Robot Vision: Geometric Algorithms – Part 2

Prof. Friedrich Fraundorfer

SS 2023

Outline

- Fundamental matrix properties
- The singularity constraint
- The normalized 8-point algorithm
- The Gold Standard method
- Camera matrices from the essential matrix
- The essential matrix for the stereo case
- Triangulation
- Camera pose estimation and the rotation matrix

Learning goals

- Understand the properties of the fundamental matrix
- Understand the singularity constraint
- Be able to compute a fundamental matrix with enforced singularity constraint
- Understand the normalized 8-point method and the Gold Standard method
- Be able to compute camera poses from an essential matrix
- Understand triangulation
- Understand the geometry of the stereo normal case
- Be able to compute camera matrices with correct rotational part

Fundamental matrix properties

- F is a **unique** 3x3 matrix with **rank 2** (singular, det(F)=0)
- If F is the fundamental matrix for camera matrices (P,P’) then the **transposed** matrix F^T is the fundamental matrix for (P’,P)
- **Epipolar lines** are computed by: $l' = Fx$, $l = F^T x'$
- **Epipoles** are the null-spaces of F. $Fe=0$, $e'^T F=0$
- F has **7 DOF**, i.e. 3x3 matrix – 1 DOF (homogeneous, scale) – 1 DOF (rank 2 constraint)

The singularity constraint of the fundamental matrix

- Other names: Rank 2 constraint, det(F) = 0 constraint
- A valid fundamental matrix has to be singular (non-invertible), which means the determinant det(F)=0.
- In addition, the rank of a valid fundamental matrix has to be 2
- Explanation: $\mathrm{F} = K^{-T}EK^{-1}$=$K^{-T}[t]_x RK^{-1}$
- $[t]_x$ has rank 2 and rank(*AB*)≤min(rank(*A*),rank(*B*)).

- 8-Point algorithm computes all the 9 elements of the F-matrix independently. It is not guaranteed that the rank(F)=2 or det(F)=0

- These properties need to be enforced!

The singularity constraint of the fundamental matrix

- SVD of a linearly computed F-matrix (rank 3):

$$\mathrm{F} = USV^T = U\begin{bmatrix} s_1 & & \\ & s_2 & \\ & & s_3 \end{bmatrix} V^T$$

- Closest rank-2 approximation by setting the last singular value to 0
- Results in the closest matrix under the Frobenius norm $min\|F - F'\|_F$

$$\mathrm{F} = USV^T = U\begin{bmatrix} s_1 & & \\ & s_2 & \\ & & 0 \end{bmatrix} V^T$$

The singularity constraint of the fundamental matrix

- Example:

A = (8x9)

1	3	5	3	4	3	7	4	1
2	9	4	2	8	2	5	7	5
7	3	6	6	8	9	9	6	5
5	7	6	5	3	6	2	7	8
2	2	7	5	6	5	1	9	5
4	3	6	6	6	6	1	9	4
6	1	9	7	5	5	1	2	7
2	6	2	4	8	6	4	2	7

F =

0.0012818033647169	-0.195296914367969	-0.404026958783203
0.592627190886001	-0.0992048118304505	-0.505391799650038
0.244770293871894	0.181983926946307	0.298529042380632

rank(F)=3

S =

0.853380835370105	0	0
0	0.521146237658923	0
0	0	0.0121551962950181

S_ =

0.853380835370105	0	0
0	0.521146237658923	0
0	0	**0**

F_ = U*S_*V^T

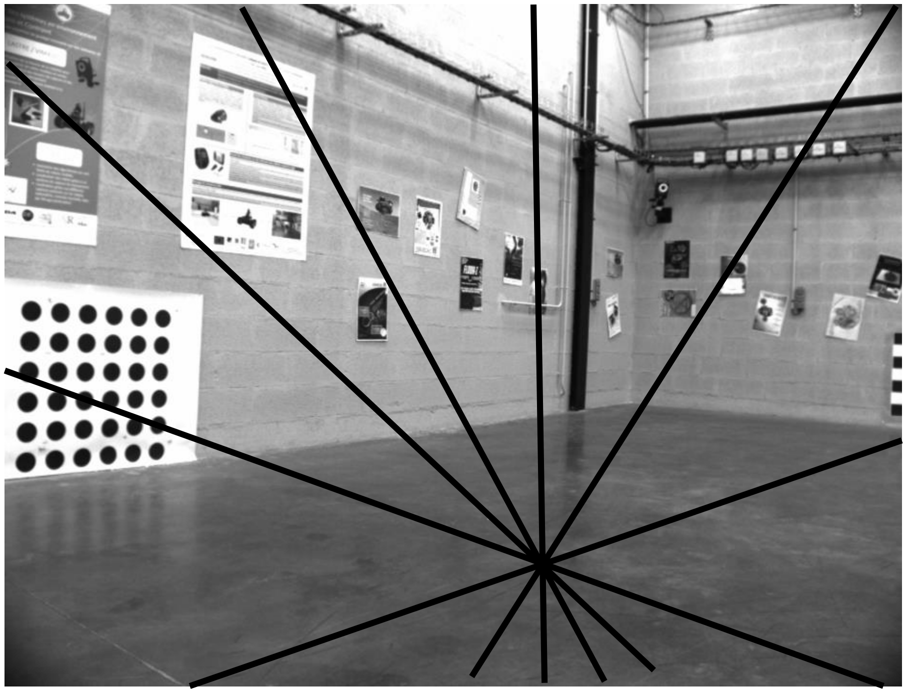
-0.000493883737627127	-0.187153153340858	-0.407762597079129
0.59321760922536	-0.101912623277308	-0.504149694914234
0.243327284554864	0.188601941472783	0.29549328182407

rank(F)=2

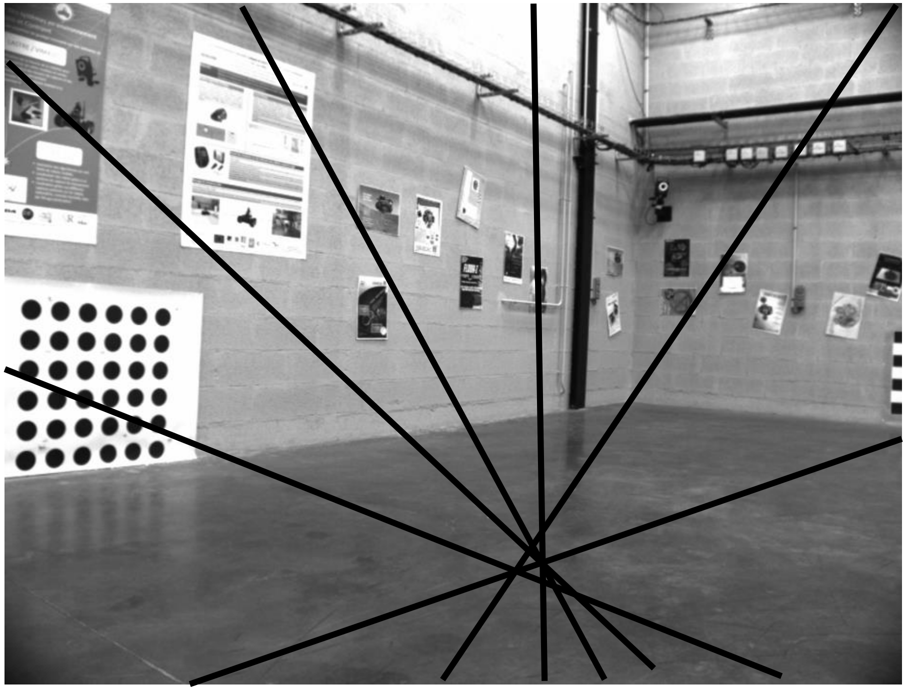
norm(F-F_)=0.0121

The singularity constraint of the fundamental matrix

- Does it make a difference?



Epipolar lines from corrected F-matrix



Epipolar lines from not corrected F-matrix
Epipolar lines don‘t intersect

The normalized 8-point algorithm

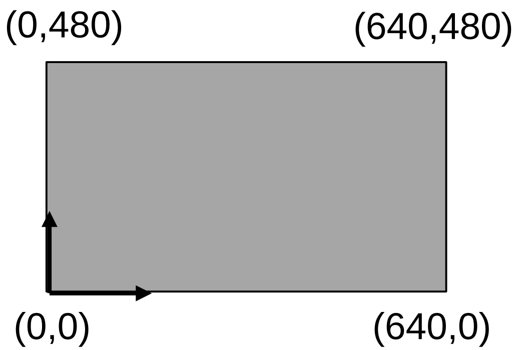
- Solving the fundamental matrix equation system using pixel coordinates can give bad results (due to numerical instabilities)
- Solution: Normalization of pixel coordinates

Algorithm:

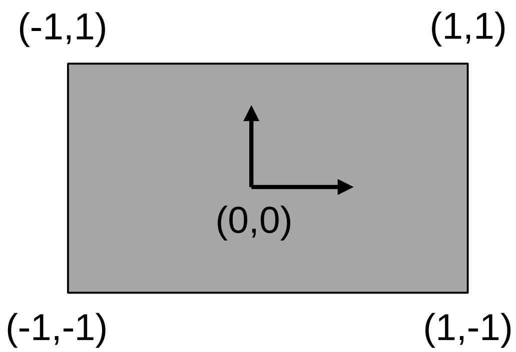
1. Transform the coordinates such that the image center is at (0,0) and that the maximum distance from the origin is $\sqrt{2}$
2. Compute Fn using the 8-point method from the normalized points
3. Enforce the singularity constraints
4. Transform the fundamental matrix back to original units

The normalized 8-point algorithm

- Example: Transform image coordinates to [-1,-1]x[1,1]



$$K = \begin{bmatrix} \frac{2}{640} & 0 & -1 \\ 0 & \frac{2}{480} & -1 \\ 0 & 0 & 1 \end{bmatrix}$$



- Transformation K is like a calibration matrix
- F = K^TFnK

The Gold Standard method

- Accurate solution using non-linear optimization

1. Compute an initial estimate for $\hat{F}$ using the normalized 8-point algorithm (enforcing rank 2 constraint)
2. Extract cameras P and P‘ from $\hat{F}$

$$P = [I|0] \qquad P' = [[e'\ ]_x \hat{F}|e']$$

3. Triangulate 3D points from point correspondences
4. Use non-linear optimization e.g. Levenberg-Marquardt to minimize the reprojection error

$$\sum_i d(x_i\ ,\hat{x}_i\ )^2 + d(x_i',\hat{x}_i')^2$$

by optimizing the parameters of P and P‘ and the 3D points.

Camera matrices from Essential matrix

- Essential matrix represents the relative motion between two cameras
- Camera matrices P and P‘ can be computed from E

$$E = [t]_x R \qquad \mathrm{E} = K^T F K$$
$$P = [\mathrm{I}\ 0]$$
$$P' = [\mathrm{R}\ \mathrm{t}]$$

- R and $[t]_x$ can be computed using the SVD of E

$$USV^T = svd(E)$$

$$R = U \begin{bmatrix} 0 & \mp 1 & 0 \\ \pm 1 & 0 & 0 \\ 0 & 0 & 1 \end{bmatrix} V^T \qquad [t]_x = U \begin{bmatrix} 0 & \pm 1 & 0 \\ \mp 1 & 0 & 0 \\ 0 & 0 & 0 \end{bmatrix} U^T$$

$$[t]_x = \begin{bmatrix} 0 & -t_z & t_y \\ t_z & 0 & -t_x \\ -t_y & t_x & 0 \end{bmatrix}$$

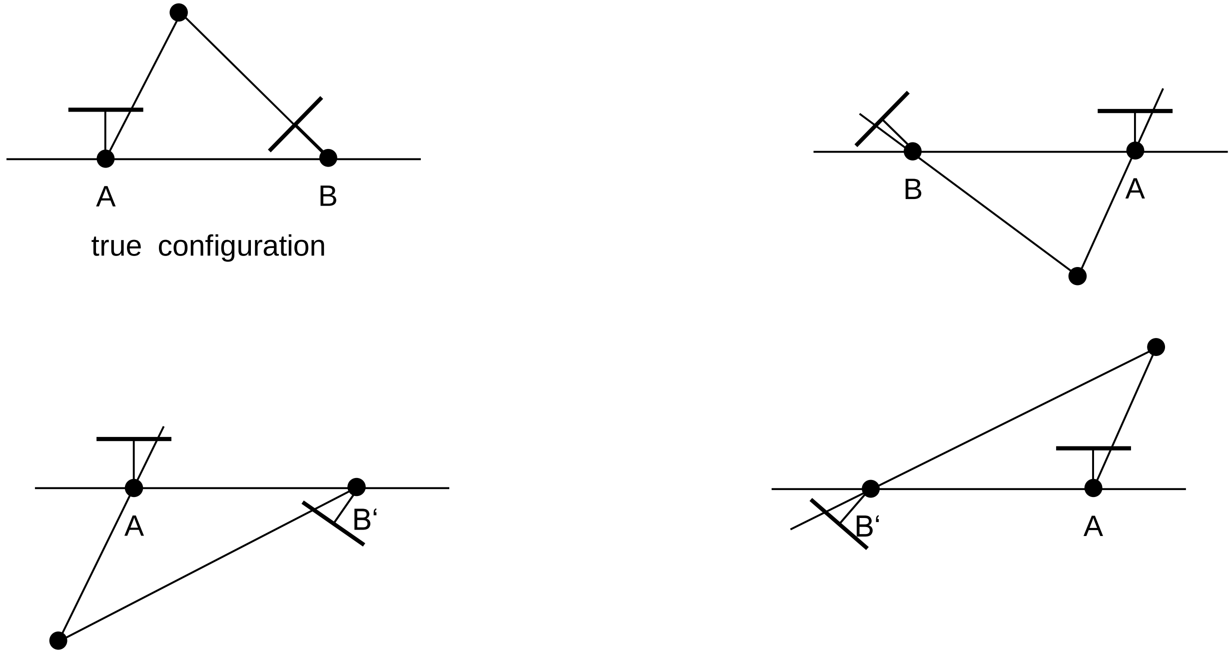
- 4 possible combinations of R and t

Camera matrices from Essential matrix

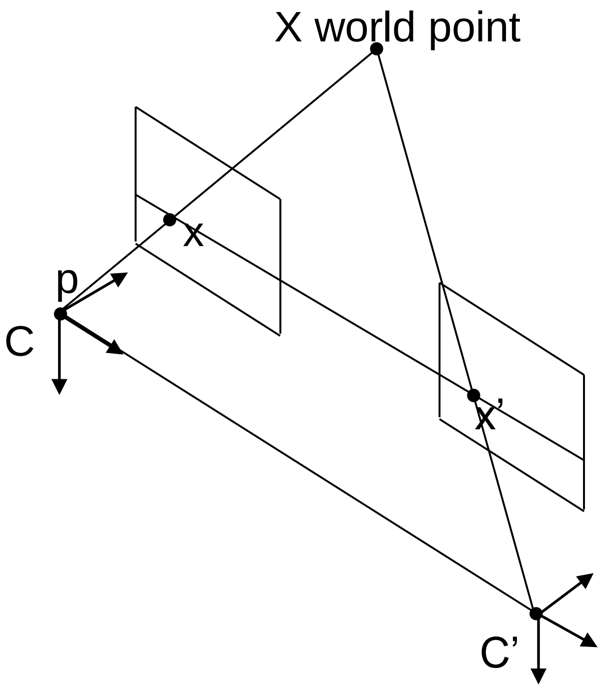
- P is set as the canonical coordinate system at the origin, $\|t\| = 1$

$$P = [\mathrm{I}\ 0] \qquad P' = [\mathrm{R}\ \mathrm{t}]$$

- Only for one of the 4 configurations the image rays intersect in front of the cameras.
- This is the true configuration and can be found by triangulating points

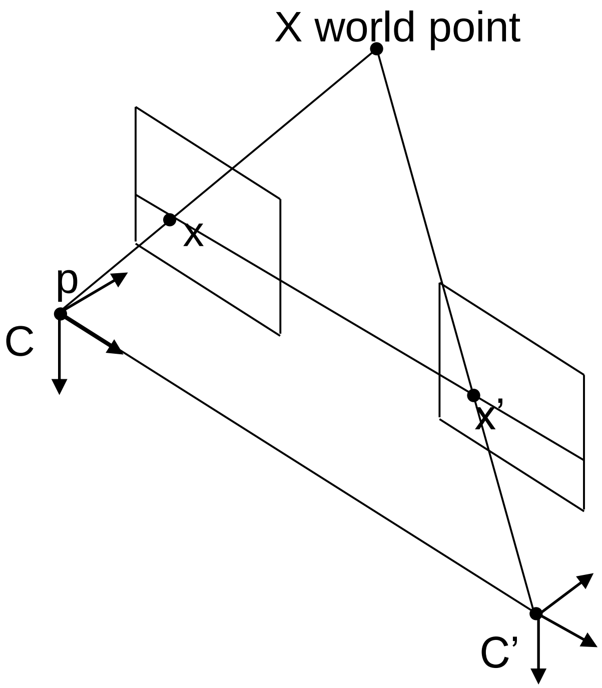


The essential matrix for the stereo case



$$R = I_{3x3} \qquad T = [T_x \quad 0 \quad 0]^T$$

The essential matrix for the stereo case



$$R = I_{3x3} \qquad T = [T_x \quad 0 \quad 0]^{T}$$

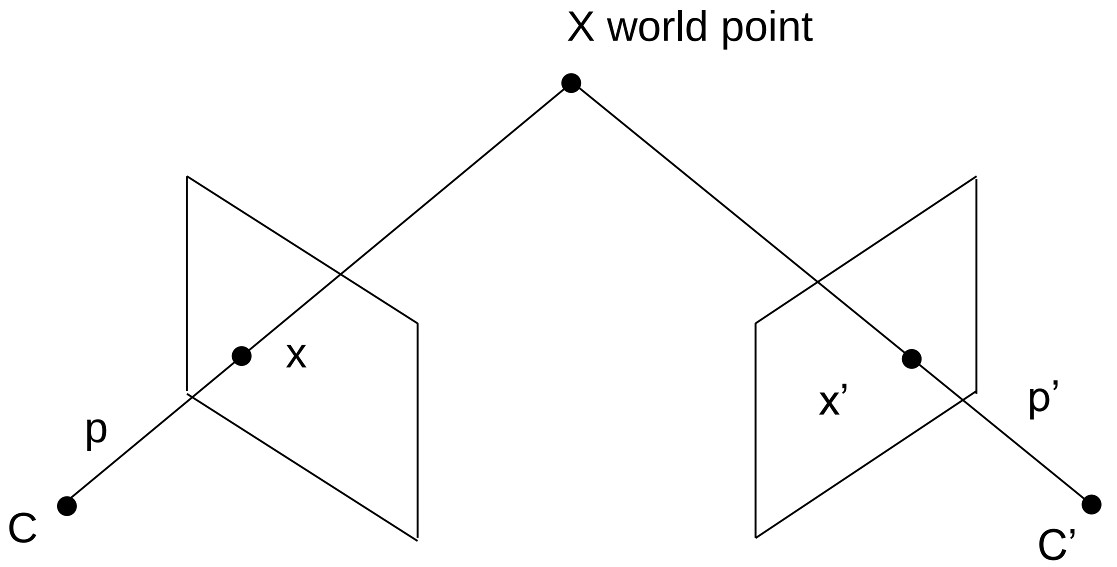
$$E = [T]_x R = \begin{bmatrix} 0 & 0 & 0 \\ 0 & 0 & -T_x \\ 0 & T_x & 0 \end{bmatrix}$$

$$[x' \quad y' \quad 1] \begin{bmatrix} 0 & 0 & 0 \\ 0 & 0 & -T_x \\ 0 & T_x & 0 \end{bmatrix} \begin{bmatrix} x \\ y \\ 1 \end{bmatrix} = 0$$

$$[x' \quad y' \quad 1] \begin{bmatrix} 0 \\ -T_x \\ T_x y \end{bmatrix} = 0$$

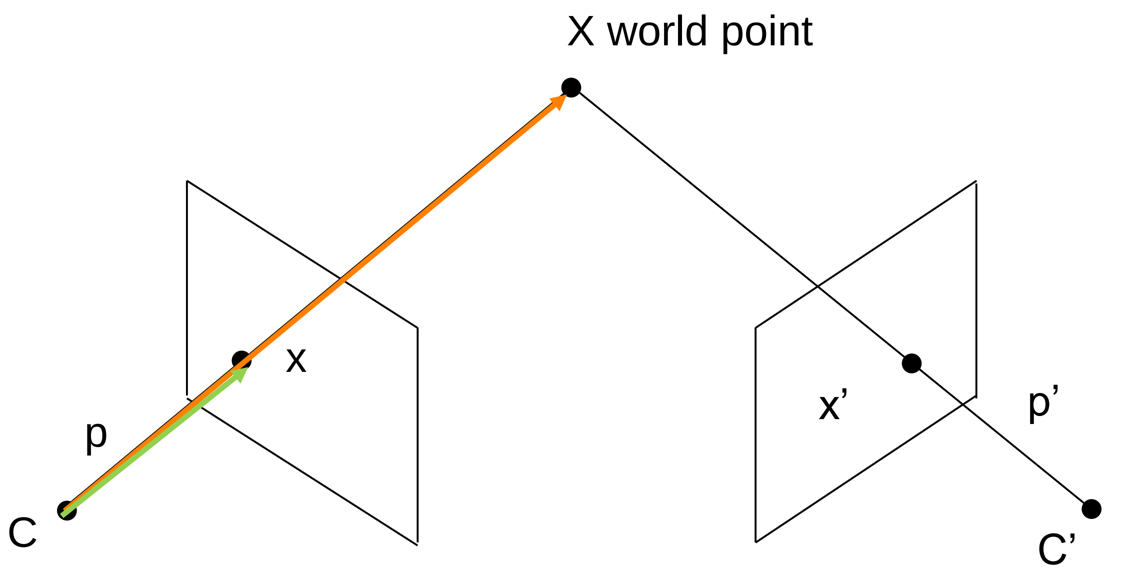
$$-y' T_x + T_x y = 0$$

Triangulation



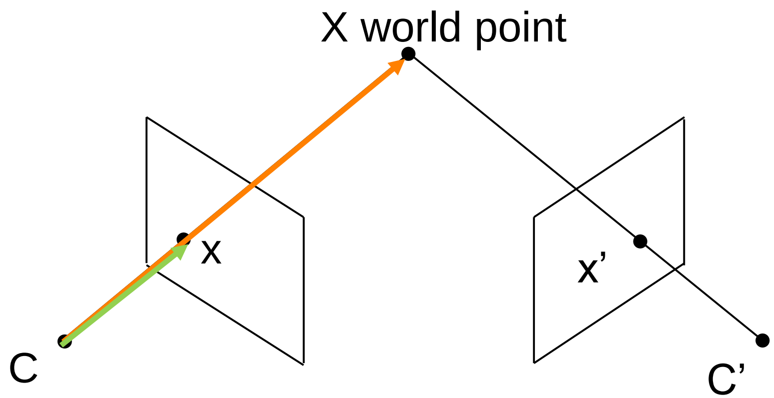
- Compute coordinates of world point X given the measurements x, x' and the camera projection matrices P and P'

Triangulation



- Condition: Measurement vector x needs to have the same direction as projection of X (cross-product equals 0)
- $x \times (PX) = 0 \; and \; x' \times (P'X) = 0$
- Can be rewritten into equation system $AX = 0$ to solve for X

Triangulation

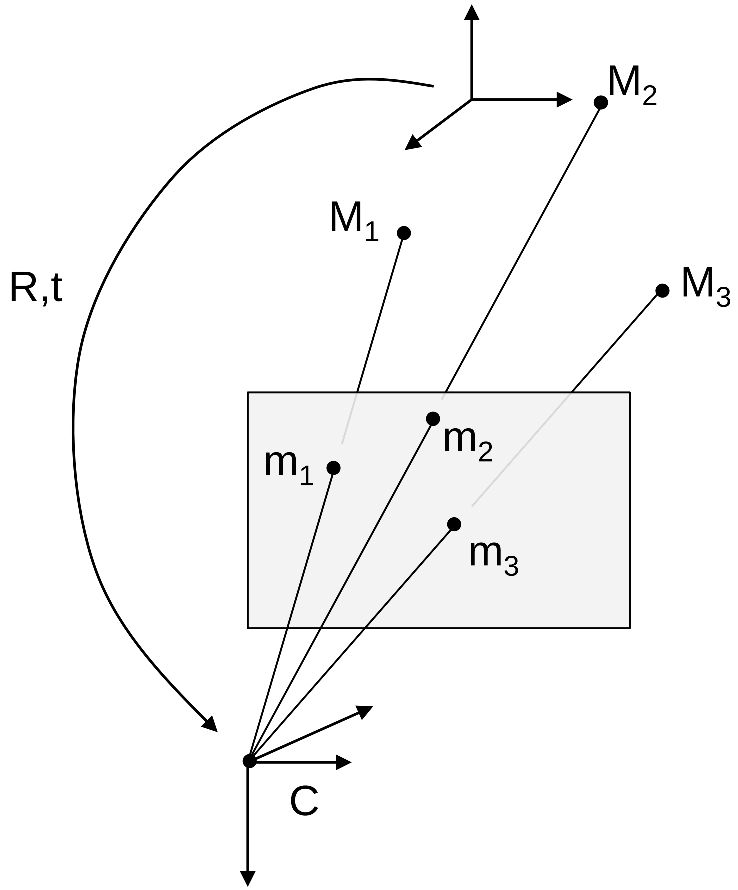


$$x \times (PX) = 0 \; and \; x' \times (P'X) = 0$$
$$x(P_3^T X) - (P_1^T X) = 0$$
$$y(P_3^T X) - (P_2^T X) = 0$$
$$x(P_2^T X) - y(P_1^T X) = 0$$

$$P = \begin{bmatrix} P_1^T \\ P_2^T \\ P_3^T \end{bmatrix}$$

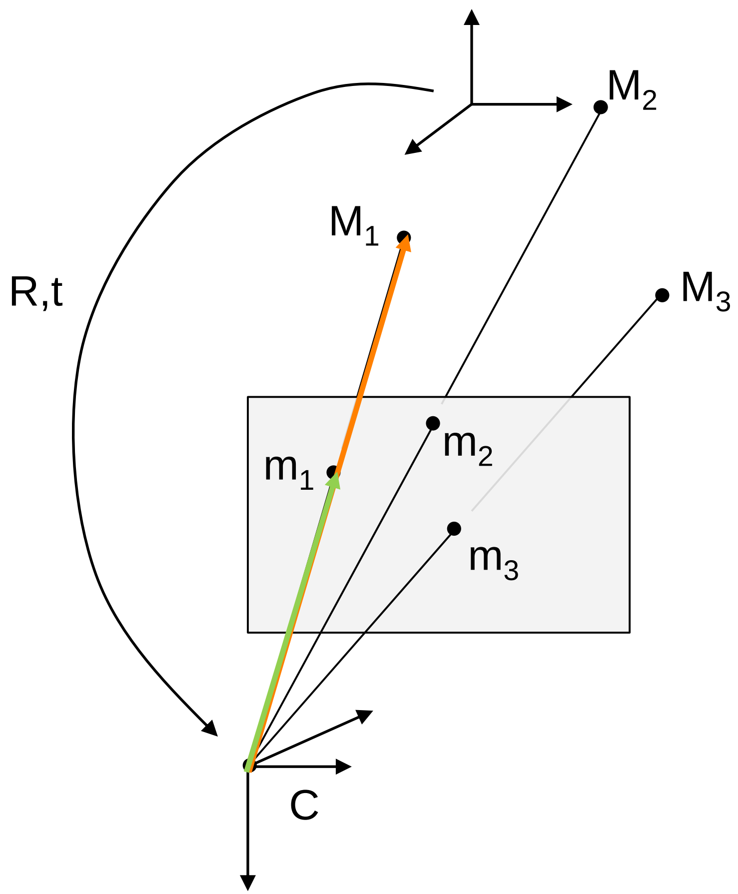
$$\begin{bmatrix} xP_3^T - P_1^T \\ yP_3^T - P_2^T \\ x'P'^T_3 - P'^T_1 \\ y'P'^T_3 - P'^T_2 \end{bmatrix} X = 0$$

Camera pose estimation



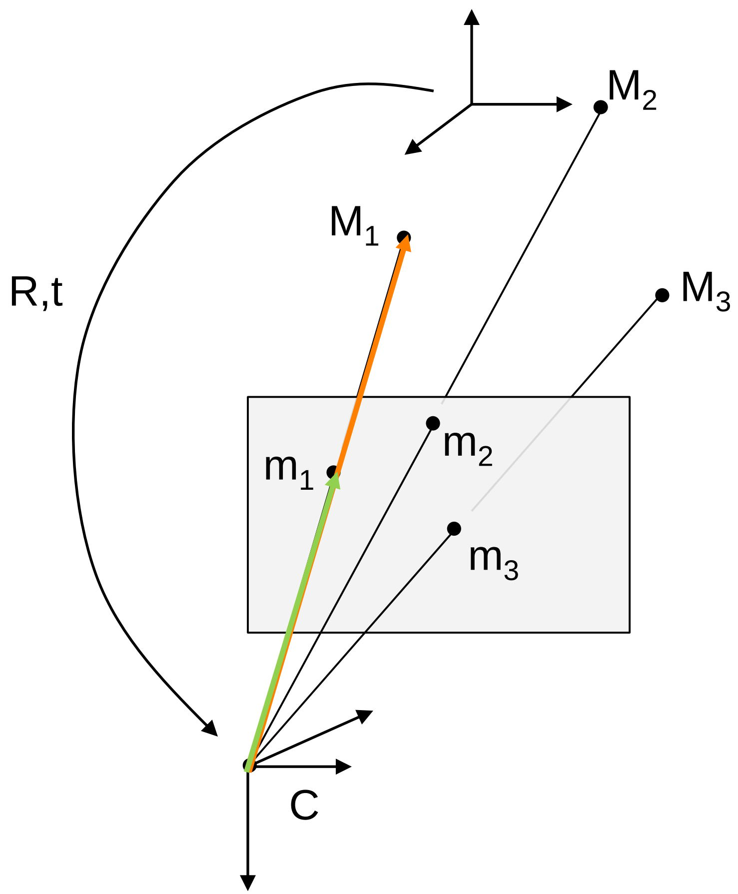
- perspective-n-point problem
- Goal is to estimate camera matrix P such that $m_1 = PM_1$
- $m_1, M_1, m_2, M_2, m_3, M_3$ are known
- Algebraic linear solution requires 6 3D-2D point correspondences, minimal nonlinear solution requires only 3

Camera pose estimation



- Derivation similar to Triangulation, but now entries of P are the unknowns instead of X
- Condition: Measurement vector x needs to have the same direction as projection of X (cross-product equals 0)

Camera pose estimation



- Derivation similar to Triangulation, but now entries of P are the unknowns instead of X
- Condition: Measurement vector x needs to have the same direction as projection of X (cross-product equals 0)

$$x \times (PX) = 0 \; for \; all \; pairs \; x \leftrightarrow X$$
$$y(P_3^T X) - w(P_2^T X) = 0$$
$$x(P_3^T X) - w(P_1^T X) = 0$$
$$x(P_2^T X) - y(P_1^T X) = 0$$

$$\begin{bmatrix} 0 & -wX^T & yX^T \\ -wX^T & 0 & xX^T \\ -yX^T & xX^T & 0 \end{bmatrix} \begin{bmatrix} P_1 \\ P_2 \\ P_3 \end{bmatrix} = 0$$

Camera pose estimation

- Linear camera pose estimation does **not** enforce inner constraints

$$P = \begin{bmatrix} p_{11} & p_{12} & p_{13} & p_{14} \\ p_{21} & p_{22} & p_{23} & p_{24} \\ p_{31} & p_{32} & p_{33} & p_{34} \end{bmatrix}$$

$$P = \begin{bmatrix} R_{11} & R_{12} & R_{13} & p_{14} \\ R_{21} & R_{22} & R_{23} & p_{24} \\ R_{31} & R_{32} & R_{33} & p_{34} \end{bmatrix}$$

- R is a 3x3 rotation matrix
- Elements of R are **not independent** of each other
- Rotation matrices belong to the matrix group SO(3)

$$R^T R = I, \det(R) = +1$$

Special orthogonal group SO(n)

- The set of all the nxn orthogonal matrices with determinant equal to +1 is a group w.r.t. the matrix multiplication:

$$SO(n) = (\{A \in O(n) |\, \det(A) = +1\}, \times)$$

Special orthogonal group

- SO(3) … group of orthogonal 3x3 matrices with det=+1 …. “rotation matrices”

- $R_3 = R_1 * R_2$ … R_3 is still an SO(3) element
- $R_3 = R_1 + R_2$ … R_3 is **NOT** an SO(3) element. Not a rotation matrix anymore.

Problems with rotation matrices

- Optimization of rotations (bundle adjustment)
  - Newton's method $x_{n+1} = x_n - \frac{f(x_n)}{f'(x_n)}$

- Filtering and averaging, e.g. $R' = \frac{R_1+R_2}{2}$ not allowed
  - E.g. averaging rotation from IMU or camera pose tracker for AR/VR glasses

Enforcing the rotation matrix constraint

- After estimating the camera matrix $\hat{P}$ it can be replaced with the closest P that consists of a valid rotational part.
- P = [R | t], where $R^T R = I, \det(R) = +1$
- Such a P can be found using SVD.

$$\hat{P} = [M \quad t]$$
$$USV = svd(M)$$
$$R = UV^T$$
$$P = [R \quad t]$$

Recap - Learning goals

- Understand the properties of the fundamental matrix
- Understand the singularity constraint
- Be able to compute a fundamental matrix with enforced singularity constraint
- Understand the normalized 8-point method and the Gold Standard method
- Be able to compute camera poses from an essential matrix
- Understand triangulation
- Understand the geometry of the stereo normal case
- Be able to compute camera matrices with correct rotational part