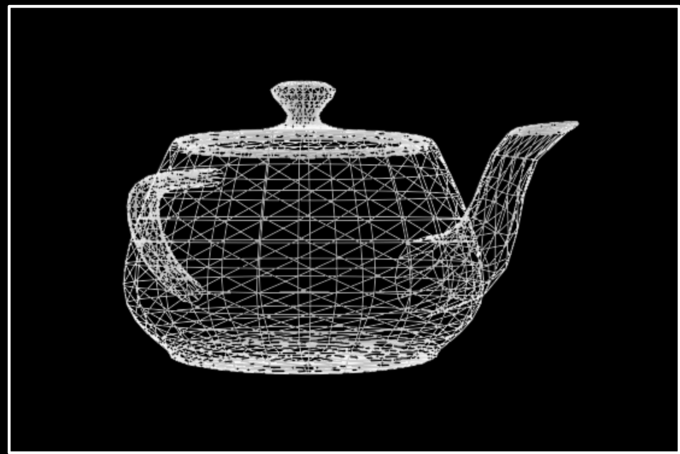
Tanks to Raul Iranzo

Efecto de la escala en la información geométrica





Cámara 1



Cámara 2

Tanks to Raul Iranzo

3. Motion from E

$$[\mathbf{U}, \boldsymbol{\Sigma}, \mathbf{V}] = \mathrm{svd}(\boldsymbol{E})$$

$$\mathbf{E} \cong \mathbf{U}\begin{bmatrix}1 & 0 & 0\\0 & 1 & 0\\0 & 0 & 0\end{bmatrix}\mathbf{V}^T$$

$$\mathbf{E} \cong [\mathbf{u}_1 \quad \mathbf{u}_2 \quad \mathbf{u}_3]\begin{bmatrix}1 & 0 & 0\\0 & 1 & 0\\0 & 0 & 0\end{bmatrix}\begin{bmatrix}\mathbf{v}_1^T\\\mathbf{v}_2^T\\\mathbf{v}_3^T\end{bmatrix}$$

4 solutions

$\mathbf{R}_{+90}, \mathbf{t};\ \mathbf{R}_{+90}, -\mathbf{t};\ \mathbf{R}_{-90}, \mathbf{t};\ \mathbf{R}_{-90}, -\mathbf{t}$

Where:

$\boldsymbol{R}_{+90} = \pm \boldsymbol{U}\boldsymbol{W}\boldsymbol{V}^T \quad \boldsymbol{R}_{-90} = \pm \boldsymbol{U}\boldsymbol{W}^T\boldsymbol{V}^T$

Sign such as $|\boldsymbol{R}_{+90}| > 0$, $|\boldsymbol{R}_{-90}| > 0$

$$\mathbf{t} = \mathbf{u}_3 \qquad \boldsymbol{W} = \begin{bmatrix}0 & -1 & 0\\1 & 0 & 0\\0 & 0 & 1\end{bmatrix}$$

11.3.1 Eight, seven, and five-point algorithms. Recovering t and R. Szeliski 2022. 2nd Edition.

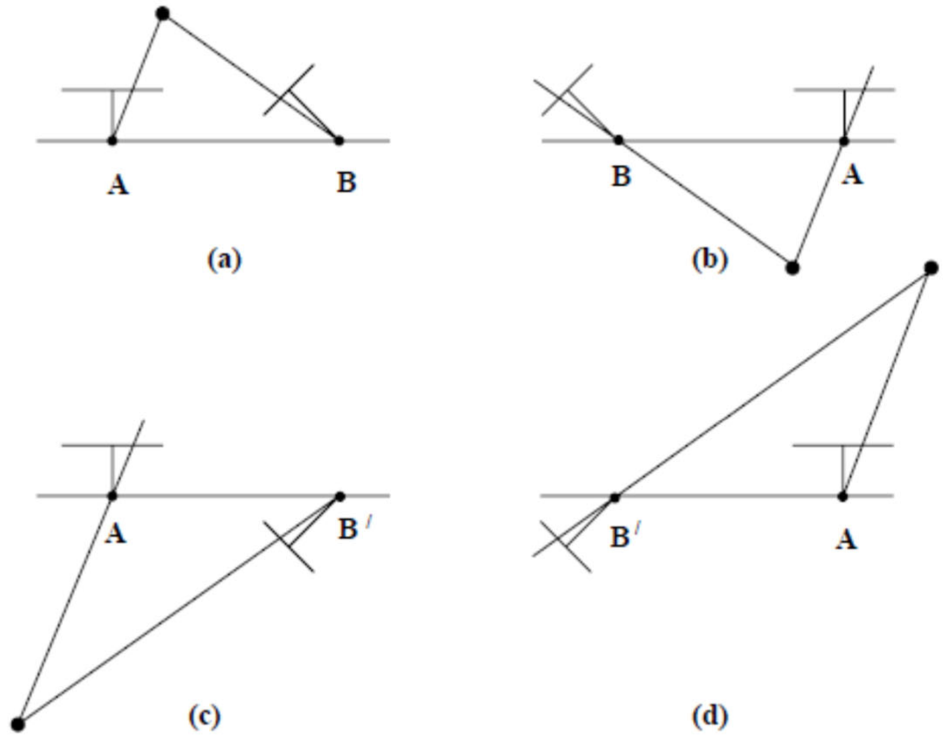


Fig. 9.12. **The four possible solutions for calibrated reconstruction from E.** *Between the left and right sides there is a baseline reversal. Between the top and bottom rows camera B rotates 180° about the baseline. Note, only in (a) is the reconstructed point in front of both cameras.*

4 motion solutions yield the same $\boldsymbol{E}$, only one of them yields 3D points in front of **both** cameras (chirality)

Universidad Zaragoza 1542

4 Structure from Motion 8 points

Input: set of $n \geq 8$ matches ***in pixels*** $\{\mathbf{u}_i^1, \mathbf{u}_i^2\}$ $i = 1..n$

cameras calibration , $\mathbf{K}^2$,$\mathbf{K}^1$

Output:

camera motion: $\mathbf{R}, \mathbf{t}$ *(up to scale)*

3D pose n points $\{\mathbf{X}_i\}$ $i = 1..n$*(up to scale)*

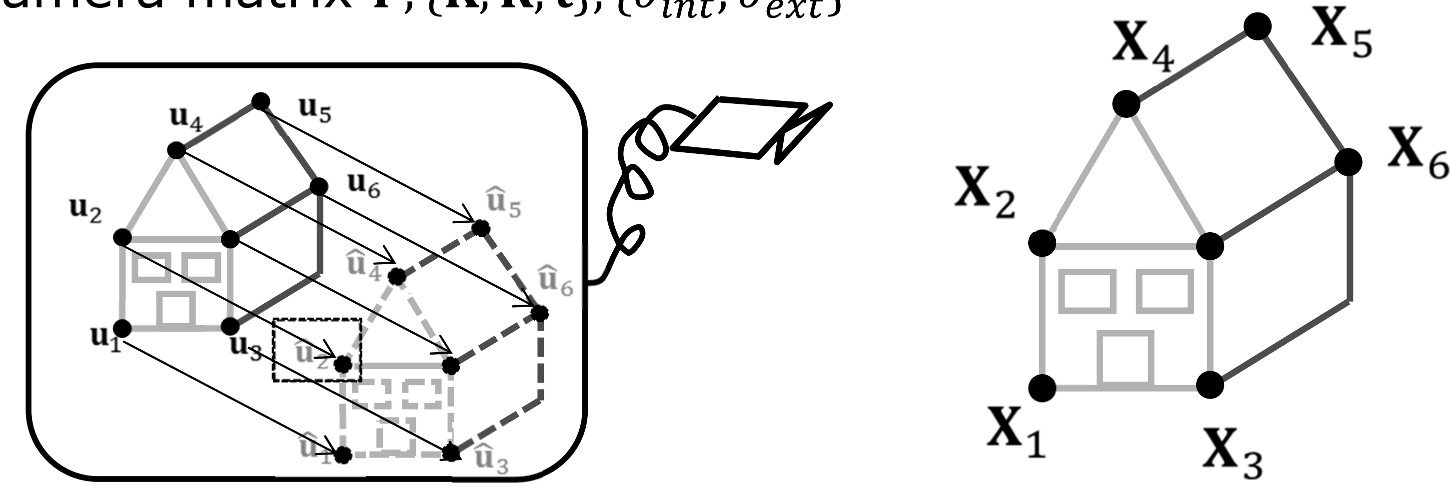
1. *Compute* $\mathbf{F}$ *matrix from point matches*
2. *Compute* $\mathbf{E} = (\mathbf{K}^2)^T \mathbf{F} \mathbf{K}^1$
3. *Recover the camera matrices & triangulate points for 4 solutions*
   - *a)* $\mathbf{P}^1 = \mathbf{K}^1[\mathbf{I}_{3\times3}|\mathbf{0}]$ $\quad \mathbf{P}^2 = \mathbf{K}^2[\mathbf{R}_{+90}|\mathbf{t}]$
   - *b)* $\mathbf{P}^1 = \mathbf{K}^1[\mathbf{I}_{3\times3}|\mathbf{0}]$ $\quad \mathbf{P}^2 = \mathbf{K}^2[\mathbf{R}_{+90}|-\mathbf{t}]$
   - *c)* $\mathbf{P}^1 = \mathbf{K}^1[\mathbf{I}_{3\times3}|\mathbf{0}]$ $\quad \mathbf{P}^2 = \mathbf{K}^2[\mathbf{R}_{-90}|\mathbf{t}]$
   - *d)* $\mathbf{P}^1 = \mathbf{K}^1[\mathbf{I}_{3\times3}|\mathbf{0}]$ $\quad \mathbf{P}^2 = \mathbf{K}^2[\mathbf{R}_{-90}|-\mathbf{t}]$
4. *Select the solution with more points in front of the two cameras*

2D-2D matches, calibration $\Rightarrow$
Camera poses + 3D points

5. Camera pose

Input: $\{\mathbf{u}_i \ , \mathbf{X}_i\}_{i=1}^{n}$ n 2D-3D matches,
2D images, of 3D points

Output: Camera matrix $\mathbf{P}, \{\mathbf{K}, \mathbf{R}, \mathbf{t}\}, \{\theta_{int}, \theta_{ext}\}$



https://youtu.be/iIhN3c7RjCI

$\mathbf{P}$ Precise description of the camera took the photo
$\{\mathbf{K}, \mathbf{R}, \mathbf{t}\}, \{\theta_{int}, \theta_{ext}\}$ Camera pose and calibration

Vista posterior de la Basílica del Pilar de Zaragoza 1886. Public Domain.
https://commons.wikimedia.org/w/index.php?curid=24970366

Universidad Zaragoza

5. DLT camera matrix $\mathbf{P}$

Recovers, the $\mathbf{P}^j$ matrix

$$\begin{bmatrix} s_i^j x_i^j \\ s_i^j y_i^j \\ s_i^j \end{bmatrix} = \begin{bmatrix} p_{00}^j & p_{01}^j & p_{02}^j & p_{03}^j \\ p_{10}^j & p_{11}^j & p_{12}^j & p_{13}^j \\ p_{20}^j & p_{21}^j & p_{22}^j & p_{23}^j \end{bmatrix} \begin{bmatrix} X_i \\ Y_i \\ Z_i \\ W_i \end{bmatrix} \qquad s_i^j = p_{20}^j X_i + p_{21}^j Y_i + p_{22}^j Z_i + p_{23}^j W_i$$

Substituting s_i^j in the first and second rows

$$p_{20}^j x_i^j X_i + p_{21}^j x_i^j Y_i + p_{22}^j x_i^j Z_i + p_{23}^j x_i^j W_i = p_{00}^j X_i + p_{01}^j Y_i + p_{02}^j Z_i + p_{03}^j W_i$$
$$p_{20}^j y_i^j X_i + p_{21}^j y_i^j Y_i + p_{22}^j y_i^j Z_i + p_{23}^j y_i^j W_i = p_{10}^j X_i + p_{11}^j Y_i + p_{12}^j Z_i + p_{13}^j W_i$$

Now, the unknowns are the camera matrix parameters

5. DLT camera matrix $\mathbf{P}$

Each 2D-3D $\{x_i \quad y_i \quad X_i \quad Y_i \quad Z_i \quad W_i\}$match provides 2 equations, if 6 or more 2D-3D matches available, $\mathbf{P}$ can be computed.

$$\begin{bmatrix} -X_i & -Y_i & -Z_i & -W_i & 0 & 0 & 0 & 0 & x_iX_i & x_iY_i & x_iZ_i & x_iW_i \\ 0 & 0 & 0 & 0 & -X_i & -Y_i & -Z_i & -W_i & y_iX_i & y_iY_i & y_iZ_i & y_iW_i \end{bmatrix} \begin{bmatrix} p_{00} \\ p_{01} \\ p_{02} \\ p_{03} \\ p_{10} \\ p_{11} \\ p_{12} \\ p_{13} \\ p_{20} \\ p_{21} \\ p_{22} \\ p_{23} \end{bmatrix}$$

Assembling the $2n$ equations we obtain the matrix $\mathbf{A}_{2n\times12}$

$$[\mathbf{U}, \mathbf{S}, \mathbf{V}] = \mathrm{svd}(\mathbf{A})$$

The solution is the 12th column of $\mathbf{V}$

Szeliski 2022. Second Edition. 11.2 Pose Estimation.
Hartley, Zisserman 2004, 7. Computation of the Camera Matrix P

5. DLT camera matrix $\mathbf{P}$

We can consider 2D-3D matches including points at infinity:

$$\{x_i \quad y_i \quad w_i \quad X_i \quad Y_i \quad Z_i \quad W_i\}$$

if 6 or more 2D-3D matches available, $\mathbf{P}$ can be computed.

$$\begin{bmatrix} -w_iX_i & -w_iY_i & -w_iZ_i & -w_iW_i & 0 & 0 & 0 & 0 & x_iX_i & x_iY_i & x_iZ_i & x_iW_i \\ 0 & 0 & 0 & 0 & -w_iX_i & -w_iY_i & -w_iZ_i & -w_iW_i & y_iX_i & y_iY_i & y_iZ_i & y_iW_i \end{bmatrix} \begin{bmatrix} p_{00} \\ p_{01} \\ p_{02} \\ p_{03} \\ p_{10} \\ p_{11} \\ p_{12} \\ p_{13} \\ p_{20} \\ p_{21} \\ p_{22} \\ p_{23} \end{bmatrix}$$

Assembling the $2n$ equations we obtain the matrix $\mathbf{A}_{2n\times 12}$

$$[\mathbf{U}, \mathbf{S}, \mathbf{V}] = \mathrm{svd}(\mathbf{A})$$

The solution is the 12th column of $\mathbf{V}$

Universidad Zaragoza 1542

Szeliski 2022 2nd Edition, 11.2 Pose Estimation.
Hartley, Zisserman 2004, 7. Computation of the Camera Matrix P

5. Decomposing P

1. $\mathbf{P} = \left[\mathbf{M} | -\mathbf{M}\tilde{\mathbf{C}}\right] = \mathbf{K}[\mathbf{R}_{\boldsymbol{cw}} | -\mathbf{R}_{\boldsymbol{cw}}\mathbf{t}_{\boldsymbol{wc}}]$
2. Computing the optical centre pose in world frame

$$\mathbf{PC} = 0 \quad \text{(solving using svd)}$$

$$\mathbf{C} = \lambda[\mathbf{t}_{\boldsymbol{wc}}\ 1]$$

3. decomposition of $\mathbf{M}$

$$\widehat{\mathbf{M}} = \mathrm{sign}(\det(\mathbf{M}))\mathbf{M}$$

$$\widehat{\mathbf{M}} = \widehat{\mathbf{K}}\widehat{\mathbf{R}} \quad \text{RQ decomposition (np.linalg.rq)}$$

4. $\mathbf{D} = \mathrm{diag}(\mathrm{sign}(k_{11}), \mathrm{sign}(k_{22}), \mathrm{sign}(k_{33}))$
5. $\mathbf{R}_{\boldsymbol{cw}} = \mathbf{D}\widehat{\mathbf{R}}$ $\quad \mathbf{K} = \frac{\mathbf{1}}{\bar{\boldsymbol{k}}_{\mathbf{33}}}\overline{\mathbf{K}}$ $\quad \overline{\mathbf{K}} = \mathbf{D}\widehat{\mathbf{K}}$
6. Using opencv

$\overline{\mathbf{K}}$, $\mathbf{R}_{\boldsymbol{cw}}$, $\mathbf{t}_{\boldsymbol{wc}}$ =cv.decomposeProjectionMatix $(\mathrm{sign}(\det(\mathbf{M})) * \mathbf{P})$

Hartley, Zisserman 2004, 6.2.4 Decomposition of the Camera Matrix

Trym Vegard Haavardsholm, Pose from known 3D points. (26/10/2021) https://www.uio.no/studier/emner/matnat/its/nedlagte-emner/UNIK4690/v16/forelesninger/lecture_5_2_pose_from_known_3d_points.pdf

5. Perspective-nPoints-Problem PnP

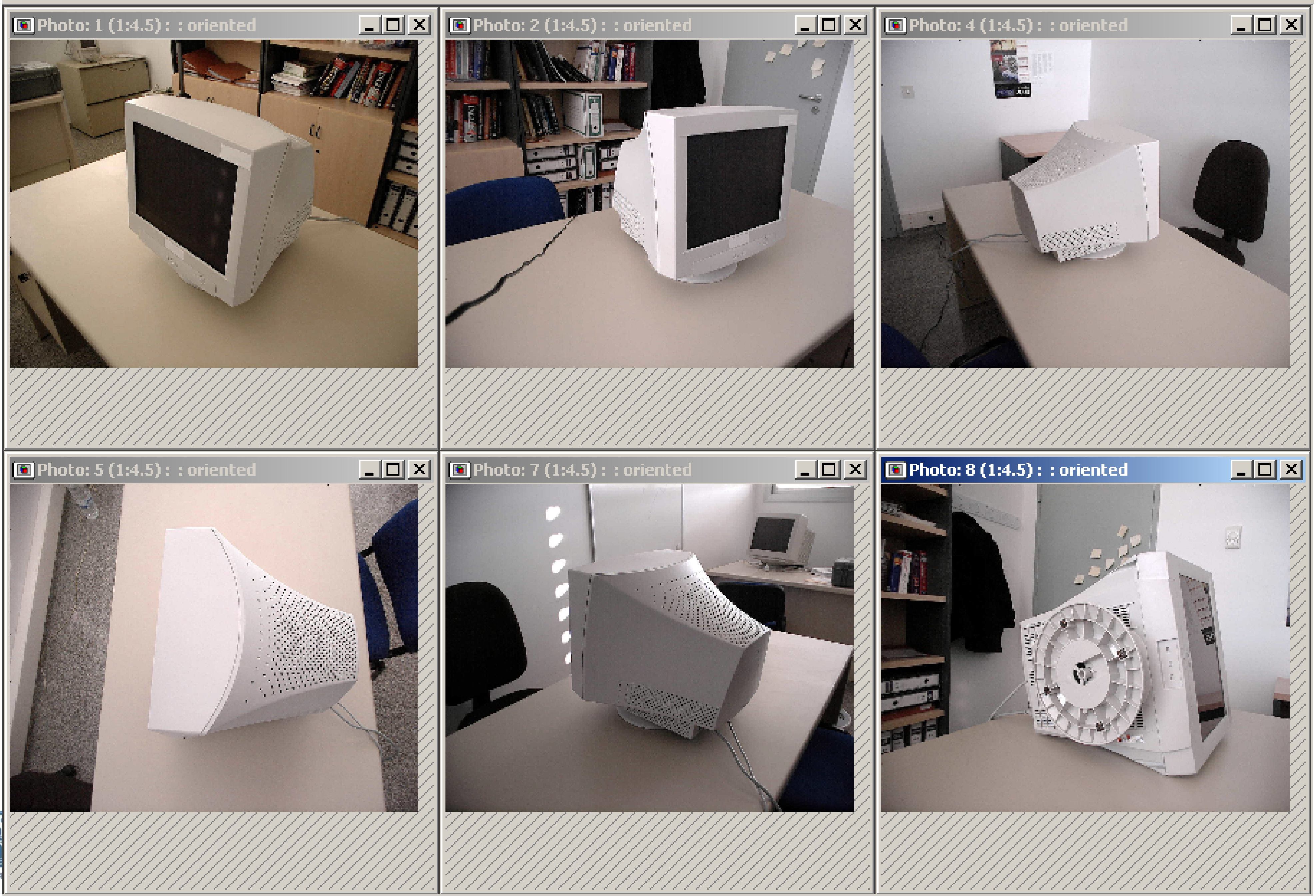
1. Exploit the available camera calibration and only compute the camera pose (6 d.o.f)
2. ‘n’ Defines how many points are used
    1. P3P, stands for calibrated camera pose from 3 3D-2D correspondences
    2. P4P, stands for calibrated camera pose from 4 3D-2D correspondences
3. P3P, algorithms [Ga0 2003] are available but they do not provide unique solution
4. P4P, can provide unique solutions. [Lepetit 2009] PnP, operates in linear time, reduced sensitivity. Interesting for RANSAC because can propose solutions just from 4 points
5. PnP $n \geq 4$ more costly but able to reduce sensitivity with respect to noise. [Lepetit 2009] efficient algorithm.

Lepetit, V., Moreno-Noguer, F., Fua, P. (2009). EPnP: An accurate O(n) solution to the PnP problem. *International journal of computer vision*, *81*(2), 155-166.

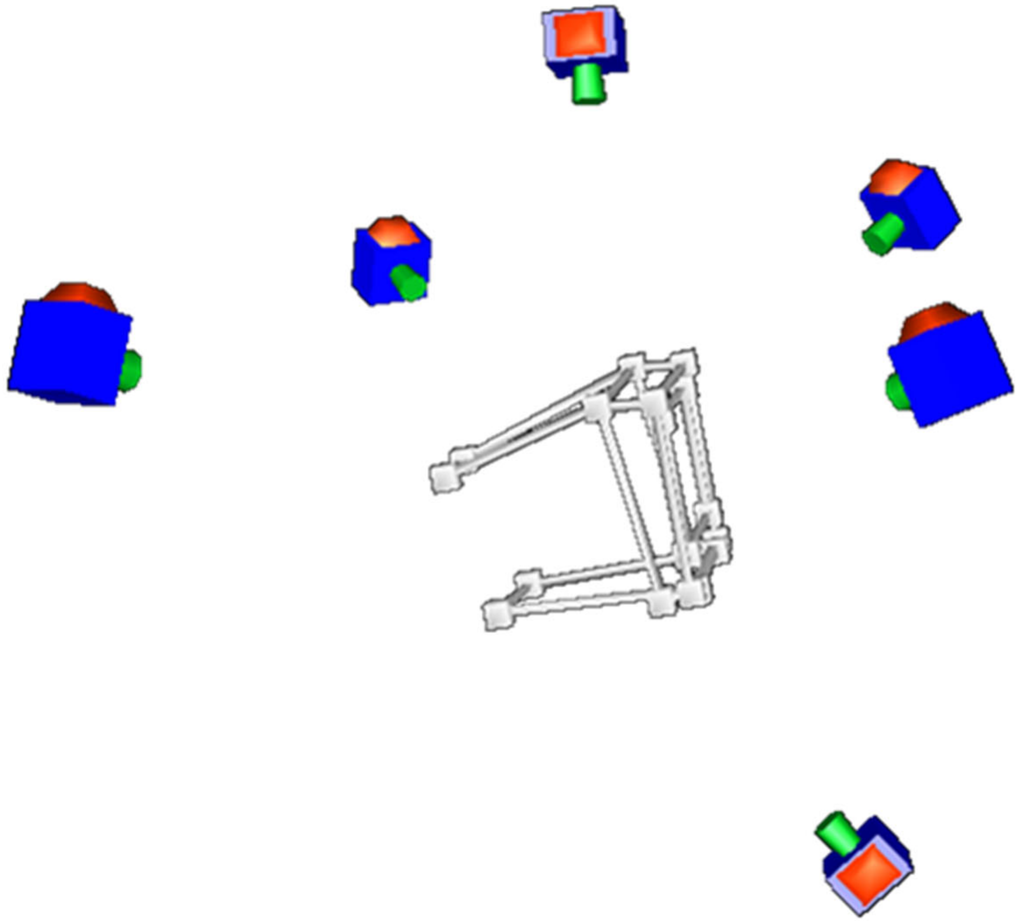
Gao, X. S., Hou, X. R., Tang, J., & Cheng, H. F. (2003). Complete solution classification for the perspective-three-point problem. *IEEE PAMI*, *25*(8), 930–943.

Two view geometry estimation from point matches					
	Matches	Camera pose	3D structure	scale	minimum # matches
Triangulation	2D-2D	Known	Estimated	Known, same as cameras	1
PnP, camera pose	2D-3D	Estimated	Known	Known, same as structure	3, 4, non-linear 5 linear
SfM	2D-2D	Estimated	Estimated	Up-to-scale	8 linear 5 non-linear

6.1 Bundle adjustment. Intro

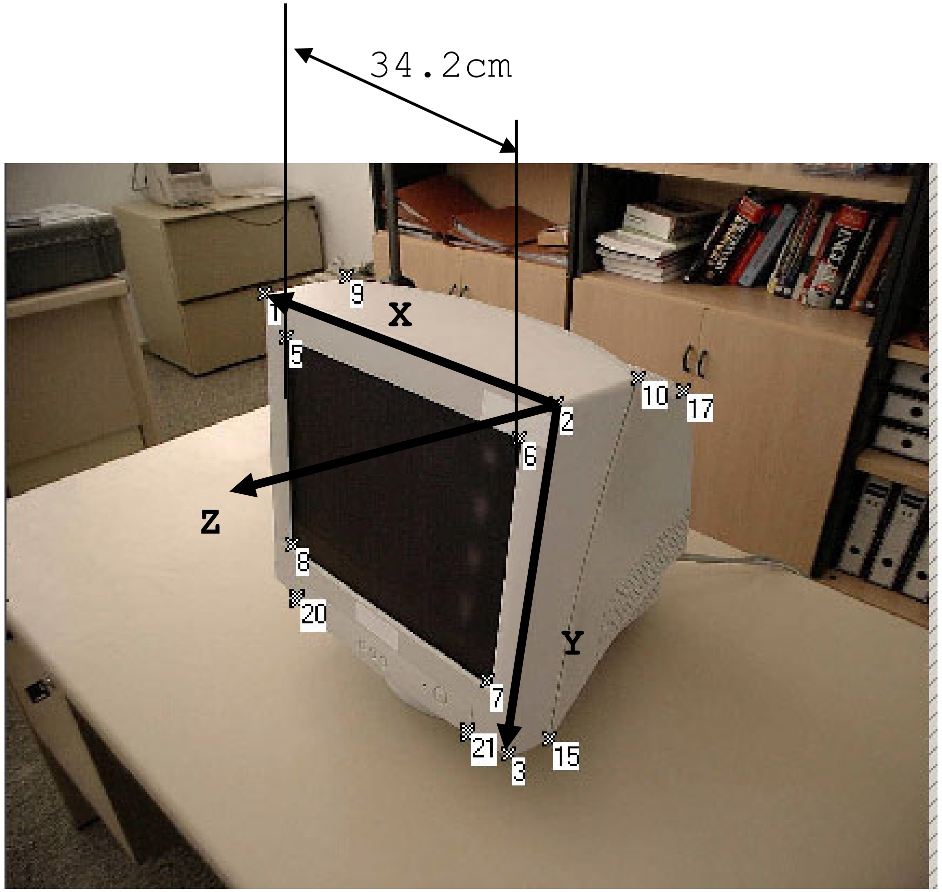


6.1 Bundle adjustment. Intro



- Camera poses and 3D point poses:
  - Unknown scale factor. Ambiguity: close small object vs far big object.
  - The absolute pose is unknwow, relative poses.
- Scale and absolute poses ***cannot be estimated, they are unobservable.***

6.1 Bundle adjustment. Intro



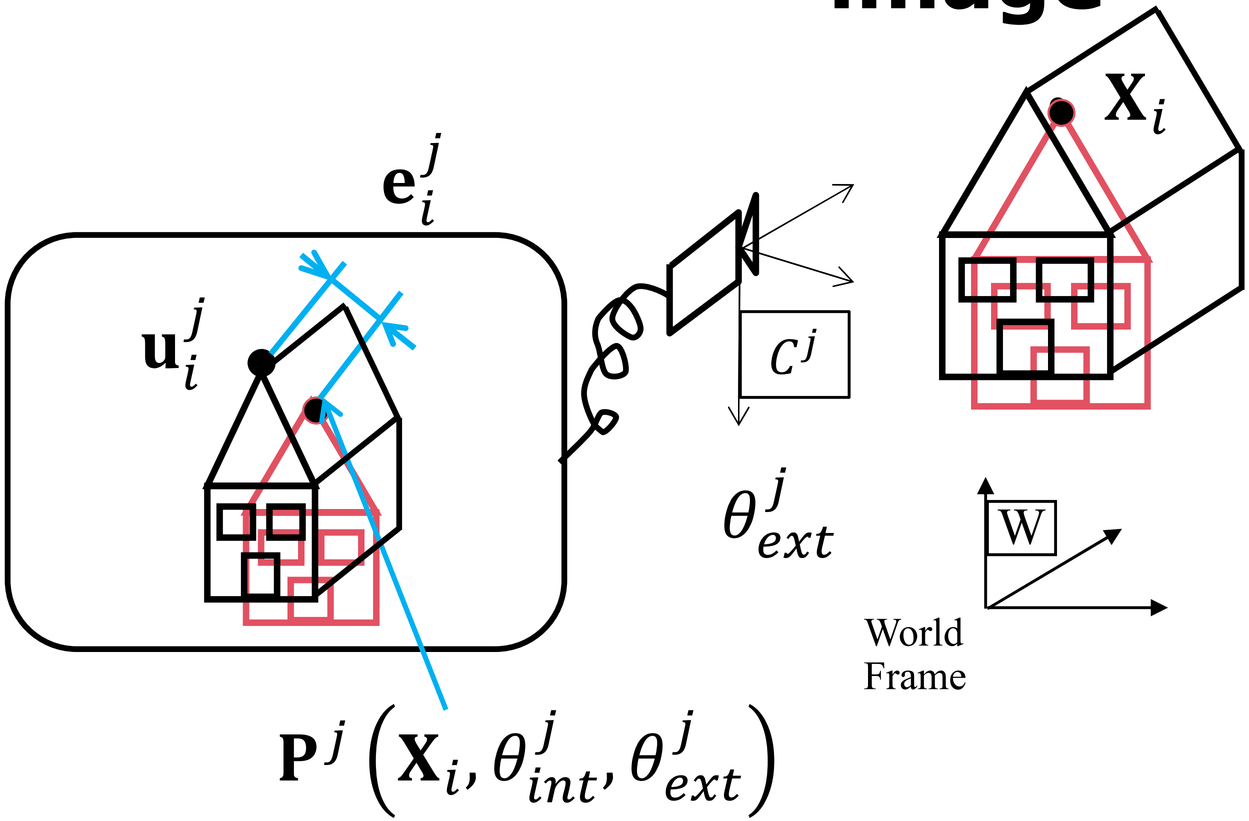
Scale recovery:
- Distance between two scene points.
- Distance between two camera poses.

Absolute pose:
- Known camera pose.

Absolute pose+scale:
- 3 or more 3D points with known 3D poses [Horn 87]

Horn, B. K. (1987). Closed-form solution of absolute orientation using unit quaternions. *Josa a*, *4*(4), 629-642.

6.2 Observation of a map point in an image



Reprojection error (pixels)

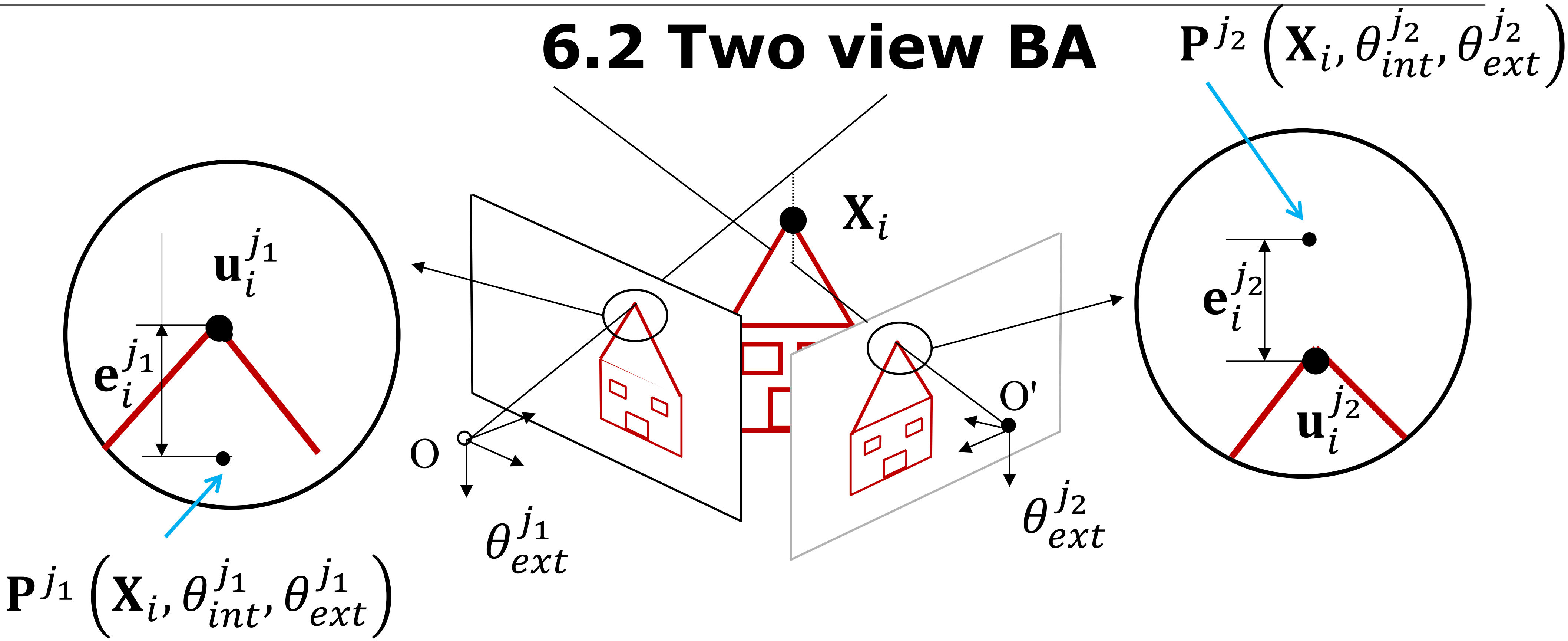
Projection Function

$$\mathbf{e}_i^j = \mathbf{u}_i^j - \mathbf{P}^j\left(\mathbf{X}_i, \theta_{int}^j, \theta_{ext}^j\right)$$

$\mathbf{X}_i \in \mathcal{R}^3$ coordinates of the i-th 3D point

$\theta_{ext}^j \in \mathrm{SE}(3)$ 6 d.o.f pose of the j-th camera

6.2 Two view BA



Reconstructed point $\mathbf{X}_i$, if reprojected in the images j_1, j_2, yields $\mathbf{e}_i^{j_2}$, $\mathbf{e}_i^{j_1}$ non-zero reprojection error.

Solution: bundle adjustment, reprojection error minimization

$$\underset{\{\theta_{ext}^j, \mathbf{X}_i\}}{\operatorname{argmin}} \sum_{i,j} \left\|\mathbf{e}_i^j\right\|^2 = \underset{\{\theta_{ext}^j, \mathbf{X}_i\}}{\operatorname{argmin}} \sum_{i,j} \left\|\mathbf{u}_i^j - \mathbf{P}^j\left(\mathbf{X}_i, \theta_{int}^j, \theta_{ext}^j\right)\right\|^2$$

6.3 Non linear optimization

$$\underset{\{\theta_{ext}^{j},\mathbf{X}_i\}}{\operatorname{argmin}} \sum_{i,j} \left\| \mathbf{u}_i^j - \mathbf{P}^j\left(\mathbf{X}_i, \theta_{int}^j, \theta_{ext}^j\right) \right\|^2$$

Non-linear optimization difficulties:

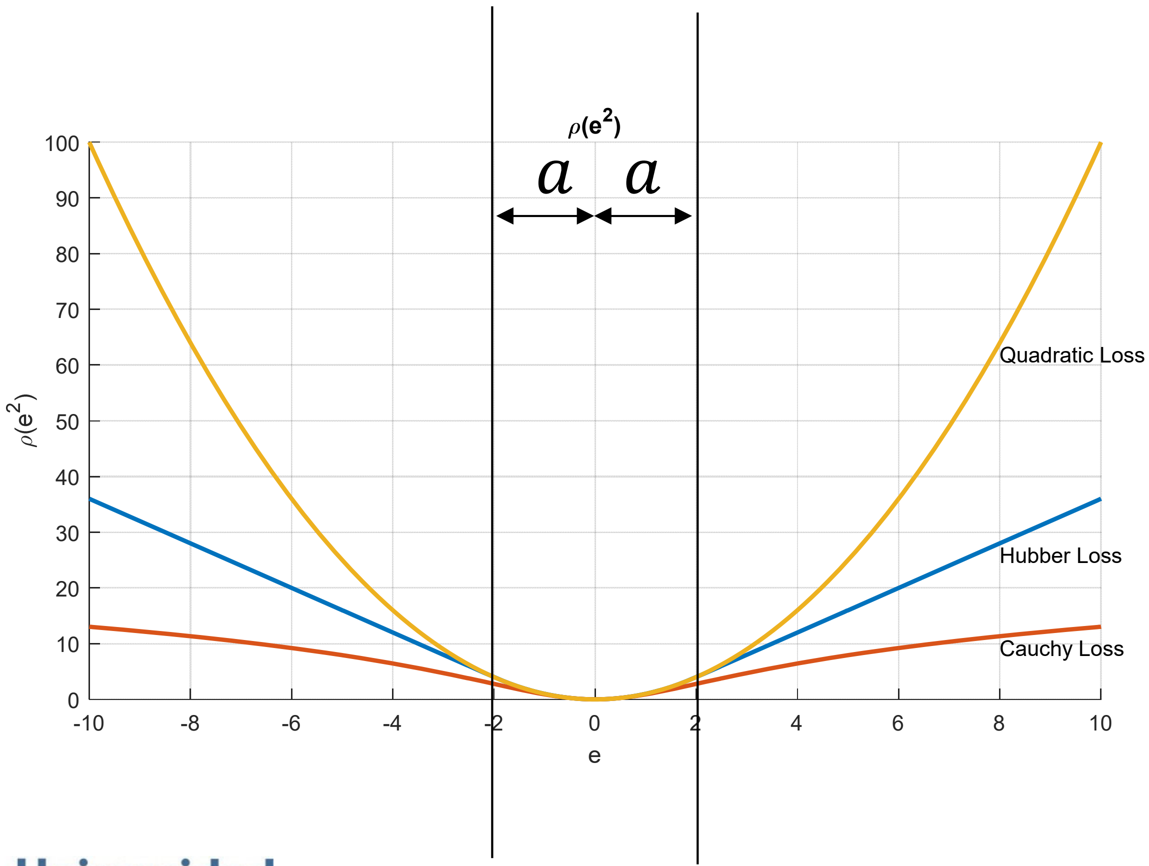
1. Failure in the presence of outliers, i.e. false positive matches.
2. Expensive iterative computation.
3. Multiple local minima.
4. Initial guess is a must.

If an accurate initial guess is available :

1. Outliers can be handle with robust influence function
2. Few iterations ⇨ absolute minimum
3. Levenberg-Marquardt,
   g2o, Ceres, lsqnonlin (Matlab) , scipy.optimize.least_squares (Python)

6.3 Robust influence function

$$\underset{\{\theta_{ext}^j, \mathbf{X}_i\}}{\operatorname{argmin}} \sum_{i,j} \left\| \mathbf{u}_i^j - \mathbf{P}^j\left(\mathbf{X}_i, \theta_{int}^j, \theta_{ext}^j\right) \right\|^2 \Rightarrow \underset{\{\theta_{ext}^j, \mathbf{X}_i\}}{\operatorname{argmin}} \sum_{i,j} \rho\left(\left\| \mathbf{u}_i^j - \mathbf{P}^j\left(\mathbf{X}_i, \theta_{int}^j, \theta_{ext}^j\right) \right\|^2 \right)$$



$\rho(\cdot)$, robust influence function to alleviate the effect of gross outliers, a robust threshold

Huber

$$\rho(s) = \begin{cases} s & s \leq a^2 \\ 2a^2\sqrt{\dfrac{s}{a^2}} - 1 & s > a^2 \end{cases}$$

Cauchy

$$\rho(s) = a^2\left(\log\left(\frac{s}{a^2}\right) + 1\right)$$

Non-linear optimization
Cost function derivatives

$$\underset{\mathbf{X}}{\operatorname{argmin}}\, \mathbf{f}(\mathbf{X})$$

Cost function $\mathbf{f}(\mathbf{X})$ $\qquad \dim(\mathbf{f}) = 1$

$\mathbf{X}$ estate to be estimated $\qquad \dim(\mathbf{X}) = \mathrm{n}$

$$\mathbf{f}(\mathbf{X} + \delta\mathbf{X}) \approx \mathbf{f}(\mathbf{X}) + \mathbf{g}^T \delta\mathbf{X} + \frac{1}{2} \delta\mathbf{X}^T \mathbf{H} \delta\mathbf{X}$$

Gradient

$$\mathbf{g} \equiv \frac{\mathrm{d}\mathbf{f}}{\mathrm{d}\mathbf{X}} = \begin{bmatrix} \frac{\partial \mathbf{f}}{\partial x_i} \\ \vdots \\ \frac{\partial \mathbf{f}}{\partial x_n} \end{bmatrix} \qquad n \times 1$$

Hessian

$$\mathbf{H} \equiv \frac{\mathrm{d}^2\mathbf{f}}{\mathrm{d}^2\mathbf{X}} = \begin{bmatrix} \frac{\partial^2 \mathbf{f}}{\partial x_1 \partial x_1} & \cdots & \frac{\partial^2 \mathbf{f}}{\partial x_1 \partial x_n} \\ \vdots & \ddots & \vdots \\ \frac{\partial^2 \mathbf{f}}{\partial x_n \partial x_1} & \cdots & \frac{\partial^2 \mathbf{f}}{\partial x_n \partial x_n} \end{bmatrix} \qquad n \times n$$

Universidad Zaragoza 1542

Non-linear optimzation. Steepest descendent method

Cost function $\mathbf{f}(\mathbf{X}+\delta\mathbf{X}) \approx \mathbf{f}(\mathbf{X}) + \mathbf{g}^T\delta\mathbf{X} \quad \dim(\mathbf{f}) = 1$

Steepest descendent step:

$$\delta\mathbf{X} = -\alpha\mathbf{g},$$
$$\mathbf{X} \leftarrow \mathbf{X} + \delta\mathbf{X}$$

1. Good performance far from the minimum ☺
2. Not easy to find the optimal step size α ☹
3. Slow convergence near the minimum ☹

Non-linear optimzation
Newton method

Cost function $\mathbf{f}(\mathbf{X}+\delta\mathbf{X}) \approx \mathbf{f}(\mathbf{X}) + \mathbf{g}^T\delta\mathbf{X} + \frac{1}{2}\delta\mathbf{X}^T\mathbf{H}\delta\mathbf{X} \quad \dim(\mathbf{f}) = 1$

Cost function derivative $\frac{\mathrm{d}\mathbf{f}}{\mathrm{d}\mathbf{X}}(\mathbf{X}+\delta\mathbf{X}) \approx \frac{\mathrm{d}\mathbf{f}}{\mathrm{d}\mathbf{X}}(\mathbf{X}) + \frac{\mathrm{d}^2\mathbf{f}}{\mathrm{d}^2\mathbf{X}}\delta\mathbf{X} = \mathbf{g} + \mathbf{H}\delta\mathbf{X}$

At the minimum

$$\frac{\mathrm{d}\mathbf{f}}{\mathrm{d}\mathbf{X}}(\mathbf{X}+\delta\mathbf{X}) = \mathbf{0} \quad \Rightarrow \mathbf{g} + \mathbf{H}\delta\mathbf{X} = 0$$

Newton step:

$$\delta\mathbf{X} = -\mathbf{H}^{-\mathbf{1}}\mathbf{g}, \qquad \text{we do not invert but solve } \mathbf{H}\delta\mathbf{X} = -\mathbf{g}$$

$$\mathbf{X} \leftarrow \mathbf{X} + \delta\mathbf{X}$$

1. Quadratic convergence if close to the minimum ☺
2. Hessian ($\mathbf{H}$) computation non trivial ☹
3. Solving linear $\mathbf{H}\delta\mathbf{X} = -\mathbf{g}$ expensive ☹
4. Overshoot far from the minimum ☹

Non-linear optimzation
Damped Newton Methods

Instead of solving:

$$\mathbf{H}\delta\mathbf{X} = -\mathbf{g}$$

Solve:

$$(\mathbf{H} + \lambda\mathbf{W})\delta\mathbf{X} = -\mathbf{g}$$

1. Small λ Newton method ☺
2. Big λ steepest descendent ☺
3. Problem: how to select λ ☹
   1. Levenberg-Marquardt (LM)
      1. Bad convergence: increase $\lambda \rightarrow$ Steepest descendent
      2. Good convergence : decrease $\lambda \rightarrow$ Newton direction
   2. Trust-Region methods: choose λ to limit the step to a maximum dynamic size

Non-linear least squares. Cost function

BA Cost function $\mathbf{f}(X) = \sum_{i,j} \left\| \mathbf{u}_i^j - \mathbf{P}^j \left(\mathbf{X}_i, \theta_{int}^j, \theta_{ext}^j \right) \right\|^2$

Standard non-linear least squares cost function

residual vector

predicted observations vector

$$\mathrm{f}(\mathbf{X}) = \frac{1}{2} \Delta\mathbf{Z}(\mathbf{X})^T \Delta\mathbf{Z}(\mathbf{X}), \qquad \Delta\mathbf{Z}(\mathbf{X}) = \underline{\mathbf{Z}} - \mathbf{Z}(\mathbf{X})$$

Observations vector

current state estimation vector

$$m = \dim(\mathbf{Z}), \; \mathrm{n} = \dim(\mathbf{X}),$$

$$\underline{\mathbf{Z}} = \left[\boldsymbol{u}_1^{1T}, \boldsymbol{u}_2^{1T} \cdots \boldsymbol{u}_{\boldsymbol{n_p}}^{1T}, \boldsymbol{u}_1^{2T}, \boldsymbol{u}_2^{2T} \cdots \boldsymbol{u}_{\boldsymbol{n_p}}^{2T}, \cdots, \boldsymbol{u}_1^{m_c T}, \boldsymbol{u}_2^{m_c T} \cdots \boldsymbol{u}_{\boldsymbol{n_p}}^{m_c T} \right]^T$$

$$\mathbf{Z}(\mathbf{X}) = \left[\boldsymbol{P}(\boldsymbol{X}_1, \boldsymbol{\theta}_{ext,}^1)^T, \boldsymbol{P}(\boldsymbol{X}_2, \boldsymbol{\theta}_{ext}^1)^T, \cdots, \boldsymbol{P}(\boldsymbol{X}_{\boldsymbol{n_p}}, \boldsymbol{\theta}_{ext}^1)^T, \cdots, \boldsymbol{P}(\boldsymbol{X}_{\boldsymbol{n_p}}, \boldsymbol{\theta}_{ext,}^{m_c})^T, \right]^T$$

$$\mathbf{X} = \left[\mathbf{X}_1^T, \mathbf{X}_2^T, \ldots, \mathbf{X}_{n_p}^T, \boldsymbol{\theta}_{ext}^{2\;T}, \boldsymbol{\theta}_{ext}^{3\;T}, \ldots, \boldsymbol{\theta}_{ext}^{m_c\;T} \right]^T$$

First camera not included!

Triggs et al. 2000. “Bundle adjustment–A modern synthesis“
Madsen et al. 2004 “Methods for Non-Linear Least Squares Problems”

Non-linear least squares optimization. Cost function derivatives

Cost function $\mathbf{f}(\mathbf{X}+\delta\mathbf{X}) \approx \mathbf{f}(\mathbf{X}) + \mathbf{g}^T\delta\mathbf{X} + \frac{1}{2}\delta\mathbf{X}^T\mathbf{H}\delta\mathbf{X} \quad \dim(\mathbf{f}) = 1$

$\mathrm{f}(\mathbf{X}) = \frac{1}{2}\Delta\mathbf{Z}(\mathbf{X})^T\Delta\mathbf{Z}(\mathbf{X}) \Longrightarrow \mathrm{f}(\mathbf{X}+\delta\mathbf{X}) \approx \mathbf{f}(\mathbf{X}) + \mathbf{J}\delta\mathbf{X}$

Gradient

$$\mathbf{g} \equiv \frac{\mathrm{d}\mathbf{f}}{\mathrm{d}\mathbf{X}} = \begin{bmatrix} \frac{\partial \mathbf{f}}{\partial x_i} \\ \vdots \\ \frac{\partial \mathbf{f}}{\partial x_n} \end{bmatrix} = -\Delta\mathbf{Z}^T\mathbf{J} \quad n \times 1$$

Hessian

$$\mathbf{H} \equiv \frac{\mathrm{d}^2\mathbf{f}}{\mathrm{d}^2\mathbf{X}} = \begin{bmatrix} \frac{\partial^2 \mathbf{f}}{\partial x_1 \partial x_1} & \cdots & \frac{\partial^2 \mathbf{f}}{\partial x_1 \partial x_n} \\ \vdots & \ddots & \vdots \\ \frac{\partial^2 \mathbf{f}}{\partial x_n \partial x_1} & \cdots & \frac{\partial^2 \mathbf{f}}{\partial x_n \partial x_n} \end{bmatrix} = \mathbf{J}^T\mathbf{J} + \sum_i \Delta z_i \frac{\mathbf{d}^2 z_i}{\mathbf{d}\mathbf{X}^2} \approx \mathbf{J}^T\mathbf{J} \quad n \times n$$

Jacobian

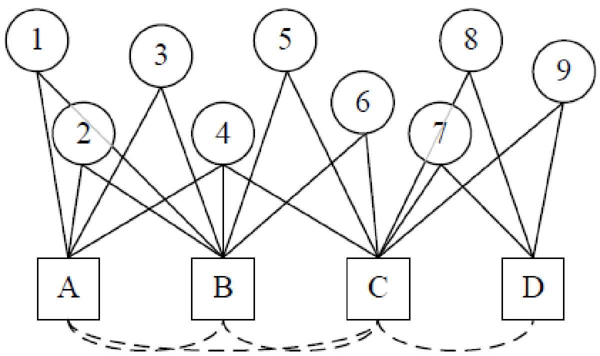
$$\mathbf{J} \equiv \frac{\mathrm{d}\mathbf{Z}}{\mathrm{d}\mathbf{X}} = \begin{bmatrix} \frac{\partial z_1}{\partial x_1} & \cdots & \frac{\partial z_1}{\partial x_n} \\ \vdots & \ddots & \vdots \\ \frac{\partial z_m}{\partial x_1} & \cdots & \frac{\partial z_m}{\partial x_n} \end{bmatrix} \quad m \times n$$

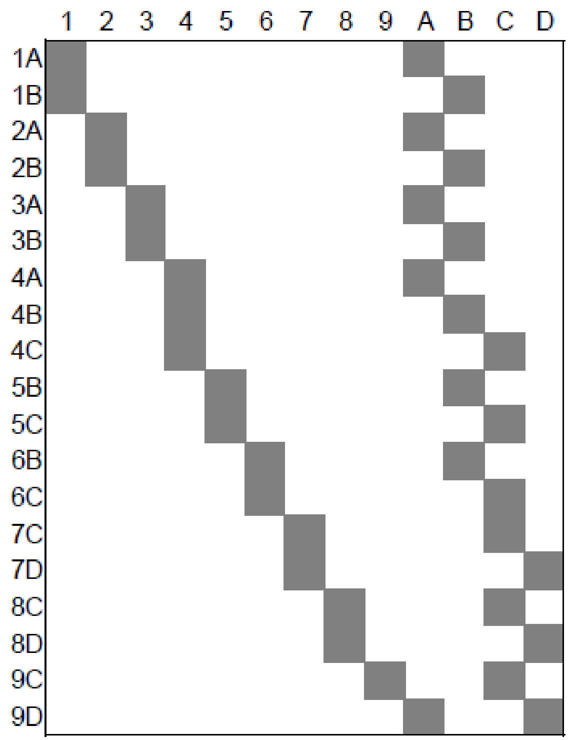
Non-linear least squares optimization

Method	Non-linear	Non-linear Least squares
Steepest descendent	$\delta\mathbf{X} = -\alpha\mathbf{g}$	$\delta\mathbf{X} = \alpha\Delta\mathbf{Z}^{T}\mathbf{J}$
Newton	$\mathbf{H}\delta\mathbf{X} = -\mathbf{g}$	
Gauss-Newton		$\mathbf{J}^{T}\mathbf{J}\delta\mathbf{X} = \Delta\mathbf{Z}^{T}\mathbf{J}$
Damped	$(\mathbf{H} + \lambda\mathbf{W})\delta\mathbf{X} = -\mathbf{g}$	$(\mathbf{J}^{T}\mathbf{J} + \lambda\mathbf{W})\delta\mathbf{X} = \Delta\mathbf{Z}^{T}\mathbf{J}$

Jacobian Sparsity Pattern

$$\mathbf{Z}(\mathbf{X}) = \left[\boldsymbol{P}(\boldsymbol{X}_1\,, \boldsymbol{\theta}^1_{ext,})^T, \boldsymbol{P}(\boldsymbol{X}_2\,, \boldsymbol{\theta}^1_{ext})^T, \cdots, \boldsymbol{P}(\boldsymbol{X}_{\boldsymbol{n_p}}, \boldsymbol{\theta}^1_{ext})^T, \cdots, \boldsymbol{P}(\boldsymbol{X}_{\boldsymbol{n_p}}, \boldsymbol{\theta}^{m_c}_{ext,})^T, \right]^T$$





Nodes

Cameras A,B,C,D

Points 1-9

Edges

Camera j detects point i

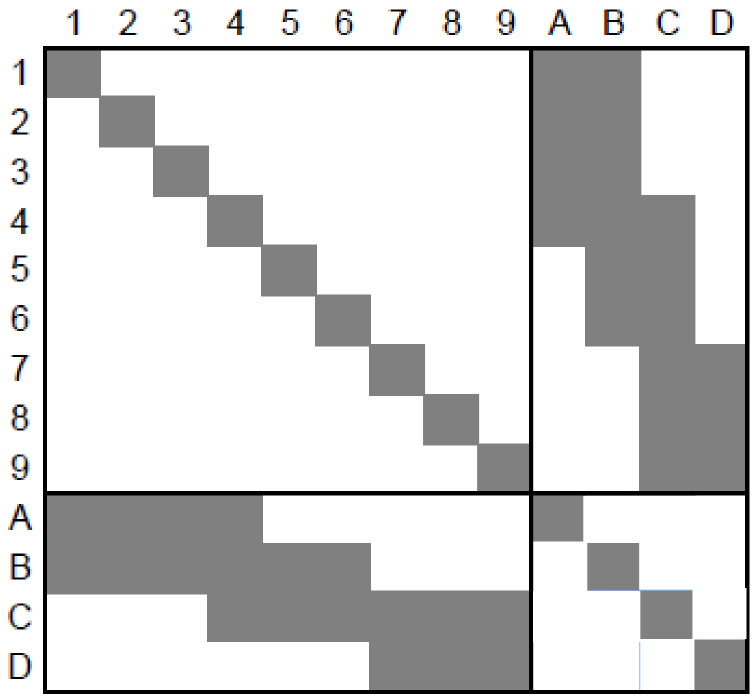
Jacbian sparsity pattern $\mathbf{J} = \frac{\mathrm{d}\mathbf{Z}}{\mathrm{d}\mathbf{X}}$

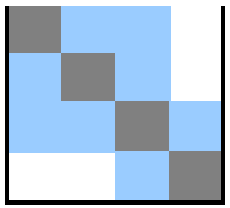
Non-null if camera j detects point i

Szeliski 2022 2nd Edition. 11.4.3 Exploiting sparsity

Universidad Zaragoza 1542

Hessian sparsity pattern





$\mathbf{A'}_{cc}$ Densifies per each pair of cameras observing common points. Covisibility

Triggs et al. 2000. 6.1 The Schur Complement and the Reduced Bundled System

In each iteration, we have to solve

$$\mathbf{J}^T\mathbf{J}\delta\mathbf{X} = \Delta\mathbf{Z}^T\mathbf{J}$$

$$\mathbf{A} = \mathbf{J}^T\mathbf{J} \qquad \mathbf{b} = \Delta\mathbf{Z}^T\mathbf{J}$$

$$\begin{bmatrix} \mathbf{A}_{pp} & \mathbf{A}_{pc} \\ \mathbf{A}_{cp} & \mathbf{A}_{cc} \end{bmatrix} \begin{bmatrix} \delta\mathbf{X}_p \\ \delta\mathbf{X}_c \end{bmatrix} = \begin{bmatrix} \mathbf{b}_p \\ \mathbf{b}_c \end{bmatrix}$$

Usual case, more points than cameras, Schur complement

$$\mathbf{A'}_{cc} = \mathbf{A}_{cc} - \mathbf{A}_{cp}\mathbf{A}_{pp}^{-1}\mathbf{A}_{pc}$$

$$\mathbf{b'}_c = \mathbf{b}_c - \mathbf{A}_{cp}\mathbf{A}_{pp}^{-1}\mathbf{b}_c$$

Solving for cameras first

$$\mathbf{A'}_{cc}\Delta\mathbf{X}_c = \mathbf{b'}_c$$

Solving for the points afterwards

$$\mathbf{A}_{pp}\Delta\mathbf{X}_p = \mathbf{b}_p - \mathbf{A}_{pc}\Delta\mathbf{X}_c$$

6.4 Initial Guess

- To start from scratch, from matches in two views
  - Two view cameras poses, motion from E
  - 3D points, triangulation
- For more than two views
  - Camera pose/PnP, to recover camera pose from 2D-3D matches.
  - 3D points, triangulation to recover 3D point poses from matches in several views.
- Intial guesses are refined by Non-linear optimizations

6.5 Same goal function. Different estimates

1. Full Bundle Adjustment, estimates scene and camera motion from matches

$$\underset{\left\{\theta_{ext}^{j},\mathbf{X}_i\right\}}{\operatorname{argmin}} \sum_{i,j} \left\| \mathbf{u}_i^j - \mathbf{P}^j\left(\mathbf{X}_i, \theta_{int}^j, \theta_{ext}^j\right) \right\|^2$$

2. Camera pose, only the camera/cameras are estimated

$$\underset{\left\{\theta_{ext}^{j}\right\}}{\operatorname{argmin}} \sum_{i,j} \left\| \mathbf{u}_i^j - \mathbf{P}^j\left(\mathbf{X}_i, \theta_{int}^j, \theta_{ext}^j\right) \right\|^2$$

3. Triangulation, only the 3D points are estimated

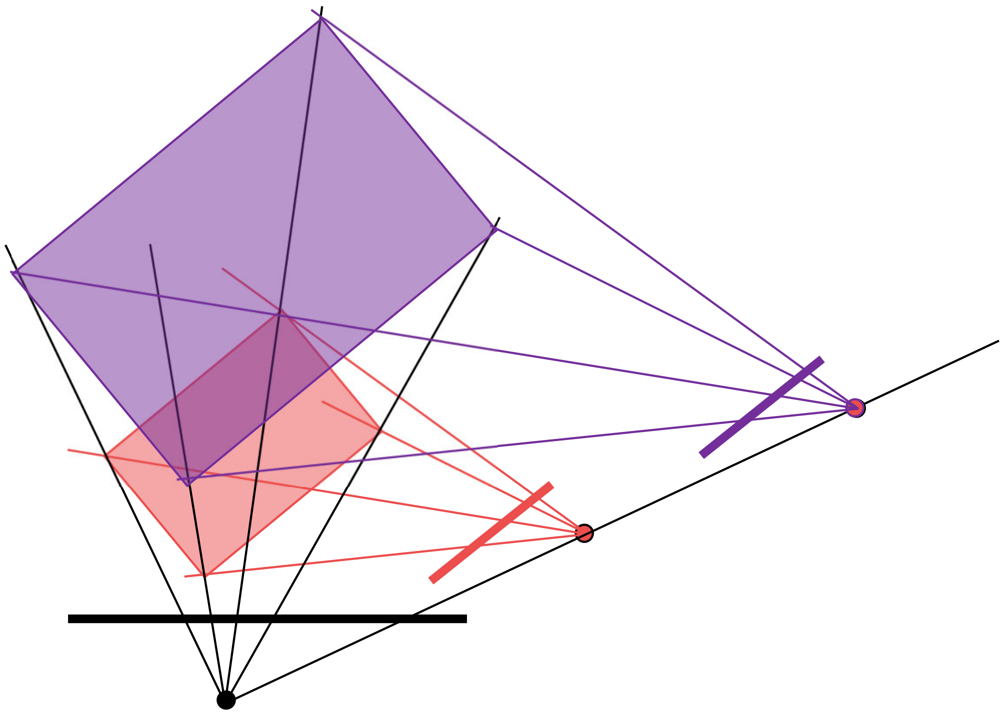
$$\underset{\left\{\boldsymbol{X}_i\right\}}{\operatorname{argmin}} \sum_{i,j} \left\| \mathbf{u}_i^j - \mathbf{P}^j\left(\mathbf{X}_i, \theta_{int}^j, \theta_{ext}^j\right) \right\|^2$$

4. Even, self calibration where camera pose, calibration parameters, and scene, are estimated

$$\underset{\left\{\theta_{ext}^{j},\theta_{int}^{j},\mathbf{X}_i\right\}}{\operatorname{argmin}} \sum_{i,j} \left\| \mathbf{u}_i^j - \mathbf{P}^j\left(\mathbf{X}_i, \theta_{int}^j, \theta_{ext}^j\right) \right\|^2$$

5. … virtually any combination, assuming enough matches, well conditioned geometry, and available initial guesses ☺

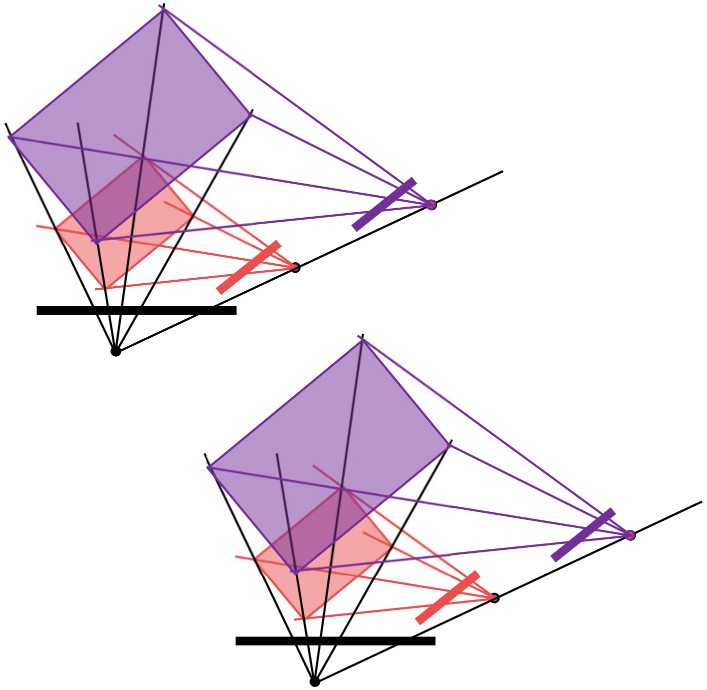
6.6 Up to scale gauge freedom



Small translation observing a small object produces the same rays than a big object after big translations

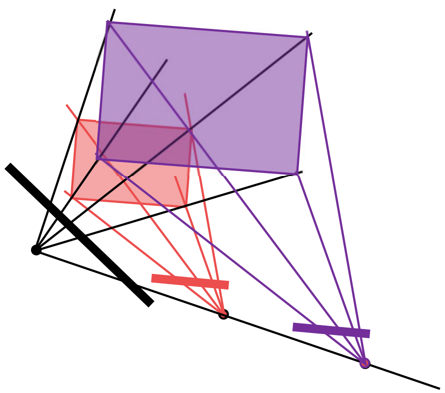
Distance between two points or two cameras determines the scale.

6.6 Absolute frame gauge freedom



Any translation/rotation affecting the cameras and scene produces the same projection rays.

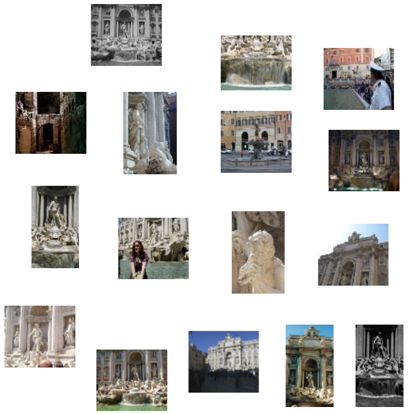
We cannot tell if the images where taken in Jaca, in China, Zaragoza or Mars.



$$\mathbf{X} = \left[\mathbf{X}_1^T, \mathbf{X}_2^T, \ldots, \mathbf{X}_{n_p}^T, {\boldsymbol{\theta}_{ext}^{2}}^T, {\boldsymbol{\theta}_{ext}^{3}}^T, \ldots, {\boldsymbol{\theta}_{ext}^{m_c}}^T\right]^T$$

First camera not included!

6.7 Phototurims example



Input photographs

Scene reconstruction

Relative camera positions and orientations

Point cloud

Sparse correspondence

N Snavely, SM Seitz, R Szeliski. Modeling the world from internet photo collections International Journal of Computer Vision 80 (2), 189-210

SfM, emparejamiento automatizado + reconstrucción. VisualSFM. http://ccwu.me/vsfm/

Video http://phototour.cs.washington.edu/PhotoTourismFull.wmv

6.7 Phototurims example

Detect features using SIFT [Lowe, IJCV 2004]

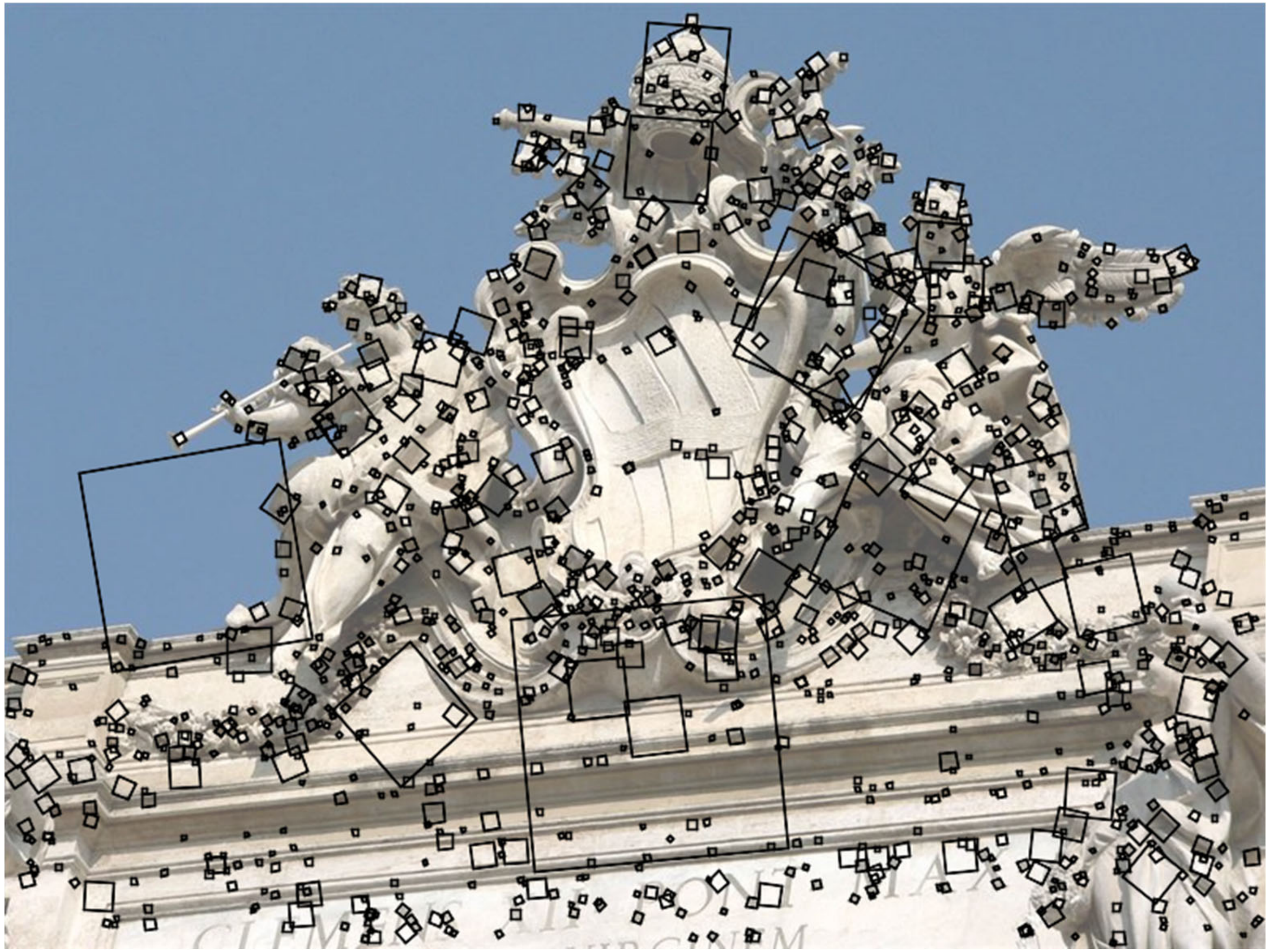


Photo Tourism Exploring Photo Collections in 3D N. Snavely, S.M. Seitz, R. Szeliski . http://phototour.cs.washington.edu/Photo_Tourism.ppt

6.7 Phototurims example

Detect features using SIFT [Lowe, IJCV 2004]

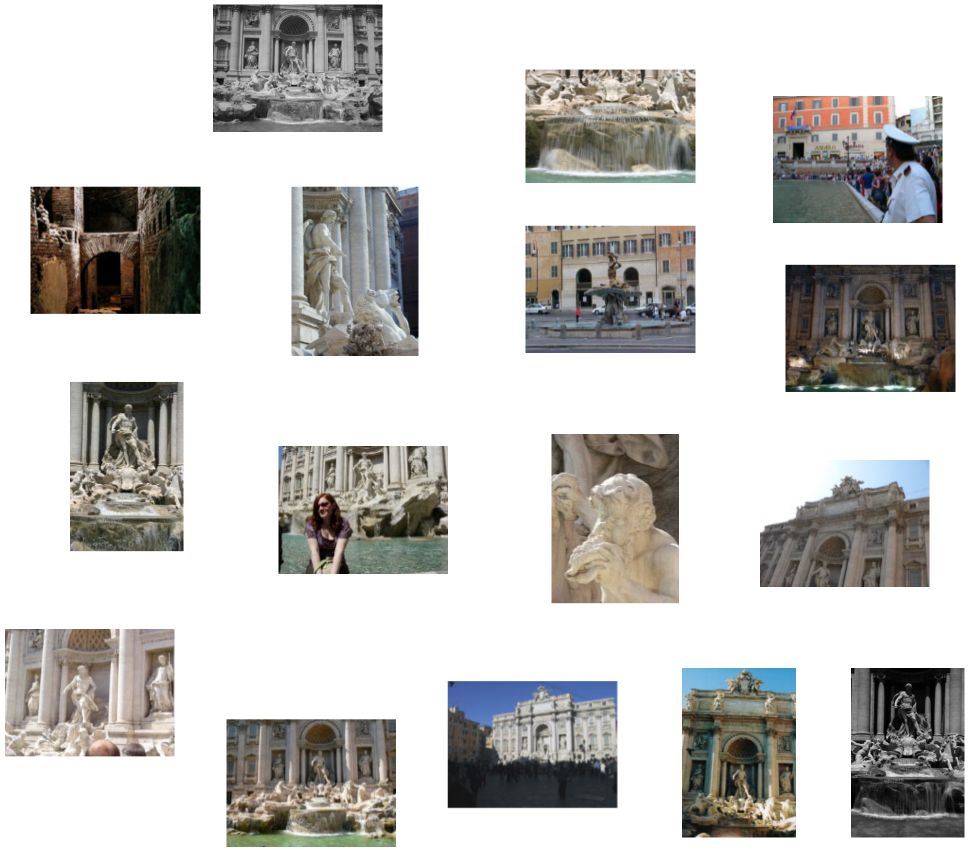


Photo Tourism Exploring Photo Collections in 3D N. Snavely, S.M. Seitz, R. Szeliski . http://phototour.cs.washington.edu/Photo_Tourism.ppt

6.7 Phototurims example

Detect features using SIFT [Lowe, IJCV 2004]

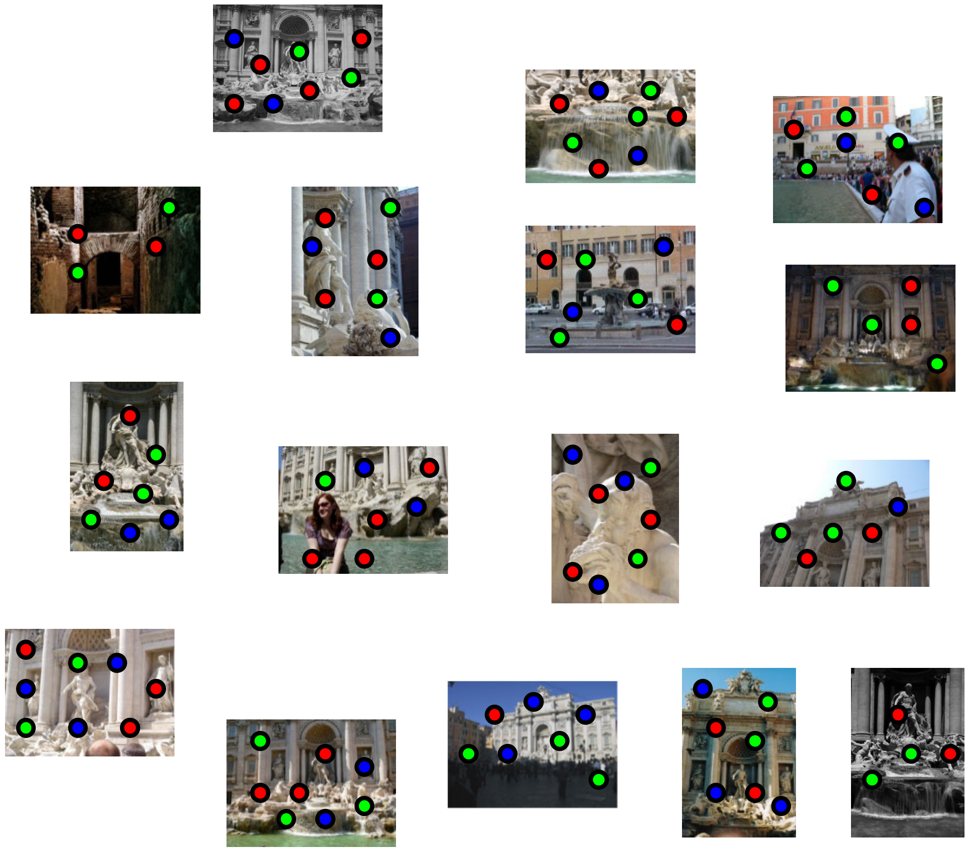


Photo Tourism Exploring Photo Collections in 3D N. Snavely, S.M. Seitz, R. Szeliski . http://phototour.cs.washington.edu/Photo_Tourism.ppt

6.7 Feature matching

Match features between each pair of images

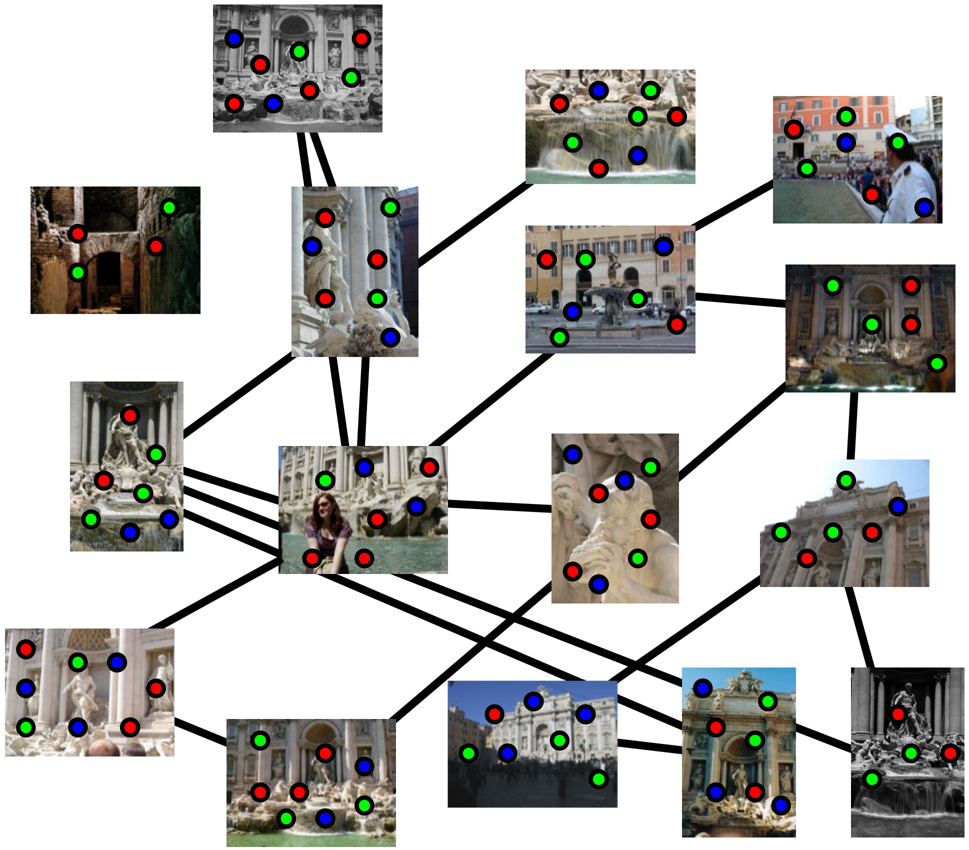


Photo Tourism Exploring Photo Collections in 3D N. Snavely, S.M. Seitz, R. Szeliski . http://phototour.cs.washington.edu/Photo_Tourism.ppt

6.7 Phototurims example

Refine matching using RANSAC [Fischler & Bolles 1987] to estimate fundamental matrices between pairs

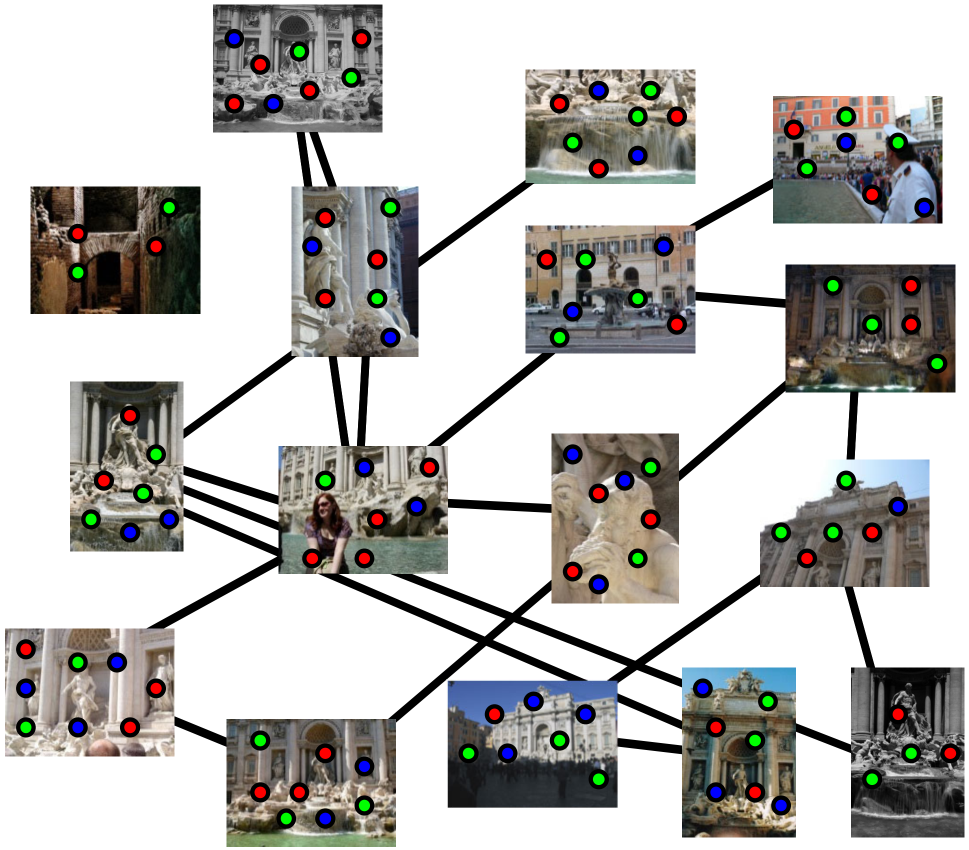


Photo Tourism Exploring Photo Collections in 3D N. Snavely, S.M. Seitz, R. Szeliski . http://phototour.cs.washington.edu/Photo_Tourism.ppt

6.7 Phototurims example

- Link up pairwise matches to form connected components of matches across several images

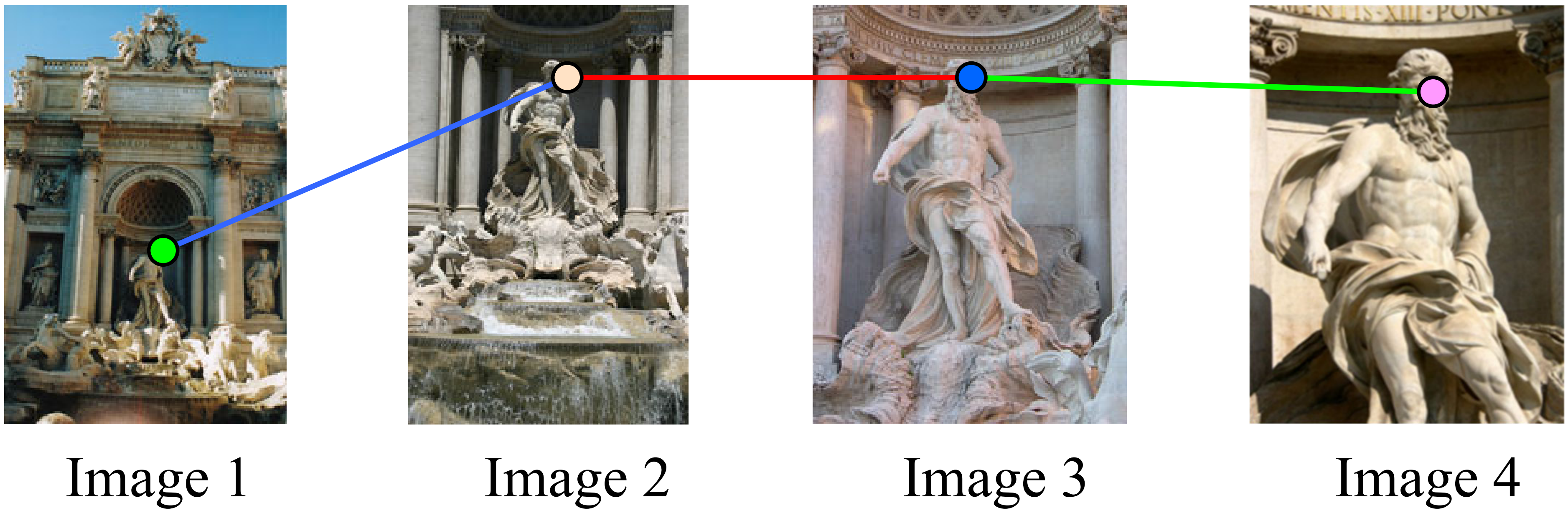
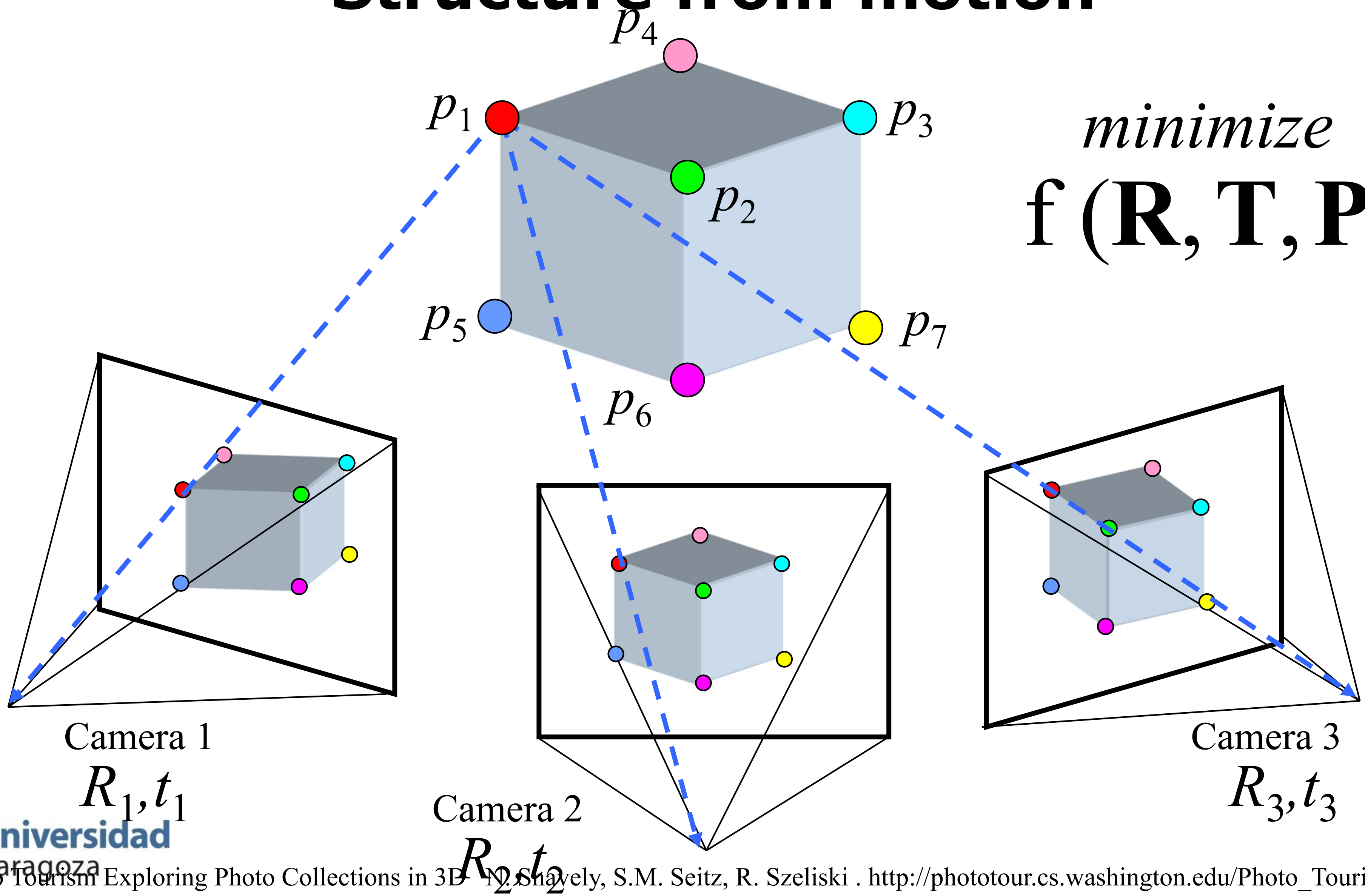


Photo Tourism Exploring Photo Collections in 3D N. Snavely, S.M. Seitz, R. Szeliski . http://phototour.cs.washington.edu/Photo_Tourism.ppt

6.7 Phototurims example
Structure from motion

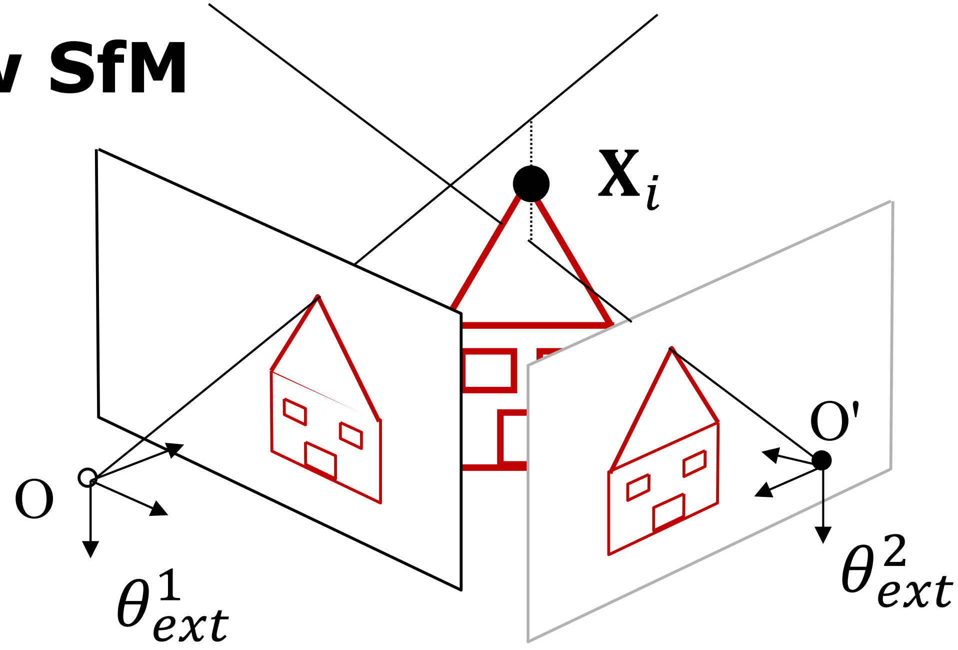


minimize

$$\mathrm{f}\,(\mathbf{R}, \mathbf{T}, \mathbf{P})$$

Photo Tourism Exploring Photo Collections in 3D . N. Snavely, S.M. Seitz, R. Szeliski . http://phototour.cs.washington.edu/Photo_Tourism.ppt

6.8 Minimal Solutions Two-View SfM

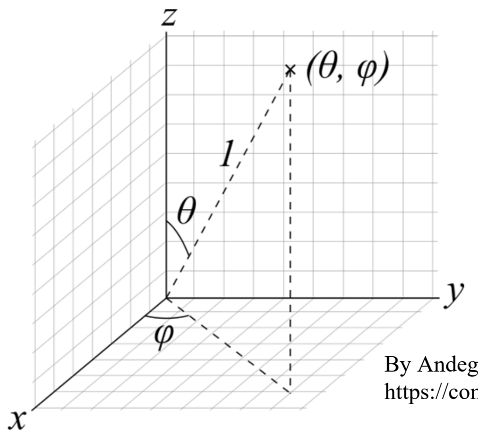


Absolute pose gauge freedom

$\theta_{ext}^1 = (0,0,0,0,0,0)$

Scale gauge freedom 5 dof θ_{ext}^2

$$\theta_{ext}^2 = (\sin\theta\cos\varphi, \sin\theta\sin\varphi, \cos\theta, R_x, R_y, R_z)$$



By Andeggs - Own work, Public Domain, https://commons.wikimedia.org/w/index.php?curid=7478049

1 point correspondence:
- 4 equations, x,y in two images
- 3 unknowns, we add a 3 point to be computed

# points	# unknowns	# equations
0	5	0
1	$5+3=8$	$1\times4=4$
2	$5+2\times3=11$	$2\times4=8$
3	$5+3\times3=14$	$3\times4=12$
4	$5+4\times3=17$	$4\times4=16$
5	$5+5\times3=20$	$5\times4=20$
6	$5+6\times3=23$	$6\times4=24$

Universidad Zaragoza 1542

6.8 BA typical sizes

Given

m calibrated cameras with unknown pose θ_{ext}^{j}

n scene points with unknown pose $\mathbf{X}_i$

Calibration for the m is known θ_{int}^{j}

Observations of the scene points in the cameras are also known $\mathbf{u}_i^j$

can be computed up to a scale factor:

6 d.o.f. localization (pose and orientation) for the m cameras θ_{ext}^{j}

3D pose for the n points $\mathbf{X}_i$

Scale factor gauge freedom affects, the point poses and the camera translations.

The absolute reference frame for the coordinates is unknown, a gauge freedom.

6.8 BA typical sizes

Redundancy, more equations than unknowns:

- Minimum camera number 2.
- A point has to be matched in at least two cameras.
- For 2 cameras, minimum number of points 5, usually tens.
- For more than two cameras, the point does not need to be seen in all the cameras.
- Tens, hundreds of cameras, thousands or points.

Bibliography

- R. Szeliski: "Computer Vision. Algorithms and Applications". 2nd Edition. Springer 2022. http://szeliski.org/Book/
- Schonberger, J. L., & Frahm, J. M. (2016). Structure-from-motion revisited. In Proceedings of the IEEE conference on computer vision and pattern recognition (pp. 4104-4113). Nice review offering a comprehensive review for the automated sparse geometry estimation from images.
- B Triggs, P McLauchlan, R Hartley, A Fitzgibbon: "Bundle adjustment–A modern synthesis". Vision Algorithms: Theory and Practice, 2000 – Springer.
- Hartley R.I y Zisserman A.: "Multiple View Geometry". Cambridge University Press. Second Edition. 2004.
- Manolis I. A. Lourakis and Antonis A. Argyros: "SBA: A Software Package for Generic Sparse Bundle Adjustment". ACM Transactions on Mathematical Software, Vol. 36, No. 1, Article 2, March 2009.
- Madsen, K, Nielsen, HB & Tingleff, O 2004, Methods for Non-Linear Least Squares Problems (2nd ed.) https://orbit.dtu.dk/files/2721358/imm3215.pdf

Software

- COLMAP, general-purpose Structure-from-Motion (SfM) and Multi-View Stereo (MVS) pipeline https://colmap.github.io/ Correspond to Schonberger 2016
- Ceres Solver open source C++ library for modeling and solving large, complicated optimization problems. Google 2010. http://ceres-solver.org/
- SfM, emparejamiento automatizado + reconstrucción. VisualSFM. http://ccwu.me/vsfm/
- Manolis Lourakis. sba : A Generic Sparse Bundle Adjustment C/C++ Package Based on the Levenberg-Marquardt Algorithm http://users.ics.forth.gr/~lourakis/sba/
- Rainer Kuemmerle; Giorgio Grisetti; Hauke Strasdat; Kurt Konolige; Wolfram Burgard. g2o: A General Framework for Graph Optimization. https://openslam.org/g2o.html
- Photomodeler, software comercial para reconstrucción fotogramétrica http://www.photomodeler.com/index.html