Pose Tracking: Structure from Motion and SLAM

CS 6334 Virtual Reality

Professor Yapeng Tian

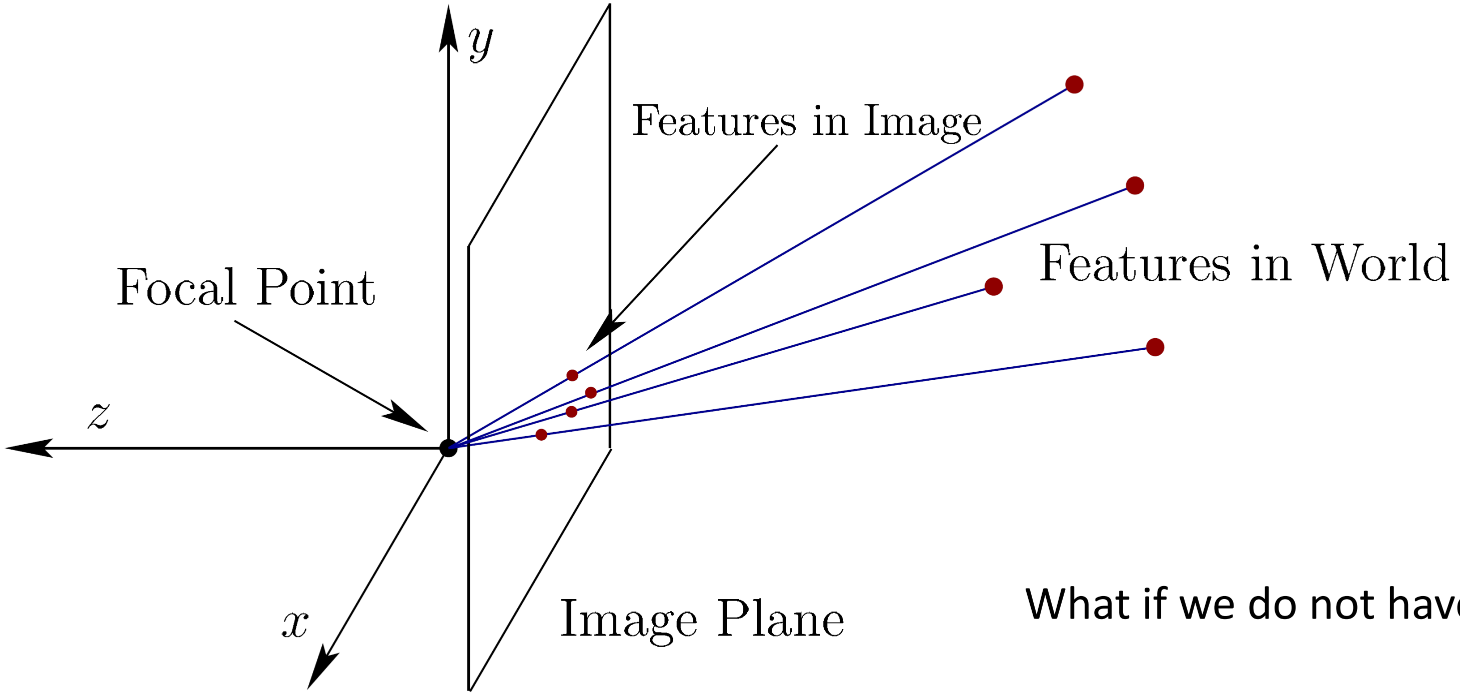
The University of Texas at Dallas

A lot of slides of course lectures borrowed from Professor Yu Xiang's VR class

Tracking in VR

- Tracking the user’s sense organs
  - E.g., Head and eye
  - Render stimulus accordingly
- Tracking user’s other body parts
  - E.g., human body and hands
  - Locomotion and manipulation
- Tracking the rest of the environment
  - Augmented reality
  - Obstacle avoidance in the real world

Feature-based Tracking



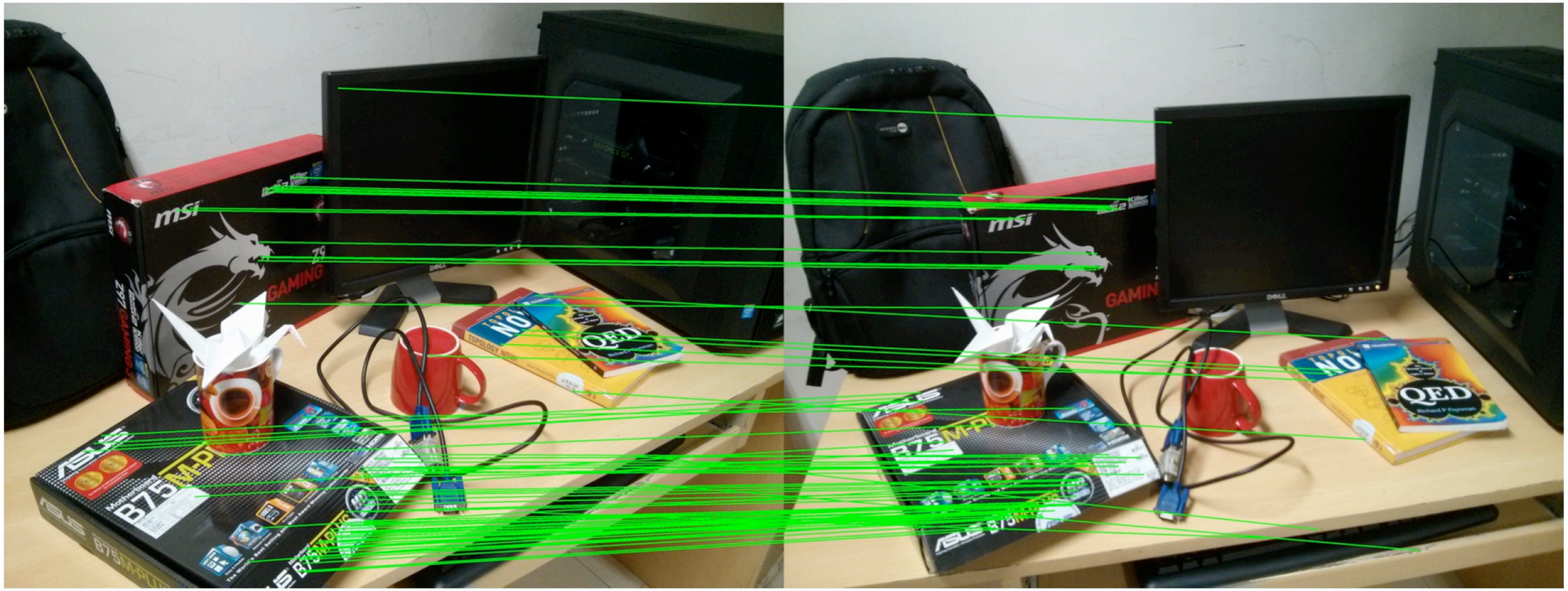
The PnP problem

- Known: 3D locations, 2D locations, camera intrinsics
- Unknown:
  6D pose of the camera

What if we do not have the 3D locations of these feature points?

Feature-based Tracking

- Idea: using images from different views and feature matching

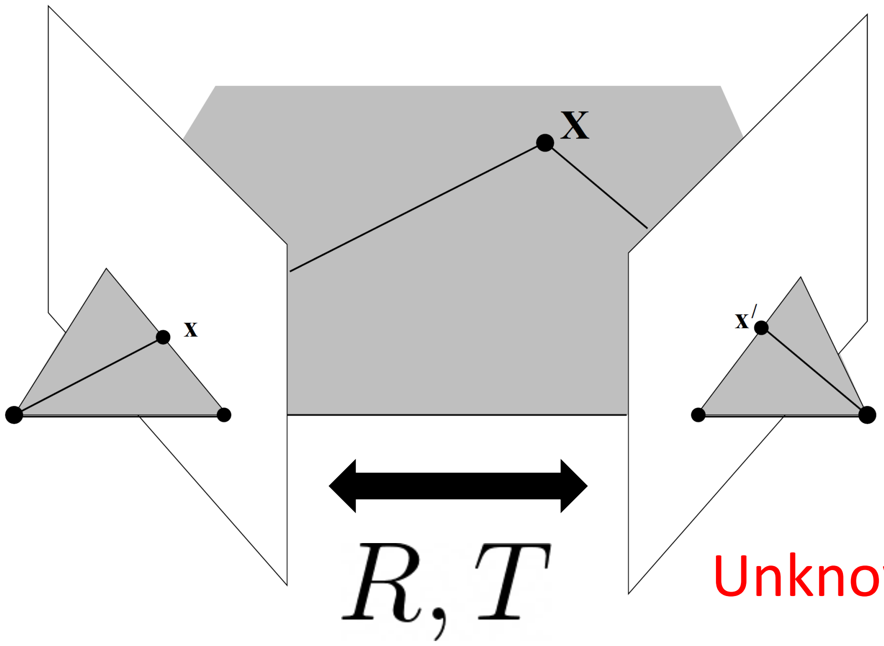


Geometry-aware Feature Matching for Structure from Motion Applications. Shah et al, WACV'15

Feature-based Tracking

- Idea: using images from different views and feature matching

- Triangulation from pixel correspondences to compute 3D location



Unknow

Intersection of two backprojected lines

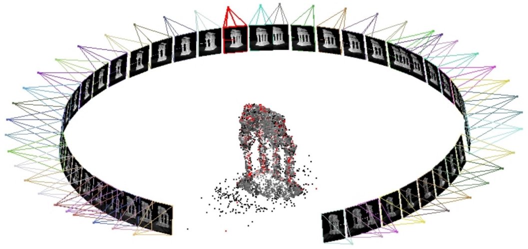
$$\mathbf{X} = \mathbf{l} \times \mathbf{l}'$$

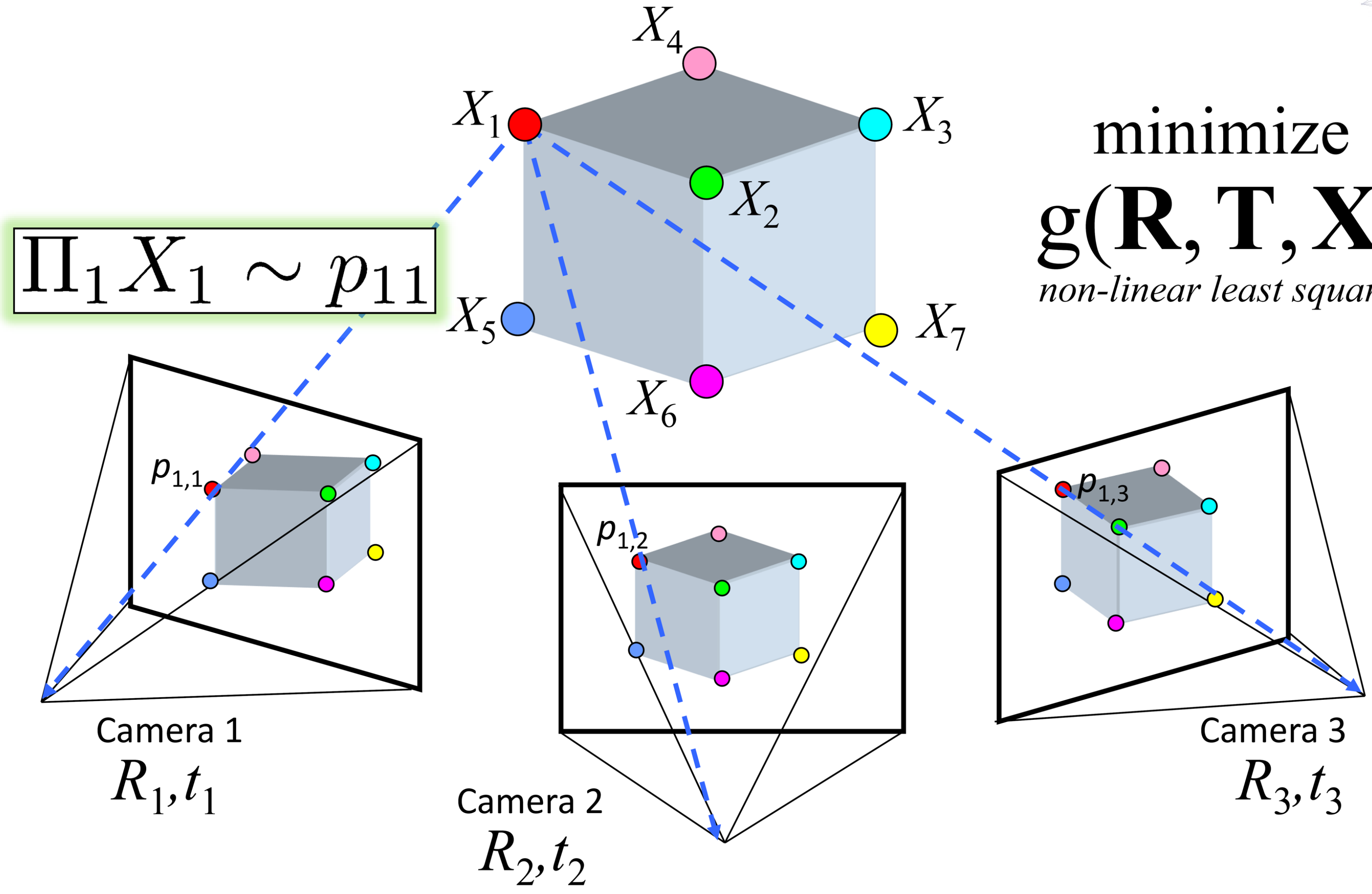
Structure from Motion

- 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





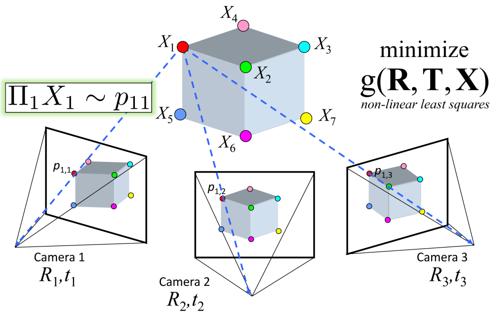
minimize

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

non-linear least squares

Structure from Motion

- Minimize sum of squared reprojection errors



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

m points, n images

A non-linear least squares problem

- E.g. Levenberg-Marquardt

The Levenberg-Marquardt Algorithm

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

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

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

Wikipedia

The Levenberg-Marquardt Algorithm

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

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

Take derivation with respect to δ and set to zero $\left(\mathbf{J}^{\mathrm{T}}\mathbf{J}\right)\boldsymbol{\delta} = \mathbf{J}^{\mathrm{T}}\left[\mathbf{y}-\mathbf{f}(\boldsymbol{\beta})\right]$

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 ?

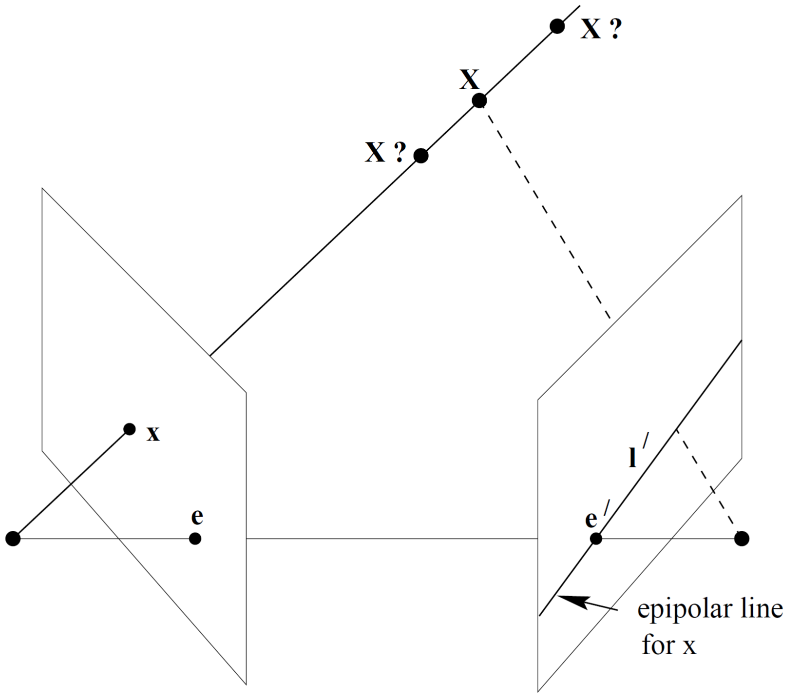
Random guess is not a good idea.

Matching Two Views

- Fundamental matrix

$\mathbf{x}'$ is on the epiploar line $\mathbf{l}' = F\mathbf{x}$

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



$$[x_i' \quad y_i' \quad 1] \begin{bmatrix} f_{11} & f_{12} & f_{13} \\ f_{21} & f_{22} & f_{23} \\ f_{31} & f_{32} & f_{33} \end{bmatrix} \begin{bmatrix} x_i \\ y_i \\ 1 \end{bmatrix} = 0$$

$$x_i x_i' f_{11} + x_i y_i' f_{21} + x_i f_{31} + y_i x_i' f_{12} + y_i y_i' f_{22} + y_i f_{32} + x_i' f_{13} + y_i' f_{23} + f_{33} = 0$$

$$\begin{bmatrix} x_1 x_1' & x_1 y_1' & x_1 & y_1 x_1' & y_1 y_1' & y_1 & x_1' & y_1' & 1 \\ \vdots & \vdots & \vdots & \vdots & \vdots & \vdots & \vdots & \vdots & \vdots \\ x_m x_m' & x_m y_m' & x_m & y_m x_m' & y_m y_m' & y_m & x_m' & y_m' & 1 \end{bmatrix} \begin{bmatrix} f_{11} \\ f_{21} \\ f_{31} \\ f_{12} \\ f_{22} \\ f_{32} \\ f_{13} \\ f_{23} \\ f_{33} \end{bmatrix} = 0$$

We need 8 points to solve this system.

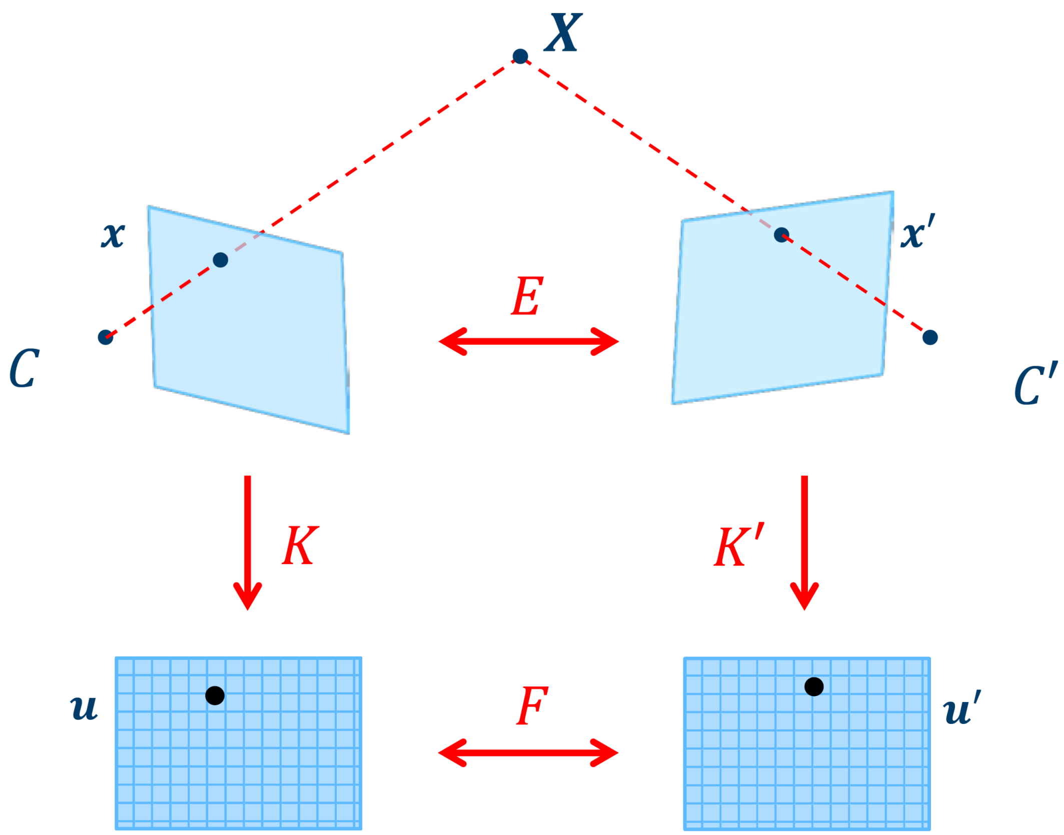
Matching Two Views

- Essential matrix E

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

$$(K'^{-1}\mathbf{x}')^T E (K^{-1}\mathbf{x}) = 0$$

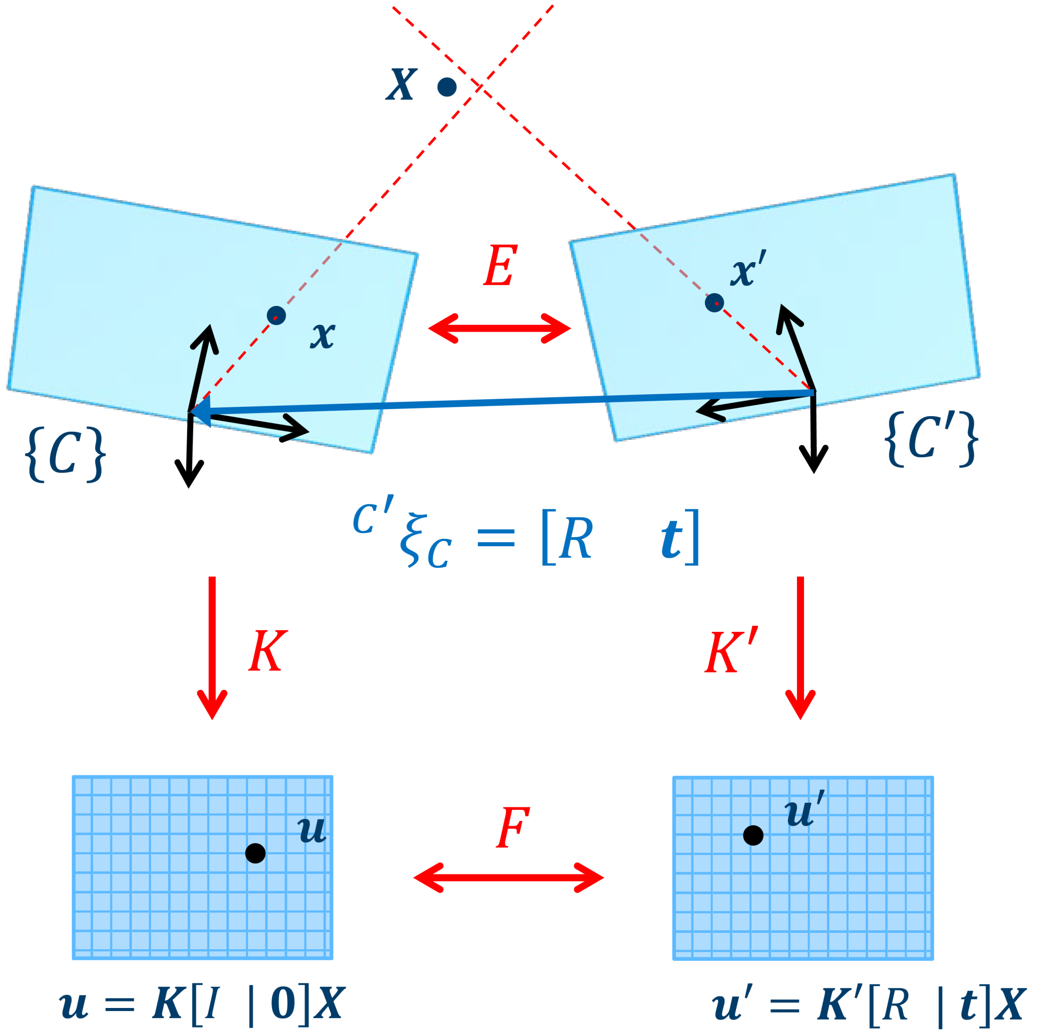
$$F = K'^{-T} E K^{-1}$$



Credit: Thomas Opsahl

Matching Two Views

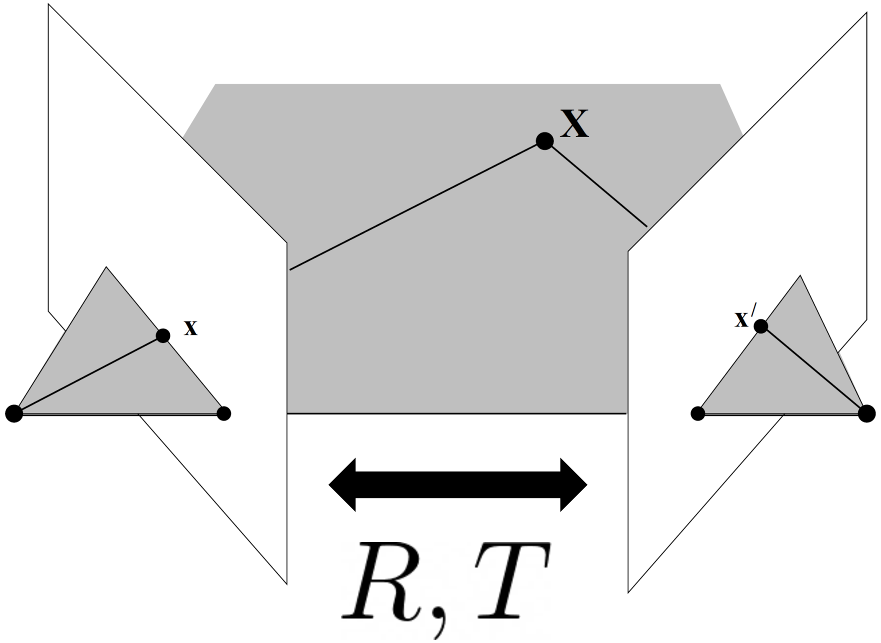
- In 1981 H. C Longuet-Higgins proved that one could recover the relative pose R and $\boldsymbol{t}$ from the essential matrix E up to the scale of $\boldsymbol{t}$



Credit: Thomas Opsahl

H. C Longuet-Higgins, *A computer algorithm for reconstructing a scene from two projections*, Nature,1981

Triangulation



Estimated from essential matrix E

Intersection of two backprojected lines

$$\mathbf{X} = \mathbf{l} \times \mathbf{l}'$$

How to get the initial estimation β_0 ?

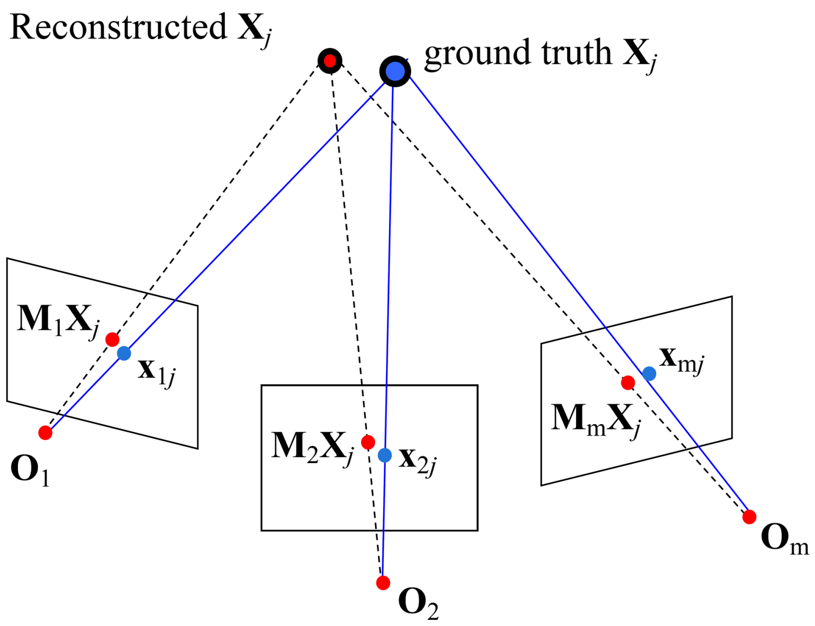
$$\beta = (\mathbf{X}, \mathbf{R}, \mathbf{T})$$

Structure from Motion

- Bundle adjustment
  - Iteratively refinement of structure (3D points) and motion (camera poses)

  - Levenberg-Marquardt algorithm

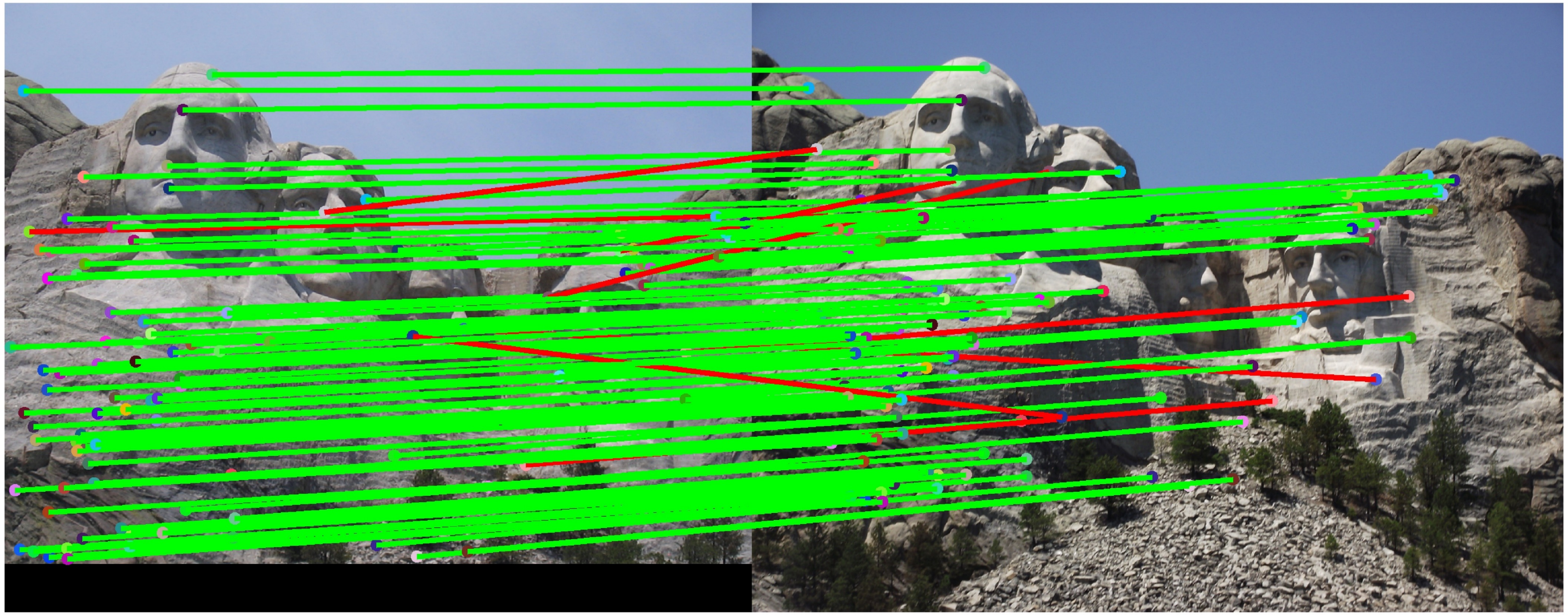
$$g(\mathbf{X},\mathbf{R},\mathbf{T}) = \sum_{i=1}^{m}\sum_{j=1}^{n} \underbrace{w_{ij}}_{\substack{\textit{indicator variable}: \\ \text{is point } i \text{ visible in image } j\ ?}} \cdot \left\| \underbrace{\mathbf{P}(\mathbf{x}_i, \mathbf{R}_j, \mathbf{t}_j)}_{\substack{\textit{predicted} \\ \text{image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



Examples: http://vision.soic.indiana.edu/projects/disco/

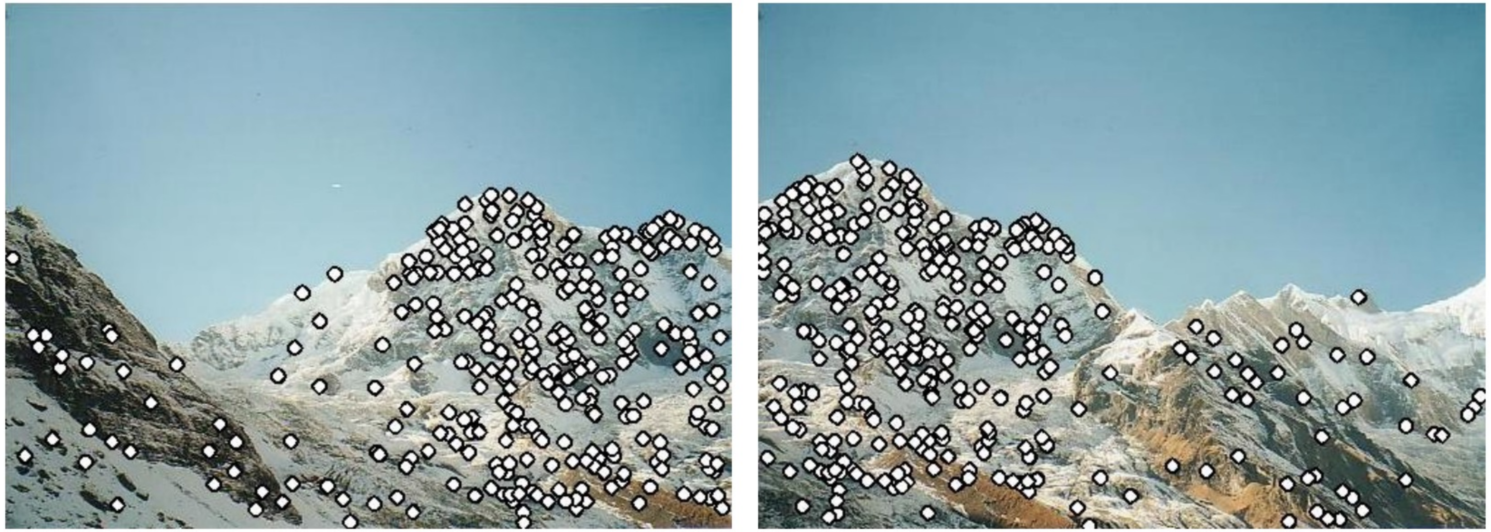
Basics

- Image feature matching

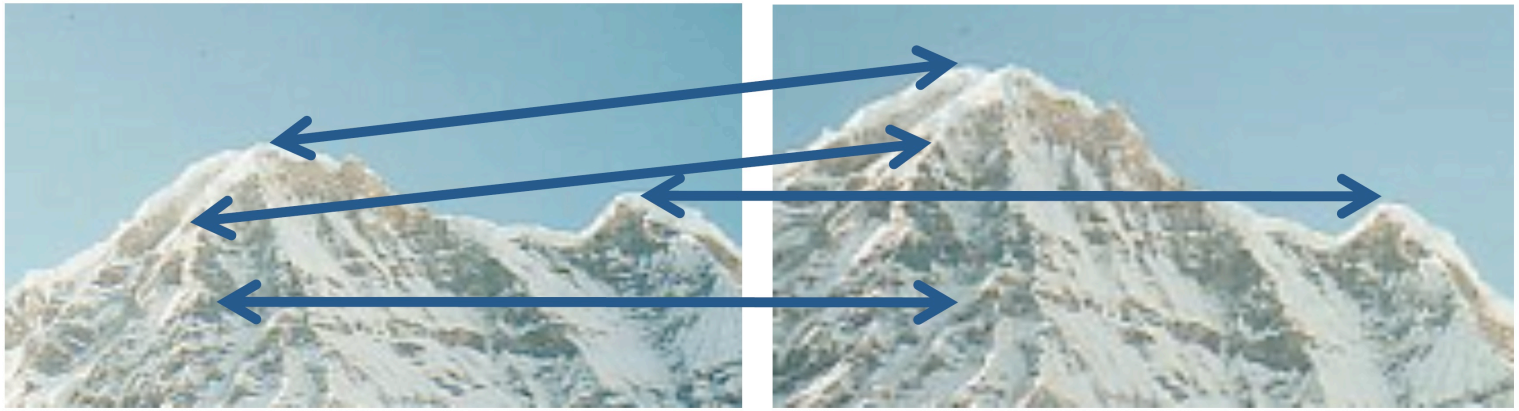


Matching with Features

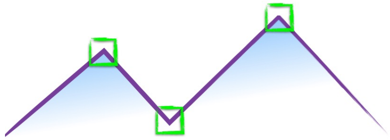
- Detecting features



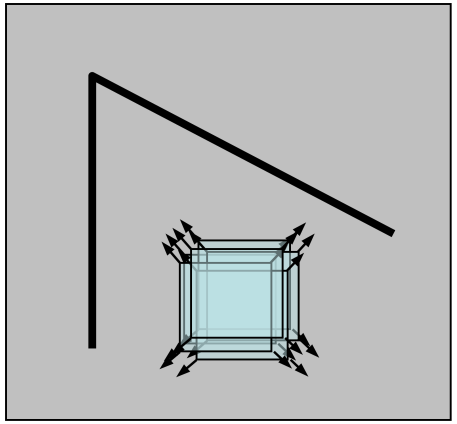
- Matching Features



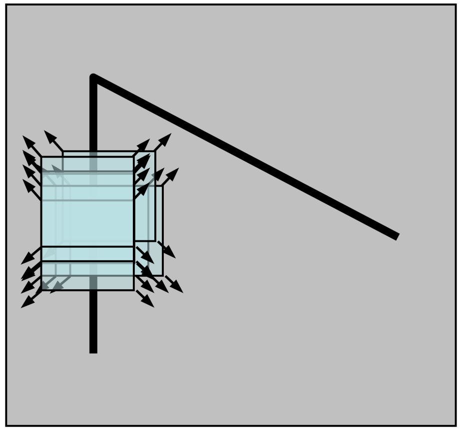
Harris Corner Detector



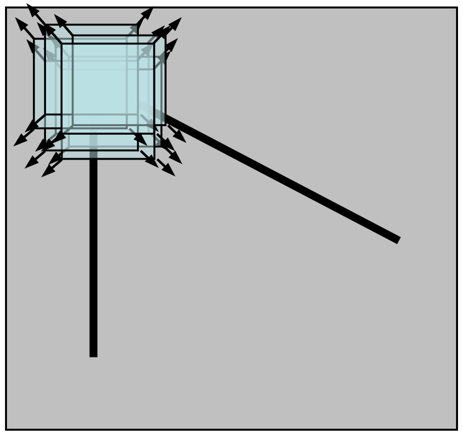
- Corners are regions with large variation in intensity in all directions



"flat" region:
no change in
all directions

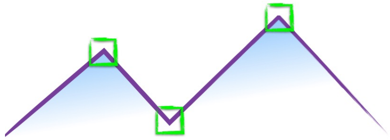


"edge":
no change
along the edge
direction



"corner":
significant
change in all
directions

Harris Corner Detector



$$f(\Delta x, \Delta y) = \sum_{(x_k, y_k) \in W} \left(I(x_k, y_k) - I(x_k + \Delta x, y_k + \Delta y)\right)^2$$

Taylor expansion

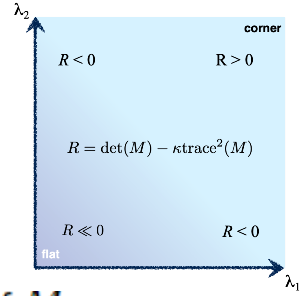
$$I(x + \Delta x, y + \Delta y) \approx I(x, y) + I_x(x, y)\Delta x + I_y(x, y)\Delta y$$

$$f(\Delta x, \Delta y) \approx \sum_{(x,y) \in W} \left(I_x(x, y)\Delta x + I_y(x, y)\Delta y\right)^2$$

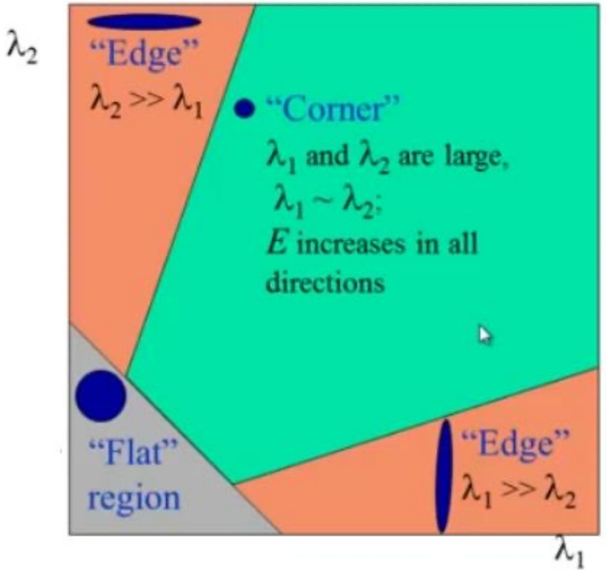
$$f(\Delta x, \Delta y) \approx \begin{pmatrix} \Delta x & \Delta y \end{pmatrix} M \begin{pmatrix} \Delta x \\ \Delta y \end{pmatrix}$$

$$M = \sum_{(x,y) \in W} \begin{bmatrix} I_x^2 & I_x I_y \\ I_x I_y & I_y^2 \end{bmatrix} = \begin{bmatrix} \sum_{(x,y) \in W} I_x^2 & \sum_{(x,y) \in W} I_x I_y \\ \sum_{(x,y) \in W} I_x I_y & \sum_{(x,y) \in W} I_y^2 \end{bmatrix}$$

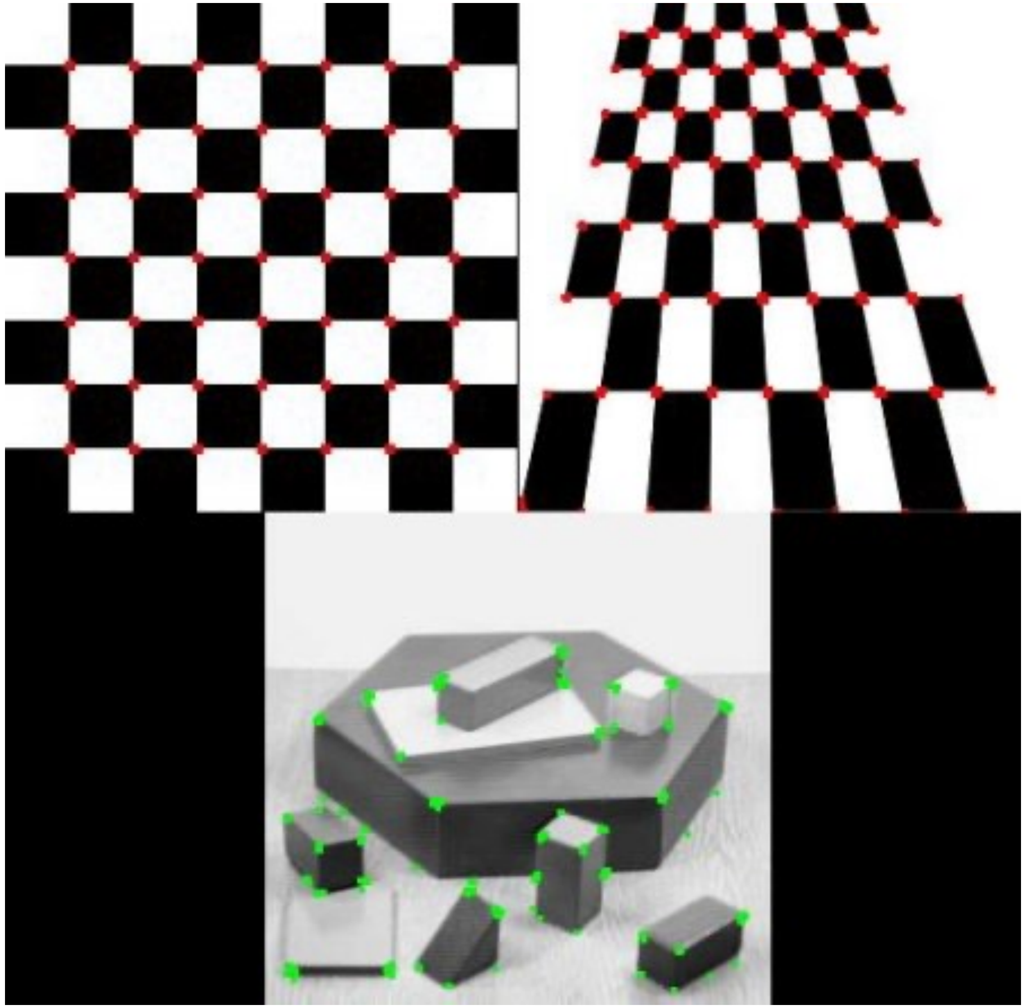
$$R = \det(M) - k(\mathrm{trace}(M))^2$$



- $\det(M) = \lambda_1 \lambda_2$
- $\mathrm{trace}(M) = \lambda_1 + \lambda_2$
- λ_1 and λ_2 are the eigenvalues of M



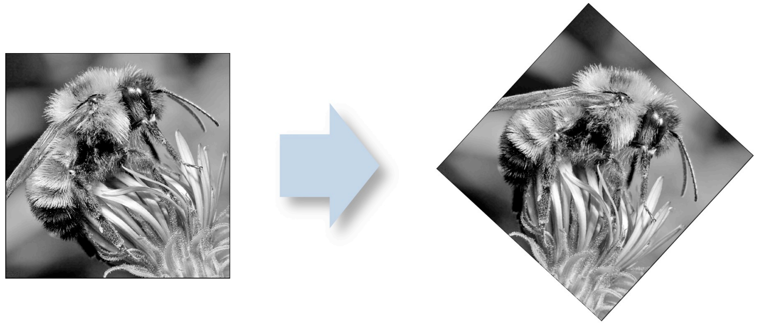
Harris Corner Detector



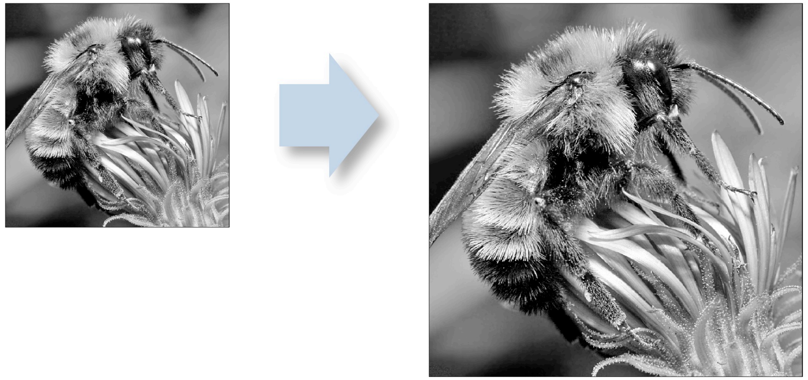
https://docs.opencv.org/master/dc/d0d/tutorial_py_features_harris.html

Invariance

- Can the same feature point be detected after some transformation?
  - Translation invariance
  - 2D rotation invariance

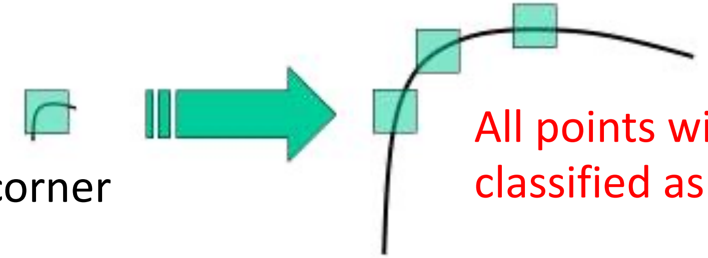


  - Scale invariance



Are Harris corners scale invariant?

No

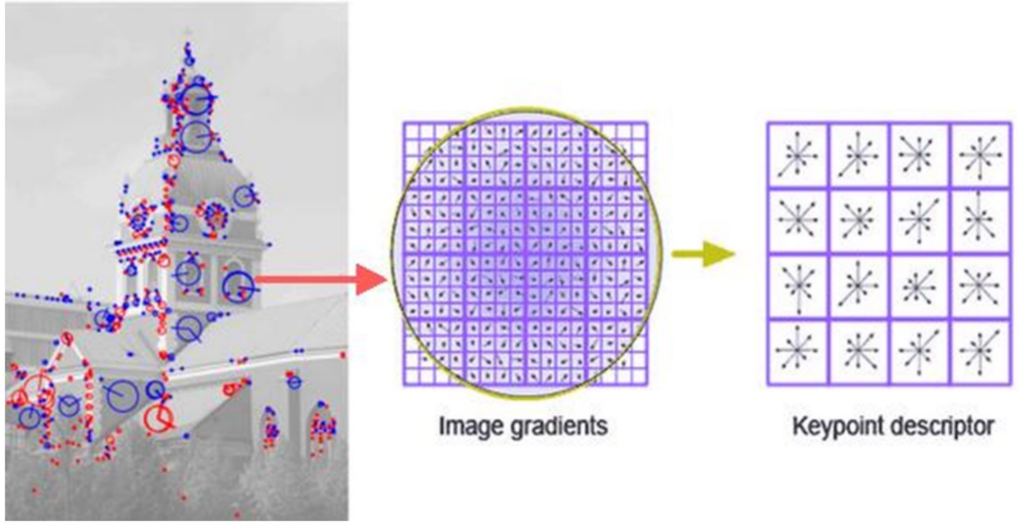


corner

All points will be classified as edges!

SIFT: Scale-invariant feature transform

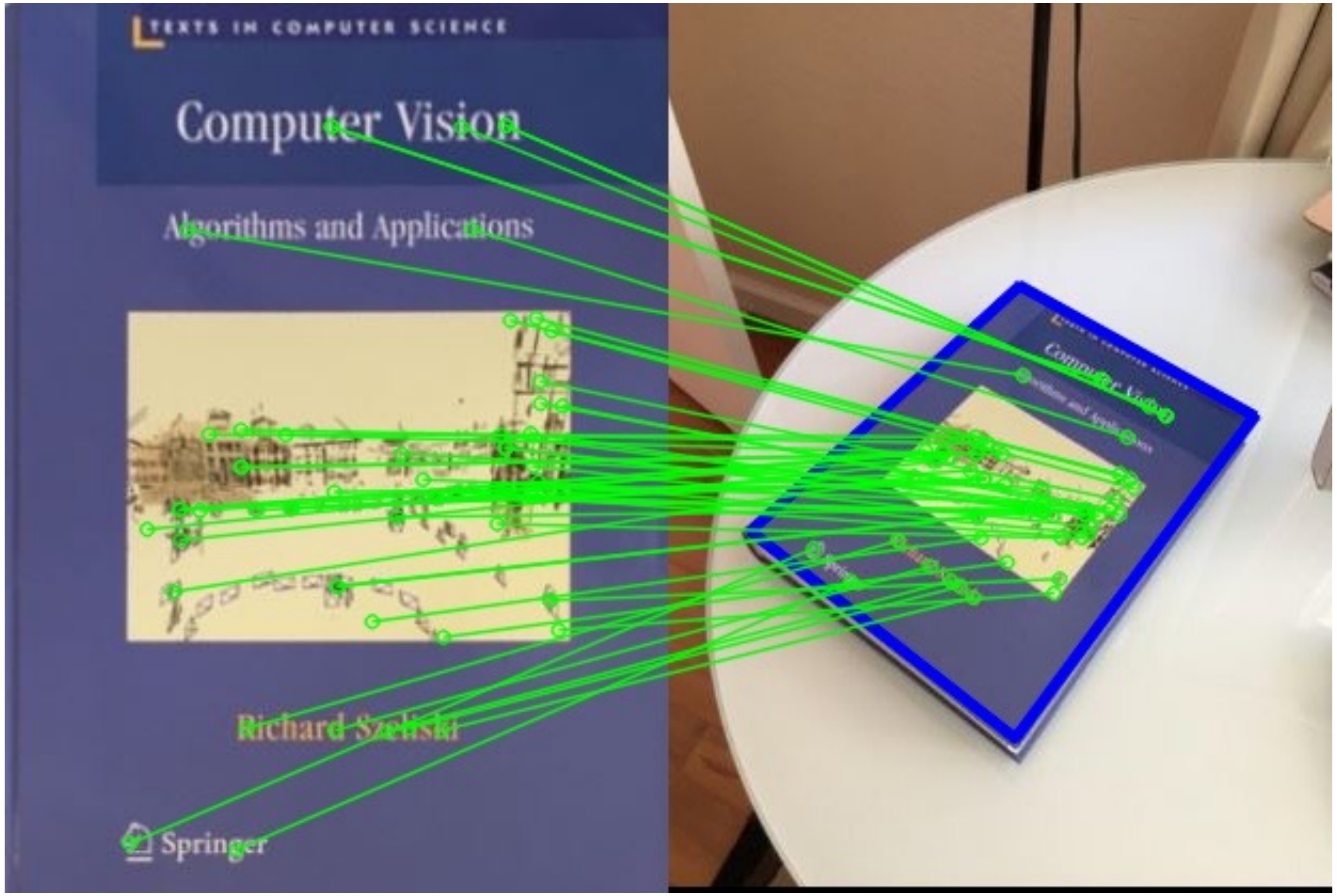
- Invariant to scaling, rotation and translation
- Partially invariant to illumination changes or affine or 3D projection
- Transforms an image into a large collection of local feature vectors (SIFT local descriptors)



The circles are scaled and rotated to reflect the scale and orientation of the features.

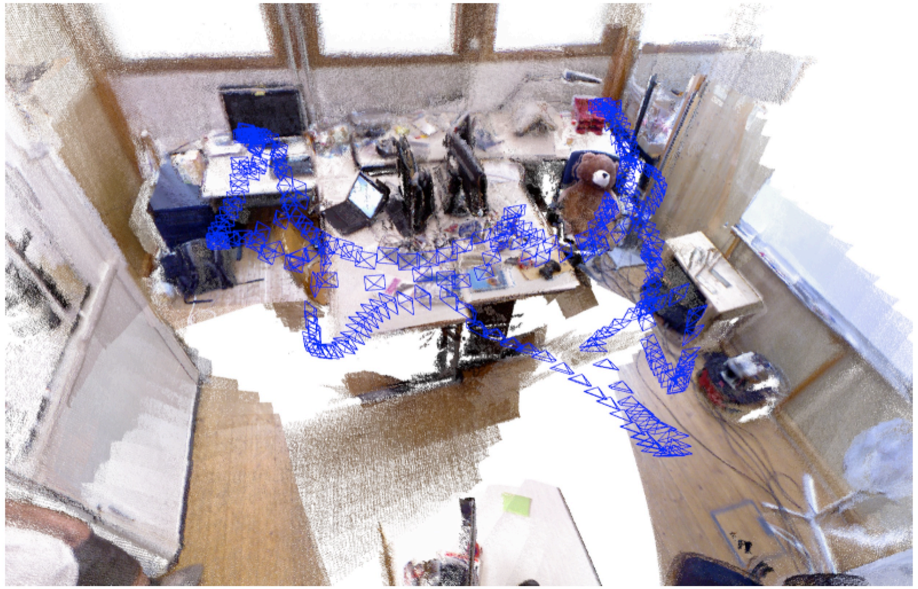
David Lowe, Distinctive image features from scale-invariant keypoints, IJCV, 2004. (SIFT has been cited by more than 90,000 times in total!)

SIFT Matching Example



Simultaneous Localization and Mapping (SLAM)

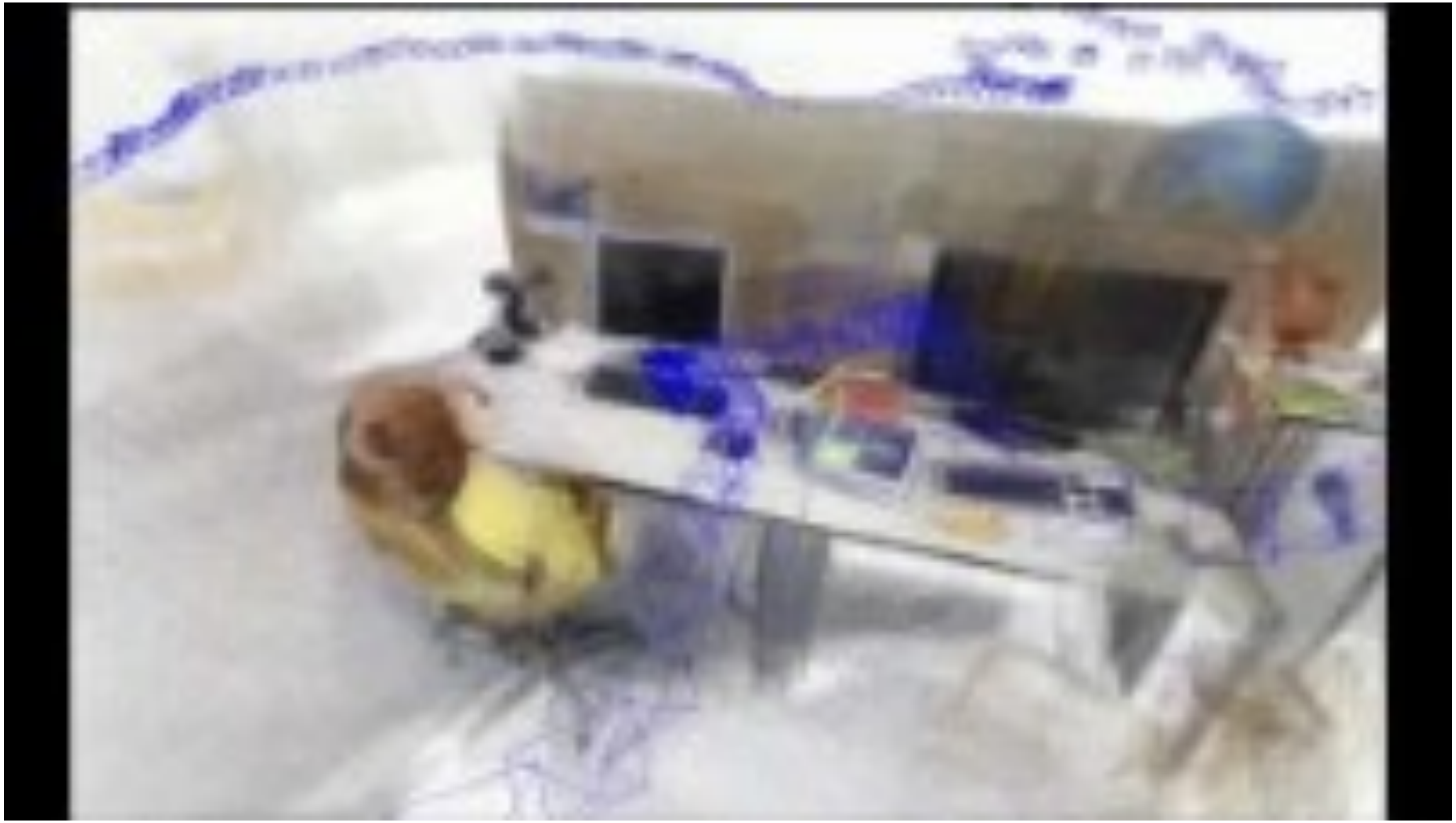
- Localization: camera pose tracking
- Mapping: building a 2D or 3D representation of the environment
- The goal here is the same as structure from motion, usually with video input



ORB-SLAM2

- Point cloud and camera poses

ORB-SLAM



https://webdiis.unizar.es/~raulmur/orbslam/

3D Scanning

- Using laser to create "point clouds"

(a) (b)

Figure 9.26: (a) The Afinia ES360 scanner, which produces a 3D model of an object while it spins on a turntable. (b) The Focus3D X 330 Laser Scanner, from FARO Technologies, is an outward-facing scanner for building accurate 3D models of large environments; it includes a GPS receiver to help fuse individual scans into a coherent map.

3D Scanning

https://matterport.com/

Further Reading

- Section 9.5, Virtual Reality, Steven LaValle
- SIFT: Distinctive Image Features from Scale-Invariant Keypoints, David Lowe, IJCV'04
- ORB-SLAM: ORB-SLAM: a Versatile and Accurate Monocular SLAM System, Mur-Artal et al., T-RO'15