CS 598 3D Vision:
Multi-View Geometry

Shenlong Wang
UIUC

ILLINOIS

Some materials borrowed from Matthew O'Toole, Kris Kitani, Jianxiong Xiao, Derek Hoeim, Sanja Fidler

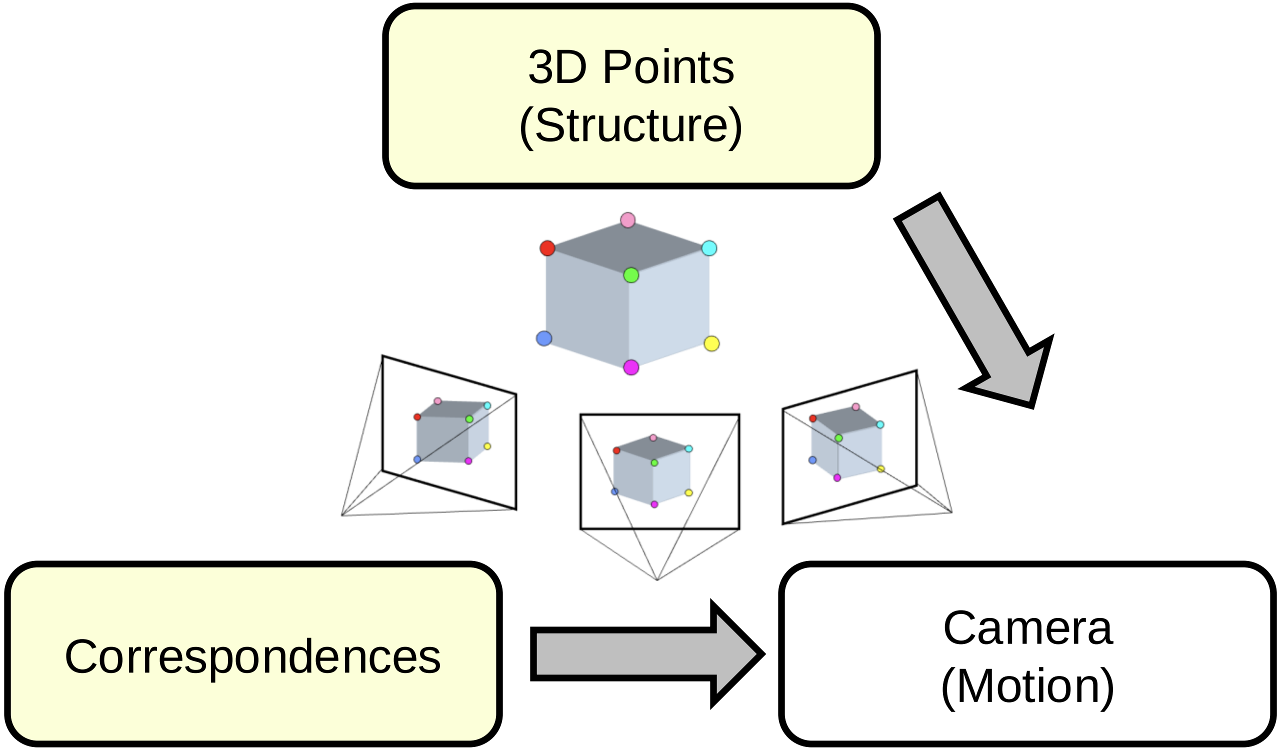
Logistics

- **Quiz 1** – If we didn't reach out, it's satisfactory!
- **Quiz 2** – Will be out tonight (due next Tuesday).
- **Group assignment** is out!
- **Survey** due date has been extended (Sept 26 → Oct 3). Do meet earlier to conduct a literature review and select 25+ papers, then organize them into groups and assign jobs within the group.
- **Role-playing group**: 1) discuss your tackling plans with us during Thursday office hours, the week before your presentation, or arrange a quick ad-hoc meeting. 2) share your presentation for feedback three days before your group presentation.

Today’s Agenda

- Camera Calibration
- Structure from Motion
- Other Cameras

Big picture: 3 key components in 3D



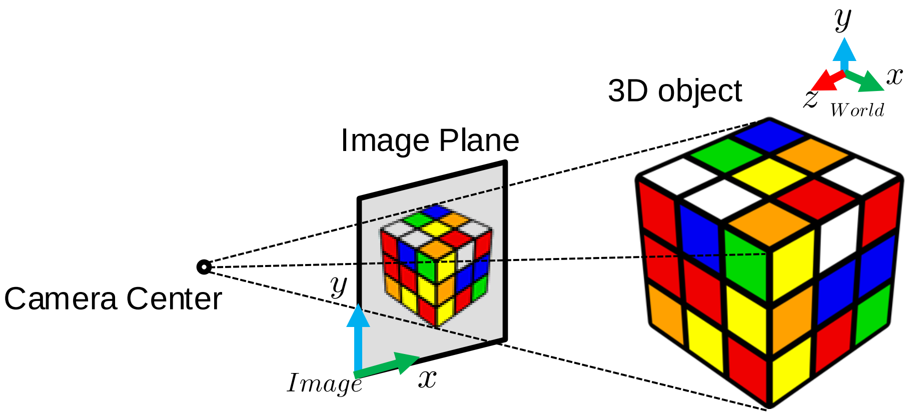
Angjoo Kanazawa

Camera Calibration

How do I know K?

$$\mathbf{P} = \mathbf{K}[\mathbf{R}|\mathbf{t}]$$

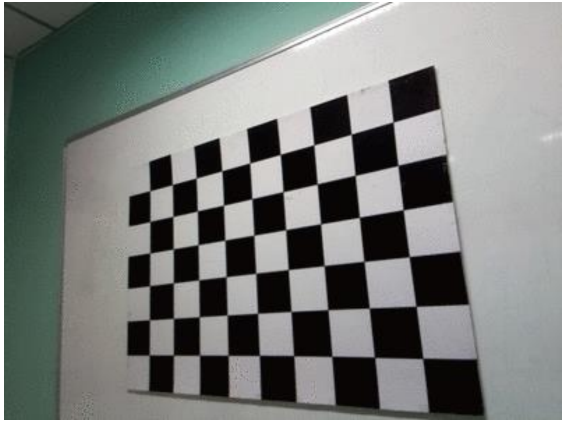
$$\mathbf{P} = \underbrace{\begin{bmatrix} f & 0 & p_x \\ 0 & f & p_y \\ 0 & 0 & 1 \end{bmatrix}}_{\text{intrinsic parameters}} \underbrace{\left[\begin{array}{ccc:c} r_1 & r_2 & r_3 & t_1 \\ r_4 & r_5 & r_6 & t_2 \\ r_7 & r_8 & r_9 & t_3 \end{array}\right]}_{\text{extrinsic parameters}}$$



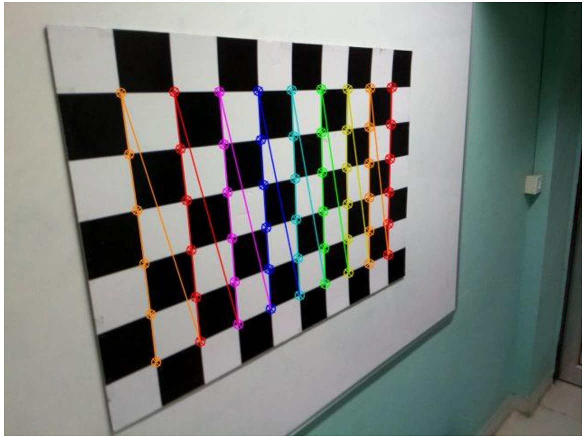
Camera Calibration

- **Inputs** : A collection of images with points whose 2D image coordinates and 3D world coordinates are known.
- **Outputs**: The 3×3 camera intrinsic matrix, the rotation and translation of each image.

Capture multiple images of the checkerboard from different viewpoints



Find checkerboard corners



Finding camera parameters by minimizing 3D-2D reprojection err

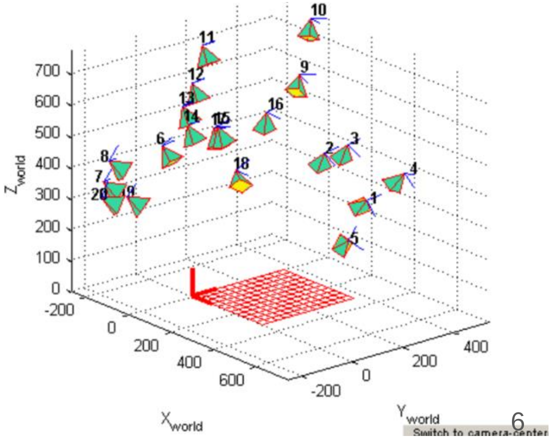


Image credit: OpenCV

Camera Calibration

- Minimizing the reprojection error

$$\min_{\{\mathbf{R}_k,\mathbf{t}_k\},\mathbf{K}} \sum_i \sum_k \|\mathbf{x}_{ik} - \pi_{\mathbf{K}}([\mathbf{R}_k|\mathbf{t}_k]\mathbf{X}_i)\|_2^2$$

Extrinsic

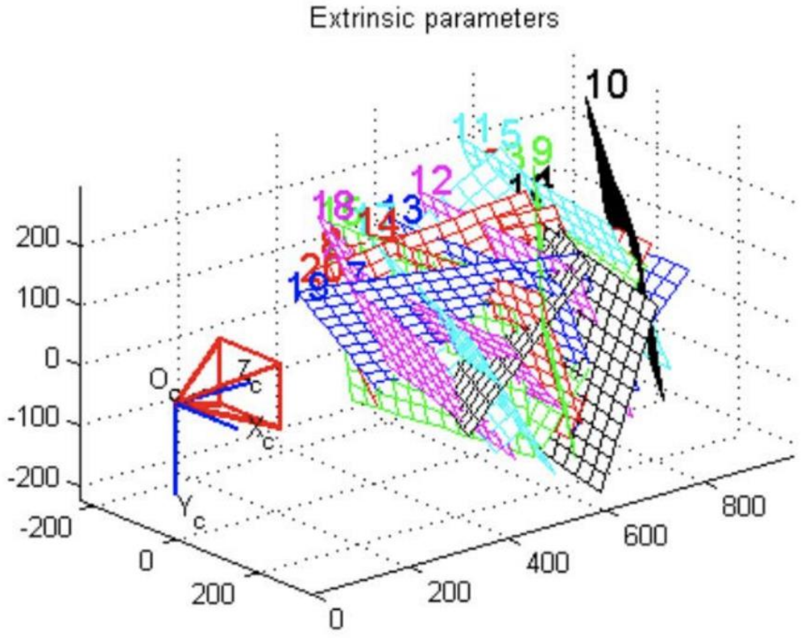
Intrinsic

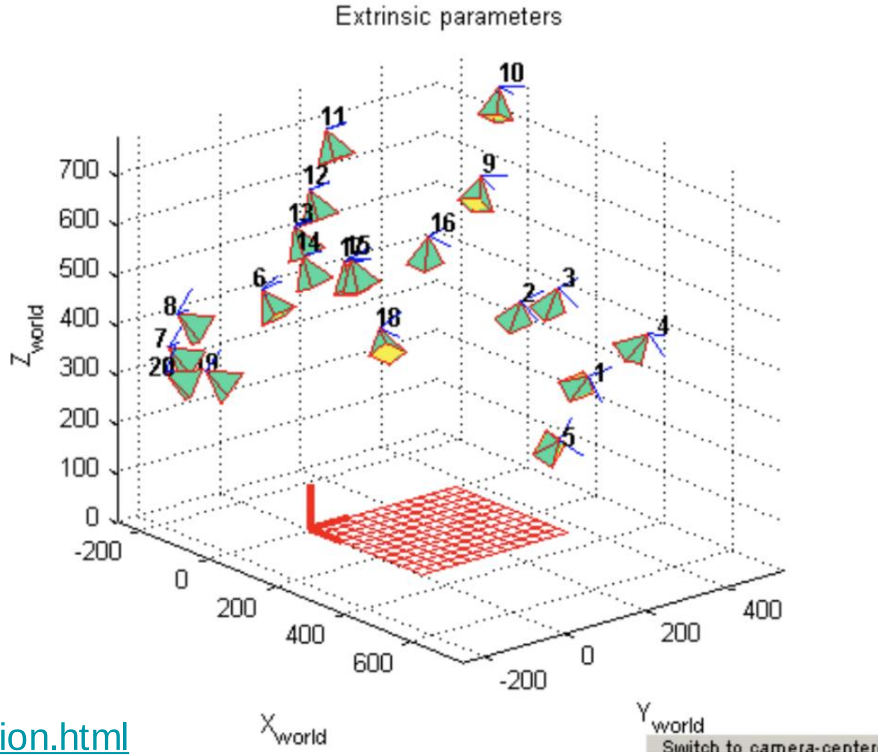
Detection Corner Points

Perspective Projection

Known 3D Location

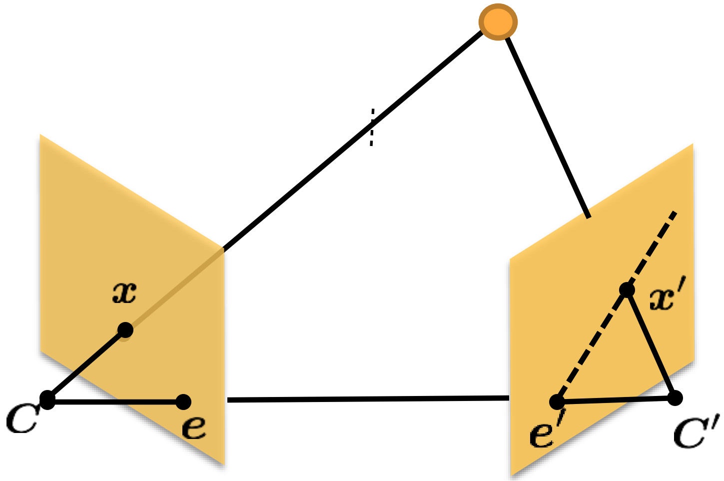
Camera Calibration



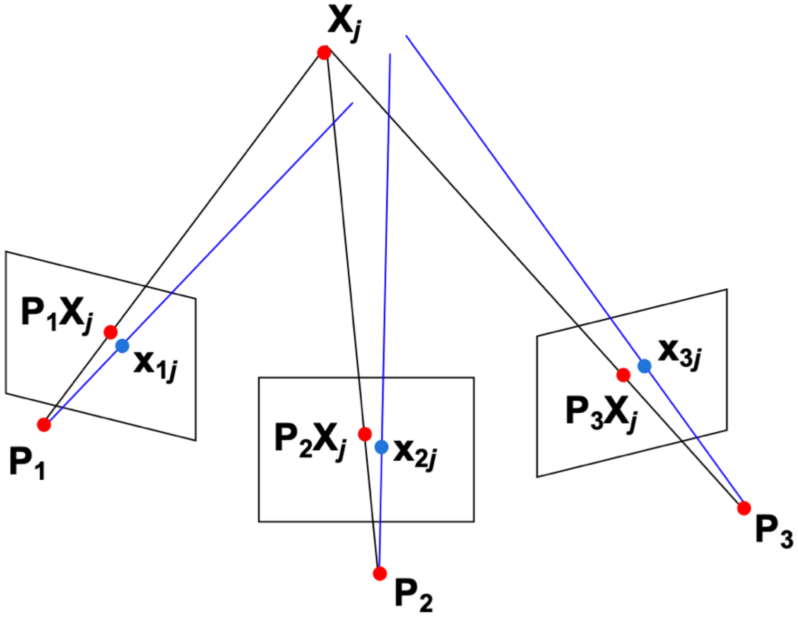


https://www.mathworks.com/help/vision/camera-calibration.html
https://github.com/ethz-asl/kalibr
https://docs.opencv.org/4.x/dc/dbb/tutorial_py_calibration.html

Structure-from-Motion



Each pair of 2D-2D correspondence establish triagulaler relationships



Structure-from-Motion

- Structure = 3D Point Cloud of the Scene
- Motion = Camera Location and Orientation
- SFM = Get the Point Cloud from Moving Cameras
- Structure and Motion: Joint Problems to Solve

Image credit: Jianxiong Xiao

Structure-from-Motion

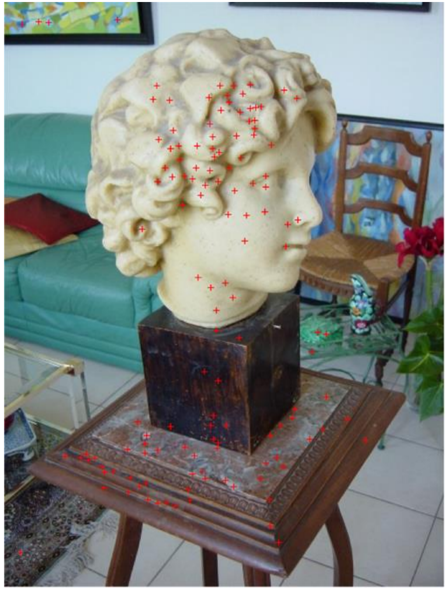
- Establish 2D-2D correspondences across images
- Jointly refine camera pose and 3D points in an optimization framework

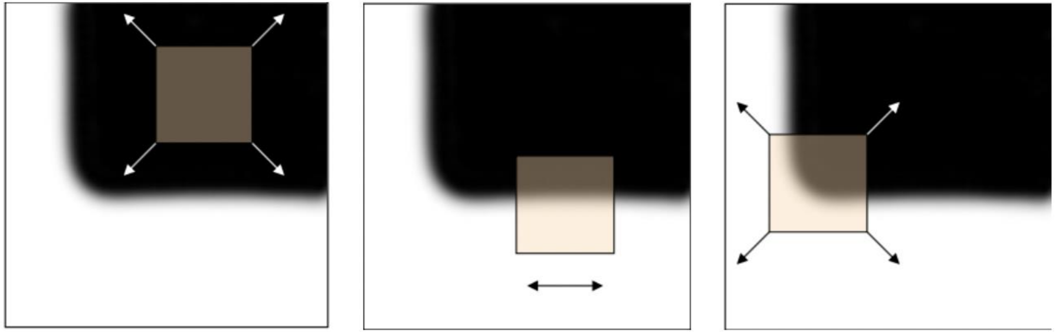
Two View Reconstruction

$\Rightarrow$ **Match** $\Rightarrow$ $\mathbf{F}$ $\Rightarrow$ $\mathbf{E}$ $\Rightarrow$ $\mathbf{R}, \mathbf{t}$ $\Rightarrow$ $\mathbf{X}$ $\Rightarrow$ **Bundle Adjustment**

Keypoints Detection

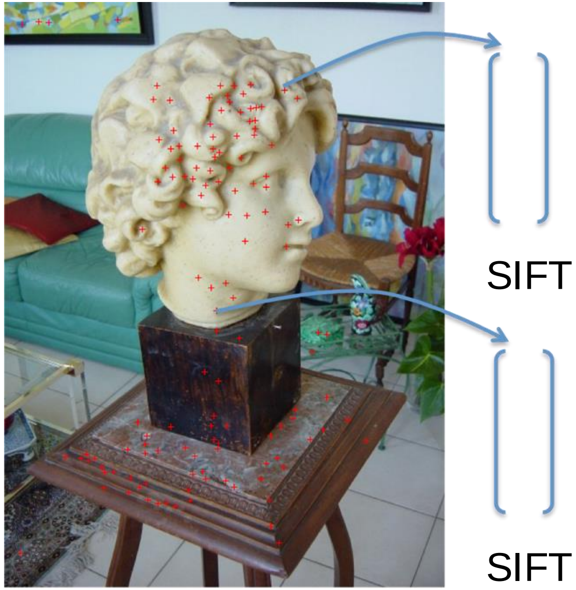
- Step 1: Detect distinctive keypoints that are suitable for matching

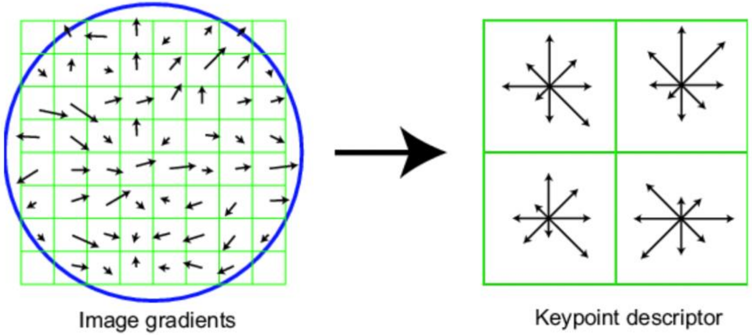




Descriptor for each point

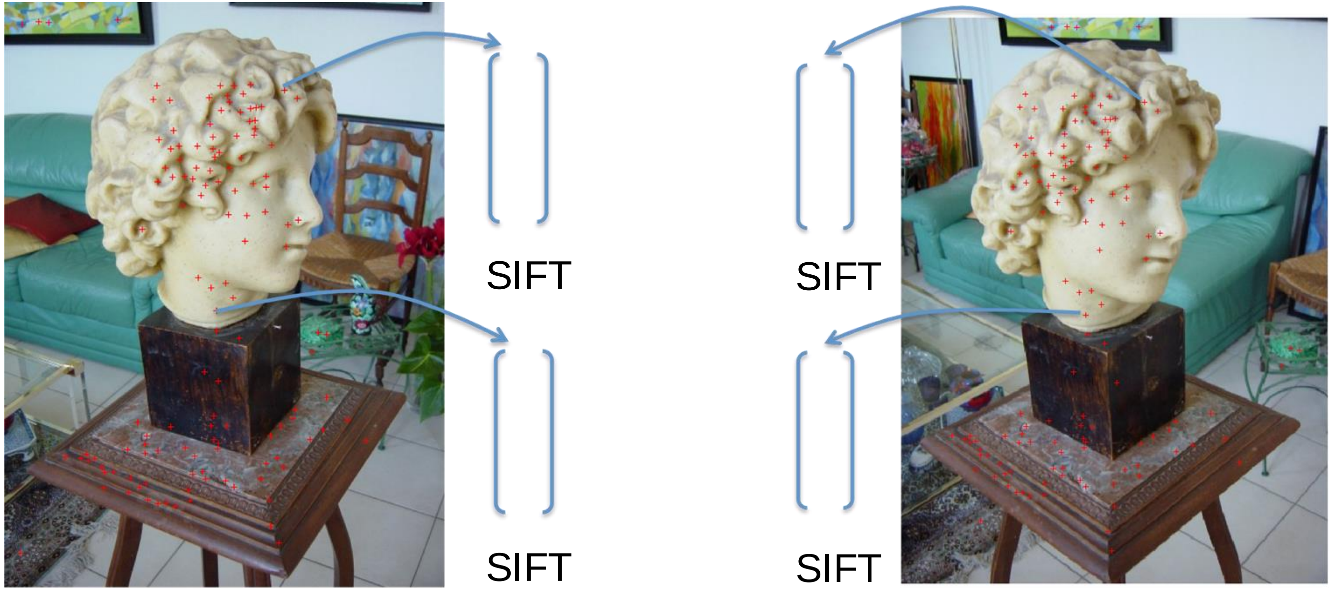
- Step 2: Compute visual descriptors (SIFT features)





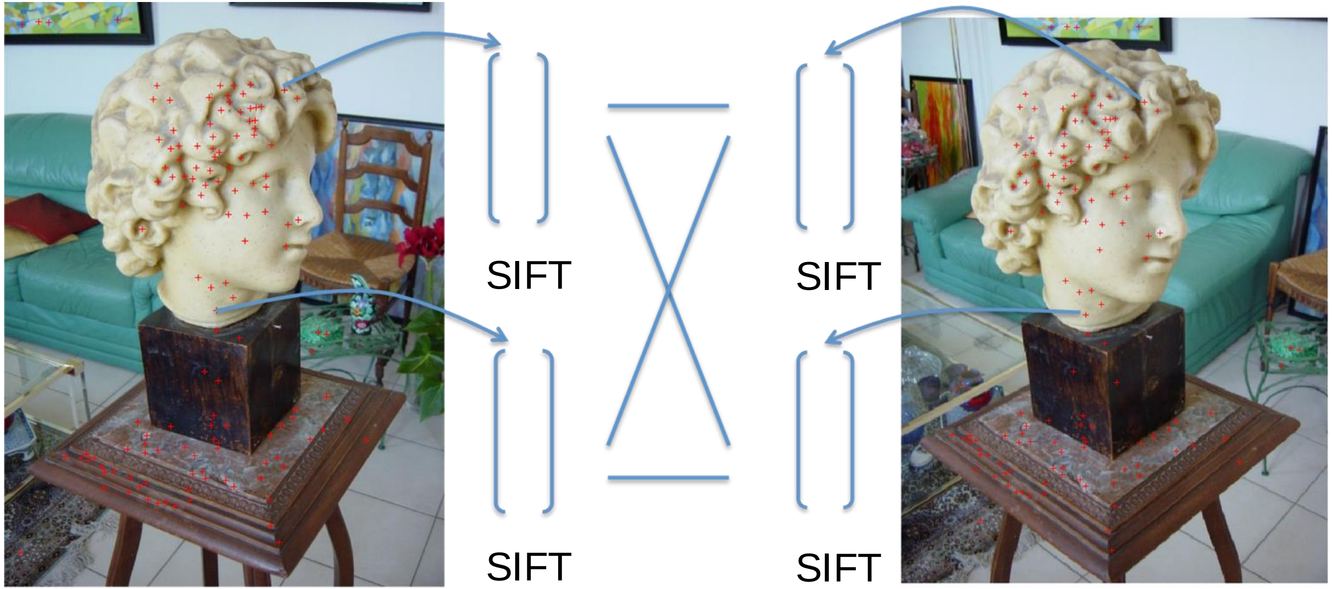
Descriptor for each point

- Step 3: Measure pairwise distance / similarity between features



Match Points

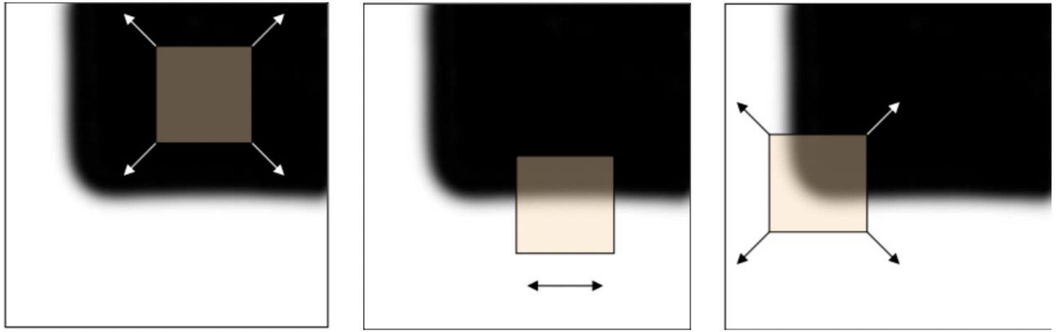
- Step 3: Measure pairwise distance / similarity between features

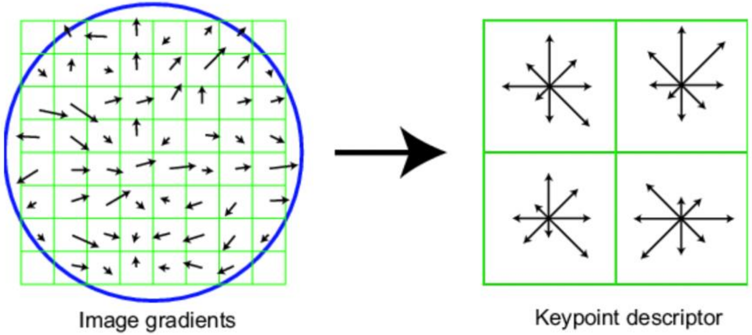


Match Points

SIFT (scale-invariant feature transform)

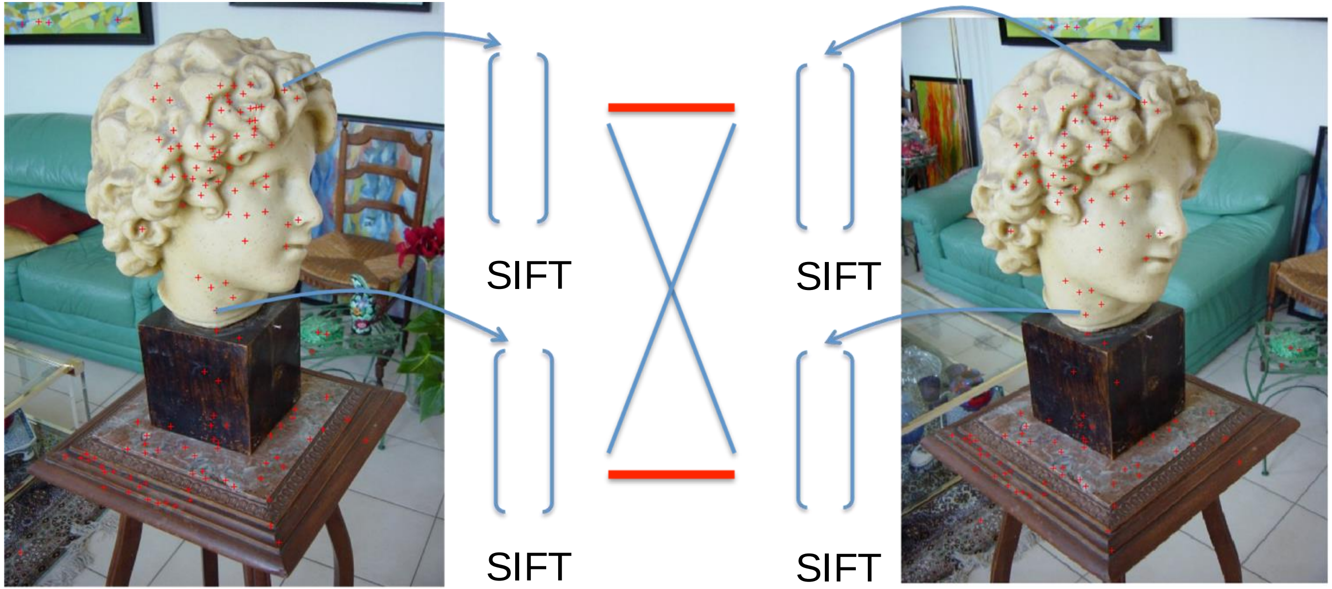
- Step 1: Detect distinctive keypoints that are suitable for matching
- Step 2: Compute oriented histogram gradient features
- Step 3: Measure distance between each pair





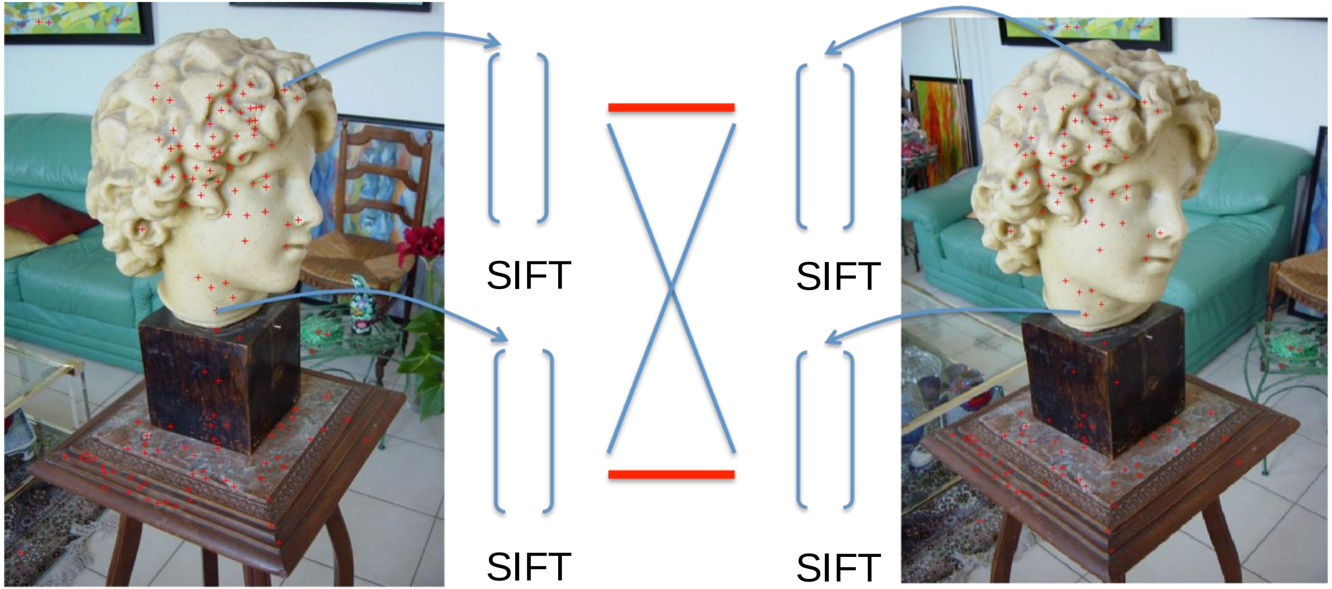
Match Points

- **How many pair-wise matching I need to conduct?**



Match Points

- **What if there are bad matches?**



Match Points in Practice

How can we make SIFT matching faster than exhaustive search?

- Approximate nearest neighbor search
- Hashing, KD-tree, etc.

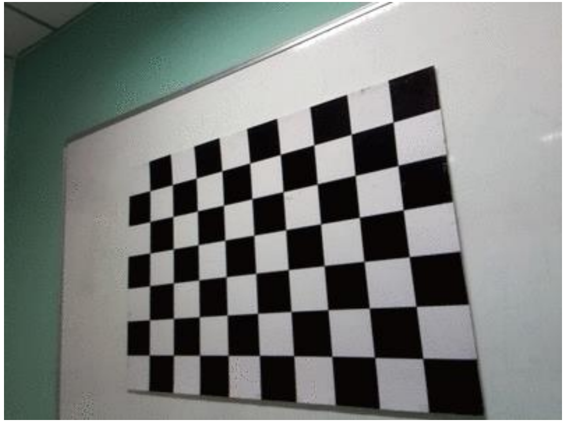
How can we ensure a pair of match is good?

- Ratio test: my nearest neighbor should be much better than other candidates
- Consistency-check: (1) keypoint A’s nearest neighbor in image 2 is keypoint B; (2) keypoint B’s nearest neighbor in image 1 is also keypoint A.

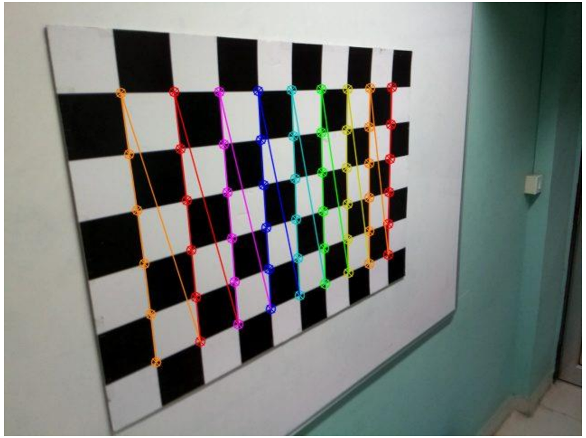
Camera Calibration

- **Inputs** : A collection of images with points whose 2D image coordinates and 3D world coordinates are known.
- **Outputs**: The 3×3 camera intrinsic matrix, the rotation and translation of each image.

Capture multiple images of the checkerboard from different viewpoints



Find checkerboard corners



Finding camera parameters by minimizing 3D-2D reprojection err

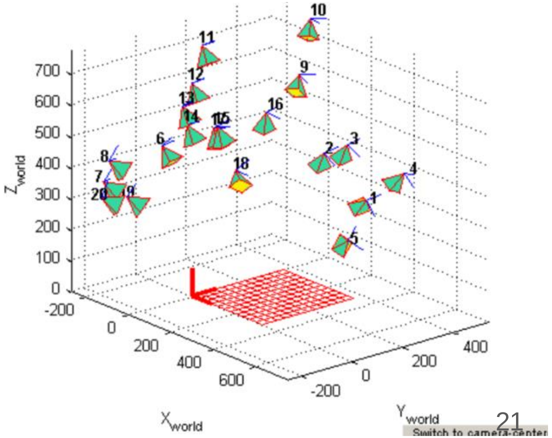


Image credit: OpenCV

Camera Calibration

- Minimizing the reprojection error

$$\min_{\{\mathbf{R}_k,\mathbf{t}_k\},\mathbf{K}} \sum_i \sum_k \|\mathbf{x}_{ik} - \pi_{\mathbf{K}}([\mathbf{R}_k|\mathbf{t}_k]\mathbf{X}_i)\|_2^2$$

Extrinsic   Intrinsic   Detection Corner Points   Perspective Projection   Known 3D Location

Two View Reconstruction

Match $\Rightarrow$ $\mathbf{F}$ $\Rightarrow$ $\mathbf{E}$ $\Rightarrow$ $\mathbf{R}, \mathbf{t}$ $\Rightarrow$ $\mathbf{X}$ $\Rightarrow$ Bundle Adjustment

Fundamental Matrix

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

Eight-Point Algorithm

- Given a correspondence $\mathbf{x} \leftrightarrow \mathbf{x}'$
- Assume

$$\mathbf{x}' = \begin{bmatrix} x' \\ y' \\ 1 \end{bmatrix} \quad \mathbf{x} = \begin{bmatrix} x \\ y \\ 1 \end{bmatrix} \quad \mathbf{F} = \begin{bmatrix} f_{11} & f_{12} & f_{13} \\ f_{21} & f_{22} & f_{23} \\ f_{31} & f_{32} & f_{33} \end{bmatrix}$$

$$\mathbf{f} = \begin{bmatrix} f_{11} & f_{12} & f_{13} & f_{21} & f_{22} & f_{23} & f_{31} & f_{32} & f_{33} \end{bmatrix}^{\mathsf{T}}$$

- We can get $\mathbf{x}'^{\mathsf{T}} \mathbf{F} \mathbf{x} = 0$

$$\Downarrow$$

$$\begin{bmatrix} x'x & x'y & x' & y'x & y'y & y' & x & y & 1 \end{bmatrix} \mathbf{f} = 0$$

Eight-Point Algorithm

- Given 8 correspondences

$$\begin{bmatrix} x'_1x_1 & x'_1y_1 & x'_1 & y'_1x_1 & y'_1y_1 & y'_1 & x_1 & y_1 & 1 \\ x'_2x_2 & x'_2y_2 & x'_2 & y'_2x_2 & y'_2y_2 & y'_2 & x_2 & y_2 & 1 \\ x'_3x_3 & x'_3y_3 & x'_3 & y'_3x_3 & y'_3y_3 & y'_3 & x_3 & y_3 & 1 \\ x'_4x_4 & x'_4y_4 & x'_4 & y'_4x_4 & y'_4y_4 & y'_4 & x_4 & y_4 & 1 \\ x'_5x_5 & x'_5y_5 & x'_5 & y'_5x_5 & y'_5y_5 & y'_5 & x_5 & y_5 & 1 \\ x'_6x_6 & x'_6y_6 & x'_6 & y'_6x_6 & y'_6y_6 & y'_6 & x_6 & y_6 & 1 \\ x'_7x_7 & x'_7y_7 & x'_7 & y'_7x_7 & y'_7y_7 & y'_7 & x_7 & y_7 & 1 \\ x'_8x_8 & x'_8y_8 & x'_8 & y'_8x_8 & y'_8y_8 & y'_8 & x_8 & y_8 & 1 \end{bmatrix} \mathbf{f} = 0 \quad \Rightarrow \quad \mathbf{Af} = 0$$

- Nontrivial solution
  - **f is in null space of A**

SVD!

Eight-Point Algorithm

- Rank constraint

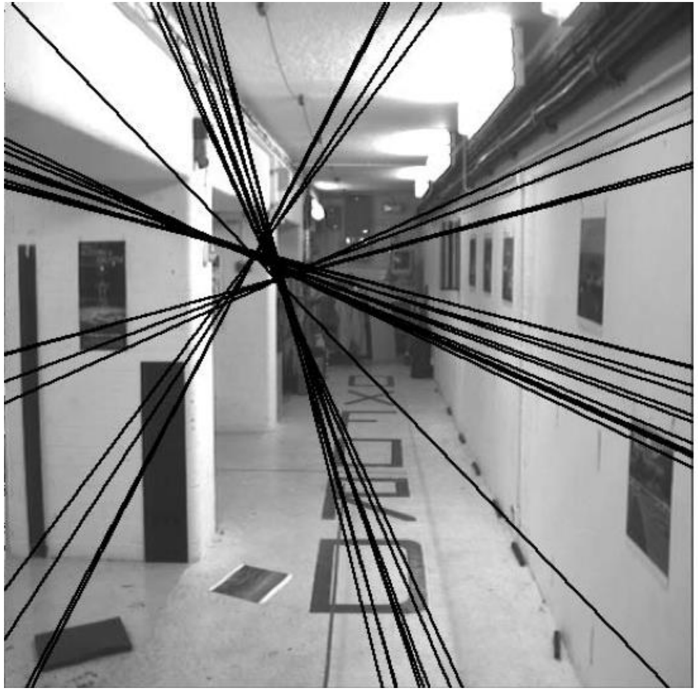
$$\mathbf{F} \rightarrow \mathbf{F}' \qquad \det \mathbf{F}' = 0$$

- Minimize Frobenius norm

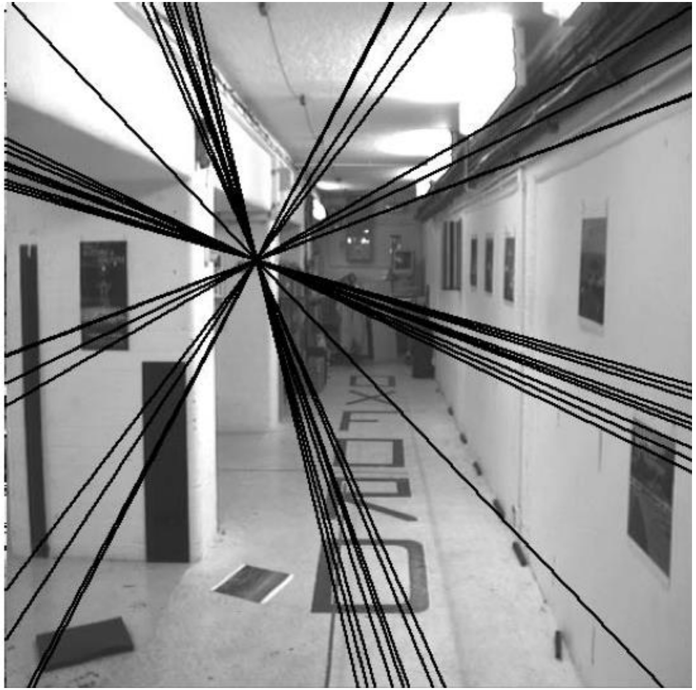
$$\min_{\mathbf{F}'} \|\mathbf{F} - \mathbf{F}'\|_{\mathsf{F}} \quad \text{subject to} \quad \det \mathbf{F}' = 0$$

$$\mathbf{F} = \mathbf{U}\,\mathrm{diag}(\sigma_1, \sigma_2, \sigma_3)\mathbf{V}^{\mathsf{T}} \Rightarrow \mathbf{F}' = \mathbf{U}\,\mathrm{diag}(\sigma_1, \sigma_2, 0)\mathbf{V}^{\mathsf{T}}$$

Rank Constraint



Before

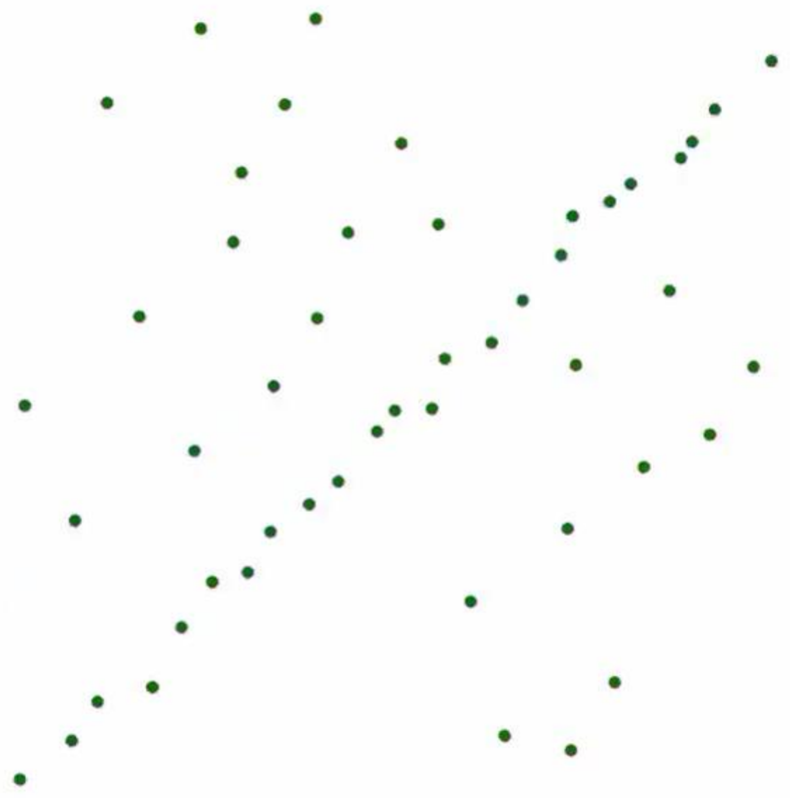


After

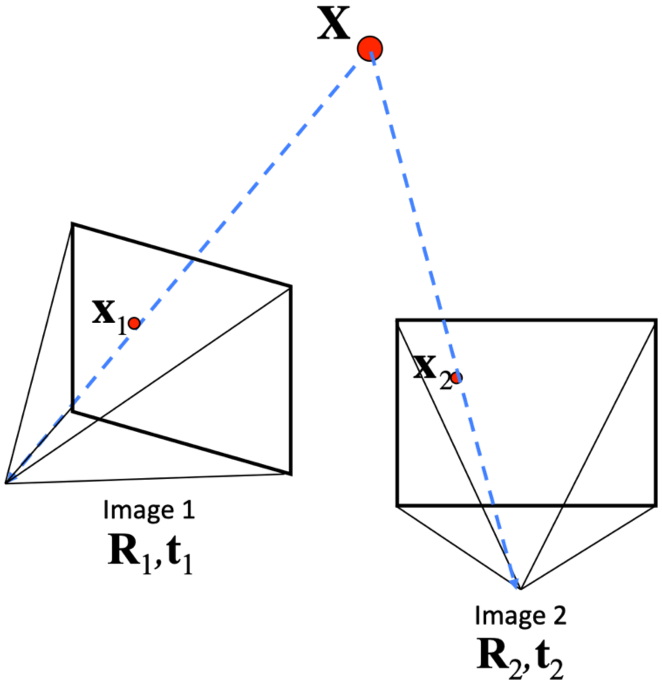
RANSAC Estimation

- For many times
    - Pick 8 points
    - Compute a solution for $\mathbf{F}$ using these 8 points
    - Count number of inliers that with geometric error close to 0

- Pick the one with the largest number of inliers
- Only the inliers are kept as correspondences



Essential Matrix



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

$$\Downarrow$$

$$\mathbf{E} = \mathbf{K}'^{\mathsf{T}}\mathbf{F}\mathbf{K}$$

$$\Downarrow$$

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

$$\Downarrow$$

$$\mathbf{R}, \mathbf{t}$$

Essential Matrix Decomposition

- Essential matrix **E** to **R** and **t**

Result 9.19. For a given essential matrix

$$\mathbf{E} = \mathbf{U}\mathrm{diag}(1,1,0)\mathbf{V}^T,$$

and the first camera matrix $\mathbf{P}_1 = [\mathbf{I}|\mathbf{0}]$, there are four possible choices for the second camera matrix $\mathbf{P}_2$:

$$\mathbf{P}_2 = [\mathbf{U}\mathbf{W}\mathbf{V}^T | +\mathbf{u}_3]$$

$$\mathbf{P}_2 = [\mathbf{U}\mathbf{W}\mathbf{V}^T | -\mathbf{u}_3]$$

$$\mathbf{P}_2 = [\mathbf{U}\mathbf{W}^T\mathbf{V}^T | +\mathbf{u}_3]$$

$$\mathbf{P}_2 = [\mathbf{U}\mathbf{W}^T\mathbf{V}^T | -\mathbf{u}_3]$$

$$\mathbf{W} = \begin{bmatrix} 0 & -1 & 0 \\ 1 & 0 & 0 \\ 0 & 0 & 1 \end{bmatrix}$$

Try to verify by yourself

Extending to Multiple Views

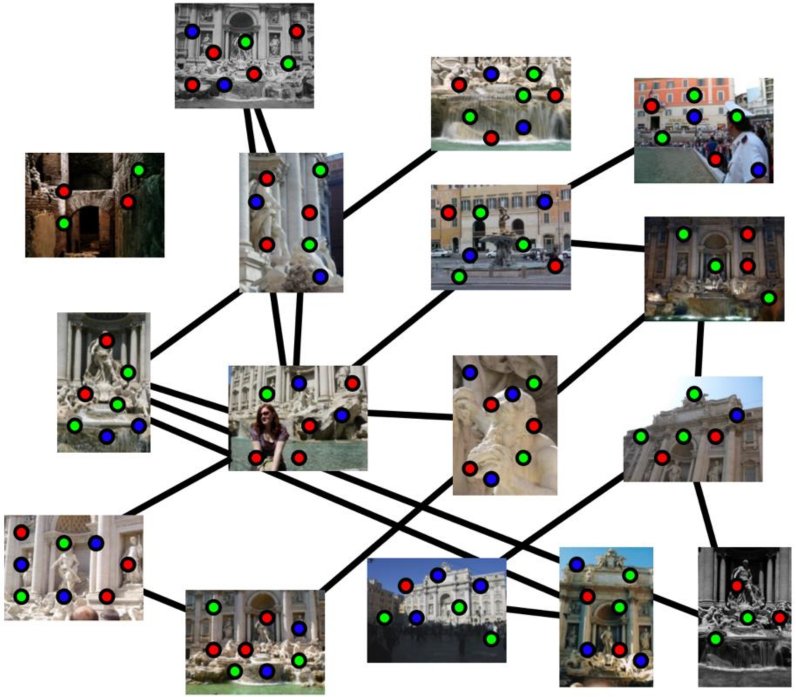
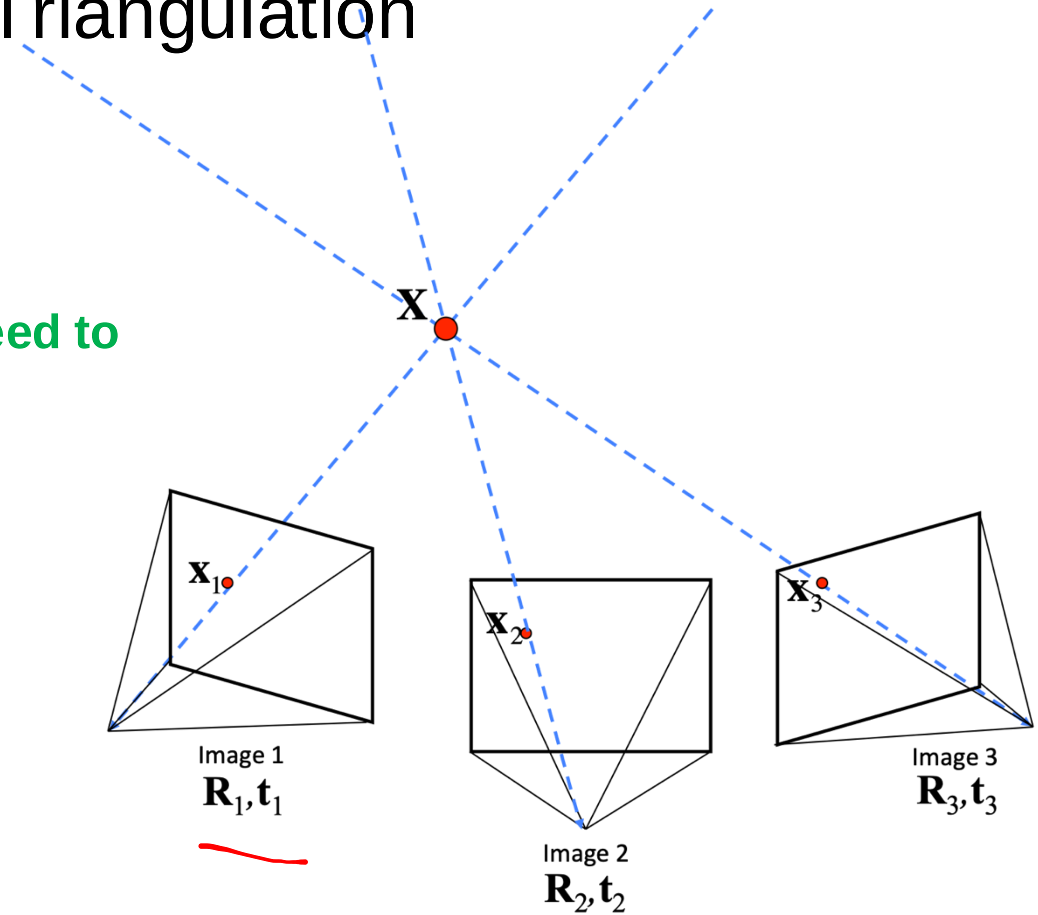


Image credit: Noah Snavely

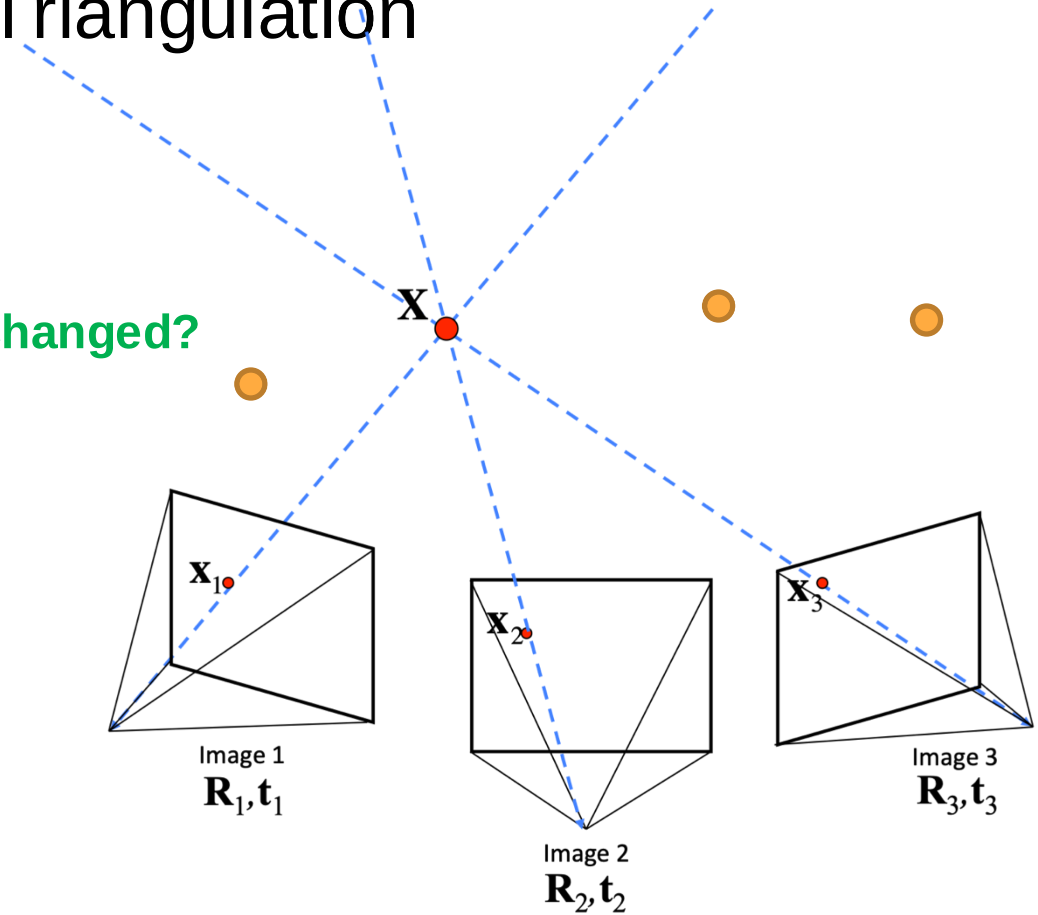
Multi-view Triangulation

Are we guaranteed to converge?



Multi-view Triangulation

What could be changed?



Bundle Adjustment

$$\min_{\{\mathbf{R}_k,\mathbf{t}_k\},\{\mathbf{X}_i\}} \sum_i \sum_k w_{ik} \|\mathbf{x}_{ik} - \pi_{\mathbf{K}}([\mathbf{R}_k|\mathbf{t}_k]\mathbf{X}_i)\|_2^2$$

Extrinsic

3D Points

Detection Keypoints

Perspective Projection

Unknow 3D Location

Bundle Adjustment

What is the difference between calibration vs structure from motion?

$$\min_{\{\mathbf{R}_k,\mathbf{t}_k\},\{\mathbf{X}_i\}} \sum_i \sum_k w_{ik} \|\mathbf{x}_{ik} - \pi_{\mathbf{K}}([\mathbf{R}_k|\mathbf{t}_k]\mathbf{X}_i)\|_2^2$$

Extrinsic

3D Points

Detection Keypoints

Perspective Projection

Unknow 3D Location

Continuous Optimization

MAP inference: find the best configuration that minimize the energy

$$\mathbf{y}^* = \operatorname*{argmin}_{\mathbf{y} \in \mathcal{Y}} E(\mathbf{x}, \mathbf{y}; \boldsymbol{\theta})$$

There is no universal solution. Inference algorithm choice is depending on:

- **Continuous vs Discrete Variables**: numerical approach or search-based
- **Energy Functions**: convex, submodular, piecewise linear, quadratic, etc.
- **Graphical Model Structures**: containing loops or not; having high-order terms or not?

MAP Inference: Gradient Descent

- Minimize continuous-valued energy based models by numerical optimization:

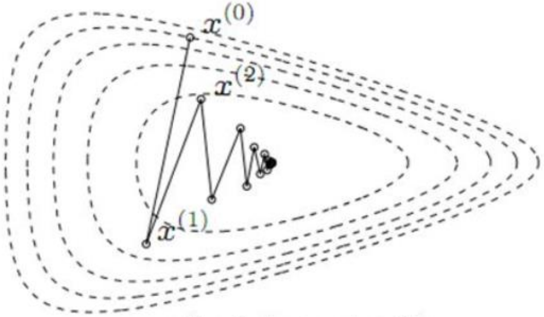
$$\mathbf{y}^{(t+1)} = \mathbf{y}^{(t)} - \gamma \nabla_{\mathbf{y}} E(\mathbf{x}, \mathbf{y}^{(t)})$$

- Pros: simple and straightforward, works for all differentiable energies
- Cons: (sub-)differentiability requirements and slow to convergence

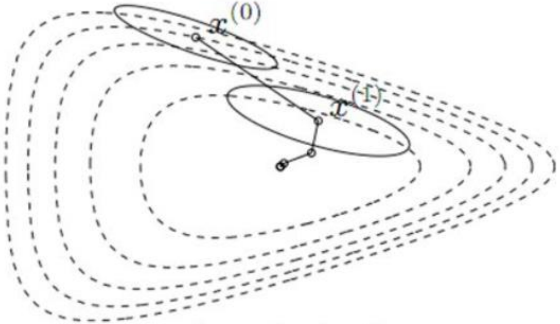
MAP Inference: Newton Method

- For twice-differentiable energy function, one could use Newton's method:

$$\mathbf{y}^{(t+1)} = \mathbf{y}^{(t)} - \left(\nabla^2_{\mathbf{y}} E(\mathbf{x}, \mathbf{y}^{(t)})\right)^{-1} \nabla_{\mathbf{y}} E(\mathbf{x}, \mathbf{y}^{(t)})$$



gradient descent with
backtracking line search



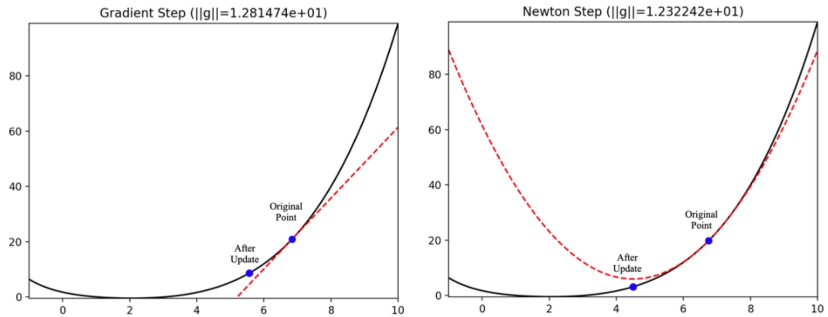
Newton's method with
backtracking line search

MAP Inference: Newton Method

- For twice-differentiable energy function, one could use Newton's method:

$$\mathbf{y}^{(t+1)} = \mathbf{y}^{(t)} - \left(\nabla^2_{\mathbf{y}} E(\mathbf{x}, \mathbf{y}^{(t)})\right)^{-1} \nabla_{\mathbf{y}} E(\mathbf{x}, \mathbf{y}^{(t)})$$

- Pros: capturing curvature, better convergence, less likely stuck, less tuning
- Cons: expensive to compute inverse Hessian, hard to scale



MAP Inference: Gauss-Newton

- If the energy has a sum of square form:

$$E(\mathbf{y}) = \sum_{\alpha} E_{\alpha}(\mathbf{y}) = \sum_{\alpha} (r_{\alpha}(\mathbf{y}))^2 = \|\mathbf{r}(\mathbf{y})\|_2^2$$

- For each iteration t:
  - Taylor approximation for the residual function: $\mathbf{r}(\mathbf{y}) \approx \mathbf{r}(\mathbf{y}^{(t)}) + \mathbf{J}_{\mathbf{r}}^T (\mathbf{y} - \mathbf{y}^{(t)})$
  - Solving least square:

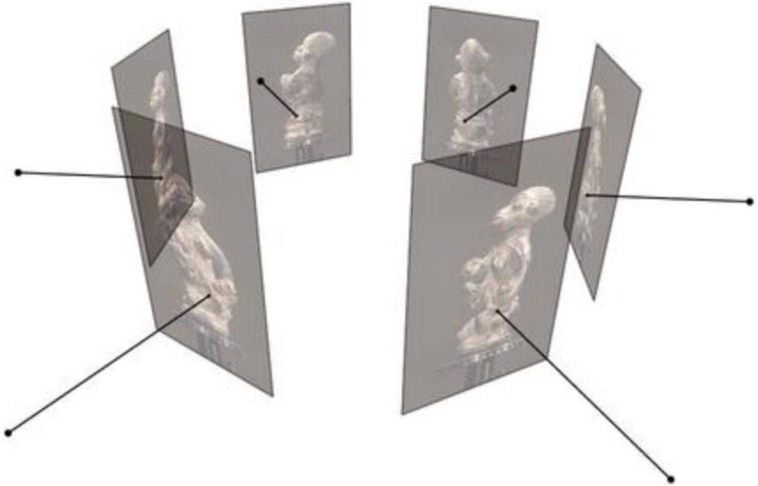
$$\mathbf{y}^{(t+1)} = \arg\min_{\mathbf{y}} \|\mathbf{r}(\mathbf{y}^{(t)}) + \mathbf{J}_{\mathbf{r}}^T (\mathbf{y} - \mathbf{y}^{(t)})\|_2^2$$

How to get the solution? Today's Quiz

Multi-View Stereo

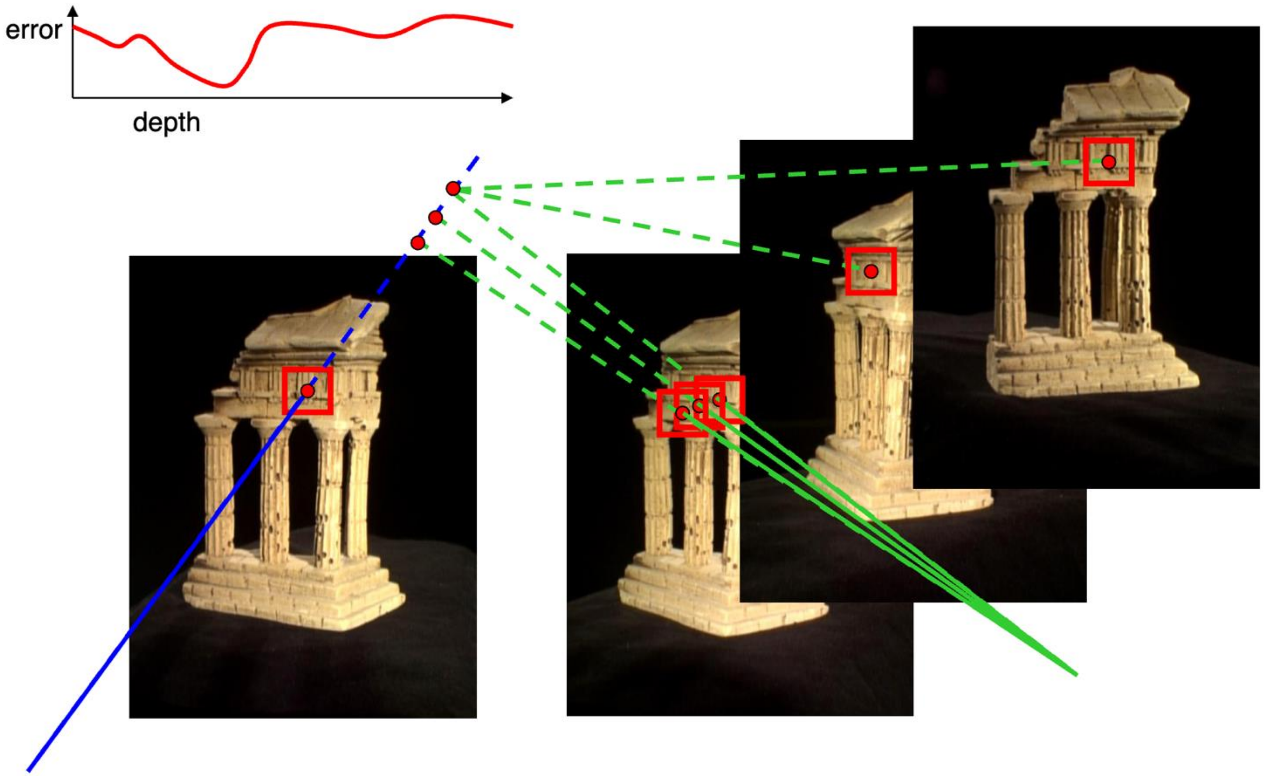
- Input: images from several viewpoints with known camera poses and calibratior
- Output: 3D object model

Why are SFM 3D points insufficient?

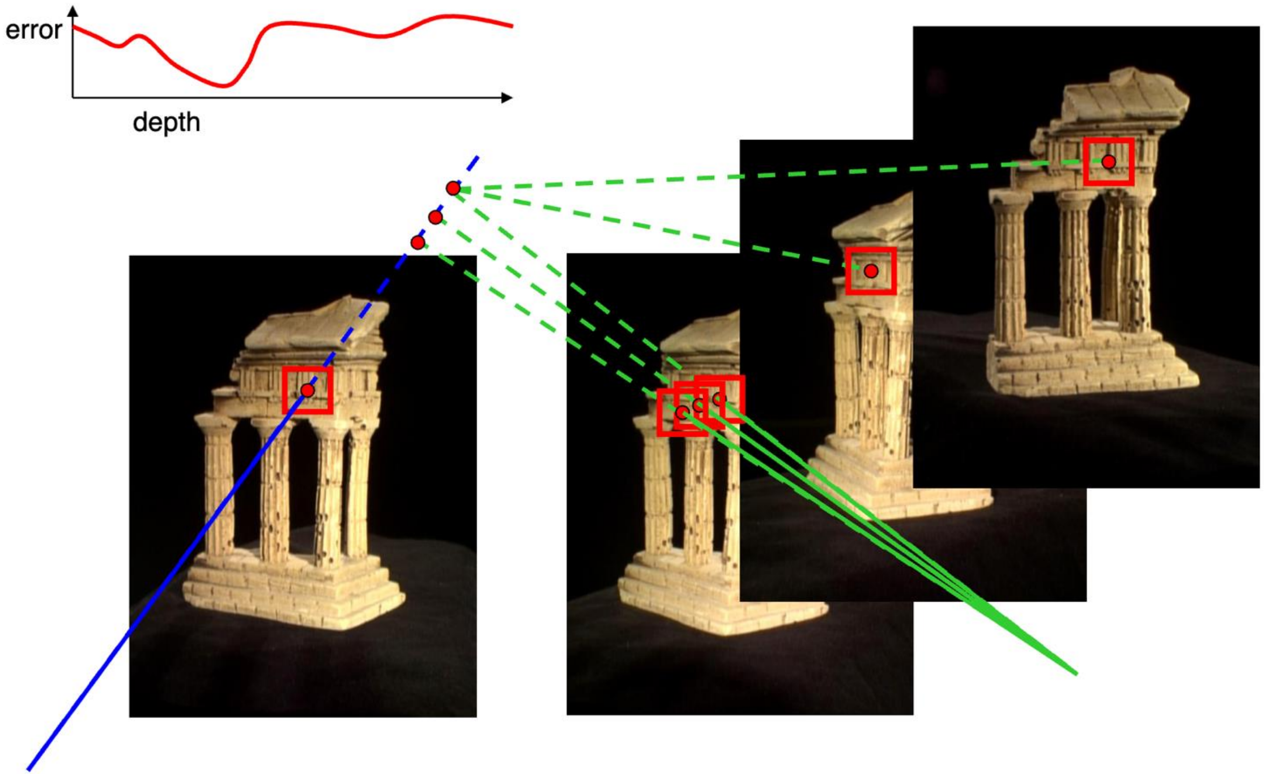


Figures by Carlos Hernandez

Measuring the matching cost



Measuring the matching cost

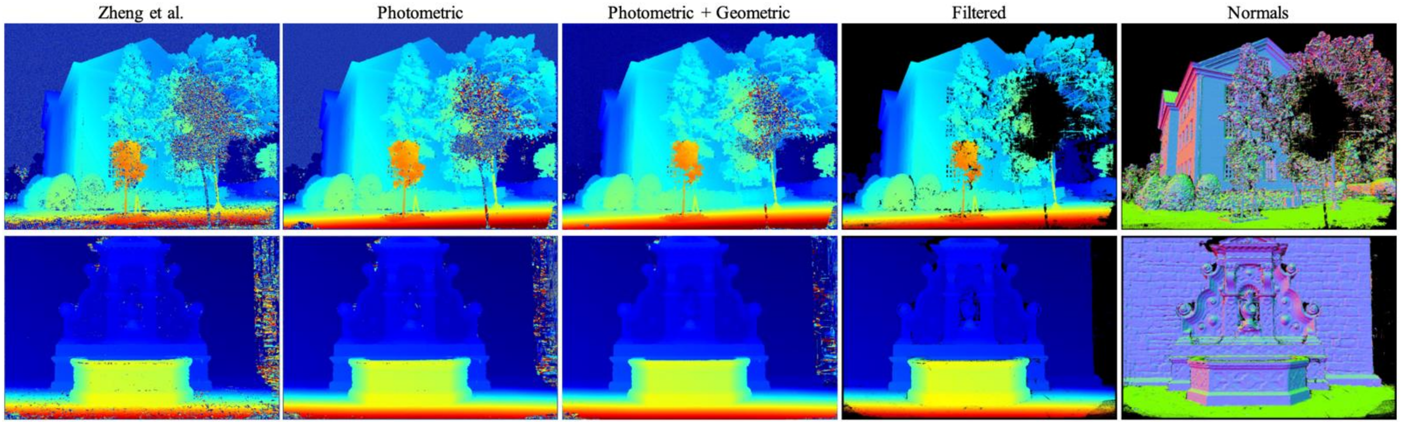


Colmap: Photometric + Geometric Cost + View Select

- Photometric consistency: normalized cross correlation

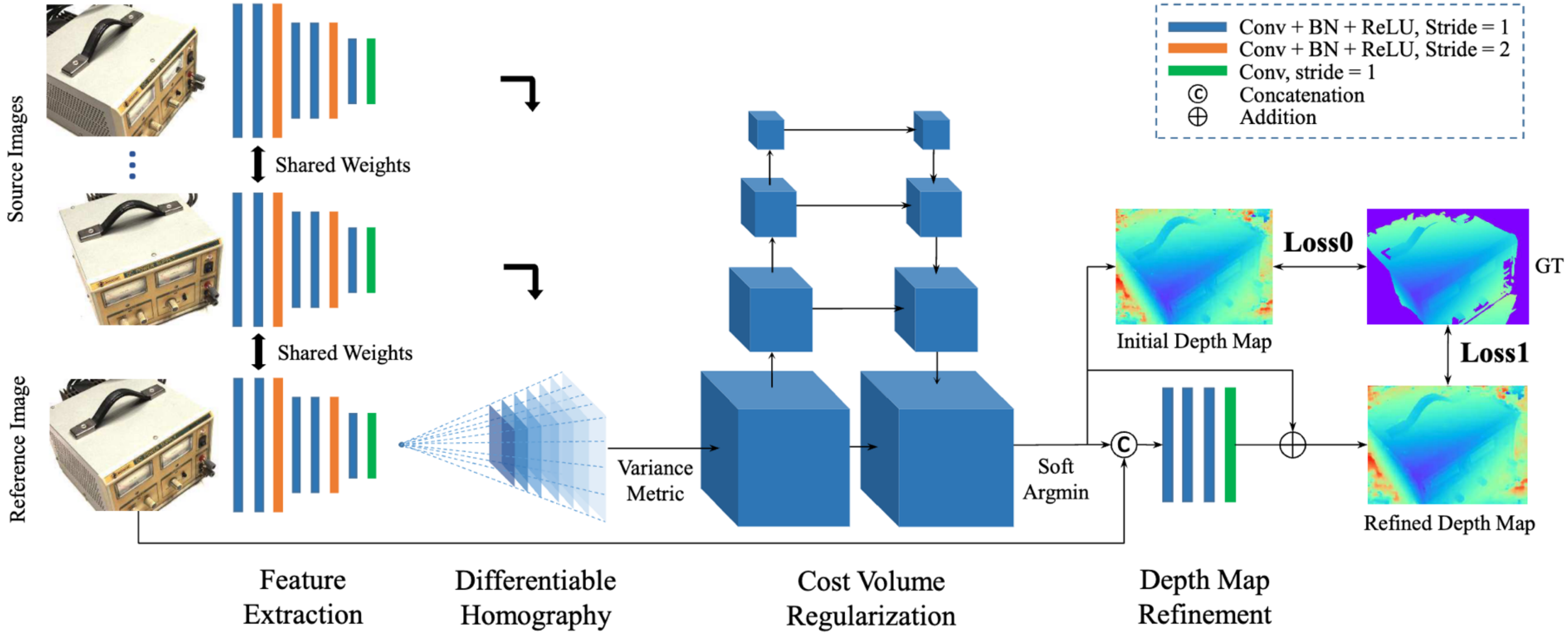
$$\rho_l^m = \frac{\mathrm{cov}_w(\boldsymbol{w}_l, \boldsymbol{w}_l^m)}{\sqrt{\mathrm{cov}_w(\boldsymbol{w}_l, \boldsymbol{w}_l)\ \mathrm{cov}_w(\boldsymbol{w}_l^m, \boldsymbol{w}_l^m)}}$$

- Geometry consistency: forward-backward reprojection error



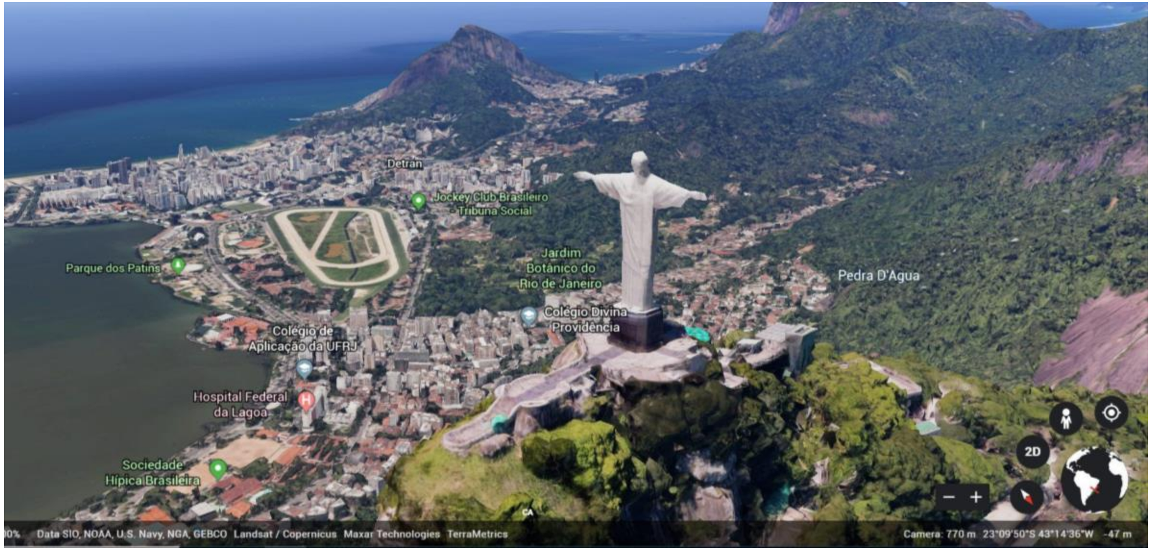
Pixelwise View Selection for Unstructured Multi-View Stereo, 2016

MVSNet



MVSNet: Depth Inference for Unstructured Multi-view Stereo, 2018

3D Reconstruction: SFM + MVS



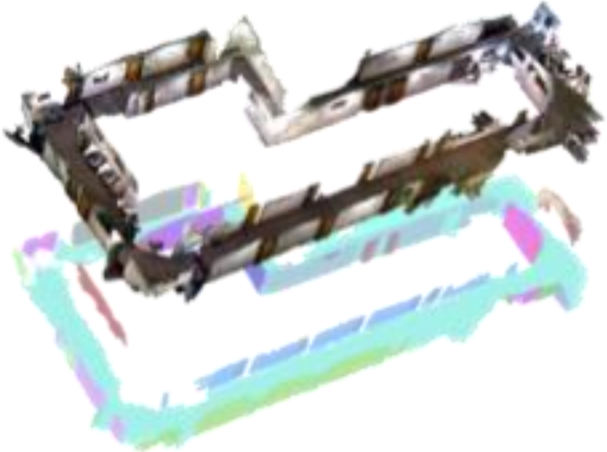


Image credit: Google, Michael Keass

Visual SLAM: Online SFM

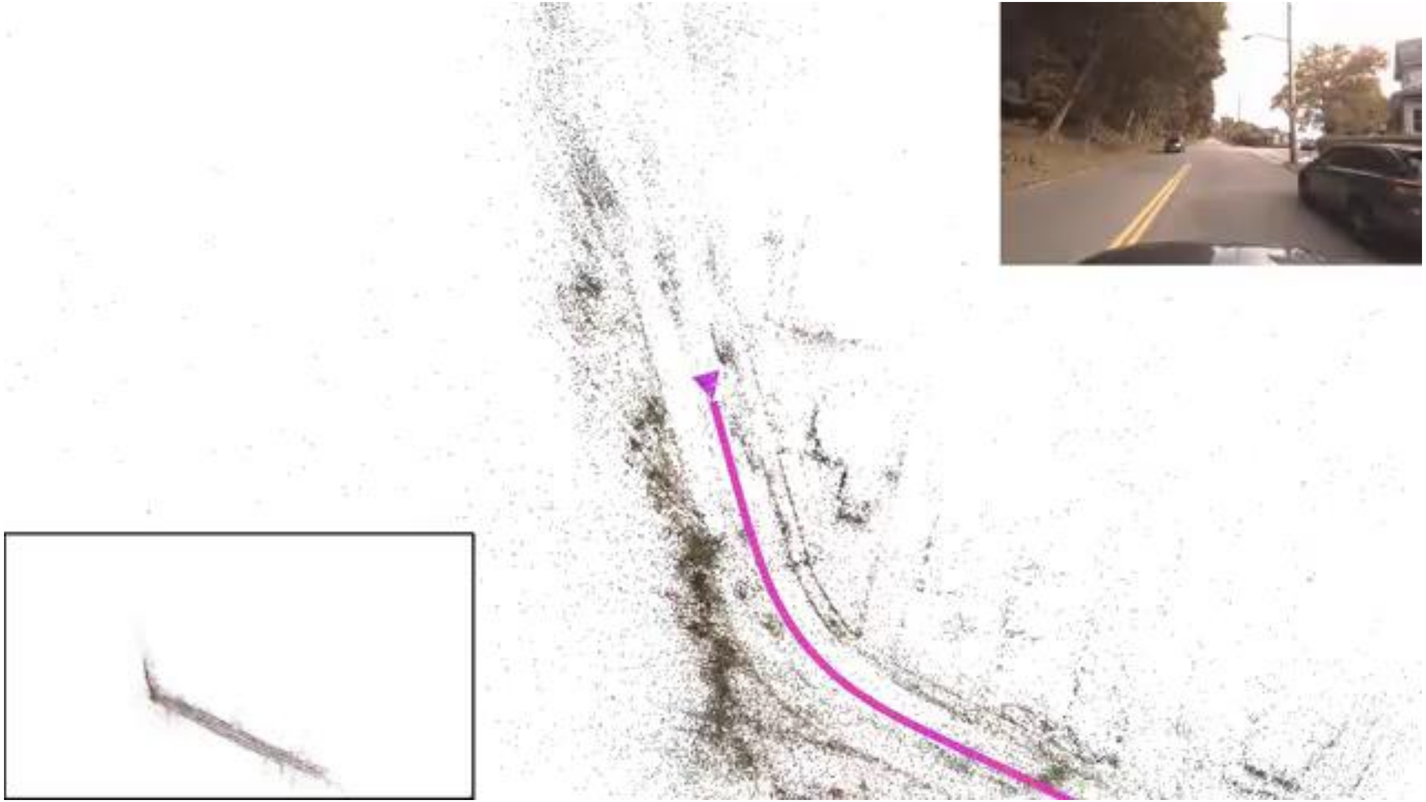
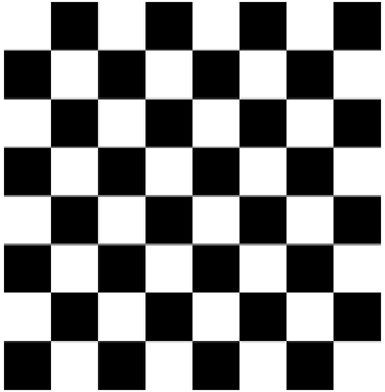
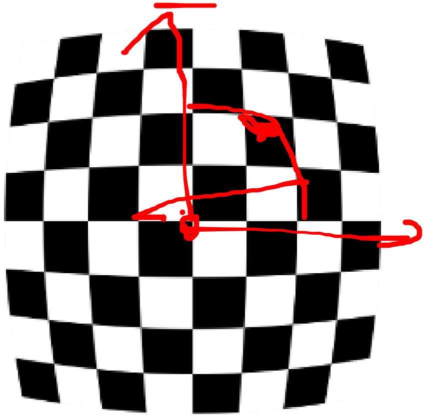


Image credit: Yang et al. ICRA 2021

Camera Distortion

$$x_{distorted} = x(1 + k_1 r^2 + k_2 r^4 + k_3 r^6)$$
$$y_{distorted} = y(1 + k_1 r^2 + k_2 r^4 + k_3 r^6)$$


No distortion


Positive radial distortion
(Barrel distortion)

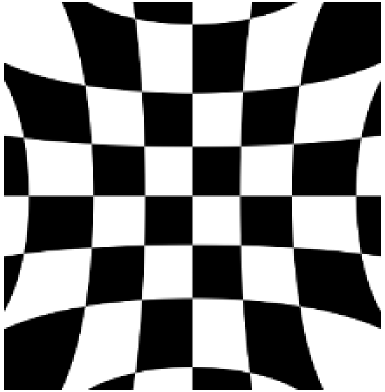

Negative radial distortion
(Pincushion distortion)

Image credit: OpenCV

Camera Distortion

- Remember to **cv2.undistort** the image if you want to reason in 3D.

before

after

Image credit: OpenCV

Event Cameras

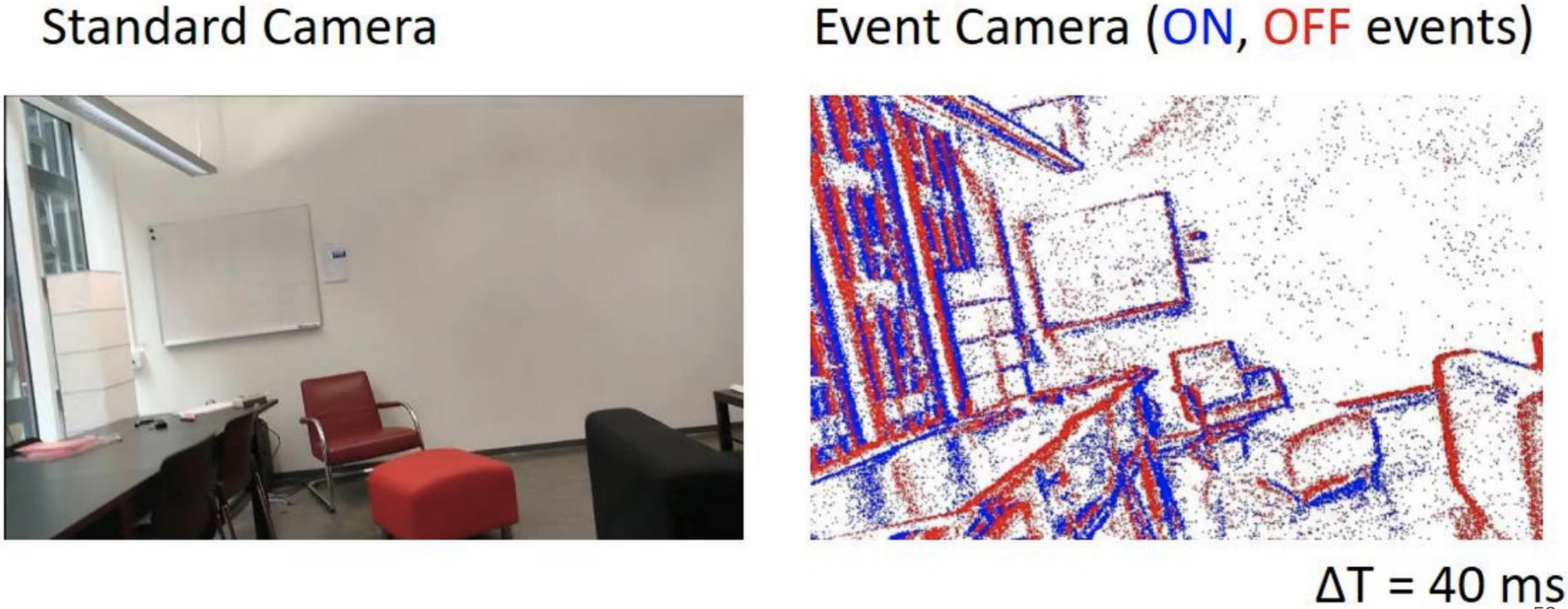


Image credit: Davide Scaramuzza

Fisheye Camera / Omnidirectional Camera

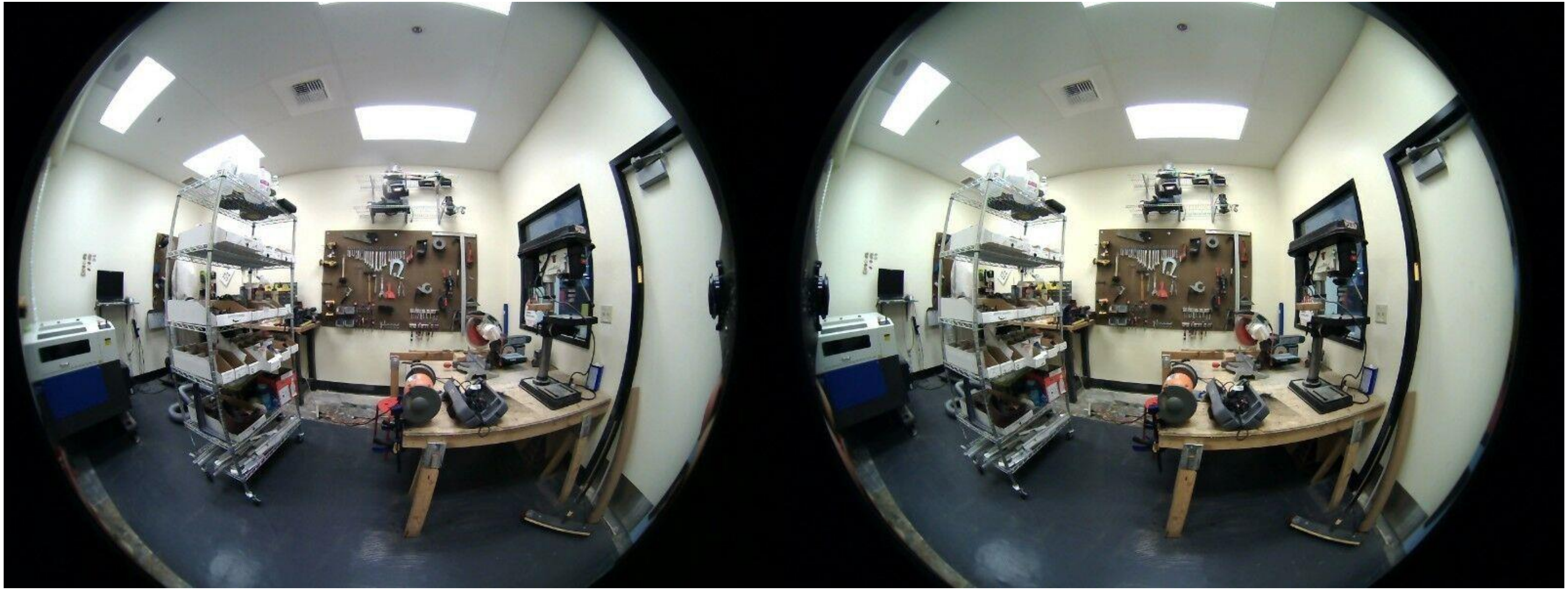


Image credit: OZ robotics

What I Didn't Cover

- Stereo Rectification

Making two stereo camera frontal parallel.

- Five-Point Algorithms

Recover Essential/Fundamental Matrix from 2D-2D Correspondences

- Projection Matrix Decomposition

Recover R and t from camera projection matrix

- Essential Decomposition

Recover R and t from essential matrix estimation

- Perspective-n-Projection (PnP)

Recover R and t from 2D-3D correspondences

Check Szeliski or MVG Book if you want to know these concepts