More SFM

D.A.Forsyth, UIUC

We have seen

- Cameras and pairs of cameras
  - also
    - some camera pair geometry
    - scene flow, optic flow, stereo reconstruction
- Structure from motion via factorization
  - for orthographic cameras
- SLAM (feature based and direct)

Borrowings from

Structure from Motion

Computer Vision

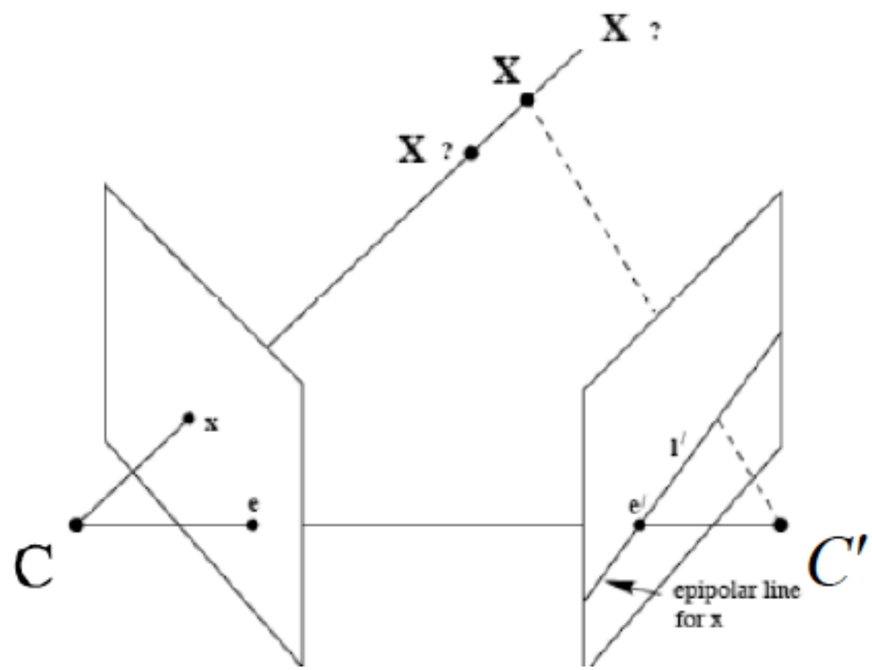
CS 543 / ECE 549

University of Illinois

Derek Hoiem

Last Class: Epipolar Geometry

- 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



Last Class: 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:

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

Incremental Structure from Motion (SfM)

Goal: Solve for camera poses and 3D points in scene

Incremental SfM

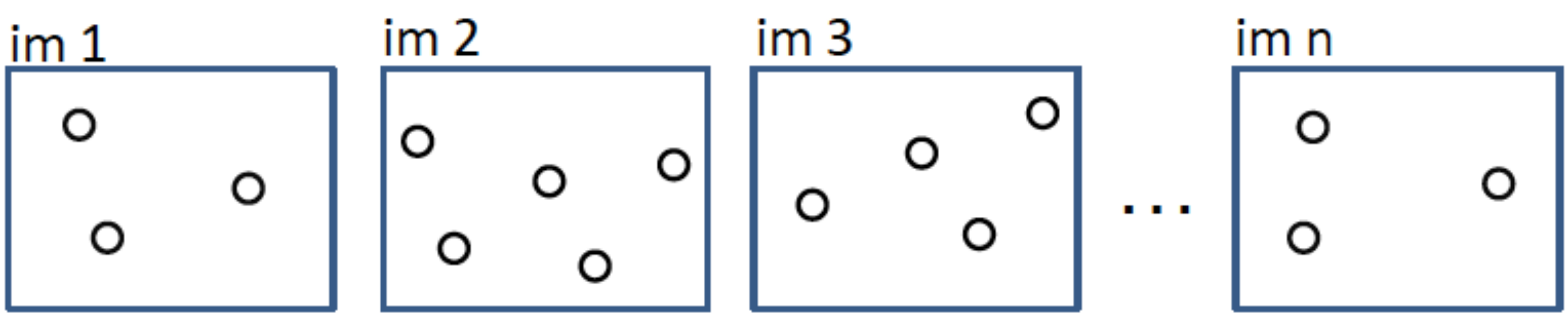
1. Compute features

2. Match images

3. Reconstruct
   a) Solve for pose and 3D points in two cameras
   b) Solve for pose of additional camera(s) that observe reconstructed 3D points
   c) Solve for new 3D points that are viewed in at least two cameras
   d) Bundle adjust to minimize reprojection error

Incremental SFM: **detect features**

- Feature types: SIFT, ORB, Hessian-Laplacian, …



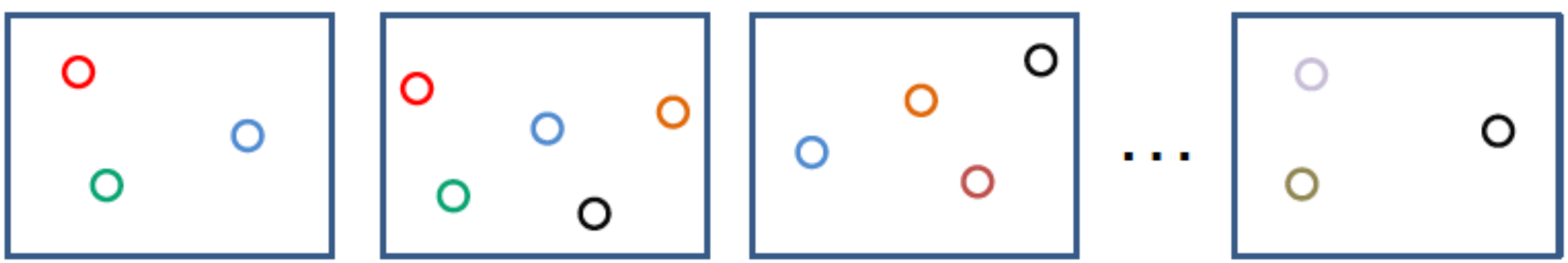
Each circle represents a set of detected features

Incremental SFM: **match features and images**

For each pair of images:

1. Match feature descriptors via approximate nearest neighbor
2. Solve for F and find inlier feature correspondences

- Speed tricks
  - Match only 100 largest features first
  - Use a bag-of-words method to find candidate matches
  - Perform initial filtering based on GPS coordinates, if available
  - Use known matches to predict new ones

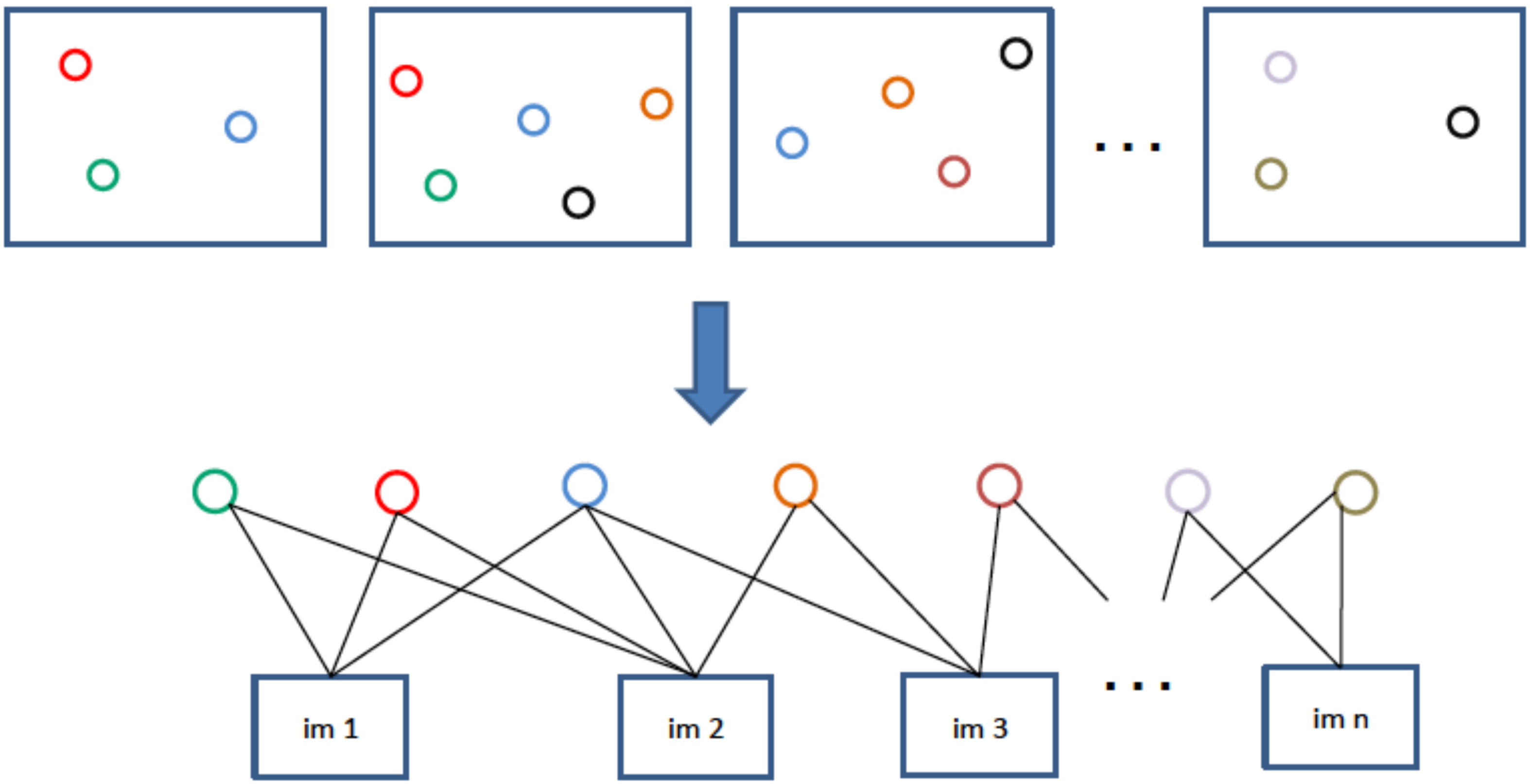


Points of same color have been matched to each other

Many pix = nasty issues

Figure 1. Result of Rome with 21K registered out of 75K images.

Incremental SFM: **create tracks graph**



tracks graph: bipartite graph between observed 3D points and images

Appearance matches may not be 3D points; recall F matrix, etc. At some point we will need to verify matches for geometric consistency.

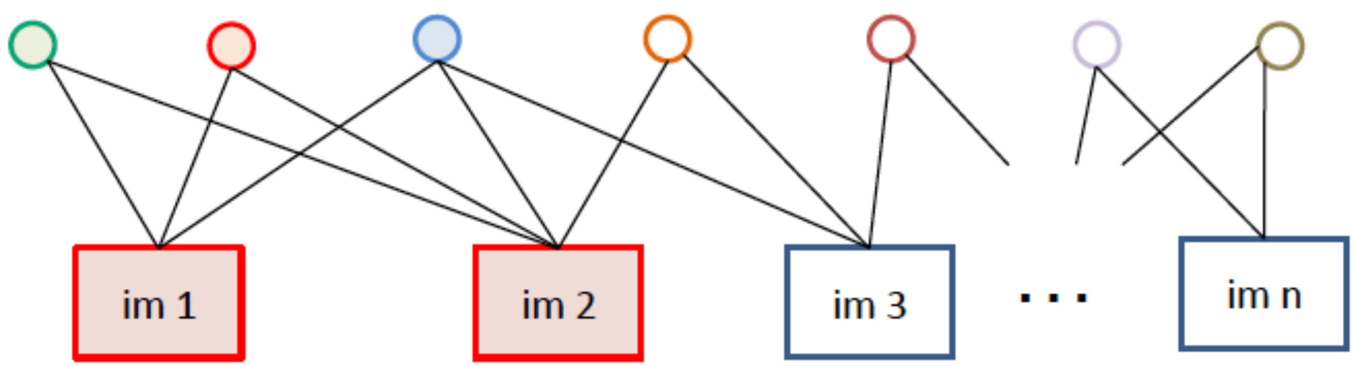
Possibilities

- Hash feature points, look for collisions
- Hash images, look for collisions

Incremental SFM: **initialize reconstruction**

1. Choose two images that are likely to provide a stable estimate of relative pose
   – E.g., $\frac{\text{\# inliers for } H}{\text{\# inliers for } F} < 0.7$ and many inliers for F
2. Get focal lengths from EXIF, estimate essential matrix using [5-point algorithm](), extract pose R_2, t_2 with $R_1 = \boldsymbol{I}, t_1 = \boldsymbol{0}$
3. Solve for 3D points given poses
4. Perform bundle adjustment to refine points and poses

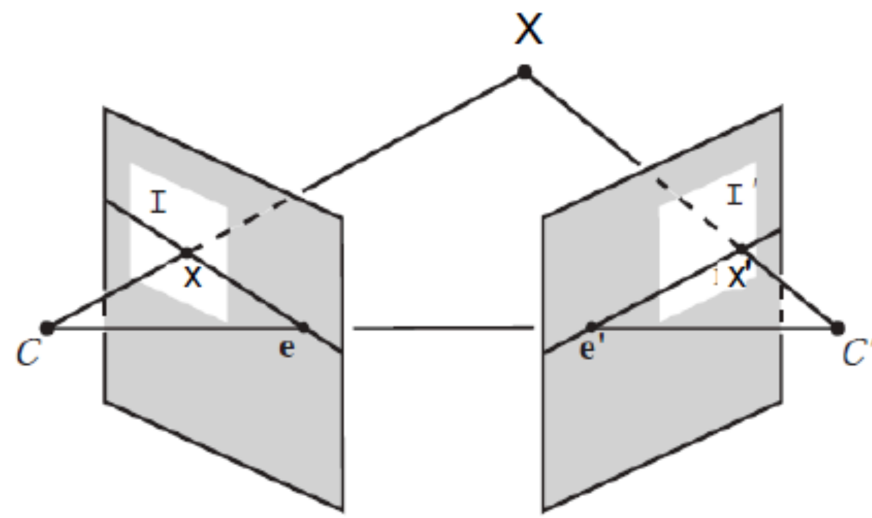
This choice matters, A LOT



filled circles = “triangulated” points
filled rectangles = “resectioned” images (solved pose)

Triangulation: Linear Solution

- Generally, rays C→x and C'→x' will not exactly intersect
- Can solve via SVD, finding a least squares solution to a system of equations



$$\mathbf{x} = \mathbf{P}\mathbf{X} \qquad \mathbf{x}' = \mathbf{P}'\mathbf{X}$$

$$\mathbf{A}\mathbf{X} = \mathbf{0} \quad \mathbf{A} = \begin{bmatrix} u\mathbf{p}_3^T - \mathbf{p}_1^T \\ v\mathbf{p}_3^T - \mathbf{p}_2^T \\ u'\mathbf{p}_3'^T - \mathbf{p}_1'^T \\ v'\mathbf{p}_3'^T - \mathbf{p}_2'^T \end{bmatrix}$$

Further reading: HZ p. 312-313

Triangulation: Linear Solution

$$\mathbf{x} = w\begin{bmatrix} u \\ v \\ 1 \end{bmatrix} \quad \mathbf{x}' = w\begin{bmatrix} u' \\ v' \\ 1 \end{bmatrix}$$

Given **P**, **P'**, **x**, **x'**

1. Precondition points and projection matrices
2. Create matrix **A**
3. [U, S, V] = svd(A)
4. **X** = V(:, end)

$$\mathbf{P} = \begin{bmatrix} \mathbf{p}_1^T \\ \mathbf{p}_2^T \\ \mathbf{p}_3^T \end{bmatrix} \quad \mathbf{P}' = \begin{bmatrix} \mathbf{p}_1'^T \\ \mathbf{p}_2'^T \\ \mathbf{p}_3'^T \end{bmatrix}$$

$$\mathbf{A} = \begin{bmatrix} u\mathbf{p}_3^T - \mathbf{p}_1^T \\ v\mathbf{p}_3^T - \mathbf{p}_2^T \\ u'\mathbf{p}_3'^T - \mathbf{p}_1'^T \\ v'\mathbf{p}_3'^T - \mathbf{p}_2'^T \end{bmatrix}$$

Pros and Cons

- Works for any number of corresponding images
- Not projectively invariant

Implicit error model is weird - the errors you make are really in point localization

Code: http://www.robots.ox.ac.uk/~vgg/hzbook/code/vgg_multiview/vgg_X_from_xP_lin.m

Triangulation: Non-linear Solution

- Minimize projected error while satisfying $\hat{\boldsymbol{x}}'^{T} \boldsymbol{F} \hat{\boldsymbol{x}} = 0$

$$cost(\boldsymbol{X}) = dist(\boldsymbol{x}, \hat{\boldsymbol{x}})^2 + dist(\boldsymbol{x}', \hat{\boldsymbol{x}}')^2$$

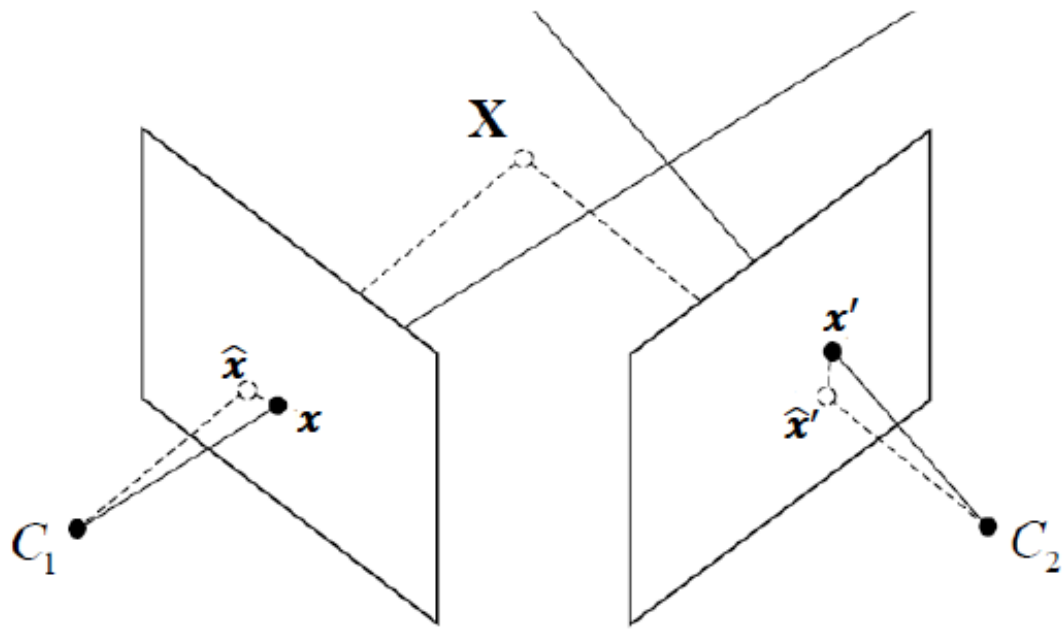
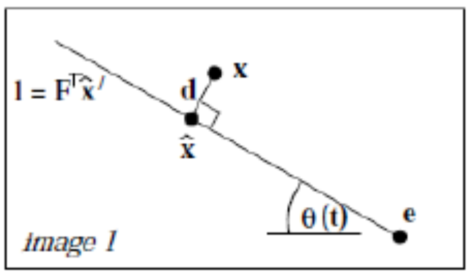


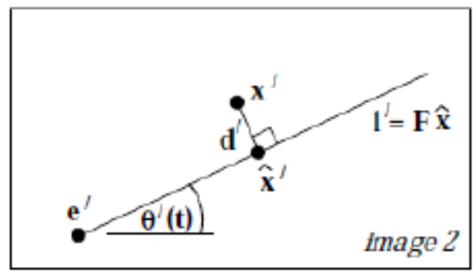
Figure source: Robertson and Cipolla (Chpt 13 of Practical Image Processing and Computer Vision)

Triangulation: Non-linear Solution

- Minimize projected error while satisfying $\hat{\boldsymbol{x}}'^T \boldsymbol{F} \hat{\boldsymbol{x}} = 0$

$$cost(\boldsymbol{X}) = dist(\boldsymbol{x}, \hat{\boldsymbol{x}})^2 + dist(\boldsymbol{x}', \hat{\boldsymbol{x}}')^2$$





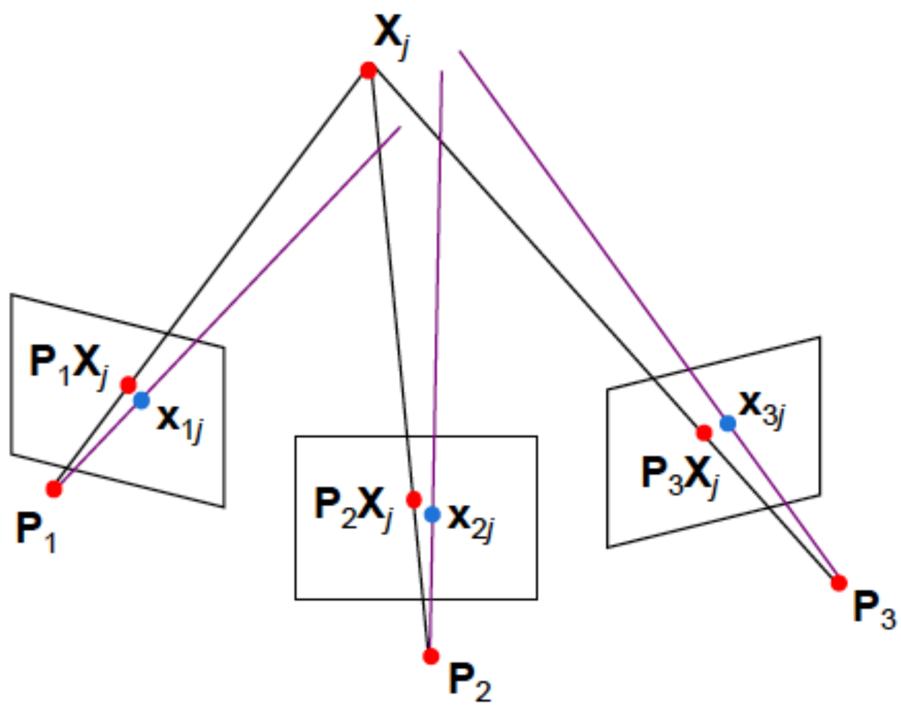
- Solution is a 6-degree polynomial of *t*, minimizing $d(\mathbf{x}, \mathbf{l}(t))^2 + d(\mathbf{x}', \mathbf{l}'(t))^2$

Further reading: HZ p. 318

Bundle adjustment

- Non-linear method for refining structure and motion
- Minimizing reprojection error

$$E(\mathbf{P},\mathbf{X}) = \sum_{i=1}^{m}\sum_{j=1}^{n} D\left(\mathbf{x}_{ij}, \mathbf{P}_i\mathbf{X}_j\right)^2$$

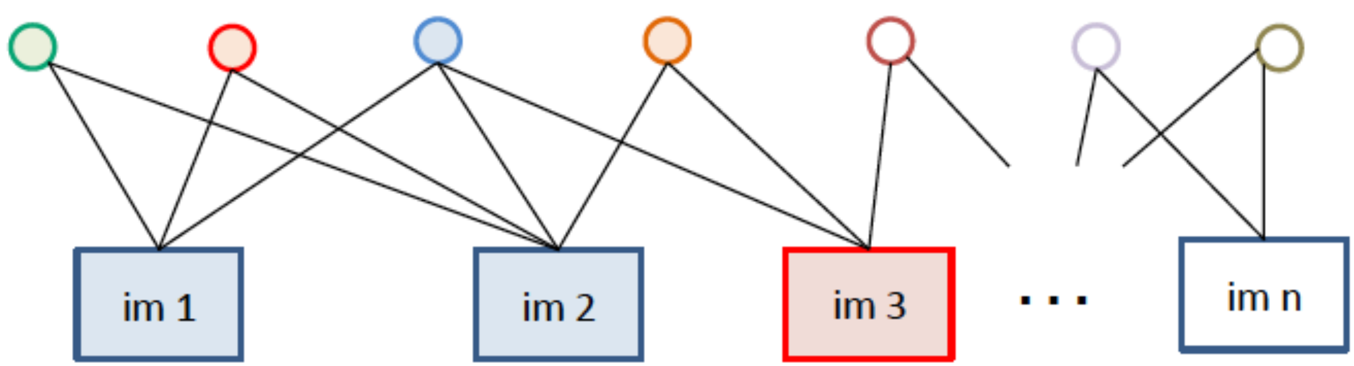


Bundle adjustment is a big deal

- Two cases
  - local (few views)
  - global (all pix)
- Significant improvements in reconstruction
- Very expensive at large scale
  - requires a second order, approximate second order method
    - typically, some version of Levenberg-Marquardt
      - massive hessian
      - usually pcg to solve at large scales
  - significant robustness issues
    - trick: filter points with large reprojection error and go again

Incremental SFM: **grow reconstruction**

1. Resection: solve pose for image(s) that have the most triangulated points
2. Triangulate: solve for any new points that have at least two cameras
3. Remove 3D points that are outliers
4. Bundle adjust
   - For speed, only do full bundle adjust after some percent of new images are resectioned
5. Optionally, align with GPS from EXIF or ground control points (GCP)



filled circles = "triangulated" points
filled rectangles = "resectioned" images (solved pose)

Bundle adjustment is a big deal

- Two cases
  - local (few views)
  - global (all pix)
- Significant improvements in reconstruction
- Very expensive at large scale
  - requires a second order, approximate second order method
    - typically, some version of Levenberg-Marquardt
      - massive hessian
      - usually pcg to solve at large scales
  - significant robustness issues
    - trick: filter points with large reprojection error and go again

Important recent papers and methods for SfM

- OpenMVG
  - https://github.com/openMVG/openMVG
  - http://imagine.enpc.fr/~moulonp/publis/iccv2013/index.html (Moulin et al. ICCV 2013)
  - Software has global and incremental methods

- OpenSfM (software only): https://github.com/mapillary/OpenSfM
  - Basis for my description of incremental SfM

- Visual SfM: Visual SfM (Wu 2013)
  - Used to be the best incremental SfM software (but not anymore and closed source); paper still very good

Reconstruction of Cornell (Crandall et al. ECCV 2011)

use as a general-purpose method. In this paper, we propose a new SfM algorithm to approach this ultimate goal. The new method is evaluated on a variety of challenging datasets and the code is contributed to the research community as an open-source implementation named *COLMAP* available at https://github.com/colmap/colmap.

Multiview Stereo (MVS)

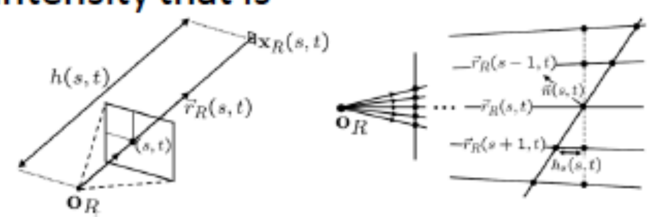
"Multiview Stereo: a tutorial" by Yasu Furukawa
http://www.cse.wustl.edu/~furukawa/papers/fnt_mvs.pdf

Software:

- MVE: https://github.com/simonfuhrmann/mve

Main ideas:

- Initialize with SfM
- MVS: For each image, find 2+ other images with similar viewpoints but substantial baselines
  - Grow regions from sparse points in SfM
  - Create a patch around each pixel and solve for depth, surface normal, and relative intensity that is consistent with all images



Where does SfM fail?

- Not enough images with enough overlap
  - Disconnected reconstructions
- Featureless or reflecting surfaces
  - No matches or bad matches
- Images with pure rotations
  - Recovery of “F” can fail or bad pose reconstruction
- Repeated structures (buildings or bridges)
  - Many consistent bad matches results in inconsistent reconstructions