Lecture 14 – Structure-from-Motion

Dec 18th, 2018

Danping Zou,

Associate Professor

Institute for Sensing and Navigation

上海交通大學
SHANGHAI JIAO TONG UNIVERSITY

Structure from motion

- We have learnt the most of knowledge to build a **Structure from Motion** (**SfM**) system.
  - Single view :
    - Perspective projection
    - Camera intrinsic, distortion
    - Camera calibration
    - Pose estimation
  - Two view :
    - Epipolar constraint
    - Essential/Fundamental matrix, RANSAC
    - Triangulation
    - Stereo rectify, calibration, and matching
  - Mathematics:
    - Homogeneous linear system (SVD)
    - Nonlinear least square (Gauss-Newton, Levenberg Marquadt)

Outline

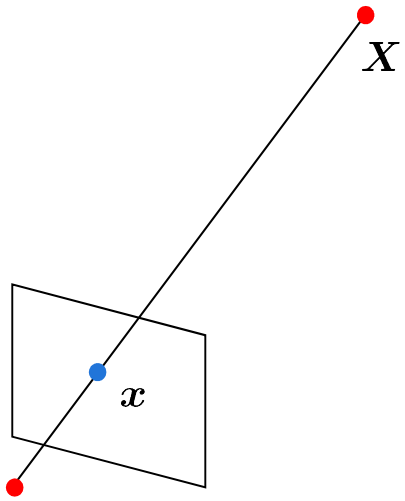
- What's structure-from-motion problem
  - Preliminarily analysis
- Factorization approach
  - Orthographic projection
  - SVD factorization get initial decomposition
  - Enforce the orthogonality for rotation matrices
  - Get the final factorization
- Incremental approach
  - Initialization
  - Pose estimation
  - Triangulation
  - Bundle adjustment

What is **structure from motion ?**

- **Structure from motion** is problem that tries to recover both 3D point clouds and camera poses from multiple views of 2D observations.
  - Inputs:
    - 2D feature points
  - Outputs:
    - Camera pose for each view
    - 3D coordinates for each point

Structure from motion

- Single view imaging model – perspective projection



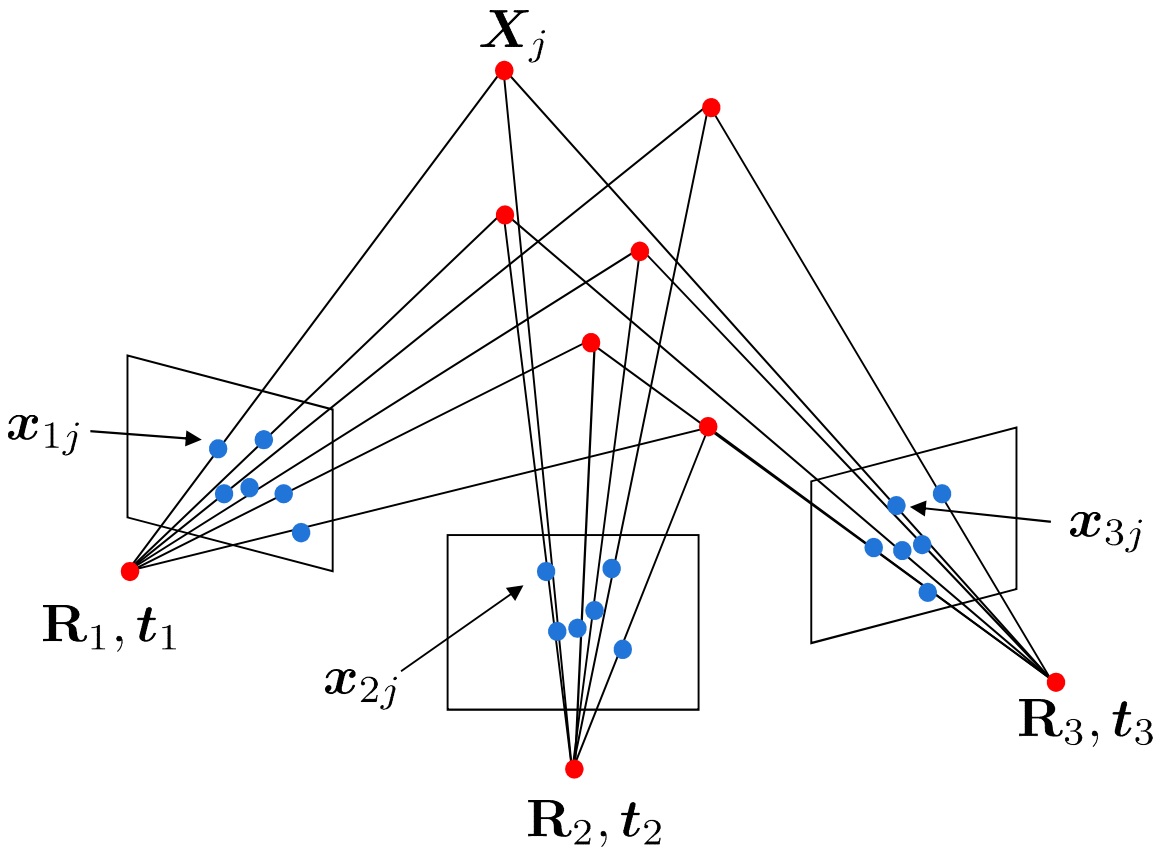
$$x \sim [\mathbf{R} \mid t]X \quad (m \sim \mathbf{K}x)$$

Pixel coordinates-> Image plane

$$x = \begin{bmatrix} \frac{\mathbf{R}_{(1,:)}X + t_1}{\mathbf{R}_{(3,:)}X + t_3} \\ \\ \frac{\mathbf{R}_{(2,:)}X + t_2}{\mathbf{R}_{(3,:)}X + t_3} \end{bmatrix} \text{ where } t = \begin{bmatrix} t_1 \\ t_2 \\ t_3 \end{bmatrix}$$

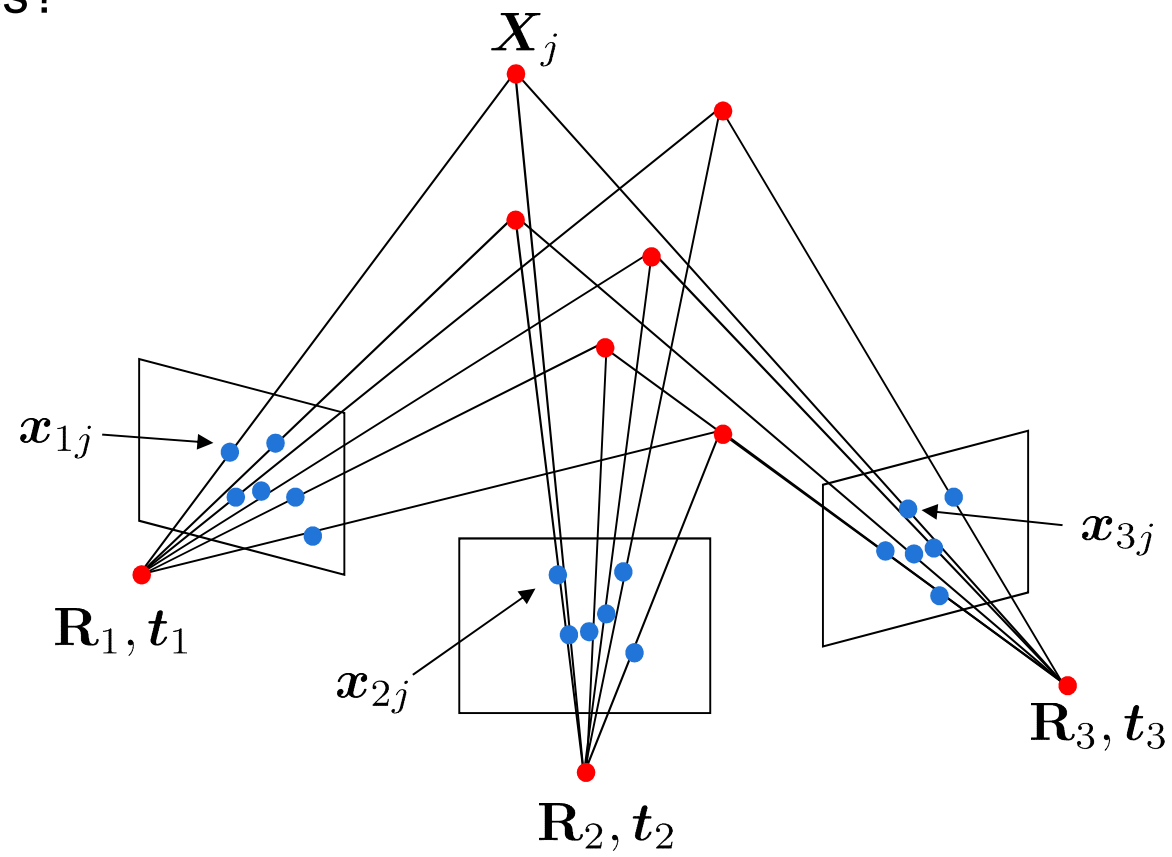
Structure from motion

- If we have multiple views of 2D observations,



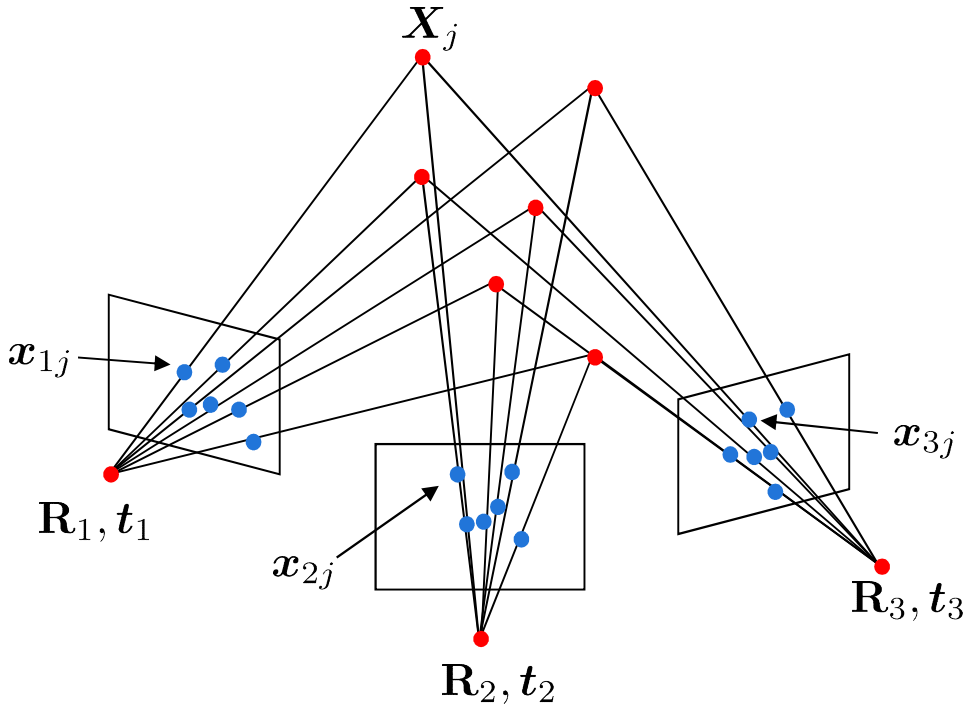
Structure from motion

- How do we compute both the 3D point coordinates and the camera poses?



Structure from motion

- Suppose there are N 3D points and M views.
- Each point is observable at every views.
- The number of 2D observations is $N \times M$



Can we solve this problem ?

Structure from motion

- The number of unknowns:
  - Camera pose (rotation : 9 unknowns + translation : 3 unknowns)
  - 3D point (3 unknowns)

$$\#unkown = 12M + 3N$$

- The number of equations:
  - Each 2D feature point gives rise to two equations (x,y)

$$\#equation = 2M \times N$$

Structure from motion

- If the number of equations is not less than the number of unknowns, the problem can be solved.

$$\#equation = 2M \times N$$
$$\geq \#unkown = 12M + 3N$$
$$(2N - 12)M \geq 3N$$

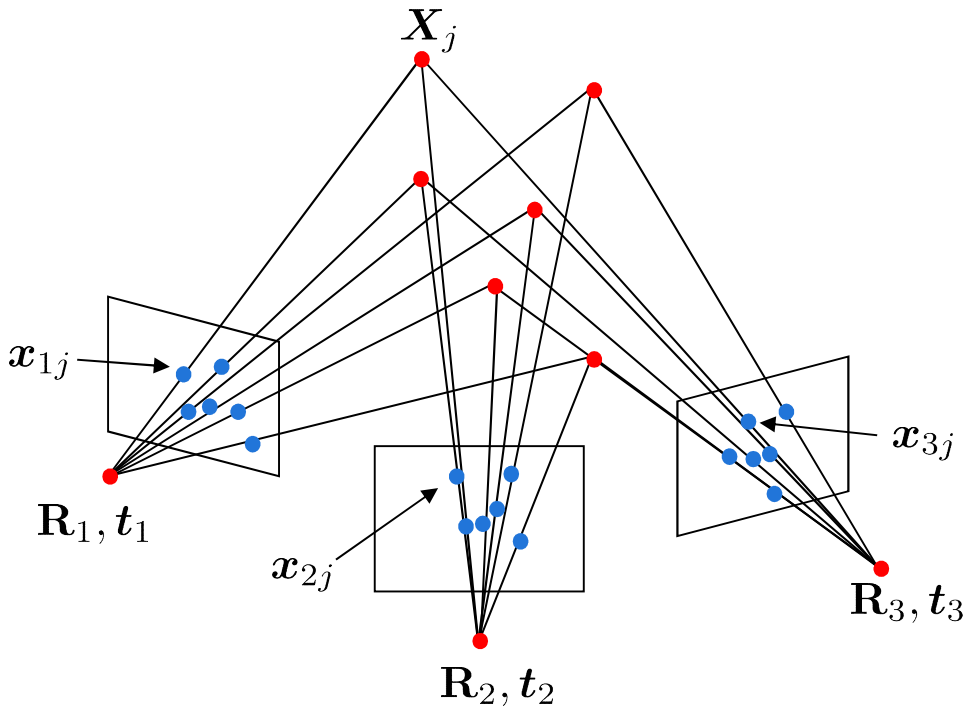
$$\Rightarrow \boxed{N > 6, M \geq \frac{3N}{2N-12}}$$

$$N = 7, M \geq 11$$
$$N = 10, M \geq 4$$
$$\cdots$$
$$N = 30, M \geq 2$$

The more points, the fewer views are required.

Structure from motion

- It is possible to solve this problem, but it is a highly nonlinear problem.
- The key is to get the initial solution.



$$\begin{matrix} \mathbf{R}_1, \boldsymbol{t}_1 \\ \vdots \\ \mathbf{R}_M, \boldsymbol{t}_M \end{matrix}$$

$$\begin{matrix} \boldsymbol{X}_1 \\ \vdots \\ \boldsymbol{X}_N \end{matrix}$$

Structure-from-motion

Factorization/Batch

1990

Tomasi C, Kanade T

Shape and motion from image streams under orthography: a factorization method,IJCV,1992

Incremental approach

2000

- **Photo Tourism (Bundler)**
- **Rome in a Day**

Factorization approach

Structure from motion

- **Factorization method**
  - Using orthographic or affine projection instead of perspective projection
  - Close-form solution to the 3D structure and the camera poses all together

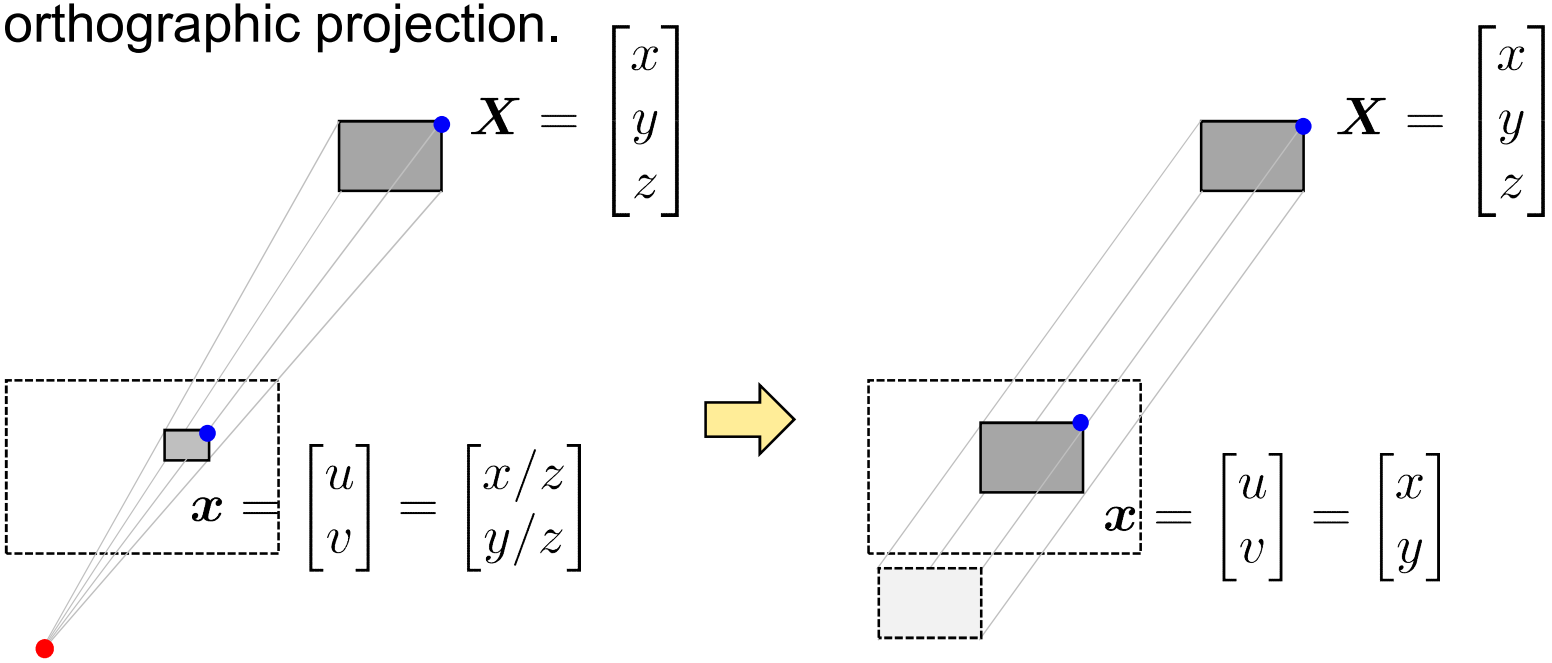
Tomasi C, Kanade T. Shape and motion from image streams under orthography: a factorization method,IJCV,1992

《像外行一样思考,像专家一样实践》
-金出武雄

金出武雄
(The Robotics Institute, CMU)

Structure from motion

- The key idea is to approximated the perspective projection by the orthographic projection.



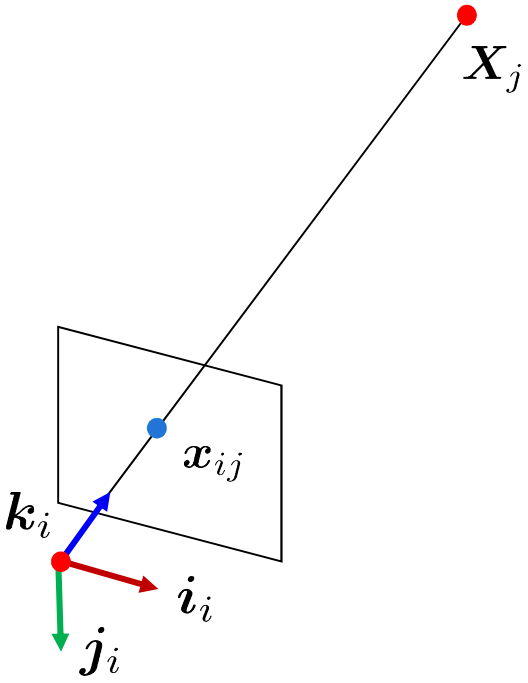
Perspective projection **Orthographic projection**

The approximation is better when the object is far away.

Factorization methods

- The camera pose of the i-th view is represented by $(\mathbf{R}_i, \boldsymbol{t}_i)$,

(where $\mathbf{R}_i = [\boldsymbol{i}, \boldsymbol{j}, \boldsymbol{k}]^{\mathrm{T}}$) which is the transformation from the world frame to the camera frame.



$$\tilde{\boldsymbol{x}}_{ij} \sim \mathbf{R}_i \quad \mathbf{X}_j + \boldsymbol{t}_i$$

$$3 \times 1 \quad 3 \times 3 \;\; 3 \times 1$$

$$\boldsymbol{x}_{ij} = [\boldsymbol{i}_i, \boldsymbol{j}_i]^{\mathrm{T}} \boldsymbol{X}_j + \boldsymbol{t}_{i,1:2}$$

$$u_{ij} = \boldsymbol{i}_i^{\mathrm{T}} \boldsymbol{X}_j + t_{i,1}$$

$$v_{ij} = \boldsymbol{j}_i^{\mathrm{T}} \boldsymbol{X}_j + t_{i,2}$$

Factorization methods

- Remove the translation :
  - Choose the image origin be the centroid of 2D points
  - Choose the scene origin be the centroid of 3D points
- We can drop the camera translation

$$\boldsymbol{x}_{ij} = \left[\boldsymbol{i}_i, \boldsymbol{j}_i\right]^{\mathrm{T}} \boldsymbol{X}_j + \cancel{\boldsymbol{t}_{i,1:2}}$$

$$\Downarrow$$

$$\boldsymbol{x}_{ij} = \left[\boldsymbol{i}_i, \boldsymbol{j}_i\right]^{\mathrm{T}} \boldsymbol{X}_j$$

$$\downarrow$$

$$\boldsymbol{x}_{ij} = \boldsymbol{\pi}_i \boldsymbol{X}_j$$

Factorization methods

- Projection of N feature points in the i-th view :

$$\left[\boldsymbol{x}_{i1}\cdots\boldsymbol{x}_{ij}\cdots\boldsymbol{x}_{iN}\right]=\boldsymbol{\pi}_i\left[\boldsymbol{X}_1\cdots\boldsymbol{X}_j\cdots\boldsymbol{X}_N\right]$$

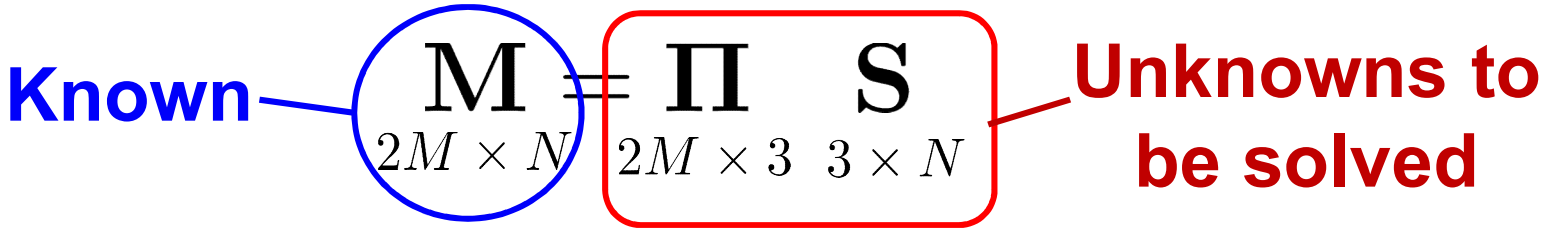
- Projection of N feature points in M views :

$$\underbrace{\begin{bmatrix}\boldsymbol{x}_{11} & \boldsymbol{x}_{12} & \cdots & \boldsymbol{x}_{1N}\\ \boldsymbol{x}_{21} & \boldsymbol{x}_{22} & \cdots & \boldsymbol{x}_{2N}\\ \vdots & \vdots & \ddots & \vdots\\ \boldsymbol{x}_{M1} & \boldsymbol{x}_{M2} & \cdots & \boldsymbol{x}_{MN}\end{bmatrix}}_{2M\times N}=\underbrace{\begin{bmatrix}\boldsymbol{\pi}_1\\ \boldsymbol{\pi}_2\\ \vdots\\ \boldsymbol{\pi}_M\end{bmatrix}}_{2M\times 3}\underbrace{\left[\boldsymbol{X}_1,\boldsymbol{X}_2\cdots\boldsymbol{X}_N\right]}_{3\times N}$$

$$\mathbf{M}=\mathbf{\Pi S}$$

Factorization methods

- We can write the measurement equations by



- Our goal is to factorize the observation matrix $\mathbf{M}$ to get the rotation matrix and the shape matrix.
- We observed that $\mathbf{M}$ is at most rank3, since
  - Rank of $\mathbf{\Pi}$ is 3
  - Rank of $\mathbf{S}$ is 3
  - Product of 2 matrices of rank 3 has rank 3.
- But with noise, rank($\mathbf{M}$) might be > 3.

Factorization methods

- SVD factorization:
  - Apply SVD to the observation matrix, we have

$$\underset{2M\times N}{\mathbf{M}} = \underset{2M\times N}{\mathbf{U}}\ \underset{N\times N}{\Sigma}\ \underset{N\times N}{\mathbf{V}^{\mathrm{T}}}$$

  - As $\mathbf{M}$ has rank 3, it has three non-zero singular values, we the get corresponding columns of $\mathbf{U}$ and $\mathbf{V}$.

$$\mathbf{M} = [\boldsymbol{U}_1, \boldsymbol{U}_2, \boldsymbol{U}_3, \cdots]\mathbf{diag}(\sigma_1, \sigma_2, \sigma_3, 0, \cdots)[\boldsymbol{V}_1, \boldsymbol{V}_2, \boldsymbol{V}_3, \cdots]^{\mathrm{T}}$$

$$\underset{2M\times N}{\mathbf{M}} = \underset{2M\times 3}{\mathbf{U}_{1:3}}\ \underset{3\times 3}{\mathbf{diag}(\sigma_1, \sigma_2, \sigma_3)}\ \underset{3\times N}{\mathbf{V}^{\mathrm{T}}_{1:3}}$$

$$\mathbf{M} = \boldsymbol{\Pi}'\ \mathbf{S}'$$

Factorization methods

- $\mathbf{\Pi}'$ differ from $\mathbf{\Pi}$ by a linear transformation $\mathbf{A}$:

$$\mathbf{M} = \mathbf{\Pi}\mathbf{S} \qquad \mathbf{M} = \mathbf{\Pi}'\mathbf{S}'$$

$$\mathbf{\Pi}' = \mathbf{\Pi}\mathbf{A} \quad 3 \times 3$$

- If we can find $\mathbf{A} \in \mathbb{R}^{3\times 3}$, we can solve both $\mathbf{\Pi}$ and $\mathbf{S}$.
- We solve $\mathbf{A}$ by enforcing the orthogonality on $\mathbf{\Pi}$.

Factorization methods

- We revisit the projection matrix $\mathbf{\Pi}$

$$\mathbf{\Pi} = \begin{bmatrix} \boldsymbol{\pi}_1 \\ \boldsymbol{\pi}_2 \\ \vdots \\ \boldsymbol{\pi}_M \end{bmatrix} \longrightarrow \boldsymbol{\pi}_i = \begin{bmatrix} \boldsymbol{i}_i^{\mathrm{T}} \\ \boldsymbol{j}_i^{\mathrm{T}} \end{bmatrix} \xrightarrow{\boldsymbol{i}_i \perp \boldsymbol{j}_i} \boldsymbol{\pi}_i \boldsymbol{\pi}_i^{\mathrm{T}} = \begin{bmatrix} 1 & 0 \\ 0 & 1 \end{bmatrix}$$

- The projection matrix of an orthographic camera has two orthonormal rows.
- Enforce the orthonormal constrains on each π_i', such that :

$$\boldsymbol{\pi}_i' \mathbf{A} \mathbf{A}^{\mathrm{T}} \boldsymbol{\pi}_i'^{\mathrm{T}} = \begin{bmatrix} 1 & 0 \\ 0 & 1 \end{bmatrix}$$

Factorization methods

- From $\boldsymbol{\pi}'_i \mathbf{A}\mathbf{A}^{\mathrm{T}} \boldsymbol{\pi}'^{\mathrm{T}}_i = \begin{bmatrix} 1 & 0 \\ 0 & 1 \end{bmatrix}$, letting $\mathbf{Q} = \mathbf{A}\mathbf{A}^{\mathrm{T}}$, we have

$$\boldsymbol{\pi}'_i = \begin{bmatrix} \boldsymbol{i}'^{\mathrm{T}}_i \\ \boldsymbol{j}'^{\mathrm{T}}_i \end{bmatrix} \Rightarrow \begin{aligned} \boldsymbol{i}'^{\mathrm{T}}_i \mathbf{Q} \boldsymbol{i}'_i &= 1 \\ \boldsymbol{j}'^{\mathrm{T}}_i \mathbf{Q} \boldsymbol{j}'_i &= 1 \\ \boldsymbol{i}'^{\mathrm{T}}_i \mathbf{Q} \boldsymbol{j}'_i &= 0 \end{aligned}$$

$$\mathbf{Q} = \begin{bmatrix} q_1 & q_2 & q_3 \\ q_2 & q_4 & q_5 \\ q_3 & q_5 & q_6 \end{bmatrix}$$

- Finally we can get a linear system of equations:

$$\underset{3M \times 6}{\mathbf{H}} \quad \underset{6 \times 1}{\boldsymbol{q}} \quad = \quad \underset{3M \times 1}{\boldsymbol{b}} \qquad (\text{where } \boldsymbol{q} = \begin{bmatrix} q_1 \\ q_2 \\ \vdots \\ q_6 \end{bmatrix})$$

Factorization methods

- This is a linear least squared problem, which can be solved by :

$$\mathbf{H}\boldsymbol{q} = \boldsymbol{b} \;\Rightarrow\; \boldsymbol{q} = (\mathbf{H}^T\mathbf{H})^{-1}\mathbf{H}\boldsymbol{b}$$

- Or using the backslash '\' operator in Matlab to get the solution:

$$\boldsymbol{q} = \mathbf{H} \setminus \boldsymbol{b}$$

- After that, we can get $\mathbf{A}$ through Cholesky decomposition : .
  In Matlab, we have

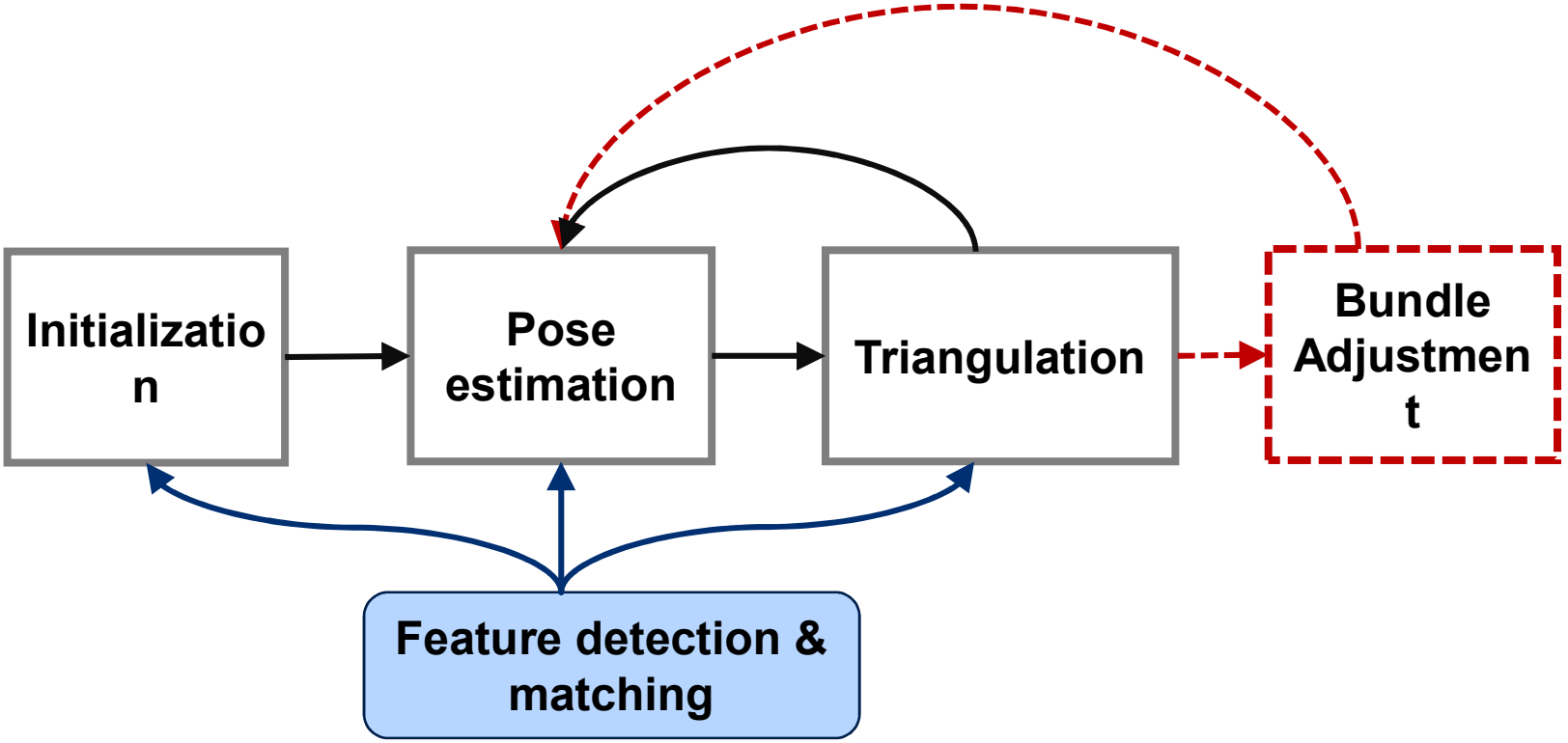
$$\mathbf{Q} = \mathbf{A}\mathbf{A}^{\mathrm{T}}$$

$$\mathbf{A} = \mathbf{chol}(\mathbf{Q})^{\mathrm{T}}$$

Incremental approach

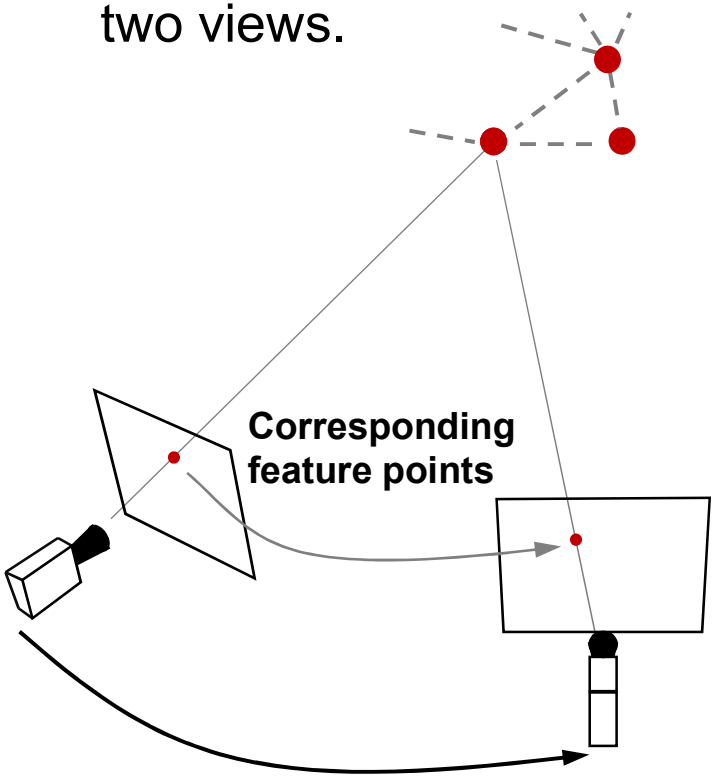
Incremental SfM

- A typical pipeline of a incremental structure from motion system:



Incremental SfM

- **Initialization** – Generate the 3D points and camera poses of the initial two views.

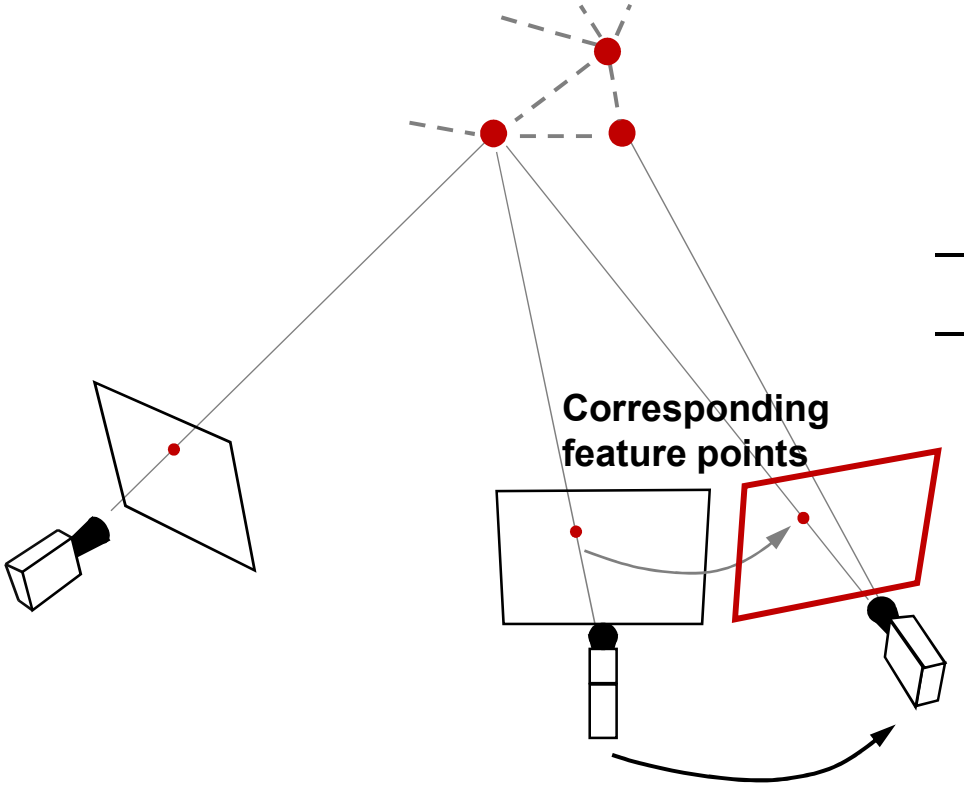


  - Feature detection & matching
    - Find correspondences between two views
      - SIFT, SURF, ORB
  - Find the essential matrix
    - RANSAC, 5-point algorithm, 8-point algorithm
  - Extract the camera poses of the two views
    - SVD
  - Generate the initial 3D points
    - Triangulation

Initialization | **Pose estimation** | **Triangulation** | **Bundle adjustment**

Incremental SfM

- **Pose estimation**



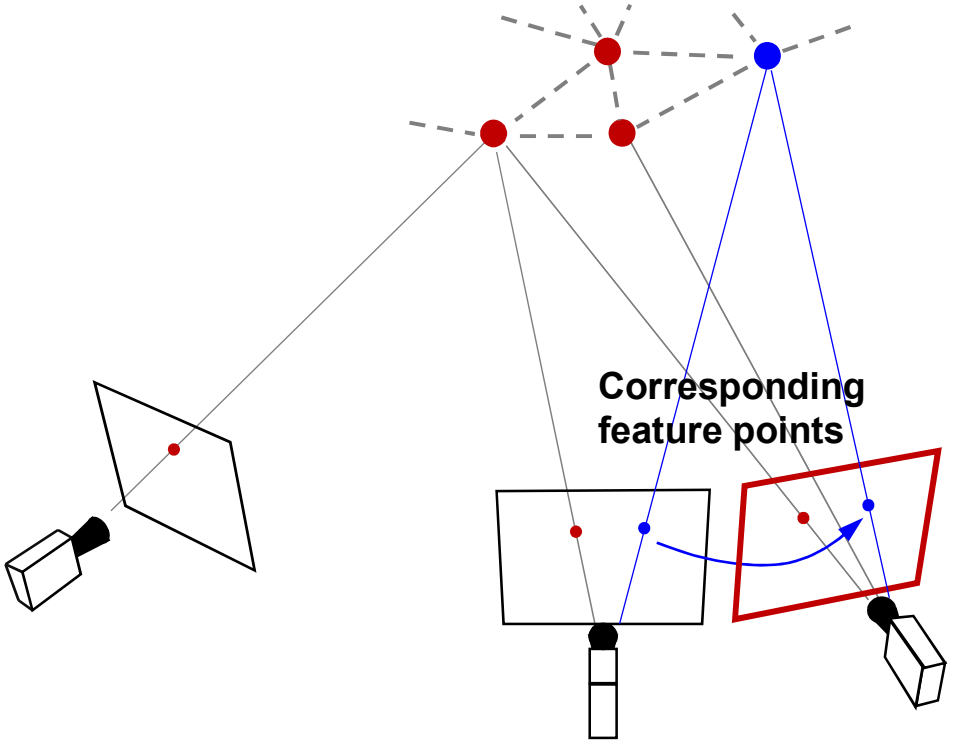
– Feature detection & matching
  - Find correspondences between the current view and previous view (KLT, ORB)
– Solve the PnP problem
– Handle the outliers (RANSAC / M-estimator)

Initialization | Pose estimation | Triangulation | Bundle adjustment

Incremental SfM

- **Triangulation**

Generate new 3D points from corresponding 2D points.



$$x \times PX = 0$$
$$x' \times P'X = 0$$

$$\Rightarrow \quad A_{6\times 4}X = 0_{6\times 1}$$

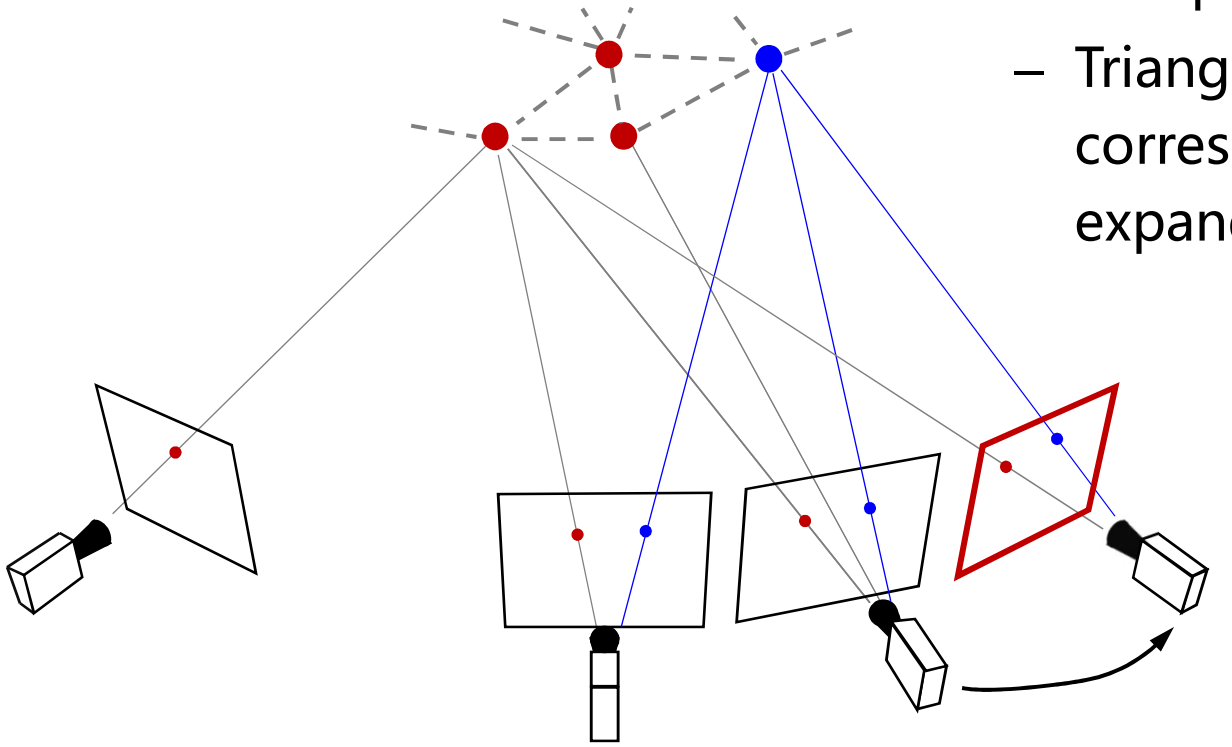
$$u = \frac{P_1^T X}{P_3^T X} \quad v = \frac{P_2^T X}{P_3^T X}$$
$$u' = \frac{P_1'^T X}{P_3'^T X} \quad v' = \frac{P_2'^T X}{P_3'^T X}$$

$$\Rightarrow \quad A_{4\times 3}x = b_{4\times 1}$$

Initialization | Pose estimation | Triangulation | Bundle adjustment

Incremental SfM

- **Repeat the above steps**

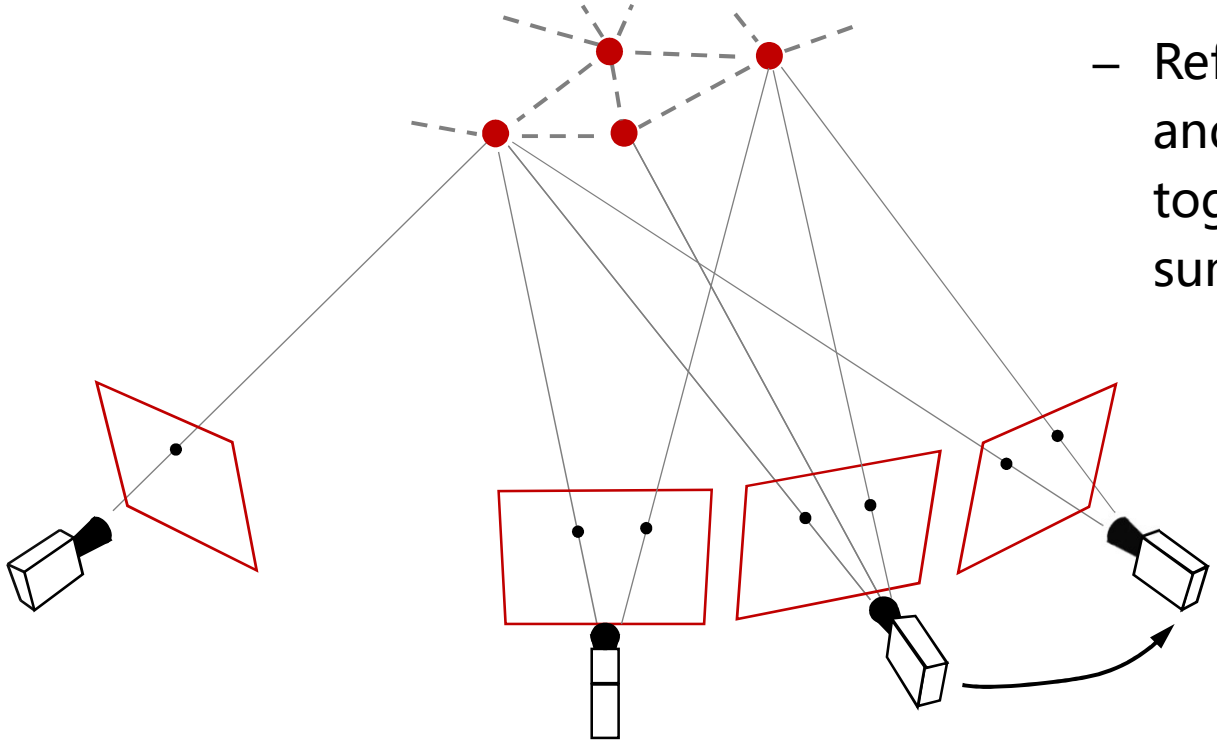


– Run pose estimation again
– Triangulate more feature correspondences to expand the 3D point cloud

Initialization | Pose estimation | Triangulation | Bundle adjustment

Incremental SfM

- **Call bundle adjustment from time to time to refine the result**



– Refine the 3D coordinates and the camera poses together by minimizing the sum of re-projection errors:

$$\sum_{ij} \boldsymbol{z}_{ij}^T \Sigma_{im}^{-1} \boldsymbol{z}_{ij}$$

Initialization | Pose estimation | Triangulation | Bundle adjustment

Bundle Adjustment

- Bundle adjustment is to refine the 3D coordinates and the camera poses together by minimizing the sum of re-projection errors:

$$\sum_{ij} \boldsymbol{z}_{ij}^T \Sigma_{im}^{-1} \boldsymbol{z}_{ij}$$

- $\boldsymbol{Z}_{ij}$ is the re-projection error of j-th point in the i-th view, defined as

$$\boldsymbol{z}_{ij} = \boldsymbol{m}_{ij} - \boldsymbol{x}_{ij} = \boldsymbol{m}_{ij} - \mathcal{P}(\theta_i, \boldsymbol{X}_j)$$

- $\Sigma_{im} \in \mathbb{R}^{2\times 2}$ represents the covariance of observation noises (1~4 pixels)

$$\Sigma_{im} \in \begin{bmatrix} \sigma_{im}^2 & 0 \\ 0 & \sigma_{im}^2 \end{bmatrix}$$

Bundle Adjustment

- This is a nonlinear least squares problem, where the parameter vector is

$$\Theta = \begin{bmatrix} \theta_1 \\ \vdots \\ \theta_M \\ \boldsymbol{X}_1 \\ \vdots \\ \boldsymbol{X}_N \end{bmatrix} = \begin{bmatrix} \mathbf{R}_1 \\ \boldsymbol{t}_1 \\ \vdots \\ \mathbf{R}_M \\ \boldsymbol{t}_M \\ \boldsymbol{X}_1 \\ \vdots \\ \boldsymbol{X}_N \end{bmatrix}$$

$$\Theta \leftarrow \Theta \boxplus \Delta\Theta$$

$$\mathbf{R}_i \leftarrow \mathbf{R}_i \oplus \exp(\Delta\theta^\wedge)$$

$$\boldsymbol{t}_i \leftarrow \boldsymbol{t}_i + \Delta\boldsymbol{t}_i$$

$$\boldsymbol{X}_i \leftarrow \boldsymbol{X}_i + \Delta\boldsymbol{X}_i$$

Bundle Adjustment

- The normal equation becomes very large to be solved

Gauss-Newton: $\mathbf{H}^{\mathrm{T}}\mathbf{H}\Delta\boldsymbol{x} = \mathbf{H}^{\mathrm{T}}\boldsymbol{r}$

Levenberg-Marquardt: $(\mathbf{H}^{\mathrm{T}}\mathbf{H} + \lambda\mathbf{I})\Delta\boldsymbol{x} = \mathbf{H}^{\mathrm{T}}\boldsymbol{r}$

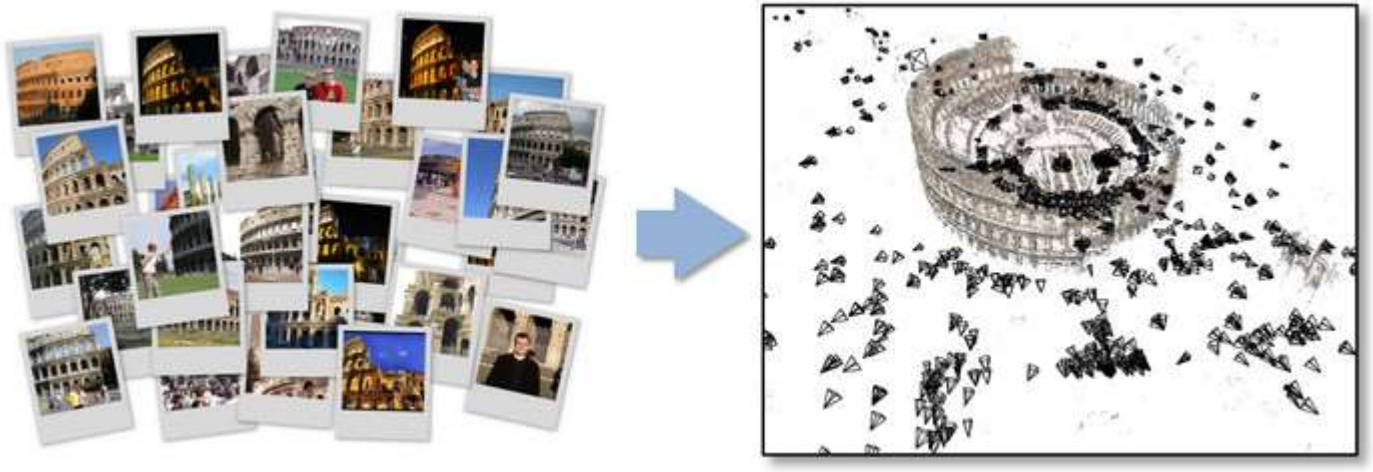
- For example, $N = 1000, M = 10$. The Jacobian matrix becomes

$$\mathbf{H} \in \mathbb{R}^{2MN\times(6N+3M)}$$

$$\mathbb{R}^{20000\times 6030}$$

SfM systems

- Bundler (2006,2007 Noah Snavely)



c++ open source

https://www.cs.cornell.edu/~snavely/bundler/

SfM systems

- Photo tourism / Modeling 3D world from massive photo collection

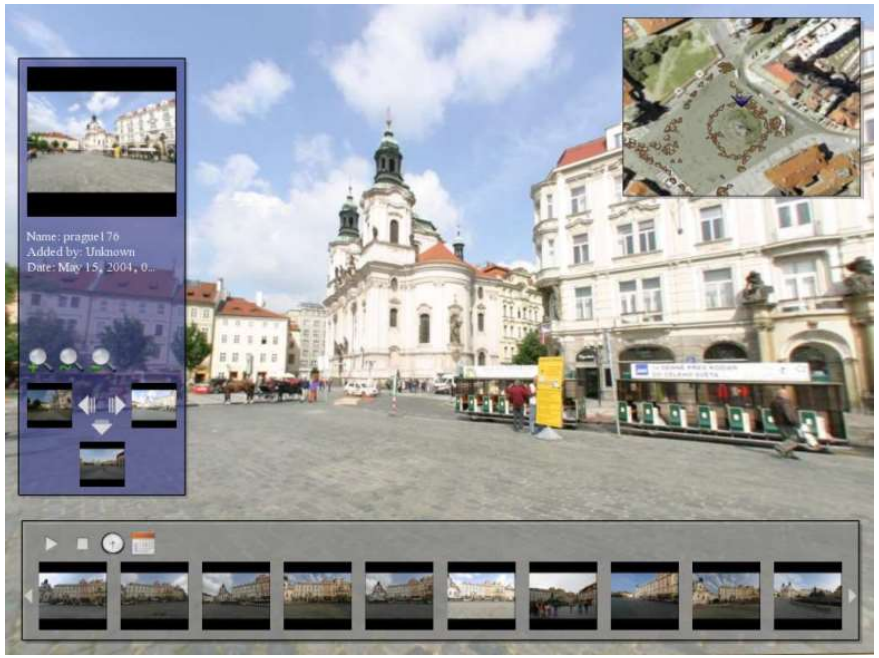


Photo Tourism

Exploring photo collections in 3D

Noah Snavely Steven M. Seitz Richard Szeliski

University of Washington *Microsoft Research*

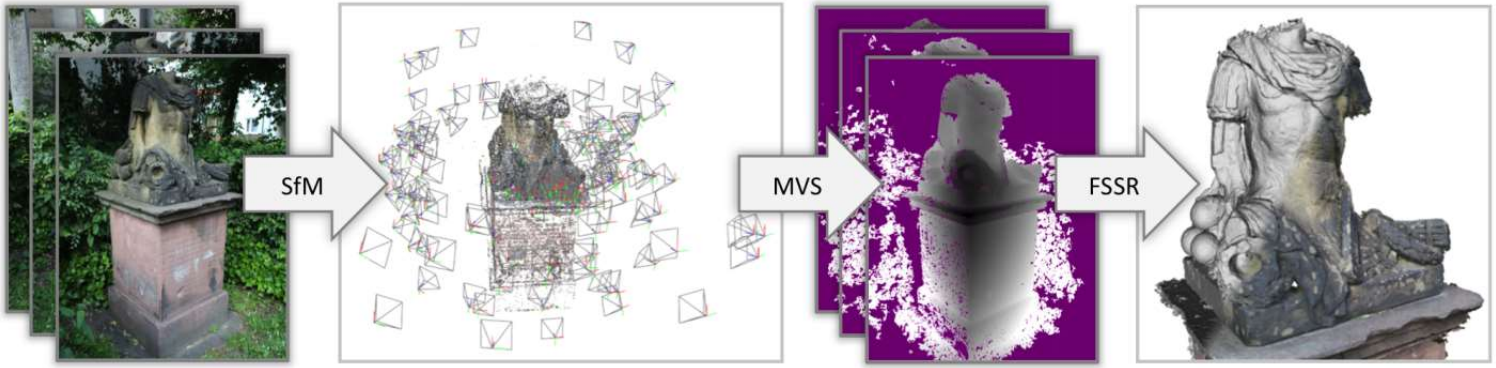
SIGGRAPH 2006

SfM systems

- Other implementations:
  - Visual SfM Changchang Wu(http://ccwu.me/vsfm/)

  - MVE (Multi-View Reconstruction Environment)

    2014, Simon Fuhrmann
    (https://github.com/simonfuhrmann/mve/wiki/MVE-Users-Guide)



SfM systems

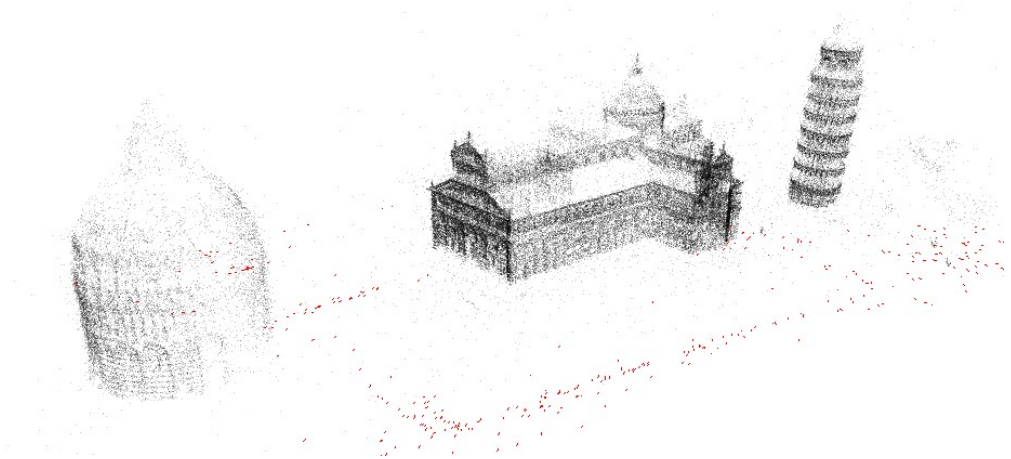
- OpenMVG

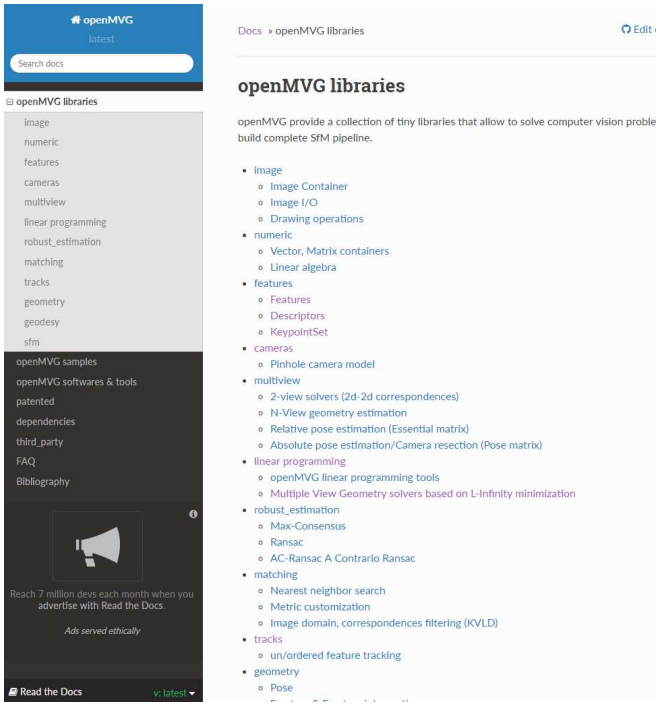
  Pierre Moulon

  (https://libraries.io/github/openMVG/openMVG)

- Theia Vision Library

  (http://www.theia-sfm.org/sfm.html)





SfM systems

- Matlab 2018b – Computer vision system toolbox (jbox.sjtu.edu.cn)
  - Camera Calibration and 3-D Vision

bundleAdjustment	Refine camera poses and 3-D points
cameraMatrix	Camera projection matrix
cameraPoseToExtrinsics	Convert camera pose to extrinsics
epipolarLine	Compute epipolar lines for stereo images
estimateCameraParameters	Calibrate a single or stereo camera
estimateEssentialMatrix	Estimate essential matrix from corresponding points in a pair of images
estimateFundamentalMatrix	Estimate fundamental matrix from corresponding points in stereo images
estimateWorldCameraPose	Estimate camera pose from 3-D to 2-D point correspondences
extrinsics	Compute location of calibrated camera
extrinsicsToCameraPose	Convert extrinsics to camera pose
isEpipoleInImage	Determine whether image contains epipole
lineToBorderPoints	Intersection points of lines in image and image border
relativeCameraPose	Compute relative rotation and translation between camera poses
triangulate	3-D locations of undistorted matching points in stereo images
triangulateMultiview	3-D locations of undistorted points matched across multiple images
undistortImage	Correct image for lens distortion
undistortPoints	Correct point coordinates for lens distortion
cameraParameters	Object for storing camera parameters

Summary

- Structure from motion problem is **solved** for static scenes (around 2006) .
- Lots of SfM software are available nowadays, both in open source and commercial products.
- SfM system is the first step to reconstruct the dense 3D model of the world.
  - Sparse 3D points
  - Camera poses -> Dense 3D points (Multi-view stereo)
- SfM technique can be applied to SLAM problem (Next week)