Structure From Motion

EECS 442 – David Fouhey

Fall 2019, University of Michigan

http://web.eecs.umich.edu/~fouhey/teaching/EECS442_F19/

Structure-from-Motion Revisited

Johannes L. Schönberger, Jan-Michael Frahm

CVPR 2016

Code available at:
https://github.com/colmap/colmap

Structure from motion

Have: 2D points $\mathbf{p}_{ij}$ seen in m images

Assume: points generated from n fixed 3D points $\mathbf{X}_j$ and cameras M_i or $\boldsymbol{p}_{ij} \equiv \boldsymbol{M}_i \boldsymbol{X}_j$

Want: Cameras $\boldsymbol{M}_i$,

points $\boldsymbol{X}_j$

(Remember)

$$\boldsymbol{M}_i \equiv \boldsymbol{K}_i[\boldsymbol{R}_i, \boldsymbol{t}_i]$$

$$\lambda \boldsymbol{p}_{ij} = \boldsymbol{M}_i \boldsymbol{X}_j, \lambda \neq 0$$

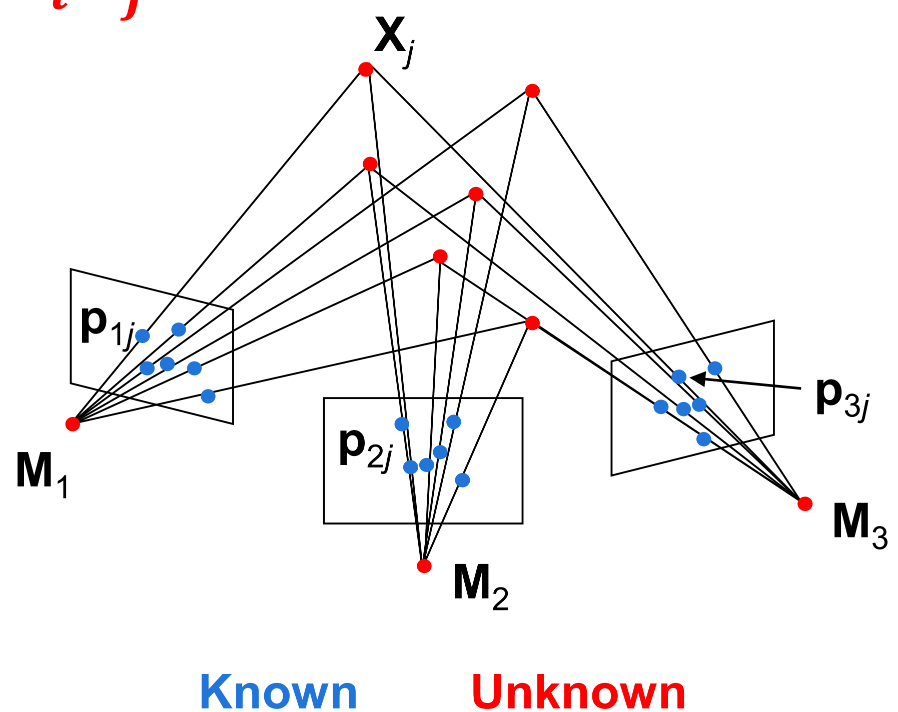
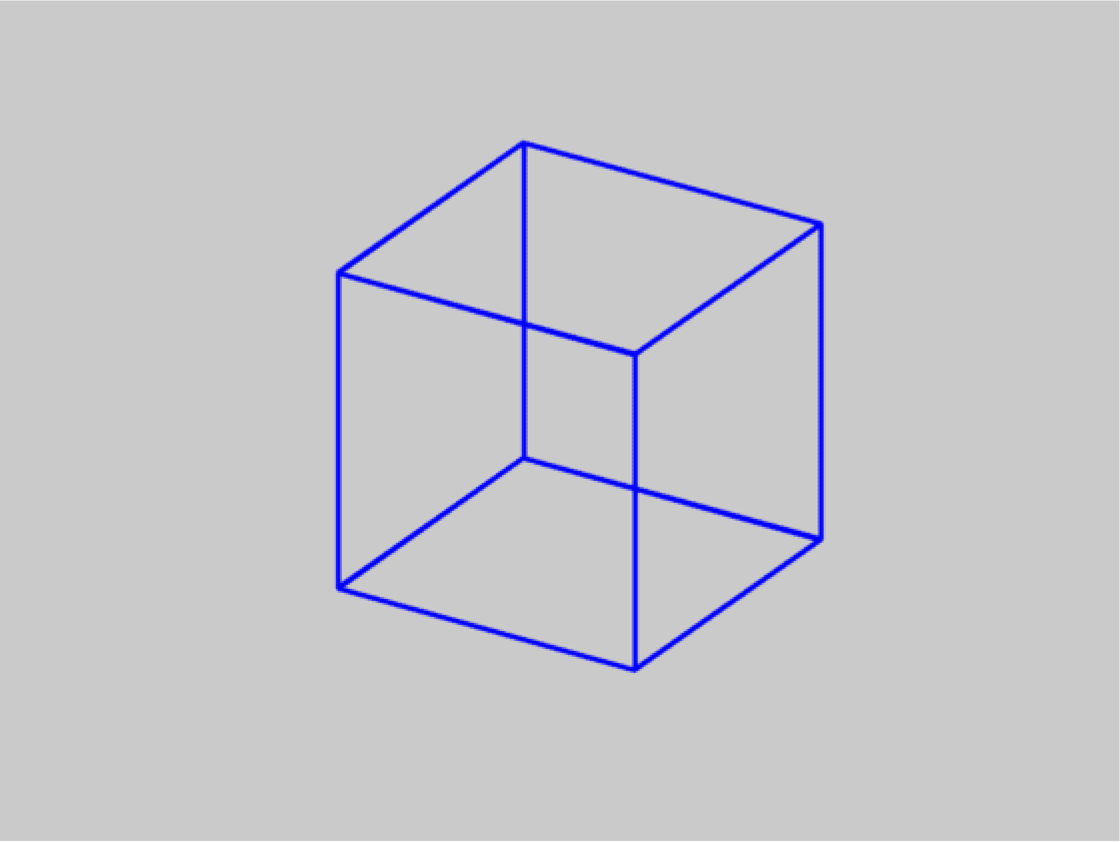


Diagram credit: S. Lazebnik

Is SFM always uniquely solvable?



- Necker cube

Source: N. Snavely

Structure from motion ambiguities

Let's first find one easy ambiguity

$$\boldsymbol{p}_{ij} \equiv \boldsymbol{M}_i \; \boldsymbol{X}_j$$

3x1 3x4 4x1

Zoolander, 2001

Structure from motion ambiguities

Let's first find one easy ambiguity

$$\boldsymbol{p}_{ij} \equiv \boldsymbol{M}_i \boldsymbol{X}_j$$

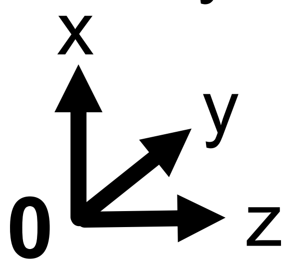
Can pick any arbitrary scaling factor k
and adjust the cameras and points

$$\boldsymbol{p}_{ij} \equiv \boldsymbol{M}_i k^{-1} k \boldsymbol{X}_j$$

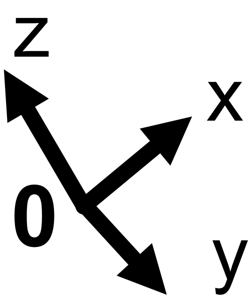
(Can usually be fixed in practice: just need a number, obtainable from heights of known objects or an IMU)

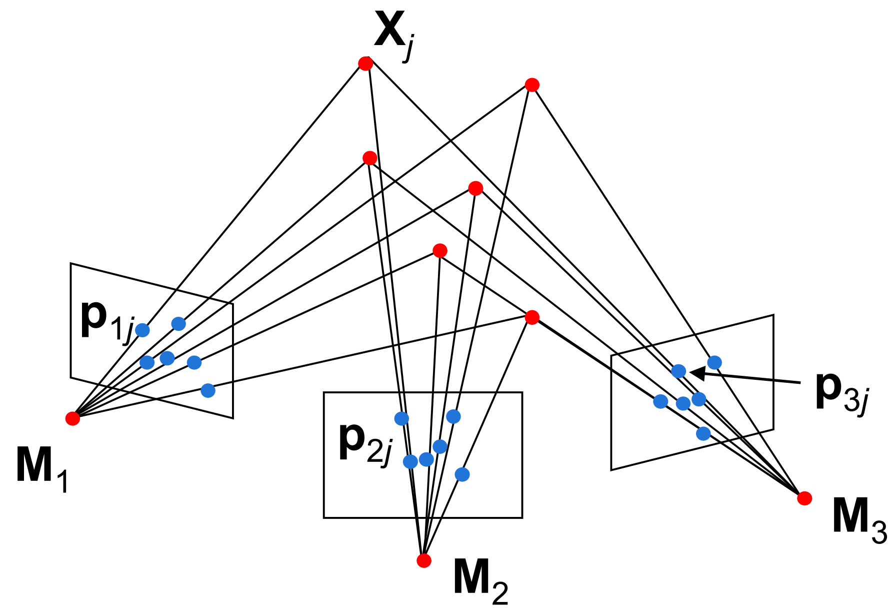
Structure from motion ambiguity

Does this diagram change meaning if I use this coordinate system?



Versus this coordinate system?





Coordinate system irrelevant!
So global **R,t** also ambiguous

Structure from motion ambiguities

Not just limited to scale. Given:

$$\boldsymbol{p}_{ij} \equiv \boldsymbol{M}_i \boldsymbol{X}_j$$

Can insert any global transform **H**

$$\boldsymbol{p}_{ij} \equiv \boldsymbol{M}_i \boldsymbol{X}_j = \boldsymbol{M}_i \boldsymbol{H}^{-1} \boldsymbol{H} \boldsymbol{X}_j$$

H is a 3D homography / perspective transform / projective transform

Similarity/Affine/Perspective

Given: 

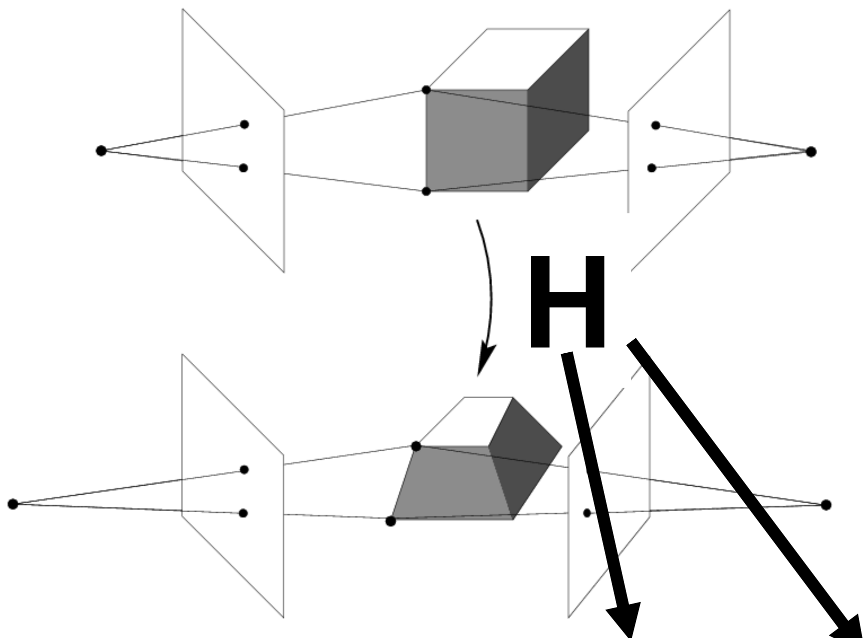
Perspective	Affine	Similarity
		
Lines	+Parallelism	+Angles
$\begin{bmatrix} a & b & c \\ d & e & f \\ g & h & i \end{bmatrix}$	$\begin{bmatrix} a & b & c \\ d & e & f \\ 0 & 0 & 1 \end{bmatrix}$	$\begin{bmatrix} s\boldsymbol{R} & \boldsymbol{t} \\ 0 & 1 \end{bmatrix}$

3D: same idea, different dimensions

House image: A. Efros

Projective ambiguity

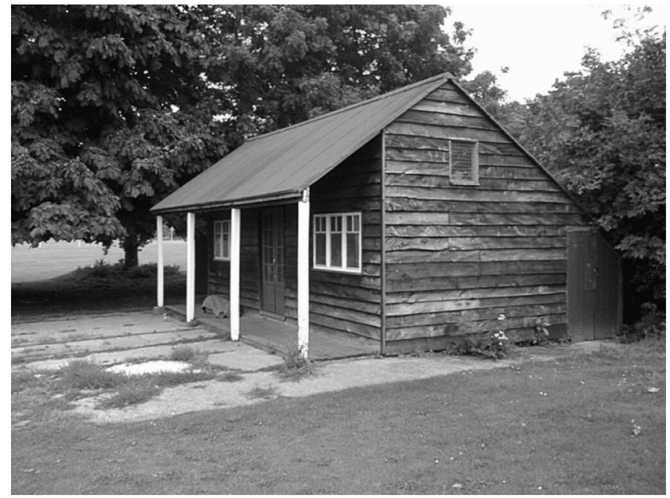
With no constraints on cameras matrices and scene, can only reconstruct up to a perspective ambiguity

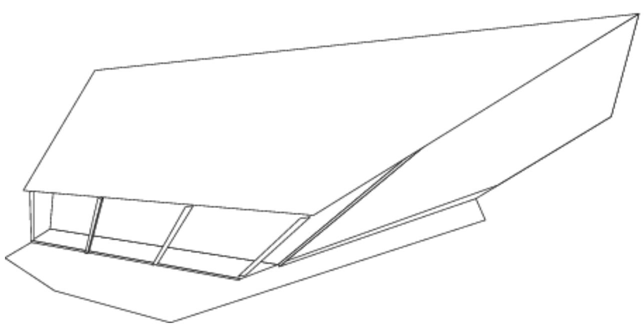


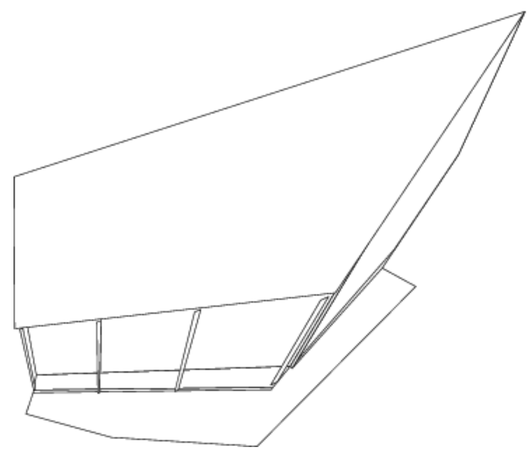
$$\mathbf{p}_{ij} \equiv \mathbf{M}_i \mathbf{X}_j = \mathbf{M}_i \mathbf{H}^{-1} \mathbf{H} \mathbf{X}_j$$

Slide credit: S. Lazebnik

Projective ambiguity



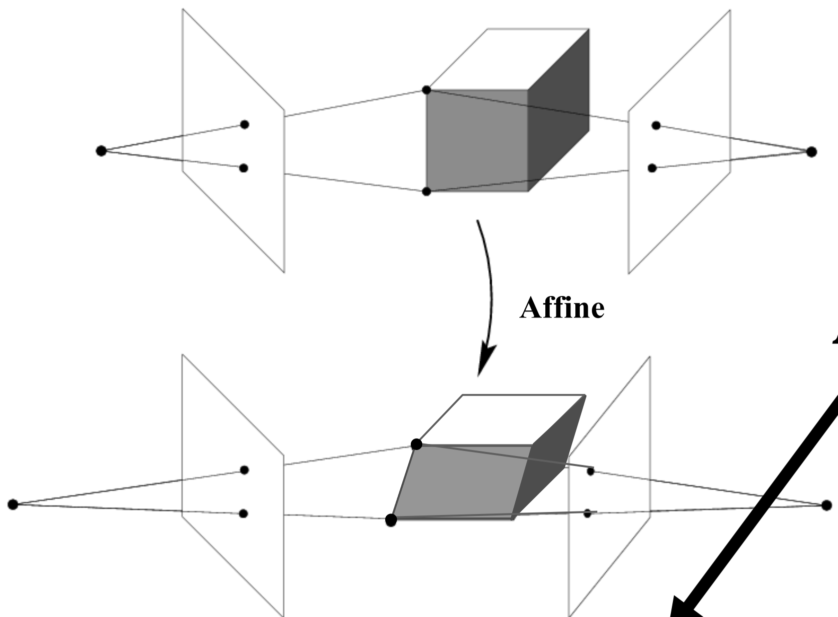




Slide credit: S. Lazebnik

Affine ambiguity

If we have constraints in the form of what lines are parallel, can reduce ambiguity to *affine ambiguity*.

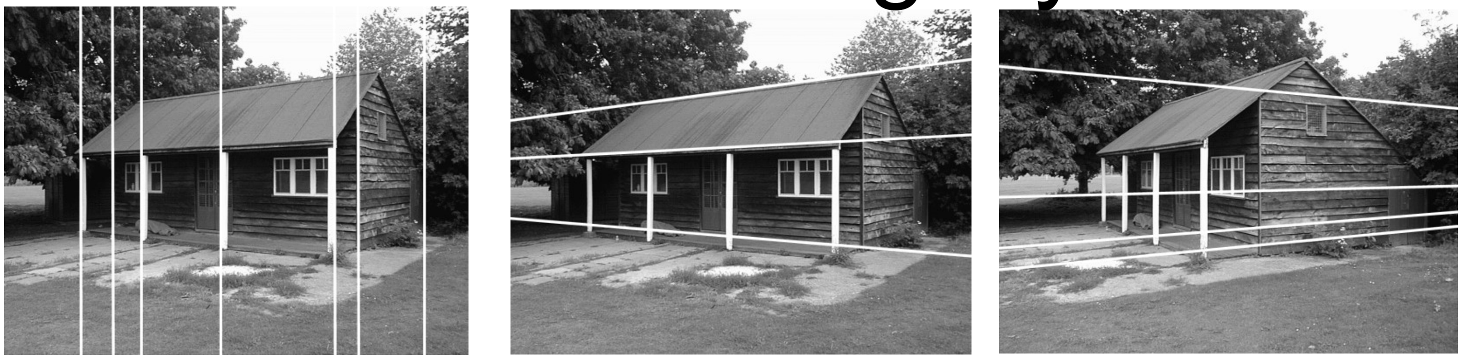


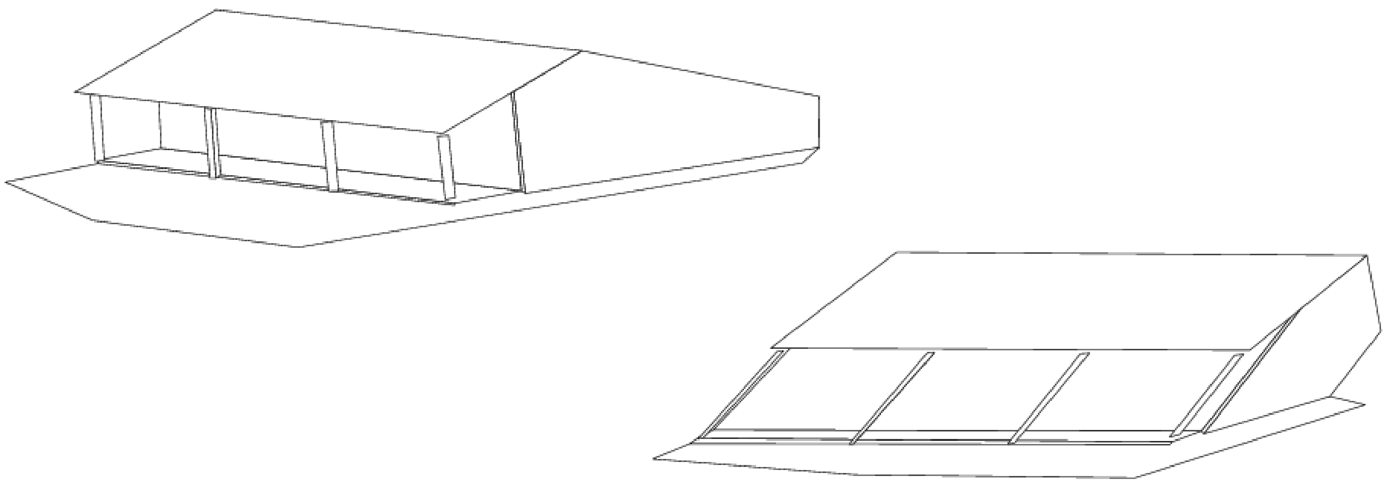
$$\boldsymbol{H} = \begin{bmatrix} \boldsymbol{A} & \boldsymbol{t} \\ 0 & 1 \end{bmatrix}$$

$$\boldsymbol{p}_{ij} \equiv \boldsymbol{M}_i \boldsymbol{X}_j = \boldsymbol{M}_i \boldsymbol{H}^{-1} \boldsymbol{H} \boldsymbol{X}_j$$

Slide credit: S. Lazebnik

Affine ambiguity

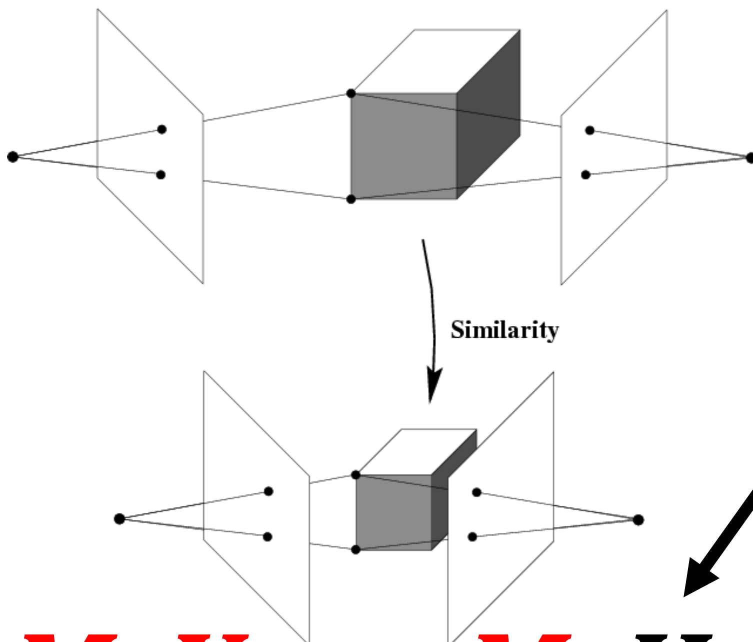




Slide credit: S. Lazebnik

Similarity ambiguity

If we have orthogonality constraints, get up to similarity transform. *Really the best we can do.* We get this if we have calibrated cameras.

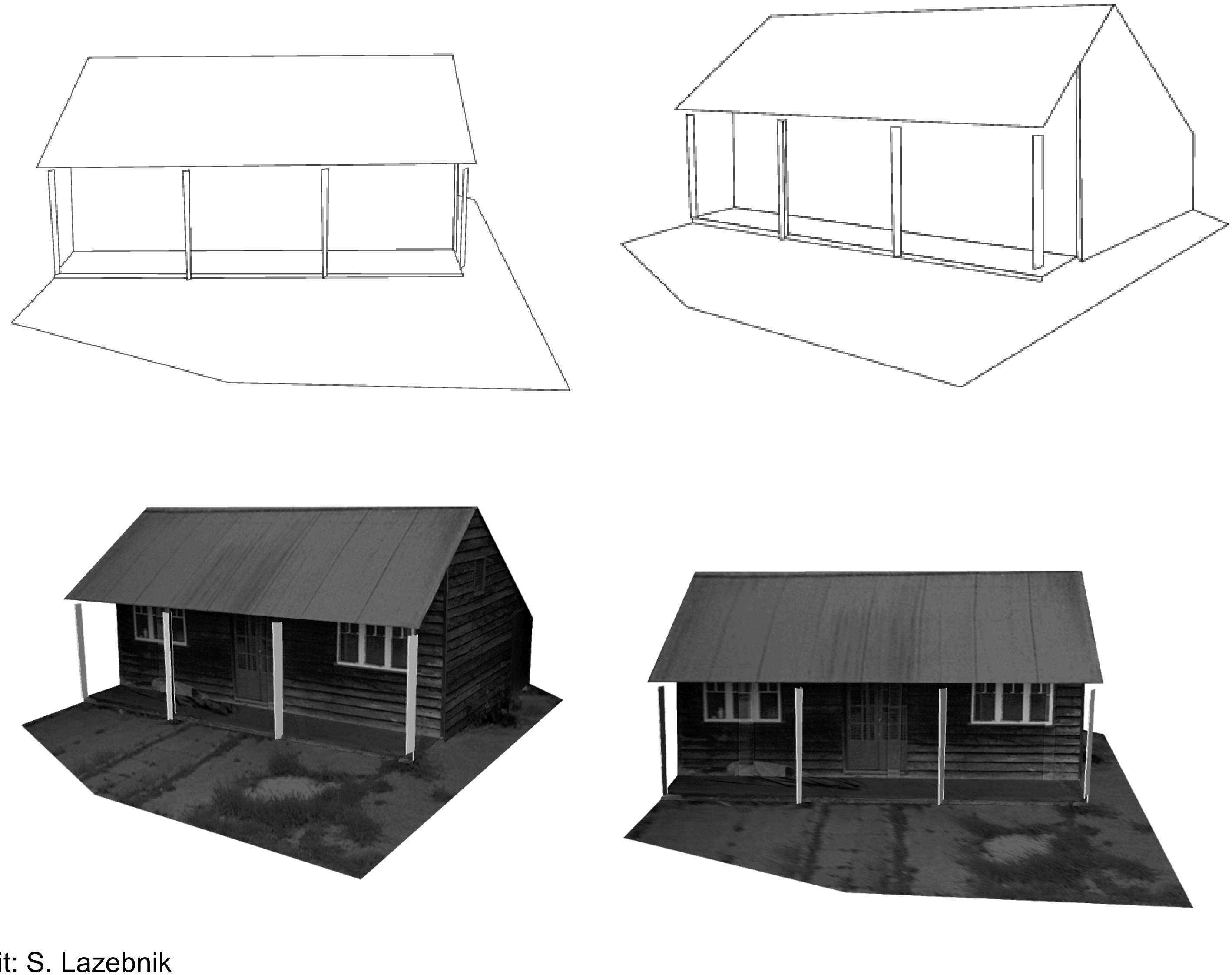


$$\boldsymbol{H} = \begin{bmatrix} s\boldsymbol{R} & \boldsymbol{t} \\ 0 & 1 \end{bmatrix}$$

$$\boldsymbol{p}_{ij} \equiv \boldsymbol{M}_i \boldsymbol{X}_j = \boldsymbol{M}_i \boldsymbol{H}^{-1} \boldsymbol{H} \boldsymbol{X}_j$$

Slide credit: S. Lazebnik

Similarity ambiguity



Slide credit: S. Lazebnik

Affine structure from motion

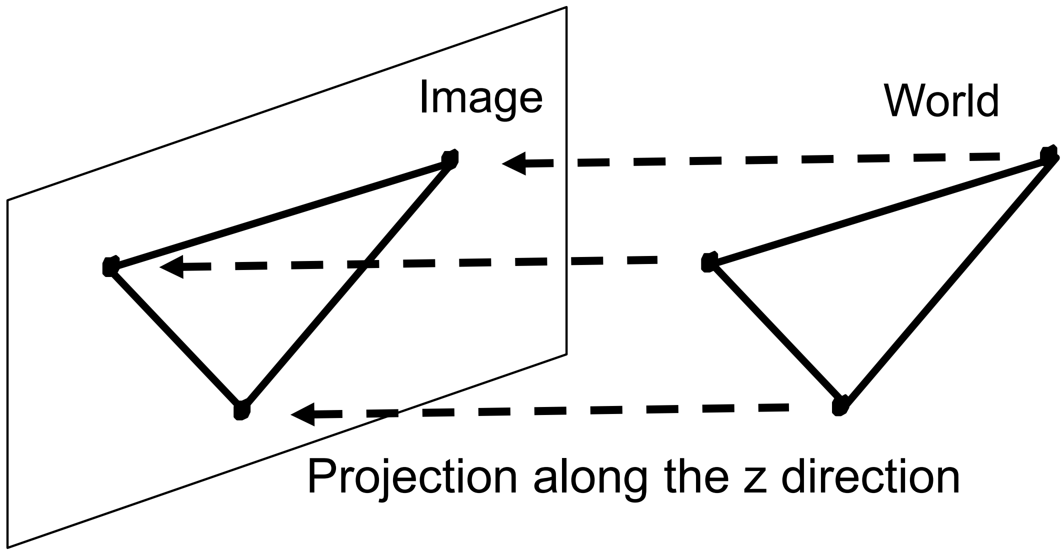
We'll do the math with affine / weak perspective cameras (math is much easier)

Perspective

Weak Perspective

Recall: orthographic projection

Orthographic camera: things infinitely far away but you have an amazing camera



$$\begin{bmatrix} u \\ v \\ 1 \end{bmatrix} = \begin{bmatrix} 1 & 0 & 0 & 0 \\ 0 & 1 & 0 & 0 \\ 0 & 0 & 0 & 1 \end{bmatrix} \begin{bmatrix} x \\ y \\ z \\ 1 \end{bmatrix} \to \begin{bmatrix} x \\ y \end{bmatrix}$$

Field of view and focal length

wide-angle

standard

telephoto

Slide Credit: F. Durand

Affine Camera

$$\boldsymbol{M} = \begin{bmatrix} \boldsymbol{A}_{2D} & \boldsymbol{t}_{2D} \\ 0 & 1 \end{bmatrix} \begin{bmatrix} 1 & 0 & 0 & 0 \\ 0 & 1 & 0 & 0 \\ 0 & 0 & 0 & 1 \end{bmatrix} \begin{bmatrix} \boldsymbol{A}_{3D} & \boldsymbol{t}_{3D} \\ 0 & 1 \end{bmatrix}$$

3x3 Matrix Affine 2D | 3x4 Ortho. Proj | 4x4 Matrix Affine 3D

Tedious math…

$$\boldsymbol{M} = \begin{bmatrix} a_{11} & a_{12} & a_{13} & b_1 \\ a_{21} & a_{22} & a_{23} & b_2 \\ 0 & 0 & 0 & 1 \end{bmatrix}$$

Affine Camera

So what? Who cares?

Examine the projection

$$\begin{bmatrix} u \\ v \\ 1 \end{bmatrix} \equiv \begin{bmatrix} a_{11} & a_{12} & a_{13} & b_1 \\ a_{21} & a_{22} & a_{23} & b_2 \\ 0 & 0 & 0 & 1 \end{bmatrix} \begin{bmatrix} X \\ Y \\ Z \\ 1 \end{bmatrix}$$

Projection becomes linear mapping + translation and doesn't involve homogeneous coordinates!

$$\begin{bmatrix} u \\ v \end{bmatrix} \equiv \begin{bmatrix} a_{11} & a_{12} & a_{13} \\ a_{21} & a_{22} & a_{23} \end{bmatrix} \begin{bmatrix} X \\ Y \\ Z \end{bmatrix} + \begin{bmatrix} b_1 \\ b_2 \end{bmatrix}$$

b is projection of origin. **Can anyone see why?**

Affine structure from motion

General structure from motion:

$$\underset{3\text{x}1}{\boldsymbol{p}_{ij}} \equiv \underset{3\text{x}4}{\boldsymbol{M}_i}\underset{4\text{x}1}{\boldsymbol{X}_j}$$

Assume M is affine camera:

$$\underset{2\text{x}1}{\boldsymbol{p}_{ij}} = \underset{2\text{x}3}{\boldsymbol{A}_i}\underset{3\text{x}1}{\boldsymbol{X}_j} + \underset{2\text{x}1}{\boldsymbol{b}_i}$$

mn 2D points, m cameras, n 3D points
up to arbitrary 3D affine (12 DOF)

Need:
$2mn \geq 8m + 3n - 12$
$(m = 2)$: $n \geq 4$
(for all m!)

One simplifying trick

$$\boldsymbol{p}_{ij} = \boxed{\boldsymbol{A}_i \boldsymbol{X}_j + \boldsymbol{b}_i}$$

Subtract off the average 2D point

$$\widehat{\boldsymbol{p}_{ij}} = \boldsymbol{p}_{ij} - \frac{1}{n}\sum_{k=1}^{n} \boldsymbol{p}_{ik} = \boxed{\boldsymbol{A}_i \boldsymbol{X}_j + \boldsymbol{b}_i} - \frac{1}{n}\sum_{k=1}^{n} \boxed{\boldsymbol{A}_i \boldsymbol{X}_k + \boldsymbol{b}_i}$$

Gather terms involving A_i ,push out b_i

$$\widehat{\boldsymbol{p}_{ij}} = \boldsymbol{A}_i \left(\boldsymbol{X}_j \underbrace{- \frac{1}{n}\sum_{k=1}^{n} \boldsymbol{X}_k}_{} \right) + \cancel{\boldsymbol{b}_i - \frac{1}{n}\sum_{k=1}^{n} \boldsymbol{b}_i} \quad 0$$

Set origin to mean of 3D points

$$\widehat{\boldsymbol{p}_{ij}} = \boldsymbol{A}_i \boldsymbol{X}_j$$

Can do this entirely in terms of **A**!

Affine structure from motion

First, make data measurement matrix consisting of all the points stacked together

m cameras

$$\begin{bmatrix} \widehat{\mathbf{p}_{11}} & \cdots & \widehat{\mathbf{p}_{1n}} \\ \vdots & \ddots & \vdots \\ \widehat{\mathbf{p}_{m1}} & \cdots & \widehat{\mathbf{p}_{mn}} \end{bmatrix} \qquad \begin{bmatrix} \widehat{u_{11}} & \cdots & \widehat{u_{1n}} \\ \widehat{v_{11}} & & \widehat{v_{1n}} \\ \vdots & \ddots & \vdots \\ \widehat{u_{m1}} & & \widehat{u_{mn}} \\ \widehat{v_{m1}} & \cdots & \widehat{v_{mn}} \end{bmatrix}$$

n points

How big is this matrix?

C. Tomasi and T. Kanade. Shape and motion from image streams under orthography: A factorization method. *IJCV*, 9(2):137-154, November 1992.

Affine structure from motion

Then, write all the equations in one in terms of product of cameras and points.

$$\boldsymbol{D} = \begin{bmatrix} \widehat{\boldsymbol{p}_{11}} & \cdots & \widehat{\boldsymbol{p}_{1n}} \\ \vdots & \ddots & \vdots \\ \widehat{\boldsymbol{p}_{m1}} & \cdots & \widehat{\boldsymbol{p}_{mn}} \end{bmatrix} = \begin{bmatrix} \boldsymbol{A}_1 \\ \vdots \\ \boldsymbol{A}_m \end{bmatrix} \begin{bmatrix} \boldsymbol{X}_1 & \cdots & \boldsymbol{X}_n \end{bmatrix}$$

2m x n	2mx3	3xn
D	**M**	**S**

What's the rank of D?

3!

C. Tomasi and T. Kanade. Shape and motion from image streams under orthography: A factorization method. *IJCV*, 9(2):137-154, November 1992.

Making Matrices Rank Deficient

Repeat of epipolar geometry class, but important enough to see twice. Given matrix **M:**

$$M \rightarrow U\Sigma V^T$$

$U_{m\times m}, V_{n\times n}$ rotation matrices

$\Sigma_{m\times n}$ diagonal scaling matrix

$$\Sigma = \begin{bmatrix} \sigma_1 & \cdots & 0 \\ \vdots & \ddots & \vdots \\ 0 & \cdots & \cancel{\sigma_n} \end{bmatrix}$$

Keep only k biggest σ; set others to 0

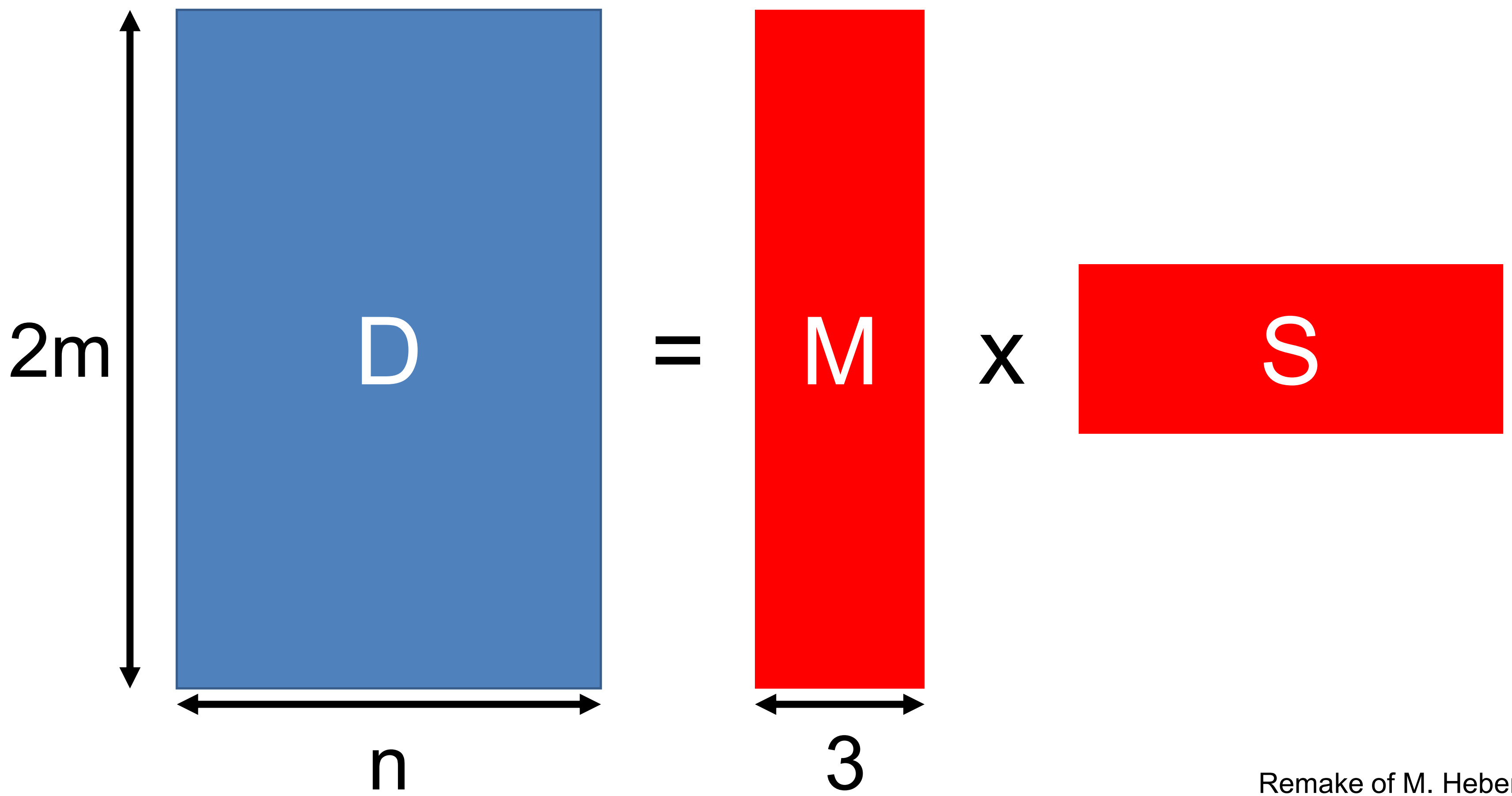
$$\hat{M} \leftarrow U\hat{\Sigma}V^T$$

Minimizes $\left\|M - \hat{M}\right\|_F$ (sum of squares) subject to rank($\hat{M}$) ≤ k

See Eckart–Young–Mirsky theorem if you're interested

Affine structure from motion

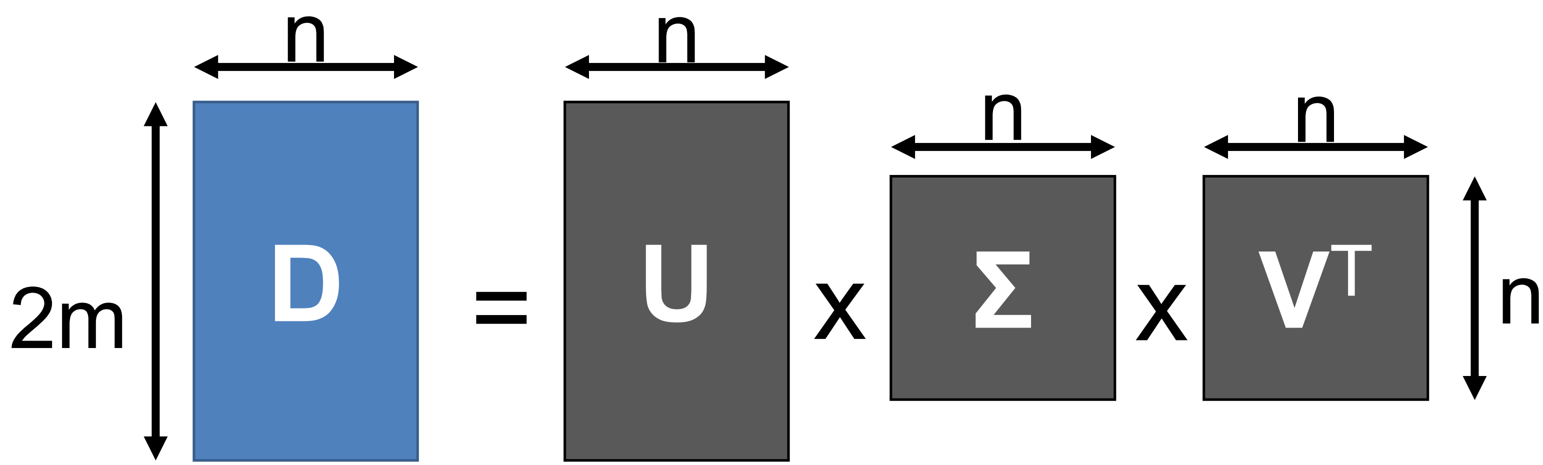
We'd like to take the measurements and convert them into **M**, **S**



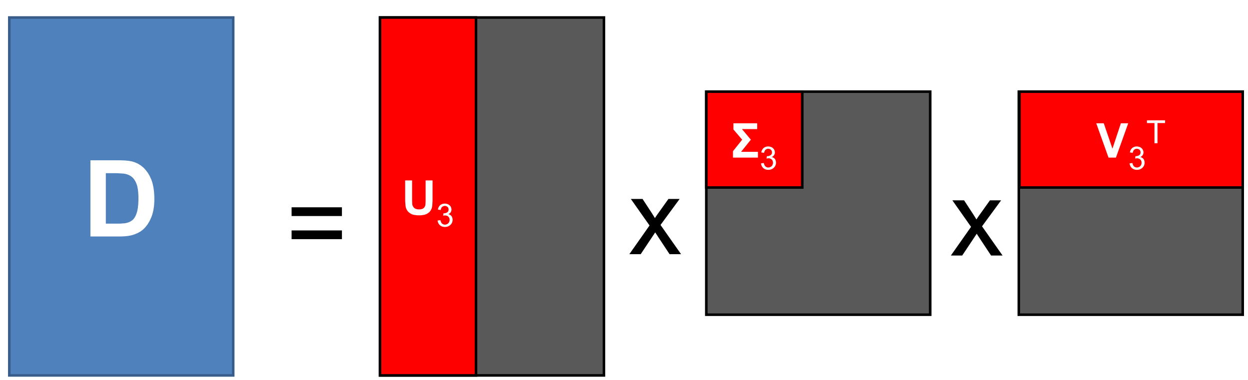
Remake of M. Hebert diagram

Affine structure from motion

Do SVD (typically you don't make full U,Σ,V)



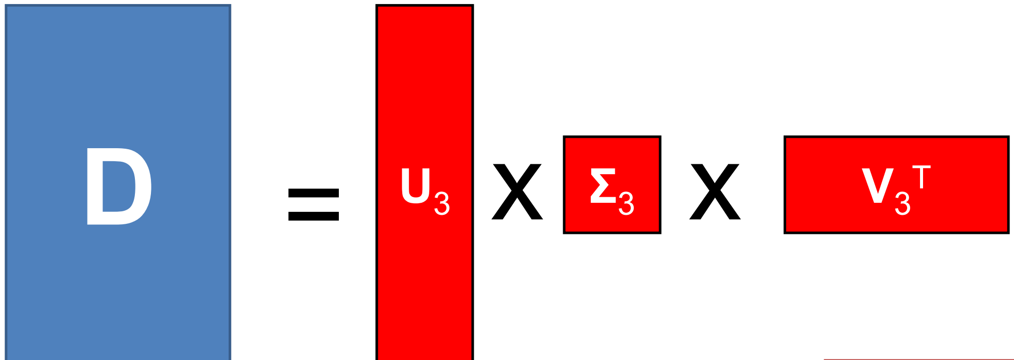
Truncate to top 3 singular values



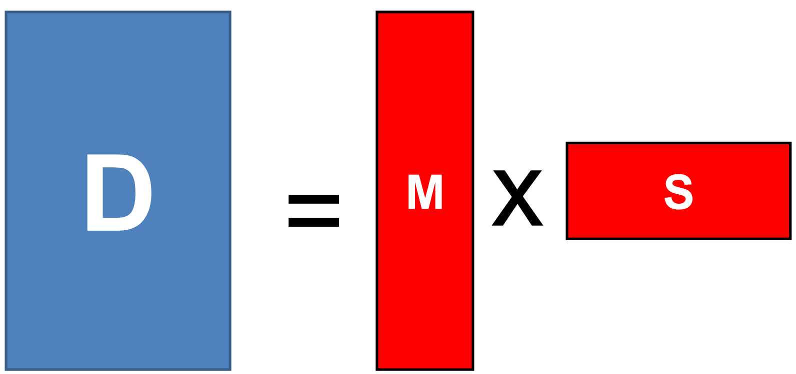
Remake of M. Hebert diagram

Affine structure from motion

Nearly there apart from this annoying $\mathbf{\Sigma}_3$.



One solution (split $\mathbf{\Sigma}_3$ in two): $D = \underbrace{\boxed{U_3\Sigma_3^{1/2}}}_{M}\underbrace{\boxed{\Sigma_3^{1/2}V_3^T}}_{S}$

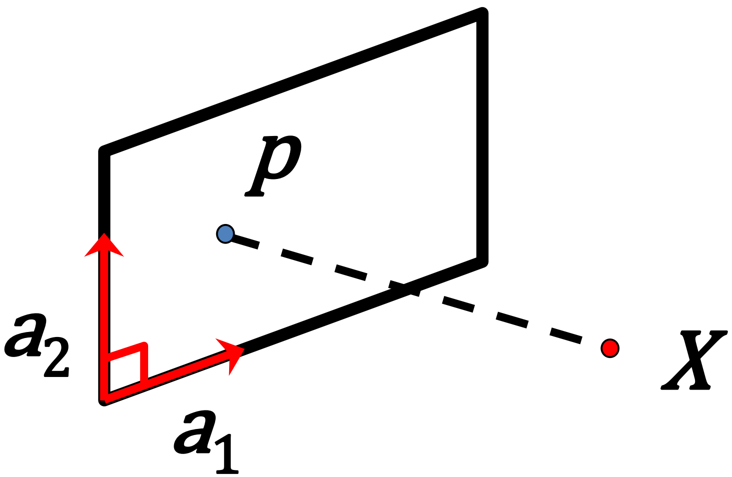


But remember that we can put **H**$\mathbf{H}^{-1}$ in the middle

Remake of M. Hebert diagram

Eliminating the affine ambiguity

Rows $\mathbf{a_i}$ of $\mathbf{A_i}$ give axes of camera. Can multiply each projection $\mathbf{A_i}$ with $\mathbf{C}$ to make $\mathbf{A_iC}$ that satisfies:

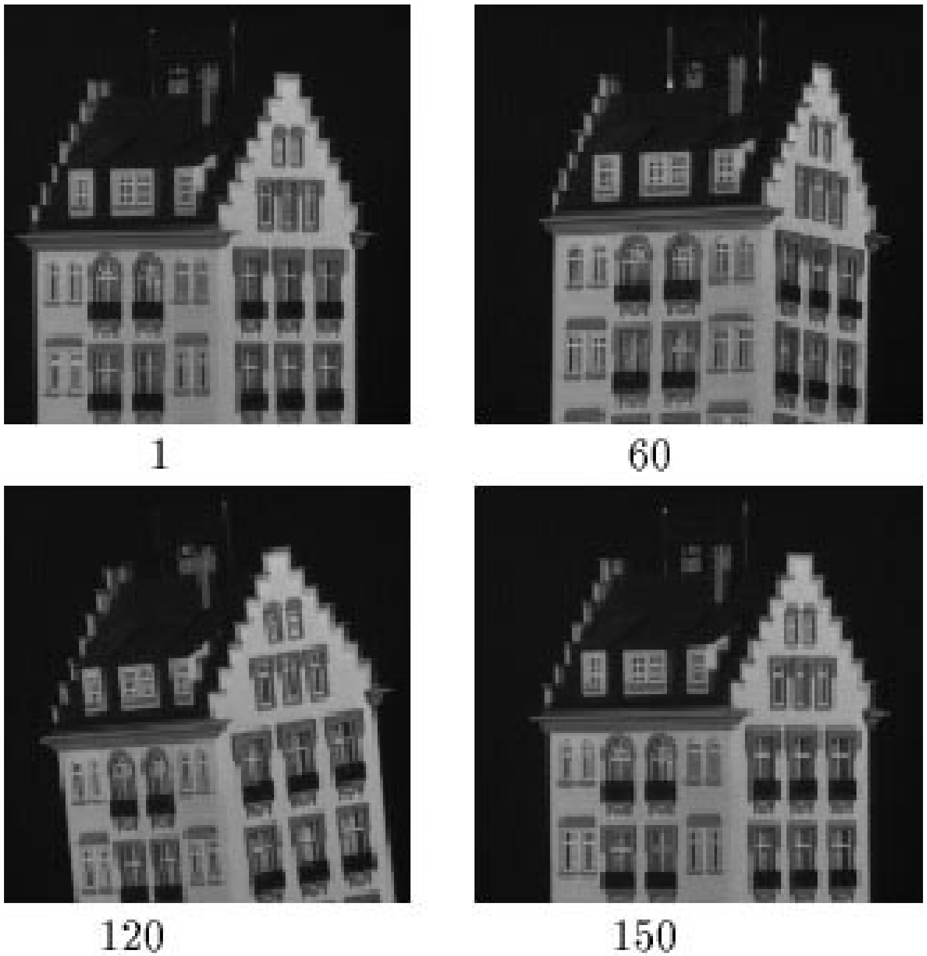


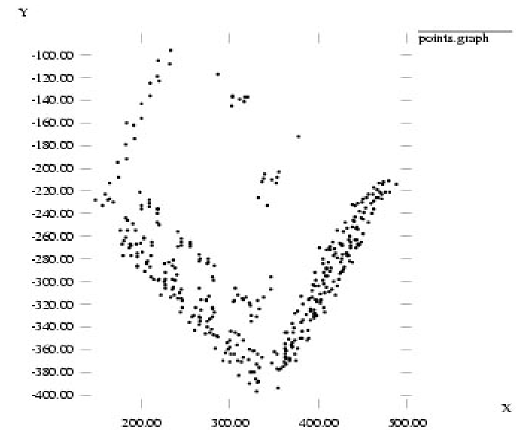
$$\boldsymbol{a_1^T a_2} = 0$$
$$\|\boldsymbol{a_1}\| = 1$$
$$\|\boldsymbol{a_2}\| = 1$$

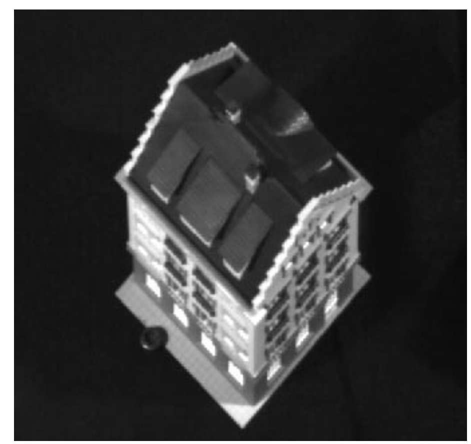
Gives 3 equations per camera, can set $\mathbf{A_iC}$ to new camera, and $\mathbf{C^{-1}S}$ to new points.
In general, a recipe for eliminating ambiguities

Remake of M. Hebert diagram

Reconstruction results



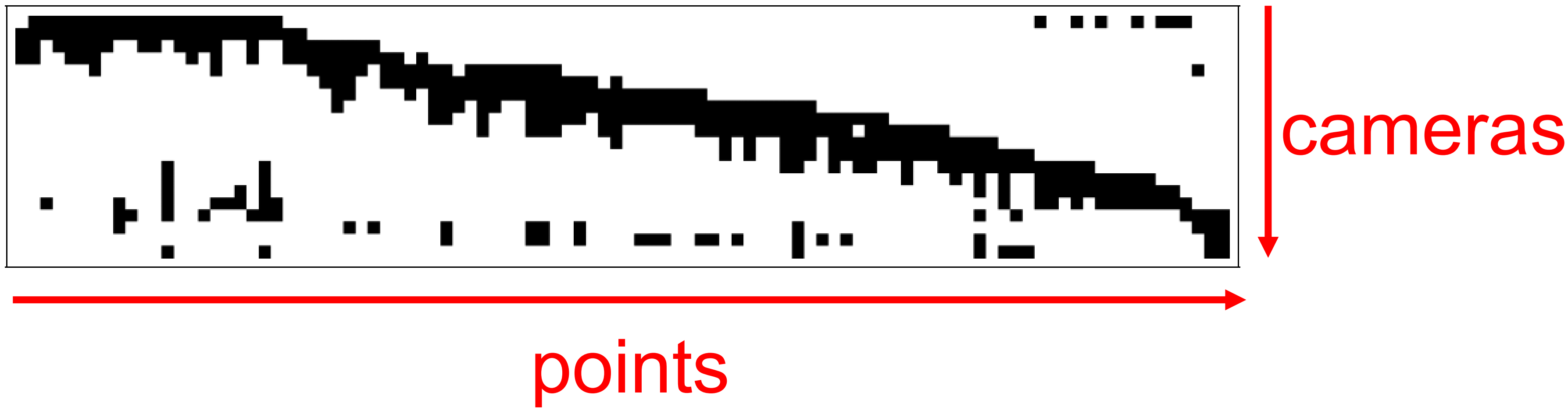




C. Tomasi and T. Kanade, Shape and motion from image streams under orthography: A factorization method, IJCV 1992

Dealing with missing data

So far, assume we can see all points in all views
In reality, measurement matrix typically looks like this:



Possible solution: find dense blocks, solve in block, fuse.
In general, finding these dense blocks is NP-complete

Figure Credit: S. Lazebnik

But cameras aren't affine!

Want: m cameras M_i, n 3D points X_j

Given: mn 2D points p_{ij}

$$\boldsymbol{p}_{ij} \equiv \boldsymbol{M}_i \boldsymbol{X}_j = \boldsymbol{M}_i \boldsymbol{H}^{-1} \boldsymbol{H} \boldsymbol{X}_j$$

When is this Possible?

Want: m cameras M_i, n 3D points X_j

Given: mn 2D points p_{ij}

$$\boldsymbol{p}_{ij} \equiv \boldsymbol{M}_i \boldsymbol{X}_j = \boldsymbol{M}_i \boldsymbol{H}^{-1} \boldsymbol{H} \boldsymbol{X}_j$$

2D point (2)

3x4 camera matrix (11) **why?**

3D point (3)

4x4 homography (15) **why?**

Need 2mn ≥ 11m + 3n – 15

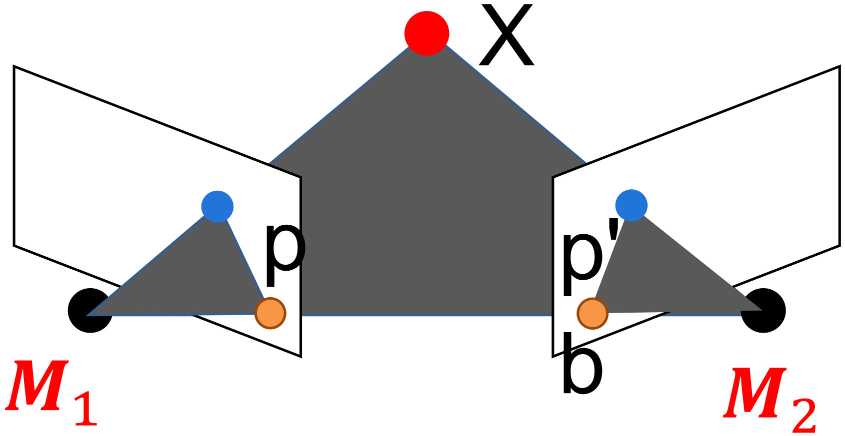
(m = 2): n ≥ 7

(m = 3): n ≥ 6 (doesn't get better after)

(m=1): n ≤ 4

Two Camera Case

For two cameras, we need 7 points. Hmm.
What else (in theory) requires 7 points?



Compute fundamental matrix **F** and epipole **b** s.t. $\mathbf{F}^T\mathbf{b} = 0$. Then:

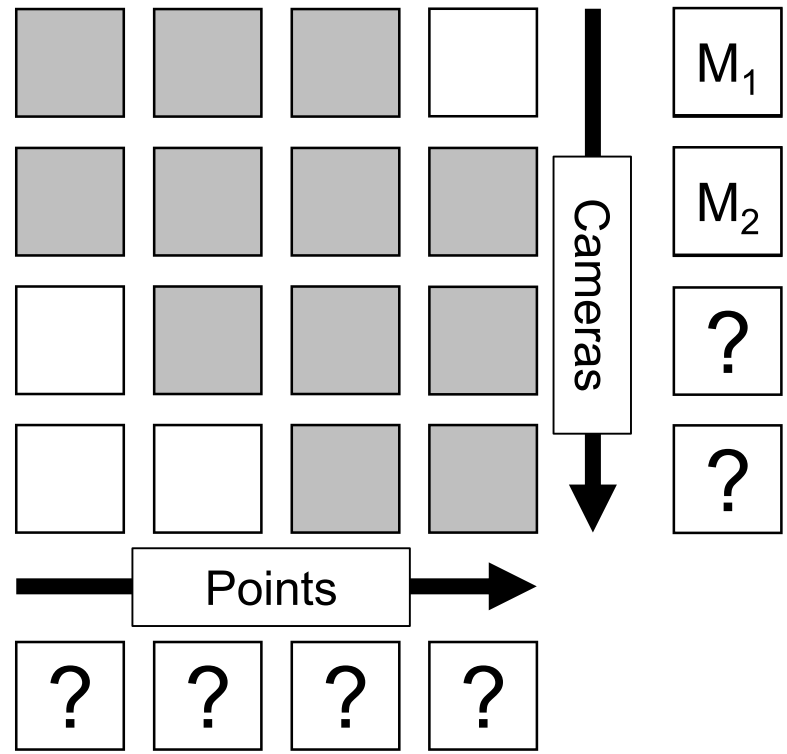
$$\boldsymbol{M}_1 = [\boldsymbol{I}, \boldsymbol{0}]$$

$$\boldsymbol{M}_2 = [-[\boldsymbol{b}_x]\boldsymbol{F}, \boldsymbol{b}]$$

Remember: this is up to a projective ambiguity!

Incremental SFM

Key idea: incrementally add cameras, points



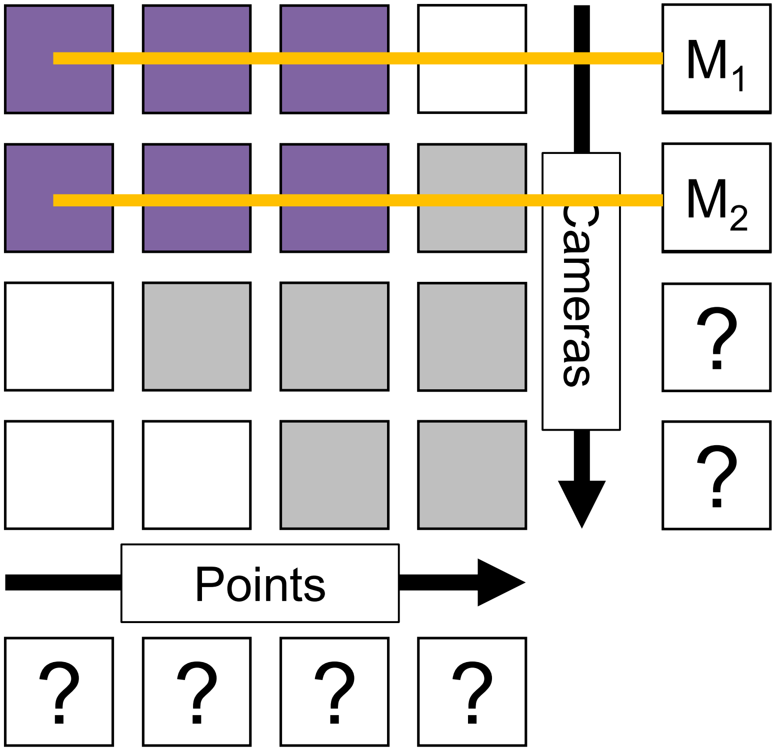
Remake of S. Lazebnik material

Note: numbers of points aren't to scale.

Incremental SFM

Key idea: incrementally add cameras, points

1. Initialize motion $M_i = [R_i, t_i]$ with fundamental matrix



Remake of S. Lazebnik material

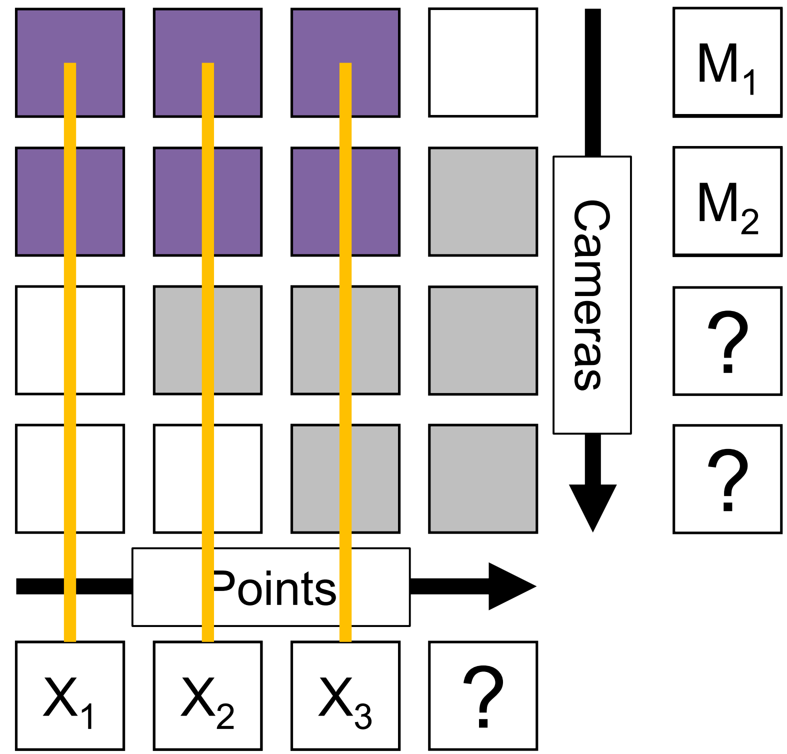
Note: numbers of points aren't to scale.

Incremental SFM

Key idea: incrementally add cameras, points

1. Initialize motion $M_i = [R_i, t_i]$ with fundamental matrix
2. Initialize structure X_j with triangulation

How could we add another camera?



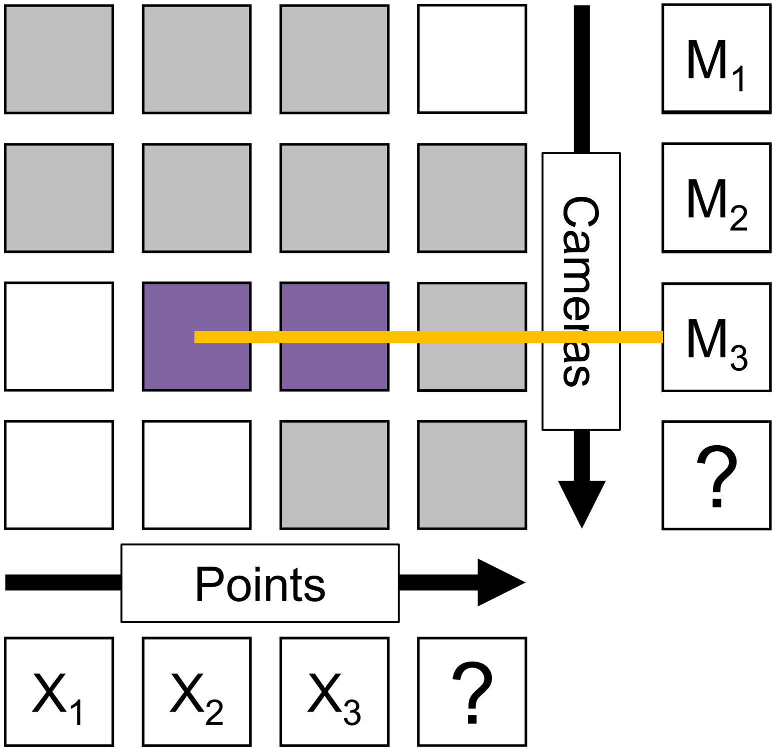
Remake of S. Lazebnik material

Note: numbers of points aren't to scale.

Incremental SFM

Key idea: incrementally add cameras, points

1. Solve for camera matrix using visible, known points using calibration



Remake of S. Lazebnik material

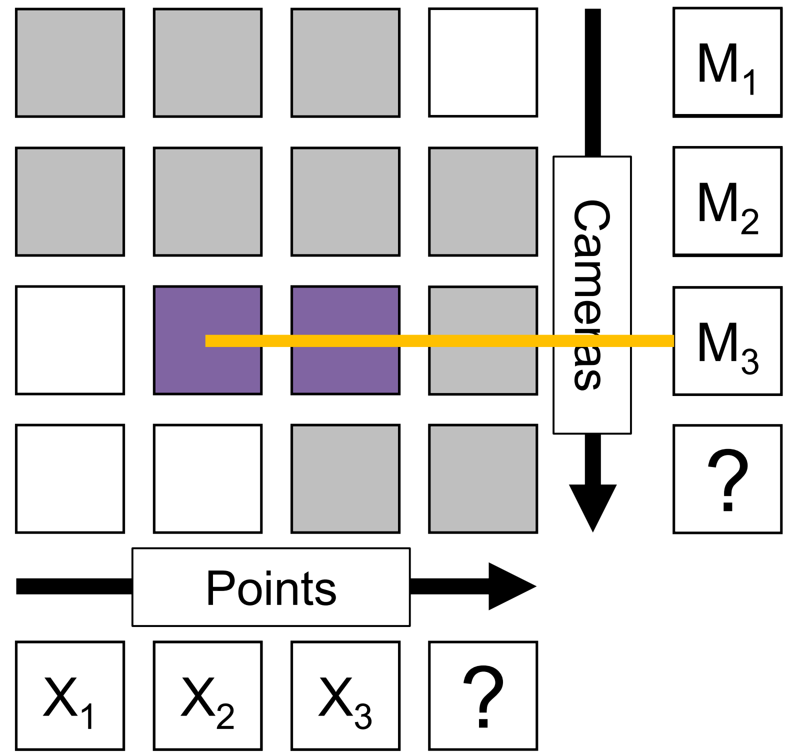
Note: numbers of points aren't to scale.

Incremental SFM

Key idea: incrementally add cameras, points

1. Solve for camera matrix using visible, known points using calibration

Now we can see the fourth point in two cameras.



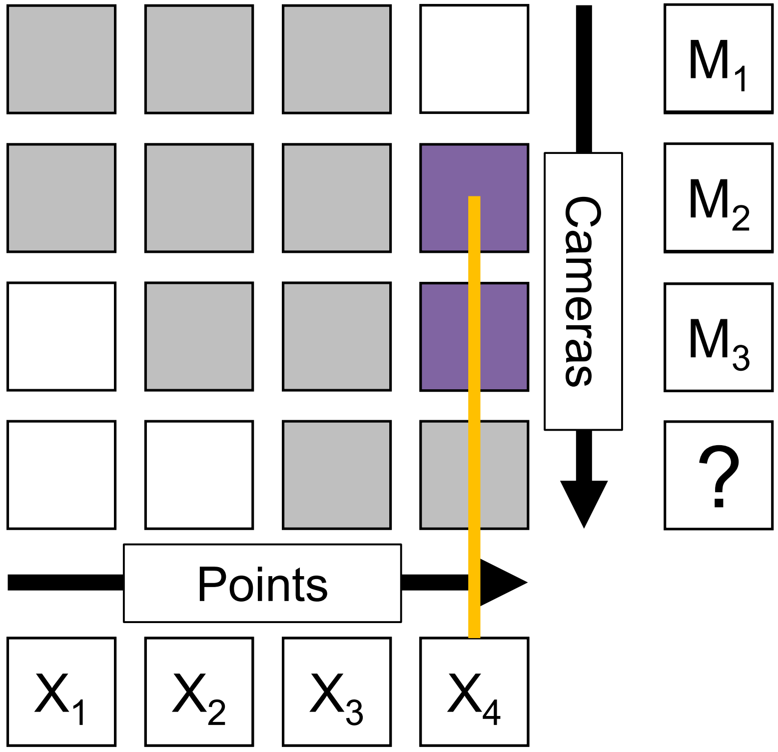
Remake of S. Lazebnik material

Note: numbers of points aren't to scale.

Incremental SFM

Key idea: incrementally add cameras, points

1. Solve for camera matrix using visible, known points using calibration
2. Solve for 3D coordinates of newly visible points using triangulation



Remake of S. Lazebnik material

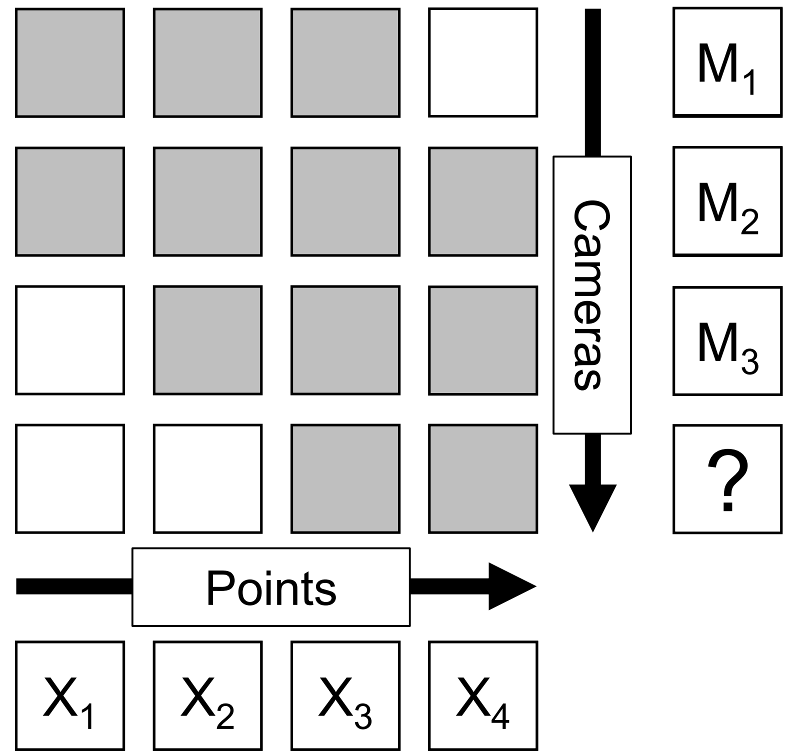
Note: numbers of points aren't to scale.

Incremental SFM

Key idea: incrementally add cameras, points

Big problem: don't ever jointly consider all the 3D points and camera.

Leads to final step, called bundle adjustment.



Remake of S. Lazebnik material

Note: numbers of points aren't to scale.

Bundle Adjustment

Do non-linear minimization over cameras M_i, points X_j to minimize distance between observed points p_{ij} and projections M_iX_j when they're visible.

$$\arg\min_{M_i, X_j} w_{ij}\, d\left(M_i X_j, p_{ij}\right)^2$$

Visibility flag

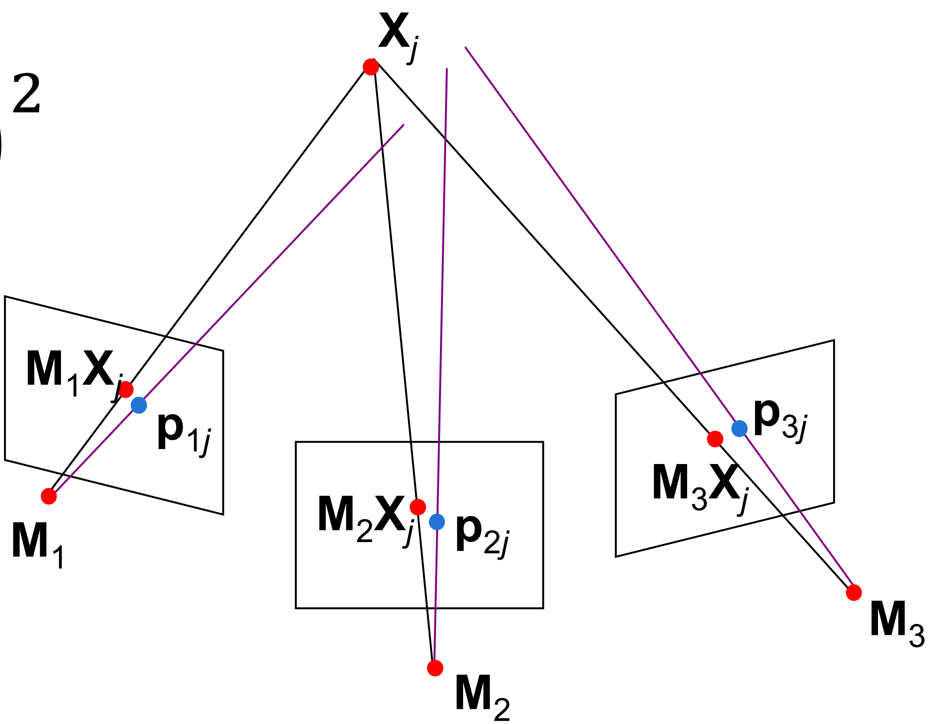


Figure Credit: S. Lazebnik

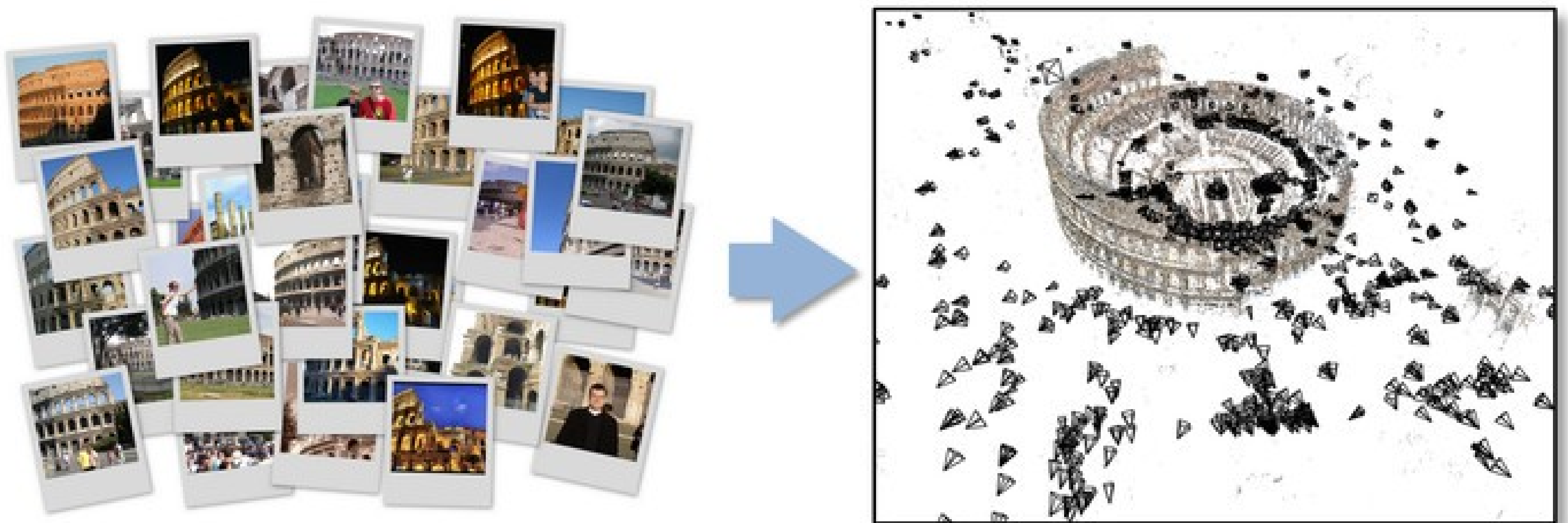
Devil is in the details

High-level idea: $$\arg\min_{M_i, X_j} w_{ij}\, d\left(M_i X_j, p_{ij}\right)^2$$

In practice:

- Have to initialize reasonably well
- Should minimize over K,R,t directly
- Problem is very sparse: w_{ij} almost always zero
- Need to integrate uncertainty information
- Probably want to use a system written by experts

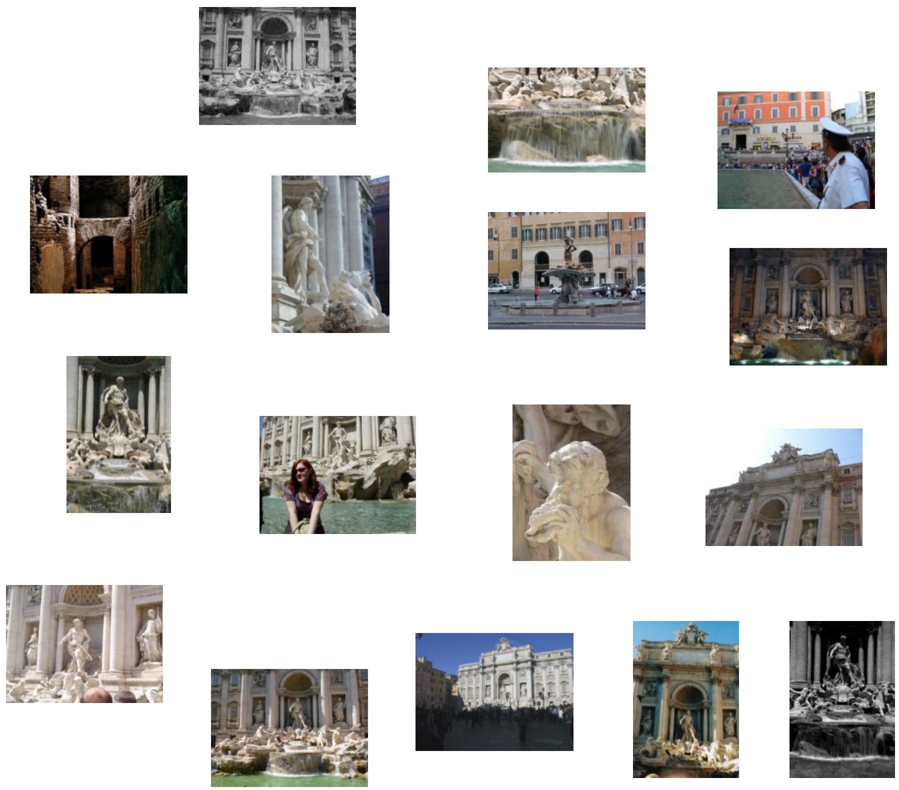
Representative SFM pipeline



N. Snavely, S. Seitz, and R. Szeliski, [Photo tourism: Exploring photo collections in 3D](http://phototour.cs.washington.edu/), SIGGRAPH 2006.
http://phototour.cs.washington.edu/

Feature detection

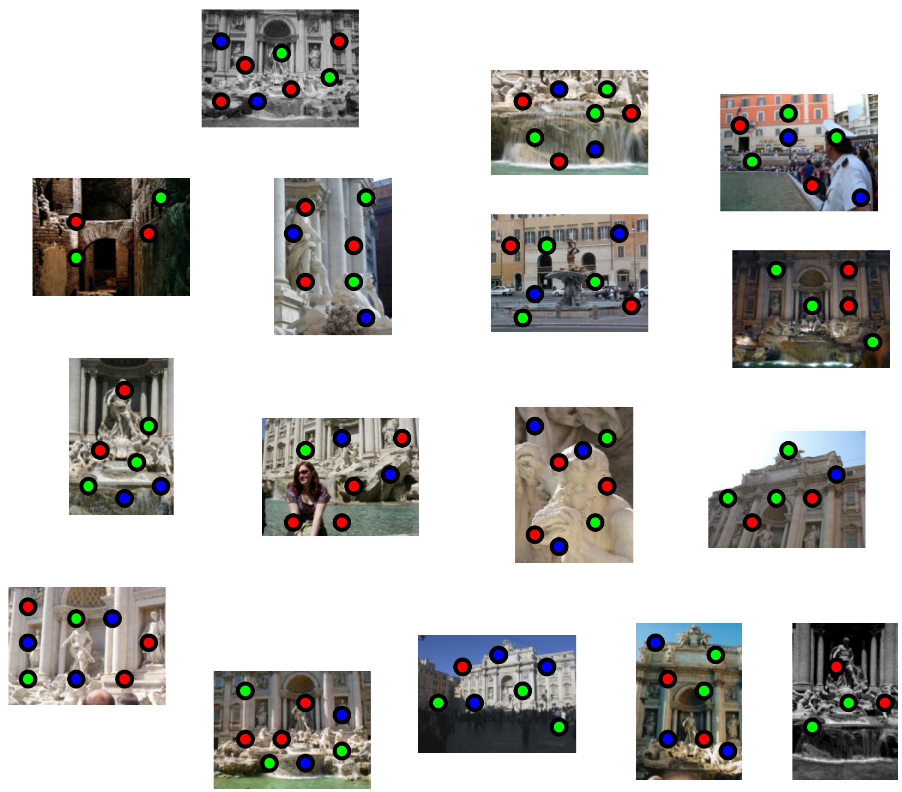
Detect SIFT features



Source: N. Snavely

Feature detection

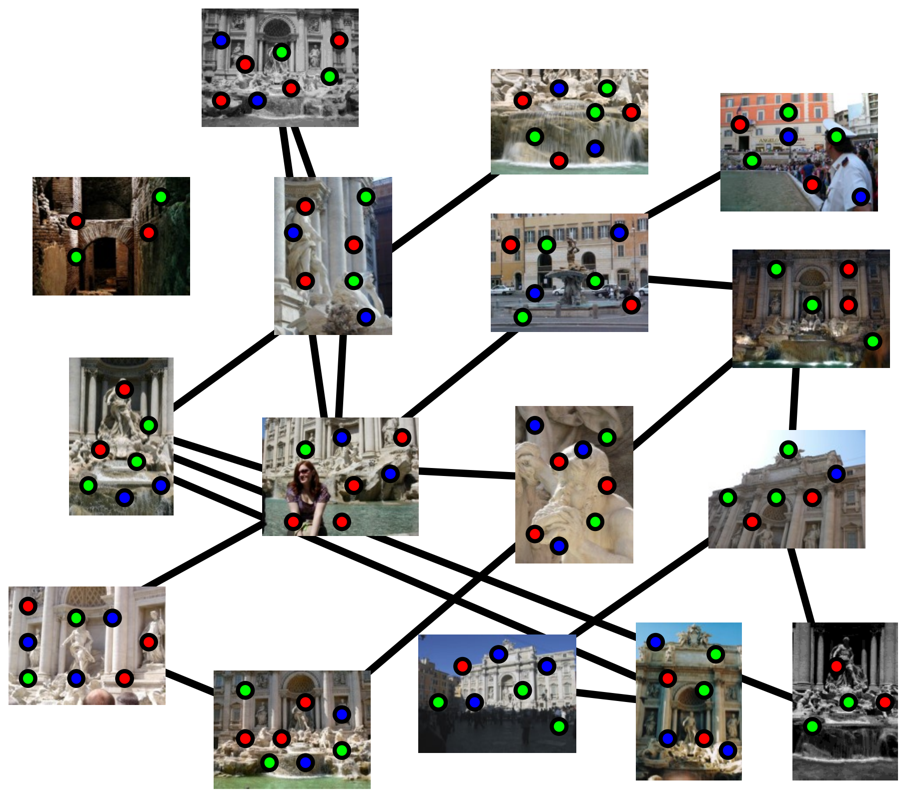
Detect SIFT features



Source: N. Snavely

Feature matching

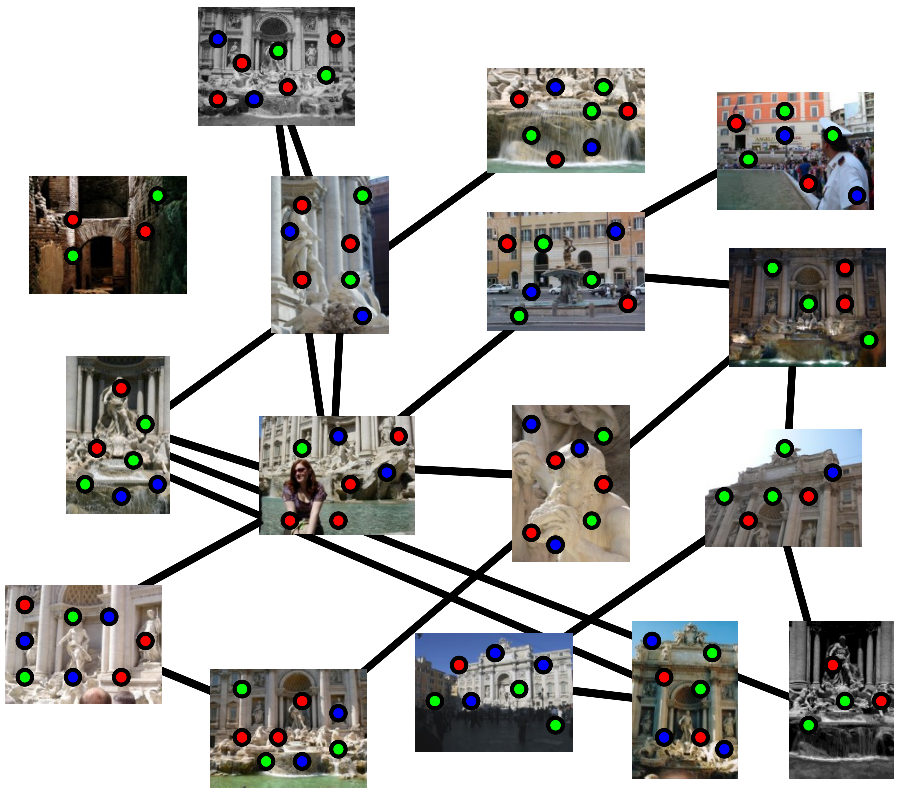
Match features between each pair of images



Source: N. Snavely

Feature matching

Use RANSAC to estimate fundamental matrix between each pair



Source: N. Snavely

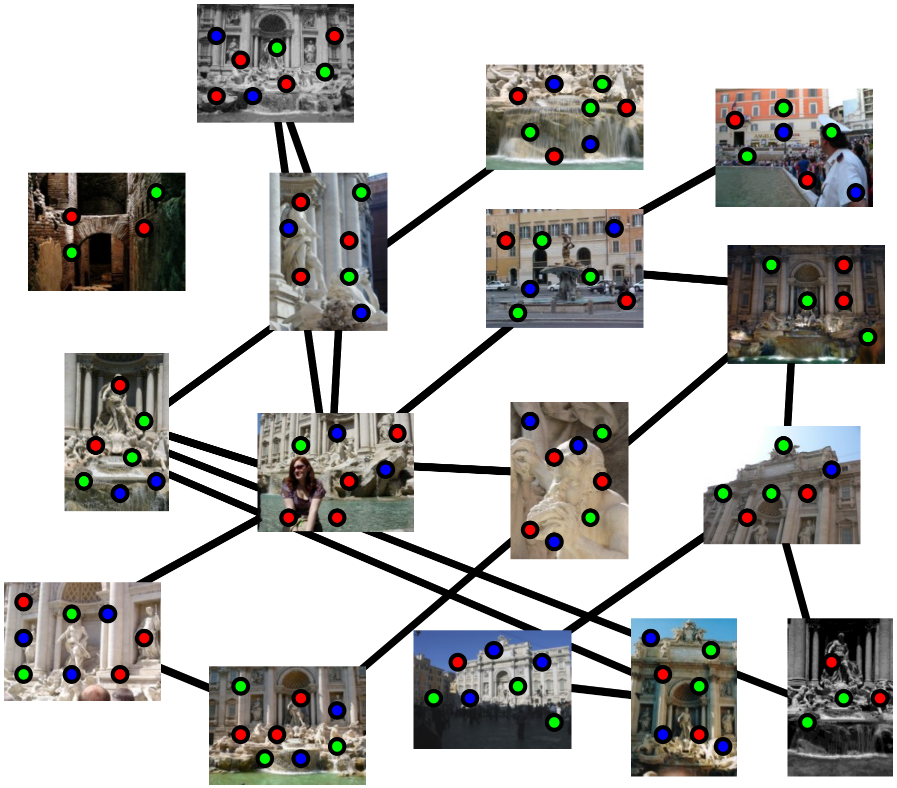
Feature matching

Use RANSAC to estimate fundamental matrix between each pair

Image source

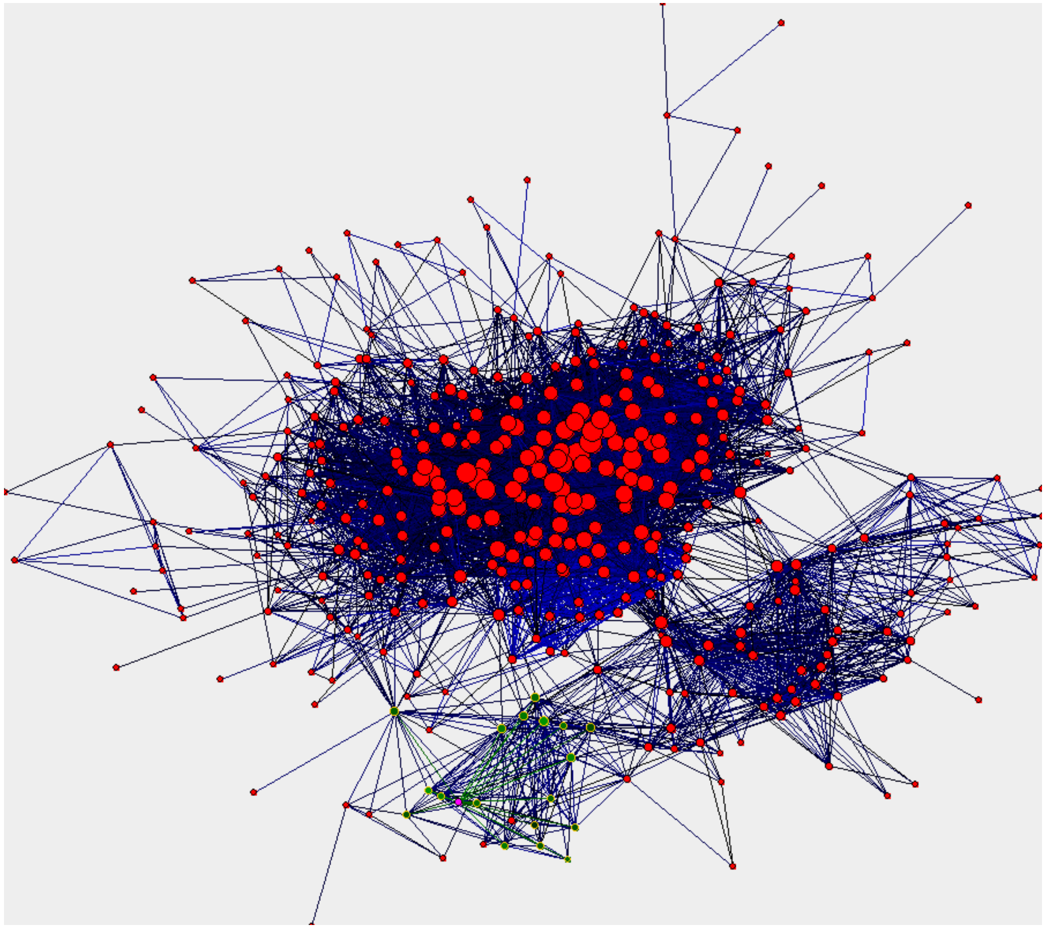
Feature matching

Use RANSAC to estimate fundamental matrix between each pair



Source: N. Snavely

Image connectivity graph



(graph layout produced using the Graphviz toolkit: http://www.graphviz.org/)

Source: N. Snavely

In practice

- Pick a pair of images with lots of inliers (and preferably, good EXIF data)
  - Initialize intrinsic parameters (focal length, principal point) from EXIF
  - Estimate extrinsic parameters (**R** and **t**) Use triangulation to initialize model points
- While remaining images exist
  - Find an image with many feature matches with images in the model
  - Run RANSAC on feature matches to register new image to model
  - Triangulate new points
  - Perform bundle adjustment to re-optimize everything

Source: N. Snavely

The devil is in the details

- Degenerate configurations (homographies)
- Eliminating outliers
- Repetition and symmetry

Slide Credit: S. Lazebnik

The devil is in the details

- Degenerate configurations (homographies)
- Eliminating outliers
- Repetition and symmetry
- Multiple connected components

Slide Credit: S. Lazebnik