Computer Vision

Direct Solutions for Computing Fundamental and Essential Matrix

Arjun Jain

Course Project Submissions

- **Good news:** No report required!
- **Not so good news:** You are required to create a github page with the code just like our course webpage. The template for the same will be given out by the TA's shortly.
- Needs to be done before 5^{th} of May (review day)
- We will then link to your project pages from the course webpage

Motivation

F/E $R, \mathbf{b}$ Image courtesy: Collins

Topics of Today's Lecture

Compute the

- **Fundamental matrix given** corresponding points
- **Essential matrix given** corresponding points

Table of Contents

1
Computing the Fundamental Matrix Given Corresponding Points

Fundamental Matrix (Uncalibrated Cameras)

- The **fundamental matrix** F **is**

$$\boxed{\mathsf{F} \quad = \quad (\mathsf{K}')^{-\mathsf{T}} R' \mathsf{S}_b R''^{\mathsf{T}} (\mathsf{K}'')^{-1}}$$

- It encodes the relative orientation for two uncalibrated cameras
- **Coplanarity constraint** through F

$$\boxed{\mathbf{x}'^{\mathsf{T}} \mathsf{F} \mathbf{x}'' \quad = \quad 0}$$

Fundamental Matrix

The fundamental matrix F can directly be computed if we know the

- $\mathsf{K}', \mathsf{K}''$ calibration matrices
- R', R'' viewing direction of the cameras
- S_b baseline
- or the projection matrices $\mathsf{P}', \mathsf{P}''$

How to compute F given ONLY corresponding points in images?

Problem Formulation

- **Given:** *N* corresponding points

$$(x', y')_n, (x'', y'')_n \quad \text{with} \quad n = 1, \ldots, N$$

- **Wanted:** fundamental matrix F

Fundamental Matrix From Corresponding Points

- For each point, we have the coplanarity constraint

$$\mathbf{x}'^{\mathsf{T}}_n \mathsf{F} \mathbf{x}''_n = 0 \qquad n = 1, ..., N$$

Fundamental Matrix From Corresponding Points

- For each point, we have the coplanarity constraint

$$\mathbf{x}'^{\mathsf{T}}_n \mathsf{F} \mathbf{x}''_n = 0 \qquad n = 1, ..., N$$

- or

$$[x'_n, y'_n, 1] \begin{bmatrix} \boxed{\begin{matrix} F_{11} & F_{12} & F_{13} \\ F_{21} & F_{22} & F_{23} \\ F_{31} & F_{32} & F_{33} \end{matrix}} \end{bmatrix} \begin{bmatrix} x''_n \\ y''_n \\ 1 \end{bmatrix} = 0$$

unknowns!

Linear Dependency

- **Linear function** in the unknowns F_{ij}

$$[\boxed{x'_n}, y'_n, 1] \begin{bmatrix} \boxed{F_{11}} & F_{12} & F_{13} \\ F_{21} & F_{22} & F_{23} \\ F_{31} & F_{32} & F_{33} \end{bmatrix} \begin{bmatrix} \boxed{x''_n} \\ y''_n \\ 1 \end{bmatrix} = 0$$

$$\boxed{x''_n F_{11} x'_n} + x''_n F_{21} y'_n + \ldots = 0$$

Linear Dependency

- **Linear function** in the unknowns F_{ij}

$$[x'_n \; \boxed{y'_n} \; 1] \begin{bmatrix} F_{11} & F_{12} & F_{13} \\ \boxed{F_{21}} & F_{22} & F_{23} \\ F_{31} & F_{32} & F_{33} \end{bmatrix} \begin{bmatrix} \boxed{x''_n} \\ y''_n \\ 1 \end{bmatrix} = 0$$

$$x''_n F_{11} x'_n + \boxed{x''_n F_{21} y'_n} + \ldots = 0$$

Linear Dependency

- **Linear function** in the unknowns F_{ij}

$$[x'_n, y'_n, \boxed{1}] \begin{bmatrix} F_{11} & F_{12} & F_{13} \\ F_{21} & F_{22} & F_{23} \\ \boxed{F_{31}} & F_{32} & F_{33} \end{bmatrix} \begin{bmatrix} \boxed{x''_n} \\ y''_n \\ 1 \end{bmatrix} = 0$$

$$x''_n F_{11} x'_n + x''_n F_{21} y'_n + \boxed{.}\,.\,. = 0$$

Linear Dependency

- **Linear function** in the unknowns F_{ij}

$$[x'_n, y'_n, 1] \begin{bmatrix} F_{11} & F_{12} & F_{13} \\ F_{21} & F_{22} & F_{23} \\ F_{31} & F_{32} & F_{33} \end{bmatrix} \begin{bmatrix} x''_n \\ y''_n \\ 1 \end{bmatrix} = 0$$

$$\Downarrow$$

$$[x''_n x'_n, x''_n y'_n, x''_n, y''_n x'_n, y''_n y'_n, y''_n, x'_n, y'_n, 1] \cdot$$
$$[F_{11}, F_{21}, F_{31}, F_{12}, F_{22}, F_{32}, F_{13}, F_{23}, F_{33}] = 0$$
$$n = 1, ..., N$$

Linear Dependency

- Linear function in the unknowns F_{ij}

$$\boldsymbol{a}_n^\mathsf{T} \rightarrow [x''_n x'_n, x''_n y'_n, x''_n, y''_n x'_n, y''_n y'_n, y''_n, x'_n, y'_n, 1] \cdot$$

$$\mathbf{f}^\mathsf{T} \rightarrow [F_{11}, F_{21}, F_{31}, F_{12}, F_{22}, F_{32}, F_{13}, F_{23}, F_{33}] = 0$$

$$n = 1, ..., N$$

$$\boldsymbol{a}_n^\mathsf{T} \cdot \mathbf{f}^\mathsf{T} = 0 \qquad n = 1, ..., N$$

Using the Kronecker Product

- Linear function in the unknowns F_{ij}

$$\boldsymbol{a}_n^\mathsf{T} \longrightarrow [x''_n x'_n, x''_n y'_n, x''_n, y''_n x'_n, y''_n y'_n, y''_n, x'_n, y'_n, 1] \cdot$$

$$\mathbf{f}^\mathsf{T} \longrightarrow [F_{11}, F_{21}, F_{31}, F_{12}, F_{22}, F_{32}, F_{13}, F_{23}, F_{33}] = 0$$

$$n = 1, ..., N$$

$$(\mathbf{x}''_n \otimes \mathbf{x}'_n)^\mathsf{T} \mathrm{vec}\mathsf{F} = \underbrace{\boldsymbol{a}_n^\mathsf{T}}_{(\mathbf{x}''_n \otimes \mathbf{x}'_n)^\mathsf{T}} \underbrace{\mathbf{f}}_{\mathrm{vec}\mathsf{F}} = 0 \qquad n = 1, ..., N$$

(it holds in general: $\mathbf{x}^\mathsf{T} \mathsf{F} \mathbf{y} = (\mathbf{y} \otimes \mathbf{x})^\mathsf{T} \mathrm{vec}\mathsf{F}$)

Linear System From All Points

- We directly obtain a linear system if we consider **all N points**

$$\underbrace{\boldsymbol{a}_n^\mathsf{T}}_{(\mathbf{x}''_n \otimes \mathbf{x}'_n)^\mathsf{T}} \underbrace{\mathbf{f}}_{\mathrm{vec}\mathsf{F}} = 0 \qquad n = 1, \ldots, N$$

$$A = \begin{bmatrix} \boldsymbol{a}_1^\mathsf{T} \\ \ldots \\ \boldsymbol{a}_n^\mathsf{T} \\ \ldots \\ \boldsymbol{a}_N^\mathsf{T} \end{bmatrix} \Rightarrow A\mathbf{f} = \mathbf{0}$$

So how to solve such a system of equations?

Solving the Linear System

- Singular value decomposition solves

$$A\mathbf{f} = \mathbf{0}$$

- and thus provides a solution for

$$\mathbf{f} = [F_{11}, F_{21}, F_{31}, F_{12}, F_{22}, F_{32}, F_{13}, F_{23}, F_{33}]^{\mathsf{T}}$$

- SVD: $\mathbf{f}$ can be characterized as a right-singular vector corresponding to a singular value of A that is zero

How Many Points Are Needed?

- The vector $\mathbf{f}$ has 9 dimensions…

$$A = \begin{bmatrix} \boldsymbol{a}_1^\top \\ \cdots \\ \boldsymbol{a}_n^\top \\ \cdots \\ \boldsymbol{a}_N^\top \end{bmatrix} \Rightarrow A\mathbf{f} = \mathbf{0}$$

How Many Points Are Needed?

- The vector $\mathbf{f}$ has 9 dimensions

$$A = \begin{bmatrix} \boldsymbol{a}_1^\top \\ \dots \\ \boldsymbol{a}_n^\top \\ \dots \\ \boldsymbol{a}_N^\top \end{bmatrix} \Rightarrow A\mathbf{f} = \mathbf{0}$$

- Fundamental matrix is **homogenous**
- Matrix A has a rank of at most **8**
- **We need ? corresponding points**

How Many Points Are Needed?

- The vector $\mathbf{f}$ has 9 dimensions

$$A = \begin{bmatrix} \boldsymbol{a}_1^\top \\ \dots \\ \boldsymbol{a}_n^\top \\ \dots \\ \boldsymbol{a}_N^\top \end{bmatrix} \Rightarrow A\mathbf{f} = \mathbf{0}$$

- Fundamental matrix is **homogenous**
- Matrix A has a rank of at most **8**
- **We need 8 corresponding points**

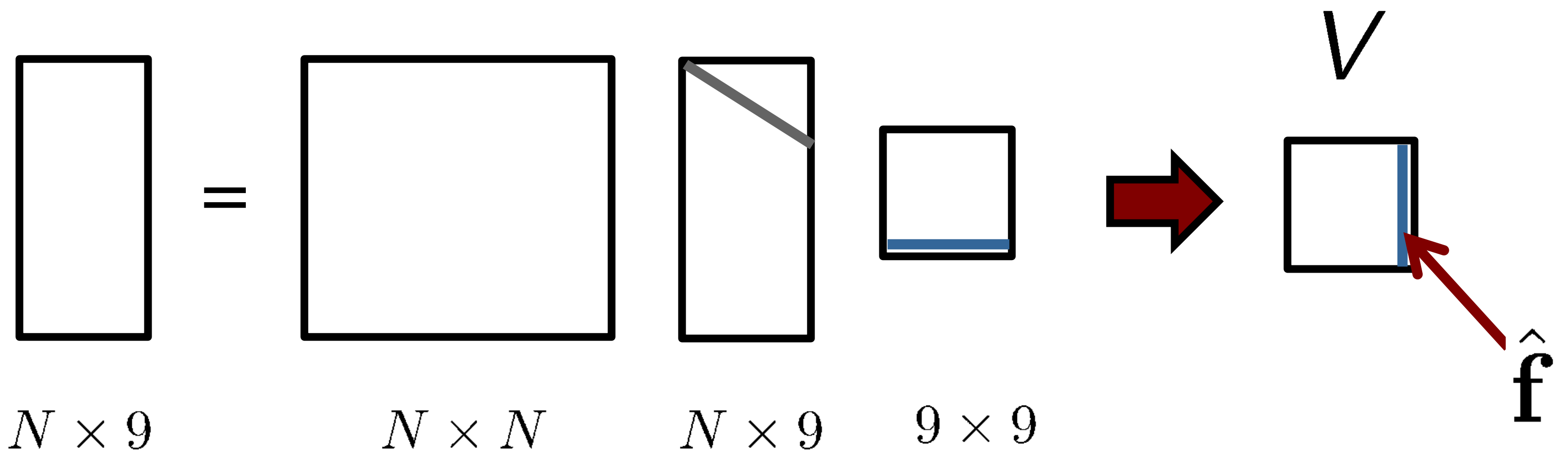
More Than 8 Points...

- In reality: noisy measurements
- With more than 8 points, the matrix A will become regular (but should not!)
- Use the singular vector $\hat{\mathbf{f}}$ of A that corresponds to the **smallest** singular value is the solution $\hat{\mathbf{f}} \to \hat{\mathsf{F}}$

Singular Vector

- Use the singular vector $\hat{\mathbf{f}}$ of A that corresponds to the **smallest** singular value is the solution $\hat{\mathbf{f}} \rightarrow \hat{\mathsf{F}}$

$$A = UDV^{\mathsf{T}}$$



8-Point Algorithm 1st Try

```
function F = F_from_point_pairs(xs, xss)
% xs, xss: Nx3 homologous point coordinates, N > 7
% F:       3x3 fundamental matrix

% coefficient matrix
for n = 1 : size(xs, 1)
    A(n, :) = kron(xss(n, :), xs(n, :));
end

```

8-Point Algorithm 1st Try

```
function F = F_from_point_pairs(xs, xss)
% xs, xss: Nx3 homologous point coordinates, N > 7
% F:        3x3 fundamental matrix

% coefficient matrix
for n = 1 : size(xs, 1)
    A(n, :) = kron(xss(n, :), xs(n, :));
end

% singular value decomposition
[U, D, V] = svd(A);

% select the singlar vector with the minimal singular value
F = reshape(V(:, 9), 3, 3)';
```

singular vector of the smallest singular value

Not necessarily a matrix of rank 2
(but F should have: rank(F)=2)

Enforcing Rank 2

- We want to enforce a matrix F with $\mathrm{rank}(\mathsf{F}) = 2$ **Why?**
- F should approximate our computed matrix $\hat{\mathsf{F}}$ as close a possible

What to do?

Enforcing Rank 2

- We want to **enforce** a matrix F with $\operatorname{rank}(\mathsf{F}) = 2$
- F should **approximate** our computed matrix $\hat{\mathsf{F}}$ as close a possible
- Use a second SVD (this time of $\hat{\mathsf{F}}$)

$$\mathsf{F} = UD^a V^\mathsf{T} = U\operatorname{diag}(D_{11}, D_{22}, 0)V^\mathsf{T}$$

with $\operatorname{svd}(\hat{\mathsf{F}}) = UDV^\mathsf{T}$
and $D_{11} \geq D_{22} \geq D_{33}$

8-Point Algorithm

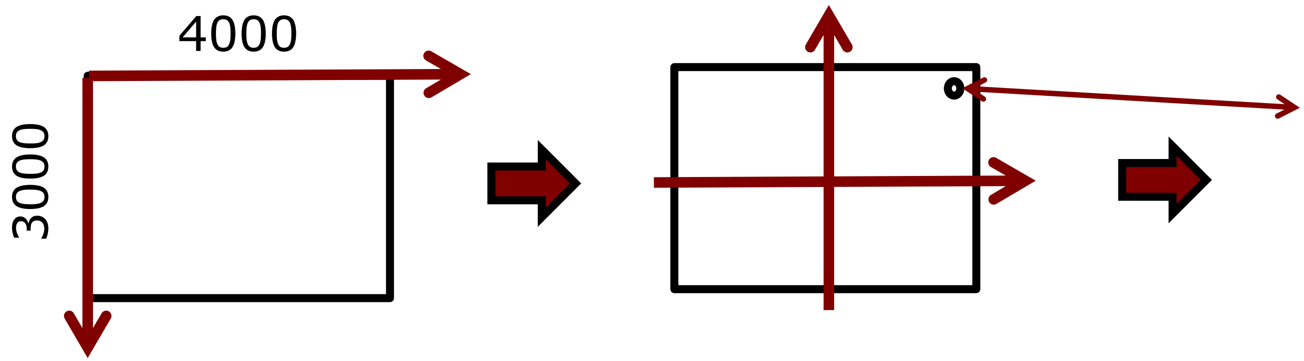
```
function F = F_from_point_pairs(xs, xss)
% xs, xss: Nx3 homologous point coordinates, N > 7
% F:        3x3 fundamental matrix

% coefficient matrix
for n = 1 : size(xs, 1)
    A(n, :) = kron(xss(n, :), xs(n, :));
end

% singular value decomposition
[U, D, V] = svd(A);

% approximate F, possibly regular
Fa = reshape(V(:, 9), 3, 3)';

```

8-Point Algorithm

```
function F = F_from_point_pairs(xs, xss)
% xs, xss: Nx3 homologous point coordinates, N > 7
% F:       3x3 fundamental matrix

% coefficient matrix
for n = 1 : size(xs, 1)
    A(n, :) = kron(xss(n, :), xs(n, :));
end

% singular value decomposition
[U, D, V] = svd(A);

% approximate F, possibly regular
Fa = reshape(V(:, 9), 3, 3)';

% svd decomposition of F
[Ua, Da, Va] = svd(Fa);

% algebraically best F, singular
F = Ua * diag([Da(1, 1), Da(2, 2), 0]) * Va';
```

Well-Conditioned Problem

- Example image 12MPixel camera



$$\begin{bmatrix} 1800 \\ 1400 \\ 1 \end{bmatrix}$$

- Ill-conditioned, numerically instable

$$\begin{bmatrix} 1800 \\ 1400 \\ 1 \end{bmatrix}$$ ☹ ➡ new coordinate system ➡ $$\begin{bmatrix} 0.9 \\ 0.7 \\ 1 \end{bmatrix}$$ ☺

When did you have to do something similar to this?

Conditioning/Normalization to Obtain a Well-Conditioned Problem

- Normalization of the point coordinates substantially **improves** the **stability**
- **Transform** the points so that the center of mass of all points is at (0,0)
- **Scale** the image so that the x and y coordinated are within [-1,1]

☺

Conditioning/Normalization

- Define $\mathsf{T} : \mathsf{T}\mathbf{x} = \hat{\mathbf{x}}$ so that coordinates are zero-centered and in [-1,1]
- Determine fundamental matrix $\hat{\mathsf{F}}$ from the transformed coordinates

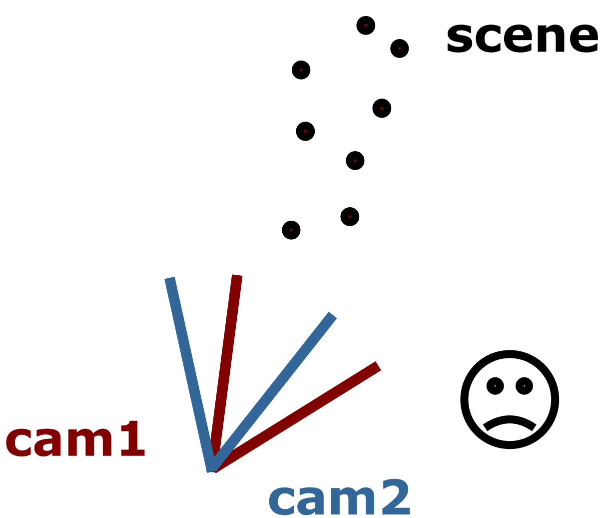
$$\begin{aligned} {\mathbf{x}'}^{\mathsf{T}} \mathsf{F} \mathbf{x}'' &= (\mathsf{T}^{-1}\hat{\mathbf{x}}')^{\mathsf{T}} \mathsf{F} (\tilde{\mathsf{T}}^{-1}\hat{\mathbf{x}}'') \\ &= {\hat{\mathbf{x}}'}^{\mathsf{T}} \mathsf{T}^{-\top} \mathsf{F} \tilde{\mathsf{T}}^{-1} \hat{\mathbf{x}}'' \\ &= {\hat{\mathbf{x}}'}^{\mathsf{T}} \hat{\mathsf{F}} \hat{\mathbf{x}}'' \end{aligned}$$

- Obtain fundamental matrix F through

$$\Rightarrow \quad \begin{aligned} \hat{\mathsf{F}} &= \mathsf{T}^{-\top} \mathsf{F} \tilde{\mathsf{T}}^{-1} \\ \mathsf{F} &= \mathsf{T}^{\mathsf{T}} \hat{\mathsf{F}} \tilde{\mathsf{T}} \end{aligned}$$

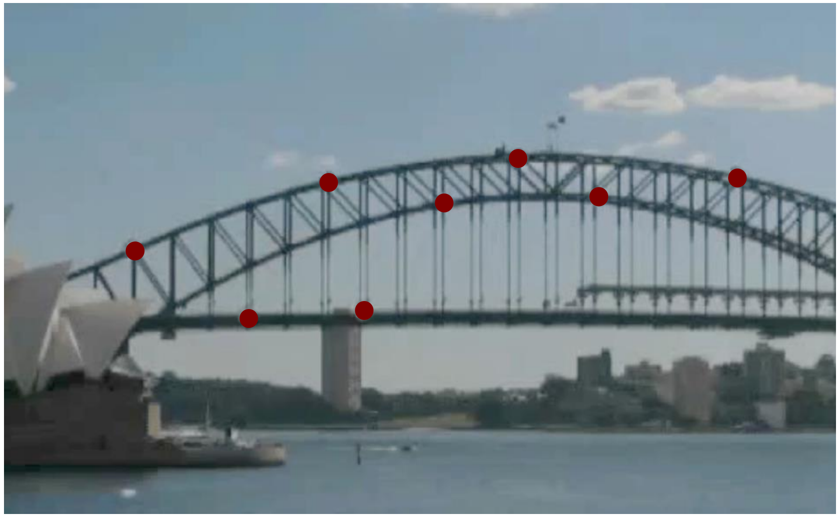
Singularity – No Translation

- The projection centers of both cameras are identical: $\boldsymbol{X}_{O'} = \boldsymbol{X}_{O''}$
- This happens if the translation of the camera is zero between both images



Singularity – Points on a Plane

- If all corresponding points lie on a plane, then we have some instabilities.
- Pg. 296, 11.9.2 from H&Z's Multiple View Geometry in Computer Vision



Images from the "Fundamental Matrix Song" Video by D. Wedge

2
Computing the Fundamental Matrix Given 7 Corresponding Points

Direct Solution with 7 Points

- We know that the fundamental matrix has **seven degrees of freedom**
- There exists a direct solution for 7 pts

The solution itself is more complex, so just the idea should matter here

Direct Solution with 7 Points

- We know that the fundamental matrix has **seven degrees of freedom**
- There exists a direct solution for 7 pts
- Idea: 2-dimensional null space of A
- Matrix F must fulfill $\mathbf{f} = \lambda \mathbf{f}_1 + (1-\lambda)\mathbf{f}_2$

vectors spanning the null space

Direct Solution with 7 Points

- We know that the fundamental matrix has **seven degrees of freedom**
- There exists a direct solution for 7 pts
- Idea: 2-dimensional null space of A
- Matrix F must fulfill $\mathbf{f} = \lambda \mathbf{f}_1 + (1-\lambda)\mathbf{f}_2$
- We also know that the determinant of the 3x3 matrix must be zero: $|\mathsf{F}| = 0$
- Can be combined to an equation of degree 3 $\Rightarrow$ up to three solutions

Summary so far

- Estimating the fundamental matrix from N pairs of corresponding points
- Direct solution of $N>7$ points based on solving a homogenous linear system ("8 point algorithm")
- Idea for a direct solution with 7 points (up to 3 solutions)

3
Let’s Do the Same for the Essential Matrix

Reminder: Essential Matrix

- **Fundamental matrix for calibrated cameras**

$$\boxed{\mathsf{E} = R' S_b R''^{\mathsf{T}}}$$

- Often parameterized through
  (general parameterization of dependent images)

$$\boxed{\mathsf{E} = S_b R^{\mathsf{T}}}$$

- Coplanarity constraint for calibrated cameras

$$\boxed{{}^k\mathbf{x}'^{\mathsf{T}} \mathsf{E}\, {}^k\mathbf{x}'' = 0}$$

Essential Matrix from 8+ Corresponding Points

- For each point, we have the coplanarity constraint

$$ {}^{k}\mathbf{x}'^{\mathsf{T}}_{n}\, \mathsf{E}\; {}^{k}\mathbf{x}''_{n} = 0 \qquad n = 1, ..., N $$

Essential Matrix from 8+ Corresponding Points

- For each point, we have the coplanarity constraint

$$ {}^{k}\mathbf{x}'^{\mathsf{T}}_{n}\ \mathsf{E}\ {}^{k}\mathbf{x}''_{n} = 0 \qquad n = 1, \ldots, N $$

- or

$$ [\,{}^{k}x'_n,\ {}^{k}y'_n, c'] \begin{bmatrix} E_{11} & E_{12} & E_{13} \\ E_{21} & E_{22} & E_{23} \\ E_{31} & E_{32} & E_{33} \end{bmatrix} \begin{bmatrix} {}^{k}x''_n \\ {}^{k}y''_n \\ c'' \end{bmatrix} = 0 $$

As for the Fundamental Matrix...

$$[\,{}^{k}x'_n,\ {}^{k}y'_n, c'\,] \begin{bmatrix} E_{11} & E_{12} & E_{13} \\ E_{21} & E_{22} & E_{23} \\ E_{31} & E_{32} & E_{33} \end{bmatrix} \begin{bmatrix} {}^{k}x''_n \\ {}^{k}y''_n \\ c'' \end{bmatrix} = 0$$

```
function E = E_from_point_pairs(xs, xss)
% xs, xss: Nx3 homologous point coordinates, N > 7
% E:       3x3 essential matrix

% coefficient matrix
for n = 1 : size(xs, 1)
    A(n, :) = kron(xss(n, :), xs(n, :));
end

% singular value decomposition
[U, D, V] = svd(A);

% select the singlar vector with the minimal si
E = reshape(V(:, 9), 3, 3)';
```

build matrix A

solve Ae=0

build matrix E

Which constraints to consider?

Constraints

- For the fundamental matrix, we enforced the $\mathrm{rank}(\mathsf{F}) = 2$ constraint

$$\mathsf{F} = UDV^\mathsf{T} = U \begin{bmatrix} D_{11} & 0 & 0 \\ 0 & D_{22} & 0 \\ 0 & 0 & 0 \end{bmatrix} V^\mathsf{T}$$

- For the essential matrix, both non-zero singular values are identical

$$\boxed{\mathsf{E} = U \begin{bmatrix} d & 0 & 0 \\ 0 & d & 0 \\ 0 & 0 & 0 \end{bmatrix} V^\mathsf{T} \underset{\text{homogenous}}{=} U \begin{bmatrix} 1 & 0 & 0 \\ 0 & 1 & 0 \\ 0 & 0 & 0 \end{bmatrix} V^\mathsf{T}}$$

8-Point Algorithm for the Essential Matrix

```
function E = E_from_point_pairs(xs, xss)
% xs, xss: Nx3 homologous point coordinates, N > 7
% E:       3x3 essential matrix

% coefficient matrix
for n = 1 : size(xs, 1)
    A(n, :) = kron(xss(n, :), xs(n, :));
end

% singular value decomposition
[U, D, V] = svd(A);

% approximate E, possibly regular
Ea = reshape(V(:, 9), 3, 3)';

% svd decomposition of E
[Ua, Da, Va] = svd(Ea);

% algebraically best E, singular, same singular values
E = Ua * diag([1, 1, 0]) * Va';
```

build matrix A

solve Ae=0

build matrix Ea

compute SVD of Ea

build matrix E from Ea by imposing constraints

Conditioning/Normalization to Obtain a Well-Conditioned Problem (As Done Before)

- As for the 8-Point algorithm for the fundamental matrix, normalization of the point coordinates is **essential**
- **Transform** the points so that the center of mass of all points is at (0,0)
- **Scale** the image so that the x and y coordinated are within [-1,1]

Conditioning/Normalization

- Define $\mathsf{T} : \mathsf{T}\mathbf{x} = \hat{\mathbf{x}}$ so that coordinates are zero-centered and in [-1,1]
- Determine essential matrix $\hat{\mathsf{E}}$ from the transformed coordinates

$$
\begin{aligned}
\mathbf{x}'^{\mathsf{T}}\mathsf{E}\mathbf{x}'' &= (\mathsf{T}^{-1}\hat{\mathbf{x}}')^{\mathsf{T}}\mathsf{E}(\tilde{\mathsf{T}}^{-1}\hat{\mathbf{x}}'') \\
&= \hat{\mathbf{x}}'^{\mathsf{T}}\mathsf{T}^{-\top}\mathsf{E}\tilde{\mathsf{T}}^{-1}\hat{\mathbf{x}}'' \\
&= \hat{\mathbf{x}}'^{\mathsf{T}}\hat{\mathsf{E}}\hat{\mathbf{x}}''
\end{aligned}
$$

- Obtain essential matrix E through

$$
\Rightarrow \quad
\begin{aligned}
\hat{\mathsf{E}} &= \mathsf{T}^{-\top}\mathsf{E}\tilde{\mathsf{T}}^{-1} \\
\mathsf{E} &= \mathsf{T}^{\mathsf{T}}\hat{\mathsf{E}}\tilde{\mathsf{T}}
\end{aligned}
$$

Properties of the Essential Mat.

- Homogenous
- Singular: $|\mathsf{E}| = 0$ (determinant is zero)
- Two identical non-zero singular values

$$\mathsf{E} = U \begin{bmatrix} 1 & 0 & 0 \\ 0 & 1 & 0 \\ 0 & 0 & 0 \end{bmatrix} V^{\mathsf{T}}$$

- As a result of the skew-sym. matrix (not currently exploiting):

$$2\mathsf{E}\mathsf{E}^{\mathsf{T}}\mathsf{E} - \operatorname{tr}(\mathsf{E}\mathsf{E}^{\mathsf{T}})\mathsf{E} = \underset{3\times 3}{\mathbf{0}}$$

4
Special Cases for Computing the Essential Matrix

Essential Matrix in Case of Known Rotations

- In case the rotations are known, the coplanarity constraints simplifies

$$^{k}\mathbf{x}'^{\mathsf{T}}\ \mathsf{E}\ \ ^{k}\mathbf{x}'' = 0$$

$$\Downarrow$$

point in the rotated image

$$^{k}\mathbf{x}'^{\mathsf{T}} \mathsf{S}_B R^{\mathsf{T}}\ ^{k}\mathbf{x}'' = \ ^{k}\mathbf{x}'^{\mathsf{T}} \mathsf{S}_B\ ^{1}\mathbf{x}'' = 0$$

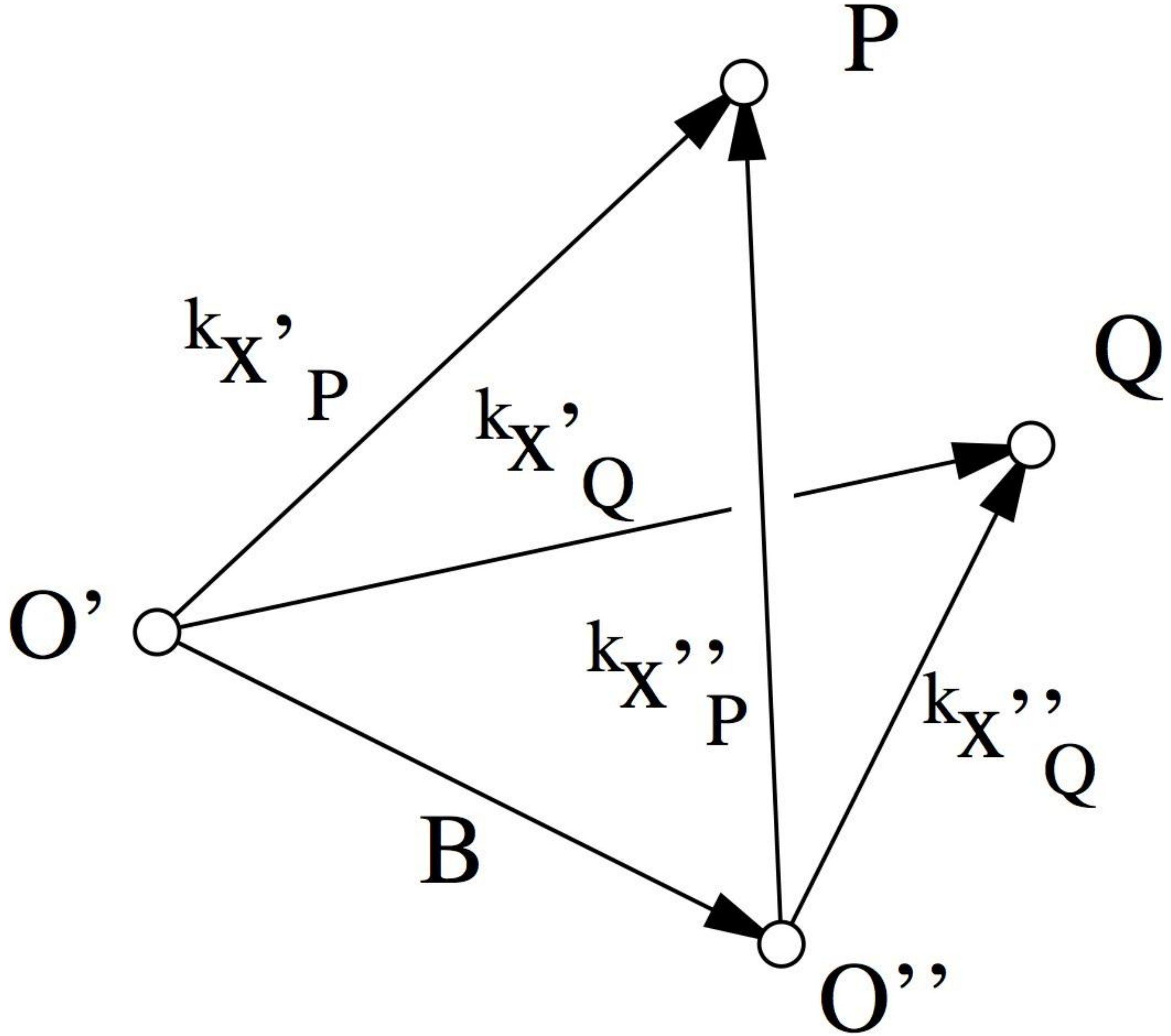
- As the rotations are known, we only have 2 DoF. Why?

Essential Matrix in Case of Known Rotations

- We only have 2 degrees of freedom
- Two corresponding points are sufficient to compute the basis

$$({}^{k}\mathbf{x}'_P,\ {}^{k}\mathbf{x}''_P),\ ({}^{k}\mathbf{x}'_Q,\ {}^{k}\mathbf{x}''_Q)$$

Computing B from P and Q

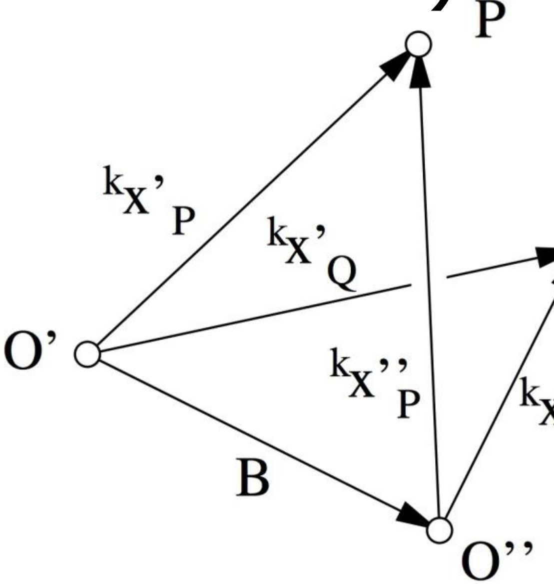


Essential Matrix in Case of Known Rotations

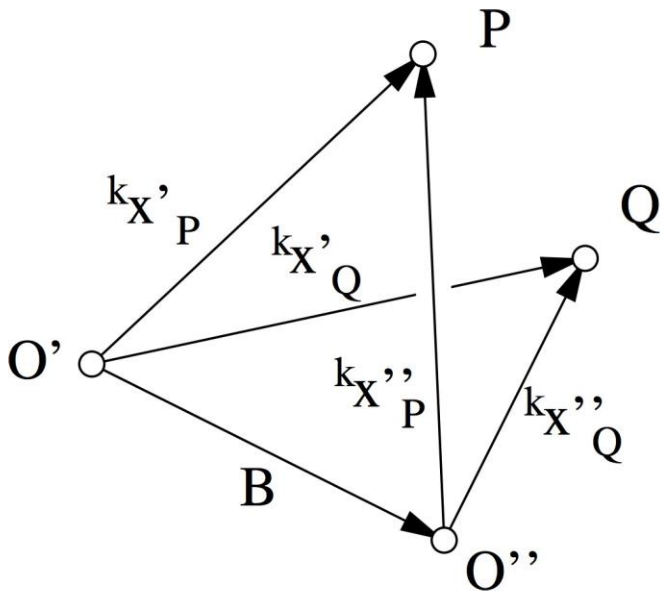
- Two corresponding points
  $({}^k\mathbf{x}'_P,\ {}^k\mathbf{x}''_P), ({}^k\mathbf{x}'_Q,\ {}^k\mathbf{x}''_Q)$
- Normal vectors of the epipolar planes (given known camera rotations)

$$\mathbf{n}_P = {}^k\mathbf{x}'_P \times {}^k\mathbf{x}''_P$$

$$\mathbf{n}_Q = {}^k\mathbf{x}'_Q \times {}^k\mathbf{x}''_Q$$



Essential Matrix in Case of Known Rotations



- Normals

$$\mathbf{n}_P = {}^k\mathbf{x}'_P \times {}^k\mathbf{x}''_P$$

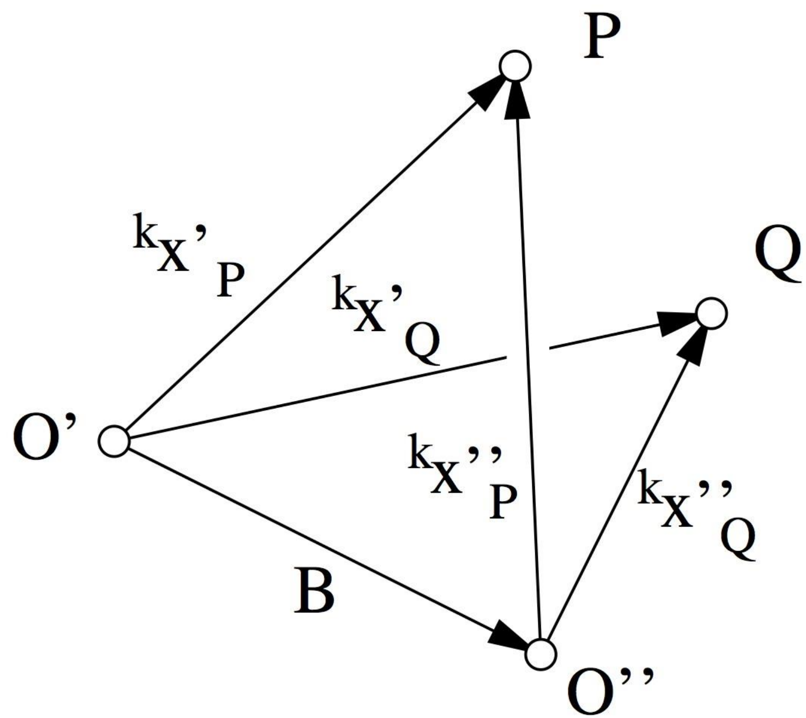
$$\mathbf{n}_Q = {}^k\mathbf{x}'_Q \times {}^k\mathbf{x}''_Q$$

- The epipolar axis and thus B must be **orthogonal to the both normals**

$$\begin{aligned} \widehat{B} &= \mathbf{n}_P \times \mathbf{n}_Q \\ &= ({}^k\mathbf{x}'_P \times {}^k\mathbf{x}''_P) \times ({}^k\mathbf{x}'_Q \times {}^k\mathbf{x}''_Q) \end{aligned}$$

Which Assumption Did We Make?

- |B| > 0
- Points in different epipolar planes
- No points at infinity
- No point on B



$$\begin{aligned}\widehat{B} &= \mathbf{n}_P \times \mathbf{n}_Q \\ &= ({}^k\mathbf{x}'_P \times {}^k\mathbf{x}''_P) \times ({}^k\mathbf{x}'_Q \times {}^k\mathbf{x}''_Q)\end{aligned}$$

5
5-Point Algorithm

5-Point Algorithm

- Proposed by Nistér in 2003/2004
- Standard solution today to obtaining a direct solution
- Solving a polynomial of degree 10
- 10 possible solutions
- Often used together RANSAC
  - RANSAC proposes correspondences
  - Evaluate all 5-point solutions based on the other corresponding points

5-Point Algorithm

- More details in the script by Förstner "Photogrammetrie II", Ch 1.2
- Stewenius, Engels, Nistér: "Recent Developments on Direct Relative Orientation", ISPRS 2006
- Li and Hartley: "Five-Point Motion Estimation Made Easy"

Summary

- Compute S_B, R given E
- Direct solutions
  - F from N>7 points
  - F from N=7 points (idea)
  - E from N>7 points
  - E from N=2/4 points under special cond.
  - E from N=5 points (idea)
- Cannot exploit overdetermined cases
- Initial guess for iterative solutions

Literature

- Förstner, Wrobel: Photogrammetric Computer Vision, Ch. 12.3.1-12.3.3
- Hartley: In Defence of the 8-point Algorithm
- Stewenius, Engels, Nistér: Recent Developments on Direct Relative Orientation, ISPRS 2006

Slide Information

- The slides have been created by Cyrill Stachniss as part of the photogrammetry and robotics courses.
- I tried to acknowledge all people from whom I used images or videos. In case I made a mistake or missed someone, please let me know.
- The photogrammetry material heavily relies on the very well written lecture notes by Wolfgang Förstner and the Photogrammetric Computer Vision book by Förstner & Wrobel.
- Parts of the robotics material stems from the great Probabilistic Robotics book by Thrun, Burgard and Fox.

Arjun Jain, ajain@cse.iitb.ac.in