Lecture 6: Multi-view Stereo & Structure from Motion

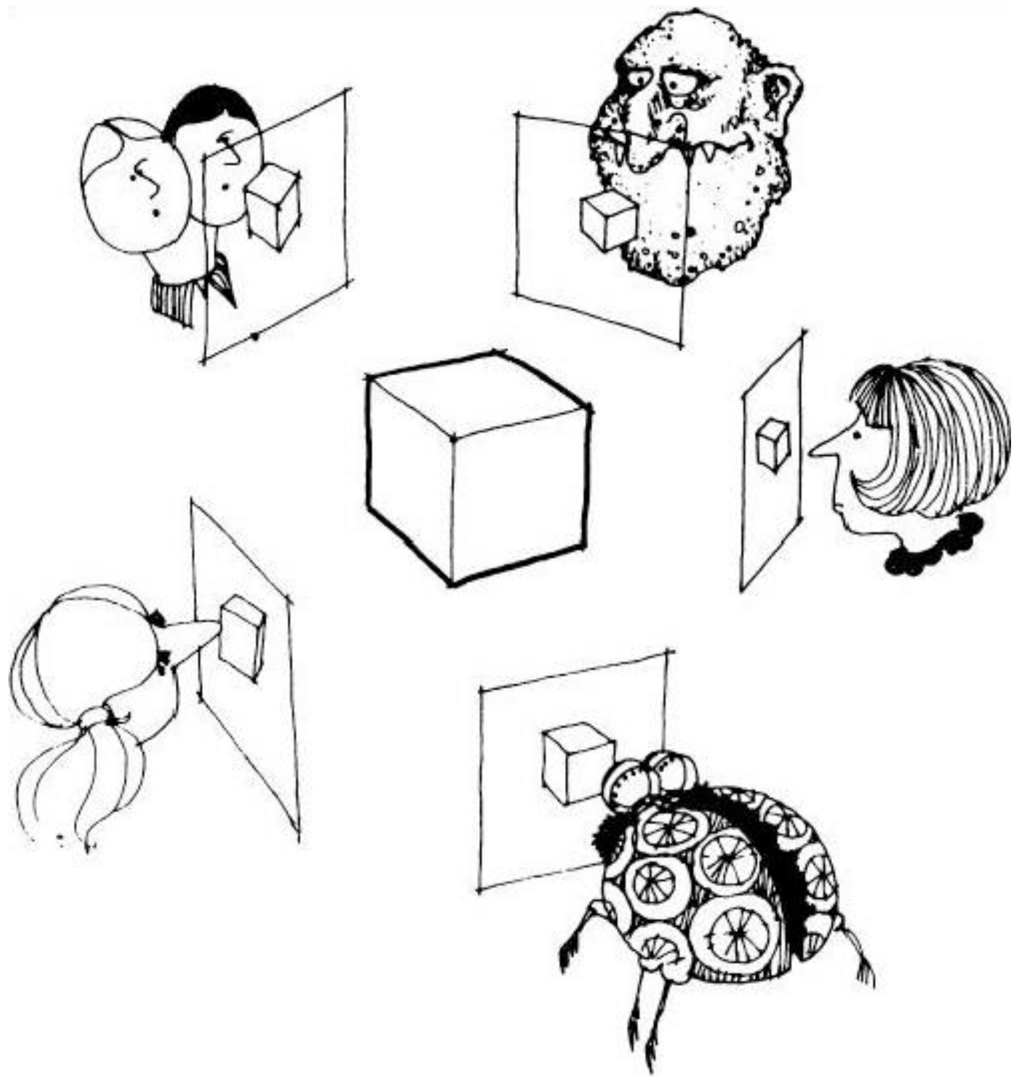
Prof. Rob Fergus

Many slides adapted from Lana Lazebnik and Noah Snavelly, who in turn adapted slides from Steve Seitz, Rick Szeliski, Martial Hebert, Mark Pollefeys, and others….

Overview

- Multi-view stereo
- Structure from Motion (SfM)
- Large scale Structure from Motion

Multi-view stereo



Slides from S. Lazebnik who adapted many from S. Seitz

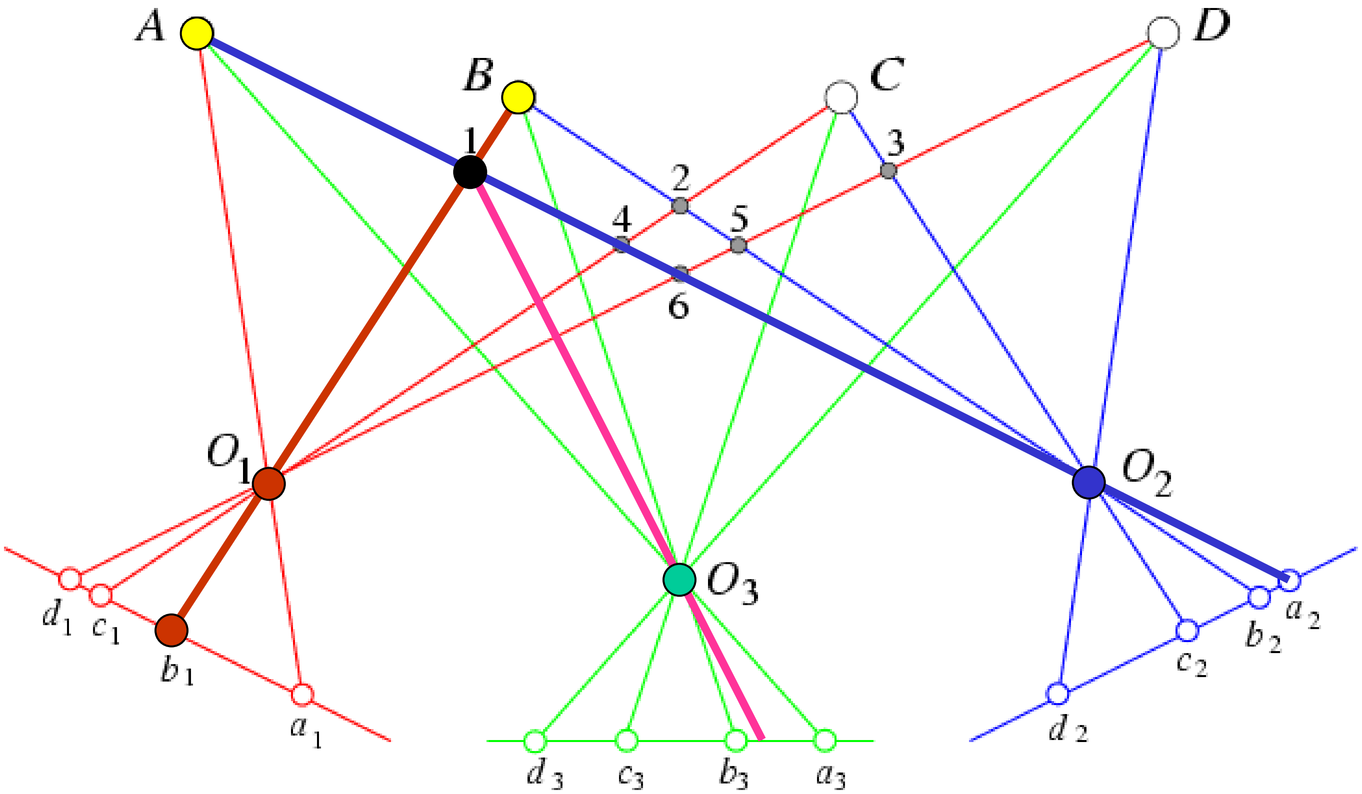
What is stereo vision?

- Generic problem formulation: given several images of the same object or scene, compute a representation of its 3D shape

What is stereo vision?

- Generic problem formulation: given several images of the same object or scene, compute a representation of its 3D shape
- "Images of the same object or scene"
  - Arbitrary number of images (from two to thousands)
  - Arbitrary camera positions (isolated cameras or video sequence)
  - Cameras can be calibrated or uncalibrated
- "Representation of 3D shape"
  - Depth maps
  - Meshes
  - Point clouds
  - Patch clouds
  - Volumetric models
  - Layered models

Beyond two-view stereo

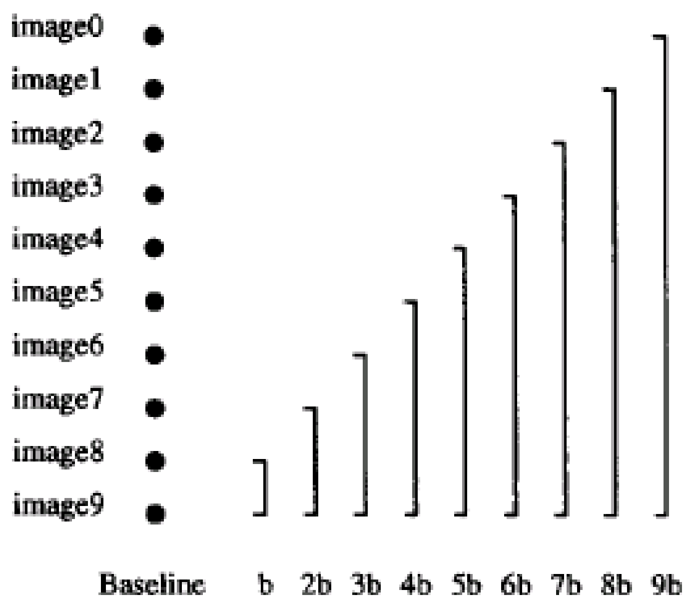


The third view can be used for verification

Multiple-baseline stereo

- Pick a reference image, and slide the corresponding window along the corresponding epipolar lines of all other images, using inverse depth relative to the first image as the search parameter

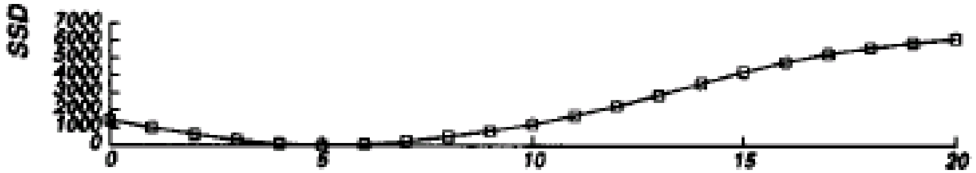
Figure 2: An example scene. The grid pattern in the background has ambiguity of matching.



M. Okutomi and T. Kanade, [“A Multiple-Baseline Stereo System,”](#) IEEE Trans. on Pattern Analysis and Machine Intelligence, 15(4):353-363 (1993).

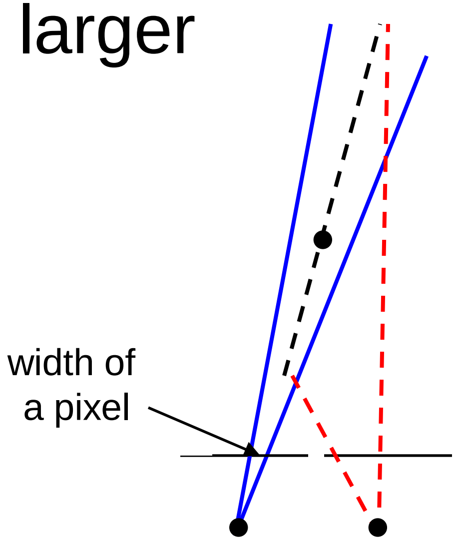
Multiple-baseline stereo

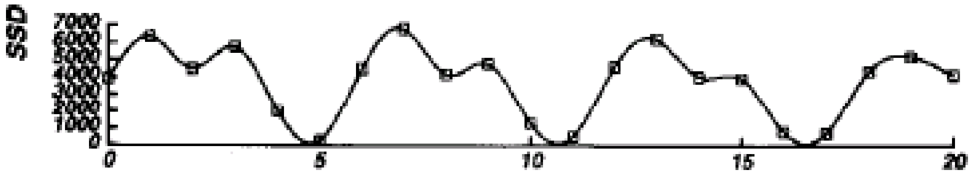
- For larger baselines, must search larger area in second image



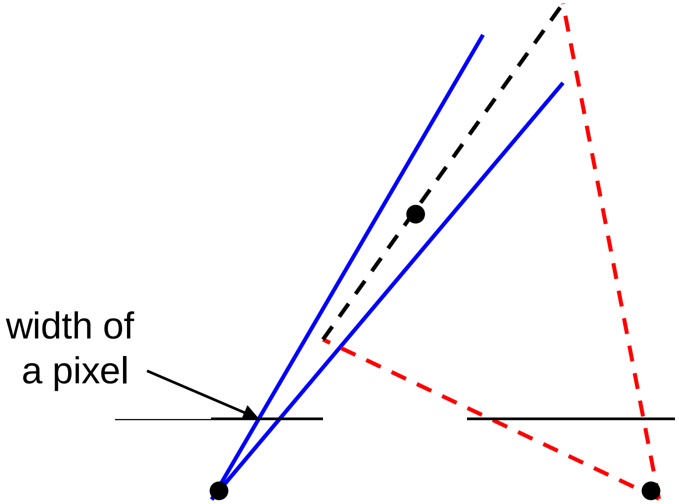
1/z

pixel matching score





1/z



Multiple-baseline stereo

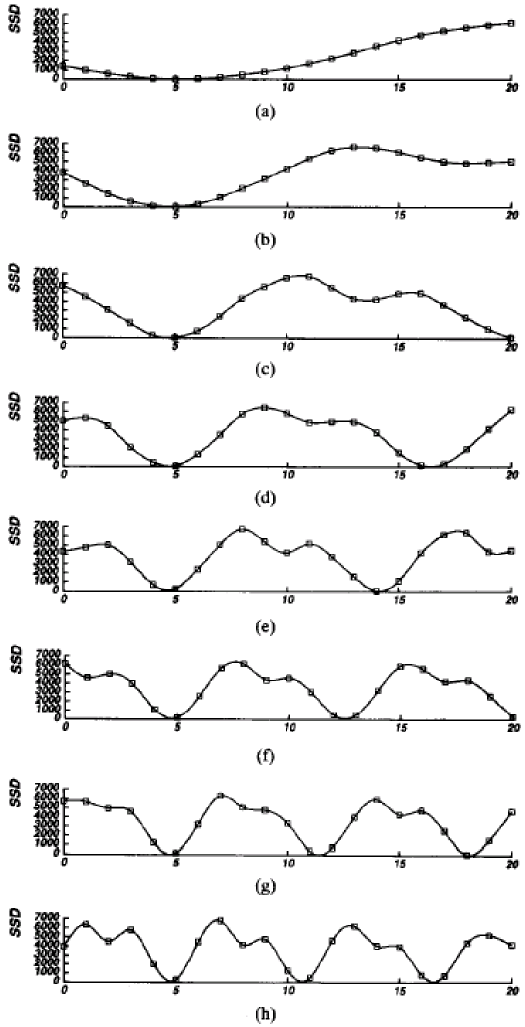


Fig. 5. SSD values versus inverse distance: (a) $B = b$; (b) $B = 2b$; (c) $B = 3b$; (d) $B = 4b$; (e) $B = 5b$; (f) $B = 6b$; (g) $B = 7b$; (h) $B = 8b$. The horizontal axis is normalized such that $8bF = 1$.

Use the sum of SSD scores to rank matches

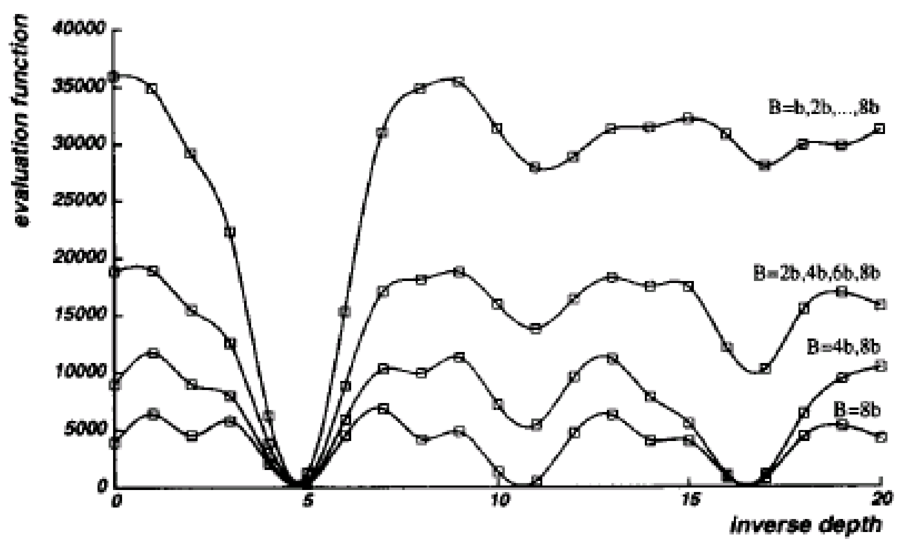


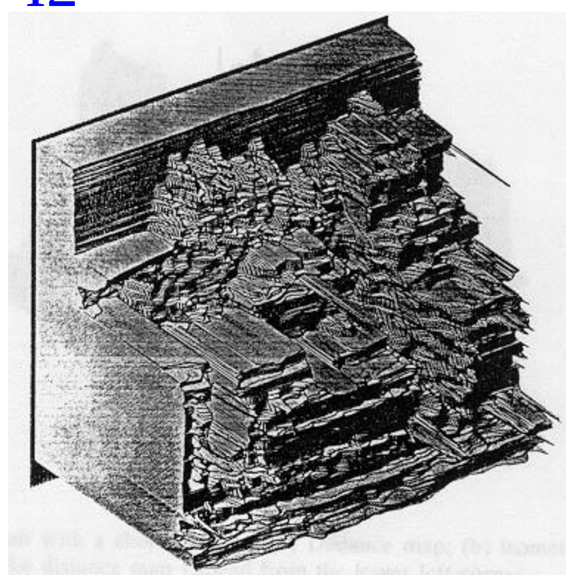
Fig. 7. Combining multiple baseline stereo pairs.

Multiple-baseline stereo results

I1

I2

I10



M. Okutomi and T. Kanade, "A Multiple-Baseline Stereo System," IEEE Trans. on Pattern Analysis and Machine Intelligence, 15(4):353-363 (1993).

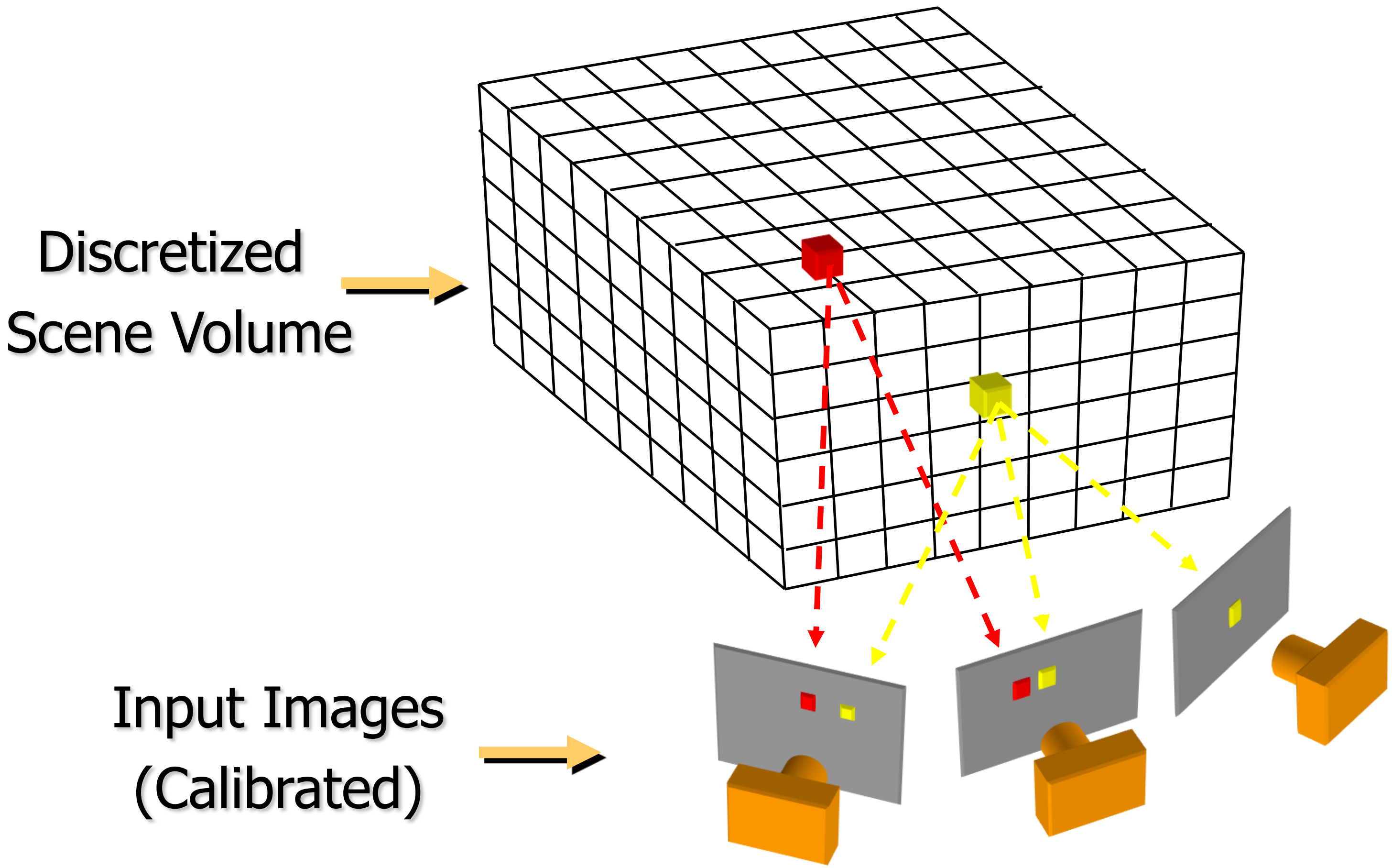
Summary: Multiple-baseline stereo

- Pros
  - Using multiple images reduces the ambiguity of matching
- Cons
  - Must choose a reference view
  - Occlusions become an issue for large baseline
- Possible solution: use a *virtual view*

Volumetric stereo

- In plane sweep stereo, the sampling of the scene still depends on the reference view
- We can use a voxel volume to get a view-independent representation

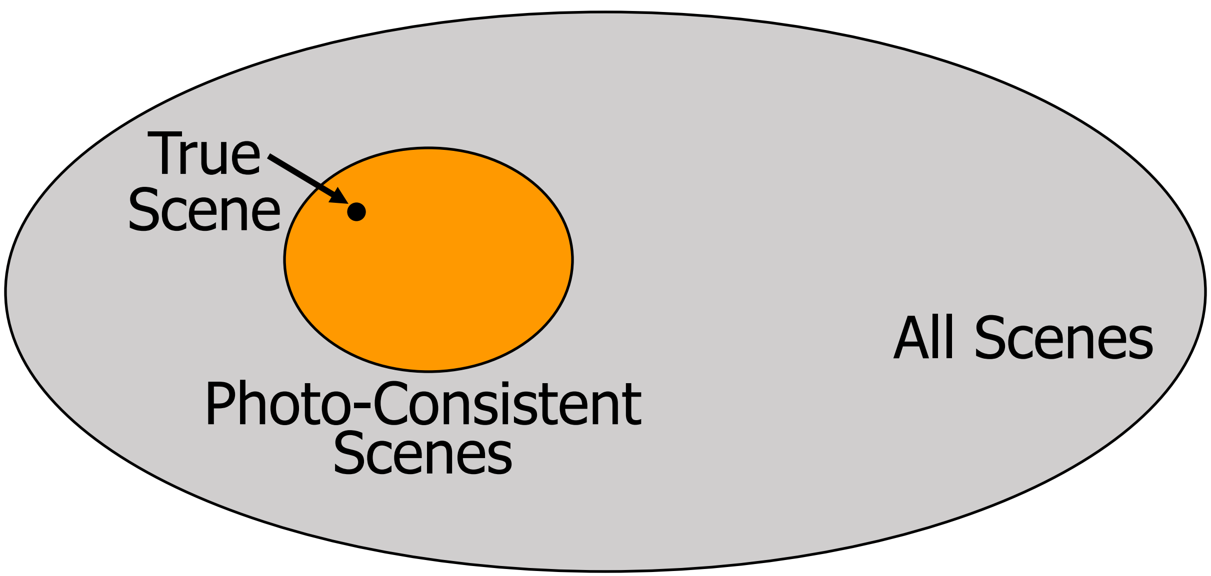
Volumetric Stereo / Voxel Coloring



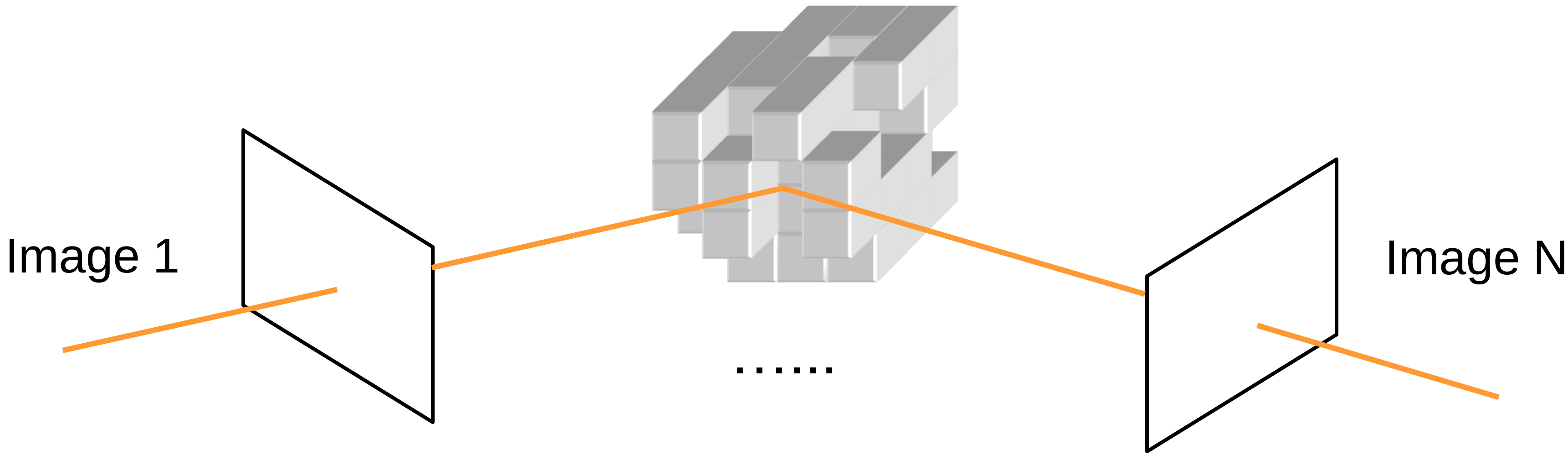
Goal: **Assign RGB values to voxels in V**
photo-consistent **with images**

Photo-consistency

- A *photo-consistent scene* is a scene that exactly reproduces your input images from the same camera viewpoints
- You can't use your input cameras and images to tell the difference between a photo-consistent scene and the true scene



Space Carving

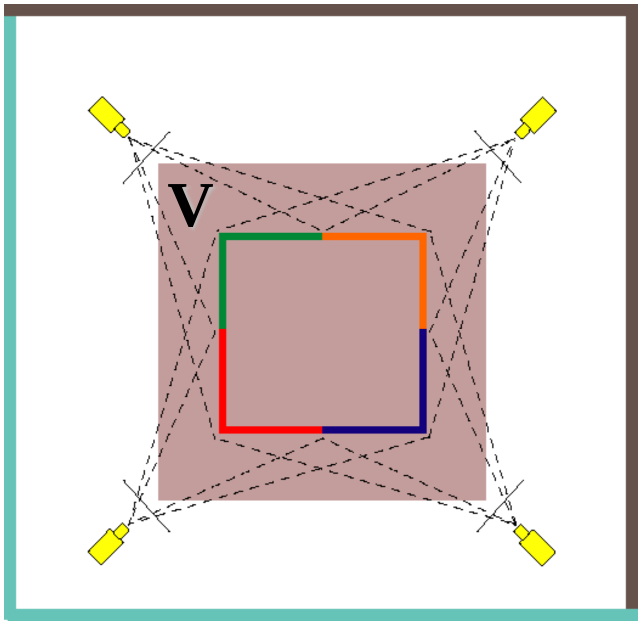


Space Carving Algorithm

- Initialize to a volume V containing the true scene
- Choose a voxel on the current surface
- Project to visible input images
- Carve if not photo-consistent
- Repeat until convergence

K. N. Kutulakos and S. M. Seitz, **A Theory of Shape by Space Carving**, *ICCV* 1999

Which shape do you get?



True Scene

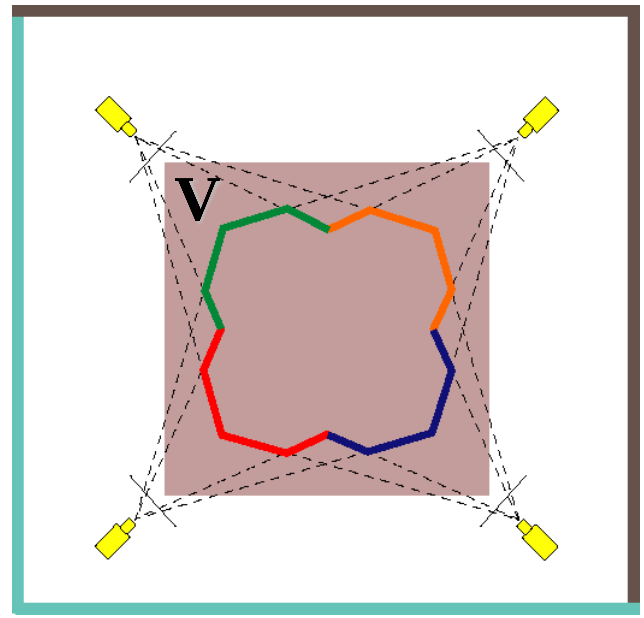


Photo Hull

The Photo Hull *is the UNION of all photo-consistent scenes in V*

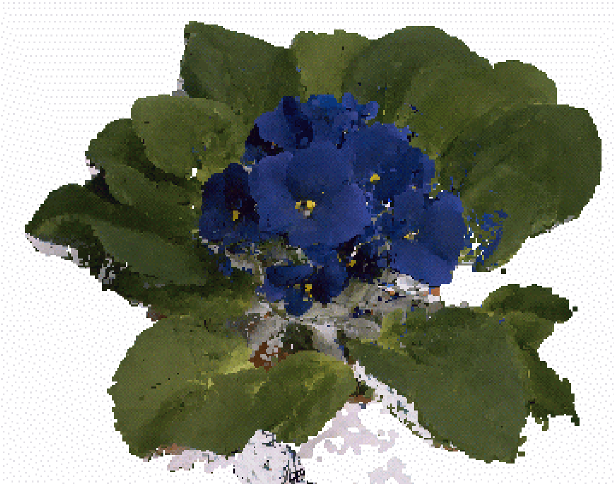
- It is a photo-consistent scene reconstruction
- Tightest possible bound on the true scene

Source: S. Seitz

Space Carving Results: African Violet

Input Image (1 of 45)

Reconstruction



Reconstruction

Reconstruction

Source: S. Seitz

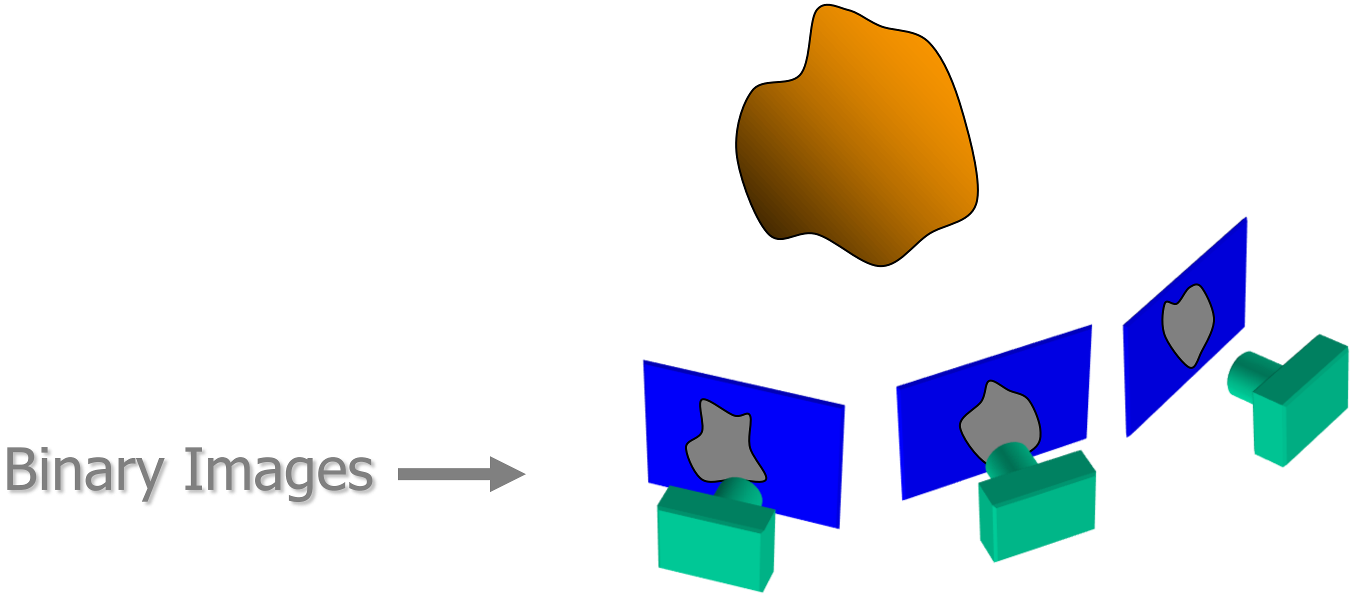
Space Carving Results: Hand

Input Image (1 of 100)

Views of Reconstruction

Reconstruction from Silhouettes

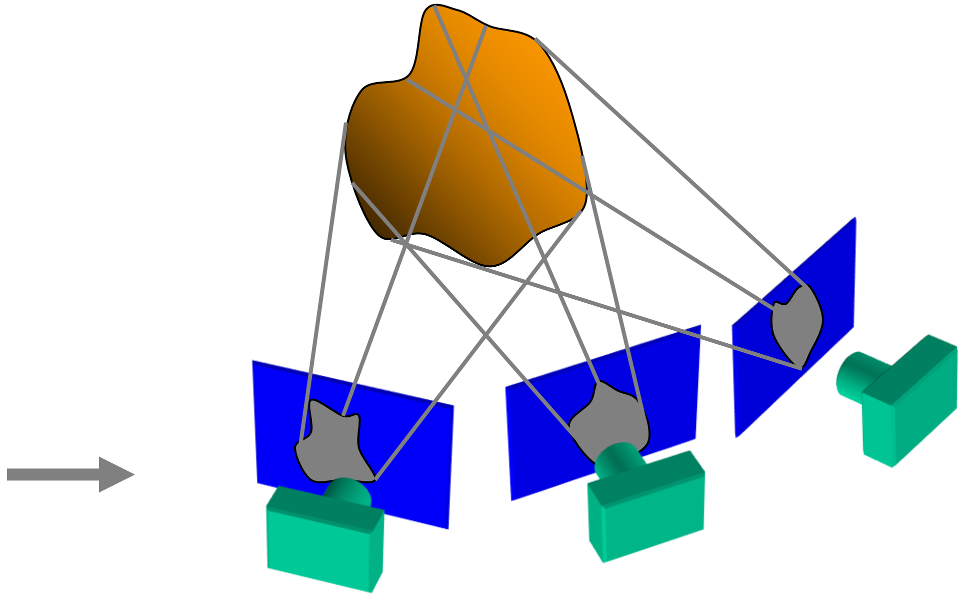
- The case of binary images: a voxel is photo-consistent if it lies inside the object's silhouette in all views



Reconstruction from Silhouettes

- The case of binary images: a voxel is photo-consistent if it lies inside the object's silhouette in all views

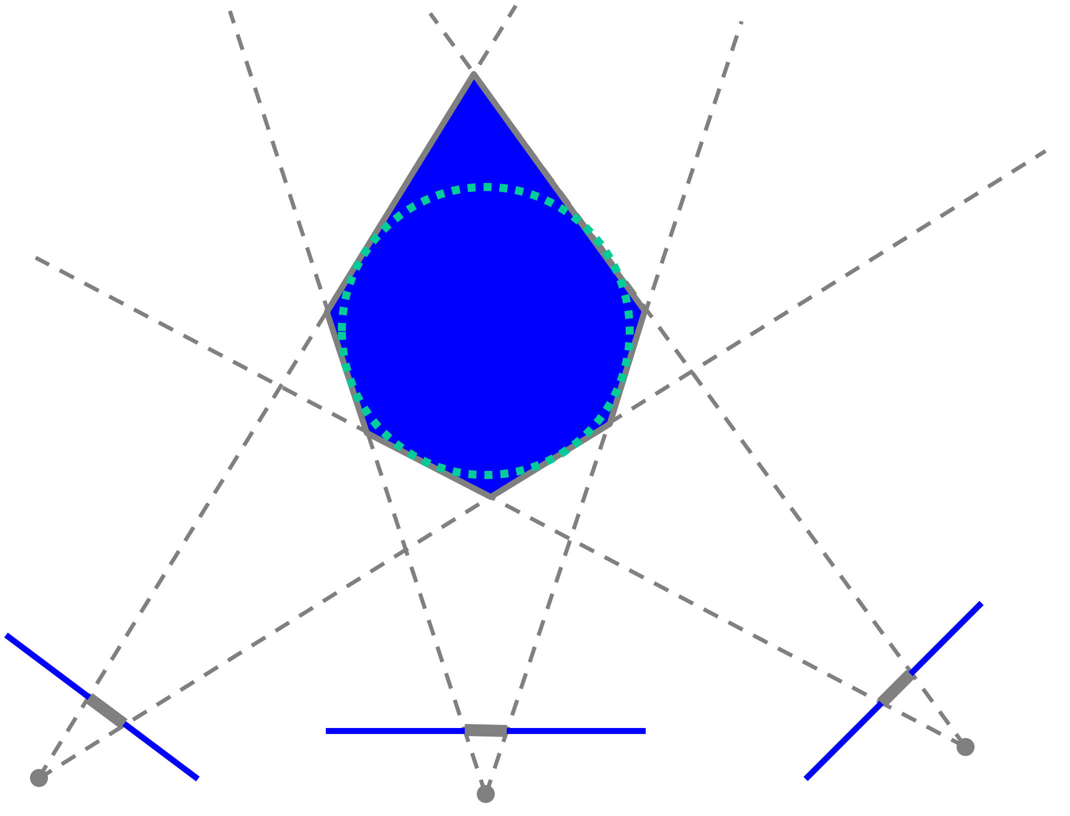
Binary Images



Finding the silhouette-consistent shape (*visual hull*):

- *Backproject* each silhouette
- Intersect backprojected volumes

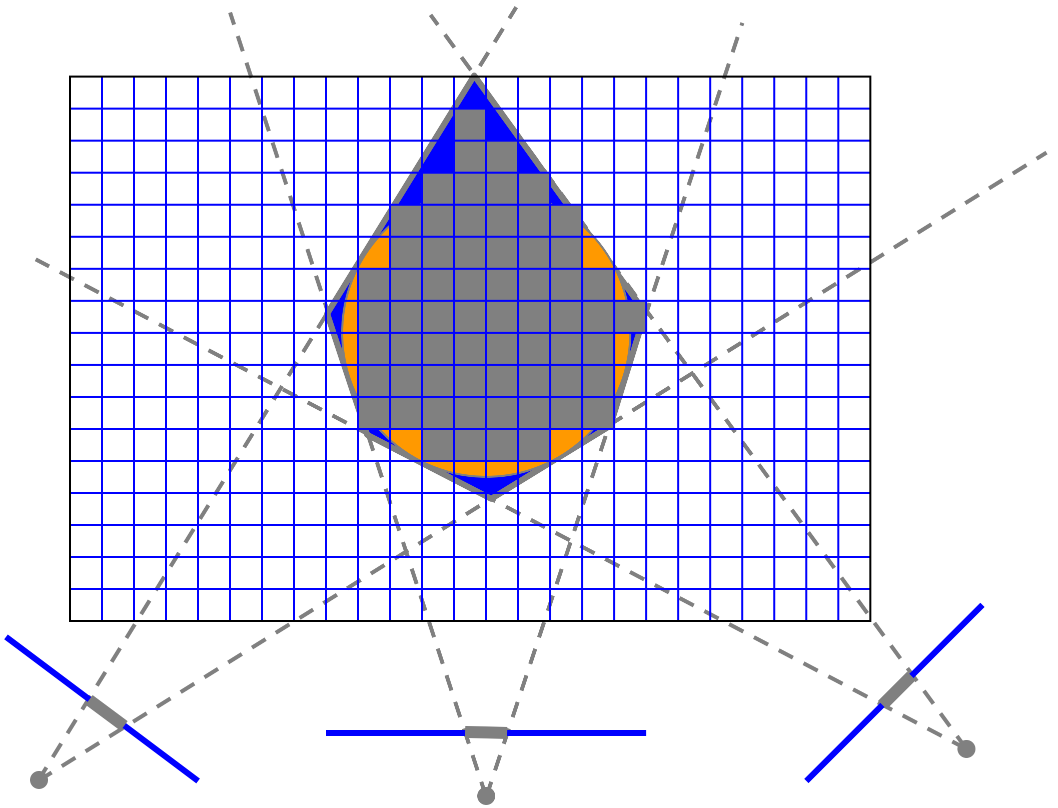
Volume intersection



Reconstruction Contains the True Scene

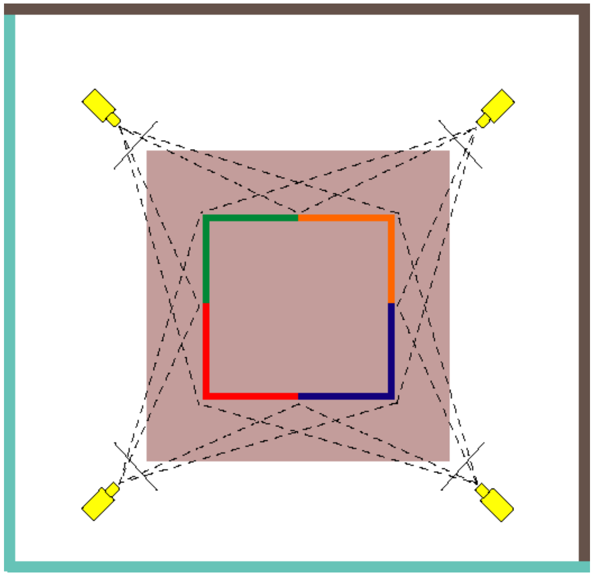
- But is generally not the same

Voxel algorithm for volume intersection



Color voxel black if on silhouette in every image

Photo-consistency vs. silhouette-consistency



True Scene

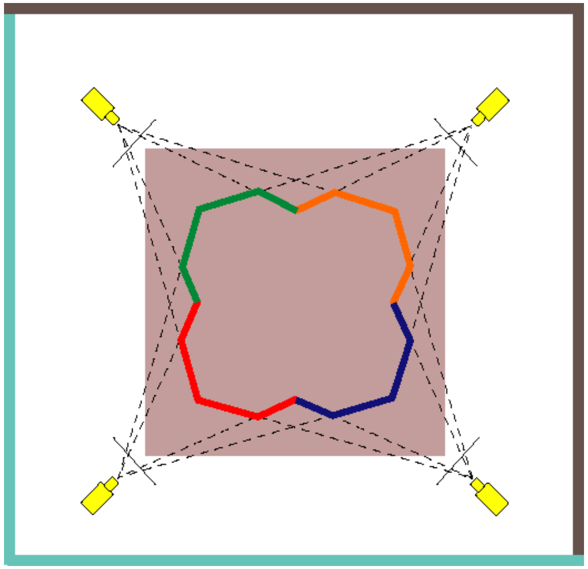
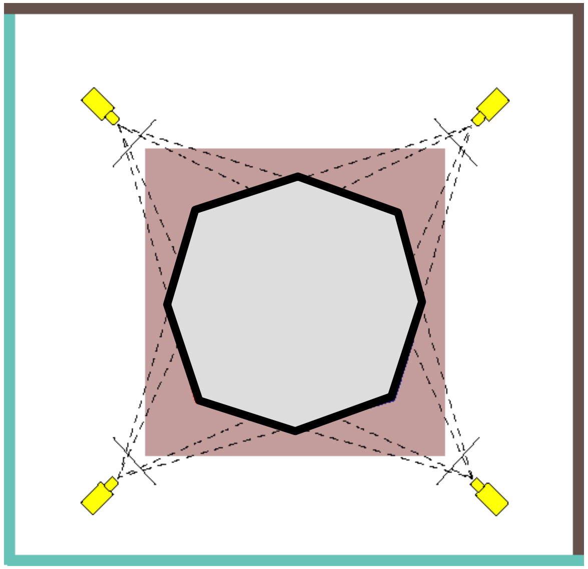


Photo Hull



Visual Hull

Carved visual hulls

- The visual hull is a good starting point for optimizing photo-consistency
  - Easy to compute
  - Tight outer boundary of the object
  - Parts of the visual hull (rims) already lie on the surface and are already photo-consistent

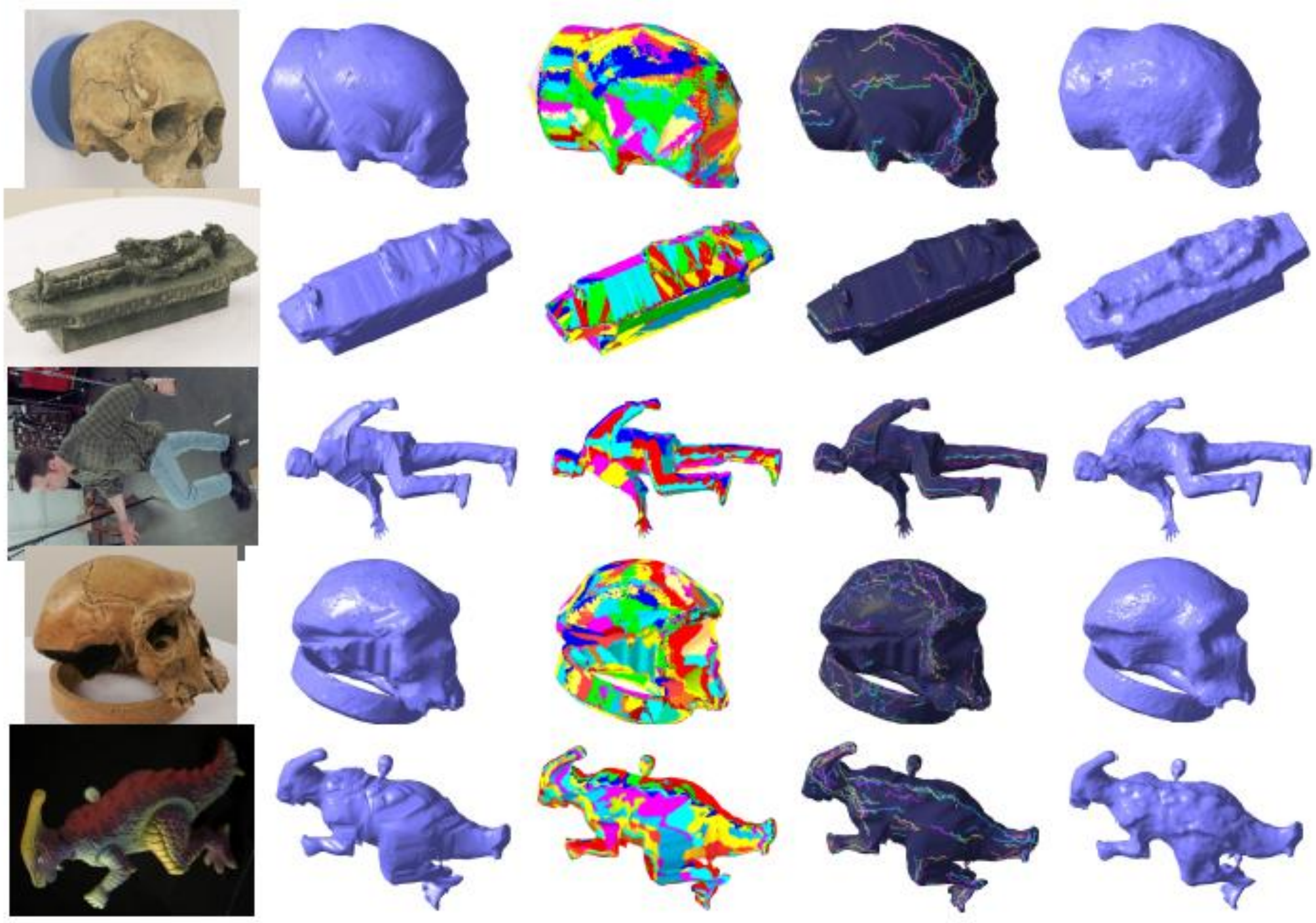
Yasutaka Furukawa and Jean Ponce, **Carved Visual Hulls for Image-Based Modeling**, ECCV 2006.

Carved visual hulls

1. Compute visual hull
2. Use dynamic programming to find rims and constrain them to be fixed
3. Carve the visual hull to optimize photo-consistency

Yasutaka Furukawa and Jean Ponce, **[Carved Visual Hulls for Image-Based Modeling]()**, ECCV 2006.

Carved visual hulls



Yasutaka Furukawa and Jean Ponce, **[Carved Visual Hulls for Image-Based Modeling](#)**, ECCV 2006.

Carved visual hulls: Pros and cons

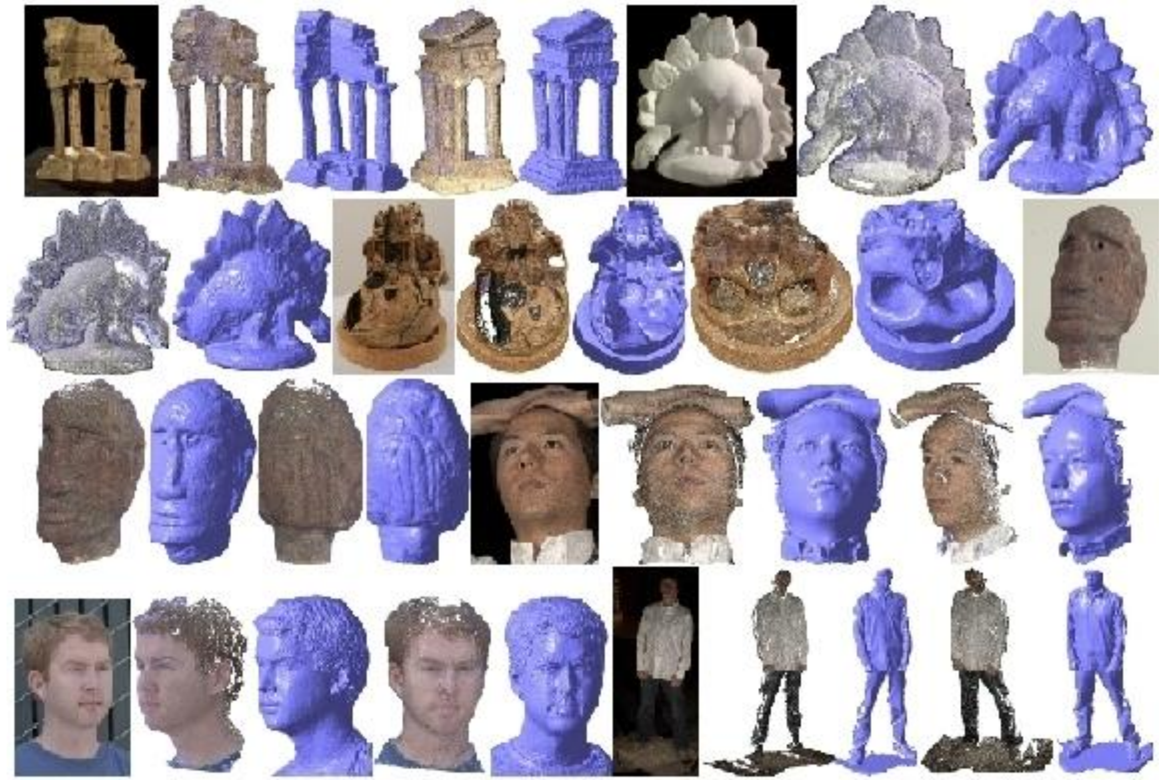
- Pros
  - Visual hull gives a reasonable initial mesh that can be iteratively deformed
- Cons
  - Need silhouette extraction
  - Have to compute a lot of points that don’t lie on the object
  - Finding rims is difficult
  - The carving step can get caught in local minima
- Possible solution: use sparse feature correspondences as initialization

From feature matching to dense stereo

1. Extract features
2. Get a sparse set of initial matches
3. Iteratively expand matches to nearby locations
4. Use visibility constraints to filter out false matches
5. Perform surface reconstruction

Yasutaka Furukawa and Jean Ponce, **Accurate, Dense, and Robust Multi-View Stereopsis**, CVPR 2007.

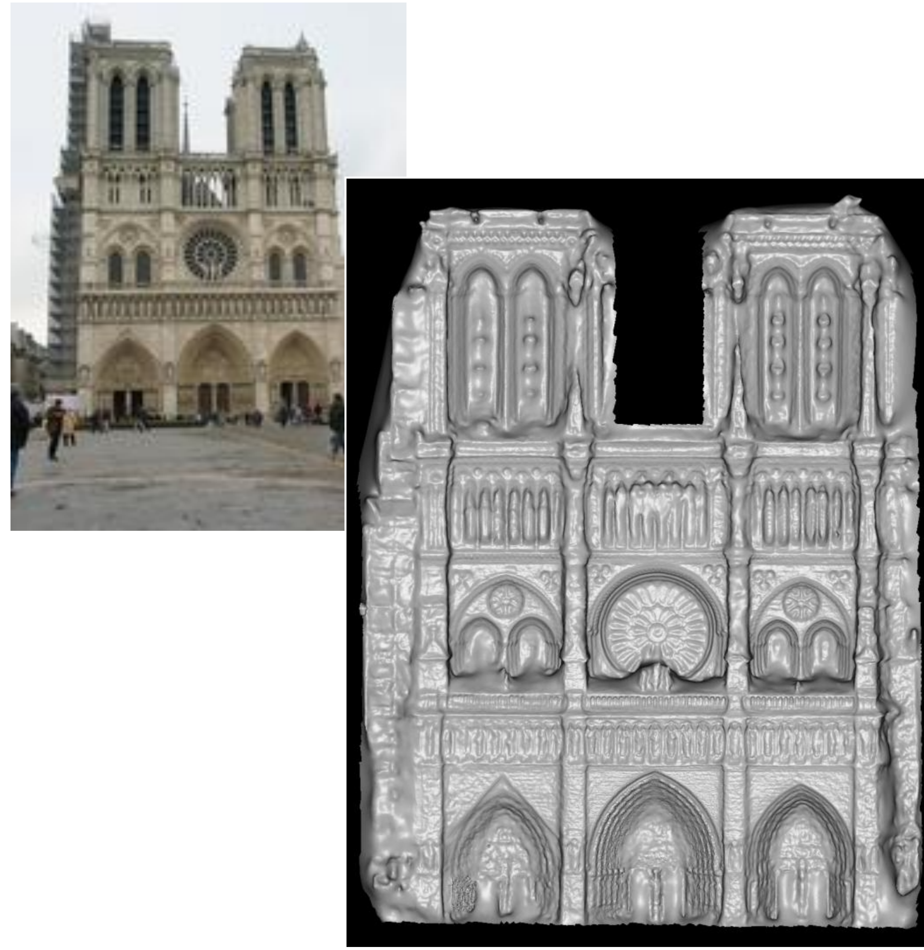
From feature matching to dense stereo



http://www.cs.washington.edu/homes/furukawa/gallery/

Yasutaka Furukawa and Jean Ponce, **Accurate, Dense, and Robust Multi-View Stereopsis**, CVPR 2007.

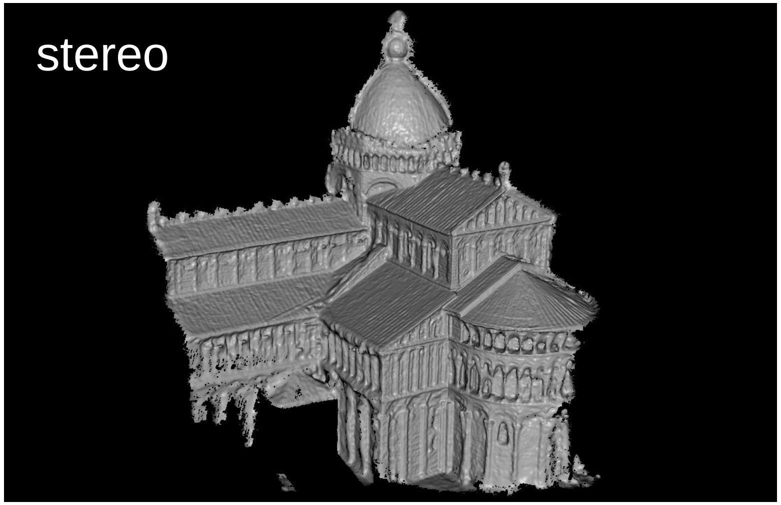
Stereo from community photo collections

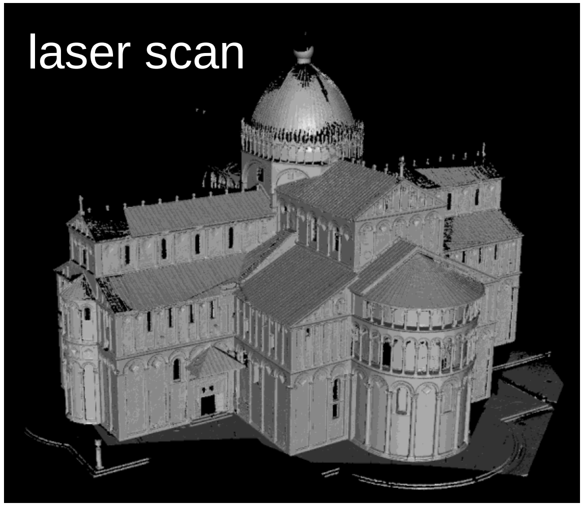


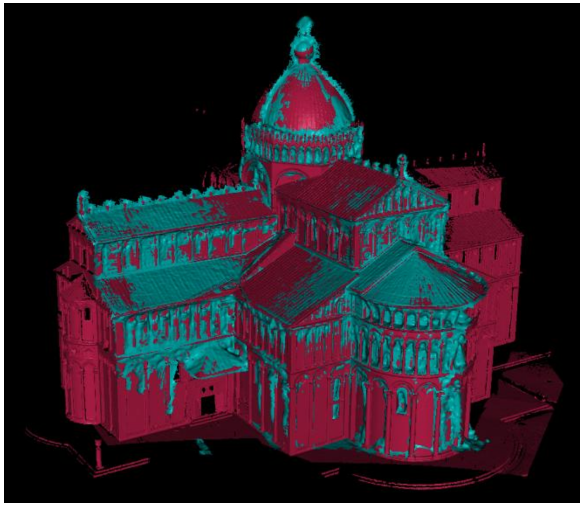
M. Goesele, N. Snavely, B. Curless, H. Hoppe, S. Seitz, Multi-View Stereo for Community Photo Collections, ICCV 2007

http://grail.cs.washington.edu/projects/mvscpc/

Stereo from community photo collections







Comparison: 90% of points within 0.128 m of laser scan (building height 51m)

M. Goesele, N. Snavely, B. Curless, H. Hoppe, S. Seitz, [Multi-View Stereo for Community Photo Collections](), ICCV 2007

Stereo from community photo collections

- Up to now, we've always assumed that camera calibration is known
- For photos taken from the Internet, we need *structure from motion* techniques to reconstruct both camera positions and 3D points

Multi-view stereo: Summary

- Multiple-baseline stereo
  - Pick one input view as reference
  - Inverse depth instead of disparity
- Volumetric stereo
  - Photo-consistency
  - Space carving
- Shape from silhouettes
  - Visual hull: intersection of visual cones
- Carved visual hulls
- Feature-based stereo
  - From sparse to dense correspondences

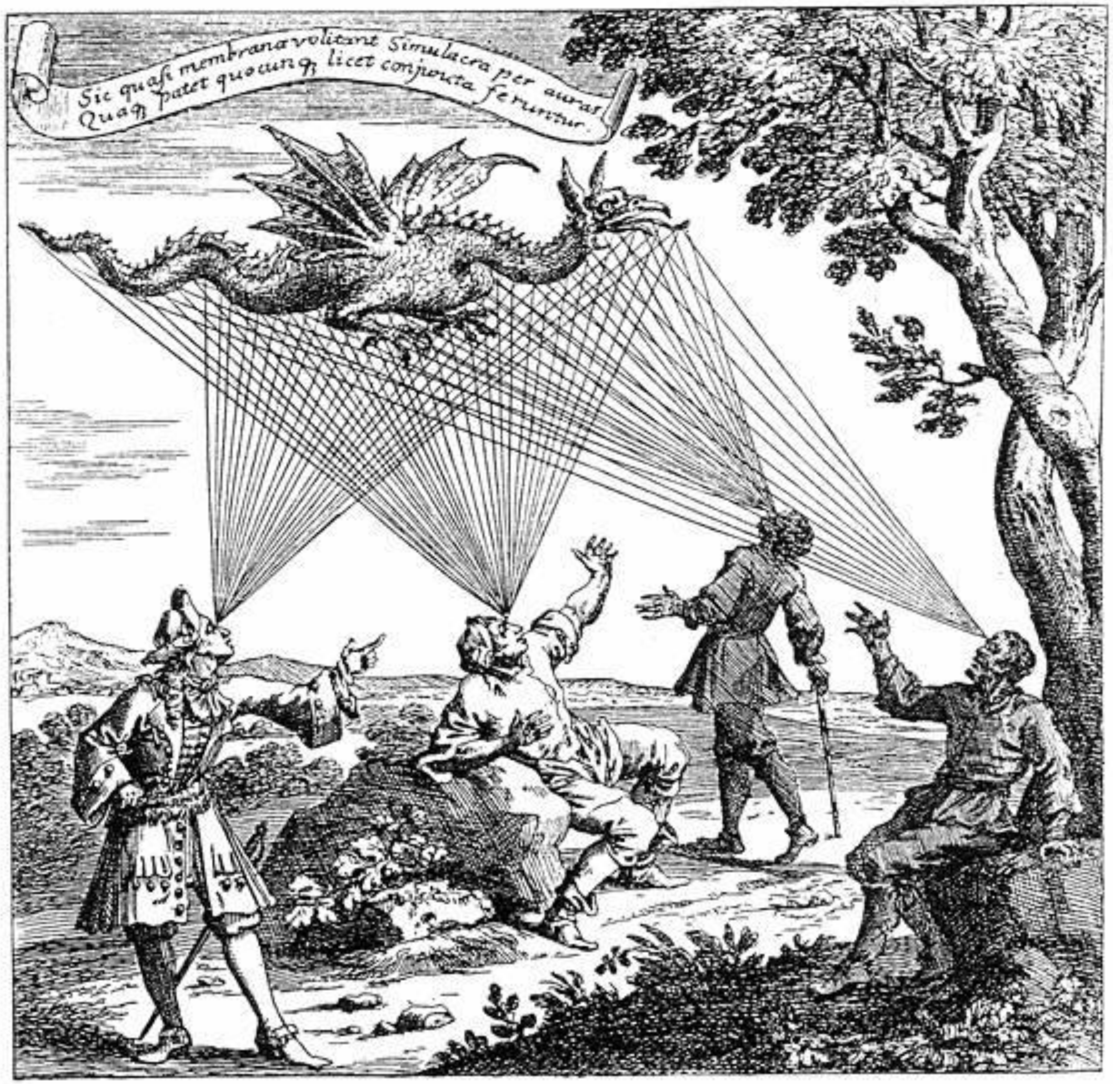
Overview

Multi-view stereo

Structure from Motion (SfM)

Large scale Structure from Motion

Structure from motion



Драконъ, видимый подъ различными углами зрѣнія
По гравюрѣ на мѣди изъ „Oculus artificialis teledioptricus" Цана. 1702 года.

Multiple-view geometry questions

- **Scene geometry (structure):** Given 2D point matches in two or more images, where are the corresponding points in 3D?

- **Correspondence (stereo matching):** Given a point in just one image, how does it constrain the position of the corresponding point in another image?

- **Camera geometry (motion):** Given a set of corresponding points in two or more images, what are the camera matrices for these views?

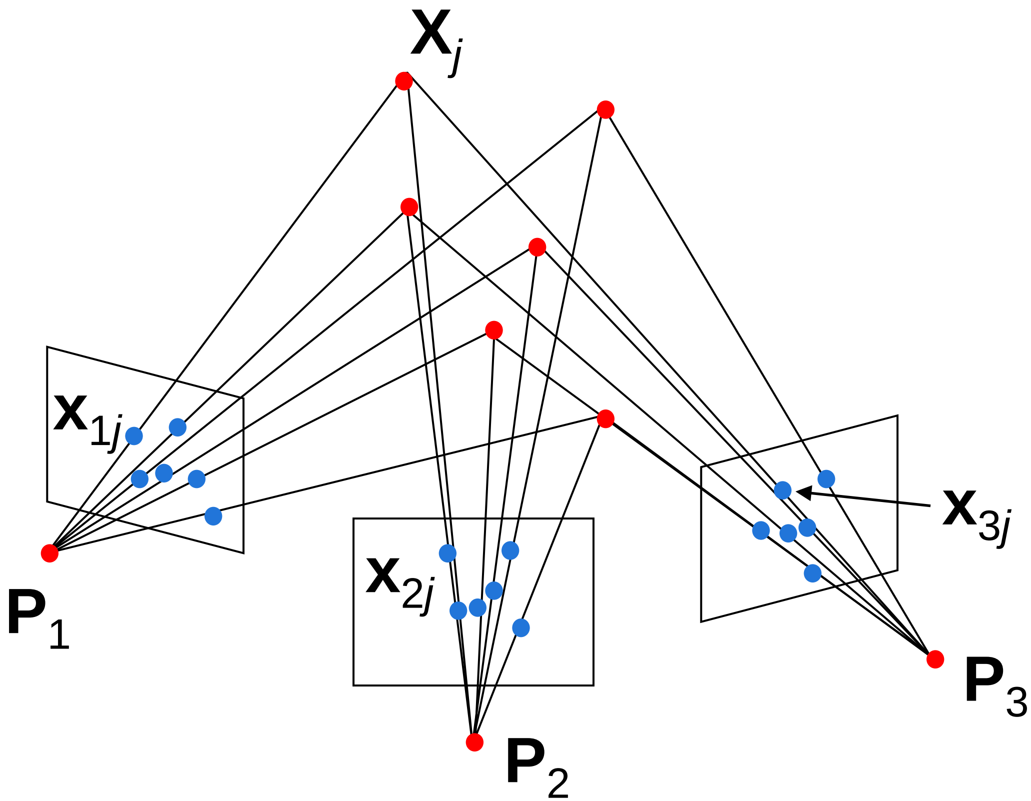
Slide: S. Lazebnik

Structure from motion

- Given: *m* images of *n* fixed 3D points

$$\mathbf{x}_{ij} = \mathbf{P}_i\, \mathbf{X}_j\,, \qquad i = 1, \ldots, m, \quad j = 1, \ldots, n$$

- Problem: estimate *m* projection matrices $\mathbf{P}_i$ and *n* 3D points $\mathbf{X}_j$ from the *mn* correspondences $\mathbf{x}_{ij}$



Slide: S. Lazebnik

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!

Slide: S. Lazebnik

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

- More generally: if we transform the scene using a transformation **Q** and apply the inverse transformation to the camera matrices, then the images do not change

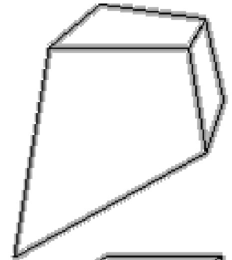
$$\mathbf{x} = \mathbf{P}\mathbf{X} = \left(\mathbf{P}\mathbf{Q}^{-1}\right)\left(\mathbf{Q}\mathbf{X}\right)$$

Slide: S. Lazebnik

Types of ambiguity

Projective
15dof

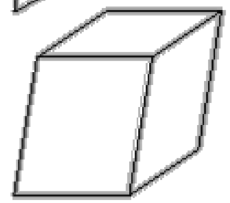
$$\begin{bmatrix} \mathrm{A} & \mathrm{t} \\ \mathrm{v}^{\mathrm{T}} & v \end{bmatrix}$$



Preserves intersection and tangency

Affine
12dof

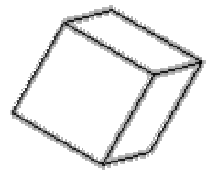
$$\begin{bmatrix} \mathrm{A} & \mathrm{t} \\ 0^{\mathrm{T}} & 1 \end{bmatrix}$$



Preserves parallellism, volume ratios

Similarity
7dof

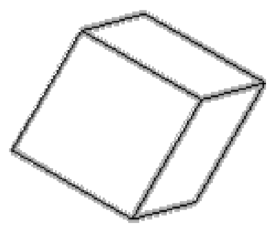
$$\begin{bmatrix} s\,\mathrm{R} & \mathrm{t} \\ 0^{\mathrm{T}} & 1 \end{bmatrix}$$



Preserves angles, ratios of length

Euclidean
6dof

$$\begin{bmatrix} \mathrm{R} & \mathrm{t} \\ 0^{\mathrm{T}} & 1 \end{bmatrix}$$

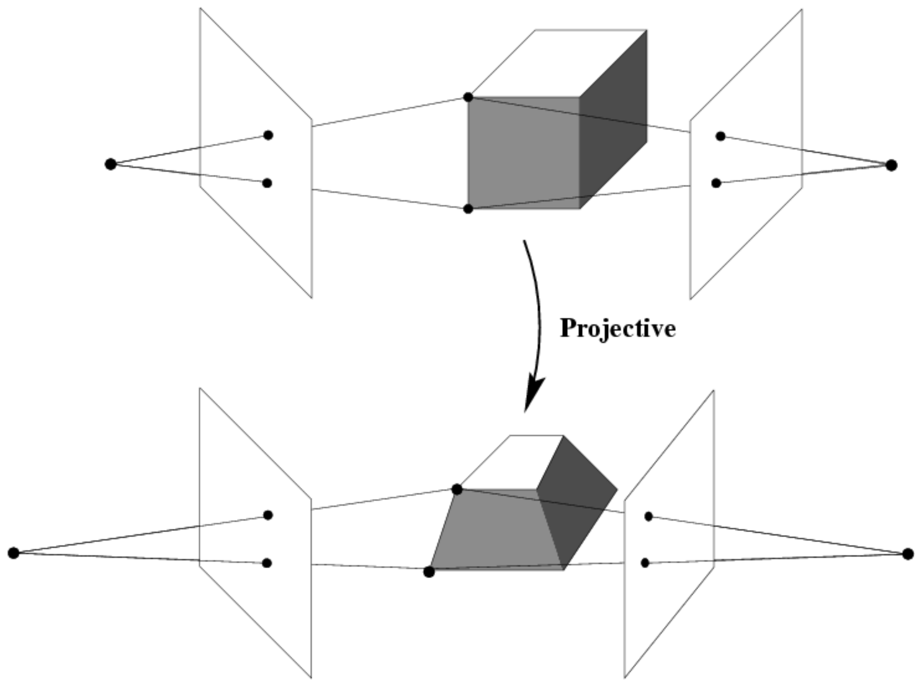


Preserves angles, lengths

- With no constraints on the camera calibration matrix or on the scene, we get a *projective* reconstruction
- Need additional information to *upgrade* the reconstruction to affine, similarity, or Euclidean

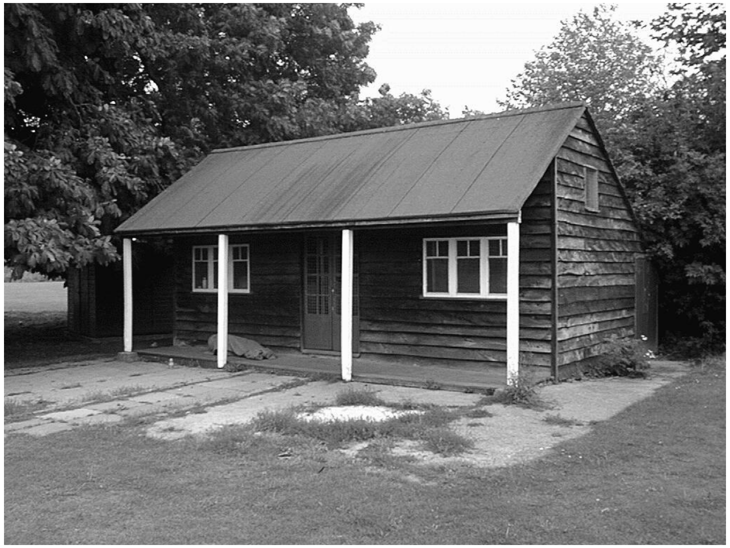
Slide: S. Lazebnik

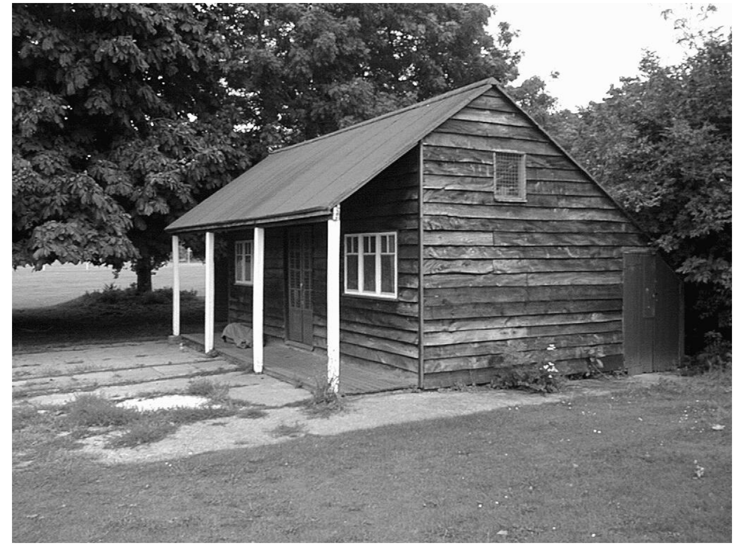
Projective ambiguity

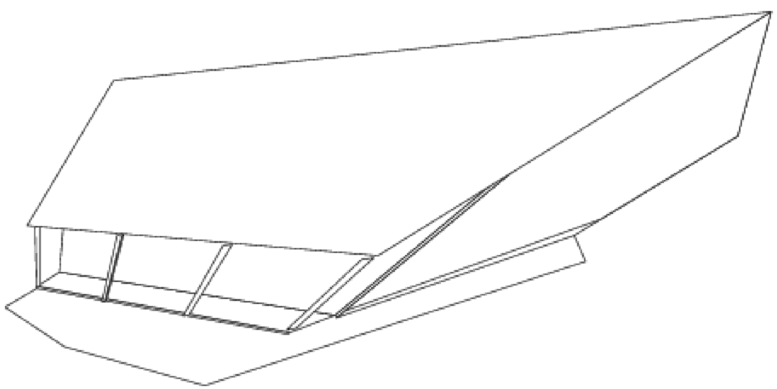


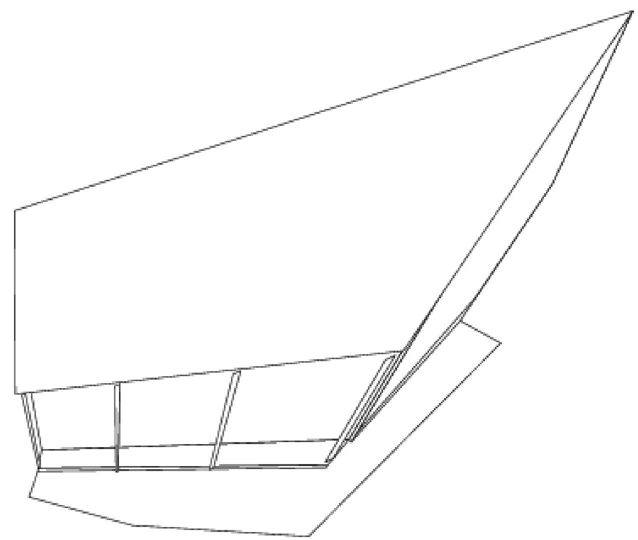
$$\mathbf{x} = \mathbf{P}\mathbf{X} = \left(\mathbf{P}\mathbf{Q}_\mathbf{P}^{-1}\right)\left(\mathbf{Q}_\mathbf{P}\mathbf{X}\right)$$

Projective ambiguity

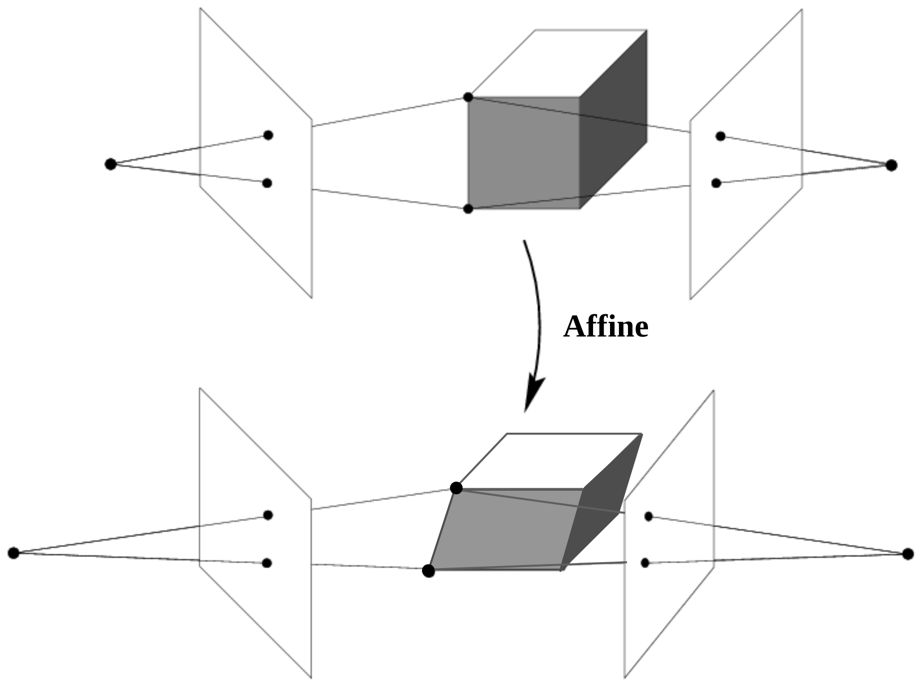






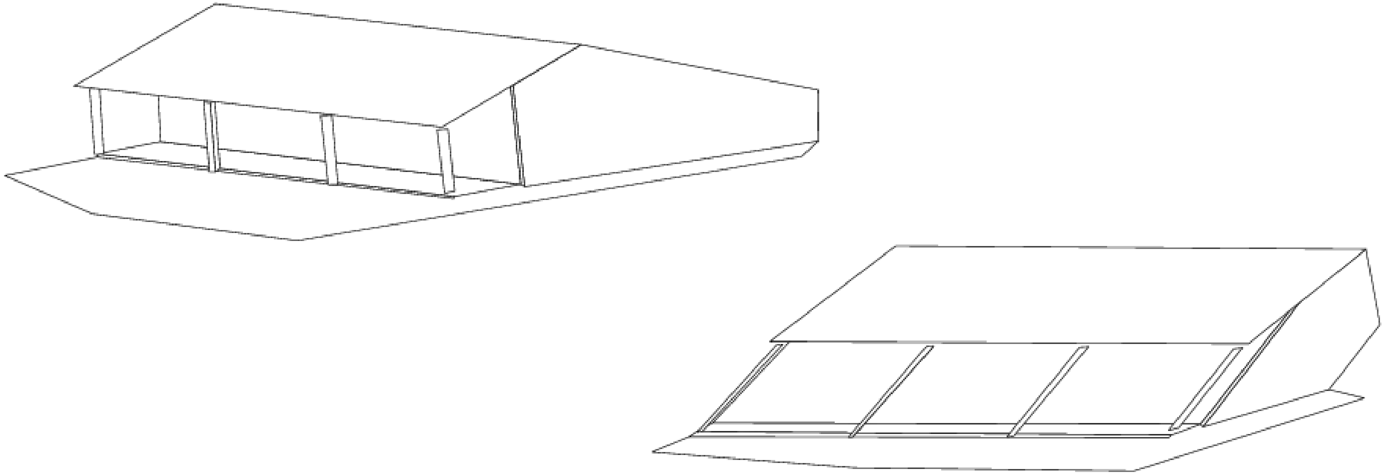


Affine ambiguity

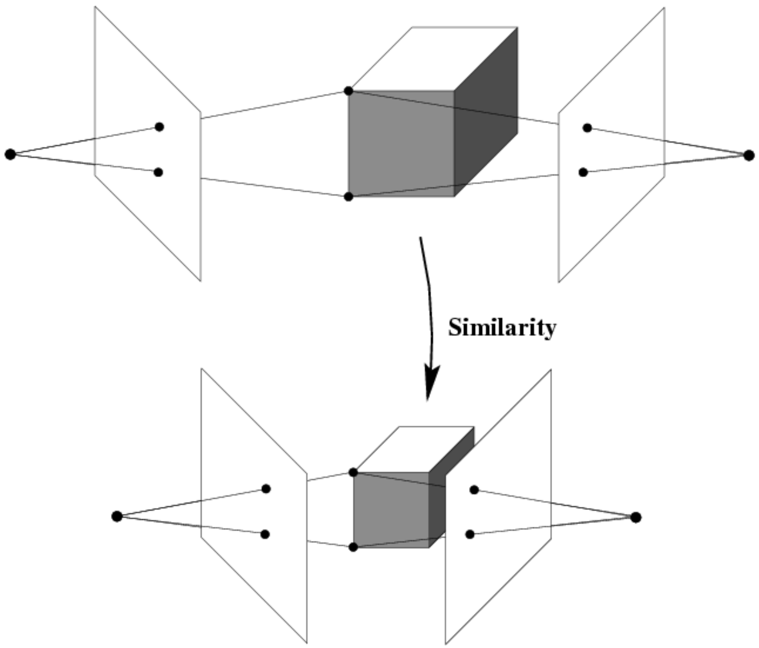


$$\mathbf{x} = \mathbf{P}\mathbf{X} = \left(\mathbf{P}\mathbf{Q}_A^{-1}\right)\left(\mathbf{Q}_A\mathbf{X}\right)$$

Affine ambiguity

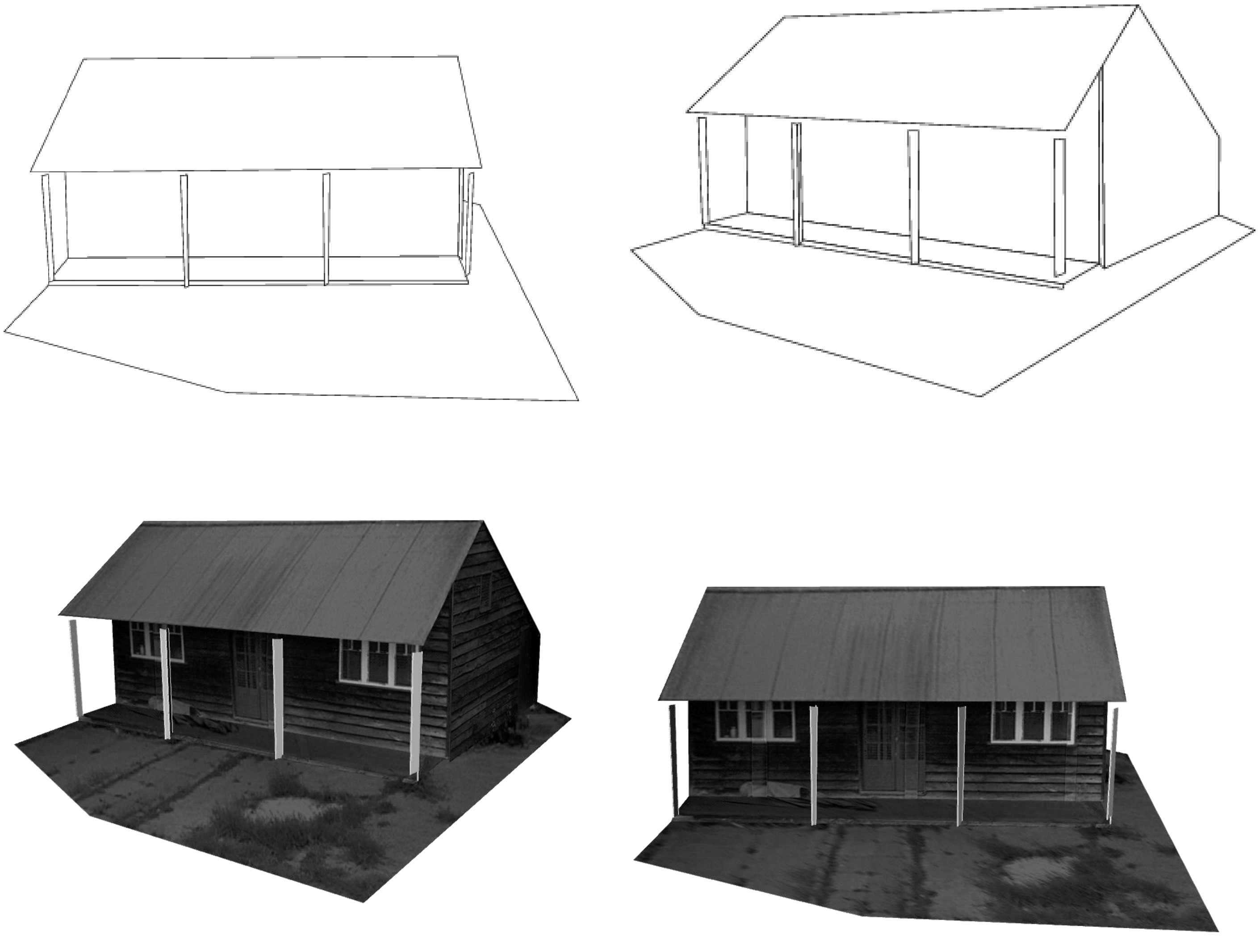


Similarity ambiguity



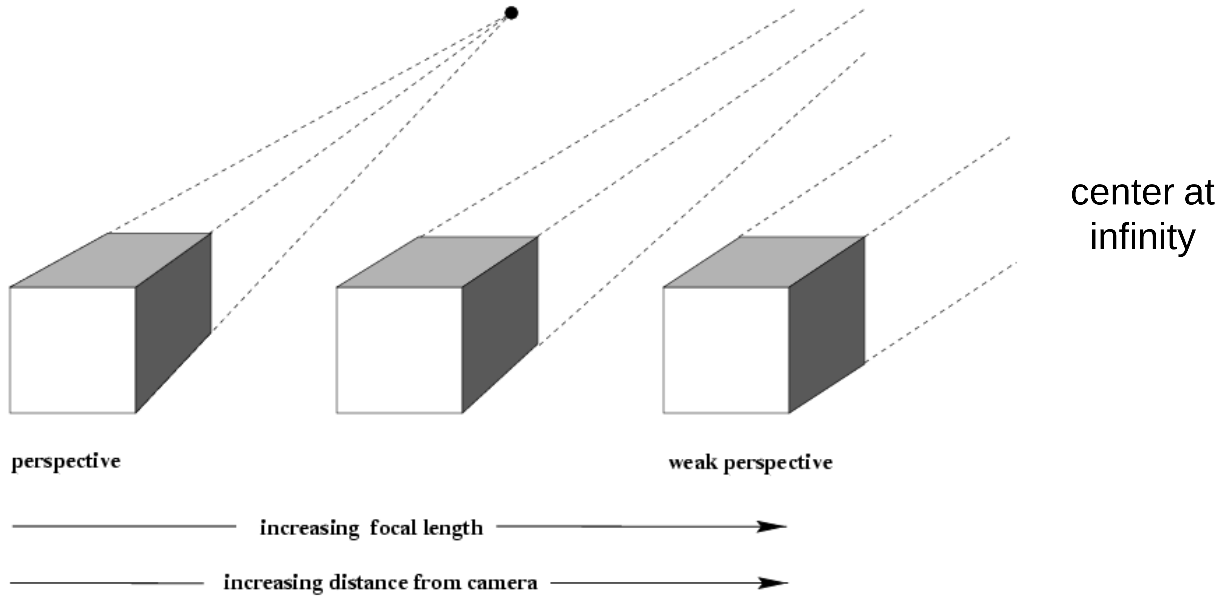
$$\mathbf{x} = \mathbf{P}\mathbf{X} = \left(\mathbf{P}\mathbf{Q}_S^{-1}\right)\left(\mathbf{Q}_S\mathbf{X}\right)$$

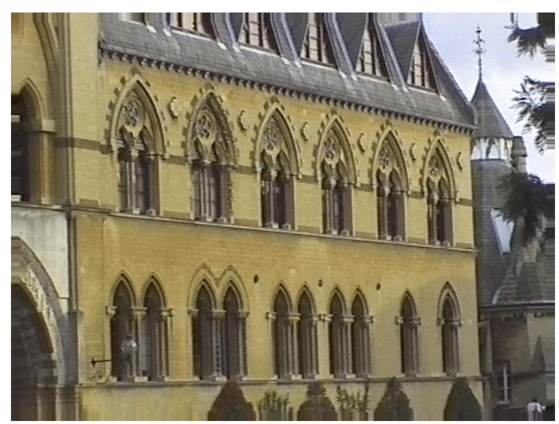
Similarity ambiguity



Structure from motion

- Let's start with *affine cameras* (the math is easier)

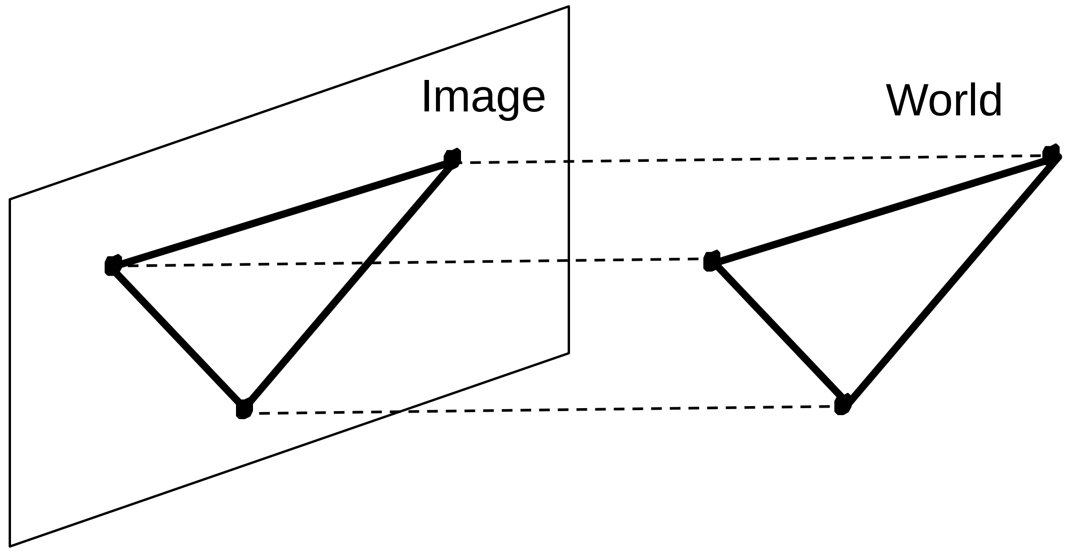




Recall: Orthographic Projection

Special case of perspective projection

- Distance from center of projection to image plane is infinite



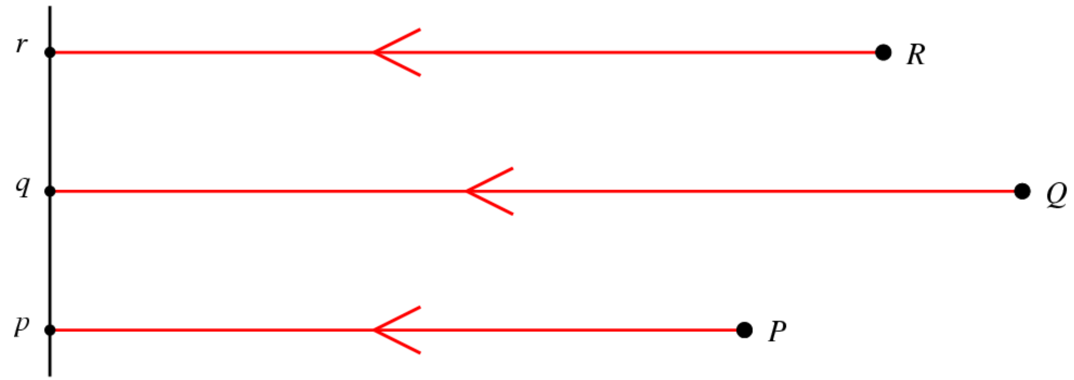
- Projection matrix:

$$\begin{bmatrix} 1 & 0 & 0 & 0 \\ 0 & 1 & 0 & 0 \\ 0 & 0 & 0 & 1 \end{bmatrix} \begin{bmatrix} x \\ y \\ z \\ 1 \end{bmatrix} = \begin{bmatrix} x \\ y \\ 1 \end{bmatrix} \Rightarrow (x, y)$$

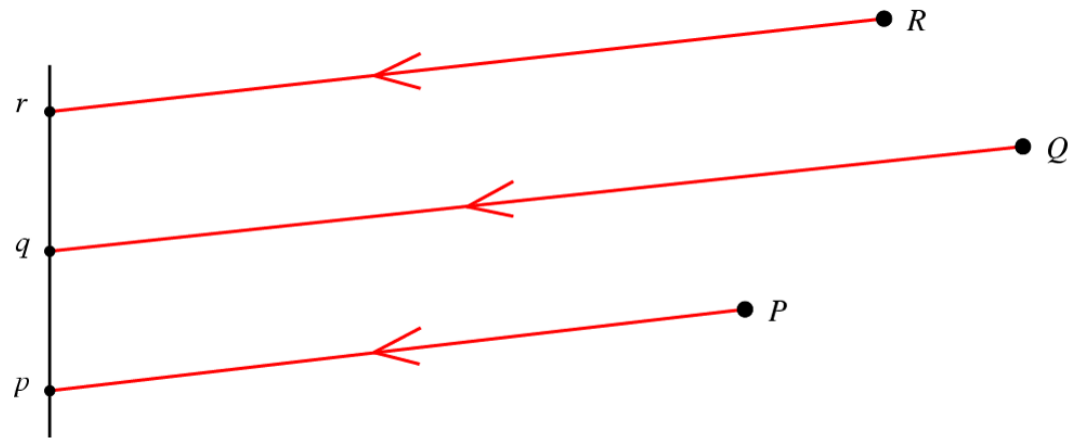
Slide by Steve Seitz

Affine cameras

Orthographic Projection



Parallel Projection

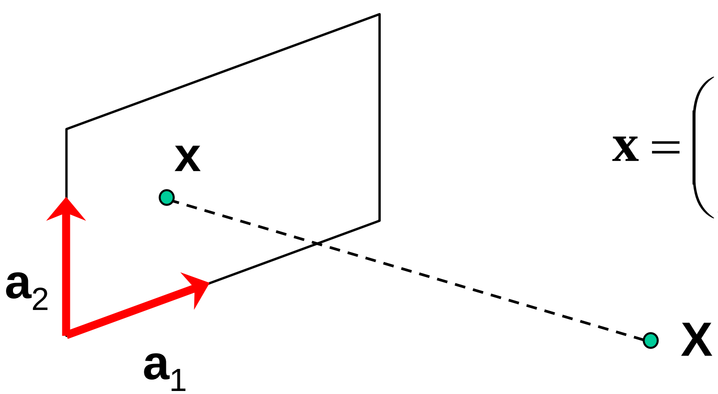


Affine cameras

- A general affine camera combines the effects of an affine transformation of the 3D space, orthographic projection, and an affine transformation of the image:

$$\mathbf{P} = [3\times 3\,\text{affine}]\begin{bmatrix} 1 & 0 & 0 & 0 \\ 0 & 1 & 0 & 0 \\ 0 & 0 & 0 & 1 \end{bmatrix}[4\times 4\,\text{affine}] = \begin{bmatrix} a_{11} & a_{12} & a_{13} & b_1 \\ a_{21} & a_{22} & a_{23} & b_2 \\ 0 & 0 & 0 & 1 \end{bmatrix} = \begin{bmatrix} \mathbf{A} & \mathbf{b} \\ \mathbf{0} & \mathbf{1} \end{bmatrix}$$

- Affine projection is a linear mapping + translation in inhomogeneous coordinates



$$\mathbf{x} = \begin{pmatrix} x \\ y \end{pmatrix} = \begin{bmatrix} a_{11} & a_{12} & a_{13} \\ a_{21} & a_{22} & a_{23} \end{bmatrix}\begin{pmatrix} X \\ Y \\ Z \end{pmatrix} + \begin{pmatrix} b_1 \\ b_2 \end{pmatrix} = \mathbf{AX} + \mathbf{b}$$

Projection of world origin

Affine structure from motion

- Given: *m* images of *n* fixed 3D points:

$$\mathbf{x}_{ij} = \mathbf{A}_i \mathbf{X}_j + \mathbf{b}_i , \quad i = 1, \dots , m, \; j = 1, \dots , n$$

- Problem: use the *mn* correspondences $\mathbf{x}_{ij}$ to estimate *m* projection matrices $\mathbf{A}_i$ and translation vectors $\mathbf{b}_i$, and *n* points $\mathbf{X}_j$
- The reconstruction is defined up to an arbitrary *affine* transformation **Q** (12 degrees of freedom):

$$\begin{bmatrix} \mathbf{A} & \mathbf{b} \\ \mathbf{0} & \mathbf{1} \end{bmatrix} \rightarrow \begin{bmatrix} \mathbf{A} & \mathbf{b} \\ \mathbf{0} & \mathbf{1} \end{bmatrix} \mathbf{Q}^{-1}, \qquad \begin{pmatrix} \mathbf{X} \\ \mathbf{1} \end{pmatrix} \rightarrow \mathbf{Q} \begin{pmatrix} \mathbf{X} \\ \mathbf{1} \end{pmatrix}$$

- We have 2*mn* knowns and 8*m* + 3*n* unknowns (minus 12 dof for affine ambiguity)
- Thus, we must have 2*mn* >= 8*m* + 3*n* – 12
- For two views, we need four point correspondences

Affine structure from motion

- Centering: subtract the centroid of the image points

$$\hat{\mathbf{x}}_{ij} = \mathbf{x}_{ij} - \frac{1}{n}\sum_{k=1}^{n}\mathbf{x}_{ik} = \mathbf{A}_i\mathbf{X}_j + \mathbf{b}_i - \frac{1}{n}\sum_{k=1}^{n}\left(\mathbf{A}_i\mathbf{X}_k + \mathbf{b}_i\right)$$

$$= \mathbf{A}_i\left(\mathbf{X}_j - \frac{1}{n}\sum_{k=1}^{n}\mathbf{X}_k\right) = \mathbf{A}_i\hat{\mathbf{X}}_j$$

- For simplicity, assume that the origin of the world coordinate system is at the centroid of the 3D points
- After centering, each normalized point $\mathbf{x}_{ij}$ is related to the 3D point $\mathbf{X}_i$ by

$$\hat{\mathbf{x}}_{ij} = \mathbf{A}_i\mathbf{X}_j$$

Affine structure from motion

- Let's create a $2m \times n$ data (measurement) matrix:

$$\mathbf{D} = \begin{bmatrix} \hat{\mathbf{x}}_{11} & \hat{\mathbf{x}}_{12} & \cdots & \hat{\mathbf{x}}_{1n} \\ \hat{\mathbf{x}}_{21} & \hat{\mathbf{x}}_{22} & \cdots & \hat{\mathbf{x}}_{2n} \\ & & \ddots & \\ \hat{\mathbf{x}}_{m1} & \hat{\mathbf{x}}_{m2} & \cdots & \hat{\mathbf{x}}_{mn} \end{bmatrix}$$

cameras ($2m$)

points (n)

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

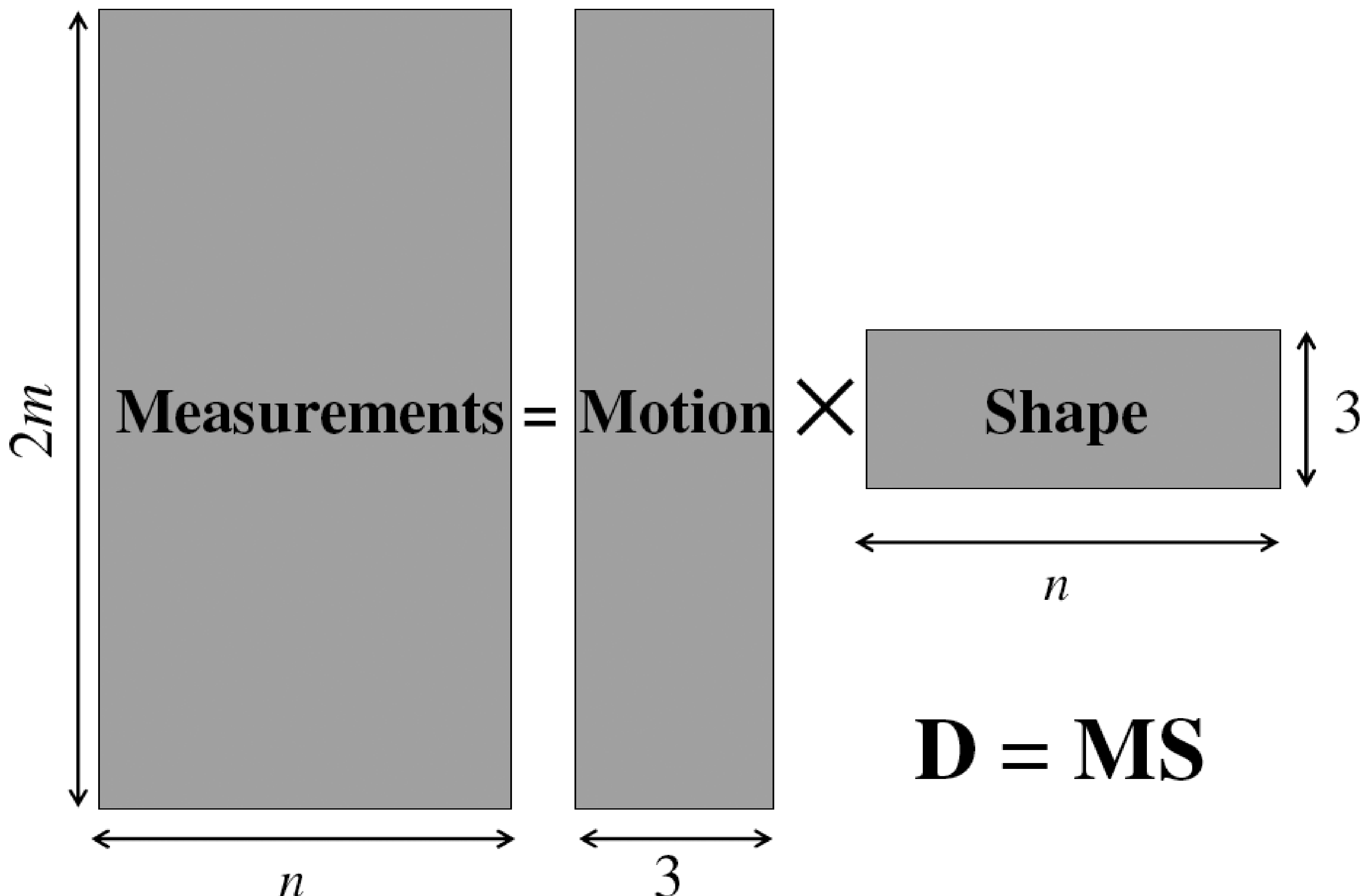
- Let's create a $2m \times n$ data (measurement) matrix:

$$\mathbf{D} = \begin{bmatrix} \hat{\mathbf{x}}_{11} & \hat{\mathbf{x}}_{12} & \cdots & \hat{\mathbf{x}}_{1n} \\ \hat{\mathbf{x}}_{21} & \hat{\mathbf{x}}_{22} & \cdots & \hat{\mathbf{x}}_{2n} \\ & & \ddots & \\ \hat{\mathbf{x}}_{m1} & \hat{\mathbf{x}}_{m2} & \cdots & \hat{\mathbf{x}}_{mn} \end{bmatrix} = \underbrace{\begin{bmatrix} \mathbf{A}_1 \\ \mathbf{A}_2 \\ \vdots \\ \mathbf{A}_m \end{bmatrix}}_{\text{cameras } (2m \times 3)} \underbrace{\begin{bmatrix} \mathbf{X}_1 & \mathbf{X}_2 & \cdots & \mathbf{X}_n \end{bmatrix}}_{\text{points } (3 \times n)}$$

The measurement matrix $\mathbf{D} = \mathbf{MS}$ must have rank 3!

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

Factorizing the measurement matrix

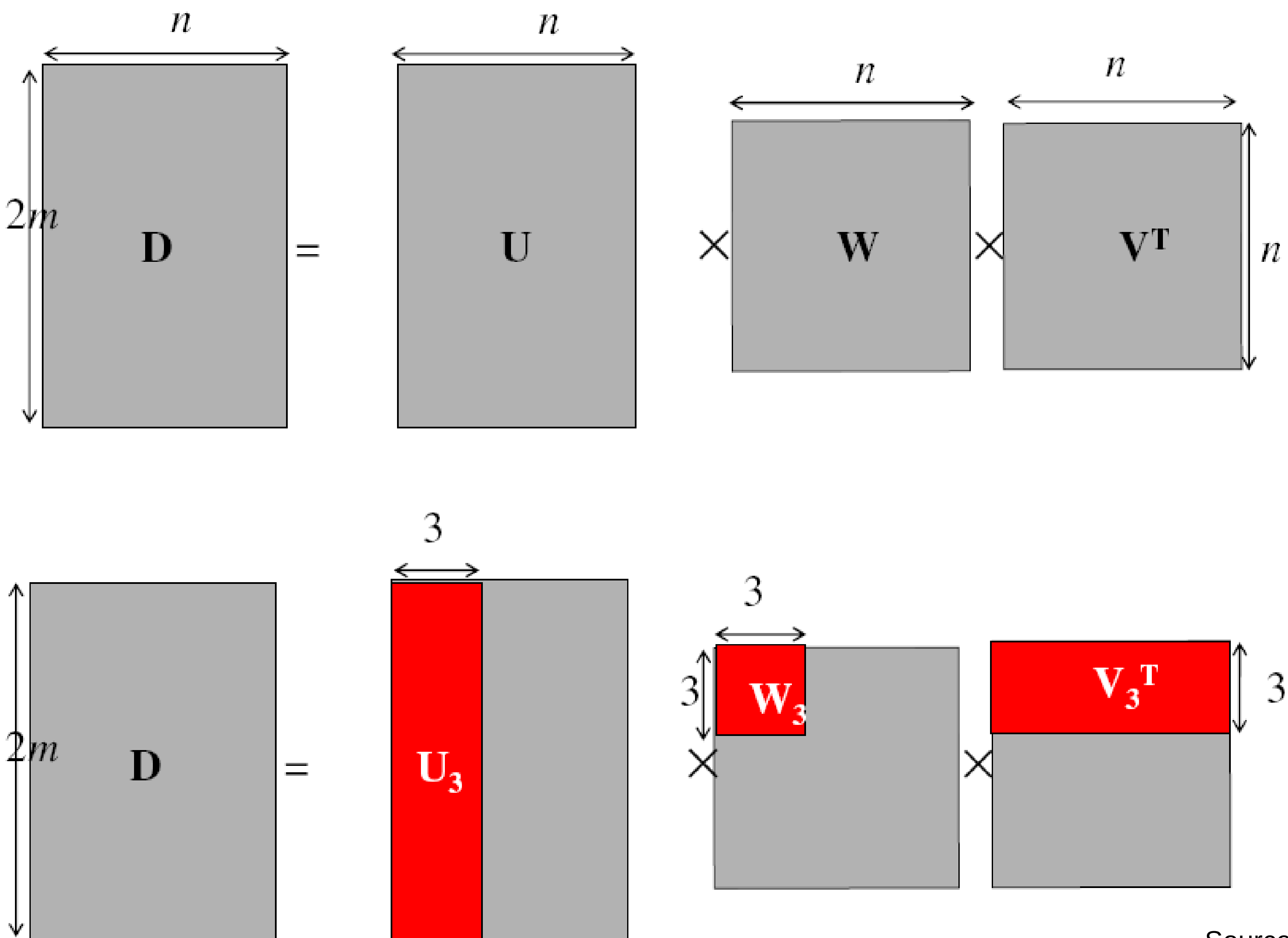


D = MS

Source: M. Hebert

Factorizing the measurement matrix

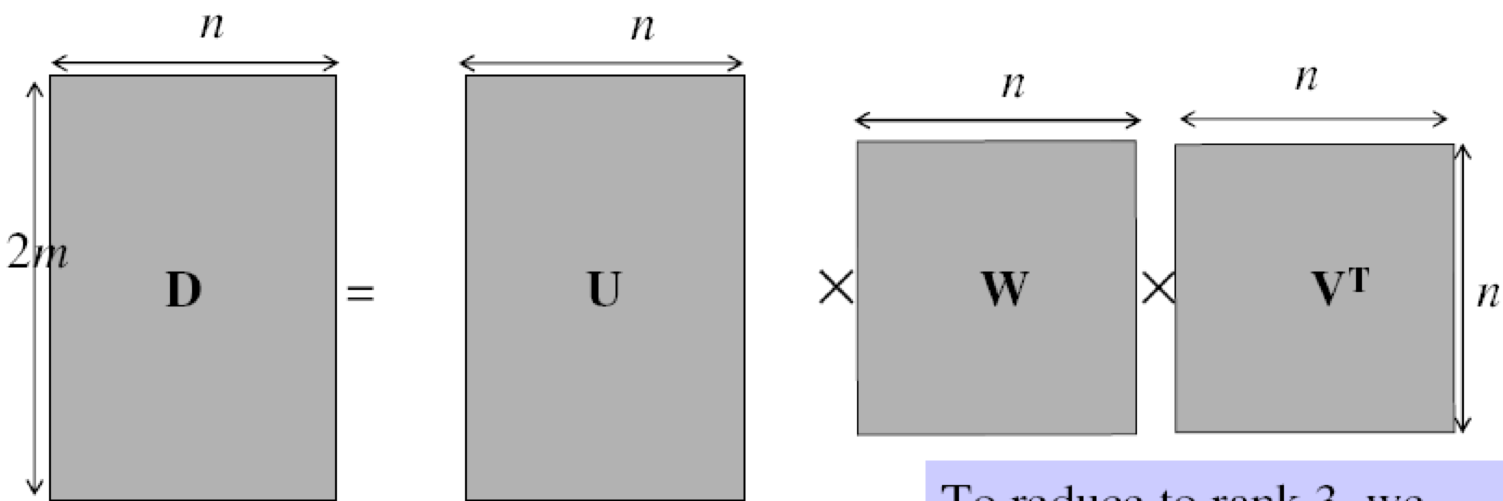
- Singular value decomposition of D:



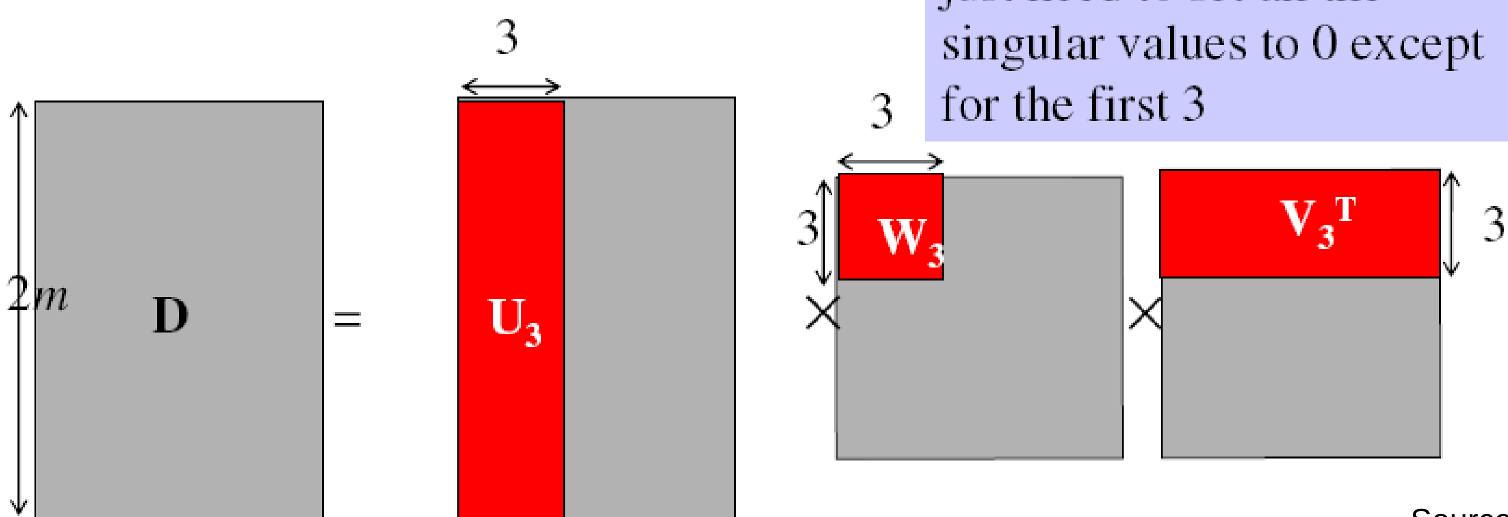
Source: M. Hebert

Factorizing the measurement matrix

- Singular value decomposition of D:



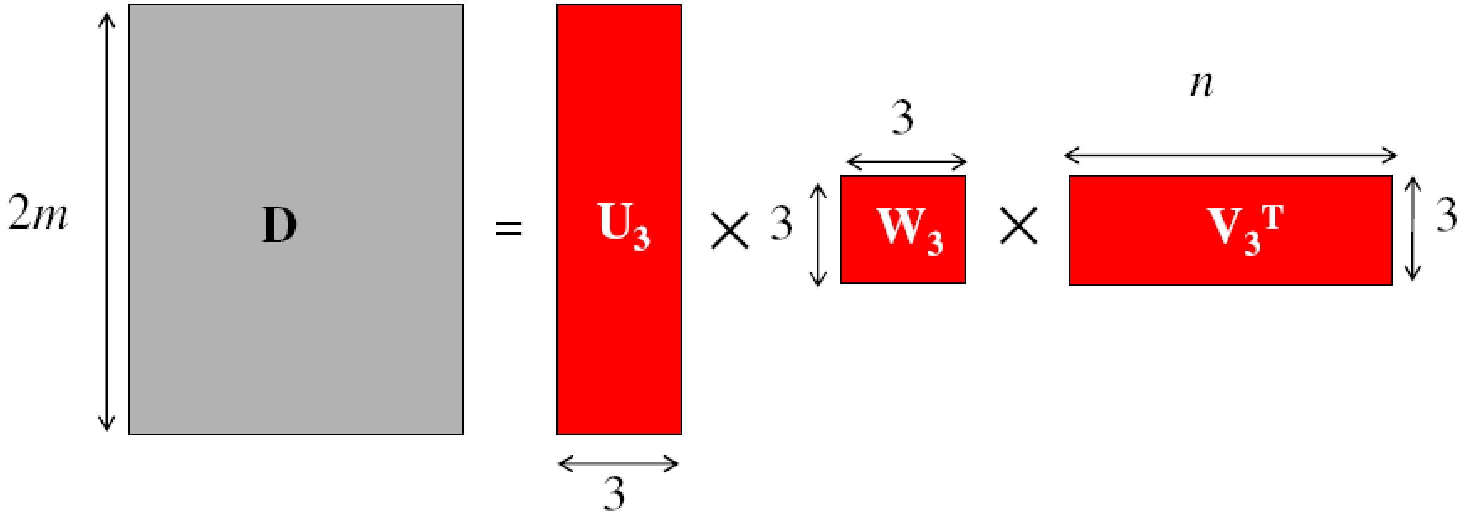
To reduce to rank 3, we just need to set all the singular values to 0 except for the first 3



Source: M. Hebert

Factorizing the measurement matrix

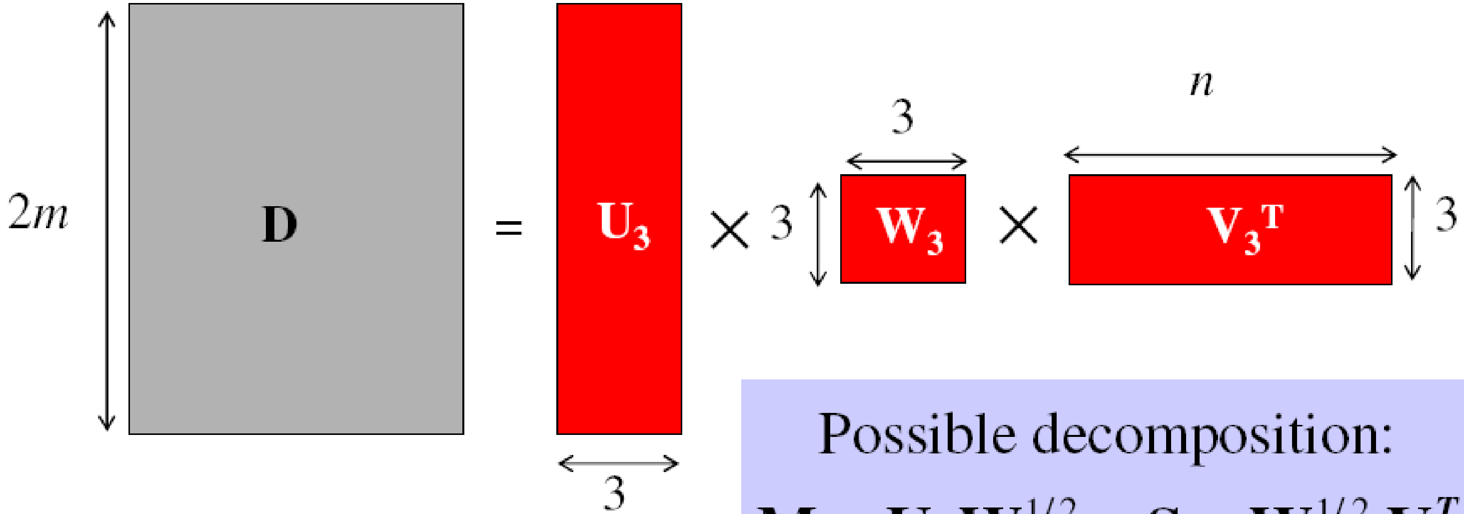
- Obtaining a factorization from SVD:



Source: M. Hebert

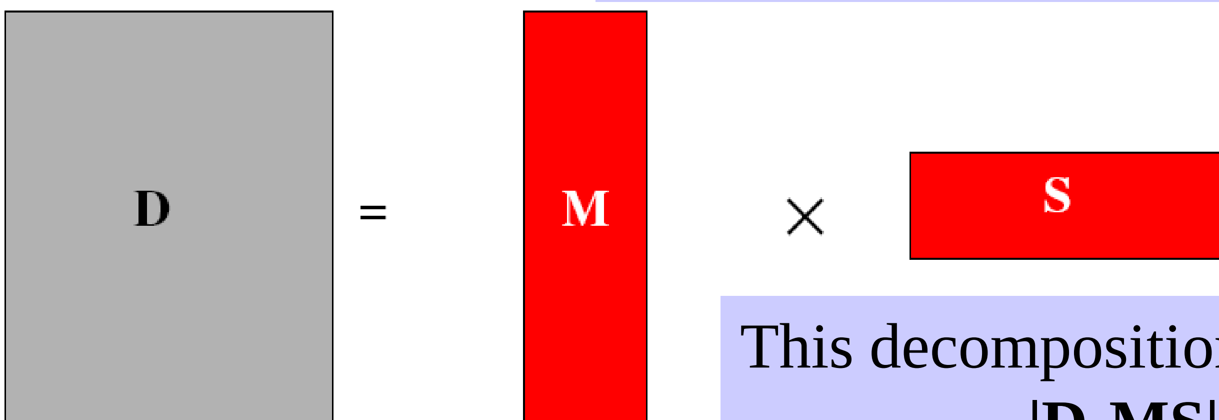
Factorizing the measurement matrix

- Obtaining a factorization from SVD:



Possible decomposition:

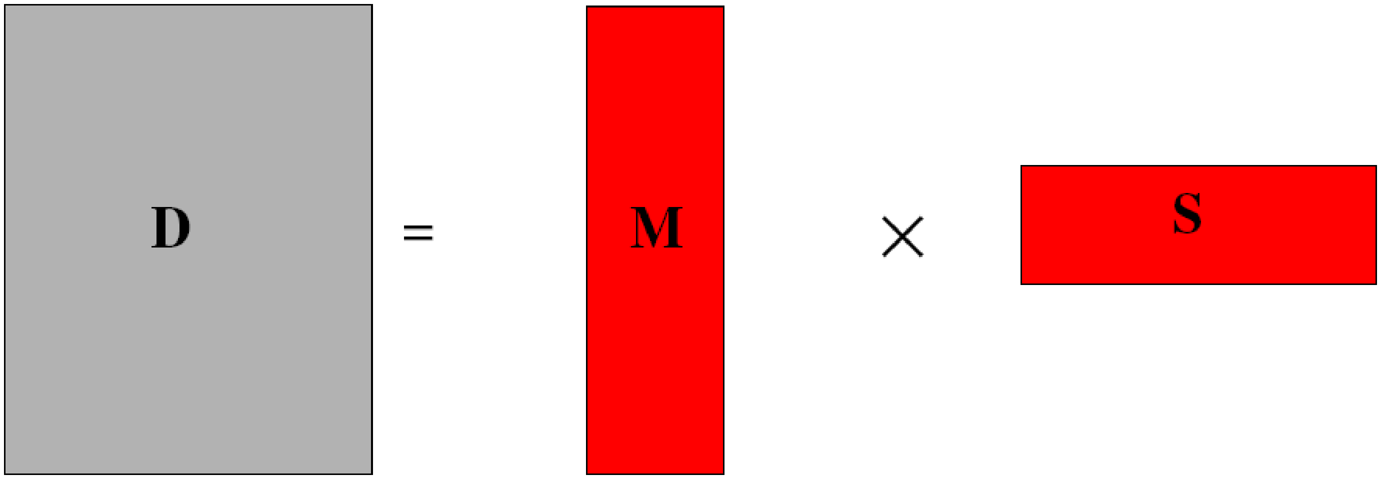
$$\mathbf{M} = \mathbf{U}_3 \mathbf{W}_3^{1/2} \qquad \mathbf{S} = \mathbf{W}_3^{1/2}\, \mathbf{V}_3^T$$



This decomposition minimizes
$|\mathbf{D}\text{-}\mathbf{MS}|^2$

Source: M. Hebert

Affine ambiguity

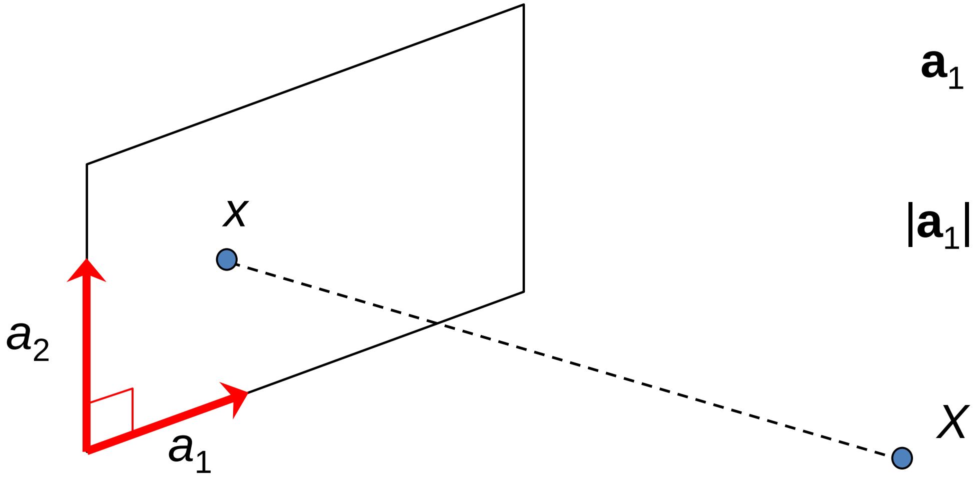


- The decomposition is not unique. We get the same **D** by using any 3×3 matrix **C** and applying the transformations **M** → **MC, S** →$\mathbf{C}^{-1}\mathbf{S}$
- That is because we have only an affine transformation and we have not enforced any Euclidean constraints (like forcing the image axes to be perpendicular, for example)

Source: M. Hebert

Eliminating the affine ambiguity

- Orthographic: image axes are perpendicular and of unit length



$$\mathbf{a}_1 \cdot \mathbf{a}_2 = 0$$

$$|\mathbf{a}_1|^2 = |\mathbf{a}_2|^2 = 1$$

Source: M. Hebert

Solve for orthographic constraints

Three equations for each image i

$$\tilde{\mathbf{a}}_{i1}^T \mathbf{C}\mathbf{C}^T \tilde{\mathbf{a}}_{i1}^T = 1$$
$$\tilde{\mathbf{a}}_{i2}^T \mathbf{C}\mathbf{C}^T \tilde{\mathbf{a}}_{i2}^T = 1 \quad \text{where} \quad \tilde{\mathbf{A}}_i = \begin{bmatrix} \tilde{\mathbf{a}}_{i1}^T \\ \tilde{\mathbf{a}}_{i2}^T \end{bmatrix}$$
$$\tilde{\mathbf{a}}_{i1}^T \mathbf{C}\mathbf{C}^T \tilde{\mathbf{a}}_{i2}^T = 0$$

- Solve for $\mathbf{L} = \mathbf{C}\mathbf{C}^T$
- Recover **C** from **L** by Cholesky decomposition: $\mathbf{L} = \mathbf{C}\mathbf{C}^T$
- Update **A** and **X**: $\mathbf{A} = \tilde{\mathbf{A}}\mathbf{C}$, $\mathbf{X} = \mathbf{C}^{-1}\tilde{\mathbf{X}}$

Slide: D. Hoiem

Algorithm summary

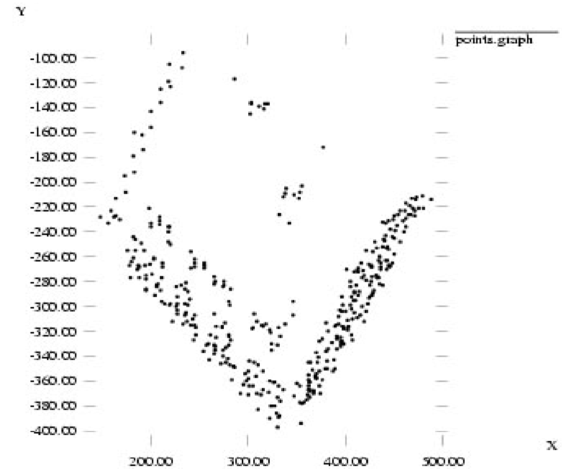
- Given: *m* images and *n* features $\mathbf{x}_{ij}$
- For each image *i, c*enter the feature coordinates
- Construct a $2m \times n$ measurement matrix **D**:
  - Column *j* contains the projection of point *j* in all views
  - Row *i* contains one coordinate of the projections of all the *n* points in image *i*
- Factorize **D**:
  - Compute SVD: $\mathbf{D} = \mathbf{U}\ \mathbf{W}\ \mathbf{V}^{\mathbf{T}}$
  - Create $\mathbf{U}_3$ by taking the first 3 columns of **U**
  - Create $\mathbf{V}_3$ by taking the first 3 columns of **V**
  - Create $\mathbf{W}_3$ by taking the upper left 3×3 block of **W**
- Create the motion and shape matrices:
  - $\mathbf{M} = \mathbf{U}_3\mathbf{W}_3^{1/2}$ and $\mathbf{S} = \mathbf{W}_3^{1/2}\ \mathbf{V}_3^T$ (**or** $\mathbf{M} = \mathbf{U}_3$ and $\mathbf{S} = \mathbf{W}_3\mathbf{V}_3^T$)
- Eliminate affine ambiguity

Source: M. Hebert

Reconstruction results

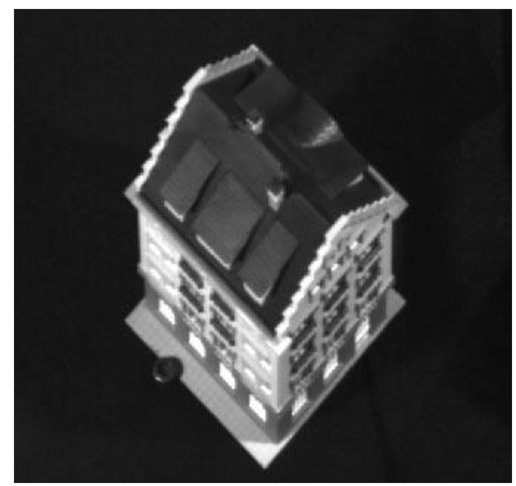
1

60



120

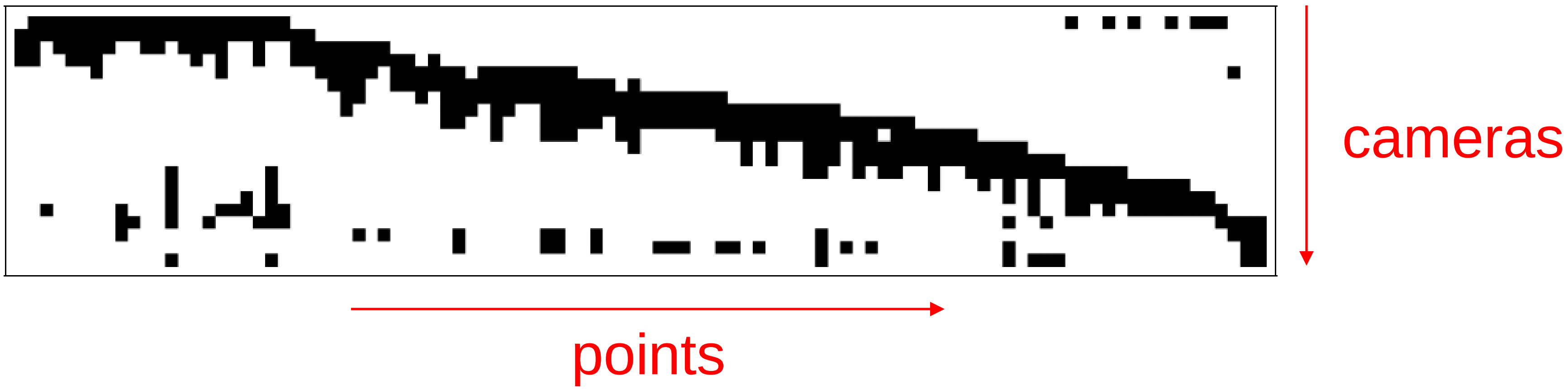
150



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

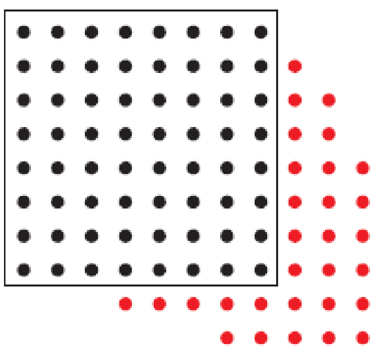
Dealing with missing data

- So far, we have assumed that all points are visible in all views
- In reality, the measurement matrix typically looks something like this:

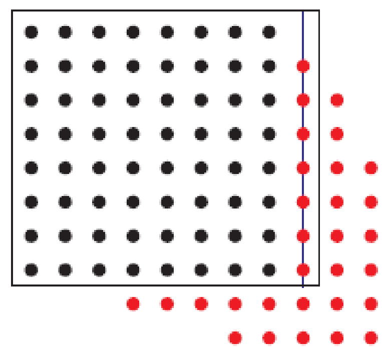


Dealing with missing data

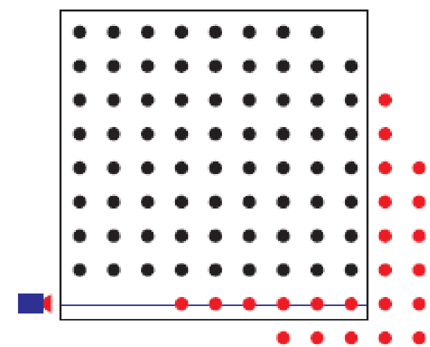
- Possible solution: decompose matrix into dense sub-blocks, factorize each sub-block, and fuse the results
  - Finding dense maximal sub-blocks of the matrix is NP-complete (equivalent to finding maximal cliques in a graph)
- Incremental bilinear refinement



(1) Perform factorization on a dense sub-block



(2) Solve for a new 3D point visible by at least two known cameras (linear least squares)



(3) Solve for a new camera that sees at least three known 3D points (linear least squares)

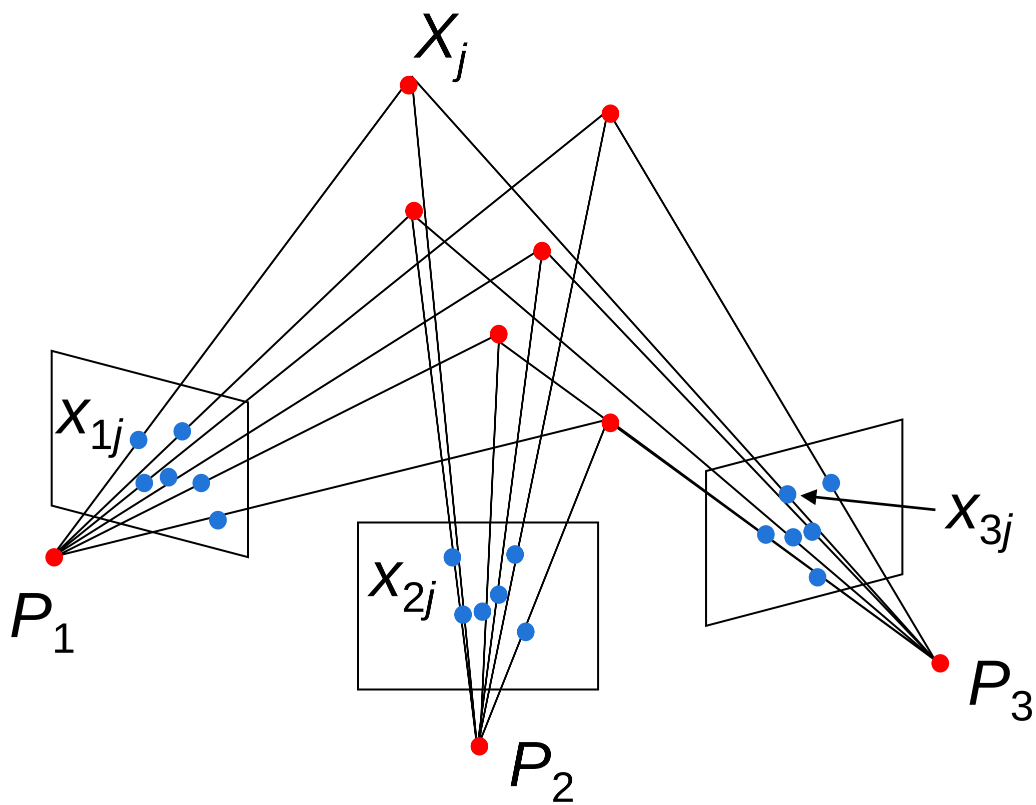
F. Rothganger, S. Lazebnik, C. Schmid, and J. Ponce. [Segmenting, Modeling, and Matching Video Clips Containing Multiple Moving Objects.](#) PAMI 2007.

Projective structure from motion

- Given: *m* images of *n* fixed 3D points

$$z_{ij}\,\mathbf{x}_{ij} = \mathbf{P}_i\,\mathbf{X}_j\,, \quad i = 1, \ldots\,, m, \quad j = 1, \ldots\,, n$$

- Problem: estimate *m* projection matrices $\mathbf{P}_i$ and *n* 3D points $\mathbf{X}_j$ from the *mn* correspondences $\mathbf{x}_{ij}$



Projective structure from motion

- Given: *m* images of *n* fixed 3D points

$$z_{ij}\,\mathbf{x}_{ij} = \mathbf{P}_i\,\mathbf{X}_j\,, \quad i = 1, \ldots , m, \quad j = 1, \ldots , n$$

- Problem: estimate *m* projection matrices $\mathbf{P}_i$ and *n* 3D points $\mathbf{X}_j$ from the *mn* correspondences $\mathbf{x}_{ij}$
- With no calibration info, cameras and points can only be recovered up to a 4x4 projective transformation **Q**:

$$\mathbf{X} \rightarrow \mathbf{QX},\ \mathbf{P} \rightarrow \mathbf{PQ}^{-1}$$

- We can solve for structure and motion when

$$2mn >= 11m + 3n - 15$$

- For two cameras, at least 7 points are needed

Projective SFM: Two-camera case

- Compute fundamental matrix $\mathbf{F}$ between the two views
- First camera matrix: $[\mathbf{I}|\mathbf{0}]$
- Second camera matrix: $[\mathbf{A}|\mathbf{b}]$
- Then $\mathbf{b}$ is the epipole ($\mathbf{F}^{T}\mathbf{b} = 0$), $\mathbf{A} = -[\mathbf{b}_{\times}]\mathbf{F}$

F&P sec. 13.3.1

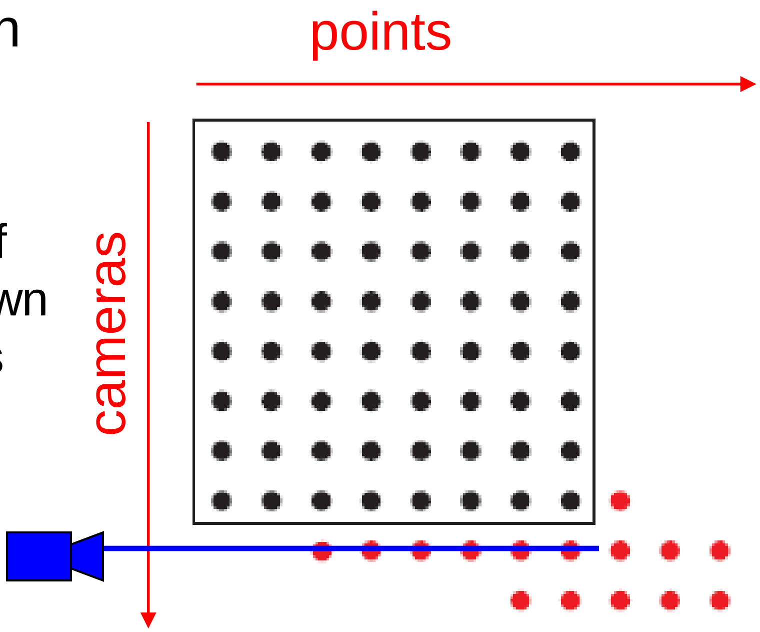
Sequential structure from motion

•Initialize motion from two images using fundamental matrix

•Initialize structure by triangulation

•For each additional view:

- Determine projection matrix of new camera using all the known 3D points that are visible in its image – *calibration*



Sequential structure from motion

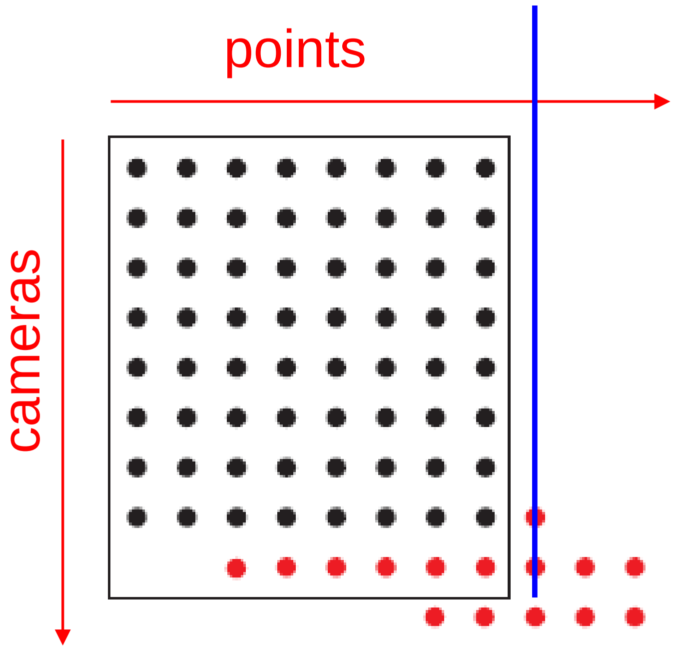
•Initialize motion from two images using fundamental matrix

•Initialize structure by triangulation

•For each additional view:

- Determine projection matrix of new camera using all the known 3D points that are visible in its image – *calibration*

- Refine and extend structure: compute new 3D points, re-optimize existing points that are also seen by this camera – *triangulation*



Sequential structure from motion

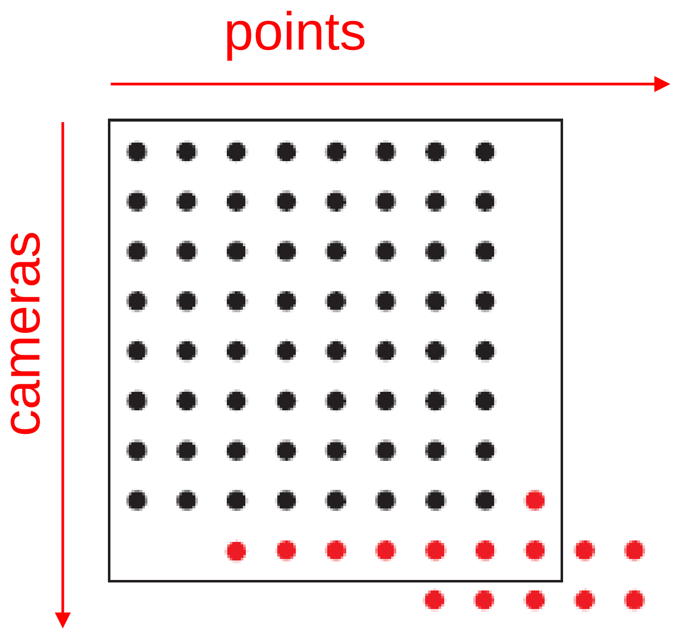
- Initialize motion from two images using fundamental matrix

- Initialize structure by triangulation

- For each additional view:
    - Determine projection matrix of new camera using all the known 3D points that are visible in its image – *calibration*

    - Refine and extend structure: compute new 3D points, re-optimize existing points that are also seen by this camera – *triangulation*

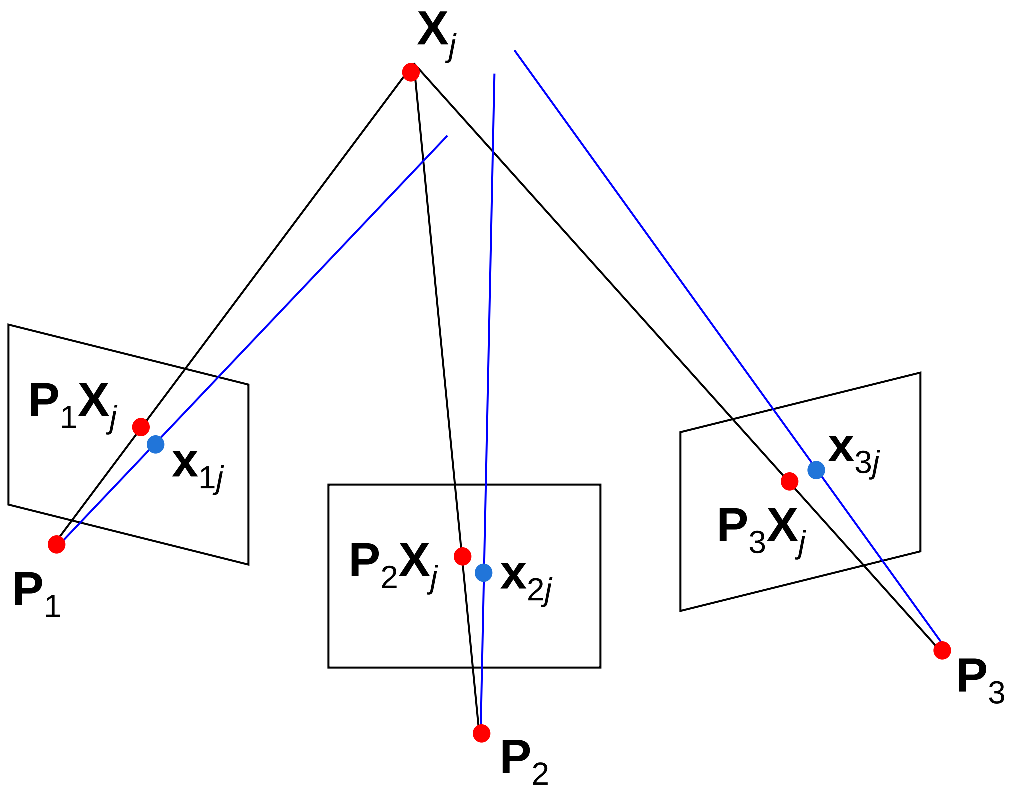
- Refine structure and motion: bundle adjustment



Bundle adjustment

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

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



Self-calibration

- Self-calibration (auto-calibration) is the process of determining intrinsic camera parameters directly from uncalibrated images
- For example, when the images are acquired by a single moving camera, we can use the constraint that the intrinsic parameter matrix remains fixed for all the images
  - Compute initial projective reconstruction and find 3D projective transformation matrix $\mathbf{Q}$ such that all camera matrices are in the form $\mathbf{P}_i = \mathbf{K}\,[\mathbf{R}_i \mid \mathbf{t}_i]$
- Can use constraints on the form of the calibration matrix: zero skew

Review: Structure from motion

- Ambiguity
- Affine structure from motion
  - Factorization
- Dealing with missing data
  - Incremental structure from motion
- Projective structure from motion
  - Bundle adjustment
  - Self-calibration

Summary: 3D geometric vision

- Single-view geometry
  - The pinhole camera model
    - Variation: orthographic projection
  - The perspective projection matrix
  - Intrinsic parameters
  - Extrinsic parameters
  - Calibration
- Multiple-view geometry
  - Triangulation
  - The epipolar constraint
    - Essential matrix and fundamental matrix
  - Stereo
    - Binocular, multi-view
  - Structure from motion
    - Reconstruction ambiguity
    - Affine SFM
    - Projective SFM

Overview

Multi-view stereo

Structure from Motion (SfM)

Large scale Structure from Motion

Large-scale Structure from motion

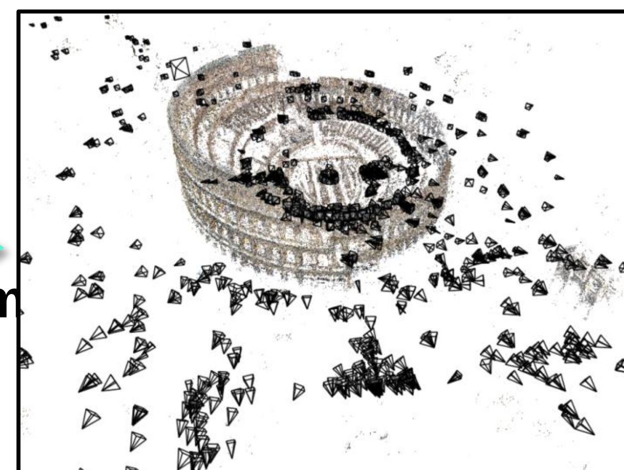
Given many images from photo collections how can we

a) figure out where they were all taken from?

b) build a 3D model of the scene?

T

from



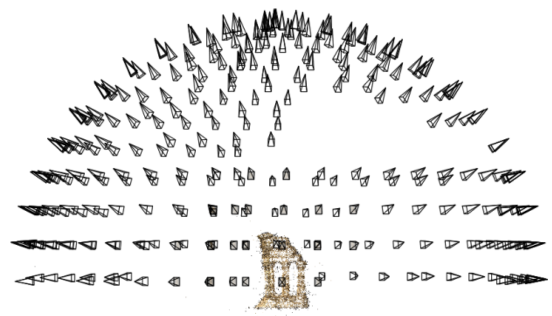
Slides from N. Snavely

Large-scale structure from motion

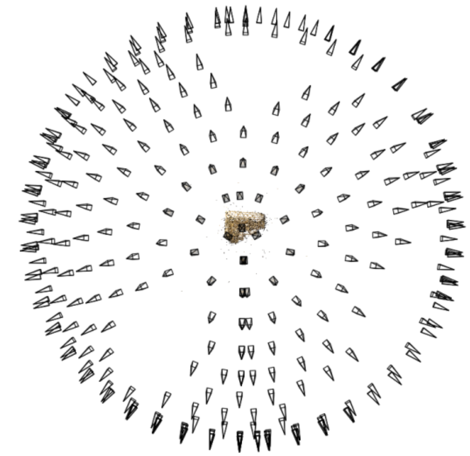
Dubrovnik, Croatia. 4,619 images (out of an initial 57,845).
Total reconstruction time: 23 hours
Number of cores: 352

Slide: N. Snavely

Structure from motion



Reconstruction (side)



(top)

- Input: images with points in correspondence
  $p_{i,j} = (u_{i,j}, v_{i,j})$

- Output
  - structure: 3D location $\mathbf{x}_i$ for each point p_i
  - motion: camera parameters $\mathbf{R}_j$, $\mathbf{t}_j$ possibly $\mathbf{K}_j$

- Objective function: minimize *reprojection error*

Photo Tourism

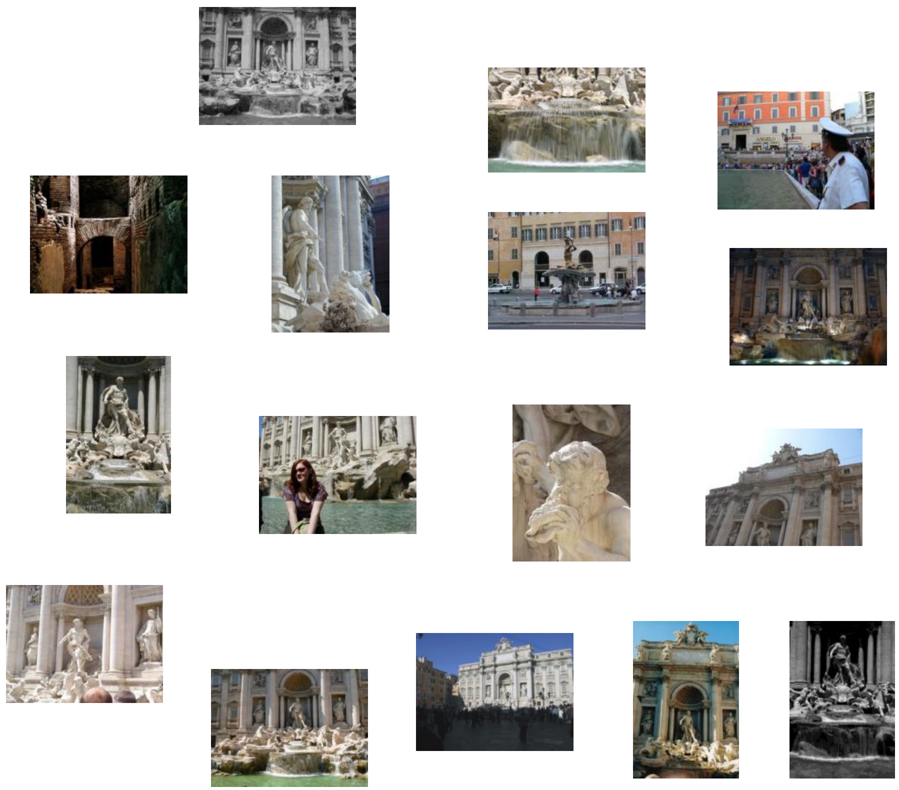
Slide: N. Snavely

First step: how to get correspondence?

Feature detection and matching

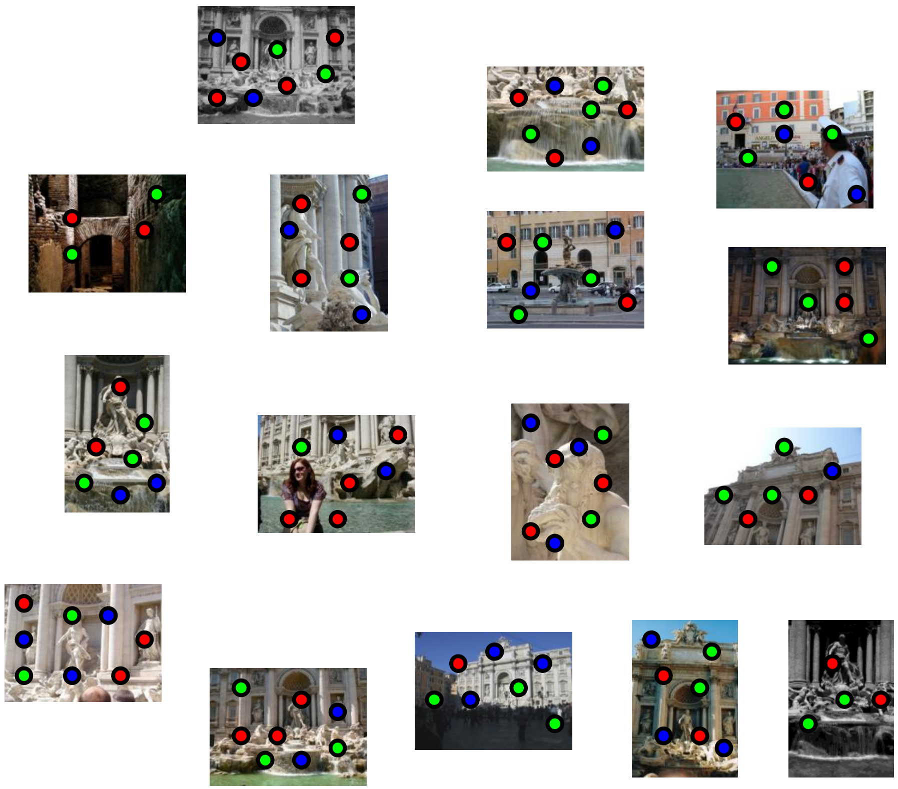
Feature detection

Detect features using SIFT [Lowe, IJCV 2004]



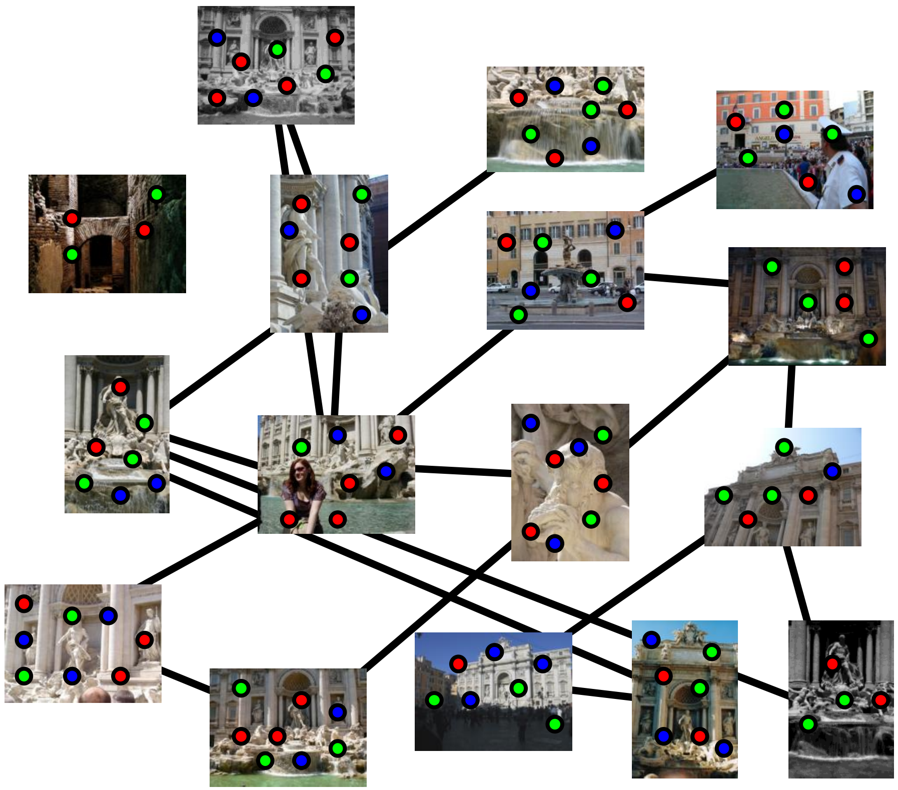
Feature detection

Detect features using SIFT [Lowe, IJCV 2004]



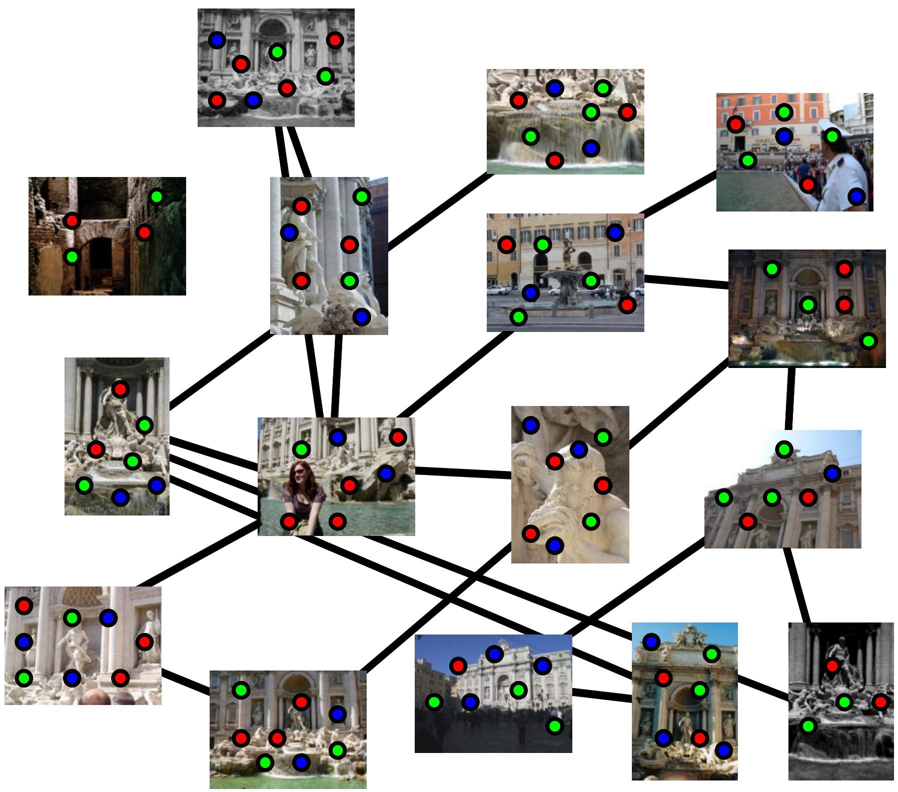
Feature matching

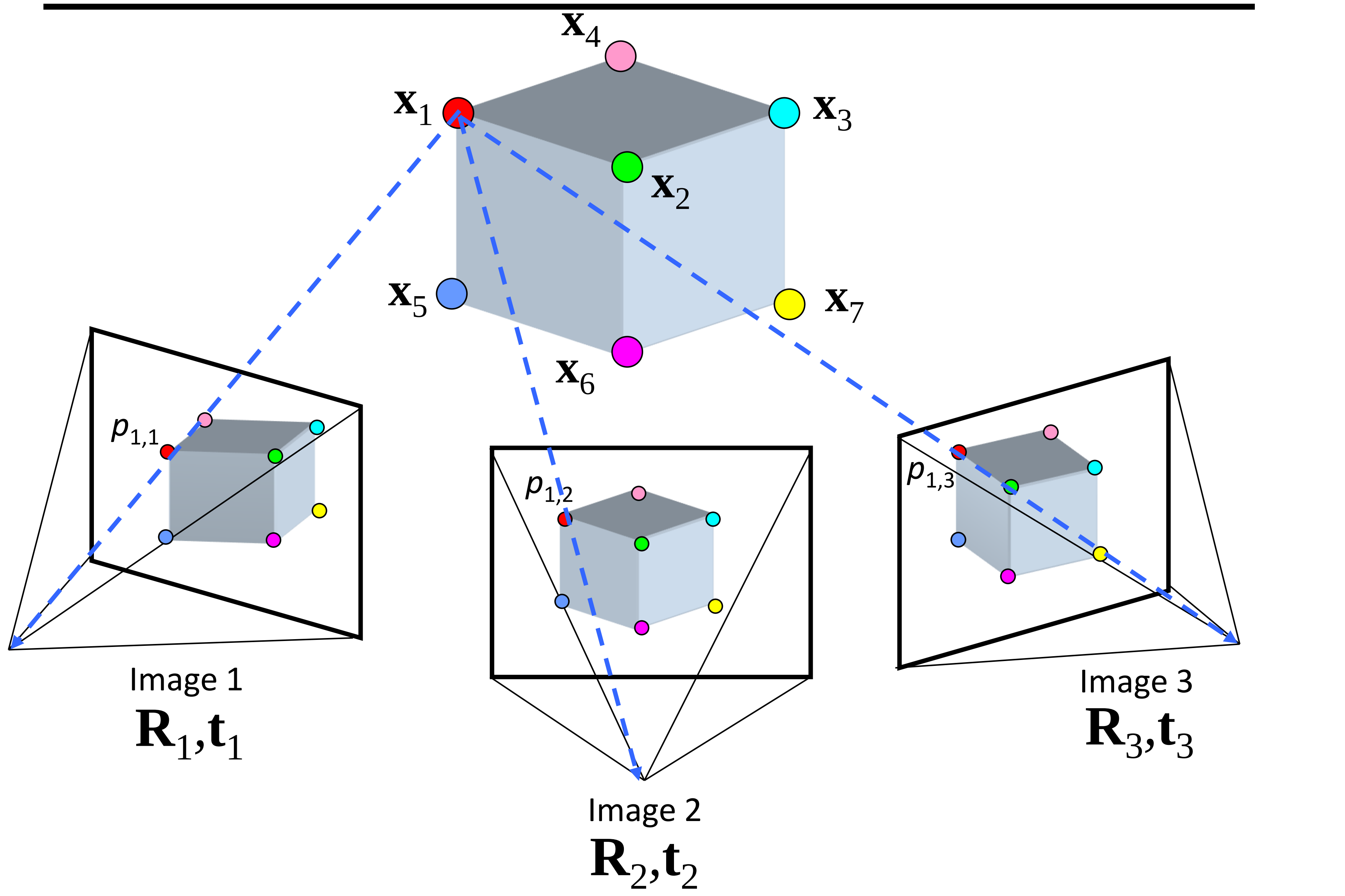
Match features between each pair of images



Feature matching

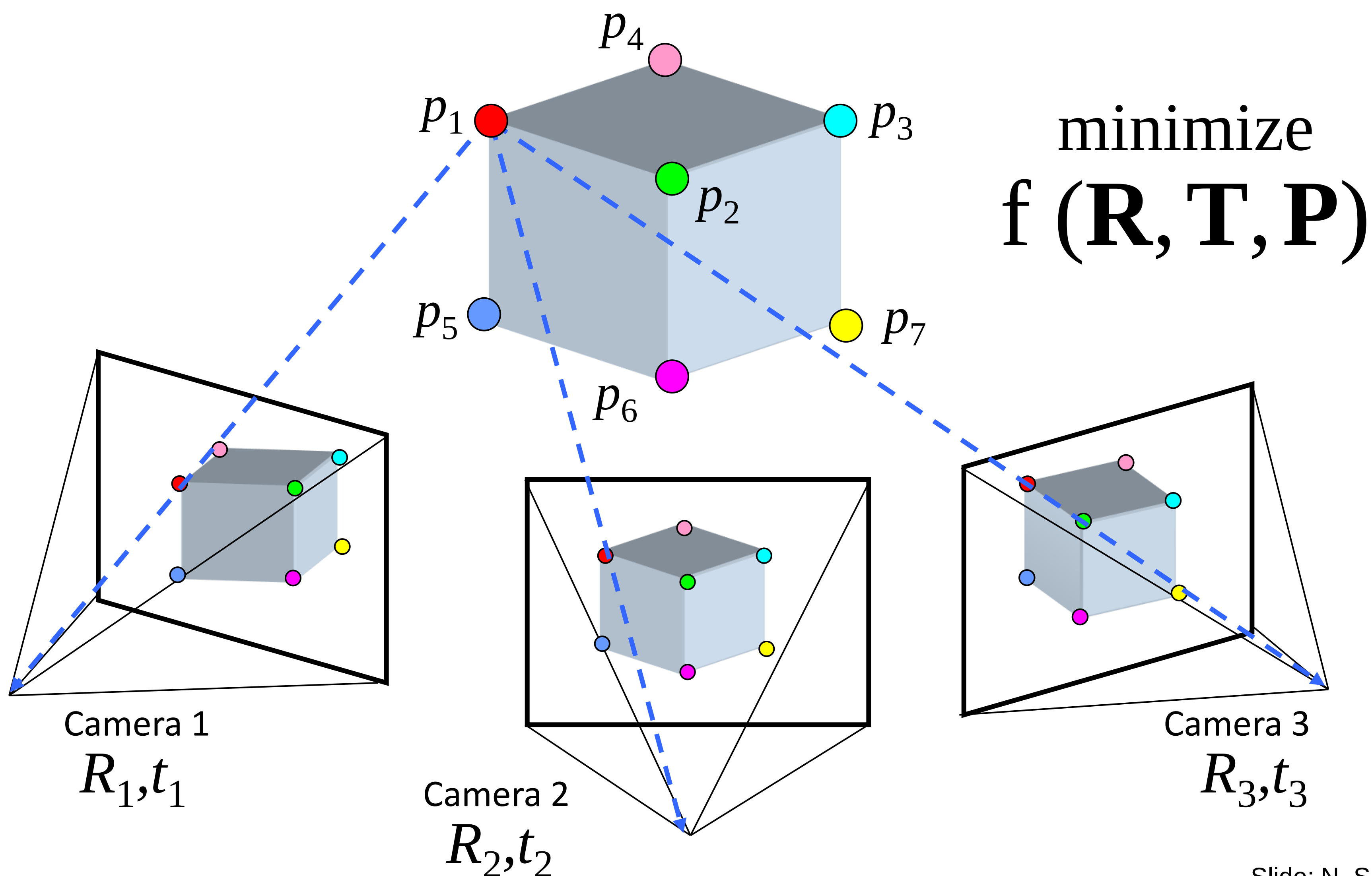
Refine matching using RANSAC to estimate fundamental matrix between each pair





Slide: N. Snavely

Structure from motion



Slide: N. Snavely

Problem size

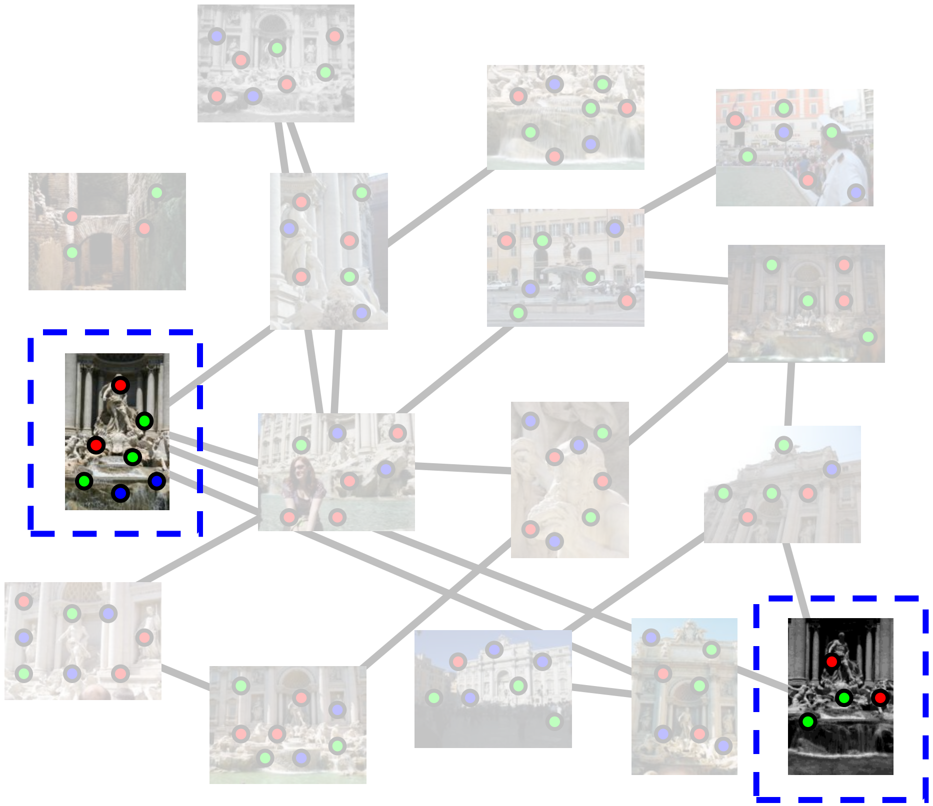
Trevi Fountain collection

466 input photos

+ > 100,000 3D points

= very large optimization problem

Incremental structure from motion



Incremental structure from motion

Slide: N. Snavely

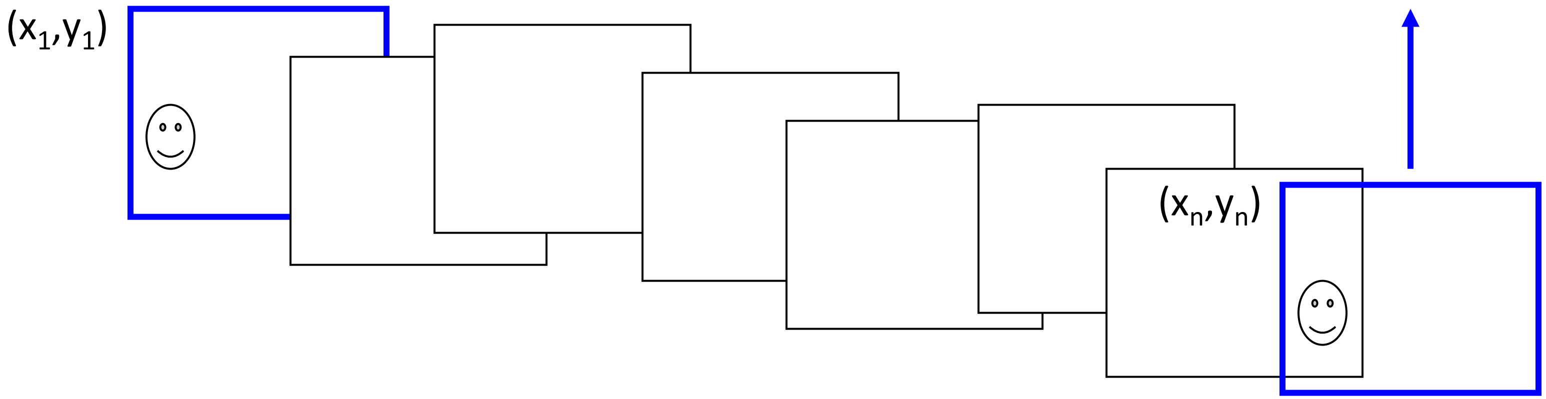
Incremental structure from motion

Slide: N. Snavely

Photo Explorer

Slide: N. Snavely

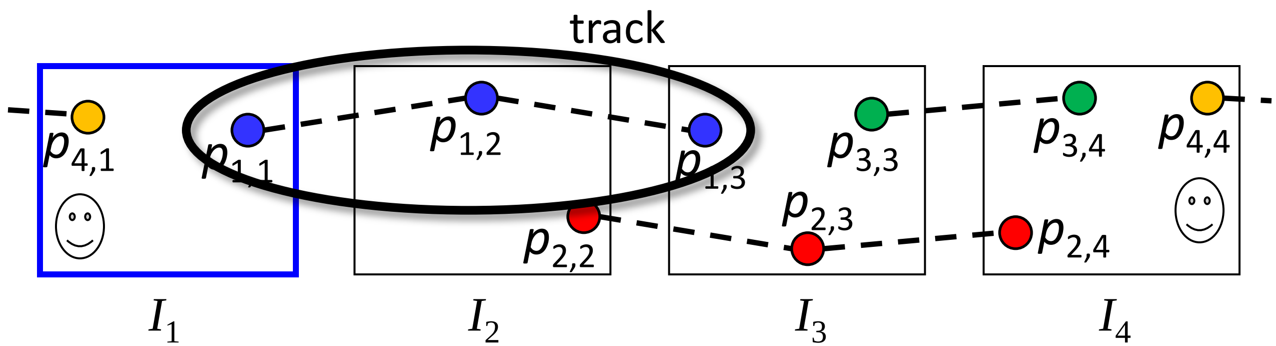
Related topic: Drift



– add another copy of first image at the end
– this gives a constraint: $y_n = y_1$
– there are a bunch of ways to solve this problem
  - add displacement of $(y_1 – y_n)/(n - 1)$ to each image after the first
  - compute a global warp: $y' = y + ax$
  - run a big optimization problem, incorporating this constraint
    – best solution, but more complicated
    – known as "bundle adjustment"

Slide: N. Snavely

Global optimization

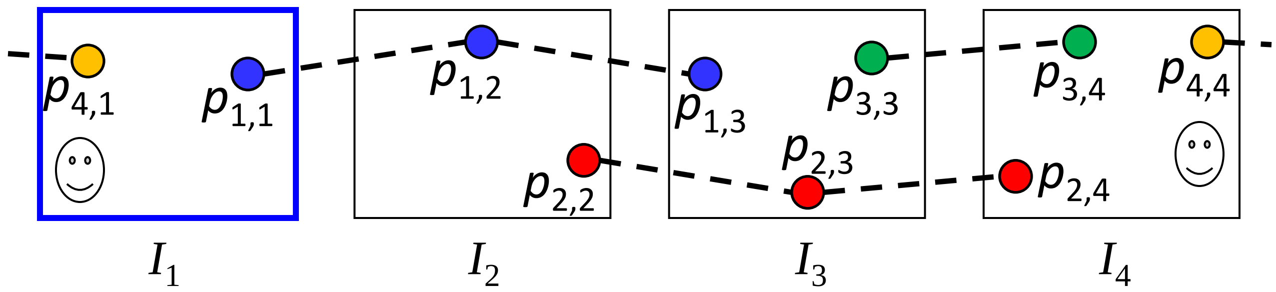


Minimize a global energy function:

- What are the variables?
  - The translation $t_j = (x_j, y_j)$ for each image I_j
- What is the objective function?
  - We have a set of matched features $p_{i,j} = (u_{i,j}, v_{i,j})$
    - » We'll call these *tracks*
  - For each point match $(p_{i,j}, p_{i,j+1})$: $p_{i,j+1} - p_{i,j} = t_{j+1} - t_j$

Slide: N. Snavely

Global optimization



$$p_{1,2} - p_{1,1} = t_2 - t_1$$
$$p_{1,3} - p_{1,2} = t_3 - t_2$$
$$p_{2,3} - p_{2,2} = t_3 - t_2$$
$$\dots$$
$$v_{4,1} - v_{4,4} = y_1 - y_4$$

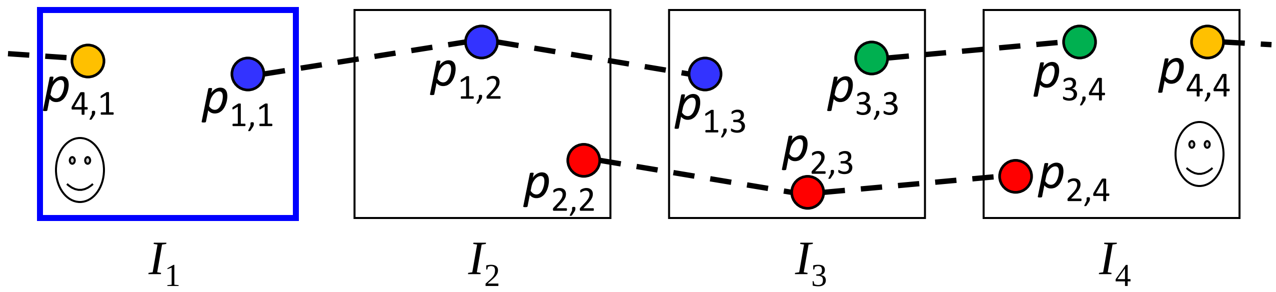
w_{ij} = 1 if track i is visible in images j and j+1
0 otherwise

minimize

$$\sum_{i=1}^{m}\sum_{j=1}^{n-1} w_{ij} \cdot \left\|(p_{i,j+1} - p_{i,j}) - (t_{j+1} - t_j)\right\|^2$$
$$+ \sum_{i=1}^{m} w_{in} \cdot \left\|(v_{i,1} - v_{i,n}) - (y_1 - y_n)\right\|^2$$

Slide: N. Snavely

Global optimization



$$\underbrace{\begin{bmatrix} -1 & 0 & 1 & 0 & 0 & 0 & 0 & 0 \\ 0 & -1 & 0 & 1 & 0 & 0 & 0 & 0 \\ 0 & 0 & -1 & 0 & 1 & 0 & 0 & 0 \\ 0 & 0 & 0 & -1 & 0 & 1 & 0 & 0 \\ & & & \cdots & & & & \\ 0 & 1 & 0 & 0 & 0 & 0 & 0 & -1 \end{bmatrix}}_{\substack{\mathbf{A} \\ 2m \times 2n}} \underbrace{\begin{bmatrix} x_1 \\ y_1 \\ x_2 \\ y_2 \\ x_3 \\ y_3 \\ x_4 \\ y_4 \end{bmatrix}}_{\substack{\mathbf{x} \\ 2n \times 1}} = \underbrace{\begin{bmatrix} u_{1,2} - u_{1,1} \\ v_{1,2} - v_{1,1} \\ \vdots \\ v_{4,1} - v_{4,4} \end{bmatrix}}_{\substack{\mathbf{b} \\ 2m \times 1}}$$

Slide: N. Snavely

Global optimization

$$\begin{bmatrix} -1 & 0 & 1 & 0 & 0 & 0 & 0 & 0 \\ 0 & -1 & 0 & 1 & 0 & 0 & 0 & 0 \\ 0 & 0 & -1 & 0 & 1 & 0 & 0 & 0 \\ 0 & 0 & 0 & -1 & 0 & 1 & 0 & 0 \\ & & & \cdots & & & & \\ 0 & 1 & 0 & 0 & 0 & 0 & 0 & -1 \end{bmatrix} \begin{bmatrix} x_1 \\ y_1 \\ x_2 \\ y_2 \\ x_3 \\ y_3 \\ x_4 \\ y_4 \end{bmatrix} = \begin{bmatrix} u_{1,2} - u_{1,1} \\ v_{1,2} - v_{1,1} \\ \vdots \\ v_{4,1} - v_{4,4} \end{bmatrix}$$

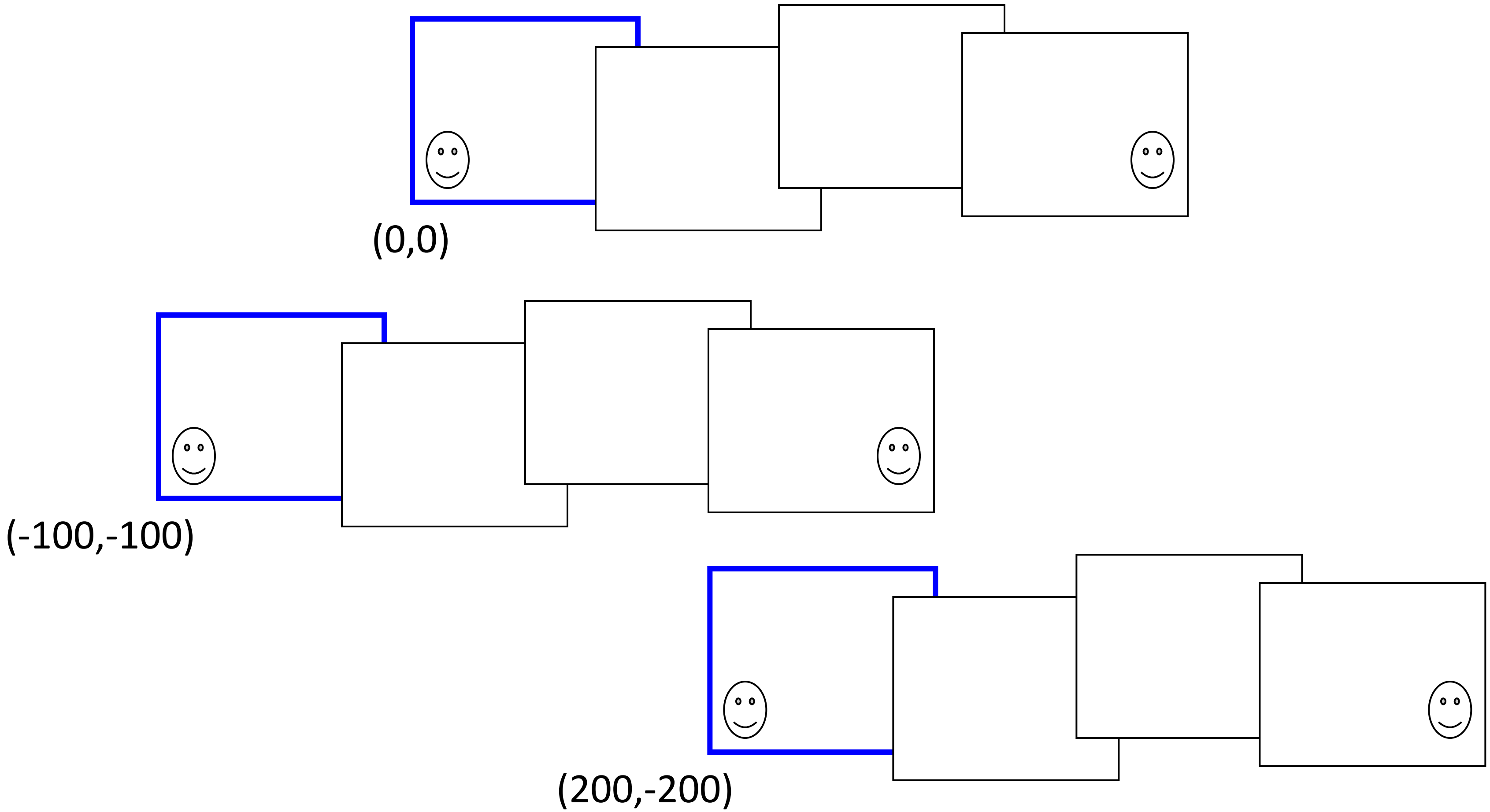
A 2m x 2n **x** 2n x 1 **b** 2m x 1

Defines a least squares problem: minimize $||\mathbf{Ax} - \mathbf{b}||$

- Solution: $\hat{\mathbf{x}} = (\mathbf{A}^T\mathbf{A})^{-1}\mathbf{A}^T\mathbf{b}$
- Problem: there is no unique solution for $\hat{\mathbf{x}}$! (det$(\mathbf{A}^T\mathbf{A})$ = 0)
- We can add a global offset to a solution $\hat{\mathbf{x}}$ and get the same error

Slide: N. Snavely

Ambiguity in global location



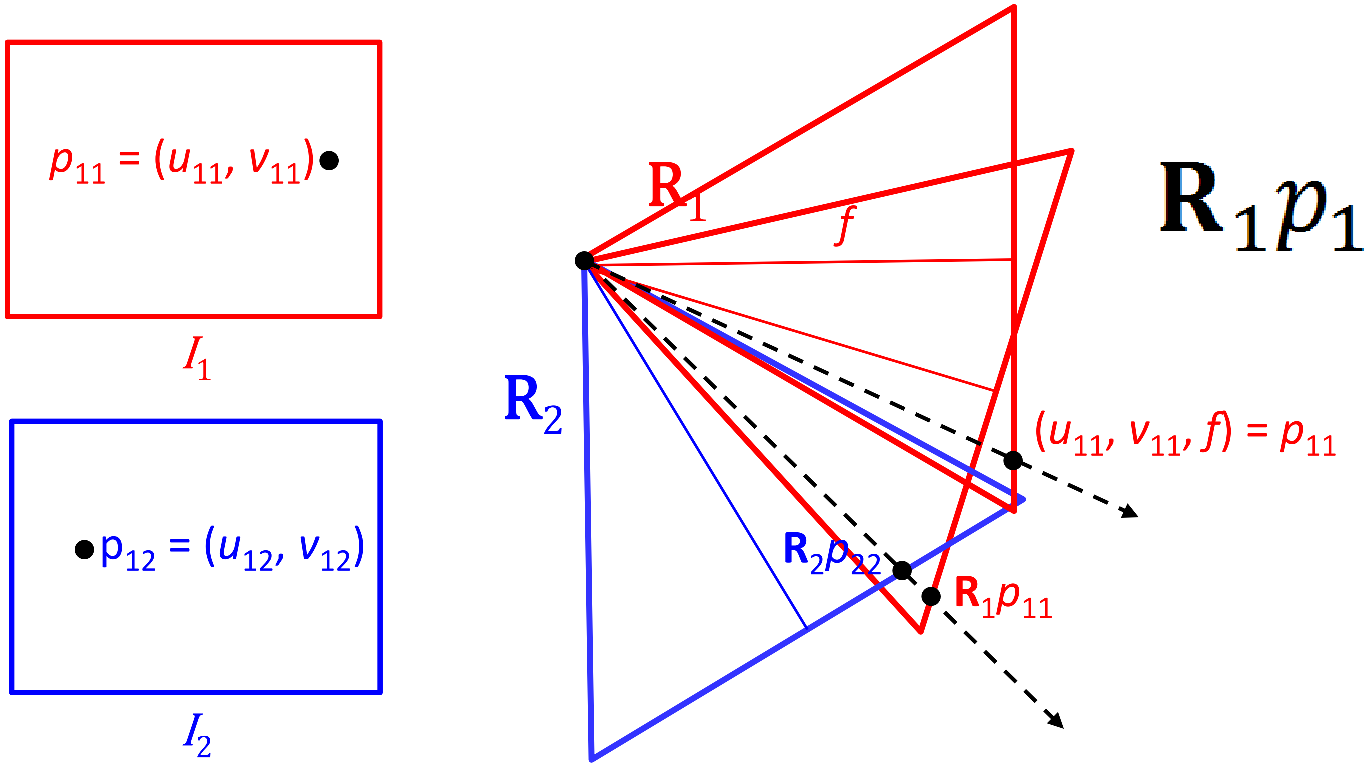
Each of these solutions has the same error
Called the *gauge ambiguity*
Solution: fix the position of one image (e.g., make the origin of the 1st image (0,0))

Solving for camera rotation

Instead of spherically warping the images and solving for translation, we can directly solve for the rotation $\mathbf{R}_j$ of each camera

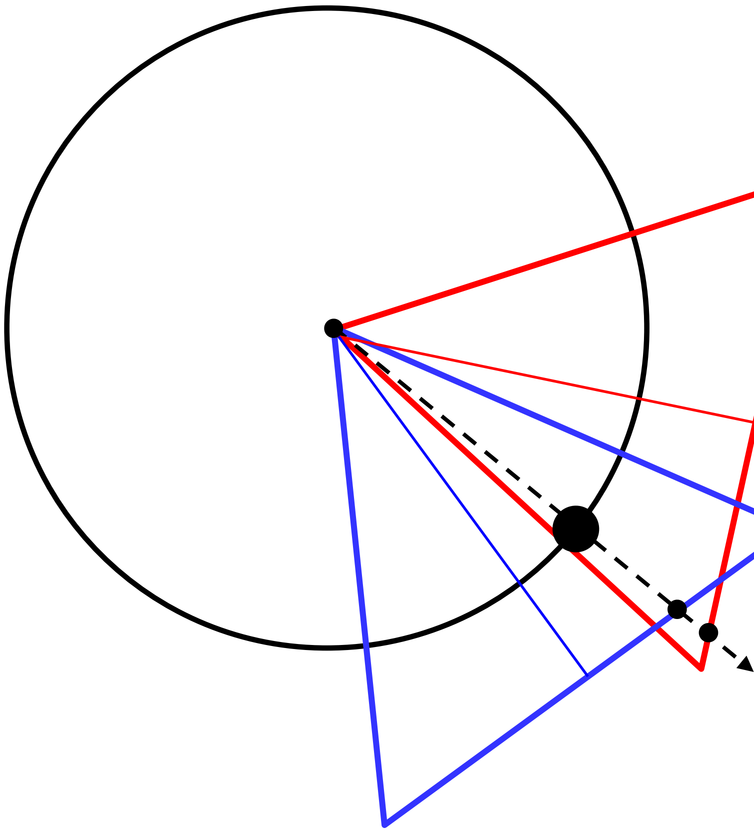
Can handle tilt / twist

Solving for rotations



$$\mathbf{R}_1 p_{11} \cong \mathbf{R}_2 p_{12}$$

Solving for rotations



$$\mathbf{R}_1 p_{11} \cong \mathbf{R}_2 p_{12}$$

$$\mathbf{R}_1 \hat{p}_{11} = \mathbf{R}_2 \hat{p}_{12}$$

$$\text{minimize} \sum_{i=1}^{m} \sum_{j=1}^{n} w_{ij} \cdot \left\| \mathbf{R}_{j+1} \hat{p}_{i,j+1} - \mathbf{R}_j \hat{p}_{i,j} \right\|^2$$

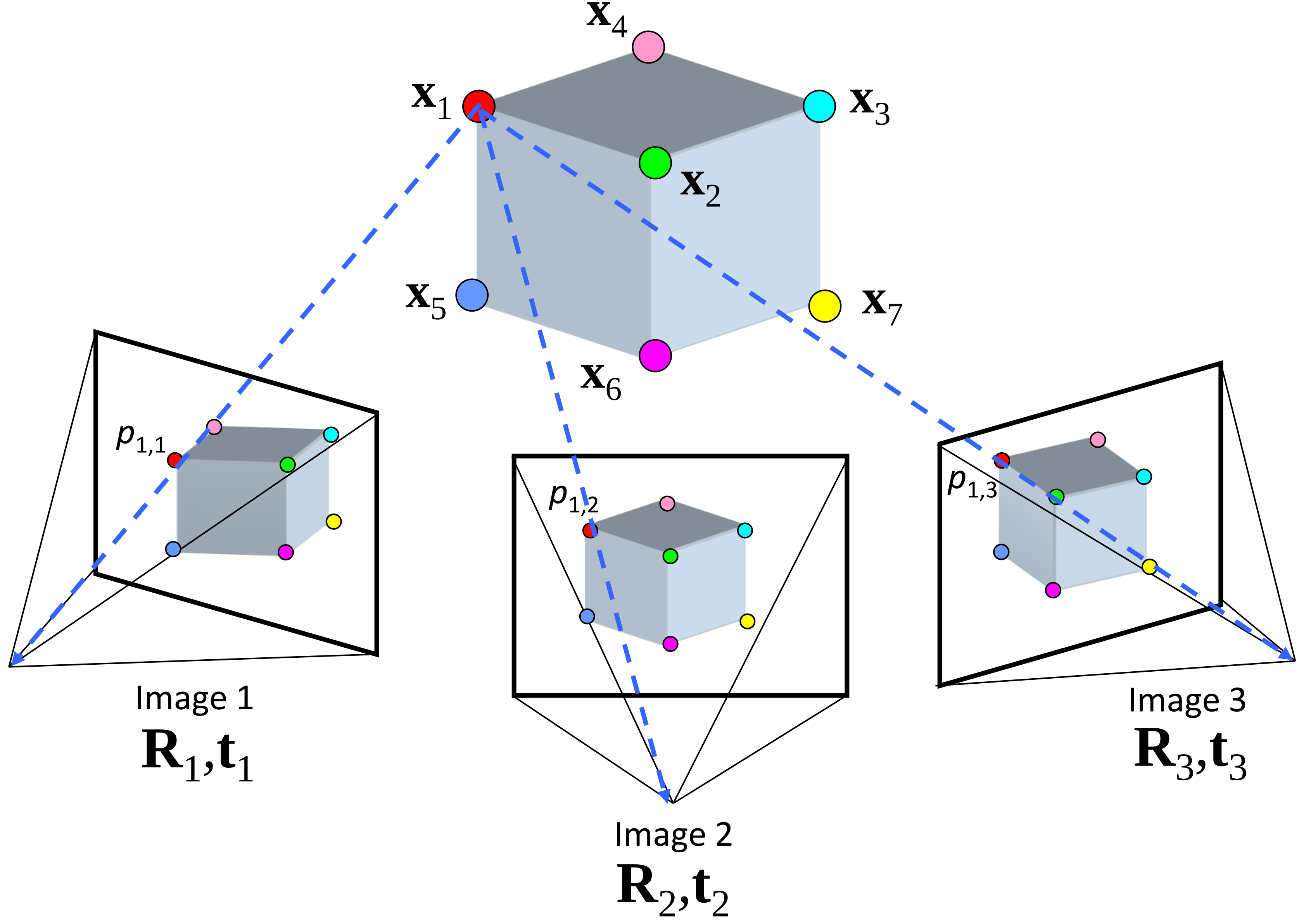
3D rotations

How many degrees of freedom are there?

How do we represent a rotation?

- Rotation matrix (too many degrees of freedom)
- Euler angles (e.g. yaw, pitch, and roll) – bad idea
- Quaternions (4-vector on unit sphere)

Usually involves non-linear optimization



$\mathbf{x}_4$
$\mathbf{x}_1$
$\mathbf{x}_3$
$\mathbf{x}_2$
$\mathbf{x}_5$
$\mathbf{x}_7$
$\mathbf{x}_6$
$p_{1,1}$
$p_{1,2}$
$p_{1,3}$
Image 1
$\mathbf{R}_1,\mathbf{t}_1$
Image 2
$\mathbf{R}_2,\mathbf{t}_2$
Image 3
$\mathbf{R}_3,\mathbf{t}_3$

SfM objective function

Given point **x** and rotation and translation **R**, **t**

$$\begin{bmatrix} x' \\ y' \\ z' \end{bmatrix} = \mathbf{R}\mathbf{x} + \mathbf{t} \qquad u' = \frac{fx'}{z'} \quad v' = \frac{fy'}{z'} \qquad \begin{bmatrix} u' \\ v' \end{bmatrix} = \mathbf{P}(\mathbf{x}, \mathbf{R}, \mathbf{t})$$

Minimize sum of squared reprojection errors:

$$g(\mathbf{X}, \mathbf{R}, \mathbf{T}) = \sum_{i=1}^{m} \sum_{j=1}^{n} w_{ij} \cdot \left\| \underbrace{\mathbf{P}(\mathbf{x}_i, \mathbf{R}_j, \mathbf{t}_j)}_{\substack{\textit{predicted} \\ \text{image location}}} - \underbrace{\begin{bmatrix} u_{i,j} \\ v_{i,j} \end{bmatrix}}_{\substack{\textit{observed} \\ \text{image location}}} \right\|^2$$

Solving structure from motion

Minimizing *g* is difficult

- *g* is non-linear due to rotations, perspective division
- lots of parameters: 3 for each 3D point, 6 for each camera
- difficult to initialize
- gauge ambiguity: error is invariant to a similarity transform (translation, rotation, uniform scale)

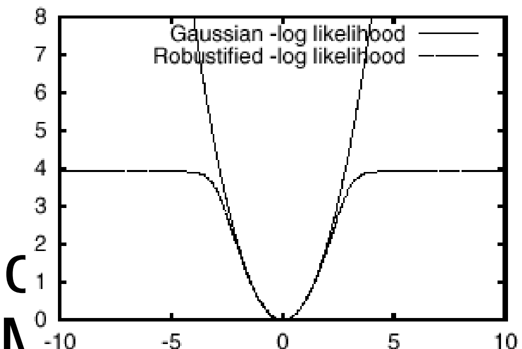
Many techniques use non-linear least-squares (NLLS) optimization (*bundle adjustment*)

- Levenberg-Marquardt is one common algorithm for NLLS
- Lourakis, **The Design and Implementation of a Generic Sparse Bundle Adjustment Software Package Based on the Levenberg-Marquardt Algorithm**, http://www.ics.forth.gr/~lourakis/sba/
- http://en.wikipedia.org/wiki/Levenberg-Marquardt_algorithm

Extensions to SfM

Can also solve for intrinsic parameters (focal length, radial distortion, etc.)

Can use a more robust function than squared error, to avoid fitting to outliers



For more informatic … t al, “Bundle Adjustment – A M … s”, *Vision Algorithms* 2000.