CSE 252D: Advanced Computer Vision

Manmohan Chandraker

Lecture 7: Advanced Structure and Motion

Virtual classrooms

- Virtual lectures on Zoom
  - Only host shares the screen
  - Keep video off and microphone muted
  - But please do speak up (remember to unmute!)
  - Slides uploaded on webpage just before class
- Virtual interactions on Zoom
  - Ask and answer plenty of questions
  - “Raise hand” feature on Zoom when you wish to speak
  - Post questions on chat window
  - Happy to try other suggestions!
- Lectures recorded and upload on Kaltura
  - Available under “My Media” on Canvas

Overall goals for the course

- Introduce the state-of-the-art in computer vision
- Study principles that make them possible
- Get understanding of tools that drive computer vision
- Enable one or all of several such outcomes
  - Pursue higher studies in computer vision
  - Join industry to do cutting-edge work in computer vision
  - Gain appreciation of modern computer vision technologies
- This is a great time to study computer vision!

Lighting Presentations

- Look out for email with steps for completing the presentation
  - Confirm the paper assignment when you receive the email
  - Send presentation and script to instructor and TA 3 days before class
  - Send final version and recorded video 1 day before class
- Time limit: 5 minutes ☺
  - **High-quality presentation:** well-practiced and fluent
  - Stay close to provided template
  - **Illustrate:** 1 image per slide, a few bullet points to explain it
  - **Big no-no's:** giant tables of numbers, hyperparameters
- Order of presentation: alphabetic
  - Papers assigned by instructor
  - https://docs.google.com/spreadsheets/d/1JcM4V2GaPf7WF2YLFQ70Rn6BaLYEOqEzzj8rXOsb5bw/edit?usp=sharing

Papers for Wed, Apr 21

- GeoNet: Unsupervised Learning of Dense Depth, Optical Flow and Camera Pose
  - https://arxiv.org/abs/1803.02276

- Unsupervised Scale-Consistent Depth and Ego-Motion Learning from Monocular Video
  - https://arxiv.org/abs/1908.10553

Papers for Fri, Apr 23

- Building Rome in a Day
  - https://grail.cs.washington.edu/rome/rome_paper.pdf
- CodeSLAM - Learning a Compact, Optimisable Representation for Dense Visual SLAM
  - https://arxiv.org/abs/1804.00874
- Beyond Tracking: Selecting Memory and Refining Poses for Deep Visual Odometry
  - https://arxiv.org/abs/1904.01892
- BA-Net: Dense Bundle Adjustment Network
  - https://arxiv.org/abs/1806.04807

Papers for Wed, Apr 28

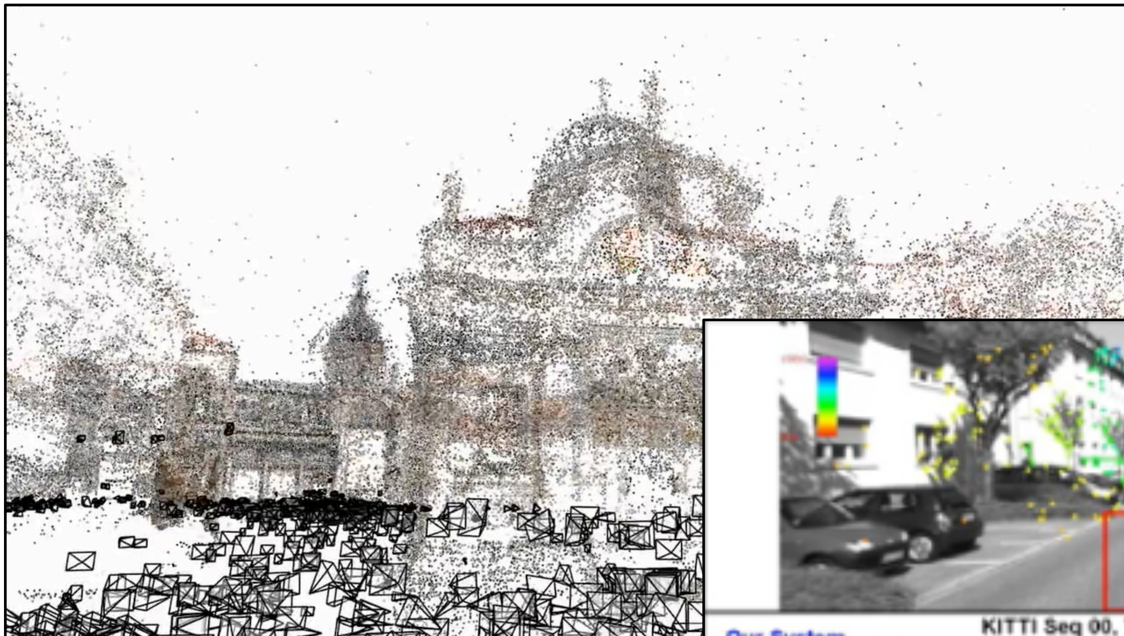
- Deep Fundamental Matrix Estimation
  - https://openaccess.thecvf.com/content_ECCV_2018/html/Rene_Ranftl_Deep_Fundamental_Matrix_ECCV_2018_paper.html
- DSAC - Differentiable RANSAC for Camera Localization
  - https://arxiv.org/abs/1611.05705
- Unsupervised Monocular Depth Estimation with Left-Right Consistency
  - https://arxiv.org/abs/1609.03677
- LSD-SLAM: Large-Scale Direct Monocular SLAM
  - https://vision.in.tum.de/_media/spezial/bib/engel14eccv.pdf

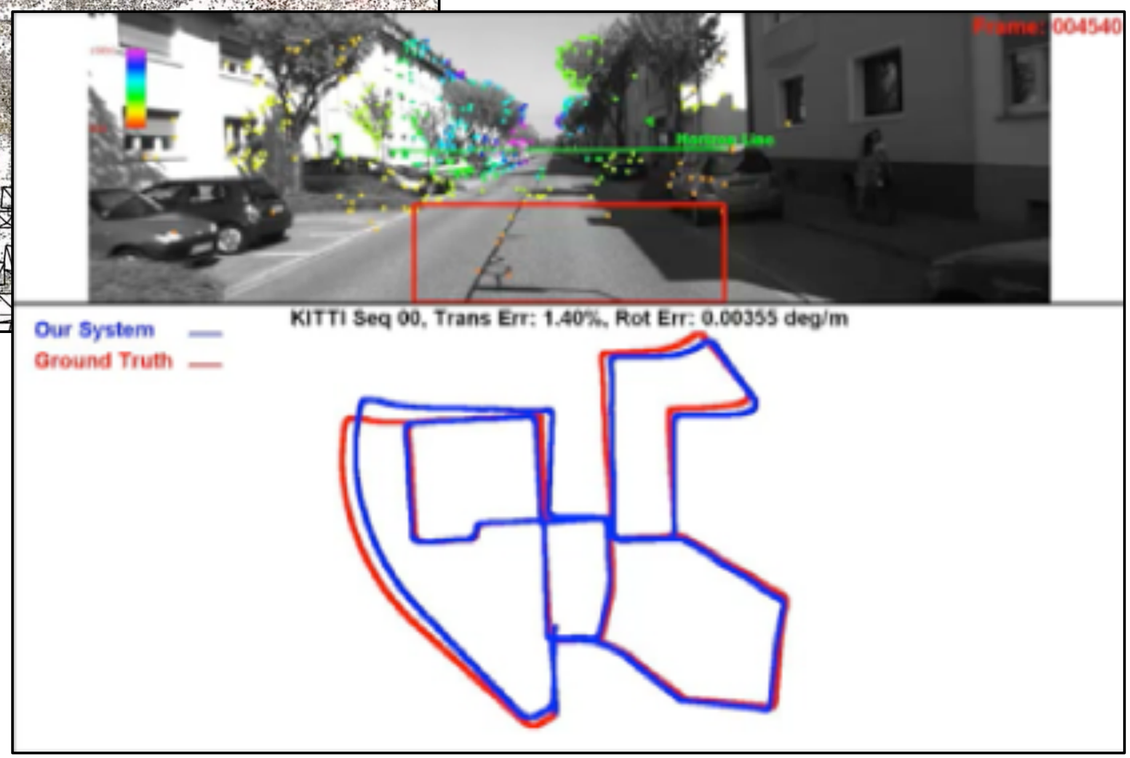
Papers for Fri, Apr 30

- A Discriminative Feature Learning Approach for Deep Face Recognition
  – https://ydwen.github.io/papers/WenECCV16.pdf
- SphereFace: Deep Hypersphere Embedding for Face Recognition
  – https://arxiv.org/abs/1704.08063
- ArcFace: Additive Angular Margin Loss for Deep Face Recognition
  – https://arxiv.org/abs/1801.07698
- CosFace: Large Margin Cosine Loss for Deep Face Recognition
  – https://arxiv.org/abs/1801.09414

Recap

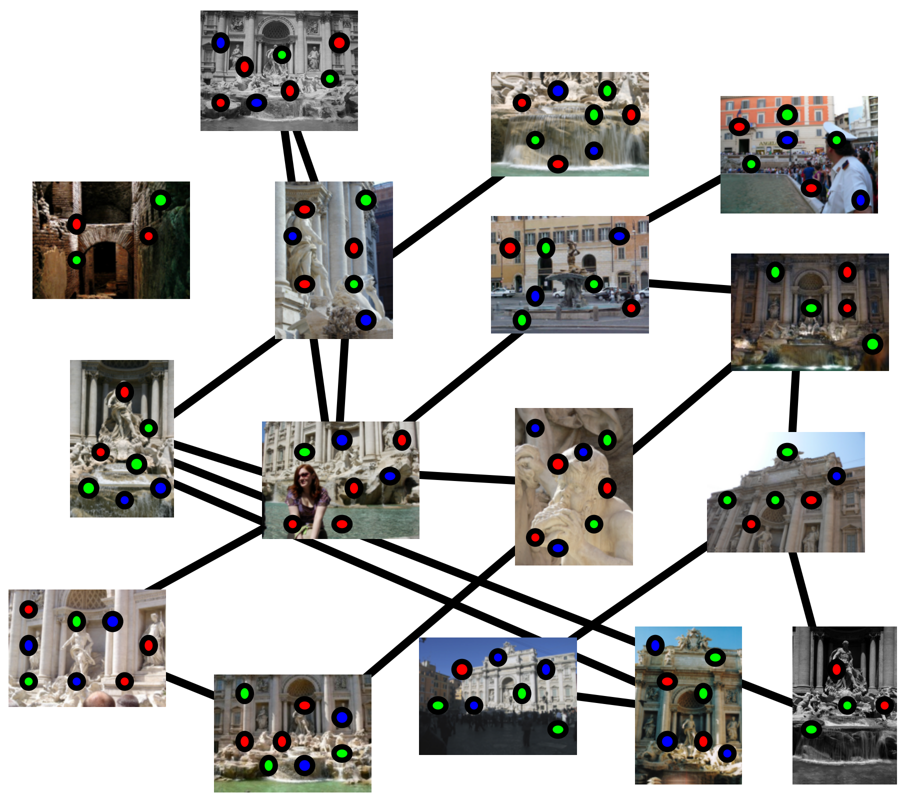
Unordered or Ordered Images



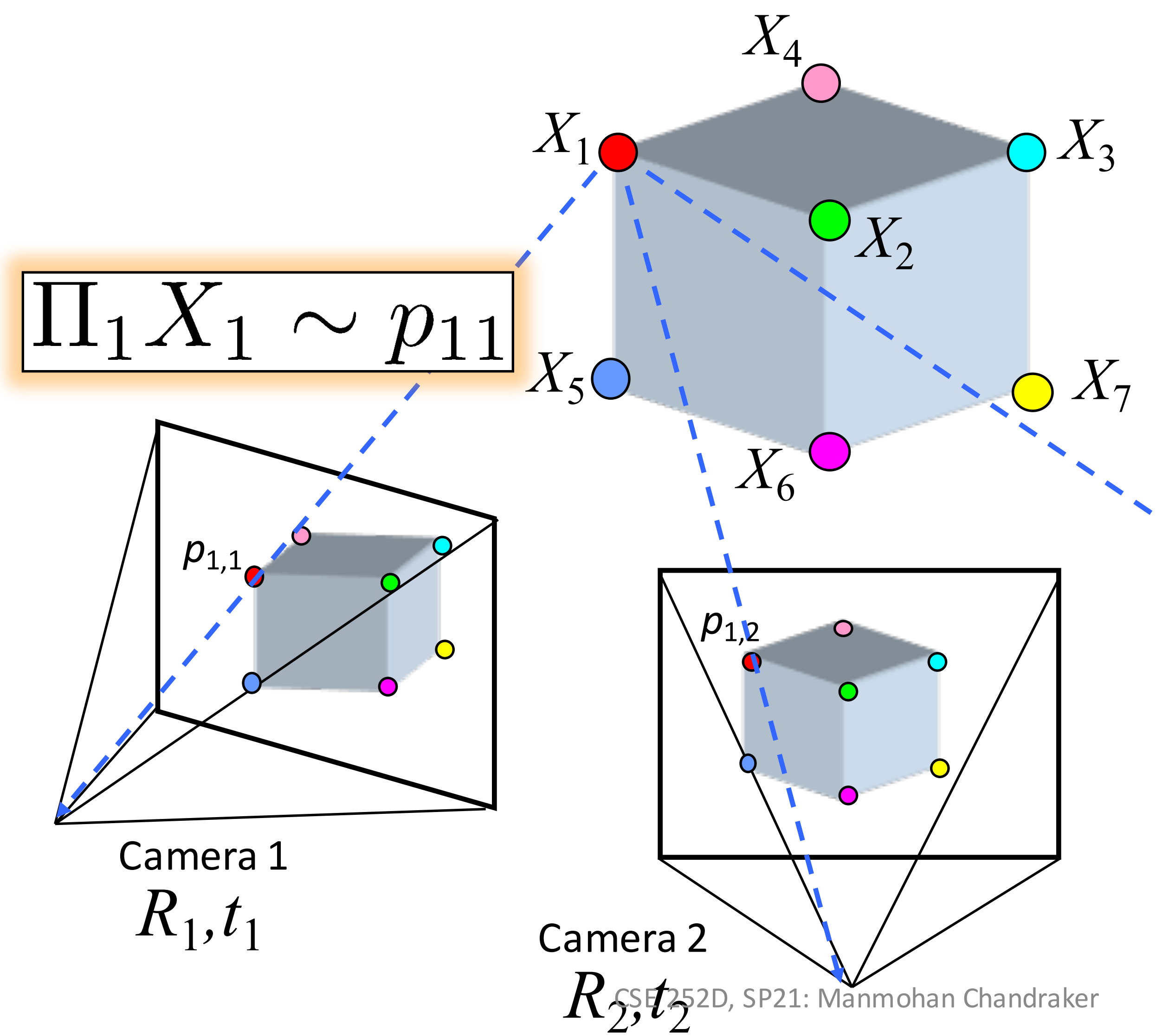


Feature matching

Match features between each pair of images

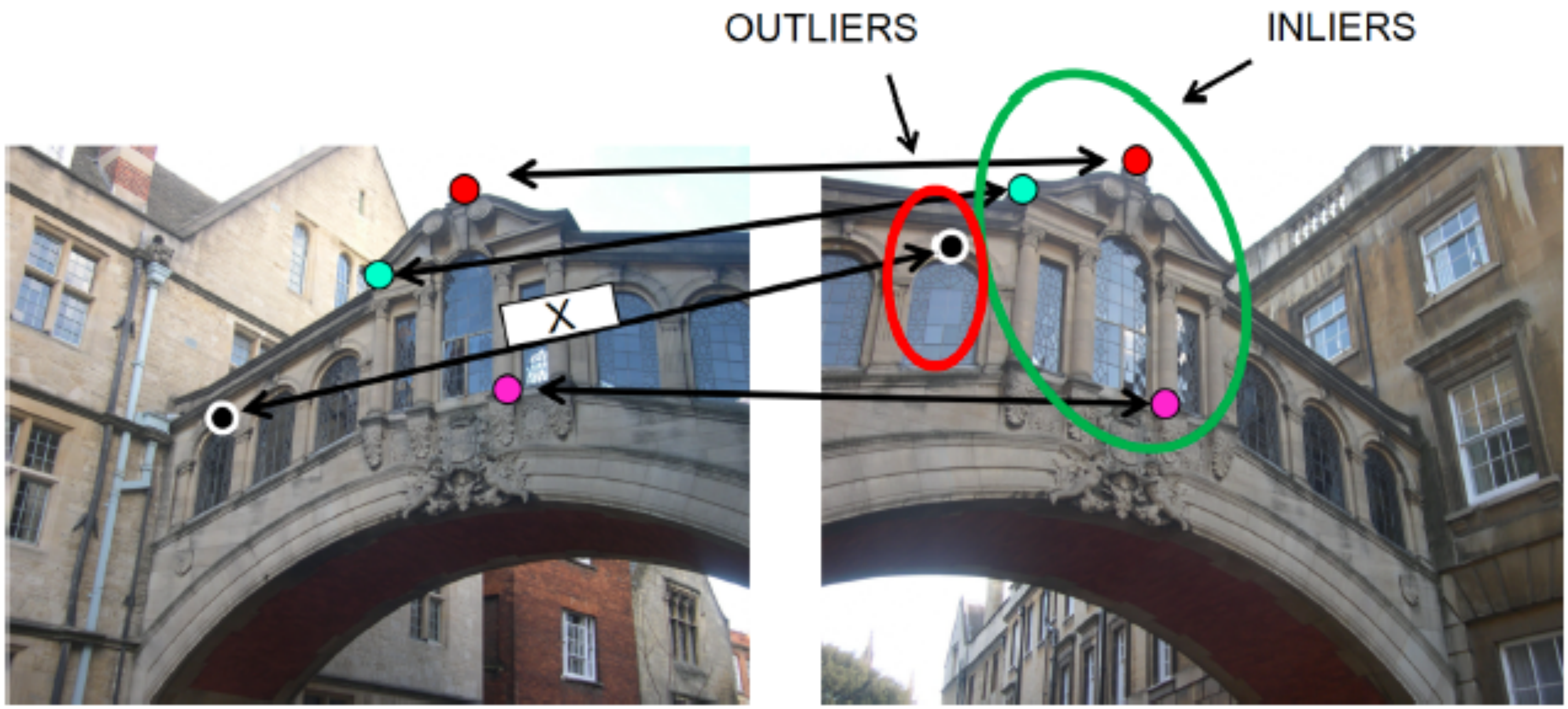


Structure from motion

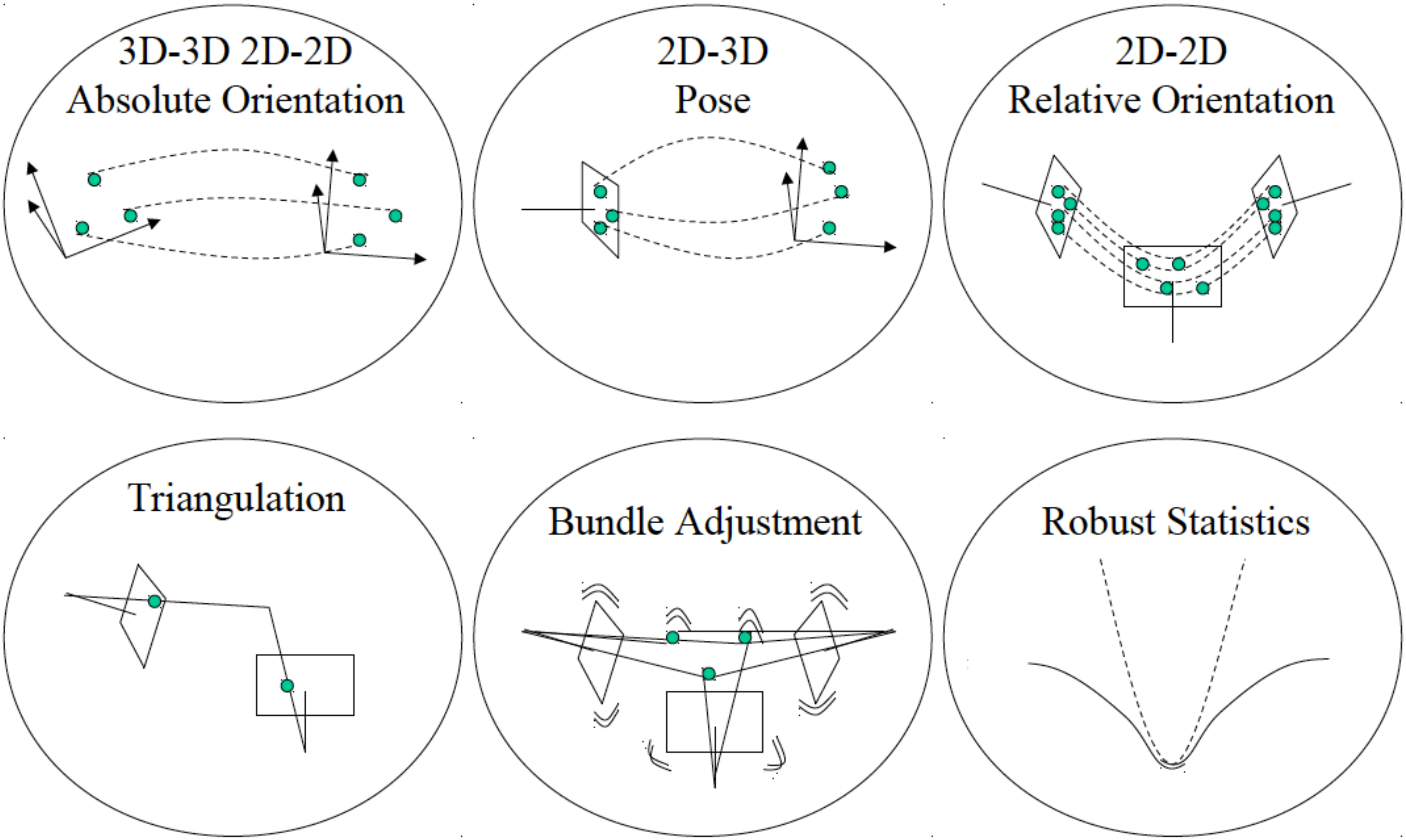


minimize
$$g(\mathbf{R}, \mathbf{T}, \mathbf{X})$$
non-linear least squares

Feature matching



Toolkit for Practical SFM



Homogeneous coordinates

- Converting to homogeneous coordinates

$$(x, y) \Rightarrow \begin{bmatrix} x \\ y \\ 1 \end{bmatrix} \qquad (x, y, z) \Rightarrow \begin{bmatrix} x \\ y \\ z \\ 1 \end{bmatrix}$$

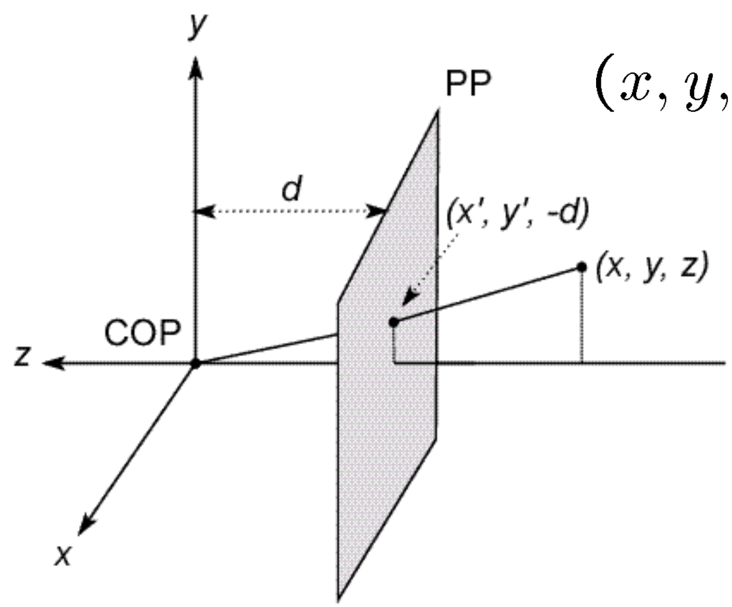
Homogeneous 2D point Homogeneous 3D point

- Converting from homogeneous coordinates

$$\begin{bmatrix} x \\ y \\ w \end{bmatrix} \Rightarrow (x/w, y/w) \qquad \begin{bmatrix} x \\ y \\ z \\ w \end{bmatrix} \Rightarrow (x/w, y/w, z/w)$$

- $(x, y, w)^\top$ and $(kx, ky, kw)^\top$ are the same point.

Modeling projection



$$(x, y, z) \rightarrow (-d\frac{x}{z},\ -d\frac{y}{z})$$

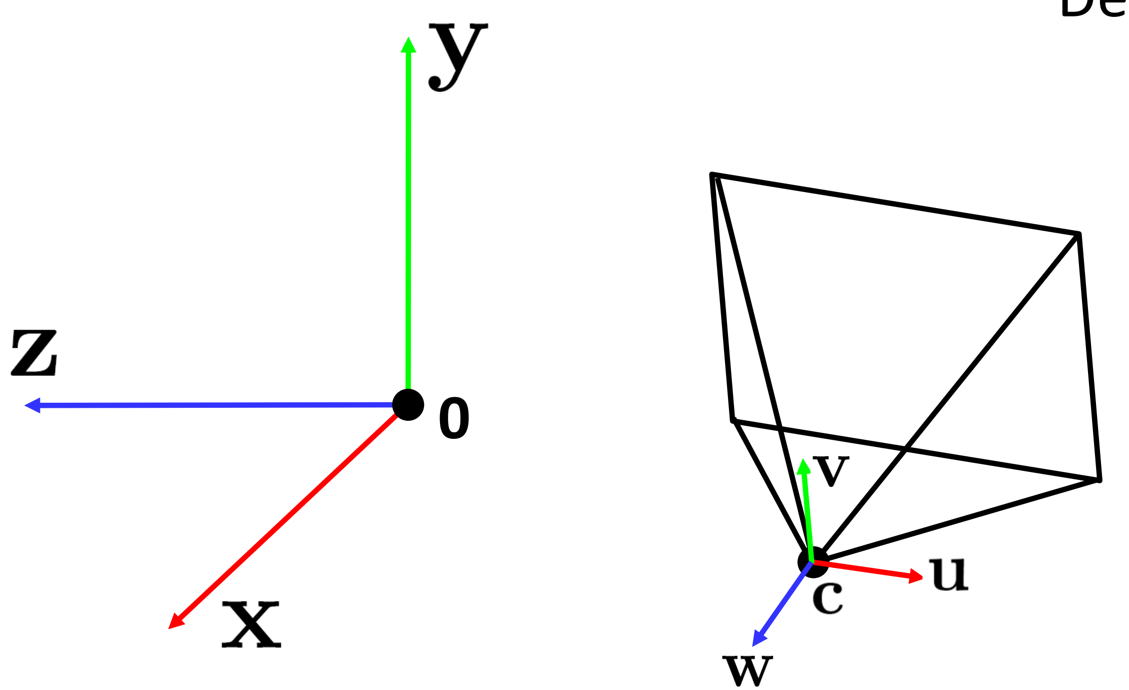
- A matrix multiplication using homogeneous coordinates

$$\begin{bmatrix} -d & 0 & 0 & 0 \\ 0 & -d & 0 & 0 \\ 0 & 0 & 1 & 0 \end{bmatrix} \begin{bmatrix} x \\ y \\ z \\ 1 \end{bmatrix} = \begin{bmatrix} -dx \\ -dy \\ z \end{bmatrix} \rightarrow (-d\frac{x}{z},\ -d\frac{y}{z})$$

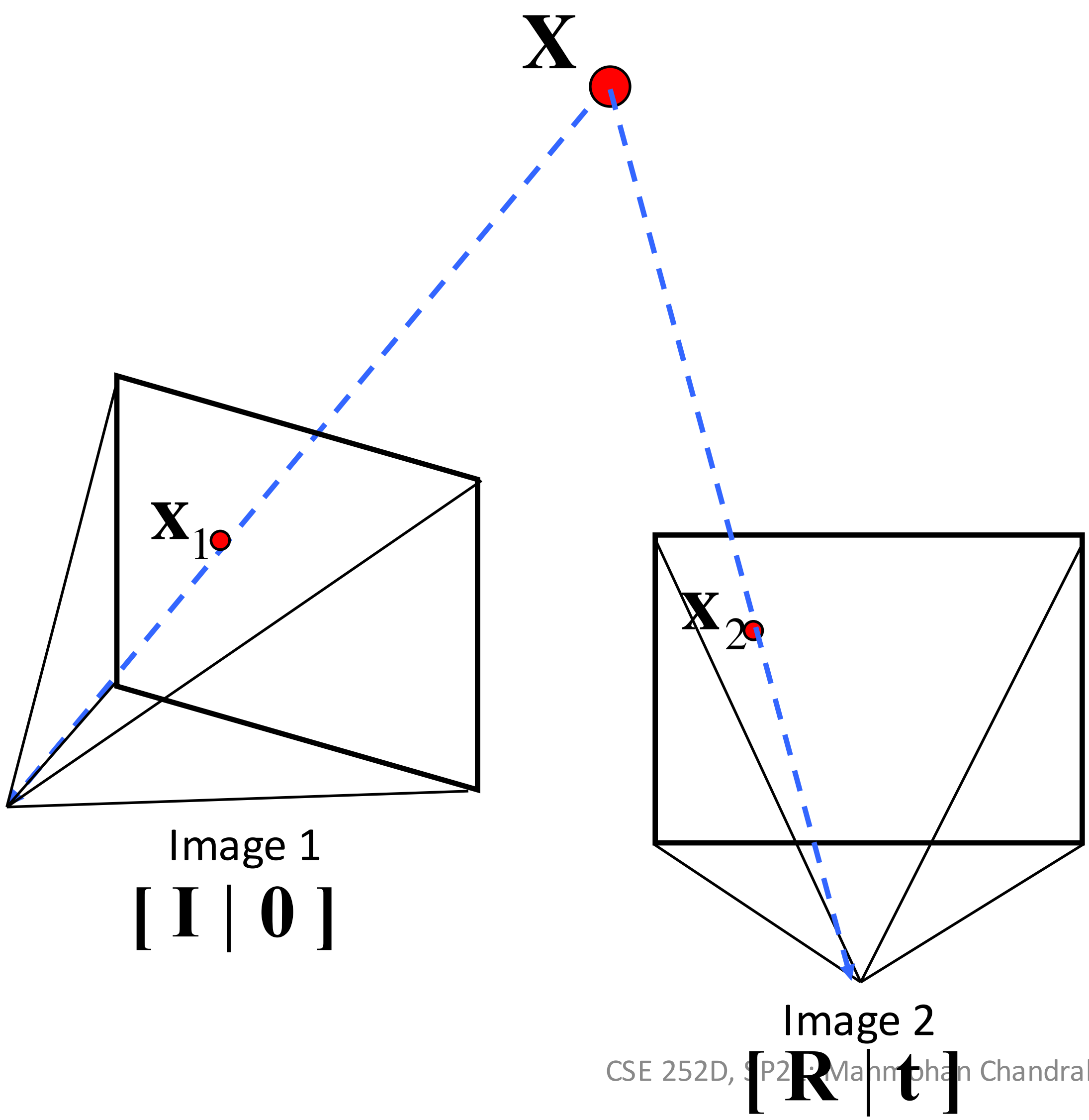
Projection matrix

$$\mathbf{\Pi} = \underbrace{\mathbf{K}}_{\text{intrinsics}} \underbrace{\begin{bmatrix} 1 & 0 & 0 & 0 \\ 0 & 1 & 0 & 0 \\ 0 & 0 & 1 & 0 \end{bmatrix}}_{\text{projection}} \underbrace{\begin{bmatrix} \mathbf{R} & \begin{matrix} 0 \\ 0 \\ 0 \end{matrix} \\ \begin{matrix} 0 & 0 & 0 \end{matrix} & 1 \end{bmatrix}}_{\text{rotation}} \underbrace{\begin{bmatrix} \mathbf{I}_{3\times 3} & -\mathbf{c} \\ \begin{matrix} 0 & 0 & 0 \end{matrix} & 1 \end{bmatrix}}_{\text{translation}} = \mathbf{K} \begin{bmatrix} \mathbf{R} & -\mathbf{Rc} \end{bmatrix}$$

Denote this by **t**



Fundamental Matrix



$$\mathbf{x}_1 \leftrightarrow \mathbf{x}_2$$

$$\mathbf{x}_1^T \mathbf{F} \mathbf{x}_2 = 0$$

$$\mathbf{F} = \mathbf{K}_2^{-\top} \mathbf{E} \mathbf{K}_1^{-1}$$

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

Direct Linear Transform Method

Given n point correspondences, set up a system of equations:

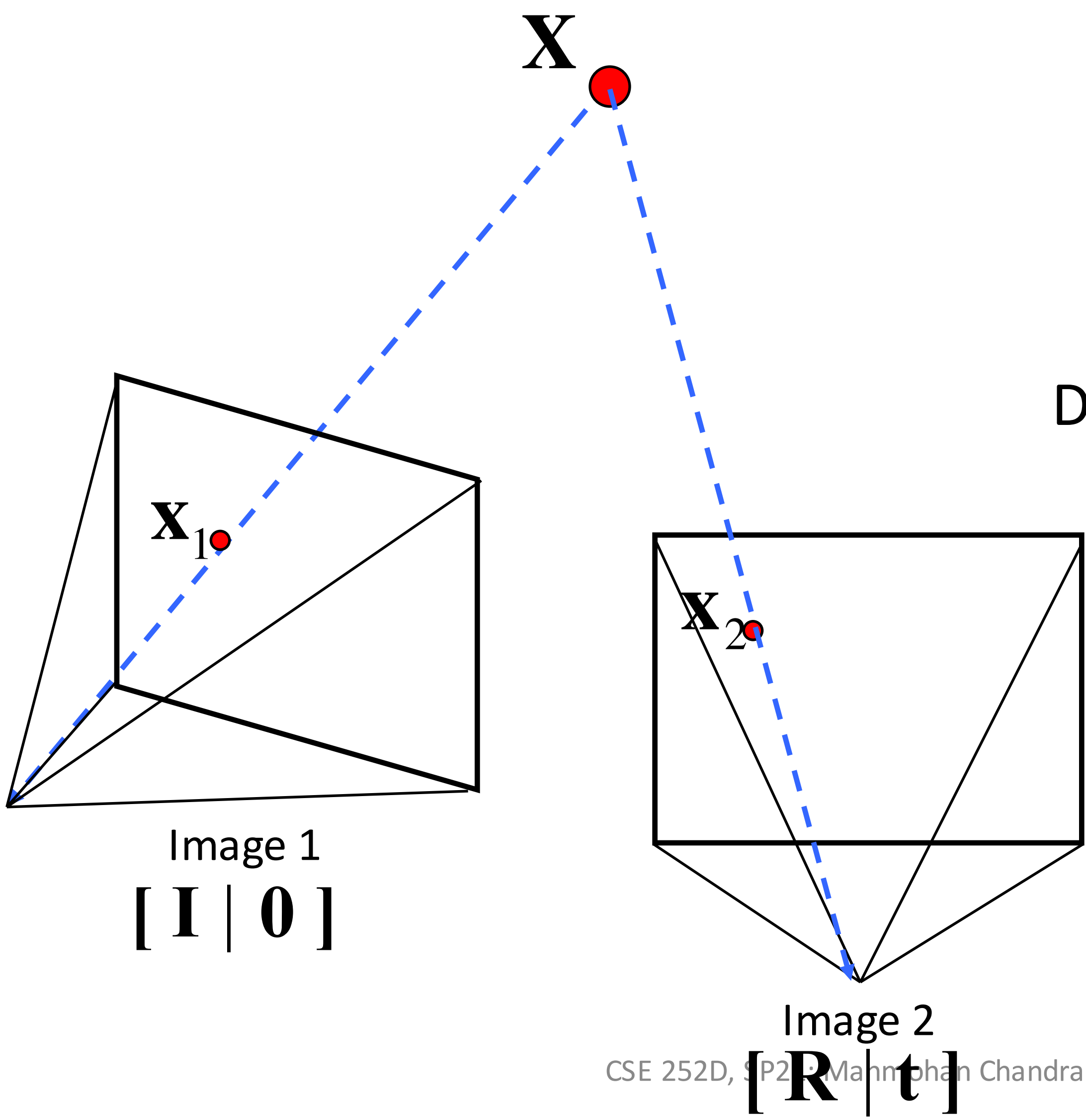
$$\begin{pmatrix} u_1 u_1' & v_1 u_1' & u_1' & u_1 v_1' & v_1 v_1' & v_1' & u_1 & v_1 & 1 \\ u_2 u_2' & v_2 u_2' & u_2' & u_2 v_2' & v_2 v_2' & v_2' & u_2 & v_2 & 1 \\ \vdots & \vdots & \vdots & \vdots & \vdots & \vdots & \vdots & \vdots & \vdots \\ u_n u_n' & v_n u_n' & u_n' & u_n v_n' & v_n v_n' & v_n' & u_n & v_n & 1 \end{pmatrix} \begin{pmatrix} f_{11} \\ f_{12} \\ f_{13} \\ f_{21} \\ f_{22} \\ f_{23} \\ f_{31} \\ f_{32} \\ f_{33} \end{pmatrix} = 0$$

- In reality, instead of solving $\mathbf{Af} = 0$, we seek $\mathbf{f}$ to minimize $\|\mathbf{Af}\|$, using SVD.

RANSAC

- General version:
  1. Randomly choose *s* samples
     - Typically *s* = minimum sample size to fit a model
  2. Fit a model (say, line) to those samples
  3. Count the number of inliers that approximately fit the model
  4. Repeat *N* times
  5. Choose the model with the largest set of inliers

Fundamental Matrix



$$\mathbf{x}_1 \leftrightarrow \mathbf{x}_2$$

$$\mathbf{x}_1^T \mathbf{F} \mathbf{x}_2 = 0$$

Degrees of freedom for F: 7

So, 7 points suffice to find F, but use 8 for linear method

RANSAC to Estimate Fundamental Matrix

- For *N* times
  - Pick 8 points
  - Compute a solution for $\mathbf{F}$ using these 8 points
  - Count number of inliers with $\mathbf{x}_1^T \mathbf{F} \mathbf{x}_2$ less than threshold
- Pick the one with the largest number of inliers

- Number of samples depends on
  - Outlier ratio
  - Probability of correct answer
  - Model size

RANSAC

- Adaptively determine number of iterations based on outlier proportion

$$N = \frac{\log(1-p)}{\log(1-(1-\epsilon)^s)}$$

Sample size	Proportion of outliers ϵ						
s	5%	10%	20%	25%	30%	40%	50%
2	2	3	5	6	7	11	17
3	3	4	7	9	11	19	35
4	3	5	9	13	17	34	72
5	4	6	12	17	26	57	146
6	4	7	16	24	37	97	293
7	4	8	20	33	54	163	588
8	5	9	26	44	78	272	1177

Values of N for $p = 0.99$

Motion from correspondences

- Use 8-point algorithm to estimate F
- Get E from F:

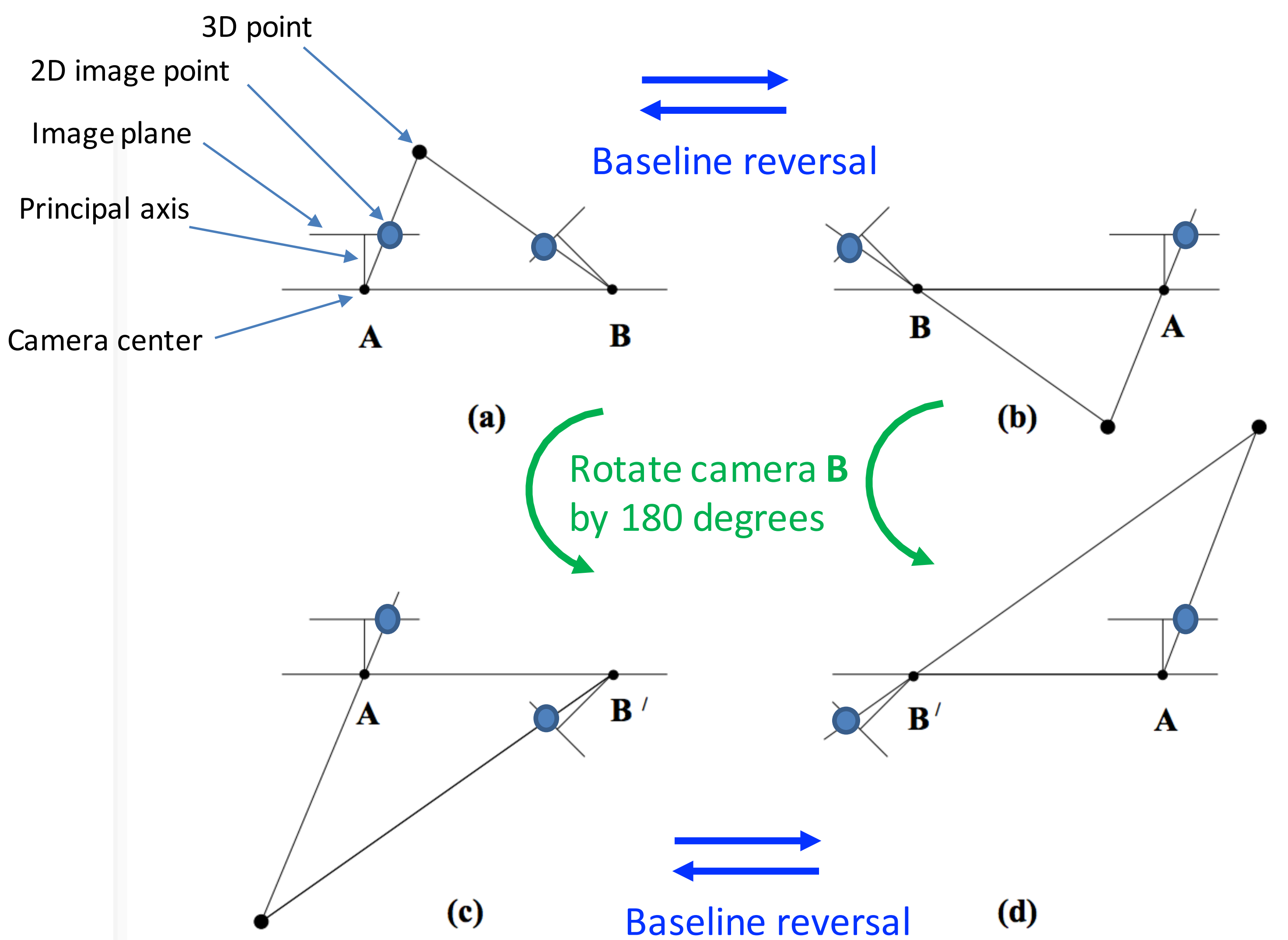
$$\mathbf{F} = \mathbf{K}_2^{-\top} \mathbf{E} \mathbf{K}_1^{-1}$$

$$\mathbf{E} = \mathbf{K}_2{}^{\top} \mathbf{F} \mathbf{K}_1$$

- Decompose E into skew-symmetric and rotation matrices:

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

Four Possible Solutions



Bundle adjustment

- Minimize sum of squared reprojection errors:

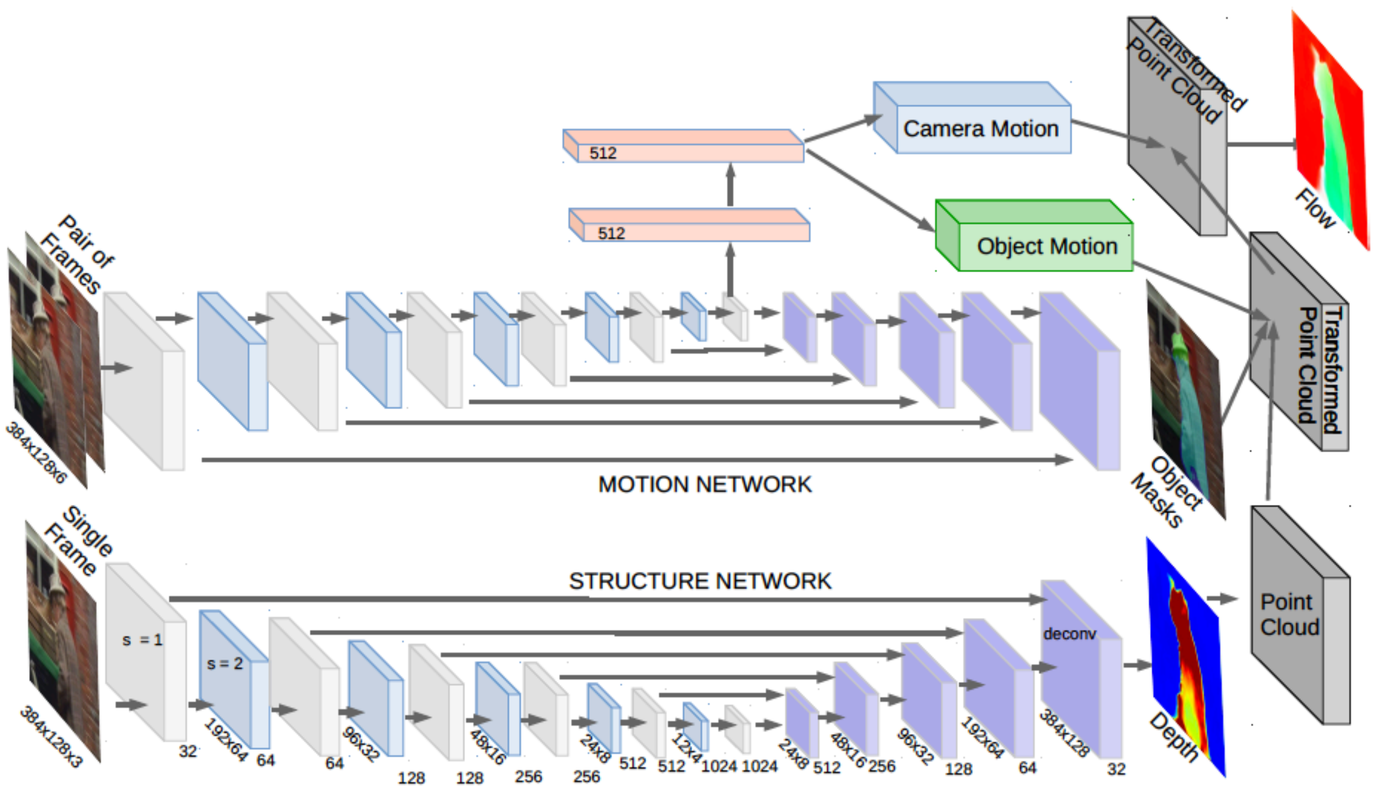
$$g(\mathbf{X}, \mathbf{R}, \mathbf{T}) = \sum_{i=1}^{m} \sum_{j=1}^{n} \underbrace{w_{ij}}_{\substack{\textit{indicator variable}:\\ \text{whether point } i \text{ visible in image } j}} \cdot \left\| \underbrace{\mathbf{P}(\mathbf{x}_i, \mathbf{R}_j, \mathbf{t}_j)}_{\substack{\textit{predicted}\\ \text{image location}}} - \underbrace{\begin{bmatrix} u_{i,j} \\ v_{i,j} \end{bmatrix}}_{\substack{\textit{observed}\\ \text{image location}}} \right\|^2$$

  - Optimized with non-linear least squares
  - Levenberg-Marquardt is a popular choice
- Practical challenges?
  - Initialization
  - Outliers

Learning Structure and Motion

Typical Way to Learn Structure and Motion

- Estimate depths (convert to 3D points given calibration) in frame t
- Estimate motion from frame t to t+1 for background and objects
- Project 3D points to frame t+1 using the estimated motions
- Use a consistency condition to declare matches as good



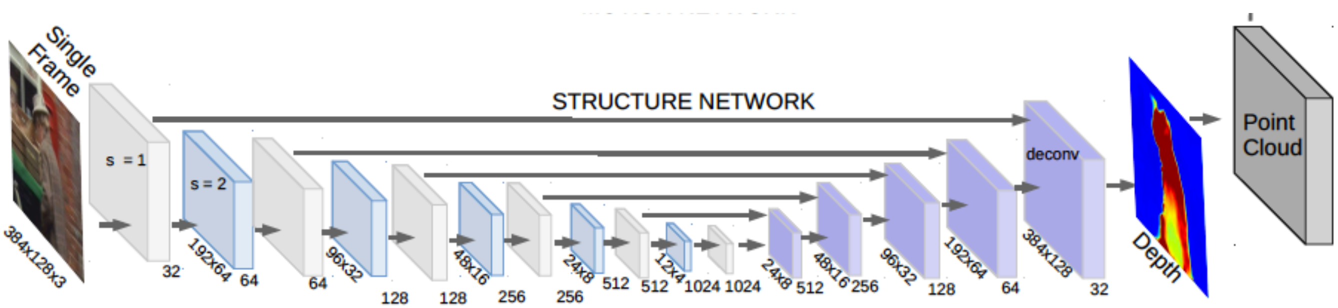
[SFM-Net, ICCV 2017]

Estimate Depth in an Image

- Depth in frame t : d_t , camera pose : $\{R_t^c, t_t^c\}$, K object motions : $\{R_t^k, t_t^k\}$

- Standard encoder-decoder for depth estimation
- For pixel (x_t^i, y_t^i) , with camera intrinsics (c_x, c_y, f) , 3D points are

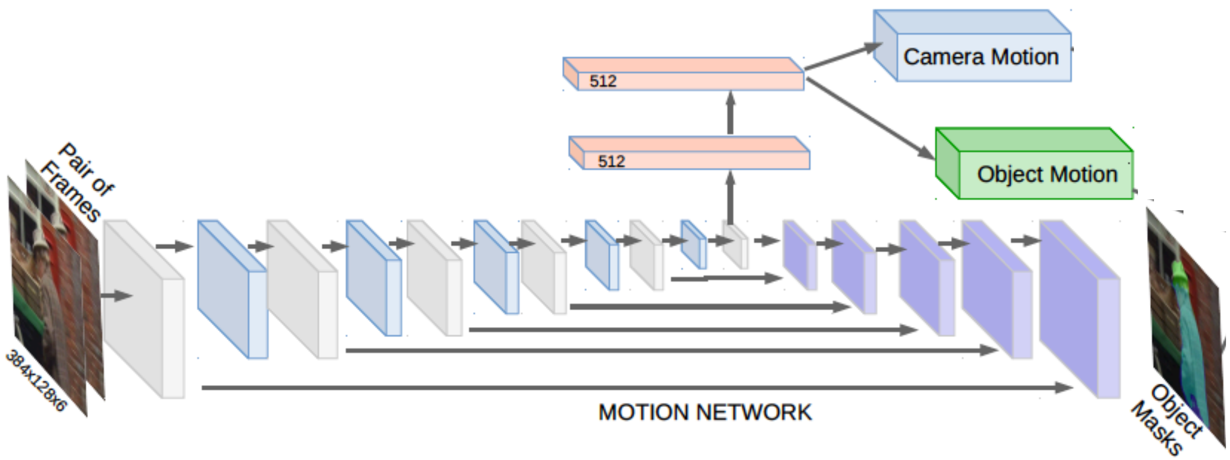
$$\mathbf{X}_t^i = \begin{bmatrix} X_t^i \\ Y_t^i \\ Z_t^i \end{bmatrix} = \frac{d_t^i}{f} \begin{bmatrix} \frac{x_t^i}{w} - c_x \\ \frac{y_t^i}{h} - c_y \\ f \end{bmatrix}$$



[SFM-Net, ICCV 2017]

Predict Motion across Frames

- Motion network takes frames t and t+1 as input
- Two fully-connected layers to predict camera and K object motions
- Transposed convolution on same embedding to get motion masks
- Same pixel can belong to multiple motions (articulations), $\mathrm{m}_t^k \in [0,1]^{(h \times w)}$



[SFM-Net, ICCV 2017]

Estimate Displacement in Image

- Apply object transformations on 3D points in frame t:

$$\mathbf{X}'_t = \mathbf{X}_t + \sum_{k=1}^{K} \mathrm{m}_t^k(i)(R_t^k(\mathbf{X}_t - p_k) + t_t^k - \mathbf{X}_t)$$

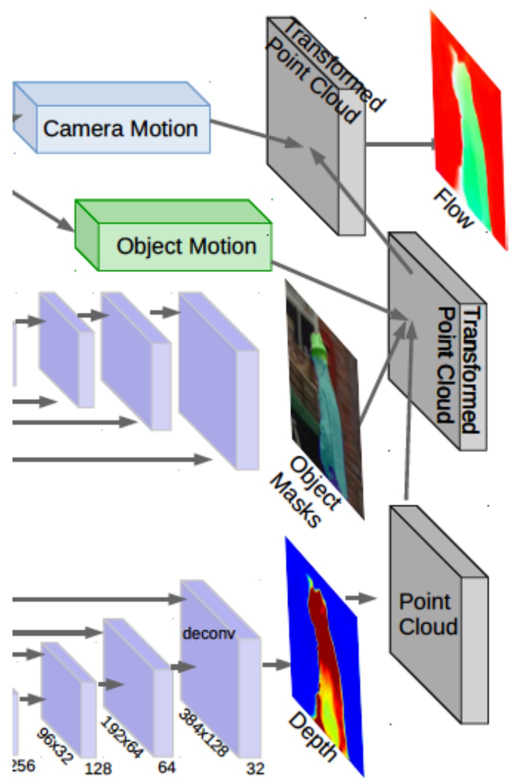
- Apply camera transformations on 3D points: $\mathbf{X}''_t = R_t^c(\mathbf{X}'_t - p_t^c) + t_t^c$

- Project to frame t+1:

$$\begin{bmatrix} \frac{x_{t+1}^i}{w} \\ \frac{y_{t+1}^i}{h} \end{bmatrix} = \frac{f}{Z''_t} \begin{bmatrix} X''_t \\ Y''_t \\ f \end{bmatrix} + \begin{bmatrix} c_x \\ c_y \end{bmatrix}$$

- Can compute flow vector:

$$(U_t(i), V_t(i)) = (x_{t+1}^i - x_t^i, y_{t+1}^i - y_t^i)$$

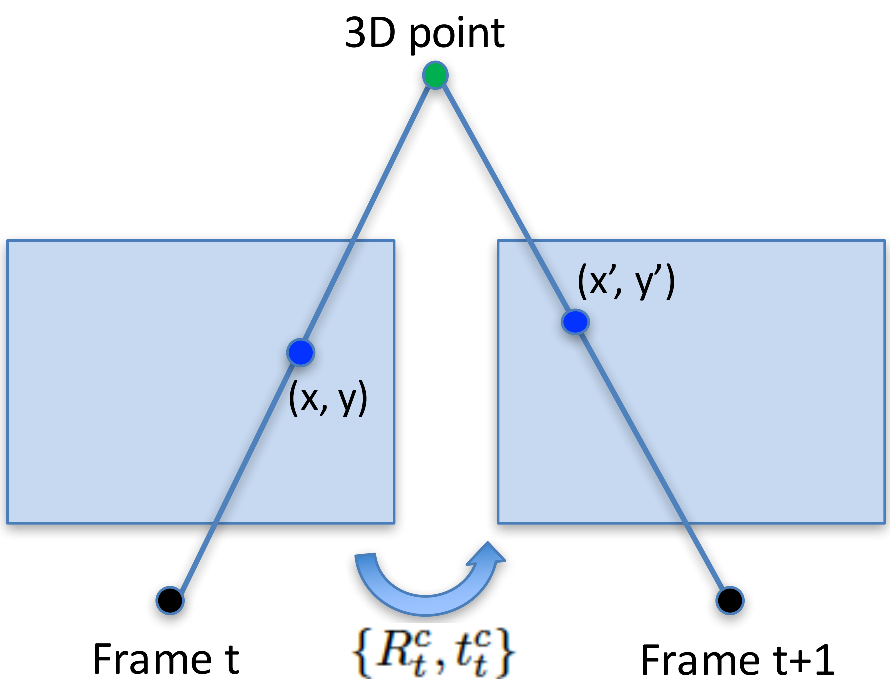


[SFM-Net, ICCV 2017]

Types of Supervision: Photoconsistency

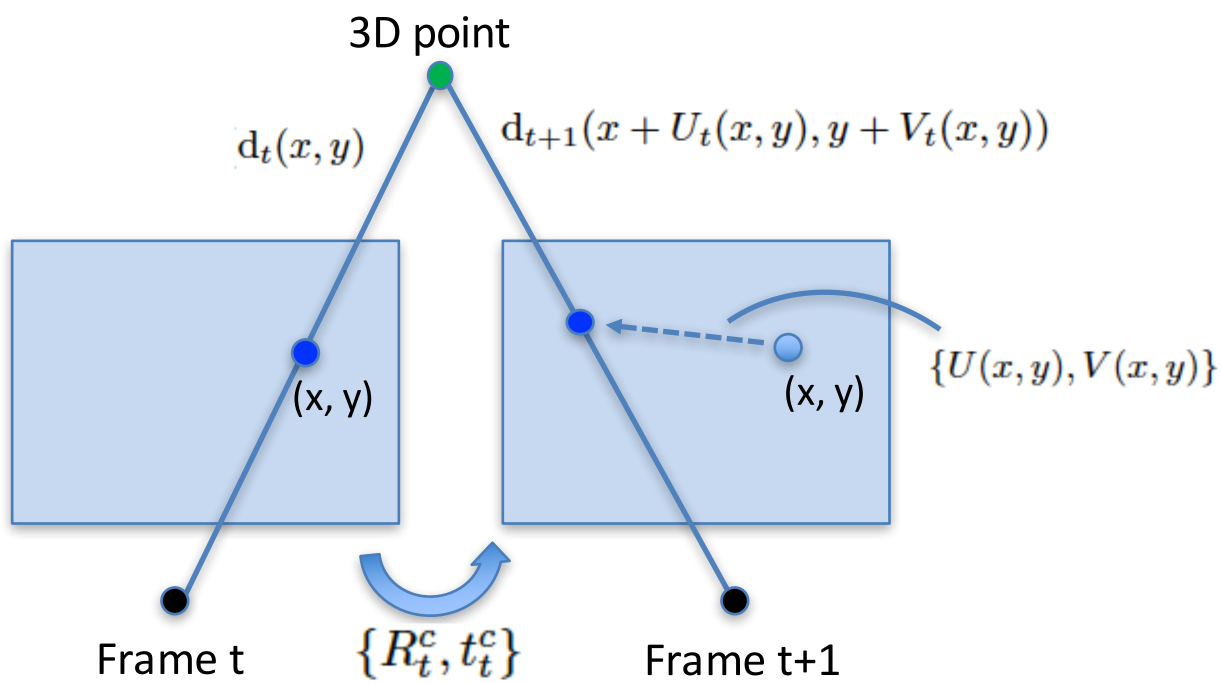
- Image intensity at a point remains the same
- Structure and motion estimates most consistent with above
- Minimize the error :

$$\sum_{x,y} \|I_t(x,y) - I_{t+1}(x',y')\|_1$$



Types of Supervision: Left-Right Consistency

- Depths of same point in two frames must be consistent with motion
- Estimate depths in frame t
- Apply estimated motion, W_t, to 3D points
- Estimate depth in frame t+1 at flow-displaced location
- Minimize error : $|(\mathrm{d}_t(x,y) + W_t(x,y)) - \mathrm{d}_{t+1}(x + U_t(x,y), y + V_t(x,y))|$



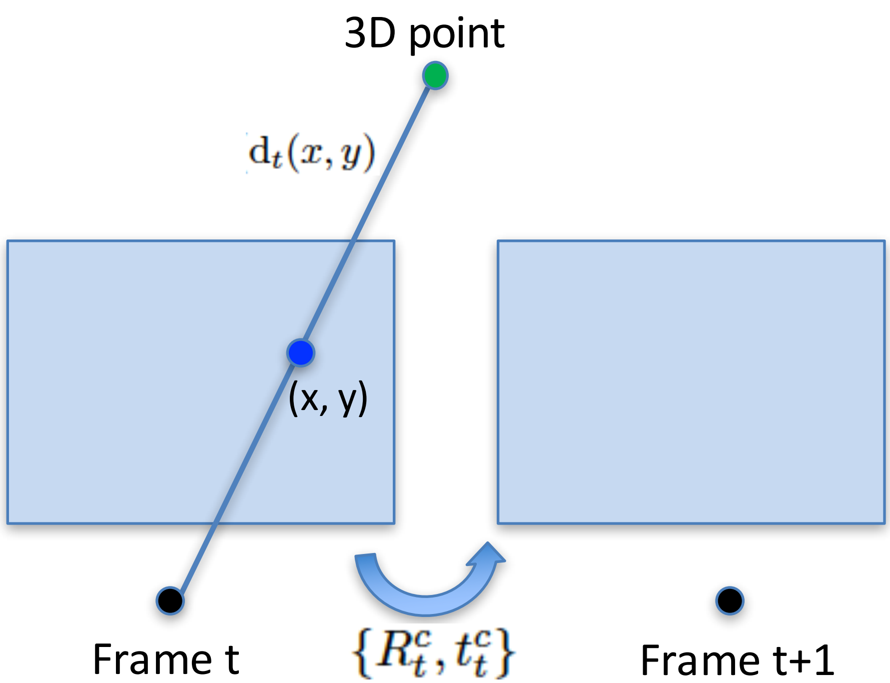
Types of Supervision: Ground Truth

- In some cases, ground truth depth is known (LIDAR, Kinect) at some points

$$\sum_{x,y} \mathrm{dmask}_t^{GT}(x,y) \cdot \| \mathrm{d}_t(x,y) - \mathrm{d}_t^{GT}(x,y) \|_1$$

- Sometimes, ground truth motion is known (GPS, IMU)

$$R_t^{\mathrm{err}} = \mathrm{inv}(R_t^c) R_t^{c-GT} \;, \quad t_t^{\mathrm{err}} = \mathrm{inv}(R_t^c)(\mathrm{t}_t^{c-GT} - \mathrm{t}_t^c)$$

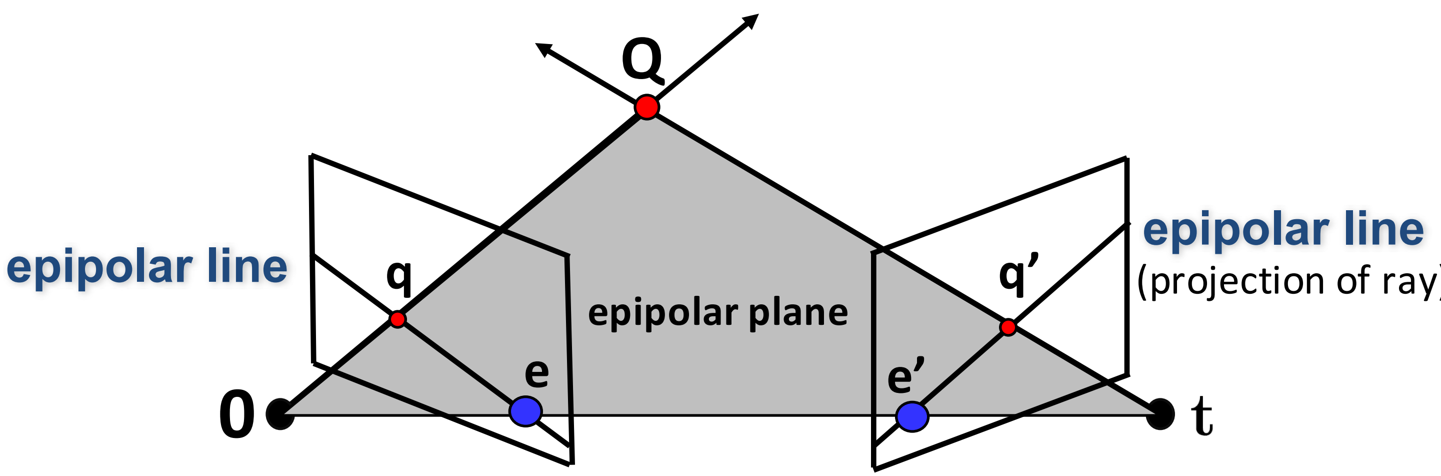


Some Actually Practical Steps

Calibrated Relative Pose Estimation

$$[t]_\times \equiv \begin{bmatrix} 0 & -t_3 & t_2 \\ t_3 & 0 & -t_1 \\ -t_2 & t_1 & 0 \end{bmatrix}$$

- Camera 1: $K_1[I \mid 0]$, camera 2: $K_2[R \mid t]$
- Fundamental matrix: $F \equiv K_2^{-\top}[t]_\times R K_1^{-1}$
- For corresponding points q and q', we have $q'^\top F q = 0$
- Condition for a matrix F to be a fundamental matrix: *det(F)* = 0



Calibrated Relative Pose Estimation

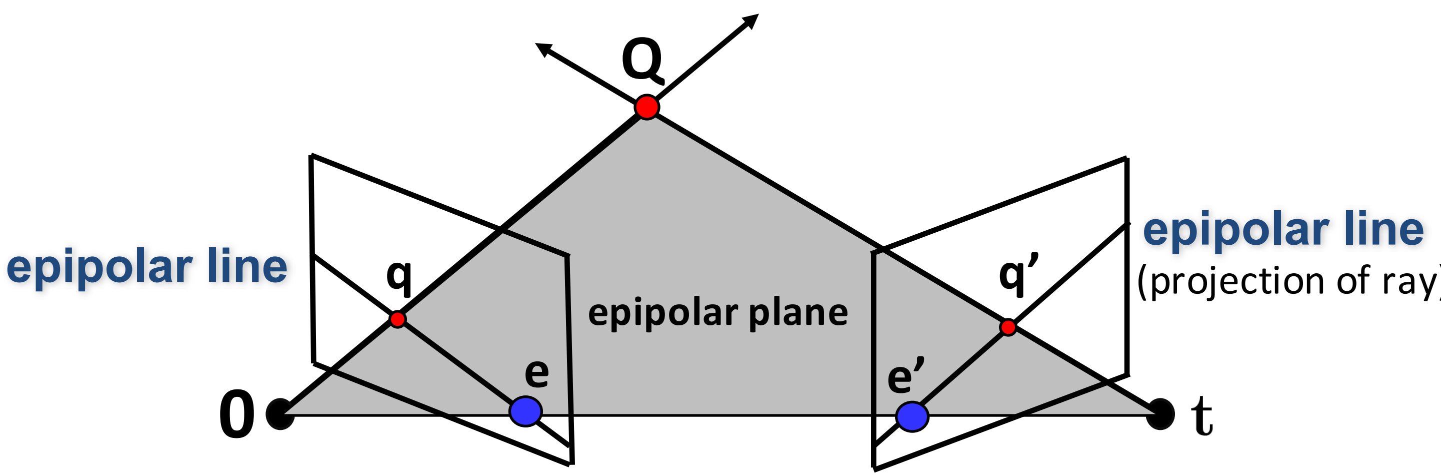
- Camera 1: $K_1[I \mid 0]$, camera 2: $K_2[R \mid t]$

$$[t]_\times \equiv \begin{bmatrix} 0 & -t_3 & t_2 \\ t_3 & 0 & -t_1 \\ -t_2 & t_1 & 0 \end{bmatrix}$$

- Fundamental matrix: $F \equiv K_2^{-\top}[t]_\times R K_1^{-1}$

Essential matrix, *E*

- For corresponding points q and q', we have $q'^\top F q = 0$
- Condition for a matrix *F* to be a fundamental matrix: *det(F)* = 0



Five-Point Method for Relative Pose Estimation

- Camera 1: $K_1[I \mid 0]$, camera 2: $K_2[R \mid t]$
- Fundamental matrix: $F \equiv K_2^{-\top}[t]_\times R K_1^{-1}$

  Essential matrix, *E*

- For corresponding points q and q', we have $q'^\top F q = 0$
- Condition for a matrix *F* to be a fundamental matrix: *det(F)* = 0
- Five degrees of freedom for the essential matrix, E

Five-Point Method for Relative Pose Estimation

- Camera 1: $K_1[I \mid 0]$, camera 2: $K_2[R \mid t]$
- Fundamental matrix: $F \equiv K_2^{-\top}[t]_\times R K_1^{-1}$

  Essential matrix, *E*
- For corresponding points q and q', we have $q'^\top F q = 0$
- Condition for a matrix *F* to be a fundamental matrix: *det(F)* = 0

- Five degrees of freedom for the essential matrix, E
- Estimating E needs 5 correspondences, better for RANSAC (F needs 7 or 8)
- Goal: should be able to estimate E efficiently from 5 correspondences

Five-Point Method for Relative Pose Estimation

- Camera 1: $K_1[I \mid 0]$, camera 2: $K_2[R \mid t]$
- Fundamental matrix: $F \equiv K_2^{-\top}[t]_\times R K_1^{-1}$

  Essential matrix, *E*
- For corresponding points q and q', we have $q'^\top F q = 0$
- Condition for a matrix *F* to be a fundamental matrix: *det(F)* = 0

- Five degrees of freedom for the essential matrix, E
- Estimating E needs 5 correspondences, better for RANSAC (F needs 7 or 8)
- Goal: should be able to estimate E efficiently from 5 correspondences

- Conditions for a matrix *E* to be an essential matrix:

  $det(E) = 0$

  $$EE^\top E - \frac{1}{2} trace(EE^\top) E = 0$$

  10 cubic equations in entries of *E*
  (not all independent)

Five-Point Method for Relative Pose Estimation

- Five point correspondences in a calibrated pair of images determine <u>*E*</u>
- Constraints from correspondences are of the form:

$$q'^{\top} E q = 0$$

[Nister, An efficient solution to the five-point relative pose problem]

Five-Point Method for Relative Pose Estimation

- Five point correspondences in a calibrated pair of images determine $\underline{E}$
- Constraints from correspondences are of the form:

$$q'^{\top} E q = 0$$

- Rewrite as 5x9 linear system, similar to fundamental matrix, with each row as:

$$\tilde{q}^{\top} \tilde{E} = 0$$

$$\tilde{q} \equiv \left[q_1 q'_1 \;\; q_2 q'_1 \;\; q_3 q'_1 \;\; q_1 q'_2 \;\; q_2 q'_2 \;\; q_3 q'_2 \;\; q_1 q'_3 \;\; q_2 q'_3 \;\; q_3 q'_3 \right]^{\top}$$

$$\tilde{E} \equiv \left[E_{11} \; E_{12} \; E_{13} \; E_{21} \; E_{22} \; E_{23} \; E_{31} \; E_{32} \; E_{33} \right]^{\top}.$$

[Nister, An efficient solution to the five-point relative pose problem]

Five-Point Method for Relative Pose Estimation

- Five point correspondences in a calibrated pair of images determine <u>*E*</u>
- Constraints from correspondences are of the form:

$$q'^{\top} E q = 0$$

- Rewrite as 5x9 linear system, similar to fundamental matrix, with each row as:

$$\tilde{q}^{\top} \tilde{E} = 0$$

$$\tilde{q} \equiv \begin{bmatrix} q_1 q'_1 & q_2 q'_1 & q_3 q'_1 & q_1 q'_2 & q_2 q'_2 & q_3 q'_2 & q_1 q'_3 & q_2 q'_3 & q_3 q'_3 \end{bmatrix}^{\top}$$

$$\tilde{E} \equiv \begin{bmatrix} E_{11} & E_{12} & E_{13} & E_{21} & E_{22} & E_{23} & E_{31} & E_{32} & E_{33} \end{bmatrix}^{\top}.$$

- System will have null space of rank 4

- Let null vectors be (*X, Y, Z, W*), then solution *E* has the form:

$$E = xX + yY + zZ + wW$$

[Nister, An efficient solution to the five-point relative pose problem]

Five-Point Method for Relative Pose Estimation

- Five point correspondences in a calibrated pair of images determine <u>*E*</u>
- Constraints from correspondences are of the form:

$$q'^{\top} E q = 0$$

- Rewrite as 5x9 linear system, similar to fundamental matrix, with each row as:

$$\tilde{q}^{\top} \tilde{E} = 0$$

$$\tilde{q} \equiv \begin{bmatrix} q_1 q'_1 & q_2 q'_1 & q_3 q'_1 & q_1 q'_2 & q_2 q'_2 & q_3 q'_2 & q_1 q'_3 & q_2 q'_3 & q_3 q'_3 \end{bmatrix}^{\top}$$

$$\tilde{E} \equiv \begin{bmatrix} E_{11} & E_{12} & E_{13} & E_{21} & E_{22} & E_{23} & E_{31} & E_{32} & E_{33} \end{bmatrix}^{\top}.$$

- System will have null space of rank 4
- Let null vectors be (*X, Y, Z, W*), then solution *E* has the form:

$$E = xX + yY + zZ + wW$$

- Since E is a homogeneous entity defined up to scale, can choose *w* = 1.

[Nister, An efficient solution to the five-point relative pose problem]

Five-Point Method for Relative Pose Estimation

- Substitute the solution for E into the cubic constraints for essential matrix

$$E = xX + yY + zZ + \cancel{w}W$$

$$det\,(E) = 0$$

$$EE^\top E - \frac{1}{2} trace(EE^\top)E = 0$$

- Obtain a system of 10 cubic equations in *x*, *y*, *z*

[Nister, An efficient solution to the five-point relative pose problem]

Five-Point Method for Relative Pose Estimation

- Use geometric constraints to obtain a system of 10 multivariate polynomials

[Nister, An efficient solution to the five-point relative pose problem]

Five-Point Method for Relative Pose Estimation

- Use geometric constraints to obtain a system of 10 multivariate polynomials
- Goal: obtain (perhaps higher order) polynomial in single variable (say, z)

[Nister, An efficient solution to the five-point relative pose problem]

Five-Point Method for Relative Pose Estimation

- Use geometric constraints to obtain a system of 10 multivariate polynomials
- Goal: obtain (perhaps higher order) polynomial in single variable (say, *z*)
- Perform Gauss-Jordan elimination

A	x^3	y^3	x^2y	xy^2	x^2z	x^2	y^2z	y^2	xyz	xy	x	y	1
$\langle a \rangle$	1	.	.	.	.	.	.	.	.	.	[2]	[2]	[3]
$\langle b \rangle$		1	.	.	.	.	.	.	.	.	[2]	[2]	[3]
$\langle c \rangle$			1	.	.	.	.	.	.	.	[2]	[2]	[3]
$\langle d \rangle$				1	.	.	.	.	.	.	[2]	[2]	[3]
$\langle e \rangle$					1						[2]	[2]	[3]
$\langle f \rangle$						1					[2]	[2]	[3]
$\langle g \rangle$							1				[2]	[2]	[3]
$\langle h \rangle$								1			[2]	[2]	[3]
$\langle i \rangle$									1		[2]	[2]	[3]
$\langle j \rangle$										1	[2]	[2]	[3]

[*n*] is polynomial in *z* of degree *n*

[Nister, An efficient solution to the five-point relative pose problem]

Five-Point Method for Relative Pose Estimation

- Use geometric constraints to obtain a system of 10 multivariate polynomials
- Goal: obtain (perhaps higher order) polynomial in single variable (say, *z*)
- Perform Gauss-Jordan elimination

A	x^3	y^3	x^2y	xy^2	x^2z	x^2	y^2z	y^2	xyz	xy	x	y	1
$\langle a \rangle$	1	.	.	.	.	.	.	.	.	.	[2]	[2]	[3]
$\langle b \rangle$		1	.	.	.	.	.	.	.	.	[2]	[2]	[3]
$\langle c \rangle$			1	.	.	.	.	.	.	.	[2]	[2]	[3]
$\langle d \rangle$				1	.	.	.	.	.	.	[2]	[2]	[3]
$\langle e \rangle$					1						[2]	[2]	[3]
$\langle f \rangle$						1					[2]	[2]	[3]
$\langle g \rangle$							1				[2]	[2]	[3]
$\langle h \rangle$								1			[2]	[2]	[3]
$\langle i \rangle$									1		[2]	[2]	[3]
$\langle j \rangle$										1	[2]	[2]	[3]

[*n*] is polynomial in *z* of degree *n*

- Eliminate leading monomials (will be in pairs related by *z* present or not)

[Nister, An efficient solution to the five-point relative pose problem]

Five-Point Method for Relative Pose Estimation

- Use geometric constraints to obtain a system of 10 multivariate polynomials
- Goal: obtain (perhaps higher order) polynomial in single variable (say, *z*)
- Perform Gauss-Jordan elimination

A	x^3	y^3	x^2y	xy^2	x^2z	x^2	y^2z	y^2	xyz	xy	x	y	1
$\langle a \rangle$	1	.	.	.	.	.	.	.	.	.	[2]	[2]	[3]
$\langle b \rangle$		1	.	.	.	.	.	.	.	.	[2]	[2]	[3]
$\langle c \rangle$			1	.	.	.	.	.	.	.	[2]	[2]	[3]
$\langle d \rangle$				1	.	.	.	.	.	.	[2]	[2]	[3]
$\langle e \rangle$					1						[2]	[2]	[3]
$\langle f \rangle$						1					[2]	[2]	[3]
$\langle g \rangle$							1				[2]	[2]	[3]
$\langle h \rangle$								1			[2]	[2]	[3]
$\langle i \rangle$									1		[2]	[2]	[3]
$\langle j \rangle$										1	[2]	[2]	[3]

[*n*] is polynomial in *z* of degree *n*

- Eliminate leading monomials (will be in pairs related by *z* present or not)

$$\langle k \rangle \equiv \langle e \rangle - z \langle f \rangle$$

[Nister, An efficient solution to the five-point relative pose problem]

Five-Point Method for Relative Pose Estimation

- Use geometric constraints to obtain a system of 10 multivariate polynomials
- Goal: obtain (perhaps higher order) polynomial in single variable (say, *z*)
- Perform Gauss-Jordan elimination

A	x^3	y^3	x^2y	xy^2	x^2z	x^2	y^2z	y^2	xyz	xy	x	y	1
$\langle a\rangle$	1	.	.	.	.	.	.	.	.	.	[2]	[2]	[3]
$\langle b\rangle$		1	.	.	.	.	.	.	.	.	[2]	[2]	[3]
$\langle c\rangle$			1	.	.	.	.	.	.	.	[2]	[2]	[3]
$\langle d\rangle$				1	.	.	.	.	.	.	[2]	[2]	[3]
$\langle e\rangle$					1						[2]	[2]	[3]
$\langle f\rangle$						1					[2]	[2]	[3]
$\langle g\rangle$							1				[2]	[2]	[3]
$\langle h\rangle$								1			[2]	[2]	[3]
$\langle i\rangle$									1		[2]	[2]	[3]
$\langle j\rangle$										1	[2]	[2]	[3]

[*n*] is polynomial in *z* of degree *n*

<*e*> <*f*>

- Eliminate leading monomials (will be in pairs related by *z* present or not)

$$\langle k\rangle \equiv \langle e\rangle - z\langle f\rangle$$

$$([2]x + [2]y + [3]) - z\,([2]x + [2]y + [3])$$

$$\Downarrow$$

$$[3]x + [3]y + [4]$$

[Nister, An efficient solution to the five-point relative pose problem]

Five-Point Method for Relative Pose Estimation

- Use geometric constraints to obtain a system of 10 multivariate polynomials
- Goal: obtain (perhaps higher order) polynomial in single variable (say, *z*)
- Perform Gauss-Jordan elimination

A	x^3	y^3	x^2y	xy^2	x^2z	x^2	y^2z	y^2	xyz	xy	x	y	1
$\langle a\rangle$	1	.	.	.	.	.	.	.	.	.	[2]	[2]	[3]
$\langle b\rangle$		1	.	.	.	.	.	.	.	.	[2]	[2]	[3]
$\langle c\rangle$			1	.	.	.	.	.	.	.	[2]	[2]	[3]
$\langle d\rangle$				1	.	.	.	.	.	.	[2]	[2]	[3]
$\langle e\rangle$					1						[2]	[2]	[3]
$\langle f\rangle$						1					[2]	[2]	[3]
$\langle g\rangle$							1				[2]	[2]	[3]
$\langle h\rangle$								1			[2]	[2]	[3]
$\langle i\rangle$									1		[2]	[2]	[3]
$\langle j\rangle$										1	[2]	[2]	[3]

[*n*] is polynomial in *z* of degree *n*

<*g*>

<*h*>

- Eliminate leading monomials (will be in pairs related by *z* present or not)

$$\langle k\rangle \equiv \langle e\rangle - z\langle f\rangle$$
$$\langle l\rangle \equiv \langle g\rangle - z\langle h\rangle$$

$([2]x + [2]y + [3]) - z\,([2]x + [2]y + [3])$

$[3]x + [3]y + [4]$

[Nister, An efficient solution to the five-point relative pose problem]

Five-Point Method for Relative Pose Estimation

- Use geometric constraints to obtain a system of 10 multivariate polynomials
- Goal: obtain (perhaps higher order) polynomial in single variable (say, z)
- Perform Gauss-Jordan elimination

A	x^3	y^3	x^2y	xy^2	x^2z	x^2	y^2z	y^2	xyz	xy	x	y	1
$\langle a\rangle$	1	.	.	.	.	.	.	.			[2]	[2]	[3]
$\langle b\rangle$		1	.	.	.	.	.	.	.	.	[2]	[2]	[3]
$\langle c\rangle$			1	.	.	.	.	.	.	.	[2]	[2]	[3]
$\langle d\rangle$				1	.	.	.	.	.	.	[2]	[2]	[3]
$\langle e\rangle$					1						[2]	[2]	[3]
$\langle f\rangle$						1					[2]	[2]	[3]
$\langle g\rangle$							1				[2]	[2]	[3]
$\langle h\rangle$								1			[2]	[2]	[3]
$\langle i\rangle$									1		[2]	[2]	[3]
$\langle j\rangle$										1	[2]	[2]	[3]

[n] is polynomial in z of degree n

<i>

<j>

- Eliminate leading monomials (will be in pairs related by z present or not)

$$\langle k\rangle \equiv \langle e\rangle - z\langle f\rangle$$
$$\langle l\rangle \equiv \langle g\rangle - z\langle h\rangle$$
$$\langle m\rangle \equiv \langle i\rangle - z\langle j\rangle.$$

$$([2]x + [2]y + [3]) - z\,([2]x + [2]y + [3])$$

$$\Downarrow$$

$$[3]x + [3]y + [4]$$

[Nister, An efficient solution to the five-point relative pose problem]

Five-Point Method for Relative Pose Estimation

- Use geometric constraints to obtain a system of 10 multivariate polynomials
- Goal: obtain (perhaps higher order) polynomial in single variable (say, *z*)
- Perform Gauss-Jordan elimination

A	x^3	y^3	x^2y	xy^2	x^2z	x^2	y^2z	y^2	xyz	xy	x	y	1
$\langle a\rangle$	1	.	.	.	.	.	.	.	.	.	[2]	[2]	[3]
$\langle b\rangle$		1	.	.	.	.	.	.	.	.	[2]	[2]	[3]
$\langle c\rangle$			1	.	.	.	.	.	.	.	[2]	[2]	[3]
$\langle d\rangle$				1	.	.	.	.	.	.	[2]	[2]	[3]
$\langle e\rangle$					1						[2]	[2]	[3]
$\langle f\rangle$						1					[2]	[2]	[3]
$\langle g\rangle$							1				[2]	[2]	[3]
$\langle h\rangle$								1			[2]	[2]	[3]
$\langle i\rangle$									1		[2]	[2]	[3]
$\langle j\rangle$										1	[2]	[2]	[3]

[*n*] is polynomial in *z* of degree *n*

- Eliminate leading monomials (will be in pairs related by *z* present or not)

$$\langle k\rangle \equiv \langle e\rangle - z\langle f\rangle$$
$$\langle l\rangle \equiv \langle g\rangle - z\langle h\rangle$$
$$\langle m\rangle \equiv \langle i\rangle - z\langle j\rangle.$$

Now <*k*>, <*l*>, <*m*> have forms $f_1(z)\ x + f_2(z)\ y + f_3 = 0$

[Nister, An efficient solution to the five-point relative pose problem]

Five-Point Method for Relative Pose Estimation

- Arrange equations into a linear system composed of polynomials in *z*:

B	x	y	1
$\langle k \rangle$	[3]	[3]	[4]
$\langle l \rangle$	[3]	[3]	[4]
$\langle m \rangle$	[3]	[3]	[4]

- Since (x, y, 1) is in the null space of B, we have *det(B)* = 0
- All entries of B are polynomials in *z*, so *det(B)* is a degree 10 polynomial in *z*
- Univariate, low degree polynomial: can be solved efficiently!
- For a root *z*, can find *x*, *y* by solving the resulting linear system *B*
- From *x*, *y*, *z* (and *w* = 1), we have the desired essential matrix:

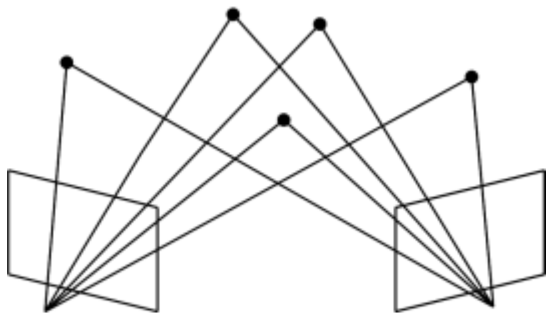
$$E = xX + yY + zZ + wW$$

[Nister, An efficient solution to the five-point relative pose problem]

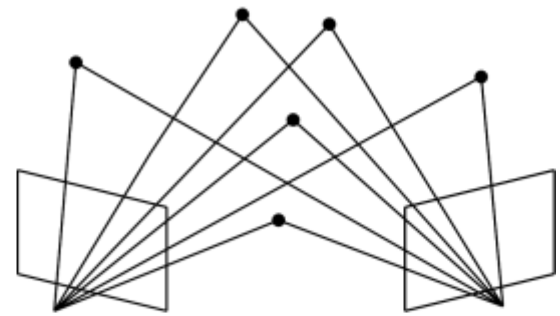
Minimal Problems in Computer Vision

- There is a large zoo of minimal problems that have known solutions

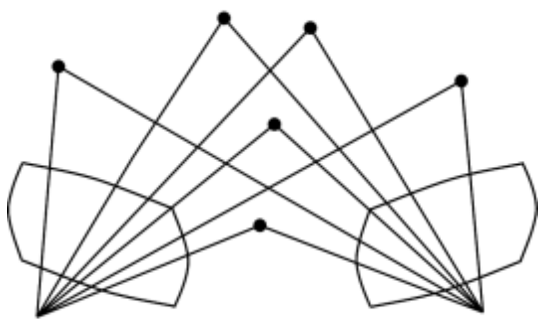
5-point relative pose with known K



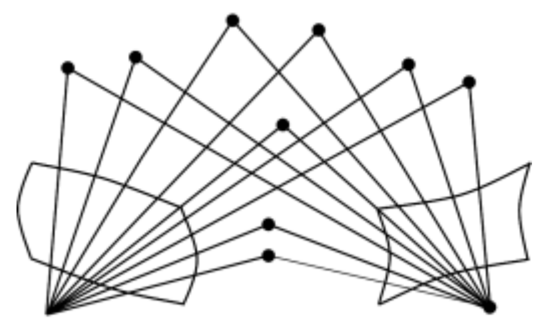
6-point relative pose with unknown f



6-point relative pose with unknown r

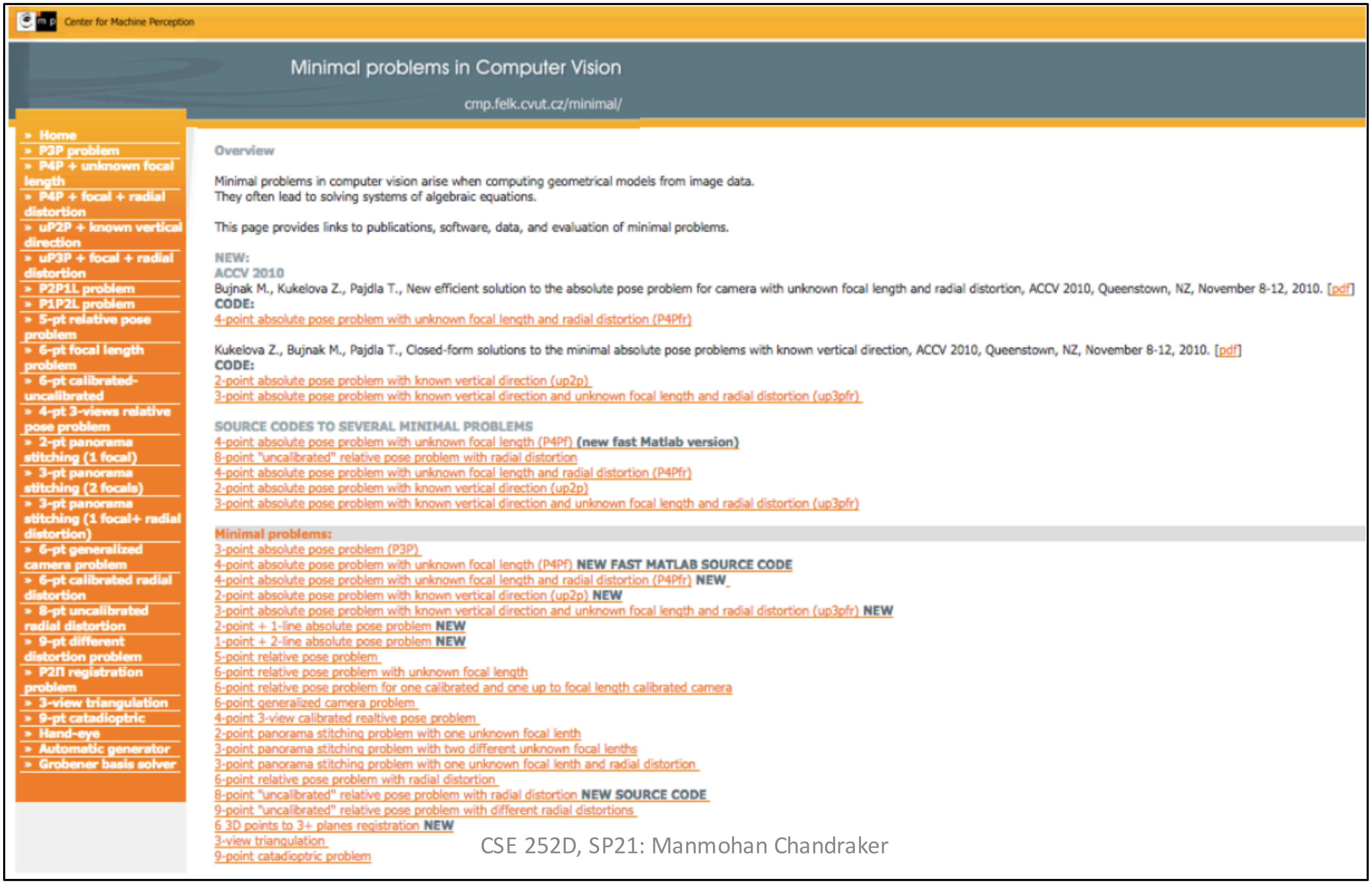


9-point relative pose with unknown and different f, r



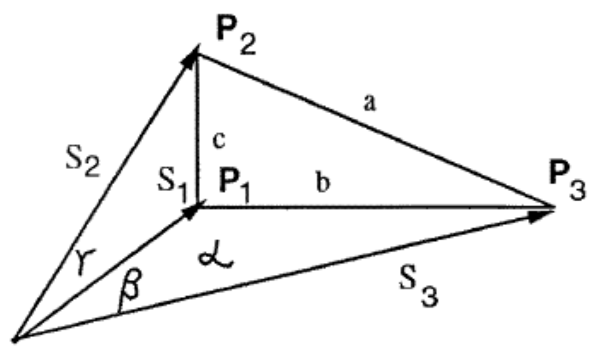
Minimal Problems in Computer Vision

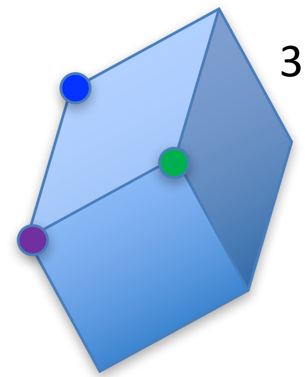
- There is a large zoo of minimal problems that have known solutions

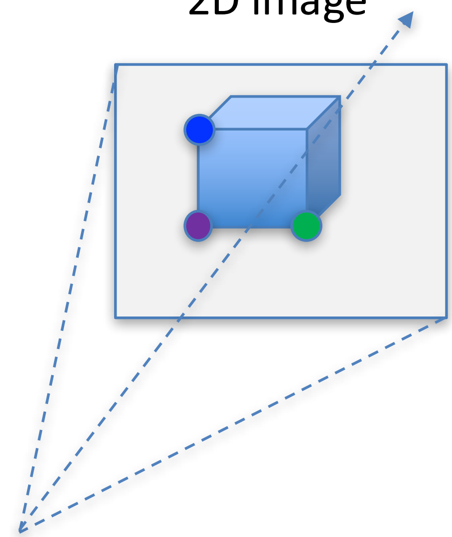


Three-Point Absolute Pose Estimation

- Find camera pose from given 3D-2D correspondences
- Minimal case is 3 points: 6 degrees of freedom, 2 constraints per point (x, y)
- Given p_1, p_2, p_3 in world coordinates, find positions in camera coordinates



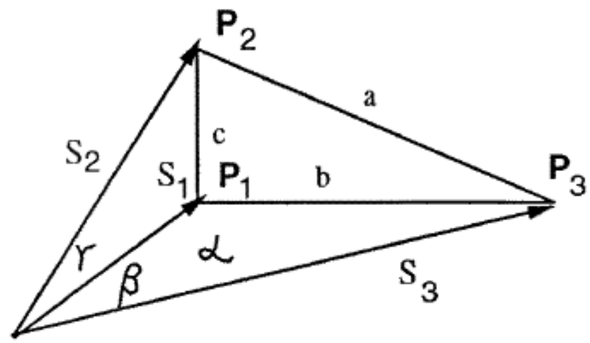




Determine how to move camera, such that corresponding 2D-3D points align

Three-Point Absolute Pose Estimation

- Find camera pose from given 3D-2D correspondences
- Minimal case is 3 points: 6 degrees of freedom, 2 constraints per point (x, y)
- Given p_1, p_2, p_3 in world coordinates, find positions in camera coordinates



Knowns

$$a = \|p_2 - p_3\|$$
$$b = \|p_1 - p_3\|$$
$$c = \|p_1 - p_2\|.$$

Distances between points

$$u_i = f\frac{x_i}{z_i}$$
$$v_i = f\frac{y_i}{z_i}.$$

Image points

$$\cos\alpha = j_2 \cdot j_3$$
$$\cos\beta = j_1 \cdot j_3$$
$$\cos\gamma = j_1 \cdot j_2$$

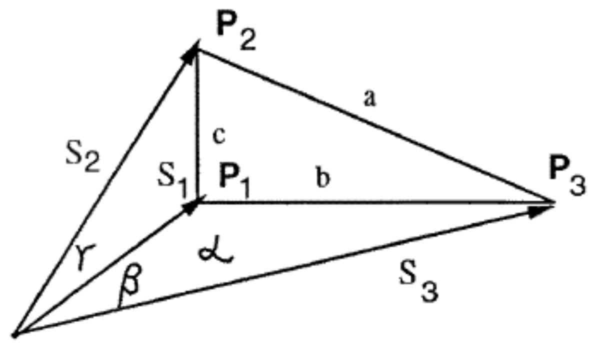
Angle between image rays j_i

Unknowns

Distance to 3D points from camera center, s_i

Three-Point Absolute Pose Estimation

- Find camera pose from given 3D-2D correspondences
- Minimal case is 3 points: 6 degrees of freedom, 2 constraints per point (x, y)
- Given p_1, p_2, p_3 in world coordinates, find positions in camera coordinates



Knowns

$$a = \|p_2 - p_3\| \quad b = \|p_1 - p_3\| \quad c = \|p_1 - p_2\|.$$

Distances between points

$$u_i = f\frac{x_i}{z_i} \qquad v_i = f\frac{y_i}{z_i}.$$

Image points

$$\cos\alpha = j_2 \cdot j_3 \quad \cos\beta = j_1 \cdot j_3 \quad \cos\gamma = j_1 \cdot j_2$$

Angle between image rays j_i

Constraints: Law of cosines

$$s_2^2 + s_3^2 - 2s_2s_3\cos\alpha = a^2$$
$$s_1^2 + s_3^2 - 2s_1s_3\cos\beta = b^2$$
$$s_1^2 + s_2^2 - 2s_1s_2\cos\gamma = c^2$$

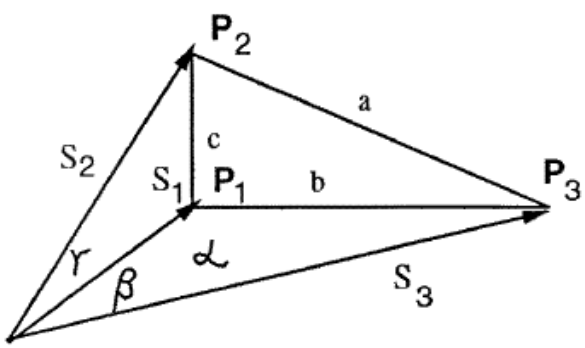
Unknowns

Distance to 3D points from camera center, s_i

All techniques yield fourth-degree polynomial

Three-Point Absolute Pose Estimation

- Find camera pose from given 3D-2D correspondences
- Minimal case is 3 points: 6 degrees of freedom, 2 constraints per point (x, y)
- Given p_1, p_2, p_3 in world coordinates, find positions in camera coordinates



Constraints: Law of cosines

$$s_2^2 + s_3^2 - 2s_2s_3\cos\alpha = a^2$$
$$s_1^2 + s_3^2 - 2s_1s_3\cos\beta = b^2$$
$$s_1^2 + s_2^2 - 2s_1s_2\cos\gamma = c^2$$

All techniques yield fourth-degree polynomial

Finsterwalder's technique recommended by [Haralick et al. 1994] as best numerically

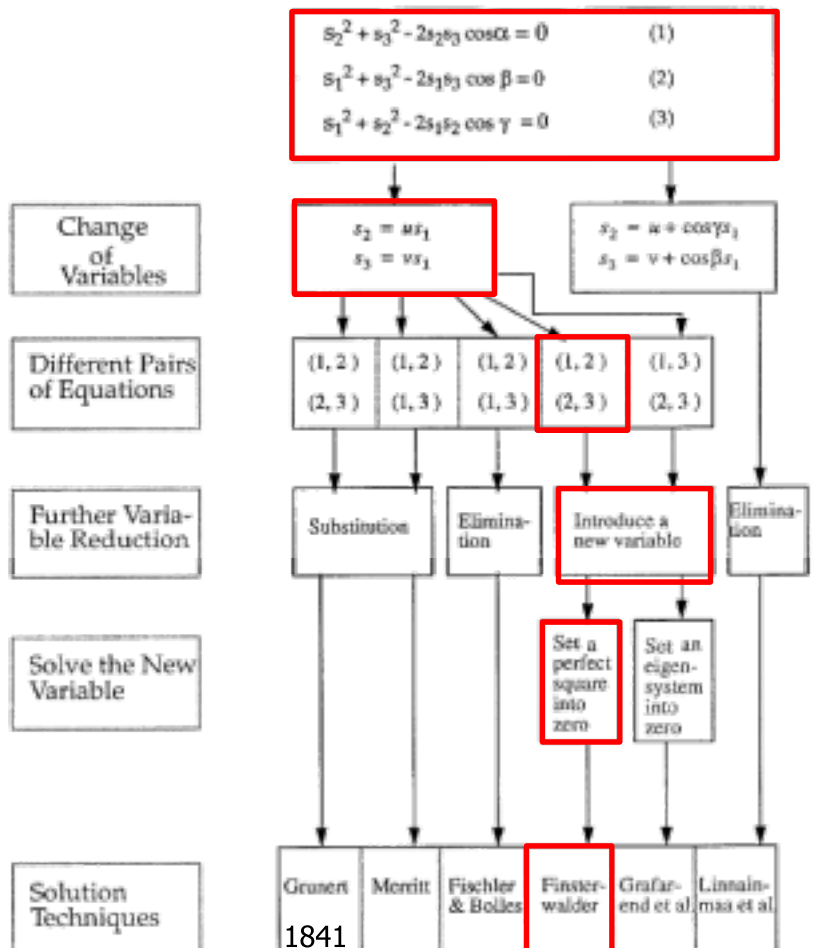
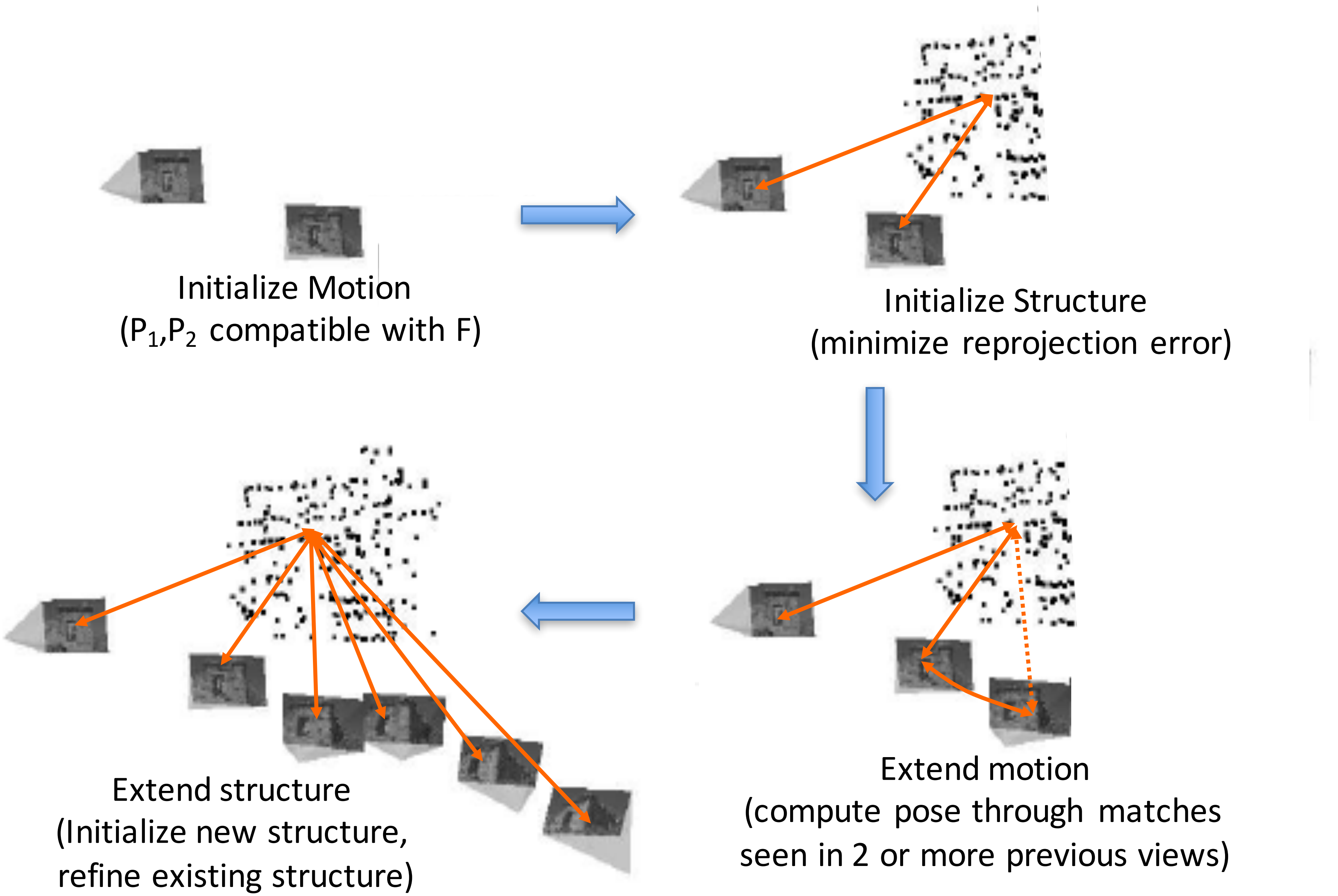


Fig. 2 Shows the differences of a algebraic derivations among six solution techniques.

Sequential Structure from Motion



Some Recipes for SFM to Work

- Do everything you can to remove outliers

- Solve minimal problems to estimate geometric entities
  - Keeps RANSAC tractable
  - Typically, expect to spend 0.01ms

- Strategically consider what variables to optimize
  - Keyframe-based designs are successful
  - Try to robustly build long feature tracks
  - Do bundle adjustment whenever possible

- Drift is inevitable, so have a plan to address it
  - Local scale correction and global pose correction when possible