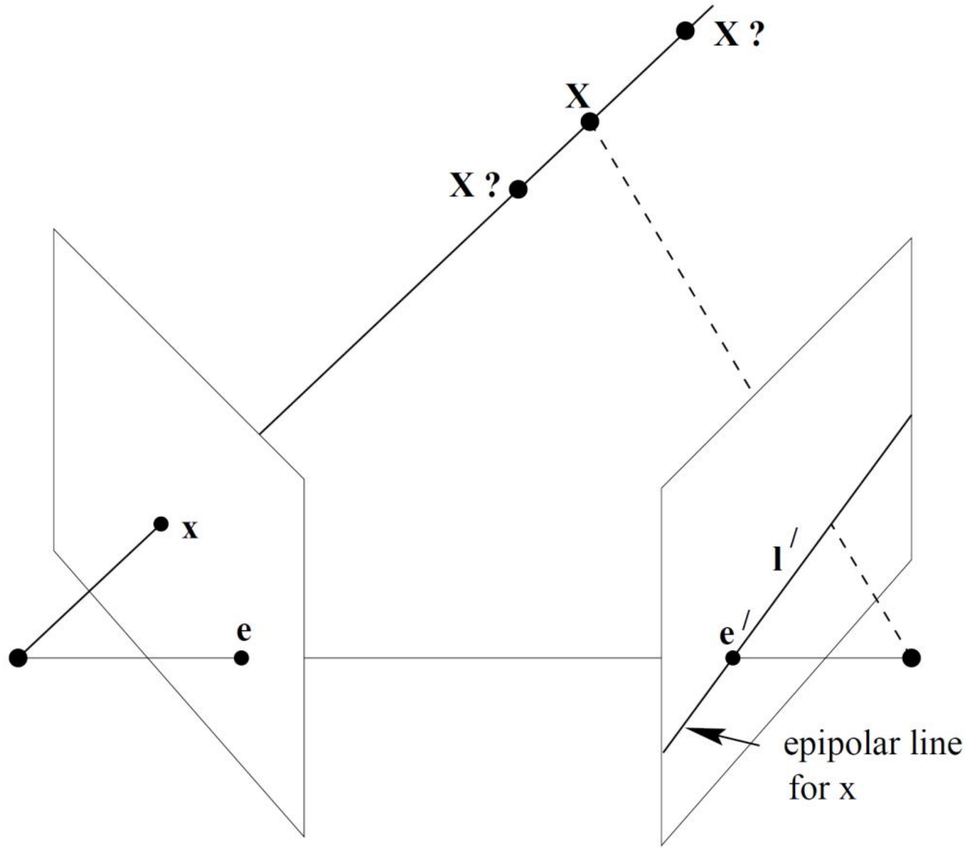
Structure from Motion I

CS 4391 Introduction Computer Vision

Professor Yu Xiang

The University of Texas at Dallas

Recall Fundamental Matrix



- Epipolar line

$$\mathbf{l}' = F\mathbf{x}$$

$$\mathbf{l} = F^T\mathbf{x}'$$

- Fundamental matrix

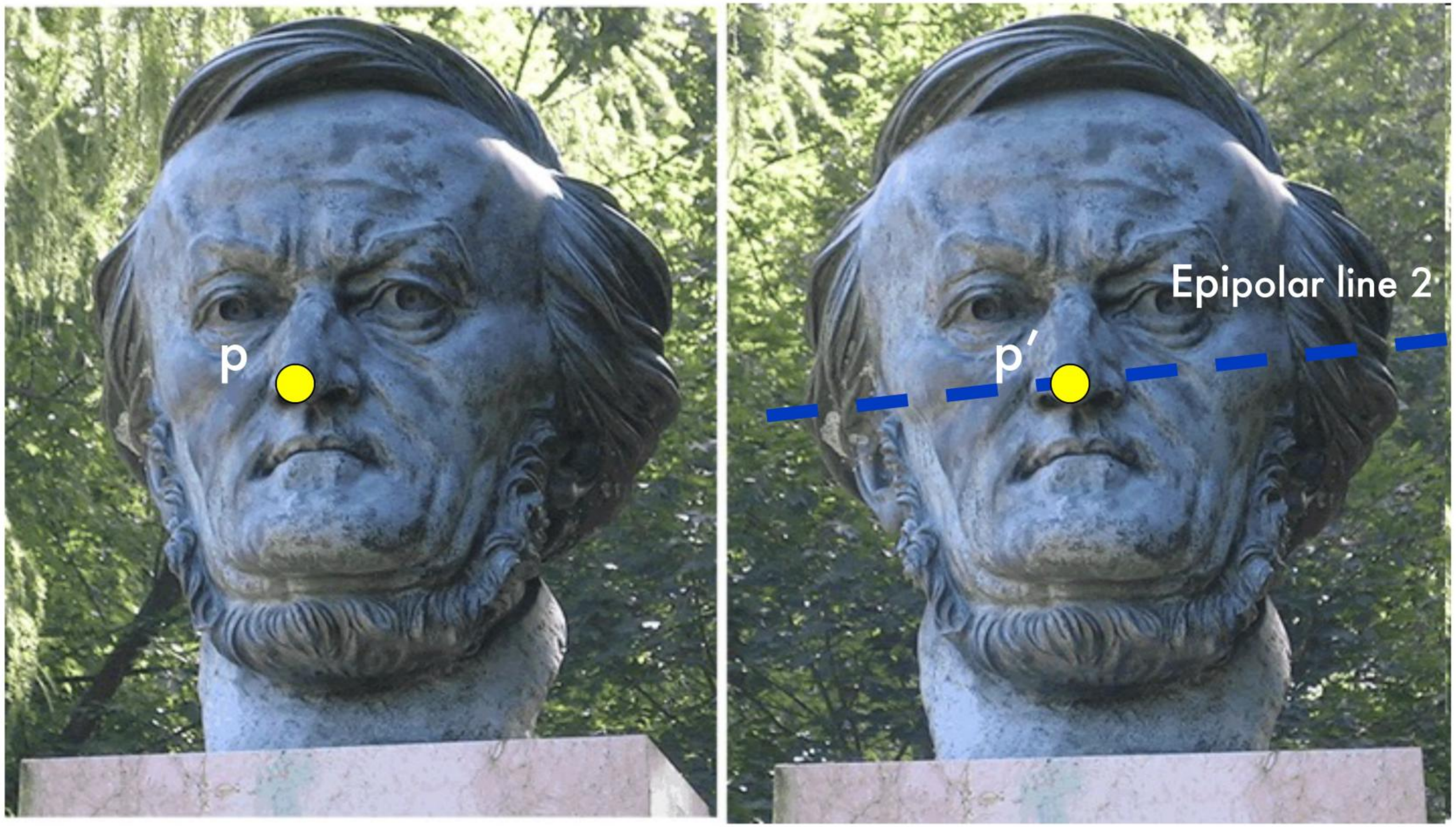
$$F = [\mathbf{e}']_\times P'P^+$$

3x3

Epipole $\mathbf{e}' = (P'C)$

$$P^+ = P^T(PP^T)^{-1}$$

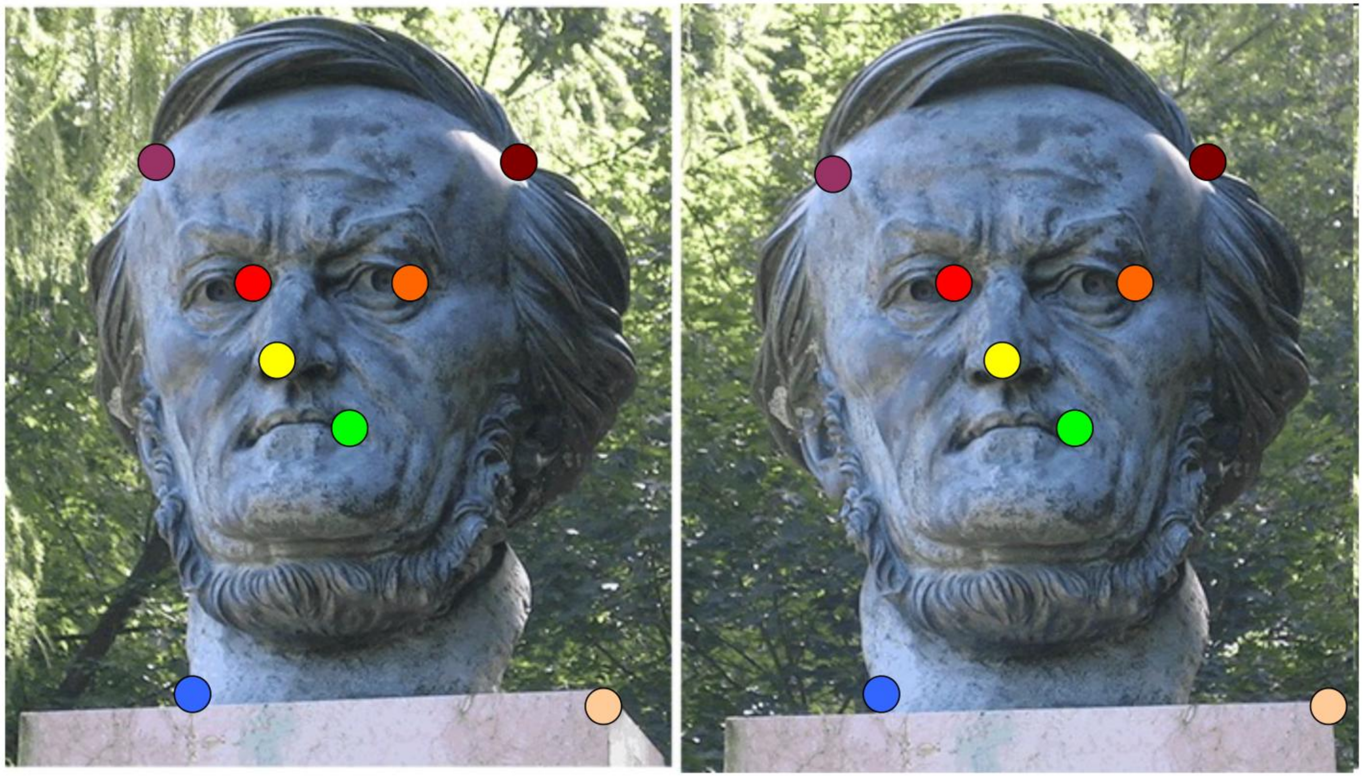
Why the Fundamental Matrix is Useful?



$$\mathbf{l}' = F\mathbf{p}$$

Estimating the Fundamental Matrix

- The 8-point algorithm

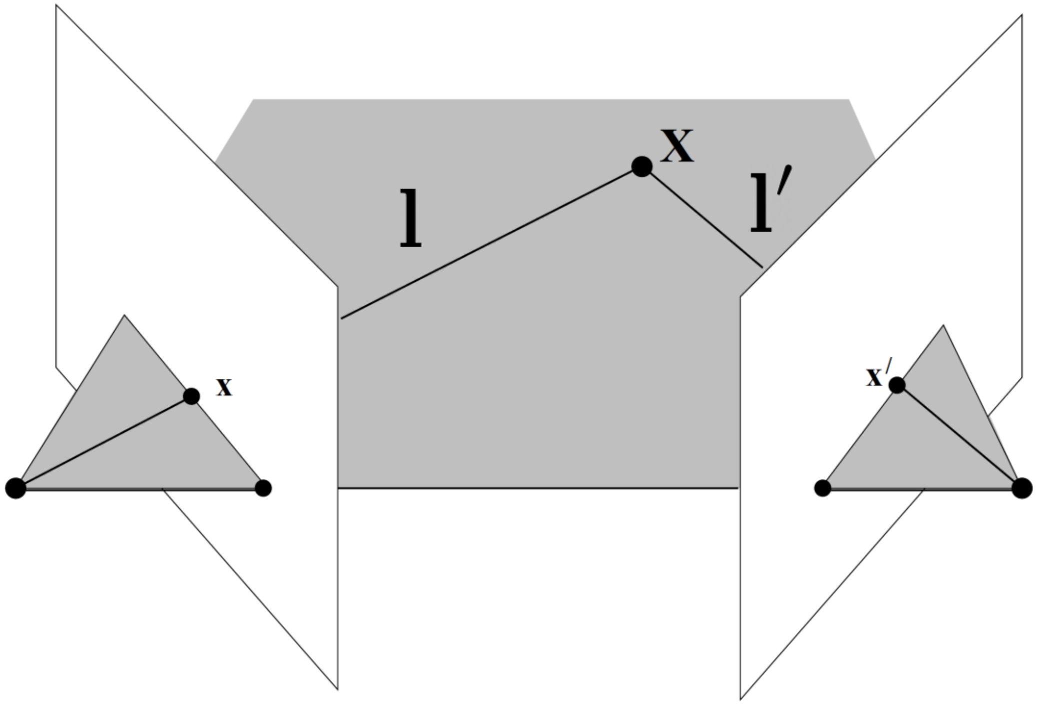


$$\mathbf{l}' = F\mathbf{x}$$

$$\mathbf{x}'^{\top}\mathbf{F}\mathbf{x} = 0$$

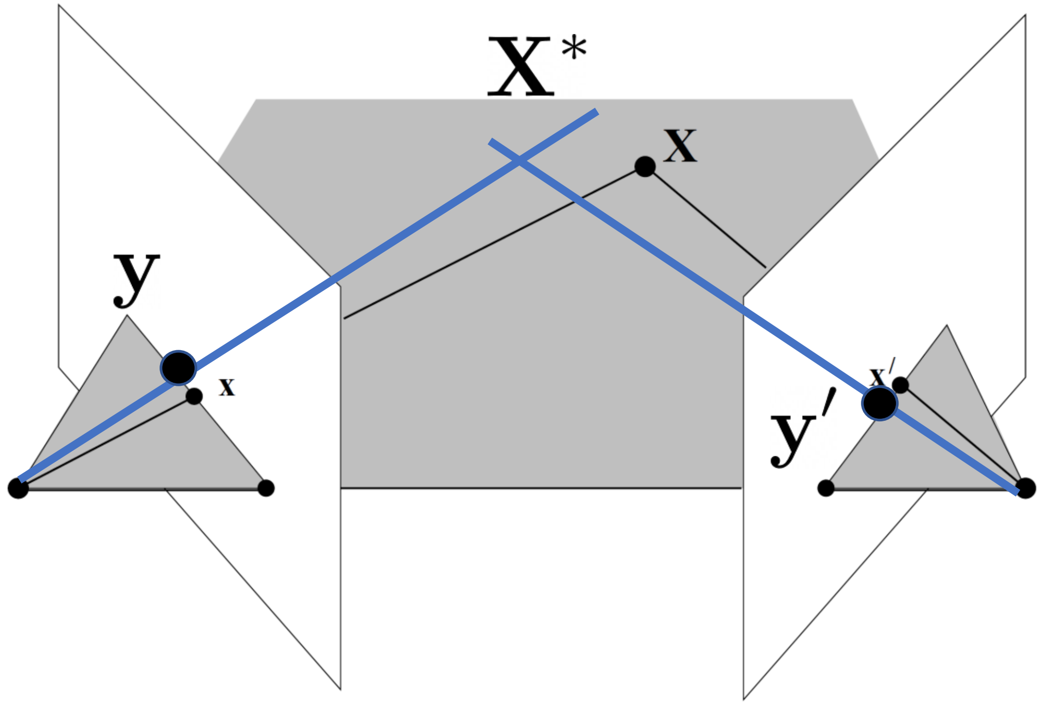
Triangulation

- Compute the 3D point given image correspondences



Intersection of two backprojected lines

Triangulation



- In practice, we find the correspondences

  $\mathbf{y} \;\; \mathbf{y}'$

- The backprojected lines may not intersect
- Find X* that minimizes

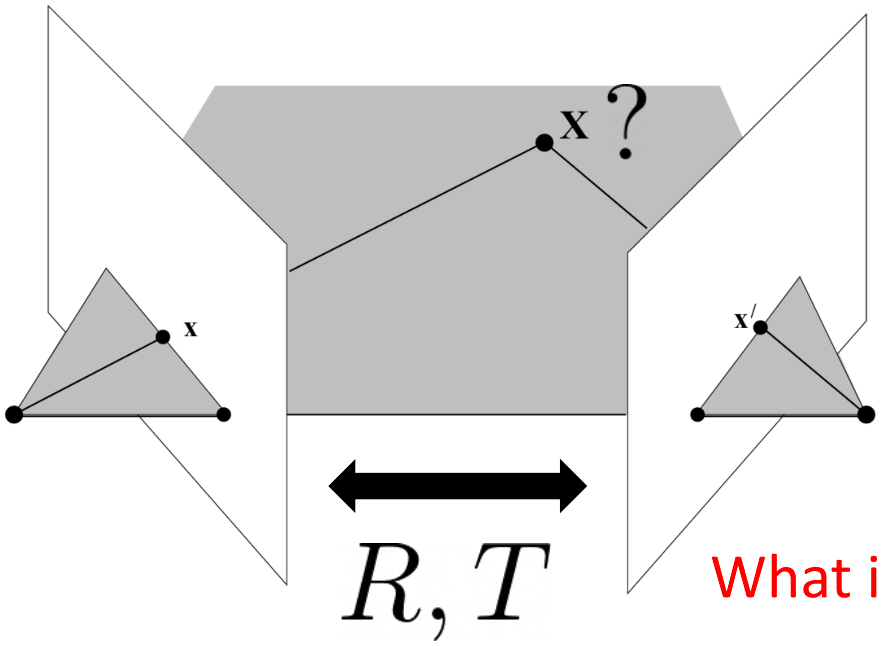
$$d(\mathbf{y}, P\mathbf{X}^*) + d(\mathbf{y}', P'\mathbf{X}^*)$$

Projection matrix

Triangulation

- Idea: using images from different views and feature matching

- Triangulation from pixel correspondences to compute 3D location



Given $\mathbf{x} \longleftrightarrow \mathbf{x}'$

Intersection of two backprojected lines

What if unknow camera pose?

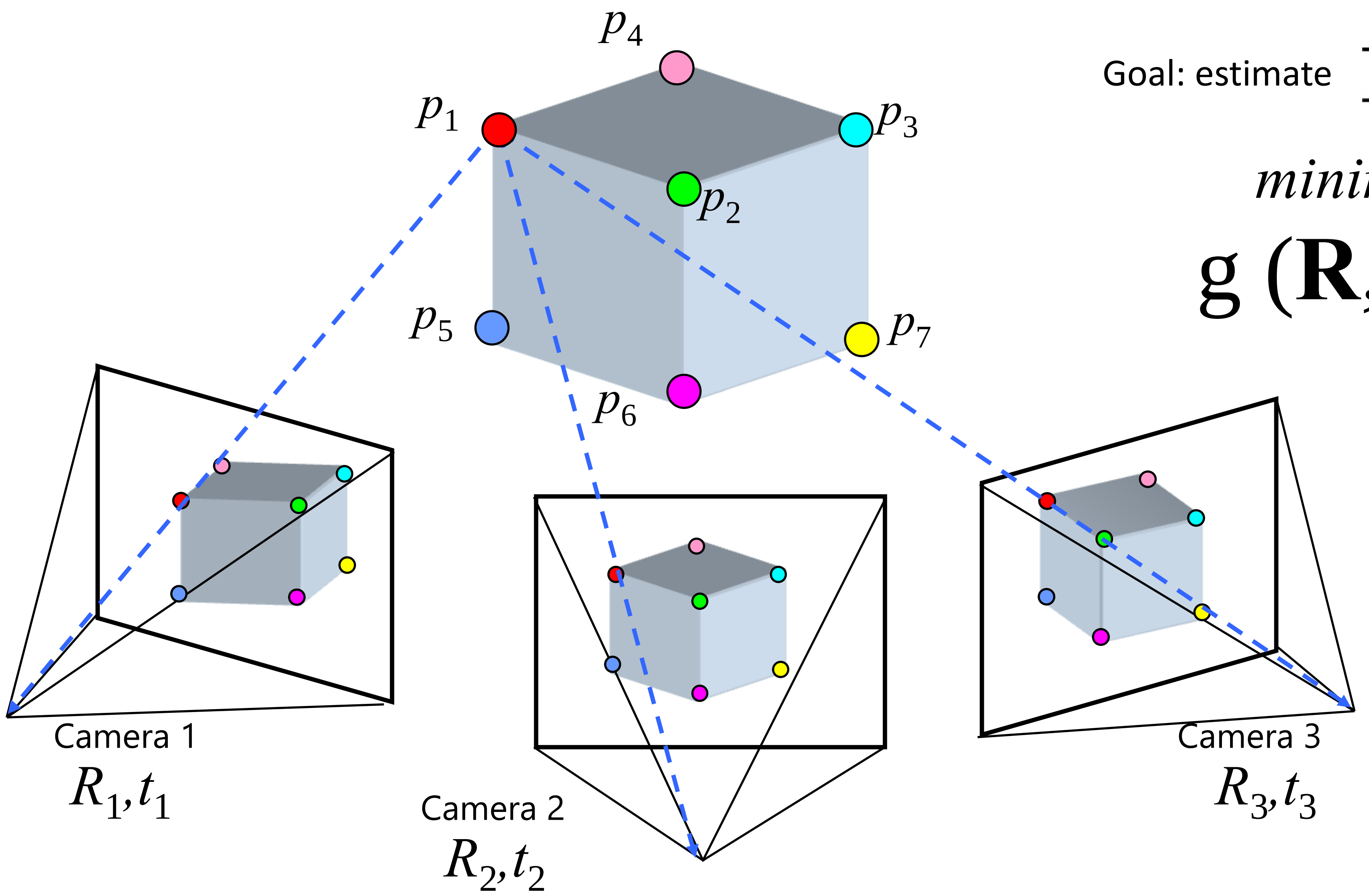
Structure from Motion

- Input
  - A set of images from different views

- Output
  - 3D Locations of all feature points in a world frame
  - Camera poses of the images

Structure from Motion

Structure from motion



Goal: estimate $\mathbf{R}, \mathbf{T}, \mathbf{P}$

minimize

$$\mathbf{g}\,(\mathbf{R}, \mathbf{T}, \mathbf{P})$$

Structure from Motion

- Minimize sum of squared reprojection errors

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

m points, n images

Indicator variable:
is point i visible in image j?

Projection

$$\begin{bmatrix} x' \\ y' \\ z' \end{bmatrix} = \mathbf{Rx} + \mathbf{t}$$

$$u' = f_x \frac{x'}{z'} + p_x$$

$$v' = f_y \frac{y'}{z'} + p_y$$

$$\begin{bmatrix} u' \\ v' \end{bmatrix} = \mathbf{P}(\mathbf{x},\mathbf{R},\mathbf{t})$$

Structure from Motion

- How to minimize

$$g(\mathbf{X}, \mathbf{R}, \mathbf{T}) = \sum_{i=1}^{m} \sum_{j=1}^{n} w_{ij} \cdot \left\| \mathbf{P}(\mathbf{x}_i, \mathbf{R}_j, \mathbf{t}_j) - \begin{bmatrix} u_{i,j} \\ v_{i,j} \end{bmatrix} \right\|^2$$

- A non-linear least squares problem (why?)
  - E.g. Levenberg-Marquardt

The Levenberg-Marquardt Algorithm

- Nonlinear least squares $\hat{\boldsymbol{\beta}} \in \operatorname{argmin}_{\boldsymbol{\beta}} S(\boldsymbol{\beta}) \equiv \operatorname{argmin}_{\boldsymbol{\beta}} \sum_{i=1}^{m} \left[y_i - f(x_i, \boldsymbol{\beta})\right]^2$

  $n \times 1$

- An iterative algorithm
  - Start with an initial guess β_0
  - For each iteration $\beta \leftarrow \beta + \delta$

- How to get δ?
  - Linear approximation $f(x_i, \boldsymbol{\beta} + \boldsymbol{\delta}) \approx f(x_i, \boldsymbol{\beta}) + \mathbf{J}_i \boldsymbol{\delta}$ $\quad \mathbf{J}_i = \frac{\partial f(x_i, \boldsymbol{\beta})}{\partial \boldsymbol{\beta}}$ $\quad 1 \times n$
  - Find δ to minimize the objective $S(\boldsymbol{\beta} + \boldsymbol{\delta}) \approx \sum_{i=1}^{m} \left[y_i - f(x_i, \boldsymbol{\beta}) - \mathbf{J}_i \boldsymbol{\delta}\right]^2$

Best to minimize the objective

Wikipedia

The Levenberg-Marquardt Algorithm

- Vector notation for $S(\boldsymbol\beta + \boldsymbol\delta) \approx \sum_{i=1}^{m} \left[y_i - f(x_i, \boldsymbol\beta) - \mathbf{J}_i \boldsymbol\delta\right]^2$

$$\begin{aligned} S(\boldsymbol\beta + \boldsymbol\delta) &\approx \|\mathbf{y} - \mathbf{f}(\boldsymbol\beta) - \mathbf{J}\boldsymbol\delta\|^2 \\ &= \left[\mathbf{y} - \mathbf{f}(\boldsymbol\beta) - \mathbf{J}\boldsymbol\delta\right]^{\mathrm{T}} \left[\mathbf{y} - \mathbf{f}(\boldsymbol\beta) - \mathbf{J}\boldsymbol\delta\right] \\ &= \left[\mathbf{y} - \mathbf{f}(\boldsymbol\beta)\right]^{\mathrm{T}} \left[\mathbf{y} - \mathbf{f}(\boldsymbol\beta)\right] - \left[\mathbf{y} - \mathbf{f}(\boldsymbol\beta)\right]^{\mathrm{T}} \mathbf{J}\boldsymbol\delta - (\mathbf{J}\boldsymbol\delta)^{\mathrm{T}} \left[\mathbf{y} - \mathbf{f}(\boldsymbol\beta)\right] + \boldsymbol\delta^{\mathrm{T}}\mathbf{J}^{\mathrm{T}}\mathbf{J}\boldsymbol\delta \\ &= \left[\mathbf{y} - \mathbf{f}(\boldsymbol\beta)\right]^{\mathrm{T}} \left[\mathbf{y} - \mathbf{f}(\boldsymbol\beta)\right] - 2\left[\mathbf{y} - \mathbf{f}(\boldsymbol\beta)\right]^{\mathrm{T}} \mathbf{J}\boldsymbol\delta + \boldsymbol\delta^{\mathrm{T}}\mathbf{J}^{\mathrm{T}}\mathbf{J}\boldsymbol\delta. \end{aligned}$$

Take derivation with respect to δ and set to zero $\left(\mathbf{J}^{\mathrm{T}}\mathbf{J}\right)\boldsymbol\delta = \mathbf{J}^{\mathrm{T}}\left[\mathbf{y} - \mathbf{f}(\boldsymbol\beta)\right]$ https://www.cs.ubc.ca/~schmidtm/Courses/340-F16/linearQuadraticGradients.pdf

Levenberg's contribution $\left(\mathbf{J}^{\mathrm{T}}\mathbf{J} + \lambda\mathbf{I}\right)\boldsymbol\delta = \mathbf{J}^{\mathrm{T}}\left[\mathbf{y} - \mathbf{f}(\boldsymbol\beta)\right]$ damped version

$$\beta \leftarrow \beta + \delta$$

Wikipedia

Structure from Motion

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

$$\beta = (\mathbf{X},\mathbf{R},\mathbf{T})$$

How to get the initial estimation β_0 ?

Random guess is not a good idea.

Next Lecture

Further Reading

- Chapter 11, Computer Vision, Richard Szeliski
- Build Rome in One Day https://grail.cs.washington.edu/rome/
- Structure from Motion Revisited https://colmap.github.io/index.html