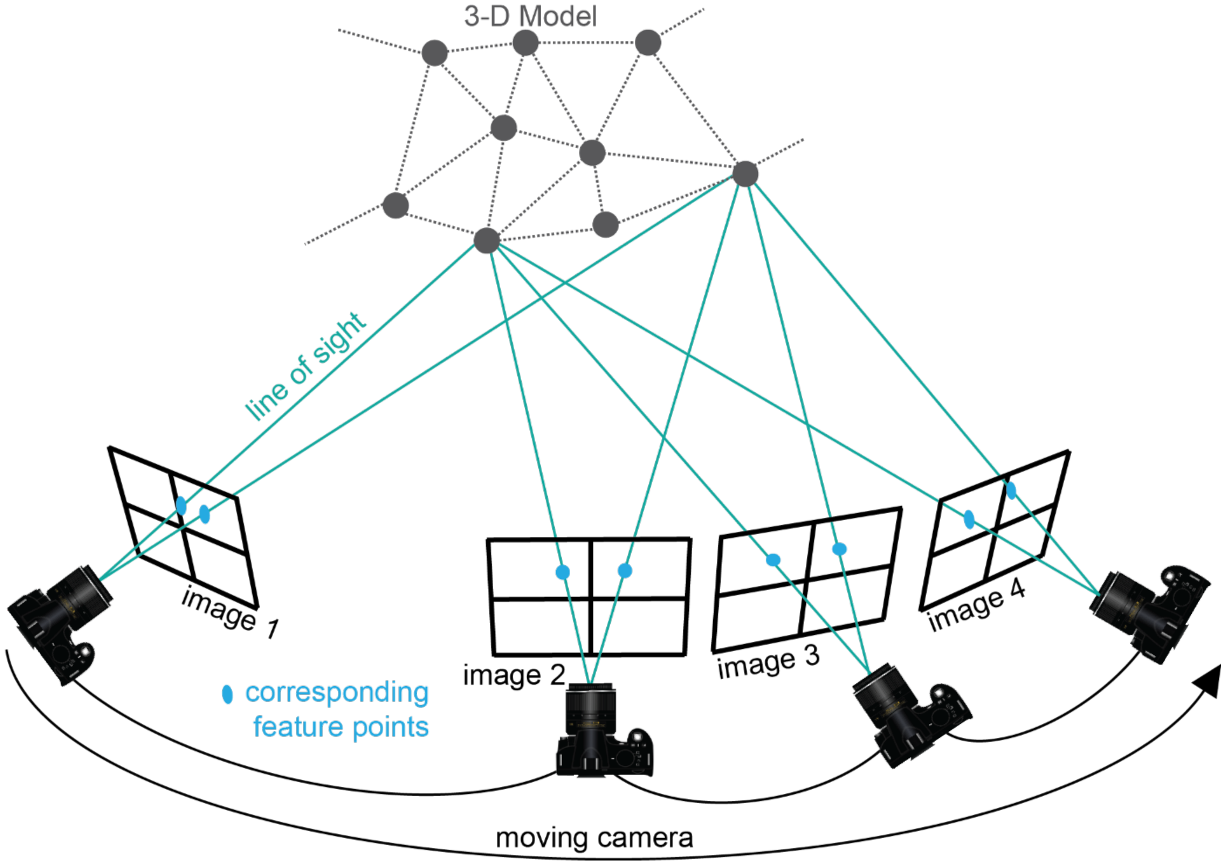
Shape/structure from motion

Václav Hlaváč
Czech Technical University in Prague
Czech Institute of Informatics, Robotics and Cybernetics
166 36 Prague 6, Jugoslávských partyzánů 1580/3, Czech Republic
http://people.ciirc.cvut.cz/hlavac; vaclav.hlavac@cvut.cz

Courtesy: T. Pajdla, R. Šára, D. Hoeim, J. Hays, J. Xiao,

SfM: What it is?

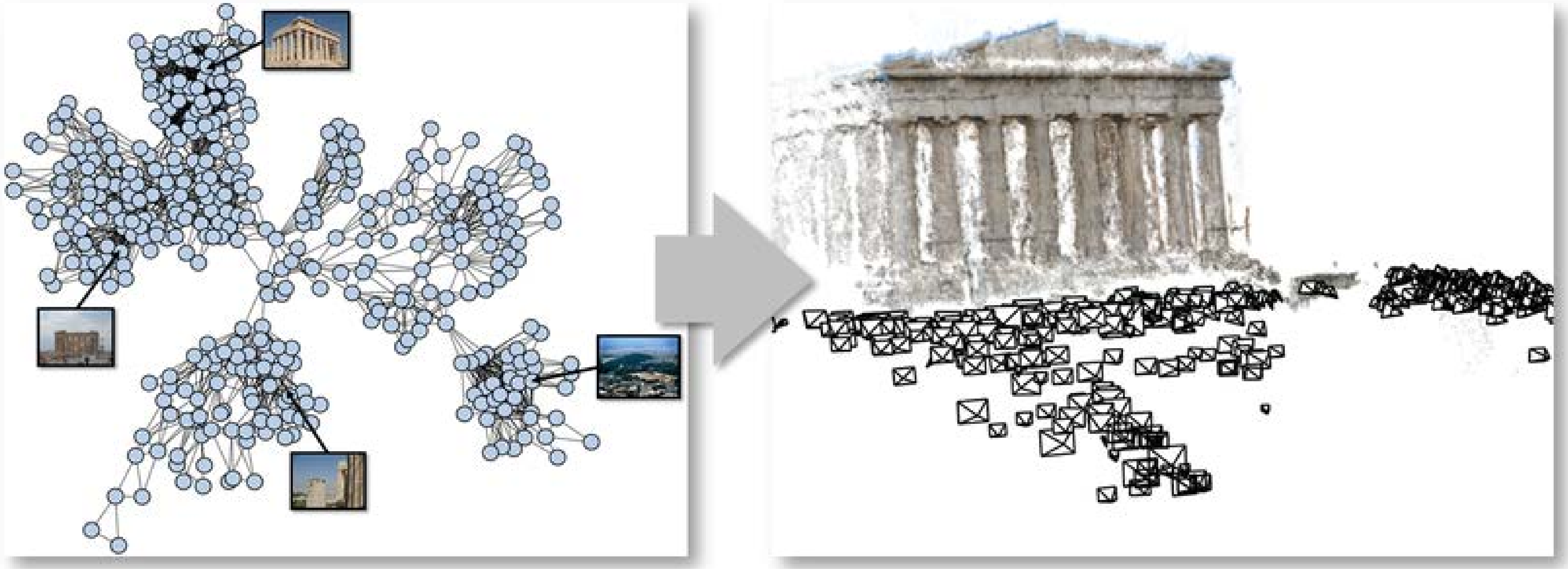
Structure/shape from motion (SfM) is a photogrammetric technique that uses overlapping images to construct a 3D model of the scene.



SfM: A typical pipeline

Track 2D Features → Estimate 3D → Optimize (Bundle Adjust) → Fit Surfaces

Reconstructing the world from Internet Photos



What can we compare SfM to?

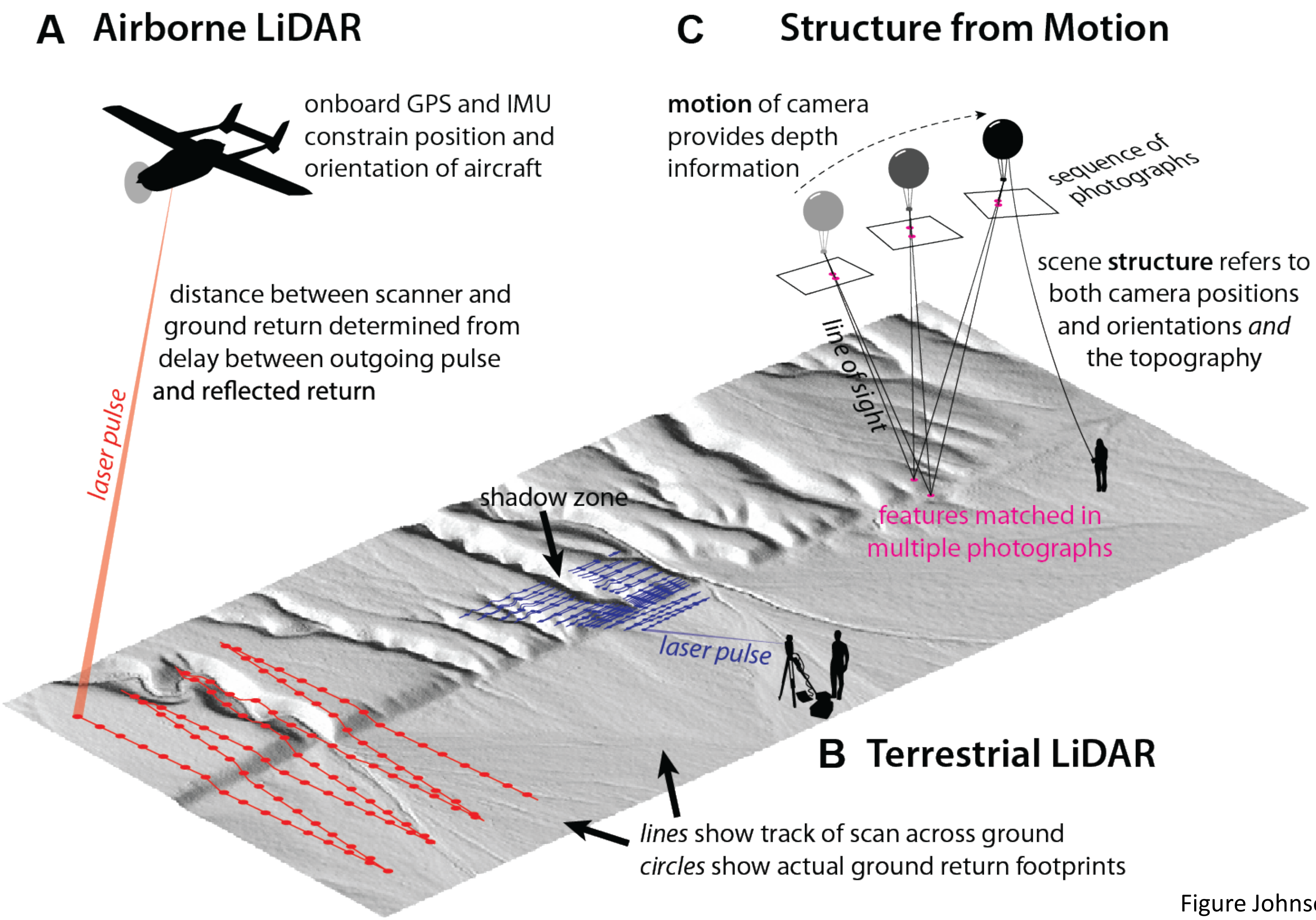
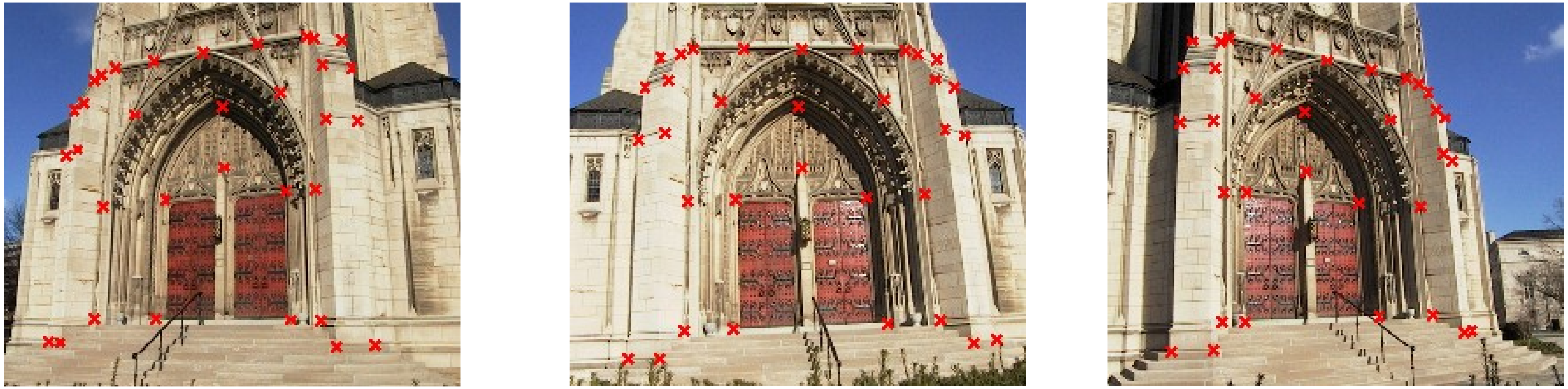


Figure Johnson 2014

SfM – Initial notes

- For now, static scene and moving camera. Equivalently, rigidly moving scene and static camera.
- Limiting case of stereo with many cameras.
- Limiting case of multiview camera calibration with unknown target.
- Given n points, $n \geq 5$ and N camera views/positions, have $2nN$ equations and $3n+6N$ unknown.
- We always like to use many more points and views.

Structure from motion (1)



Step 1: Track Features

- Detect good features
  - corners, line segments
- Find correspondences between frames
  - Lucas & Kanade-style motion estimation
  - window-based correlation

Structure from motion (2)

Step 2: Estimate motion and structure

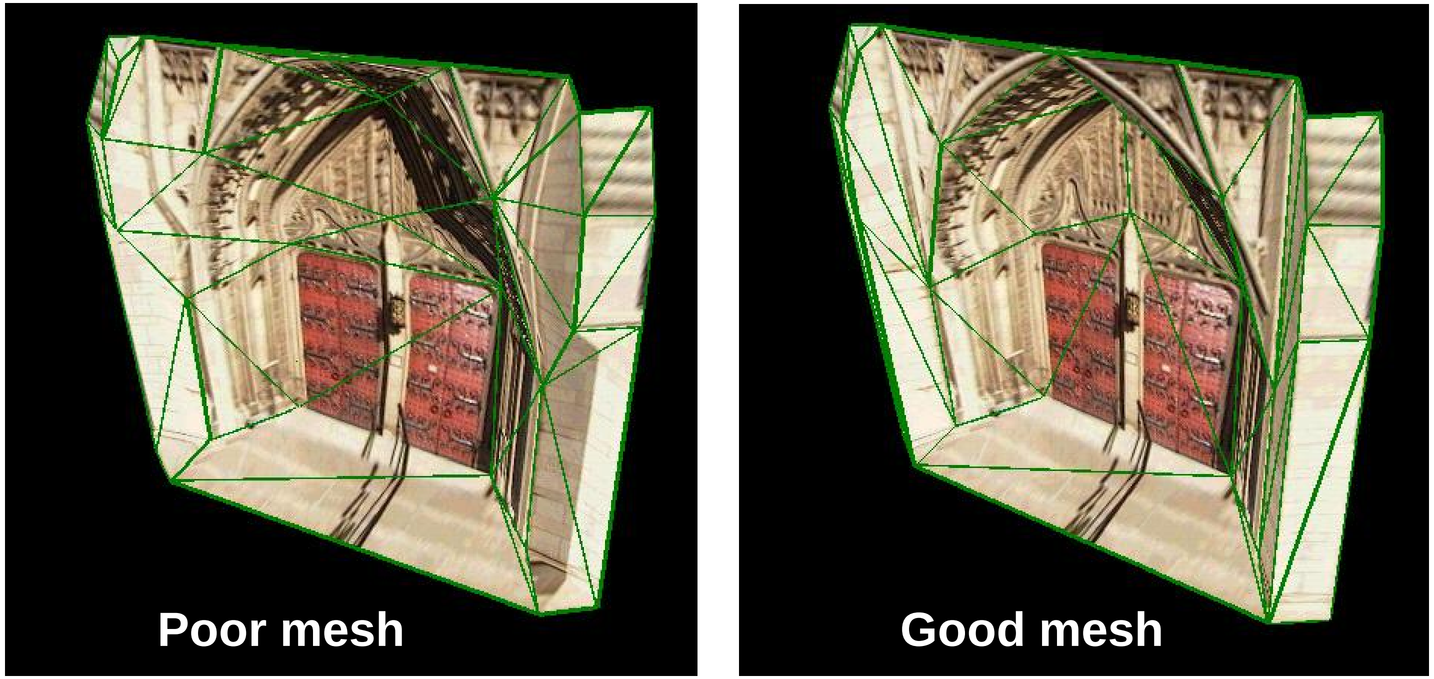
- Simplified projection model, e.g., [Tomasi 92]
- 2 or 3 views at a time [Hartley 00]

$$\underbrace{\begin{bmatrix} \mathbf{I}_1 \\ \mathbf{I}_2 \\ \vdots \\ \mathbf{I}_f \end{bmatrix}}_{\text{Images}} = \underbrace{\begin{bmatrix} \mathbf{\Pi}_1 \\ \mathbf{\Pi}_2 \\ \vdots \\ \mathbf{\Pi}_f \end{bmatrix}}_{\text{Motion}} \underbrace{\begin{bmatrix} \mathbf{X}_1 & \mathbf{X}_2 & \cdots & \mathbf{X}_n \end{bmatrix}}_{\text{Structure}}$$

Step 3: Refine estimates

- "Bundle adjustment" in photogrammetry; reprojection error minimization

Structure from motion (3)



Morris and Kanade, 2000

Step 4: Recover Surfaces

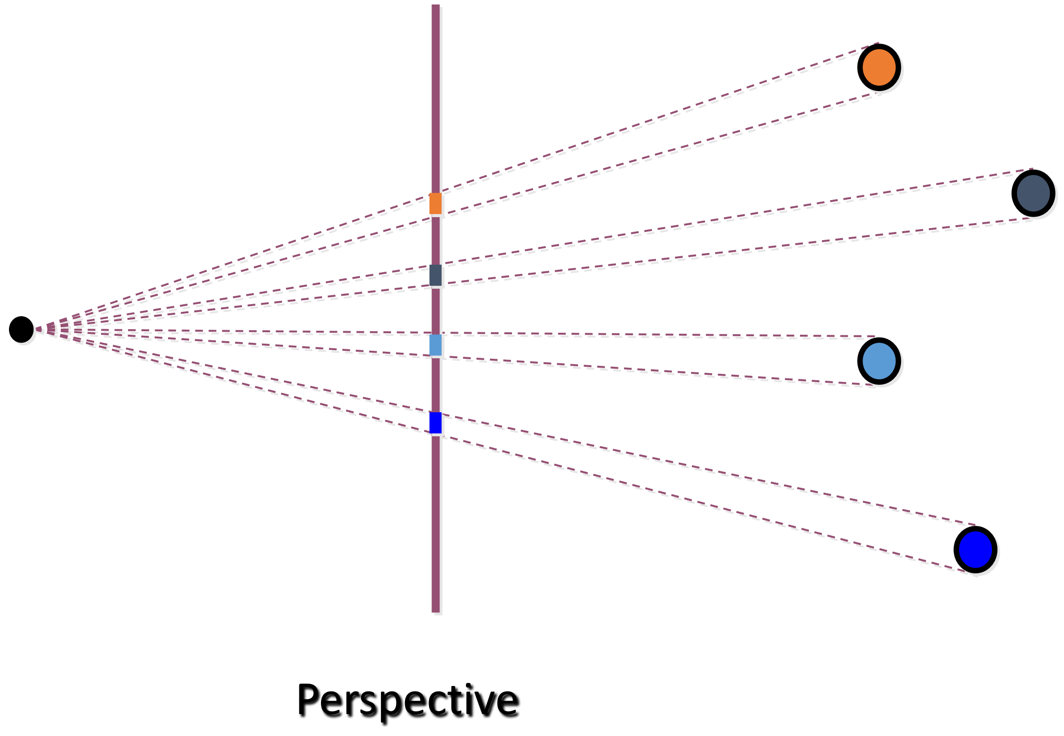
- Image-based triangulation [Morris 00, Baillard 99]
- Silhouettes [Fitzgibbon 98]
- Stereo [Pollefeys 99]

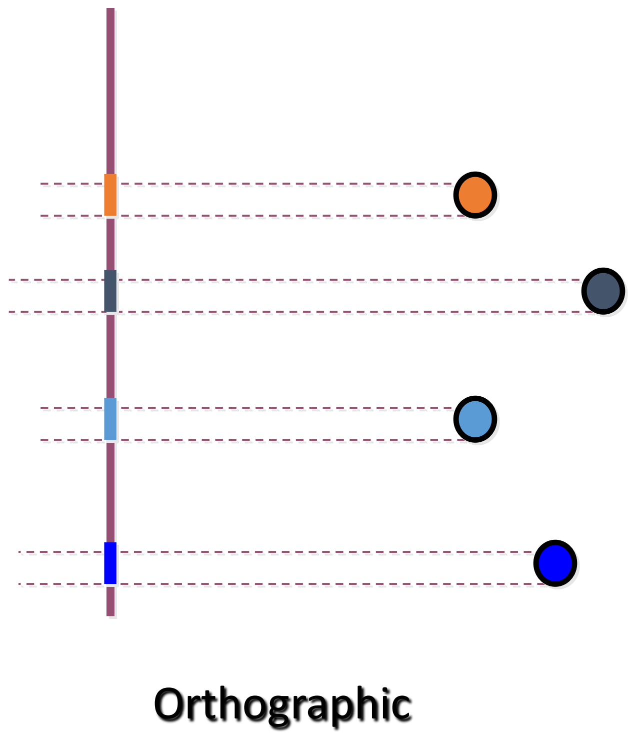
Approaches to SfM

- Obtaining point correspondences
  - Optical flow
  - Stereo methods: correlation, feature matching
- Solving for points and camera motion
  - Nonlinear minimization (bundle adjustment)
  - Various approximations…

Orthographic approximation

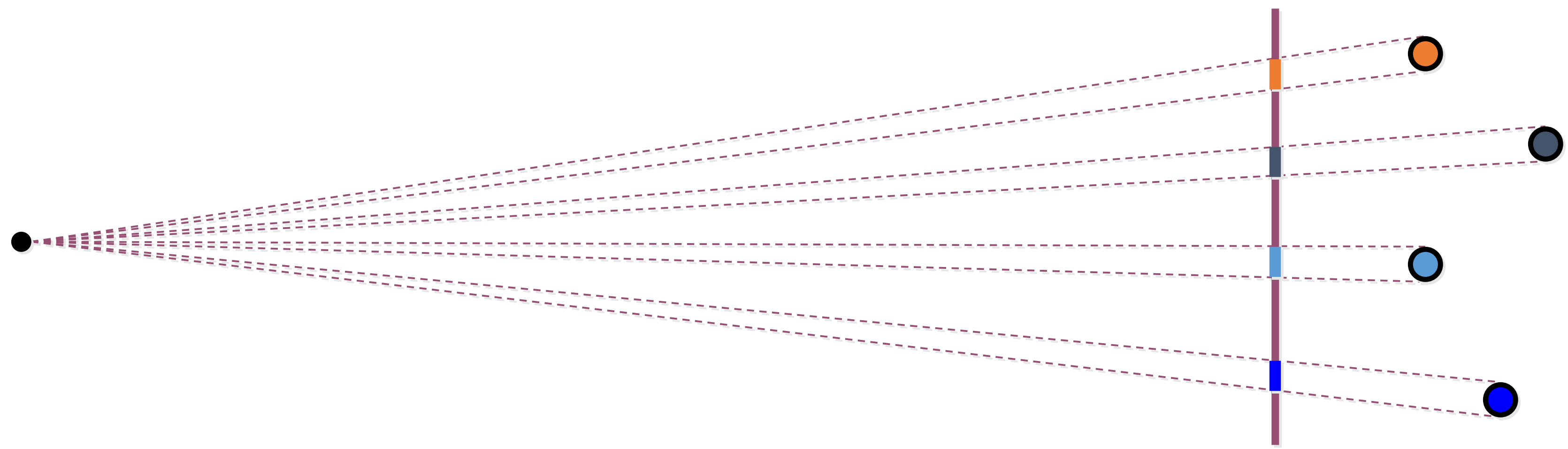
Simplest SFM case: camera approximated by orthographic projection





Weak perspective

An orthographic assumption is sometimes well approximated by a telephoto lens



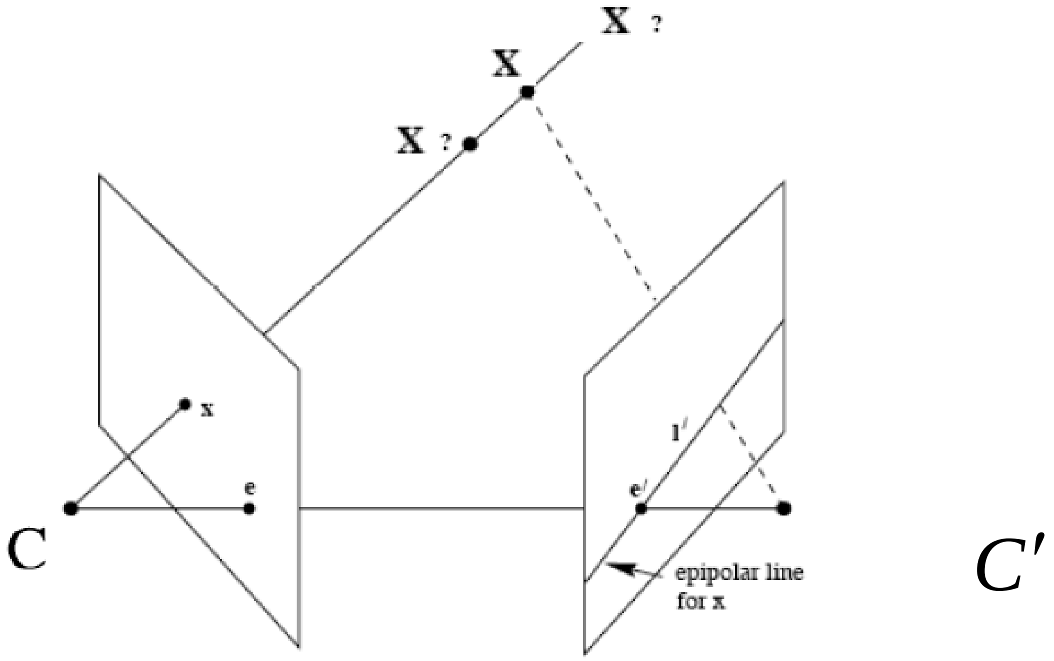
Weak Perspective

Consequences of orthographic projection

- Scene can be recovered up to scale
- Translation perpendicular to image plane can never be recovered

Recap: Epipoles

- Point x in left image corresponds to **epipolar line** l' in right image
- Epipolar line passes through the epipole (the intersection of the cameras' baseline with the image plane



Recap: Fundamental matrix

- Fundamental matrix maps from a point in one image to a line in the other

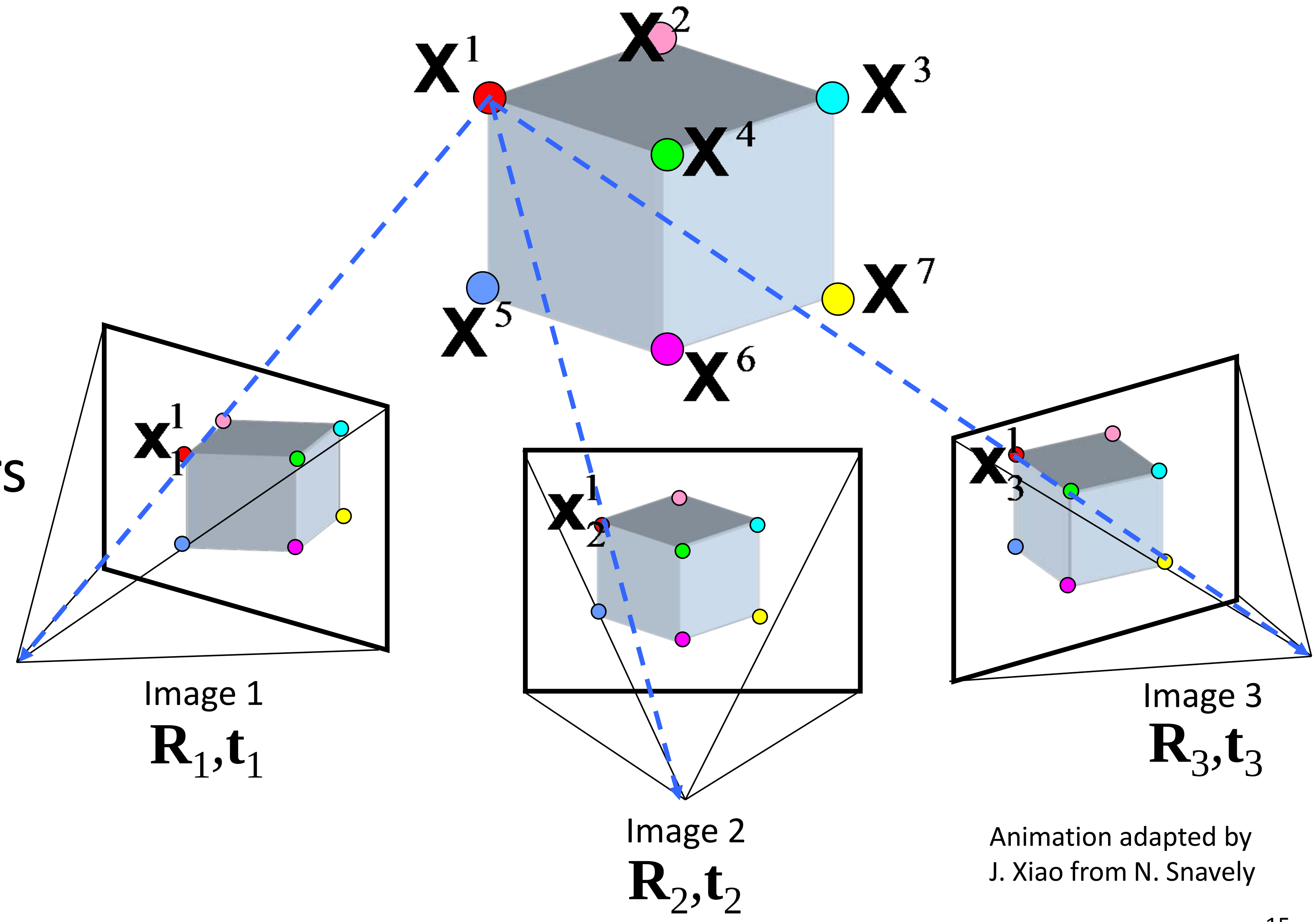
$$\mathbf{l}' = \mathbf{F}\mathbf{x} \qquad \mathbf{l} = \mathbf{F}^\top \mathbf{x}'$$

- If x and x’ correspond to the same 3d point X:

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

Structure from motion

Given a set of corresponding points in two or more images, compute the camera parameters and the 3D point coordinates



Animation adapted by J. Xiao from N. Snavely

Structure from motion ambiguity

- If we scale the entire scene by some factor *k* and, at the same time, scale the camera matrices by the factor of 1/*k*, the projections of the scene points in the image remain exactly the same:

$$\mathbf{x} = \mathbf{P}\mathbf{X} = \left(\frac{1}{k}\mathbf{P}\right)(k\mathbf{X})$$

- It is impossible to recover the absolute scale of the scene!

How do we know the scale of image content?

Projection matrix

- Perspective projection:

$$\begin{bmatrix} u_i \\ v_i \\ 1 \end{bmatrix} = \begin{bmatrix} f_x & \alpha & u_0 \\ 0 & f_y & v_0 \\ 0 & 0 & 1 \end{bmatrix} \bullet \begin{bmatrix} r_1^T & t_1 \\ r_2^T & t_2 \\ r_3^T & t_3 \end{bmatrix} \bullet \begin{bmatrix} x_i \\ y_i \\ z_i \\ 1 \end{bmatrix}$$

R T P

- 2D coordinates are just a nonlinear function of its 3D coordinates and camera parameters:

$$u_i = \frac{(f_x r_1^T + \alpha r_2^T + u_0 r_3^T) \bullet P + f_x t_1 + \alpha t_2 + u_0 t_3}{r_3^T \bullet P + t_3} \leftarrow f(K,R,T;P_i)$$

$$v_i = \frac{(f_y r_2^T + v_0 r_3^T) \bullet P + f_y t_2 + t_3}{r_3^T \bullet P + t_3} \leftarrow g(K,R,T;P_i)$$

Nonlinear approach for SFM

What's the difference between camera calibration and SFM?

- Camera calibration: known 3D and 2D

$$\arg\min_{K,\{R_j\},\{T_j\}} \sum_{j=1}^{M}\sum_{i=1}^{N} (u_i^j - f(K,R_j,T_j,P_i))^2 + (v_i^j - g(K,R_j,T_j,P_i))^2$$

- SFM: unknown 3D and known 2D

$$\arg\min_{\{P_i\},K,\{R_j\},\{T_j\}} \sum_{j=1}^{M}\sum_{i=1}^{N} (u_i^j - f(K,R_j,T_j,P_i))^2 + (v_i^j - g(K,R_j,T_j,P_i))^2$$

- What's 3D-to-2D registration problem?

Count # of constraints vs # of unknowns

$$\arg\min_{\{P_i\},K,\{R_j\},\{T_j\}} \sum_{j=1}^{M}\sum_{i=1}^{N}(u_i^j - f(K,R_j,T_j,P_i))^2 + (v_i^j - g(K,R_j,T_j,P_i))^2$$

- *M* camera poses
- *N* points
- *2MN* point constraints
- 6*M*+3*N* unknowns (known intrinsic camera parameters)
- Suggests: need $2mn \geq 6m + 3n$
- But: Can we *really* recover all parameters???
  - Can't recover origin, orientation (6 params)
  - Can't recover scale (1 param)
- Thus, we need $2mn \geq 6m + 3n - 7$

SFM: Bundle adjustment

$$\arg\min_{\{P_i\},K,\{R_j\},\{T_j\}} \sum_{j=1}^{M}\sum_{i=1}^{N}(u_i^j - f(K,R_j,T_j,P_i))^2 + (v_i^j - g(K,R_j,T_j,P_i))^2$$

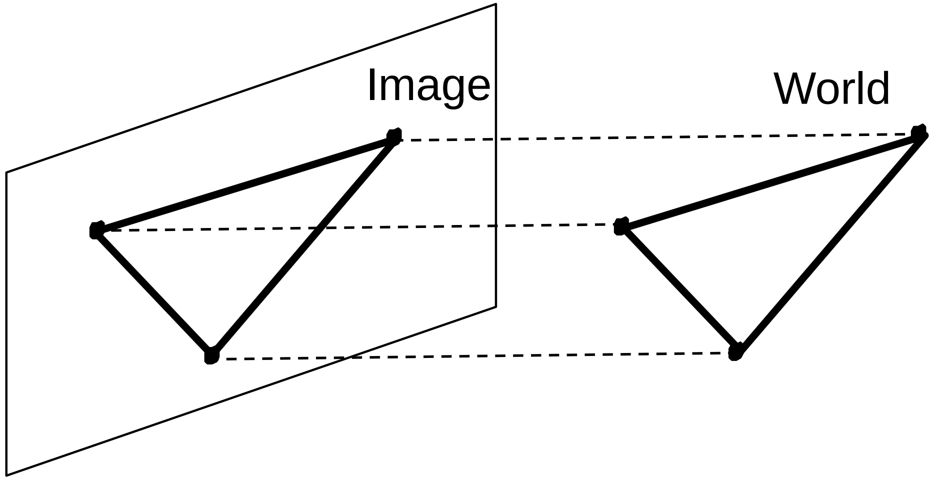
- SFM = Nonlinear Least Squares problem
- Minimize through
  - Gradient Descent
  - Conjugate Gradient
  - Gauss-Newton
  - Levenberg Marquardt common method

- Prone to local minima

Are we done?

No, bundle adjustment has many local minima.

SFM using factorization (1)



$$\begin{bmatrix} u_i \\ v_i \end{bmatrix} = \begin{bmatrix} r_1^T & t_1 \\ r_2^T & t_2 \end{bmatrix} \bullet \begin{bmatrix} x_i \\ y_i \\ z_i \\ 1 \end{bmatrix}$$

Subtract the mean

$$\begin{bmatrix} u_i - \dfrac{\sum_{i=1}^{N} u_i}{N} \\ v_i - \dfrac{\sum_{i=1}^{N} v_i}{N} \end{bmatrix} = \begin{bmatrix} r_1^T \\ r_2^T \end{bmatrix} \bullet \begin{bmatrix} x_i \\ y_i \\ z_i \end{bmatrix}$$

SFM using factorization (2)

Stack all the features from the same frame:

$$\begin{bmatrix} \tilde{u}_1 \, \tilde{u}_2 \ldots \tilde{u}_N \\ \tilde{v}_1 \, \tilde{v}_2 \ldots \tilde{v}_N \end{bmatrix} = \begin{bmatrix} r_1^T \\ r_2^T \end{bmatrix} \bullet \begin{bmatrix} x_1 \; x_2 \ldots x_N \\ y_1 \; y_2 \cdots y_N \\ z_1 \; z_2 \ldots z_N \end{bmatrix}$$

Stack all the features from all the images:

$$\underbrace{\begin{bmatrix} \tilde{u}_{F,1} \, \tilde{u}_{F,2} \ldots \tilde{u}_{F,N} \\ \tilde{v}_{F,1} \, \tilde{v}_{F,2} \ldots \tilde{v}_{F,N} \\ \vdots \\ \tilde{u}_{F,1} \, \tilde{u}_{F,2} \ldots \tilde{u}_{F,N} \\ \tilde{v}_{F,1} \, \tilde{v}_{F,2} \ldots \tilde{v}_{F,N} \end{bmatrix}}_{\tilde{W}_{2F\times N}} = \underbrace{\begin{bmatrix} r_{1,1}^T \\ r_{1,2}^T \\ \vdots \\ r_{F,1}^T \\ r_{F,2}^T \end{bmatrix}}_{M_{2F\times 3}} \bullet \underbrace{\begin{bmatrix} x_1 \; x_2 \ldots x_N \\ y_1 \; y_2 \ldots y_N \\ z_1 \; z_2 \ldots z_N \end{bmatrix}}_{S_{3\times N}}$$

SFM using factorization (3)

Stack all the features from all the images:

$$\underbrace{\begin{bmatrix} \tilde{u}_{F,1}\ \tilde{u}_{F,2} \dots \tilde{u}_{F,N} \\ \tilde{v}_{F,1}\ \tilde{v}_{F,2} \dots \tilde{v}_{F,N} \\ \vdots \\ \tilde{u}_{F,1}\ \tilde{u}_{F,2} \dots \tilde{u}_{F,N} \\ \tilde{v}_{F,1}\ \tilde{v}_{F,2} \dots \tilde{v}_{F,N} \end{bmatrix}}_{\tilde{W}_{2F\times N}} = \underbrace{\begin{bmatrix} r_{1,1}^T \\ r_{1,2}^T \\ \vdots \\ r_{F,1}^T \\ r_{F,2}^T \end{bmatrix}}_{M_{2F\times 3}} \bullet \underbrace{\begin{bmatrix} x_1\ x_2 \dots x_N \\ y_1\ y_2 \dots y_N \\ z_1\ z_2 \dots z_N \end{bmatrix}}_{S_{3\times N}}$$

Factorize the matrix $\tilde{W}_{2F\times N}$ into two matrix using SVD:

$$\tilde{W}_{2F\times N} = U\Sigma V^T \qquad \tilde{M}_{2F\times 3} = U\Sigma^{\frac{1}{2}} \qquad \tilde{S}_{3\times N} = \Sigma^{\frac{1}{2}} V^T$$

SFM using factorization (4)

Stack all the features from all the images:

$$\underbrace{\begin{bmatrix} \tilde{u}_{F,1} & \tilde{u}_{F,2} & \dots & \tilde{u}_{F,N} \\ \tilde{v}_{F,1} & \tilde{v}_{F,2} & \dots & \tilde{v}_{F,N} \\ & \vdots & & \\ \tilde{u}_{F,1} & \tilde{u}_{F,2} & \dots & \tilde{u}_{F,N} \\ \tilde{v}_{F,1} & \tilde{v}_{F,2} & \dots & \tilde{v}_{F,N} \end{bmatrix}}_{\tilde{W}_{2F\times N}} = \underbrace{\begin{bmatrix} r_{1,1}^T \\ r_{1,2}^T \\ \vdots \\ r_{F,1}^T \\ r_{F,2}^T \end{bmatrix}}_{M_{2F\times 3}} \bullet \underbrace{\begin{bmatrix} x_1 & x_2 & \dots & x_N \\ y_1 & y_2 & \dots & y_N \\ z_1 & z_2 & \dots & z_N \end{bmatrix}}_{S_{3\times N}}$$

Factorize the matrix $\tilde{W}_{2F\times N}$ into two matrices using SVD:

$$\tilde{W}_{2F\times N} = U\Sigma V^T \qquad \tilde{M}_{2F\times 3} = U\Sigma^{\frac{1}{2}} \qquad \tilde{S}_{3\times N} = \Sigma^{\frac{1}{2}} V^T$$

$$M_{2F\times 3} = \tilde{M}_{2F\times 3} Q_{3\times 3} \qquad S_{3\times N} = Q_{3\times 3}^{-1} \tilde{S}_{3\times N}$$

Is the solution unique? No! How to compute the matrix $Q_{3\times 3}$?

SFM using factorization (5)

$$\begin{bmatrix} \tilde{u}_{F,1} \; \tilde{u}_{F,2} \dots \tilde{u}_{F,N} \\ \tilde{v}_{F,1} \; \tilde{v}_{F,2} \dots \tilde{v}_{F,N} \\ \vdots \\ \tilde{u}_{F,1} \; \tilde{u}_{F,2} \dots \tilde{u}_{F,N} \\ \tilde{v}_{F,1} \; \tilde{v}_{F,2} \dots \tilde{v}_{F,N} \end{bmatrix} = \begin{bmatrix} r_{1,1}^T \\ r_{1,2}^T \\ \vdots \\ r_{F,1}^T \\ r_{F,2}^T \end{bmatrix} \bullet \begin{bmatrix} x_1 \; x_2 \dots x_N \\ y_1 \; y_2 \dots y_N \\ z_1 \; z_2 \dots z_N \end{bmatrix}$$

$$\tilde{W}_{2F\times N} \qquad M_{2F\times 3} \qquad S_{3\times N}$$

$$\tilde{W}_{2F\times N} = U\Sigma V^T \qquad \tilde{M}_{2F\times 3} = U\Sigma^{\frac{1}{2}} \qquad \tilde{S}_{3\times N} = \Sigma^{\frac{1}{2}} V^T$$

$$M_{2F\times 3} = \tilde{M}_{2F\times 3} Q_{3\times 3} \qquad S_{3\times N} = Q_{3\times 3}^{-1} \tilde{S}_{3\times N}$$

$$Q_{3\times 3}$$

SFM using factorization (6)

M is the stack of rotation matrix:

$$M_{2F\times 3} M_{2F\times 3}^T = \begin{bmatrix} r_{1,1}^T \\ r_{1,2}^T \\ \vdots \\ r_{F,1}^T \\ r_{F,2}^T \end{bmatrix} \bullet \begin{bmatrix} r_{1,1} & r_{1,2} & \cdots & r_{F,2} & r_{F,2} \end{bmatrix}$$

Orthogonal constraints from rotation matrix

$$= \begin{bmatrix} r_{1,1}^T r_{1,1} & r_{1,1}^T r_{1,2} & * & * & * \\ r_{1,2}^T r_{1,1} & r_{1,2}^T r_{1,2} & * & * & * \\ \vdots & \vdots & \vdots & \vdots & \vdots \\ * & * & * & r_{F,1}^T r_{F,1} & r_{F,1}^T r_{F,2} \\ * & * & * & r_{F,2}^T r_{F,1} & r_{F,2}^T r_{F,2} \end{bmatrix}$$

$$= \tilde{M}_{2F\times 3} Q_{3\times 3} Q_{3\times 3}^T \tilde{M}_{2F\times 3}^T$$

SFM using factorization (7)

Orthogonal constraints from rotation matrices:

$$\tilde{M}_{2F\times 3} \boxed{Q_{3\times 3} Q_{3\times 3}^T} \tilde{M}_{2F\times 3}^T = \begin{bmatrix} r_{1,1}^T r_{1,1} = 1 & r_{1,1}^T r_{1,2} = 0 & * & * & * \\ r_{1,2}^T r_{1,1} = 0 & r_{1,2}^T r_{1,2} = 1 & * & * & * \\ \vdots & \vdots & \vdots & \vdots & \vdots \\ * & * & * & r_{F,1}^T r_{F,1} = 1 & r_{F,1}^T r_{F,2} = 0 \\ * & * & * & r_{F,2}^T r_{F,1} = 0 & r_{F,2}^T r_{F,2} = 1 \end{bmatrix}$$

QQ: symmetric 3 by 3 matrix

How to compute QQ^T?

Least square solution

- 4F linear constraints, 9 unknowns (6 independent due to symmetric matrix)

How to compute Q from QQ^T ?

SVD again: $QQ = U\Sigma V^T$ $\quad Q = U\Sigma^{\frac{1}{2}}$

SFM using factorization (8)

M is the stack of rotation matrix:

$$M_{2F\times3}M_{2F\times3}^{T}=\begin{bmatrix} r_{1,1}^{T} \\ r_{1,2}^{T} \\ \vdots \\ r_{F,1}^{T} \\ r_{F,2}^{T} \end{bmatrix} \bullet \begin{bmatrix} r_{1,1} & r_{1,2} & \cdots & r_{F,2} & r_{F,2} \end{bmatrix}$$

Orthogonal constraints
from rotation matrix

$$=\begin{bmatrix} r_{1,1}^{T}r_{1,1} = 1 & r_{1,1}^{T}r_{1,2} = 0 & * & * & * \\ r_{1,2}^{T}r_{1,1} = 0 & r_{1,2}^{T}r_{1,2} = 1 & * & * & * \\ \vdots & \vdots & \vdots & \vdots & \vdots \\ * & * & * & r_{F,1}^{T}r_{F,1} = 1 & r_{F,1}^{T}r_{F,2} = 0 \\ * & * & * & r_{F,2}^{T}r_{F,1} = 0 & r_{F,2}^{T}r_{F,2} = 1 \end{bmatrix}$$

$$=\tilde{M}_{2F\times3}\boxed{Q_{3\times3}Q_{3\times3}^{T}}\tilde{M}_{2F\times3}^{T}$$

QQ^T: symmetric 3 by 3 matrix

Computing QQ^T is easy:

- 3F linear equations

- 6 independent unknowns

SFM using factorization (8)

1. Form the measurement matrix $\tilde{W}_{2F \times N}$

2. Decompose the matrix into two matrices $\tilde{M}_{2F \times 3}$ and $\tilde{S}_{3 \times N}$ using SVD

3. Compute the matrix Q with least square and SVD

4. Compute the rotation matrix and shape matrix:

   $M = \tilde{M}_{2F \times 3} Q$ and $S = Q^{-1} \tilde{S}_{2F \times 3}$

SFM Summary

- Bundle adjustment (nonlinear optimization)
- - work with perspective camera model
- - work with incomplete data
- - prone to local minima

- Factorization:
- - closed-form solution for weak perspective camera
- - simple and efficient
- - usually need complete data
- - becomes complicated for full-perspective camera model

- Phil Torr's structure from motion toolkit in matlab (click here)

- Voodoo camera tracker (click here)