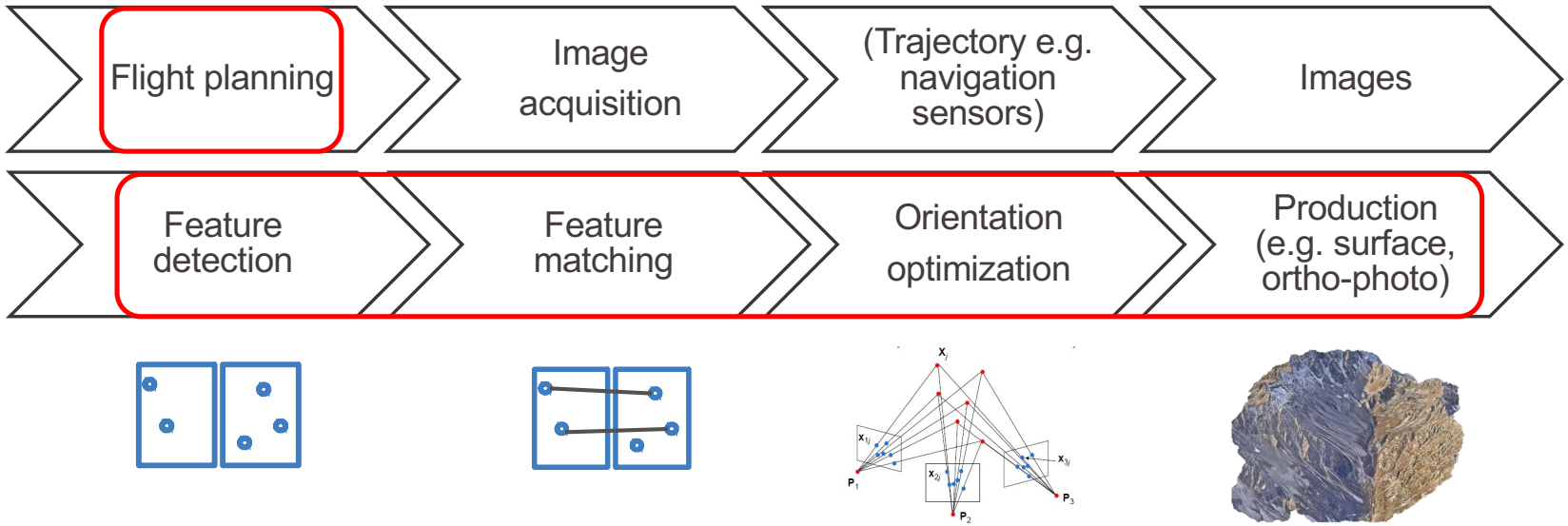
Lecture 3
Image Orientation

ENV408: Optical Sensing & Modeling for Earth Observations

Jan Skaloud



Lectures

- Image primes (L1)
- Salient *features* (L2)
- Image *orientation* (L3)
- *Stereo* vision (L4)
- Many photos & mapping products (L5)

Exercises

- Image 'corrections' (Lab01)
- Detection & matching (Lab02)
- Approx. absolute orientation (Lab03)
- Approx. relative orientation (Lab04)
- Calibration, DEM, ortho-photo (Lab05)

Motivation ?

- Ambitions of the 1st time.
- Can you do it (in 2002)?
- Challenge 1:
  - DTM resolution ~ 1 m

Image orientation (= camera calibration)

Joint problem to solve: determine intrinsic & extrinsic (interior & exterior orientation) of a camera

Strategies:

I. Unknown camera *calibration* & *pose*
   i. Approximate first (fast & linear)
   ii. Improve (optimization & linearized)

II. Unknown *pose* but known *calibration*
   i. Pose(EO) unknown: fast & linear
   ii. Pose(EO) ~known: non-linear/optimal

Typical setups for lab camera calibration (arXiv)

3D field

Planar surfaces

Terminology – CV and photogrammetry

Camera parameters

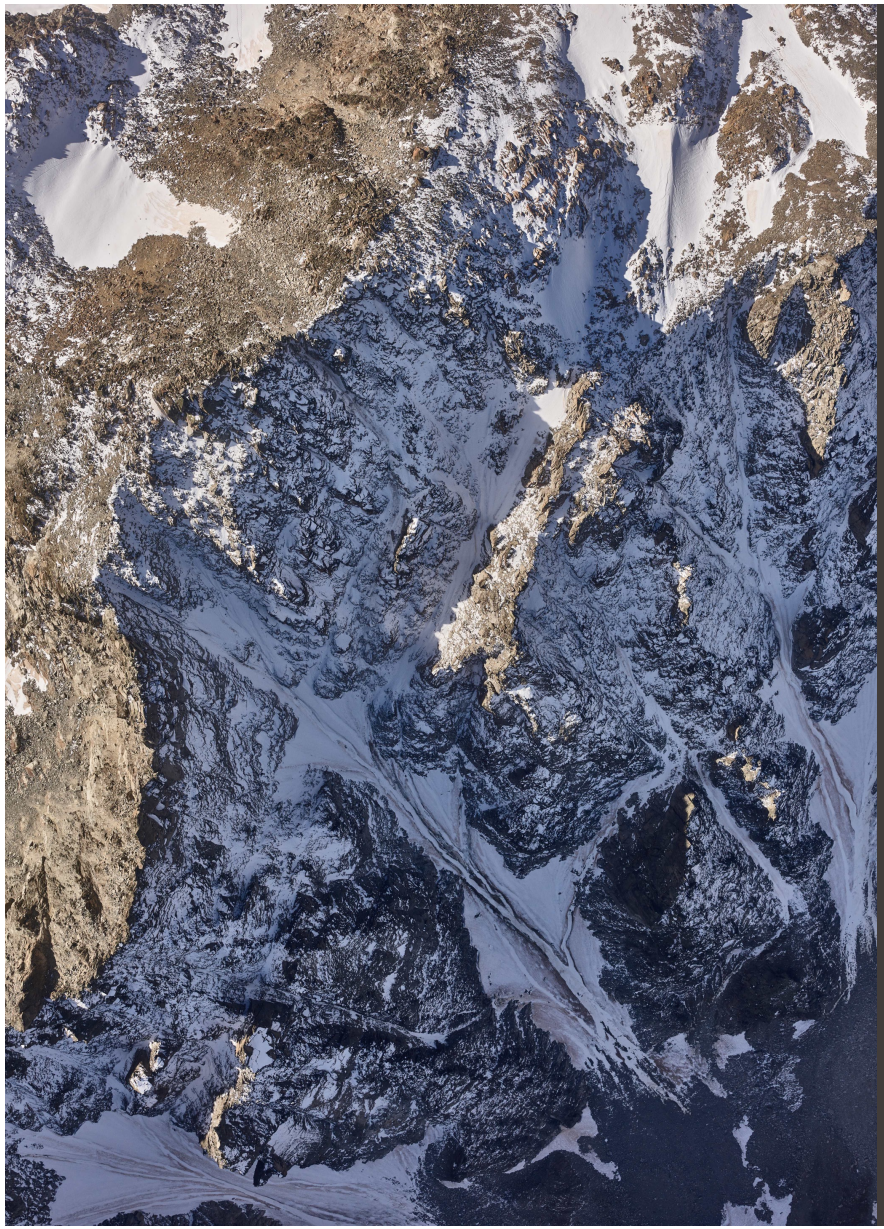
Intrinsic parameters (= interior orientation):
Parameters related to camera optic

i. focal length
ii. optical center
iii. distortion model

Extrinsic parameters (= exterior orientation EO, = camera pose):
Position and orientation in space of the camera in space (can be absolute or relative)

Extrinsic + intrinsic:
also called camera calibration or orientation

Photogrammetric term vs Computer Vision term



Perspective Camera Model

DLT algorithm

Refine orientation and calibration

Alternative algorithm

Image Orientation

How to estimate camera extrinsic (position and orientation in space) ?

Given:

i. set of points with know 3D coordinates $\mathbf{P}_i$
ii. their corresponding coordinates (undistorted) in image plane $\mathbf{p}_i$
iii. (known camera intrinsic)

Main steps:

i. Build COLLINEARITY equations given $\mathbf{p}_i$ and $\mathbf{P}_i$
ii. Estimate camera pose with DLT algorithm
iii. Estimate quality of the reprojection with reprojection error

Today: background, formulation, algorithms

Tomorrow: practical implementation starting from Lab 01 undistorted measurements

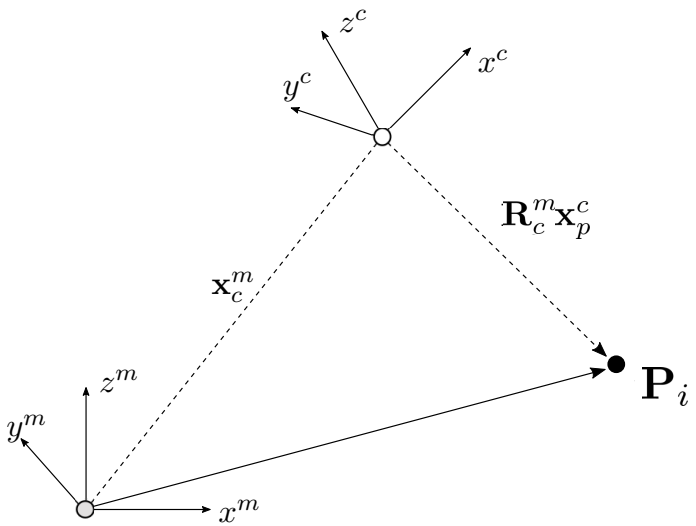
Frames

Mapping frame (m):
abstract coordinate system with ref. surface to provide known locations and create maps.

e.g. International Terrestrial Reference System (ITRF) + Year

country-wide, in Switzerland CH1903+/LV95 - EPSG:2056

Camera frame (c):
coordinate system defined by the camera position and orientation



Notation:

- Position of point P in m frame $\mathbf{P}_i = \mathbf{x}_i^m = \left(X_i^m, Y_i^m, Z_i^m\right)^T$
- Position of point p in image plane [px] $\mathbf{p}_i = \left(u^i, v^i,\right)^T$
- Position of camera c in m frame $\mathbf{x}_c^m$
- Rotation matrix (3x3, $\mathbf{R}^{-1} = \mathbf{R}^T$) from camera to mapping frame $\mathbf{R}_c^m$
  (aligns camera frame axis to that of mapping frame)

Changing (transforming) Frame

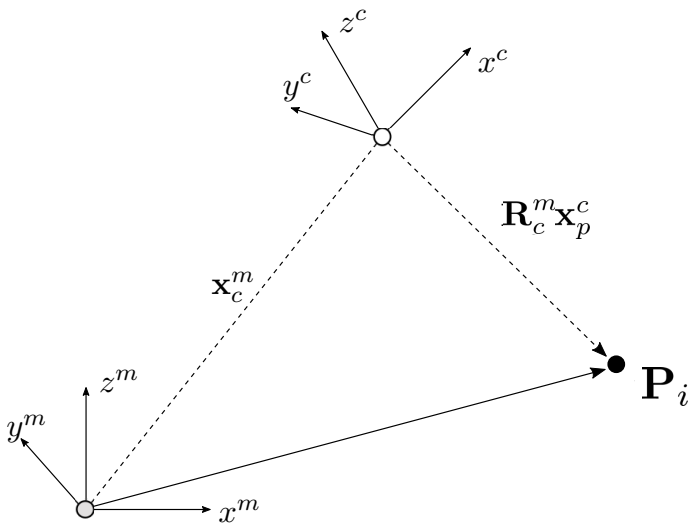
Camera position, mapping frame

Point position, in mapping frame

$$\mathbf{x}_p^m = \mathbf{x}_c^m + \mathbf{R}_c^m \mathbf{x}_p^c$$

Point position, camera frame

Rotation from camera to mapping frame



Homogeneous coordinates

Goal: express more "efficiently"

$$\mathbf{x}_p^m = \mathbf{x}_c^m + \mathbf{R}_c^m \mathbf{x}_p^c$$

How: Add 4th term / coordinate

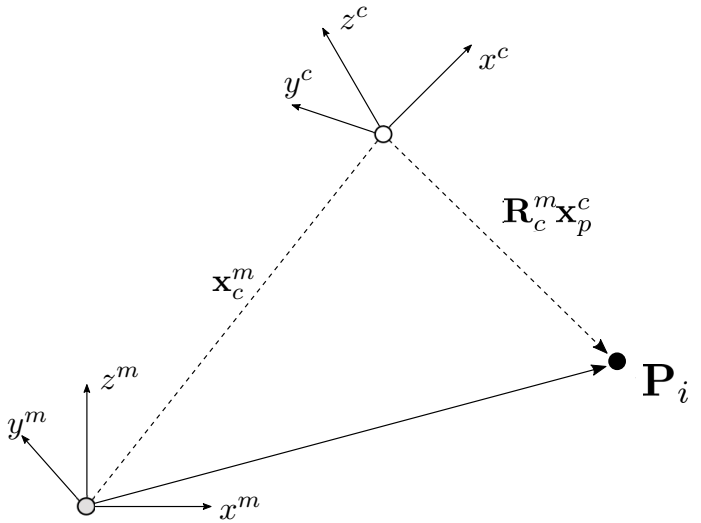
$$\overline{\mathbf{x}} \doteq \begin{pmatrix} \mathbf{x} \\ 1 \end{pmatrix} = \begin{pmatrix} X \\ Y \\ Z \\ 1 \end{pmatrix}$$

Rewrite frame transformation as:

$$\overline{\mathbf{x}}_p^m = \begin{pmatrix} \mathbf{x}^m \\ 1 \end{pmatrix} = \begin{pmatrix} \mathbf{R}_c^m & \mathbf{x}_c^m \\ 0 & 1 \end{pmatrix} \begin{pmatrix} \mathbf{x}_p^c \\ 1 \end{pmatrix} \doteq \overline{[\mathbf{R}|\mathbf{t}]}_c^m \, \overline{\mathbf{x}}_p^c = \overline{\mathbf{T}}_c^m \overline{\mathbf{x}}_p^c$$

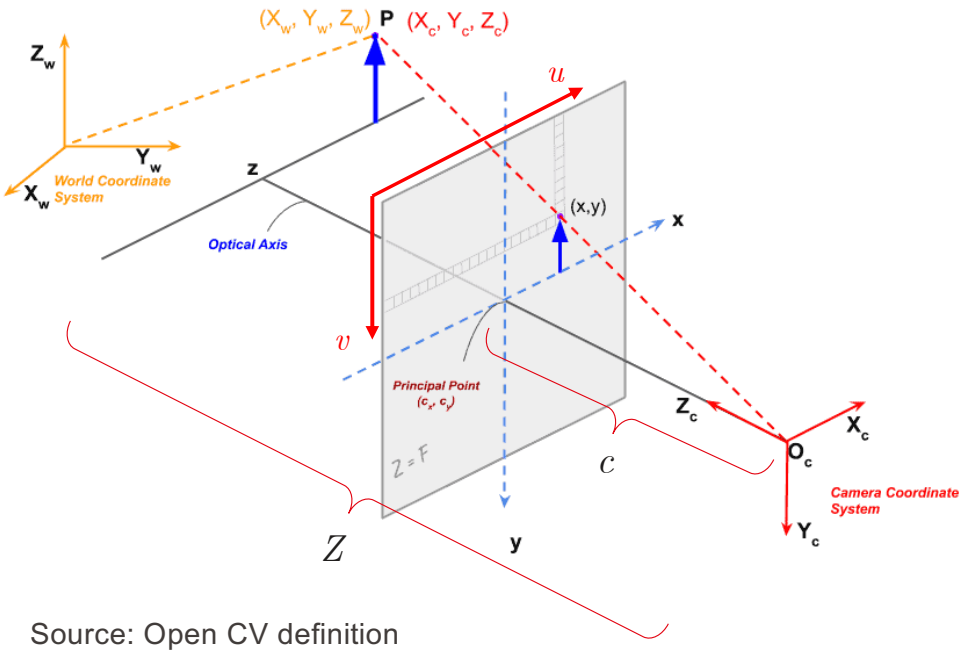
Practical advantage: concatenation by multiplication:

$$\overline{\mathbf{T}}_c^a = \overline{\mathbf{T}}_b^a \, \overline{\mathbf{T}}_c^b = \begin{pmatrix} \mathbf{R}_b^a & \mathbf{x}_b^a \\ 0 & 1 \end{pmatrix} \begin{pmatrix} \mathbf{R}_c^b & \mathbf{x}_c^b \\ 0 & 1 \end{pmatrix} = \begin{pmatrix} \mathbf{R}_c^a & \mathbf{x}_c^a \\ 0 & 1 \end{pmatrix}$$



$$(\overline{\mathbf{T}}_c^a)^{-1} = \begin{pmatrix} (\mathbf{R}_c^a)^T & -(\mathbf{R}_c^a)^T \mathbf{x}_c^a \\ 0 & 1 \end{pmatrix} = \overline{\mathbf{T}}_a^c$$

Perspective camera model



Source: Open CV definition

Ratios

$$\mathbf{x} = \frac{1}{c}\begin{pmatrix} x \\ y \end{pmatrix} = \frac{1}{Z}\begin{pmatrix} X \\ Y \end{pmatrix}$$ (c unitless here)

Perspective camera model:

Relates point p on image plane to corresponding point P in 3D

$$\mu \cdot p = \mathbf{K} \cdot P$$

With $\mu = Z$

See next slide for developed equation

Perspective camera model

Developed form:

$$p \qquad\qquad P$$

$$\mu \cdot \begin{bmatrix} u \\ v \\ 1 \end{bmatrix} = \underbrace{\begin{bmatrix} c & 0 & x_{PPS} + c_x & 0 \\ 0 & c & y_{PPS} + c_y & 0 \\ 0 & 0 & 1 & 0 \end{bmatrix}}_{\mathbf{K}} [\mathbf{R}|\mathbf{t}]_m^c \begin{bmatrix} X^m \\ Y^m \\ Z^m \\ 1 \end{bmatrix}$$

Link with lab 01:

$\mathbf{K} \iff$ perspective2topleft(xy, h, w, cx, cy, c)

$\mathbf{K}^{-1} \iff$ topleft2perspective(uv, h, w, cx, cy, c)

i.e. $\mathbf{K}$ matrix operation is equivalent to converting measurements from perspective centered to top-left and vice-versa

Perspective camera model

Developed form:

$$\mu \cdot \underbrace{\begin{bmatrix} u \\ v \\ 1 \end{bmatrix}}_{p} = \overbrace{\begin{bmatrix} c & 0 & x_{PPS} + c_x & 0 \\ 0 & c & y_{PPS} + c_y & 0 \\ 0 & 0 & 1 & 0 \end{bmatrix}}^{\mathbf{K}} \overbrace{[\mathbf{R}|\mathbf{t}]_m^c \begin{bmatrix} X^m \\ Y^m \\ Z^m \\ 1 \end{bmatrix}}^{P}$$

Today's assumption: intrinsic known

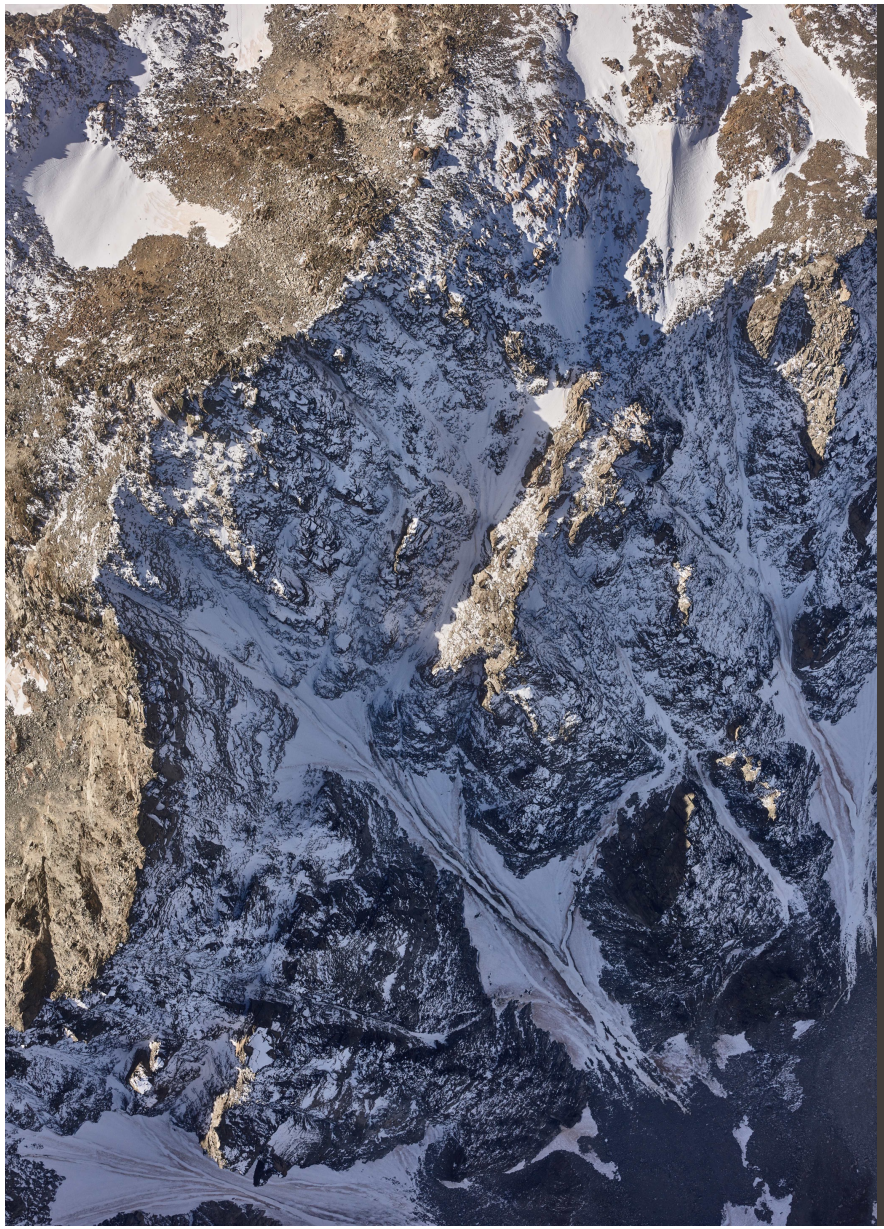
- K is known, no need to solve for it
- System becomes:

Solve this to find **R**, **t**

$$\Longrightarrow f_{undistort}\left(\mathbf{K}^{-1}\begin{bmatrix} u' \\ v' \\ 1 \end{bmatrix}\right) = \begin{bmatrix} x \\ y \\ 1 \end{bmatrix} = \tfrac{1}{\mu}[\mathbf{R}|\mathbf{t}]_m^c \begin{bmatrix} X^m \\ Y^m \\ Z^m \\ 1 \end{bmatrix}$$

Lab 01

Lab 03



Perspective Camera Model

DLT algorithm

Refine orientation and calibration

Alternative algorithm

Direct Linear Transformation (DLT)

Approach:

Solve collinearity equation in homogenous coordinates for unknown terms: $\mathbf{R}, \mathbf{t}$

From image obs. $\mathbf{p}_i$ and known 3D points $\mathbf{P}_i$

$$f_{undistort}\left(\mathbf{K}^{-1}\begin{bmatrix}u'\\v'\\1\end{bmatrix}\right) = \begin{bmatrix}x\\y\\1\end{bmatrix} = \tfrac{1}{\mu}[\mathbf{R}|\mathbf{t}]_m^c \begin{bmatrix}X^m\\Y^m\\Z^m\\1\end{bmatrix}$$

$$\implies \mu\begin{bmatrix}x\\y\\1\end{bmatrix} = \begin{bmatrix}r_{11} & r_{12} & r_{13} & t_1\\ r_{21} & r_{22} & r_{23} & t_2\\ r_{31} & r_{32} & r_{33} & t_3\end{bmatrix}\begin{bmatrix}X^m\\Y^m\\Z^m\\1\end{bmatrix}$$

Direct Linear Transformation (DLT)

$$\overbrace{M = [\mathbf{R}|\mathbf{t}]}^{} \qquad \overbrace{P}^{}$$

$$\implies \mu \begin{bmatrix} x \\ y \\ 1 \end{bmatrix} = \begin{bmatrix} r_{11} & r_{12} & r_{13} & t_1 \\ r_{21} & r_{22} & r_{23} & t_2 \\ r_{31} & r_{32} & r_{33} & t_3 \end{bmatrix} \begin{bmatrix} X^m \\ Y^m \\ Z^m \\ 1 \end{bmatrix} = \begin{bmatrix} m_{11} & m_{12} & m_{13} & m_{14} \\ m_{21} & m_{22} & m_{23} & m_{24} \\ m_{31} & m_{32} & m_{33} & m_{34} \end{bmatrix} \begin{bmatrix} X^m \\ Y^m \\ Z^m \\ 1 \end{bmatrix}$$

$$\implies \mu \begin{bmatrix} x \\ y \\ 1 \end{bmatrix} = M \cdot P = \begin{bmatrix} m_1^T \\ m_2^T \\ m_3^T \end{bmatrix} \cdot P$$

Divide the first and second equations by the third, forms the following system :

$$x = \frac{\mu x}{\mu} = \frac{m_1^T \cdot P}{m_3^T \cdot P}$$

$$y = \frac{\mu y}{\mu} = \frac{m_1^T \cdot P}{m_3^T \cdot P}$$

$$\implies \begin{matrix} (m_1^T - x m_3^T) \cdot P = 0 \\ (m_2^T - y m_3^T) \cdot P = 0 \end{matrix} \implies \begin{pmatrix} P_1^T & 0^T & -x_1 P_1^T \\ 0^T & P_1^T & -y_1 P_1^T \end{pmatrix} \cdot \begin{pmatrix} m_1 \\ m_2 \\ m_3 \end{pmatrix} = \begin{pmatrix} 0 \\ 0 \end{pmatrix}$$

Direct Linear Transformation (DLT)

$$\begin{pmatrix} P_1^T & 0^T & -x_1 P_1^T \\ 0^T & P_1^T & -y_1 P_1^T \end{pmatrix} \cdot \begin{pmatrix} m_1 \\ m_2 \\ m_3 \end{pmatrix} = \begin{pmatrix} 0 \\ 0 \end{pmatrix}$$

- For n observations of points (image-frame and map-frame), we obtain a large matrix:

$$\begin{pmatrix} P_1^T & 0^T & -x_1 P_1^T \\ 0^T & P_1^T & -y_1 P_1^T \\ & \vdots & \\ P_n^T & 0^T & -x_n P_n^T \\ 0^T & P_n^T & -y_n P_n^T \end{pmatrix} \begin{pmatrix} m_1 \\ m_2 \\ m_3 \end{pmatrix} = \begin{pmatrix} 0 \\ 0 \\ \vdots \\ 0 \\ 0 \end{pmatrix}$$

- In details

$$\underbrace{\begin{pmatrix} X_1^m & Y_1^m & Z_1^m & 1 & 0 & 0 & 0 & 0 & -x_1 X_1^m & -x_1 Y_1^m & -x_1 Z_1^m & -x_1 \\ 0 & 0 & 0 & 0 & X_1^m & Y_1^m & Z_1^m & 1 & -y_1 X_1^m & -y_1 Y_1^m & -y_1 Z_1^m & -y_1 \\ & & & & \cdots & \cdots & \cdots & & & & & \\ X_n^m & Y_n^m & Z_n^m & 1 & 0 & 0 & 0 & 0 & -x_n X_n^m & -x_n Y_n^m & -x_n Z_n^m & -x_n \\ 0 & 0 & 0 & 0 & X_n^m & Y_n^m & Z_n^m & 1 & -y_n X_n^m & -y_n Y_n^m & -y_n Z_n^m & -y_n \end{pmatrix}}_{\text{known!}} \underbrace{\begin{pmatrix} m_{11} \\ m_{12} \\ m_{13} \\ m_{14} \\ m_{21} \\ m_{22} \\ m_{23} \\ m_{24} \\ m_{31} \\ m_{32} \\ m_{33} \\ m_{34} \end{pmatrix}}_{?} = \begin{pmatrix} 0 \\ 0 \\ \vdots \\ 0 \\ 0 \end{pmatrix} \quad \Longrightarrow \boxed{Q \cdot \mathtt{vec}\,(M) = 0}$$

Direct Linear Transformation (DLT)

$$Q \cdot \mathtt{vec}(M) = 0$$

Solution

Solving the system give us $M = \Pi = [R|t]$

Unique solution

- $Q_{(2n \times 12)}$ should have minimum rank 11 to have unique (up to a scale) *non-zero* solution $\mathtt{vec}(M)$
- What is the minimum number of point-correspondences?

Overdetermined

- For $n \geq min.$, solution is to minimize the sum of squared residuals (least-squares) $||Q \cdot \mathtt{vec}(M)||_2$
- Constraint: $||\mathtt{vec}(M)||_2 = 1$
- Done by singular value decomposition (SVD) of $Q^T Q$

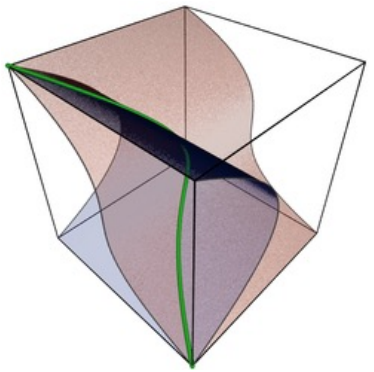
Direct Linear Transformation (DLT)

$$Q \cdot \mathtt{vec}(M) = 0$$

Degenerated situation (rare yet possible…):

In some specific configuration of the points in space, DLT can lead to degenerated solutions, e.g.

- points lying on a plane passing through the center of projection
- camera & points on twisted cubic



Twisted cubic: source Wiki

DLT with unknown intrinsic (interior orientation)

(complementary slide)

Approach:

This time, unknowns to solve for include camera intrinsic: $K(u_0, v_0, c_u, c_v),\ R,\ t$

(more complex, yet possible !)

$$\mu \begin{pmatrix} u \\ v \\ 1 \end{pmatrix} = K[R|t] \cdot \begin{pmatrix} X^m \\ Y^m \\ Z^m \\ 1 \end{pmatrix} = \begin{pmatrix} c_u & 0 & u_0 \\ 0 & c_v & v_0 \\ 0 & 0 & 1 \end{pmatrix} \begin{pmatrix} r_{11} & r_{12} & r_{13} & t_1 \\ r_{21} & r_{22} & r_{23} & t_2 \\ r_{31} & r_{32} & r_{33} & t_3 \end{pmatrix} \begin{pmatrix} X^m \\ Y^m \\ Z^m \\ 1 \end{pmatrix}$$

$$\Longrightarrow \mu \begin{pmatrix} u \\ v \\ 1 \end{pmatrix} = \begin{pmatrix} c_u r_{11} + u_0 r_{31} & c_u r_{12} + u_0 r_{32} & c_u r_{13} + u_0 r_{33} & c_u t_1 + u_0 t_3 \\ c_v r_{21} + v_0 r_{31} & c_v r_{22} + v_0 r_{32} & c_v r_{23} + v_0 r_{33} & c_v t_2 + v_0 t_3 \\ r_{31} & r_{32} & r_{33} & t_3 \end{pmatrix} \begin{pmatrix} X^m \\ Y^m \\ Z^m \\ 1 \end{pmatrix}$$

Direct Linear Transformation (DLT)

(complementary slide)

$$M = \mathbf{K}\,[R|t]$$

$$\Longrightarrow \mu \begin{bmatrix} u \\ v \\ 1 \end{bmatrix} = \begin{bmatrix} c_u r_{11} + u_0 r_{31} & c_u r_{12} + u_0 r_{32} & c_u r_{13} + u_0 r_{33} & c_u t_1 + u_0 t_3 \\ c_v r_{21} + v_0 r_{31} & c_v r_{22} + v_0 r_{32} & c_v r_{23} + v_0 r_{33} & c_v t_2 + v_0 t_3 \\ r_{31} & r_{32} & r_{33} & t_3 \end{bmatrix} \begin{bmatrix} X^m \\ Y^m \\ Z^m \\ 1 \end{bmatrix} = \begin{bmatrix} m_{11} & m_{12} & m_{13} & m_{14} \\ m_{21} & m_{22} & m_{23} & m_{24} \\ m_{31} & m_{32} & m_{33} & m_{34} \end{bmatrix} \begin{bmatrix} X^m \\ Y^m \\ Z^m \\ 1 \end{bmatrix}$$

$\mathbf{M}$ $\mathbf{P}$

$$\Longrightarrow \mu \begin{bmatrix} u \\ v \\ 1 \end{bmatrix} = M \cdot P = \begin{bmatrix} m_1^T \\ m_2^T \\ m_3^T \end{bmatrix} \cdot P$$

- M matrix is more complex to form since it includes unknowns for intrinsic parameters.
- Formulation of Q matrix stays the same

Direct Linear Transformation (DLT)

(complementary slide)

$$Q \cdot \mathtt{vec}(M) = 0$$

Solution

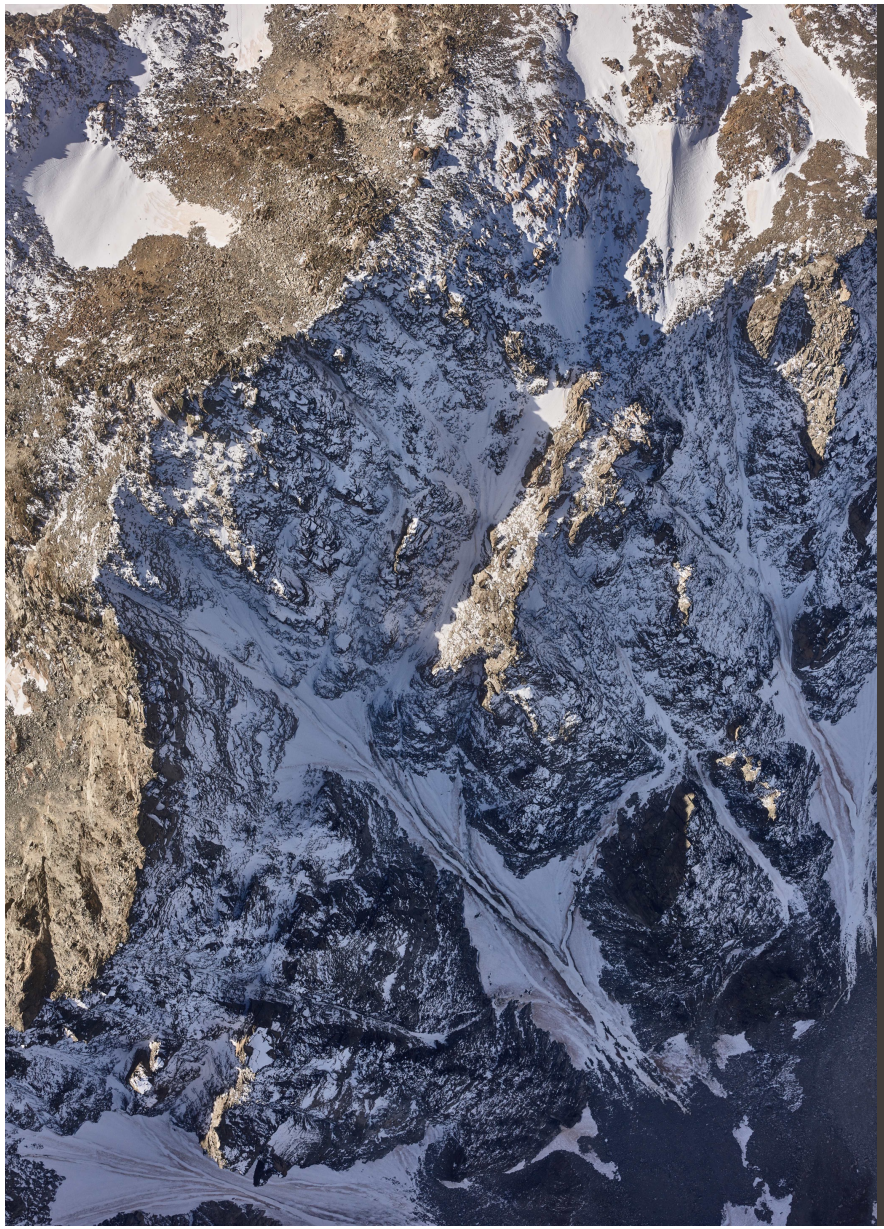
Solving the system give us $\Pi = \mathbf{K}\left[R|t\right]$ instead of $\Pi = \left[R|t\right]$ in the known intrinsic case

Unique solution

- $Q_{(2n \times 12)}$ should have min. rank 11 to have unique (up to a scale) *non-zero* solution M_s
- What is the minimum number of point-correspondences?

Overdetermined

- For $n \geq min.$, solution is to minimize the sum of squared residuals (least-squares) $||Q \cdot \mathtt{vec}(M)||_2$
- Constraint: $||\mathtt{vec}(M)||_2 = 1$
- Done by singular value decomposition (SVD) of $Q^T Q$



Perspective Camera Model

DLT algorithm

Refine orientation and calibration

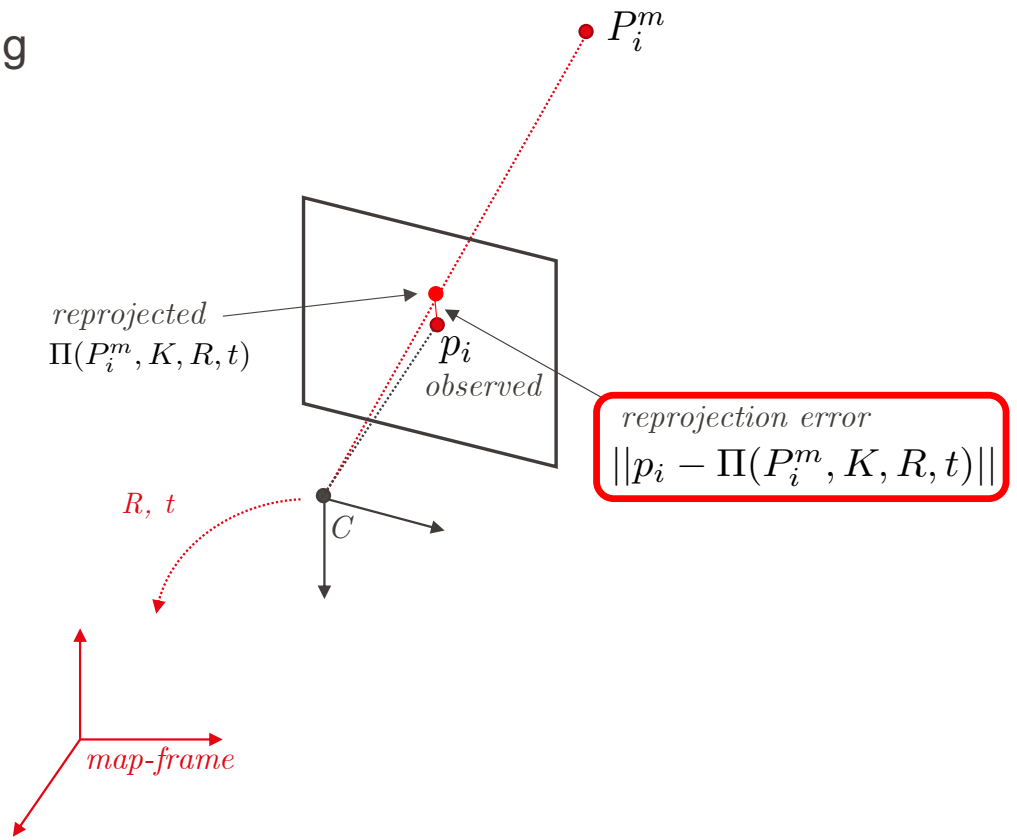
Alternative algorithm

Reprojection error (e_r)

- e_r = Euclidian distance [pixels] between **observed** image point and the corresponding 3D point **reprojected** on the camera frame

- Can be used to access the quality of "calibration".

- What is an acceptable value of e_r ?
- What are the sources of reprojection error?
- How this can be further improved?

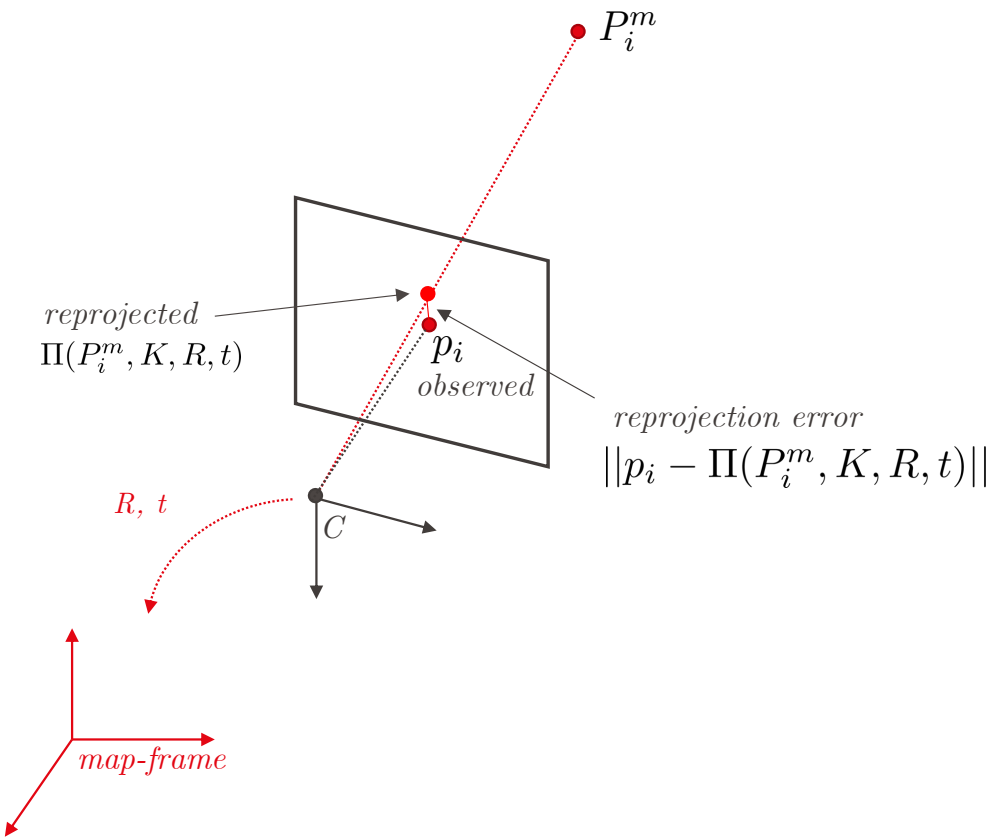


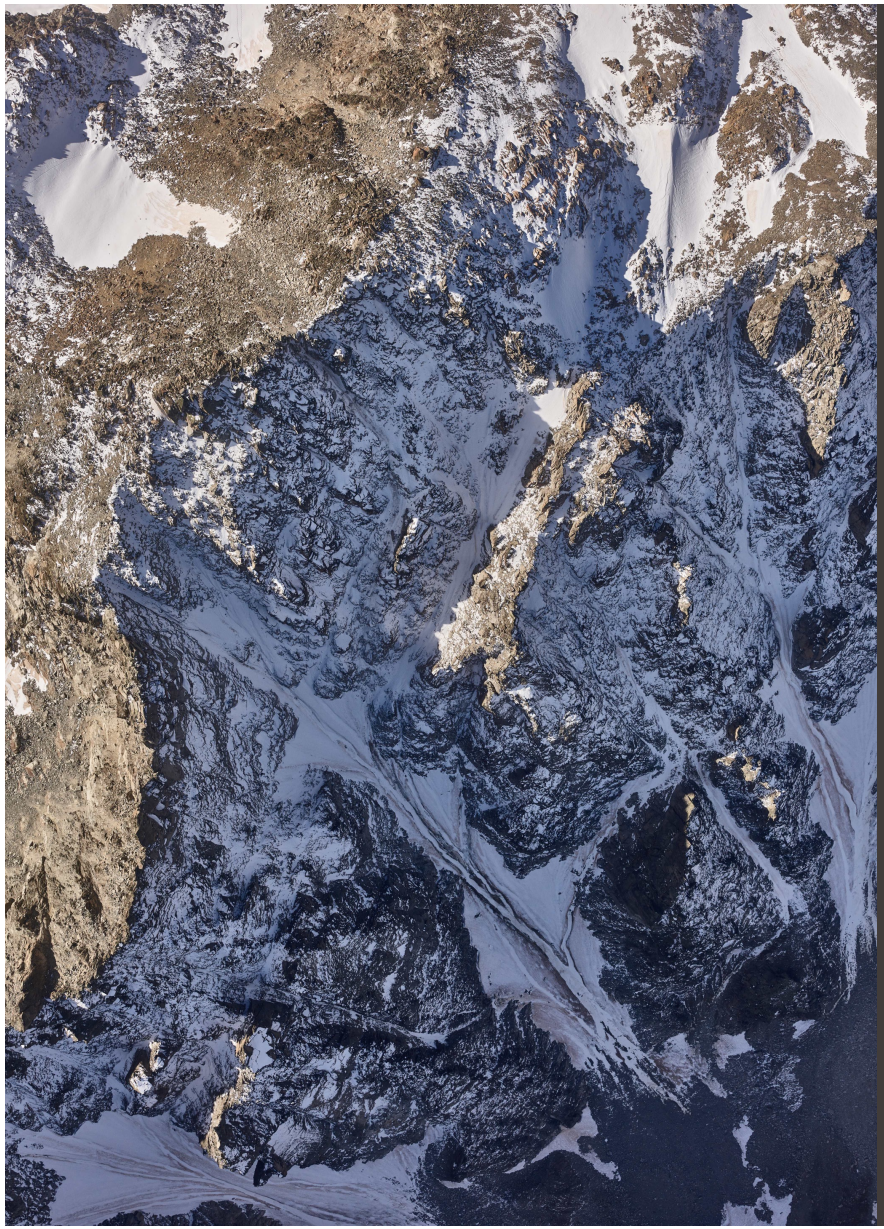
Refinement – orientation & calibration

The orientation parameters (exterior, interior) can be refined by minimizing the cost function:

$$argmin_{K,R,t,a_p} \sum_{i=1}^{n} \|p_i - \Pi(P_i^m, K, R, t, a_p\|_2$$

- Here, lens distortions are included as additional parameters a_p to refine !
- Cost-function *highly* non-linear!
- After DLT, all but a_p have initial values.
- Solving via iterative least-square (linearization).
- **Levenberg-Marquardt** solver: more robust (to local minima) than Gauss-Newton





Perspective Camera Model

DLT algorithm

Refine orientation and calibration

Alternative algorithm

Perspective from n Points (PnP) – Localization (EO)

Task: same as DLT, solve the **pose** of camera in map-frame from a set of 2D-3D point correspondences …

… but *assuming a known camera calibration* !

DLT is possible but sub-optimal (speed, n, accuracy in noise)

Why using PnP?

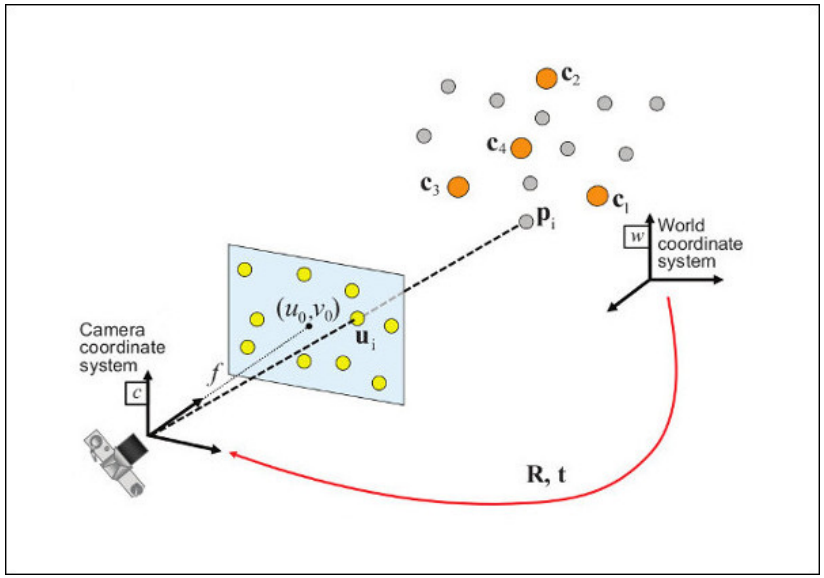
- Stereo-vision (later subject) will provide points in object space
- Then image (camera) moves … slightly & takes an image:
- This new image is localized by previously mapped points!
- Applications (robotics, automated driving, UAV – obstacle avoidance, etc.)

Solutions to perspective *n* point (PnP)

Efficient PnP by Lepetit et al.* and is implemented in OpenCV (solvePnP_EPnP)

$$n \geq 4$$

- EP*n*P expresses n-points (in object frame) as weighted sum of 4 virtual control points (VCPs)
- VCPs' coordinate become the unknown, solvable in $O(n)$ time, solving a constant number of quadratic polynomial equations.
- Final pose of the camera solved from these control points.



Source: OpenCV

* Lepetit et al, EP*n*P: An accurate O(n) solutions to the PnP problem, *Int. J. of Computer Vision*, 2009

DLT vs EPnP

Scenario

Uncalibrated camera (intrinsic w. ap unknown)

- **only DLT** can be used!

Calibrated camera (intrinsic, ap known)

- **either DLT or EPnP** can be used

Minimum number of points

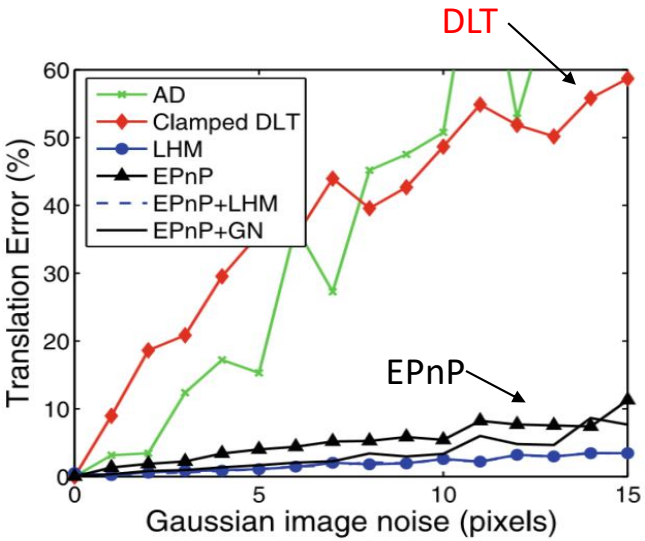
EPnP: **3(P3P)** + 1 for ambiguity resolution

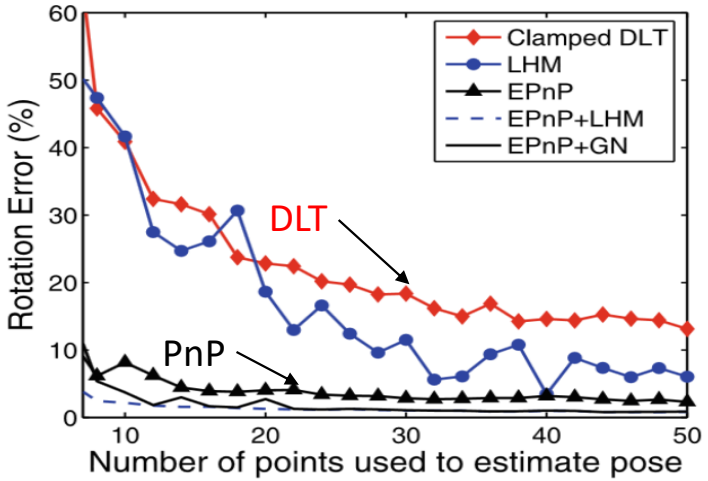
DLT: min. number of points: **4** if coplanar, **6** if non-coplanar

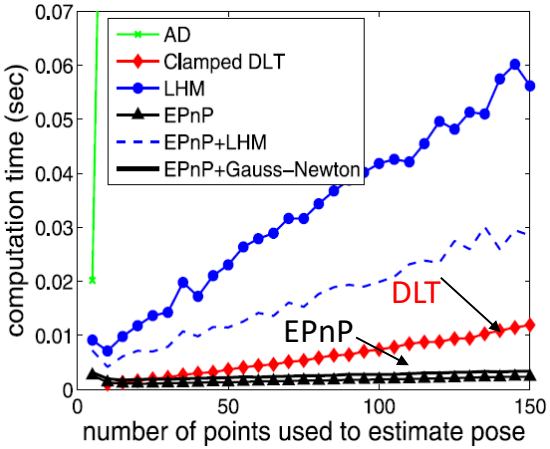
The localization/pose (with intrinsic calibration) can be refined by **non-linear optimization** (min. the sum of squared reprojection errors)

DLT vs EPnP: efficiency & accuracy in noise

- According to Lepetit et al.* EPnP is more than 10x more efficient and supports considerably higher noise levels than DLT:







* Lepetit et al, EP*n*P: An accurate O(n) solutions to the PnP problem, *Int. J. of Computer Vision*, 2009

Understanding - self assessment

- Explain and derive the DLT (via the presented, so-called Tsai's method).
- What is the minimum number of correspondences required for DLT by Tsai's method?
- What is the reprojection error?
- How the reprojection error can be used to improve the orientation?
- Describe the general PnP problem. What is the behavior of its solutions?
- Why and where PnP can be useful?