Computer Vision – Lecture 21

Structure-from-Motion

29.01.2015

Bastian Leibe
RWTH Aachen
http://www.vision.rwth-aachen.de

leibe@vision.rwth-aachen.de

Many slides adapted from Svetlana Lazebnik, Martial Hebert, Steve Seitz

Announcements

- Exam
  - 1st Date: Monday, 23.02., 13:30 - 17:30h
  - 2nd Date: Thursday, 26.03., 09:30 - 12:30h
  - Closed-book exam, the core exam time will be 2h.
  - Admission requirement: 50% of the exercise points or passed test exam
  - We will send around an announcement with the exact starting times and places by email.
- Test exam
  - Date: Thursday, 05.02., 09:15 - 10:45h, room UMIC 025
  - Core exam time will be 1h
  - Purpose: Prepare you for the questions you can expect.
  - *Possibility to collect bonus exercise points!*

Announcements (2)

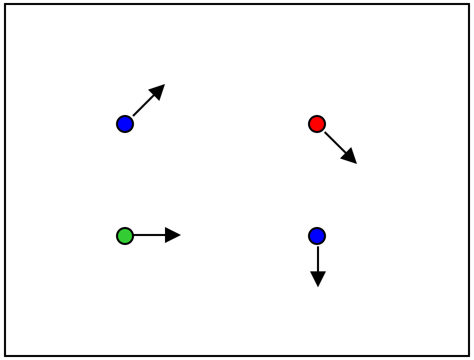
- Last lecture next Monday: Repetition
  - Summary of all topics in the lecture
  - "Big picture" and current research directions
  - Opportunity to ask questions

  - *Please use this opportunity and prepare questions!*

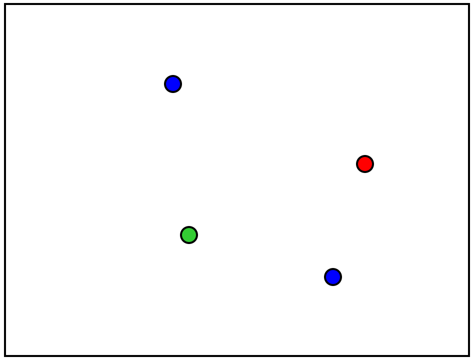
Course Outline

- Image Processing Basics
- Segmentation & Grouping
- Object Recognition
- Local Features & Matching
- Object Categorization
- 3D Reconstruction
  - Epipolar Geometry and Stereo Basics
  - Camera calibration & Uncalibrated Reconstruction
  - Active Stereo
- Motion
  - Motion and Optical Flow
- 3D Reconstruction (Reprise)
  - Structure-from-Motion

Recap: Estimating Optical Flow



$I(x,y,t-1)$



$I(x,y,t)$

- **Given two subsequent frames, estimate the apparent motion field *u*(*x*,*y*) and *v*(*x*,*y*) between them.**
- **Key assumptions**
  - **Brightness constancy: projection of the same point looks the same in every frame.**
  - **Small motion: points do not move very far.**
  - **Spatial coherence: points move like their neighbors.**

Recap: Lucas-Kanade Optical Flow

see Exercise 6.4!

- Use all pixels in a K×K window to get more equations.
- Least squares problem:

$$\begin{bmatrix} I_x(\mathrm{p}_1) & I_y(\mathrm{p}_1) \\ I_x(\mathrm{p}_2) & I_y(\mathrm{p}_2) \\ \vdots & \vdots \\ I_x(\mathrm{p}_{25}) & I_y(\mathrm{p}_{25}) \end{bmatrix} \begin{bmatrix} u \\ v \end{bmatrix} = - \begin{bmatrix} I_t(\mathrm{p}_1) \\ I_t(\mathrm{p}_2) \\ \vdots \\ I_t(\mathrm{p}_{25}) \end{bmatrix} \qquad \underset{25\text{x}2}{A} \; \underset{2\text{x}1}{d} = \underset{25\text{x}1}{b}$$

- Minimum least squares solution given by solution of

$$\underset{2\text{x}2}{(A^T A)} \; \underset{2\text{x}1}{d} = \underset{2\text{x}1}{A^T b}$$

Recall the Harris detector!

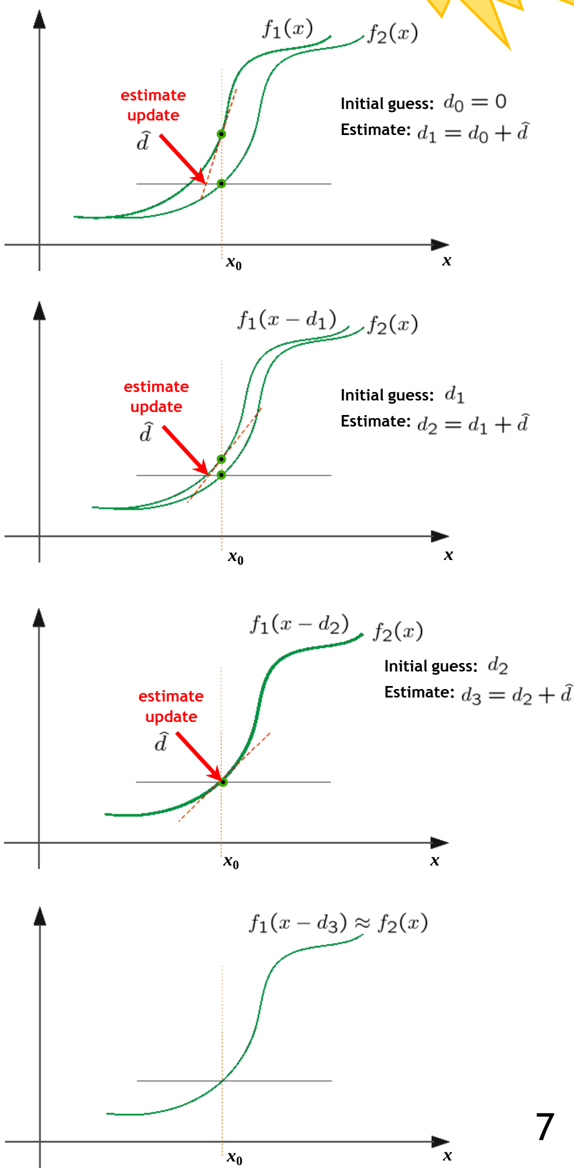
$$\underbrace{\begin{bmatrix} \sum I_x I_x & \sum I_x I_y \\ \sum I_x I_y & \sum I_y I_y \end{bmatrix}}_{A^T A} \begin{bmatrix} u \\ v \end{bmatrix} = - \underbrace{\begin{bmatrix} \sum I_x I_t \\ \sum I_y I_t \end{bmatrix}}_{A^T b}$$

Recap: Iterative Refinement

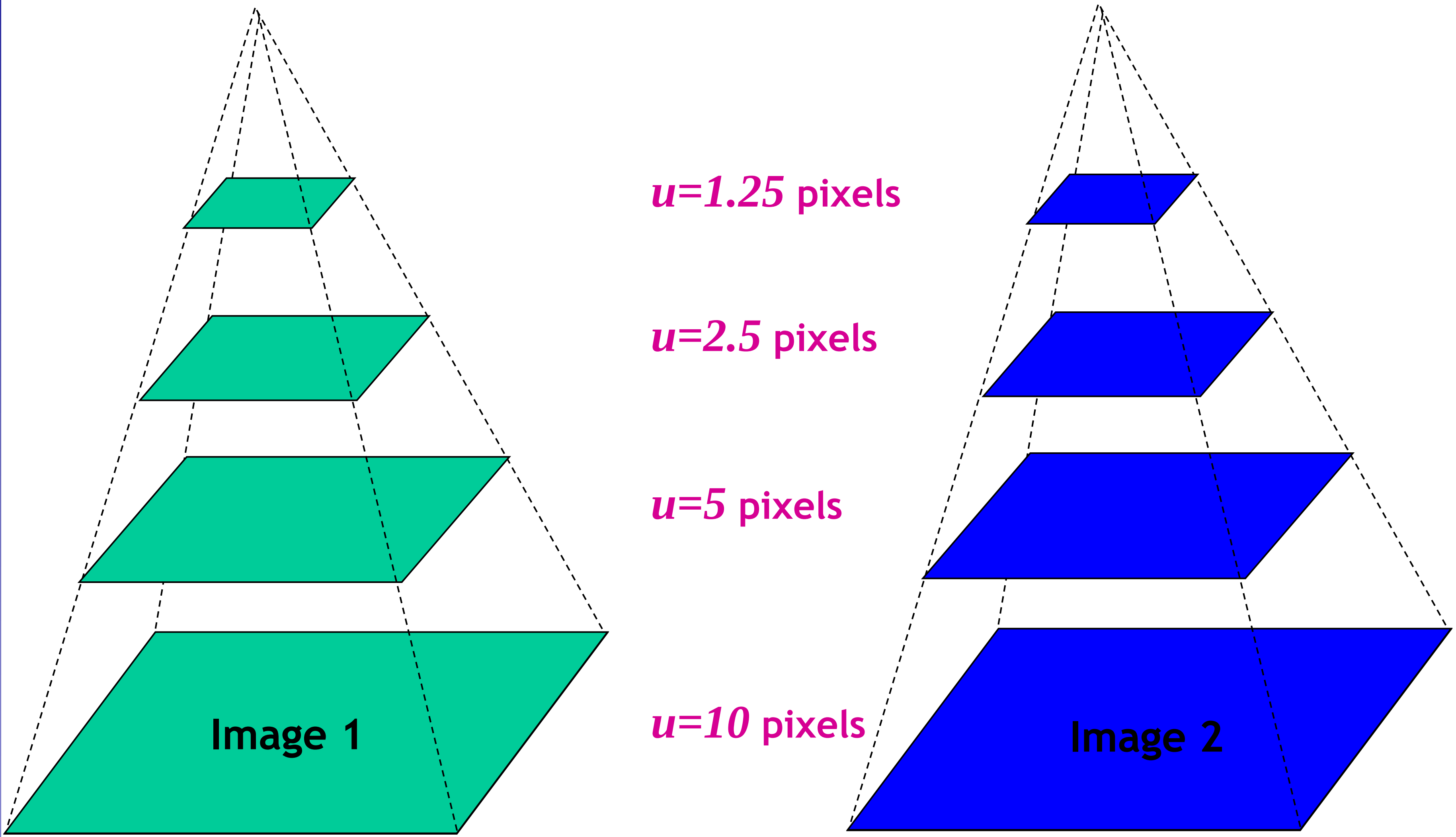
- Estimate velocity at each pixel using one iteration of LK estimation.
- Warp one image toward the other using the estimated flow field.
- Refine estimate by repeating the process.

- Iterative procedure
  - Results in subpixel accurate localization.
  - Converges for small displacements.

see Exercise 6.4!



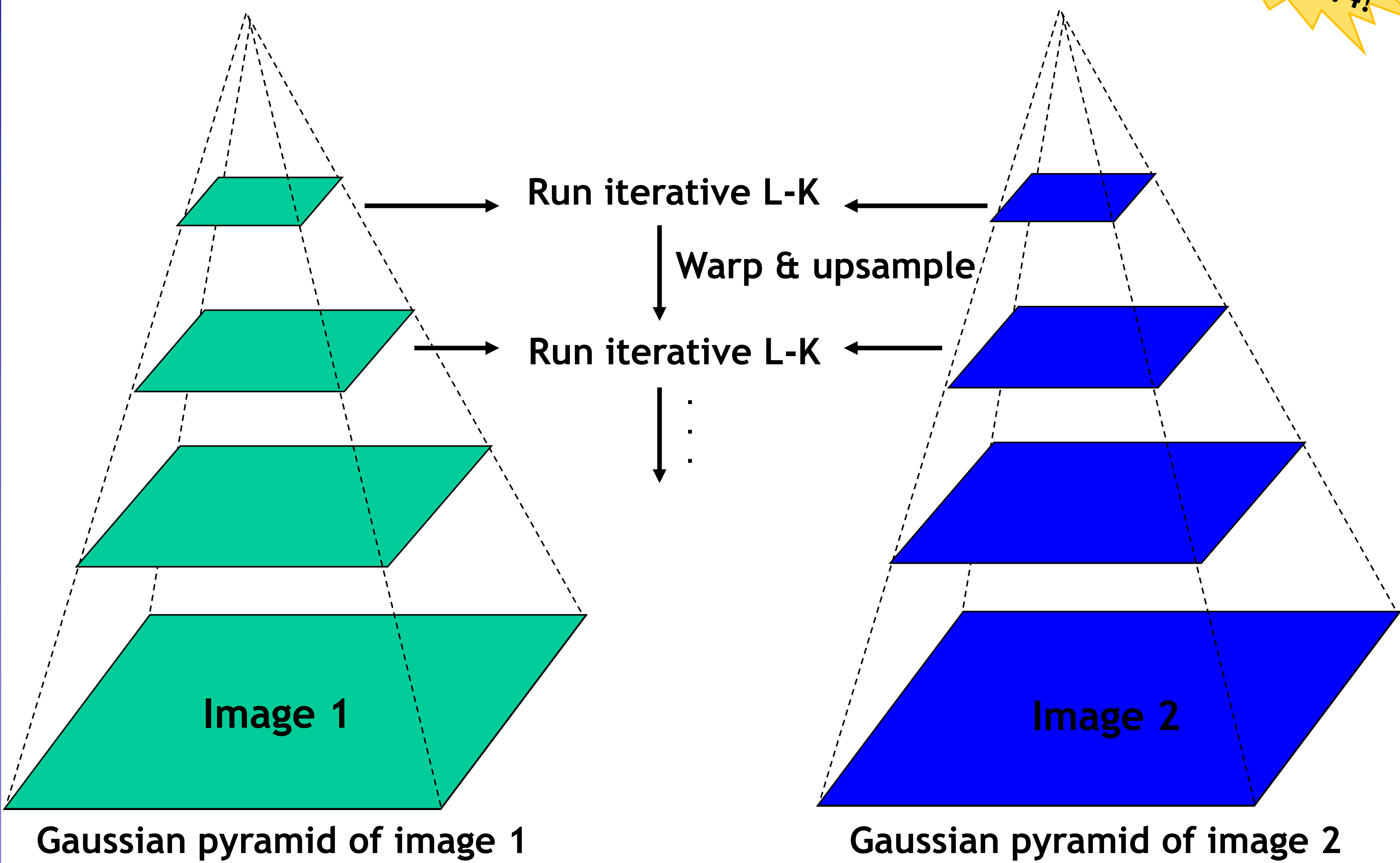
Recap: Coarse-to-fine Estimation



Gaussian pyramid of image 1

Gaussian pyramid of image 2

Recap: Coarse-to-fine Estimation

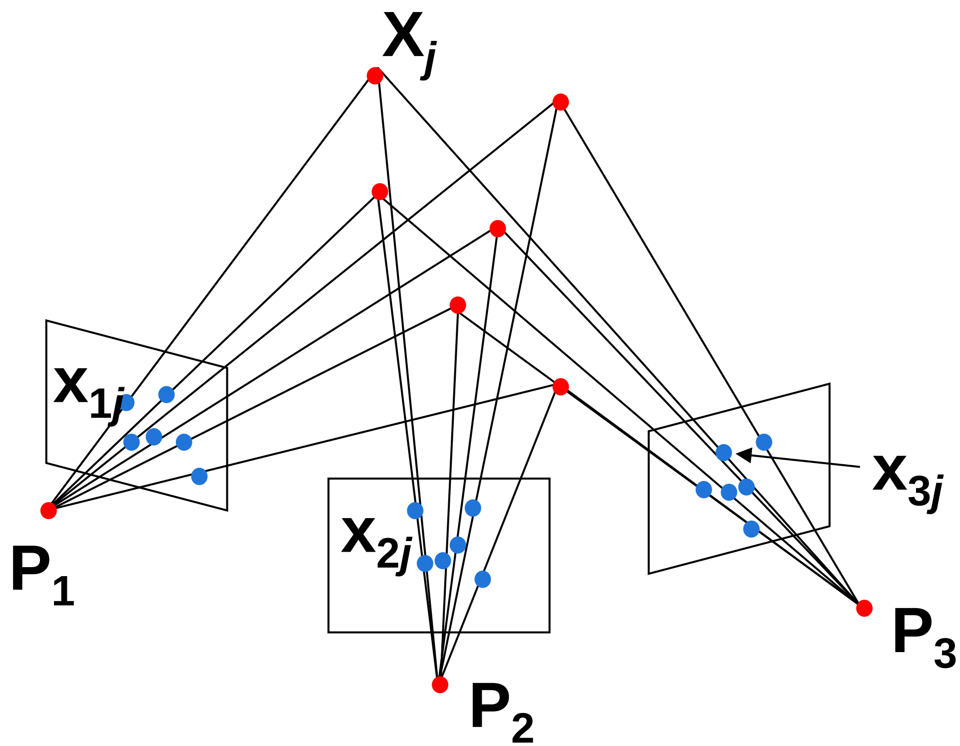


Slide credit: Steve Seitz

Topics of This Lecture

- Structure from Motion (SfM)
  - Motivation
  - Ambiguity
- Affine SfM
  - Affine cameras
  - Affine factorization
  - Euclidean upgrade
  - Dealing with missing data
- Projective SfM
  - Two-camera case
  - Projective factorization
  - Bundle adjustment
  - Practical considerations
- Applications

Structure from Motion



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

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

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

What Can We Use This For?

- E.g. movie special effects

Video

Video Credit: Stefan Hafenegger

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:

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

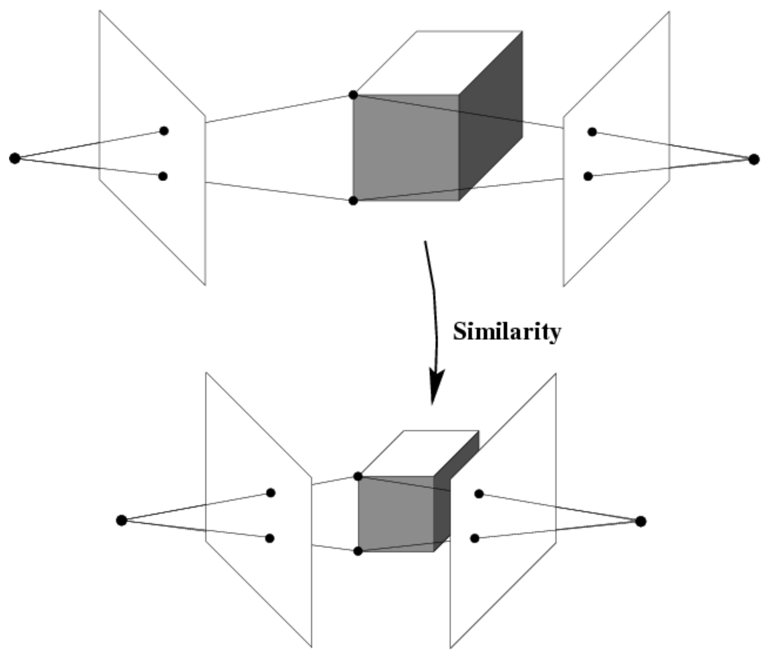
⇒ It is impossible to recover the absolute scale of the scene!

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 $\mathbf{Q}$ and apply the inverse transformation to the camera matrices, then the images do not change

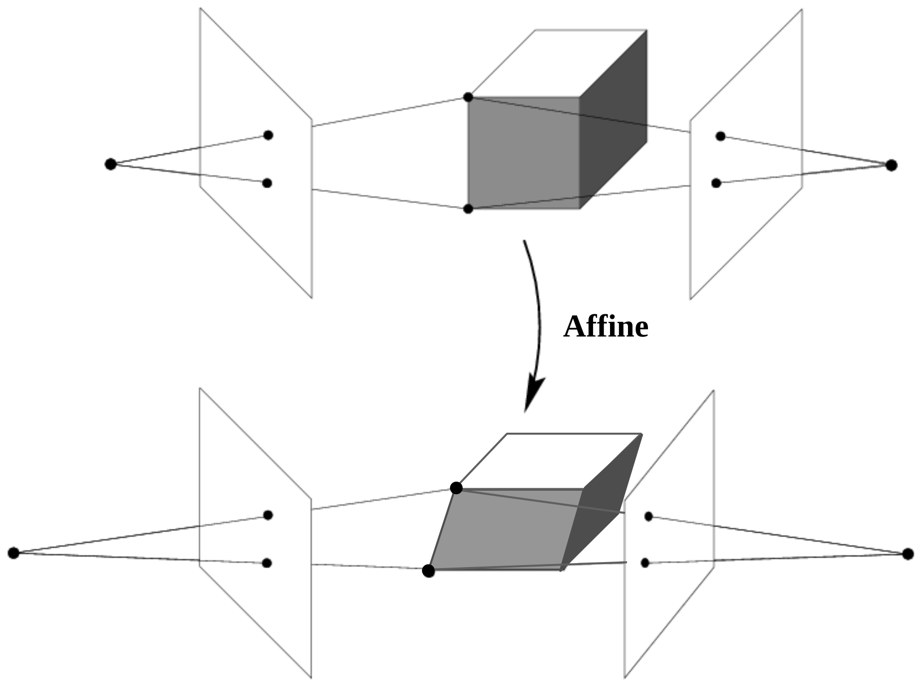
$$\mathbf{x} = \mathbf{P}\mathbf{X} = (\mathbf{P}\mathbf{Q}^{-1})\mathbf{Q}\mathbf{X}$$

Reconstruction Ambiguity: Similarity



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

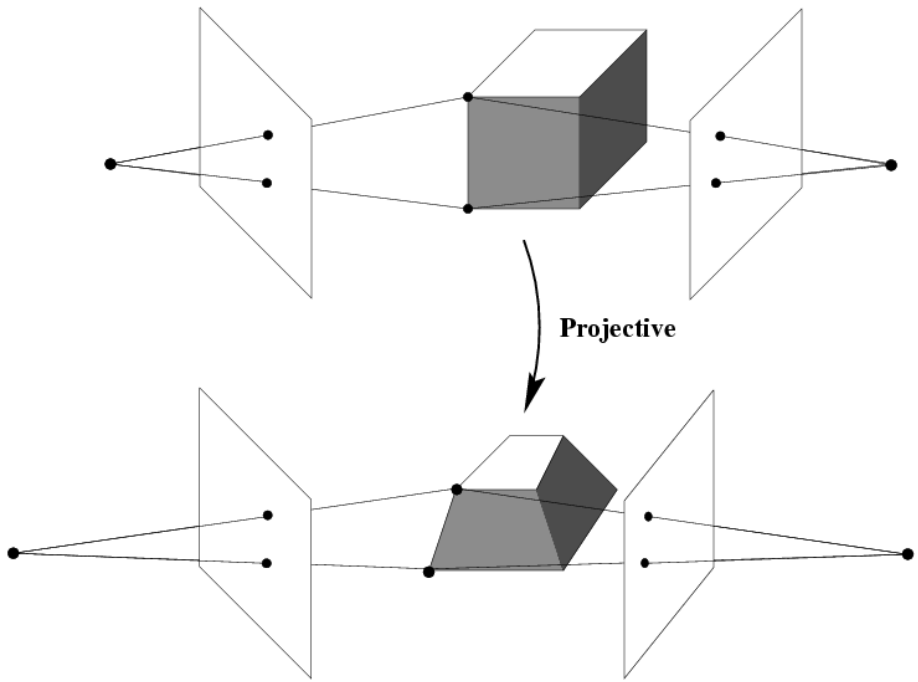
Reconstruction Ambiguity: Affine



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

Images from Hartley & Zisserman

Reconstruction Ambiguity: Projective

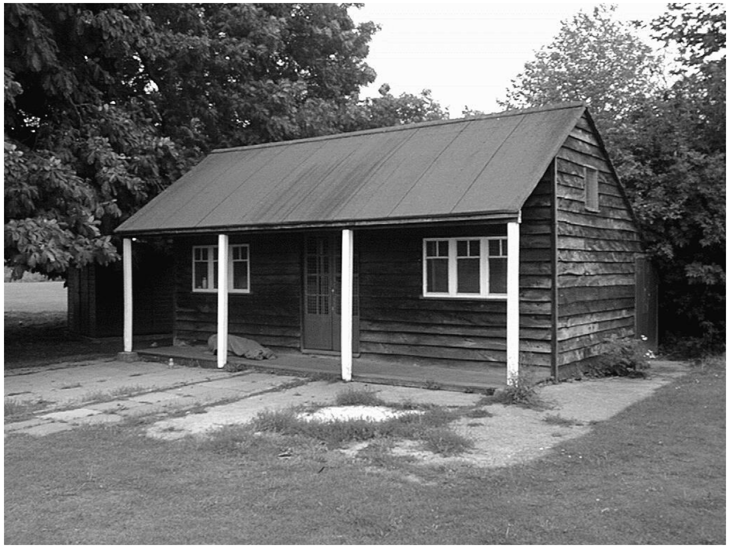


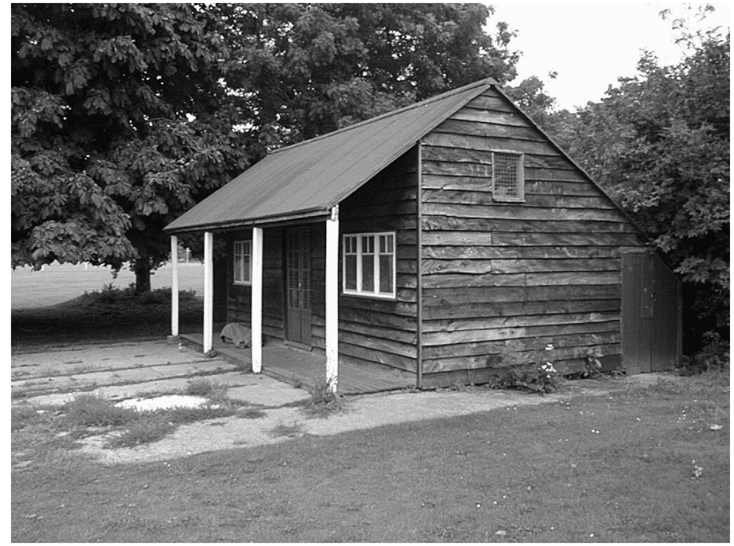
$$\mathbf{x} = \mathbf{P}\mathbf{X} = (\mathbf{P}\mathbf{Q}_P^{-1})\mathbf{Q}_P\mathbf{X}$$

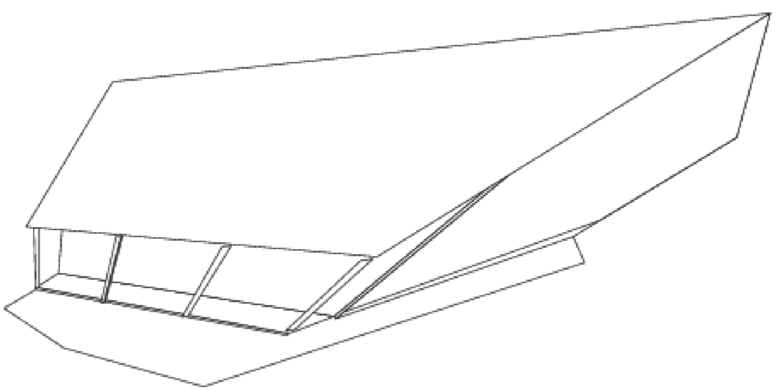
Slide credit: Svetlana Lazebnik

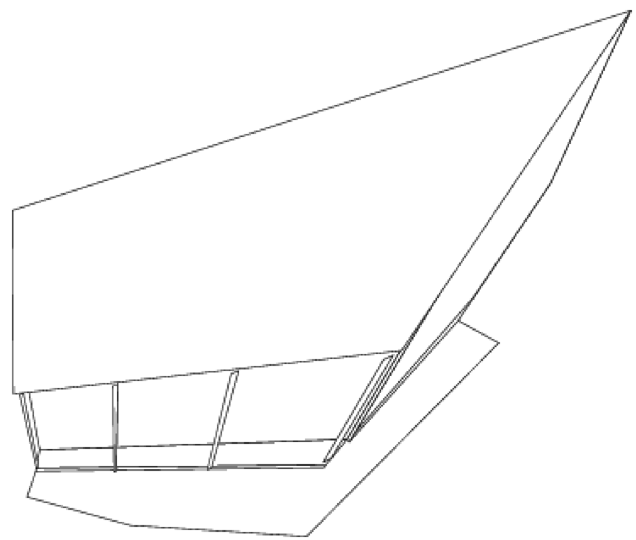
Images from Hartley & Zisserman

Projective Ambiguity

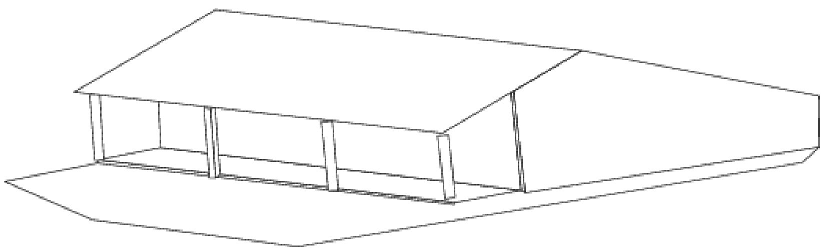


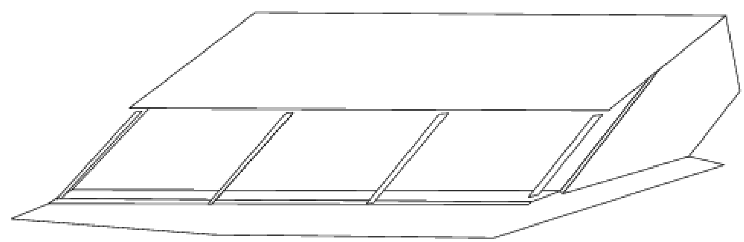






From Projective to Affine

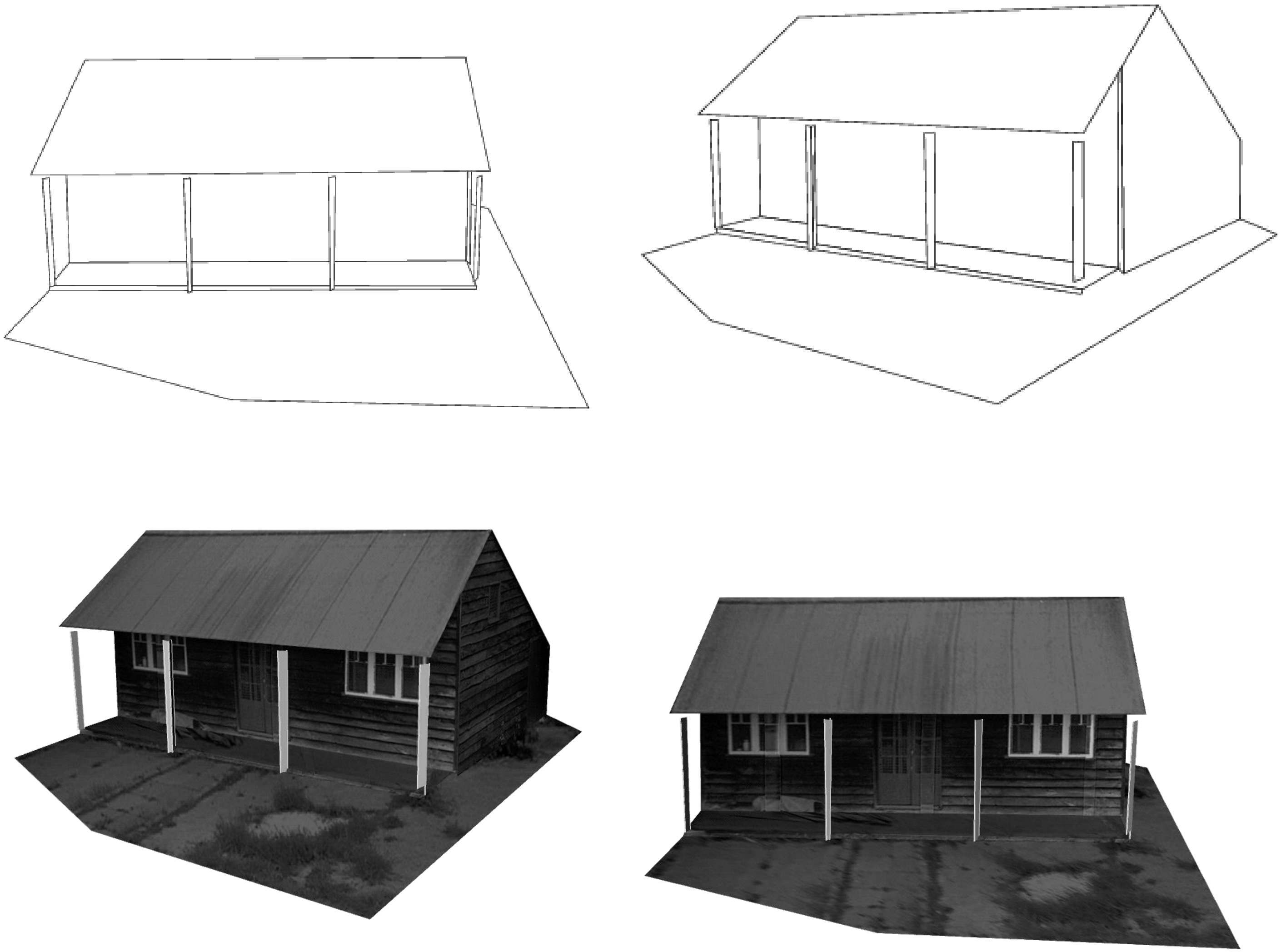




Slide credit: Svetlana Lazebnik

Images from Hartley & Zisserman

From Affine to Similarity

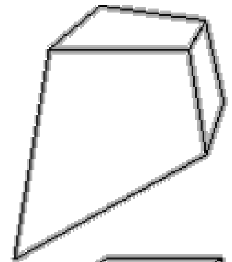


Images from Hartley & Zisserman

Hierarchy of 3D Transformations

Projective
15dof

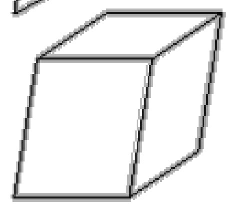
$$\begin{bmatrix} A & t \\ v^T & v \end{bmatrix}$$



Preserves intersection and tangency

Affine
12dof

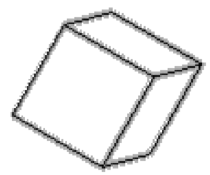
$$\begin{bmatrix} A & t \\ 0^T & 1 \end{bmatrix}$$



Preserves parallellism, volume ratios

Similarity
7dof

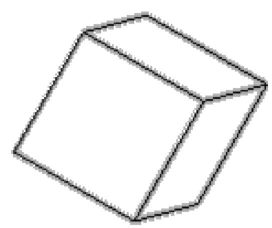
$$\begin{bmatrix} sR & t \\ 0^T & 1 \end{bmatrix}$$



Preserves angles, ratios of length

Euclidean
6dof

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



Preserves angles, lengths

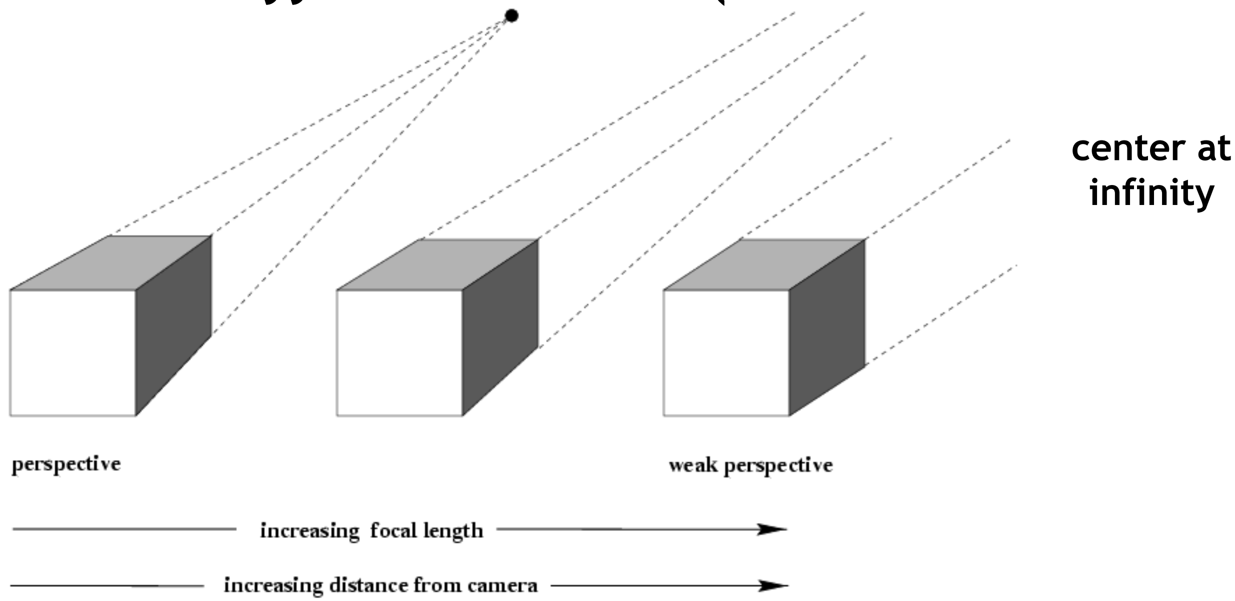
- **With no constraints on the camera calibration matrix or on the scene, we get a *projective* reconstruction.**
- **Need additional information to *upgrade* the reconstruction to affine, similarity, or Euclidean.**

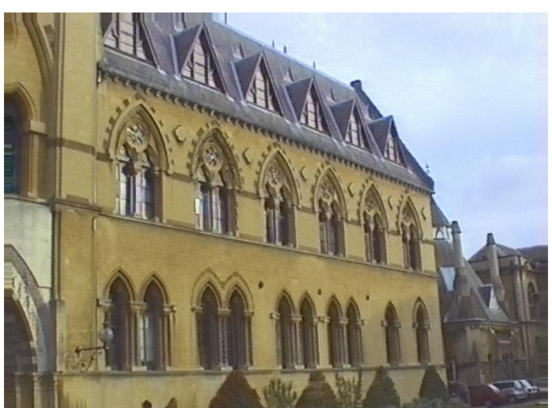
Topics of This Lecture

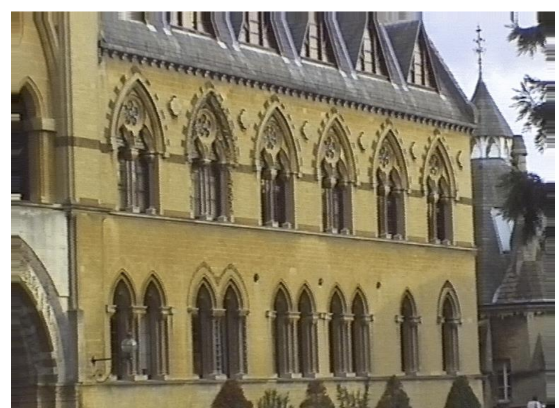
- Structure from Motion (SfM)
  - Motivation
  - Ambiguity
- Affine SfM
  - Affine cameras
  - Affine factorization
  - Euclidean upgrade
  - Dealing with missing data
- Projective SfM
  - Two-camera case
  - Projective factorization
  - Bundle adjustment
  - Practical considerations
- Applications

Structure from Motion

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

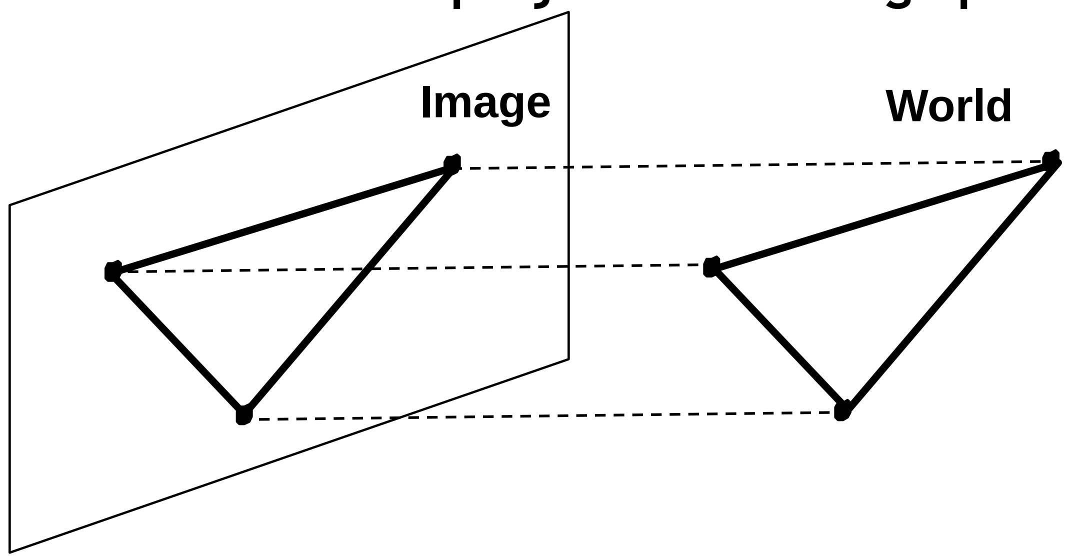






Orthographic Projection

- Special case of perspective projection
  - Distance from center of projection to image plane is infinite



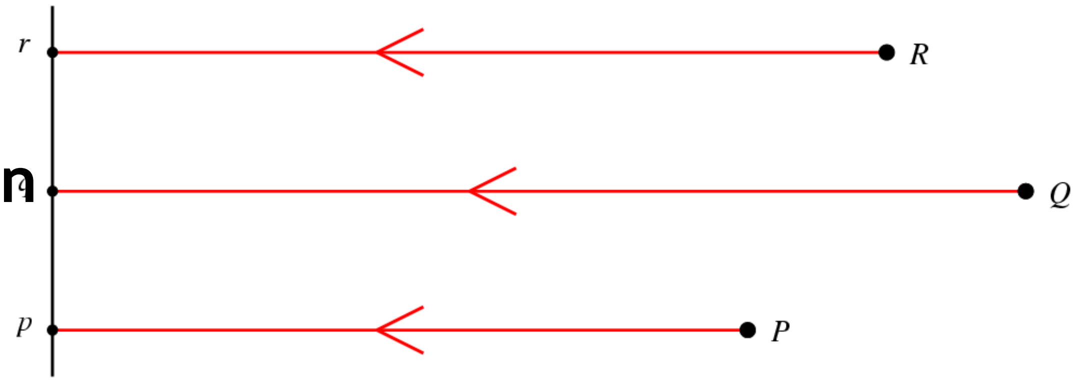
  - Projection matrix:

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

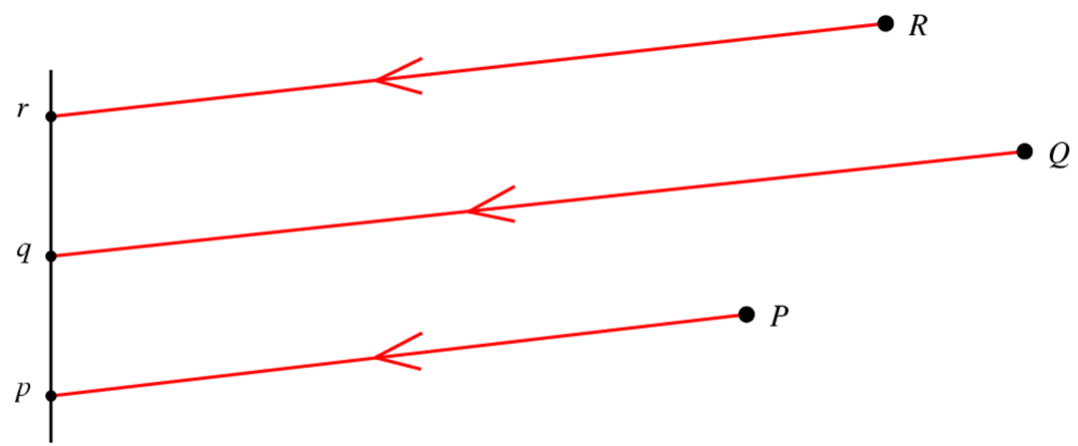
Slide credit: Steve Seitz

Affine Cameras

Orthographic Projection



Parallel Projection

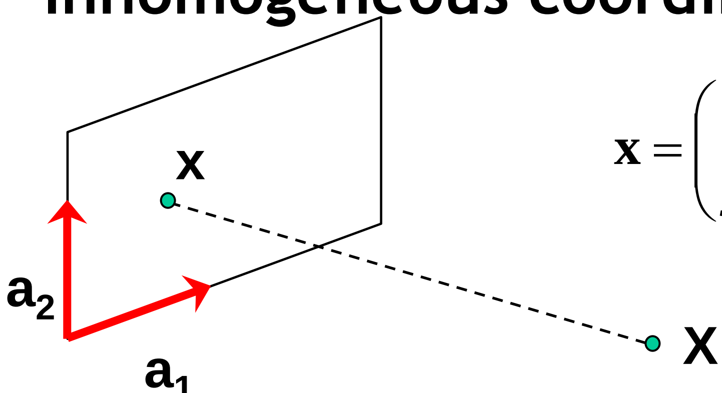


Affine Cameras

- 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 x_{ij} to estimate *m* projection matrices A_i and translation vectors b_i, and *n* points X_j
- The reconstruction is defined up to an arbitrary *affine* transformation Q (12 degrees of freedom):

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

- 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:

$$
\begin{array}{cl}
\mathbf{D} = \begin{bmatrix} \hat{\mathbf{x}}_{11} & \hat{\mathbf{x}}_{12} & \cdots & \hat{\mathbf{x}}_{1n} \\ \hat{\mathbf{x}}_{21} & \hat{\mathbf{x}}_{22} & \cdots & \hat{\mathbf{x}}_{2n} \\ & & \ddots & \\ \hat{\mathbf{x}}_{m1} & \hat{\mathbf{x}}_{m2} & \cdots & \hat{\mathbf{x}}_{mn} \end{bmatrix} & \begin{array}{c} \downarrow \\ \text{Cameras} \\ (2m) \end{array} \\
\xrightarrow{\hspace{6em}} & \\
\text{Points } (n) &
\end{array}
$$

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

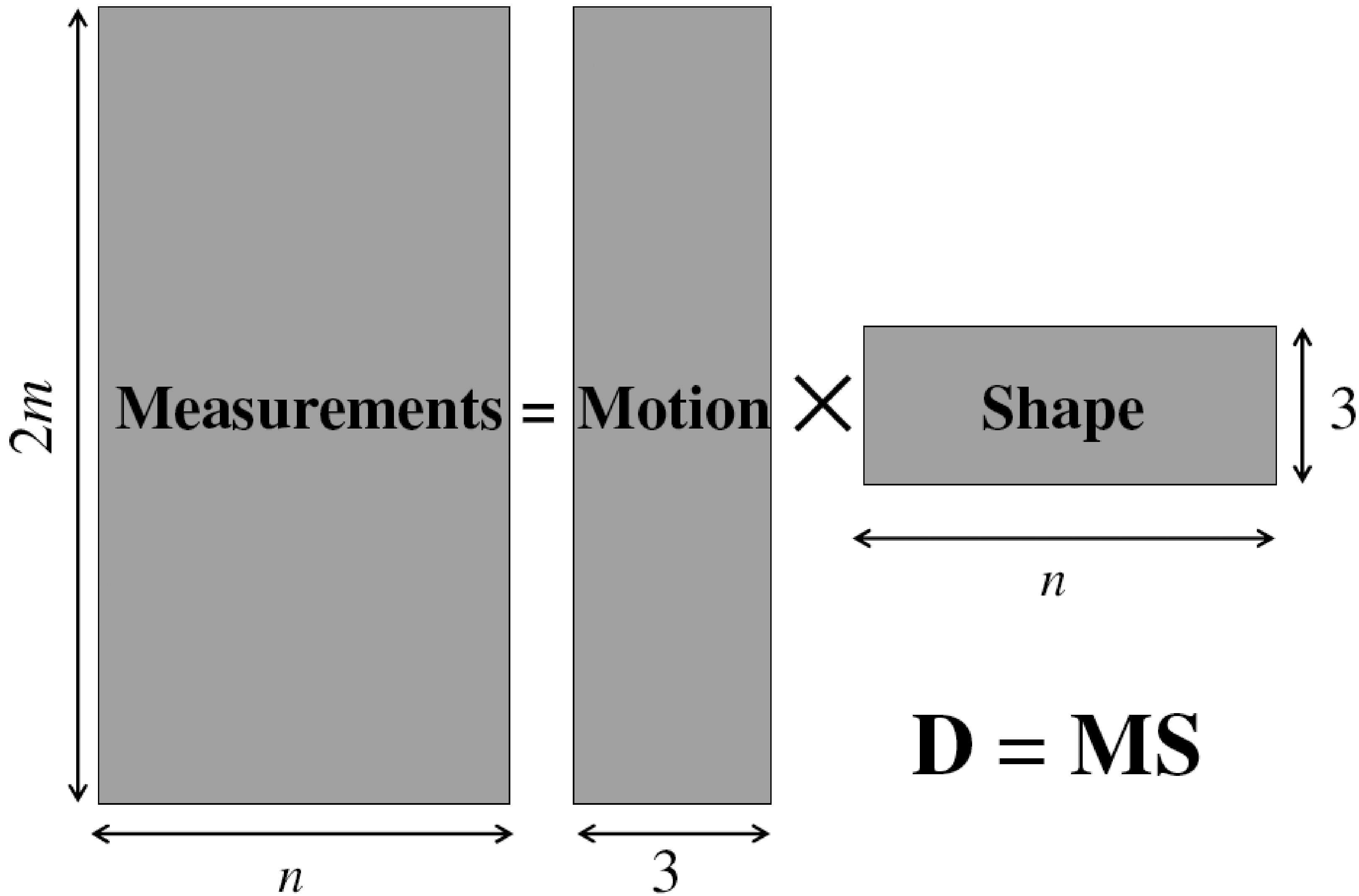
- Let's create a $2m \times n$ data (measurement) matrix:

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

- 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.

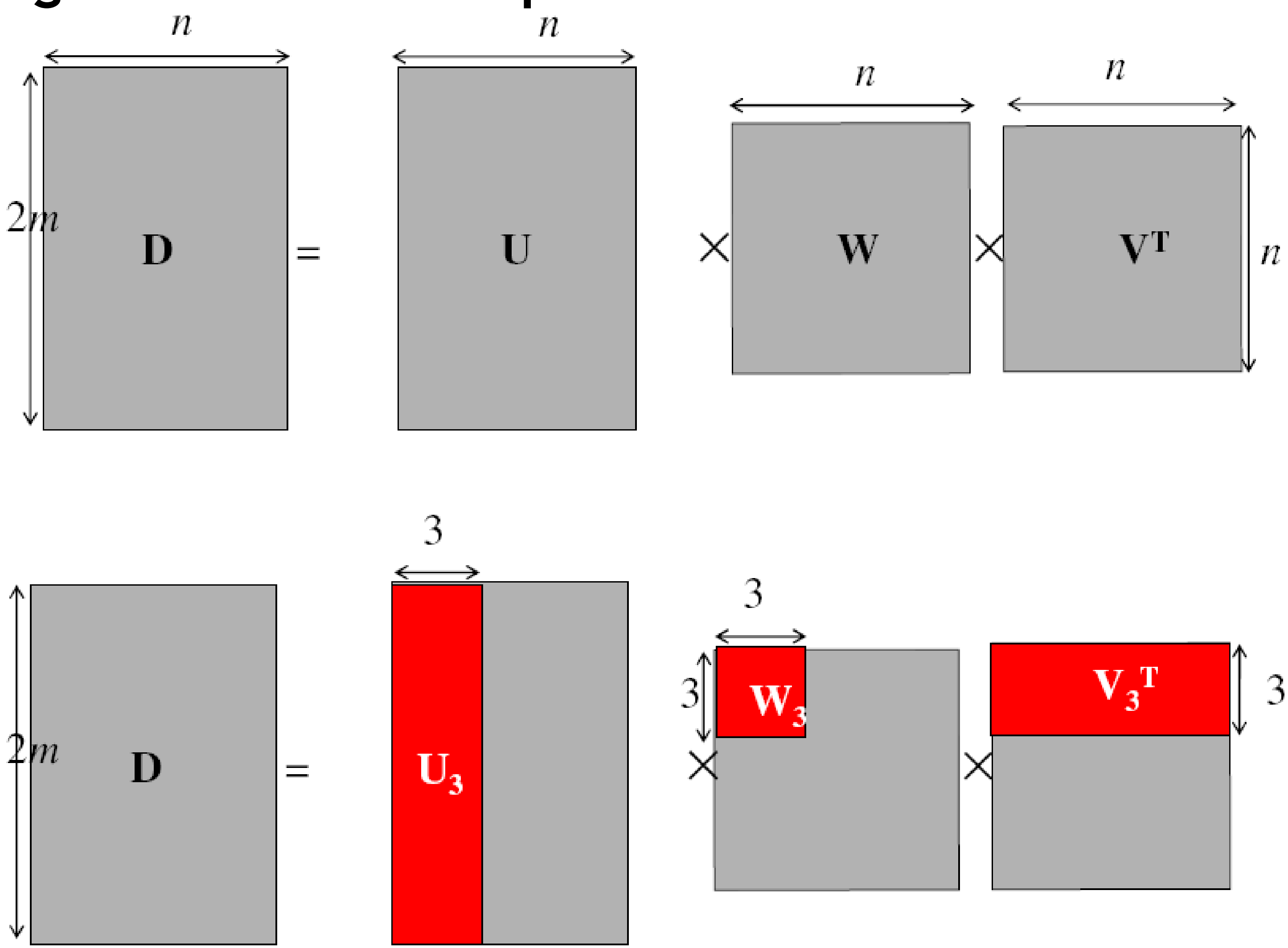
Factorizing the Measurement Matrix



$$\mathbf{D} = \mathbf{MS}$$

Factorizing the Measurement Matrix

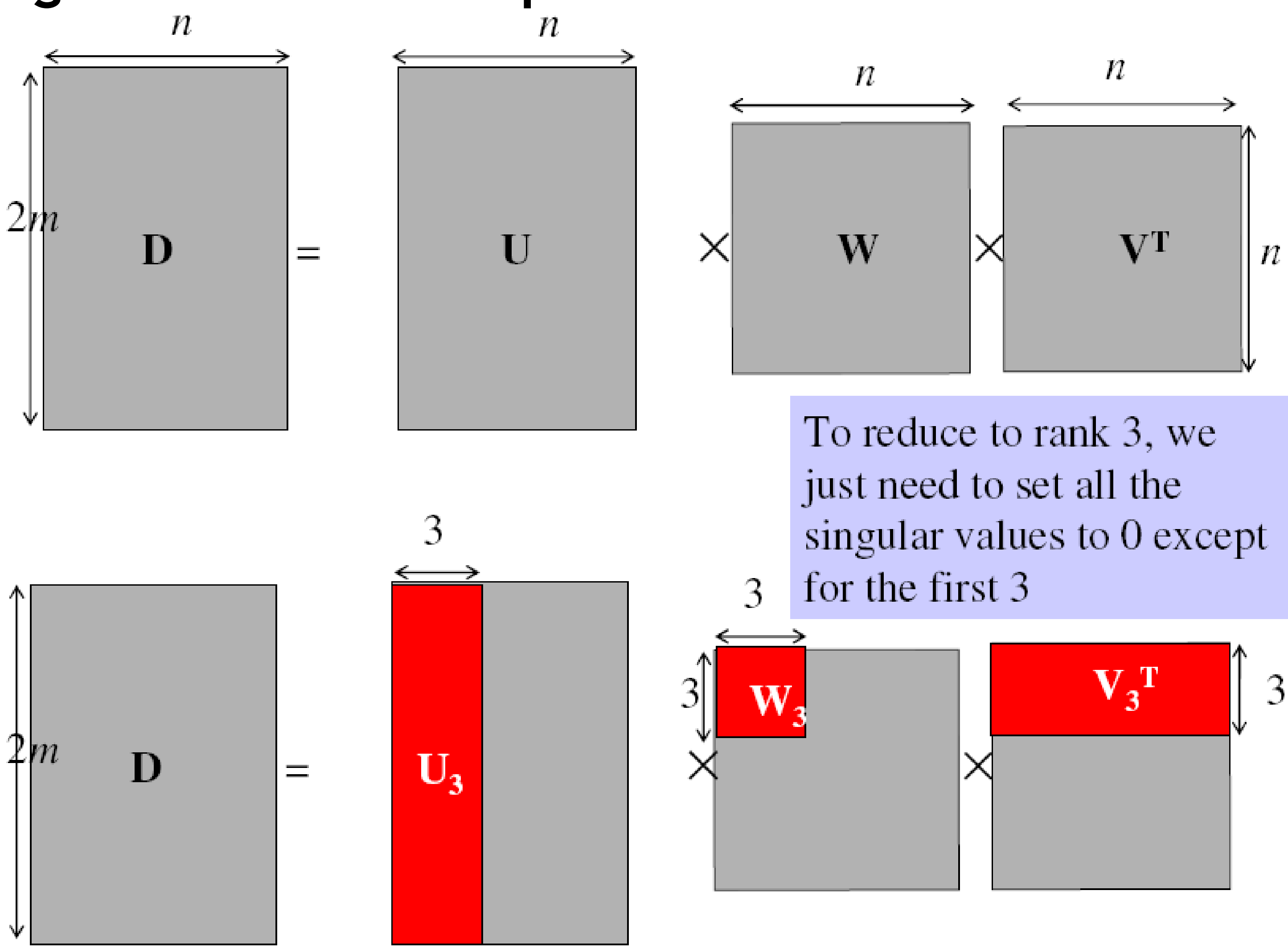
- **Singular value decomposition of D:**



Slide credit: Martial Hebert

Factorizing the Measurement Matrix

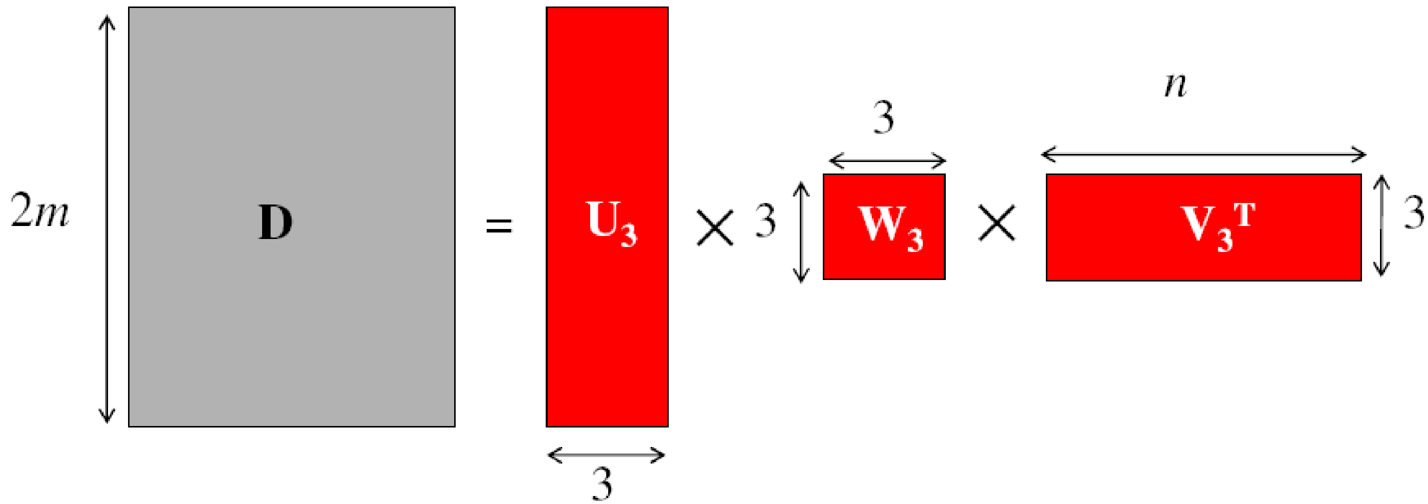
- Singular value decomposition of D:



Slide credit: Martial Hebert

Factorizing the Measurement Matrix

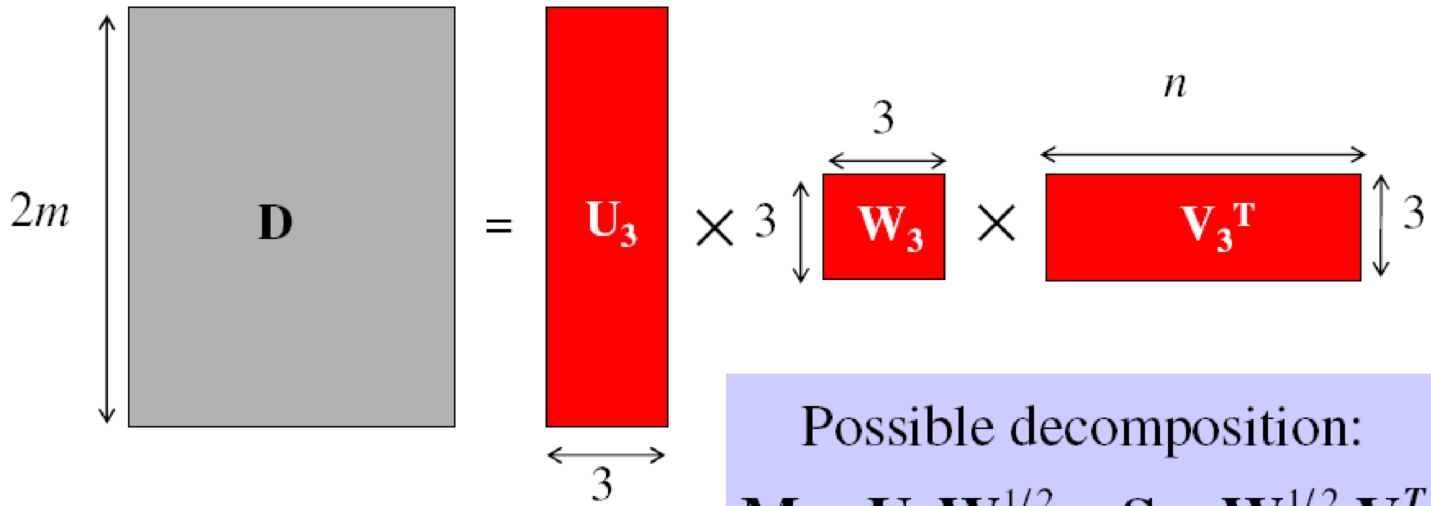
- **Obtaining a factorization from SVD:**



Slide credit: Martial Hebert

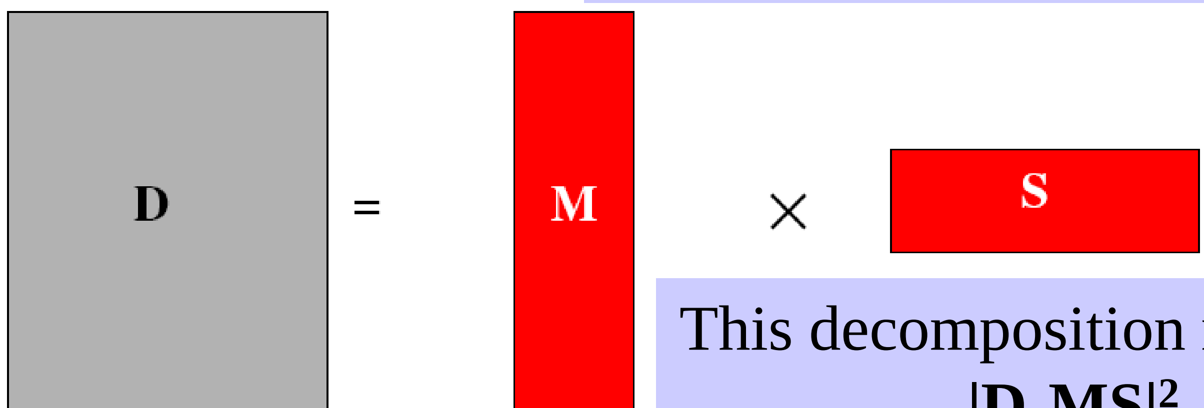
Factorizing the Measurement Matrix

- **Obtaining a factorization from SVD:**



Possible decomposition:

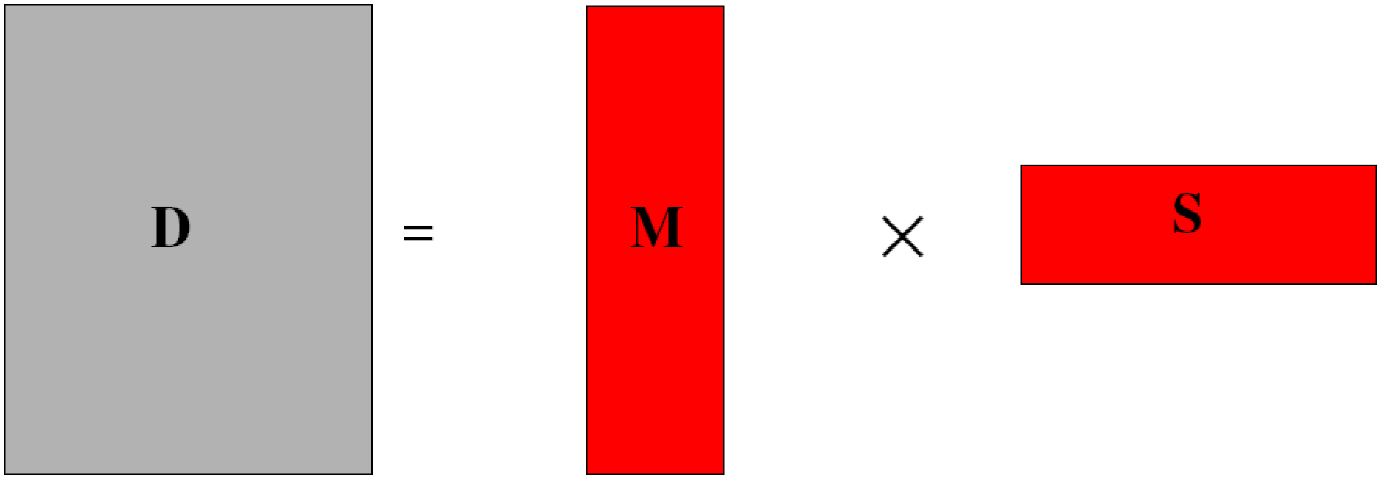
$$\mathbf{M} = \mathbf{U}_3 \mathbf{W}_3^{1/2} \qquad \mathbf{S} = \mathbf{W}_3^{1/2}\, \mathbf{V}_3^T$$



This decomposition minimizes **|D-MS|²**

Slide credit: Martial Hebert

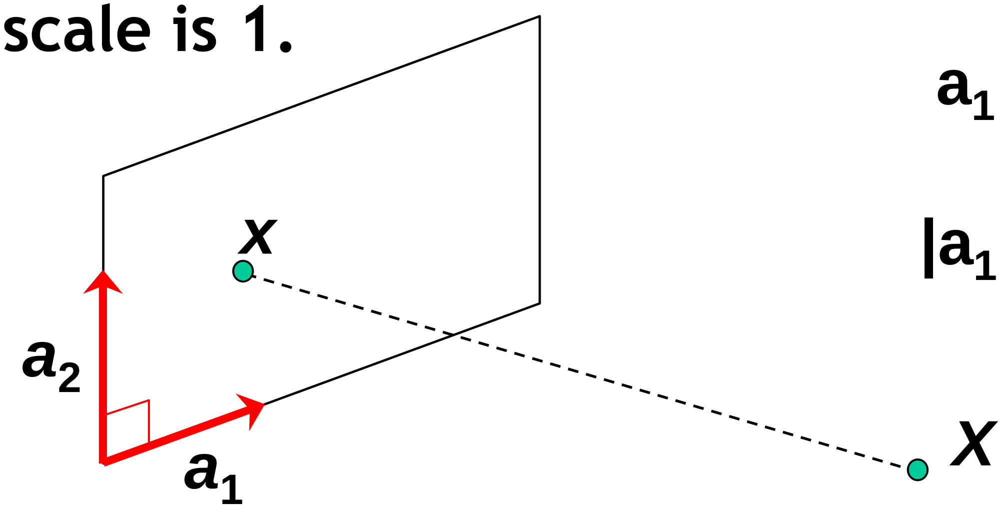
Affine Ambiguity



- The decomposition is not unique. We get the same D by using any 3×3 matrix C and applying the transformations $M \rightarrow MC$, $S \rightarrow C^{-1}S$.
- 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). We need a *Euclidean upgrade*.

Estimating the Euclidean Upgrade

- Orthographic assumption: image axes are perpendicular and scale is 1.



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

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

- This can be converted into a system of 3*m* equations:

$$\begin{cases} \hat{a}_{i1} \cdot \hat{a}_{i2} = 0 \\ |\hat{a}_{i1}| = 1 \\ |\hat{a}_{i2}| = 1 \end{cases} \Leftrightarrow \begin{cases} a_{i1}^T C C^T a_{i2} = 0 \\ a_{i1}^T C C^T a_{i1} = 1, \quad i = 1, \ldots, m \\ a_{i2}^T C C^T a_{i2} = 1 \end{cases}$$

for the transformation matrix C $\Rightarrow$ goal: estimate C

Estimating the Euclidean Upgrade

- **System of 3*m* equations:**

$$\begin{cases} \hat{a}_{i1} \cdot \hat{a}_{i2} = 0 \\ |\hat{a}_{i1}| = 1 \\ |\hat{a}_{i2}| = 1 \end{cases} \Leftrightarrow \begin{cases} a_{i1}^T C C^T a_{i2} = 0 \\ a_{i1}^T C C^T a_{i1} = 1, \quad i = 1, \ldots, m \\ a_{i2}^T C C^T a_{i2} = 1 \end{cases}$$

- **Let** $L = CC^T \qquad A_i = \begin{bmatrix} a_{i1}^T \\ a_{i2}^T \end{bmatrix}, \quad i = 1, \ldots, m$

- **Then this translates to 3*m* equations in L**

$$A_i L A_i^T = I, \quad i = 1, \ldots, m$$

  - **Solve for L**
  - **Recover C from L by Cholesky decomposition: L = CC^T**
  - **Update M and S: M = MC, S = C^{-1}S**

Algorithm Summary

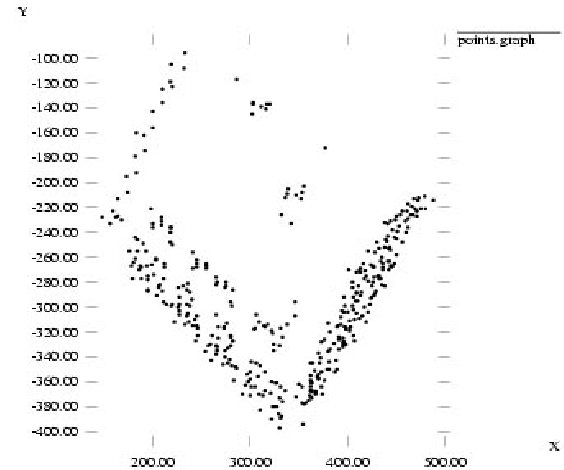
- Given: m images and n features x_{ij}
- For each image i, center the feature coordinates.
- Construct a $2m \times n$ measurement matrix D:
  - Column j contains the projection of point j in all views
  - Row i contains one coordinate of the projections of all the n points in image i
- Factorize D:
  - Compute SVD: $D = U W V^T$
  - Create U_3 by taking the first 3 columns of U
  - Create V_3 by taking the first 3 columns of V
  - Create W_3 by taking the upper left 3×3 block of W
- Create the motion and shape matrices:
  - $M = U_3 W_3^{½}$ and $S = W_3^{½} V_3^T$ (or $M = U_3$ and $S = W_3 V_3^T$)
- Eliminate affine ambiguity

Slide credit: Martial Hebert

Reconstruction Results

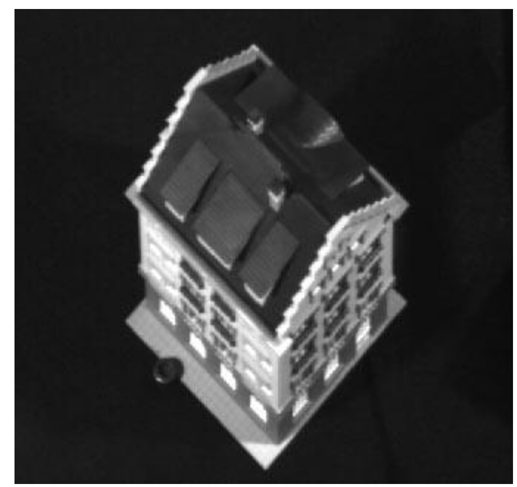
1

60



120

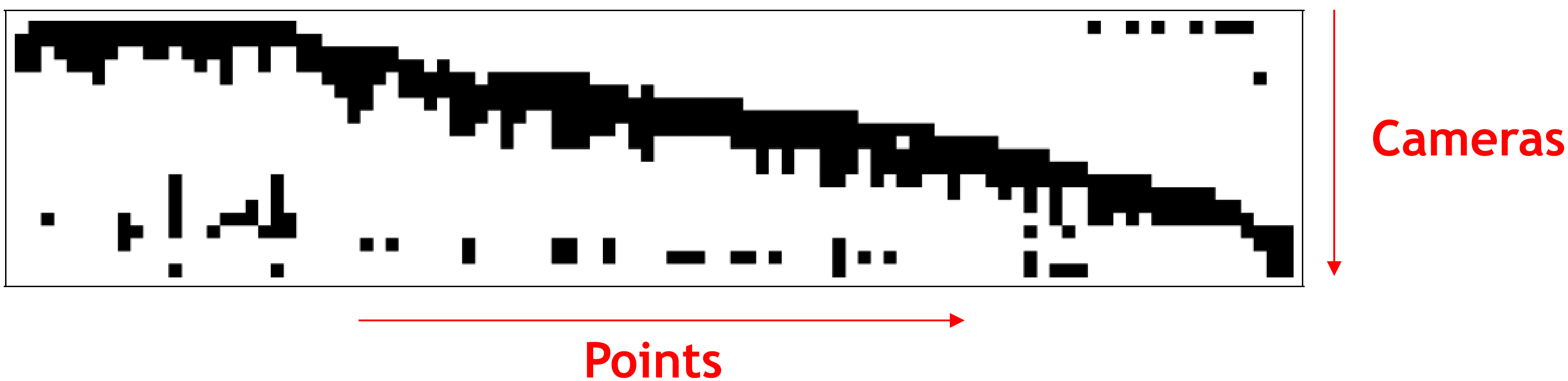
150



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

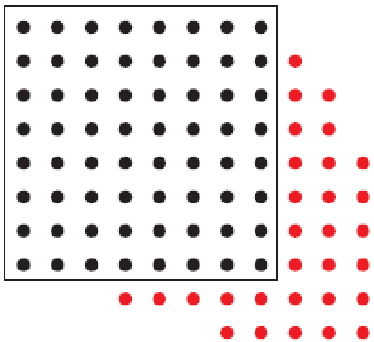
Dealing with Missing Data

- 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**



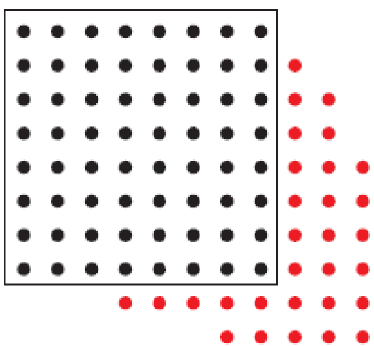
(1) Perform factorization on a dense sub-block

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

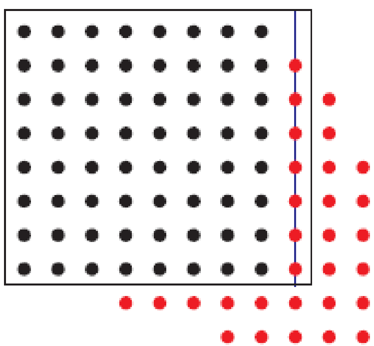
Slide credit: Svetlana Lazebnik

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



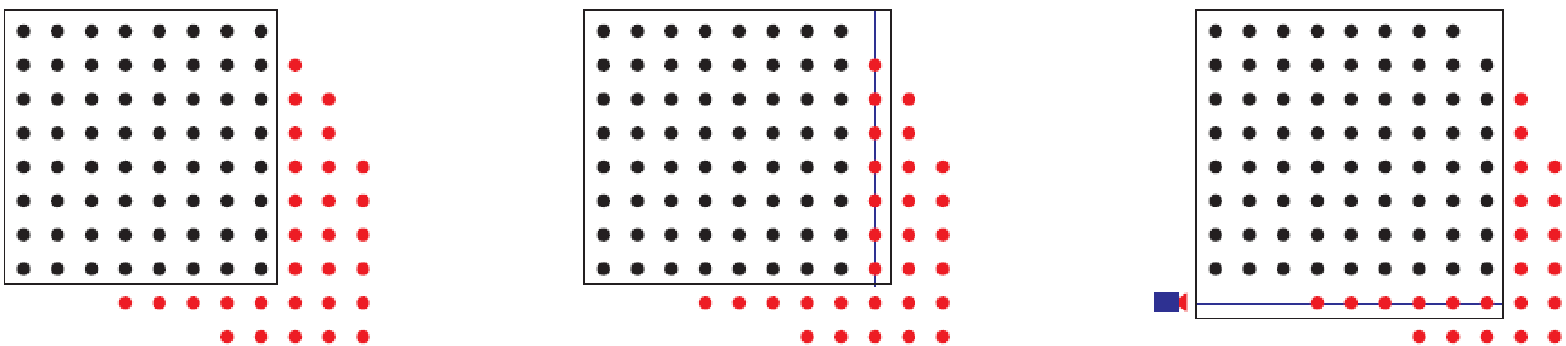
(2) Solve for a new 3D point visible by at least two known cameras (linear least squares)

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

Slide credit: Svetlana Lazebnik

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.

Slide credit: Svetlana Lazebnik

Comments: Affine SfM

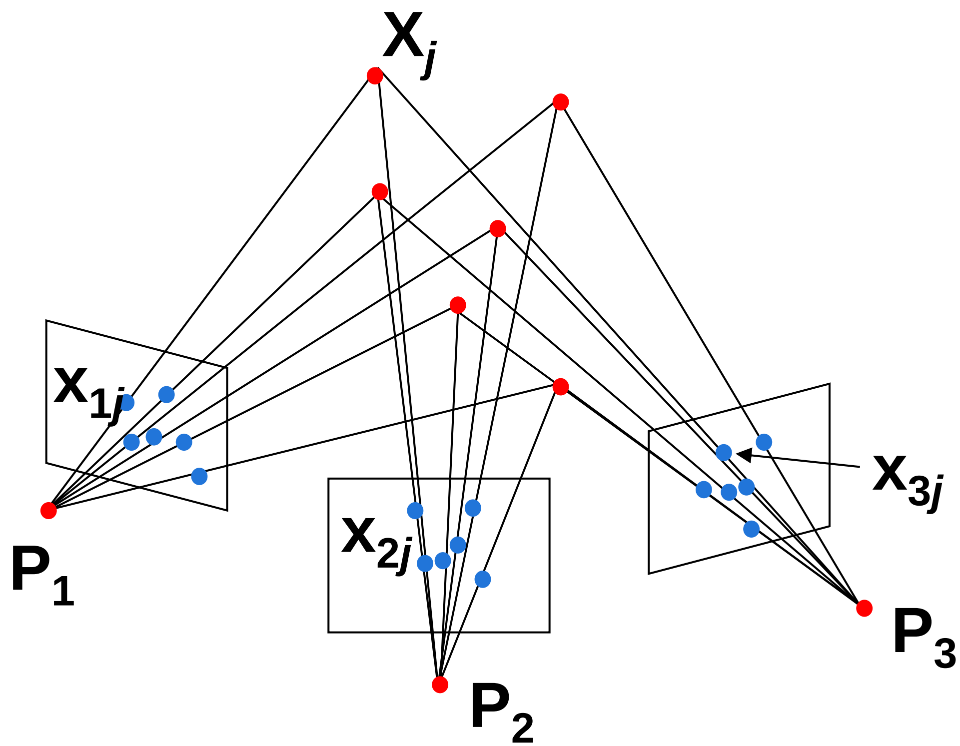
- Affine SfM was historically developed first.
- It is valid under the assumption of *affine cameras*.
  - Which does not hold for real physical cameras…
  - …but which is still tolerable if the scene points are far away from the camera.

- For good results with real cameras, we typically need projective SfM.
  - Harder problem, more ambiguity
  - Math is a bit more involved…
    (Here, only basic ideas. If you want to implement it, please look at the H&Z book for details).

Topics of This Lecture

- Structure from Motion (SfM)
  - Motivation
  - Ambiguity
- Affine SfM
  - Affine cameras
  - Affine factorization
  - Euclidean upgrade
  - Dealing with missing data
- **Projective SfM**
  - **Two-camera case**
  - **Projective factorization**
  - **Bundle adjustment**
  - **Practical considerations**
- Applications

Projective 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 $\mathbf{x}_{ij}$

Projective Structure from Motion

- Given: *m* images of *n* fixed 3D points
  - $z_{ij}\, \mathbf{x}_{ij} = \mathbf{P}_i\, \mathbf{X}_j\,, \qquad i = 1,\ldots, m, \quad j = 1, \ldots, n$
- Problem: estimate *m* projection matrices P_i and *n* 3D points X_j from the *mn* correspondences x_{ij}
- With no calibration info, cameras and points can only be recovered up to a 4×4 projective transformation Q:

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

- 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

- Assume fundamental matrix $\mathbf{F}$ between the two views
  - First camera matrix: $[\mathbf{I}|\mathbf{0}]\mathbf{Q}^{-1}$
  - Second camera matrix: $[\mathbf{A}|\mathbf{b}]\mathbf{Q}^{-1}$
- Let $\tilde{\mathbf{X}} = \mathbf{Q}\mathbf{X}$, then $z\boldsymbol{x} = [\boldsymbol{I}\,|\,\boldsymbol{0}]\tilde{\boldsymbol{X}}, \quad z'\boldsymbol{x}' = [\boldsymbol{A}|\boldsymbol{b}]\tilde{\boldsymbol{X}}$
- And

$$z'\boldsymbol{x}' = \boldsymbol{A}[\boldsymbol{I}\,|\,\boldsymbol{0}]\tilde{\boldsymbol{X}} + \boldsymbol{b} = z\boldsymbol{A}\boldsymbol{x} + \boldsymbol{b}$$

$$z'\boldsymbol{x}' \times \boldsymbol{b} = z\boldsymbol{A}\boldsymbol{x} \times \boldsymbol{b}$$

$$(z'\boldsymbol{x}' \times \boldsymbol{b}) \cdot \boldsymbol{x}' = (z\boldsymbol{A}\boldsymbol{x} \times \boldsymbol{b}) \cdot \boldsymbol{x}'$$

$$0 = (z\boldsymbol{A}\boldsymbol{x} \times \boldsymbol{b}) \cdot \boldsymbol{x}'$$

- So we have $\mathbf{x}'^{\mathrm{T}}[\mathbf{b}_\times]\mathbf{A}\mathbf{x} = 0$

$\mathbf{F} = [\mathbf{b}_\times]\mathbf{A}$ **b: epipole ($\mathbf{F}^{\mathrm{T}}\mathbf{b} = 0$),** $\mathbf{A} = -[\mathbf{b}_\times]\mathbf{F}$

F&P sec. 13.3.1

Projective SfM: Two-Camera Case

- This means that if we can compute the fundamental matrix between two cameras, we can directly estimate the two projection matrices from $\mathbf{F}$.
- Once we have the projection matrices, we can compute the 3D position of any point $\mathbf{X}$ by triangulation.
- How can we obtain both kinds of information at the same time?

Projective Factorization

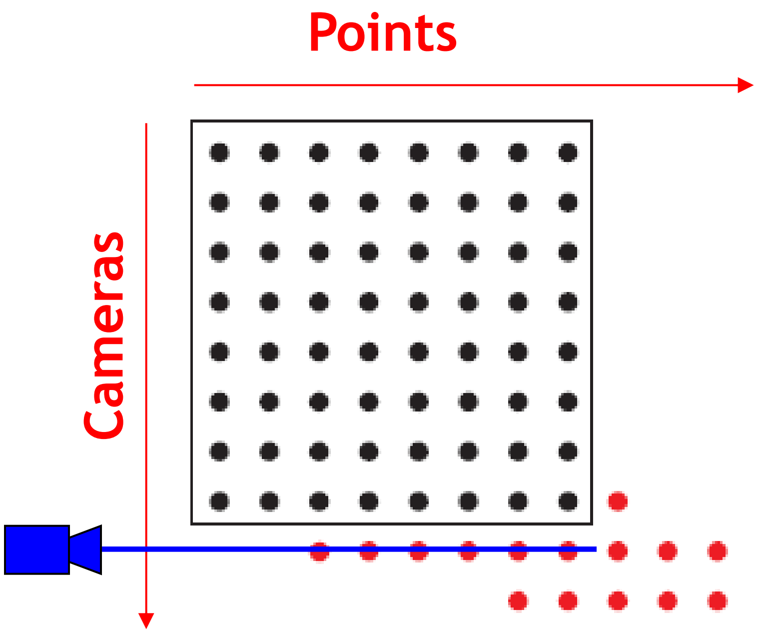
$$\mathbf{D} = \begin{bmatrix} z_{11}\mathbf{x}_{11} & z_{12}\mathbf{x}_{12} & \cdots & z_{1n}\mathbf{x}_{1n} \\ z_{21}\mathbf{x}_{21} & z_{22}\mathbf{x}_{22} & \cdots & z_{2n}\mathbf{x}_{2n} \\ & & \ddots & \\ z_{m1}\mathbf{x}_{m1} & z_{m2}\mathbf{x}_{m2} & \cdots & z_{mn}\mathbf{x}_{mn} \end{bmatrix} = \underbrace{\begin{bmatrix} \mathbf{P}_1 \\ \mathbf{P}_2 \\ \vdots \\ \mathbf{P}_m \end{bmatrix}}_{\text{Cameras } (3m \times 4)} \underbrace{\begin{bmatrix} \mathbf{X}_1 & \mathbf{X}_2 & \cdots & \mathbf{X}_n \end{bmatrix}}_{\text{Points } (4 \times n)}$$

$\mathbf{D} = \mathbf{MS}$ has rank 4

- If we knew the depths *z*, we could factorize D to estimate M and S.
- If we knew M and S, we could solve for *z*.
- Solution: iterative approach (alternate between above two steps).

Sequential Structure from Motion

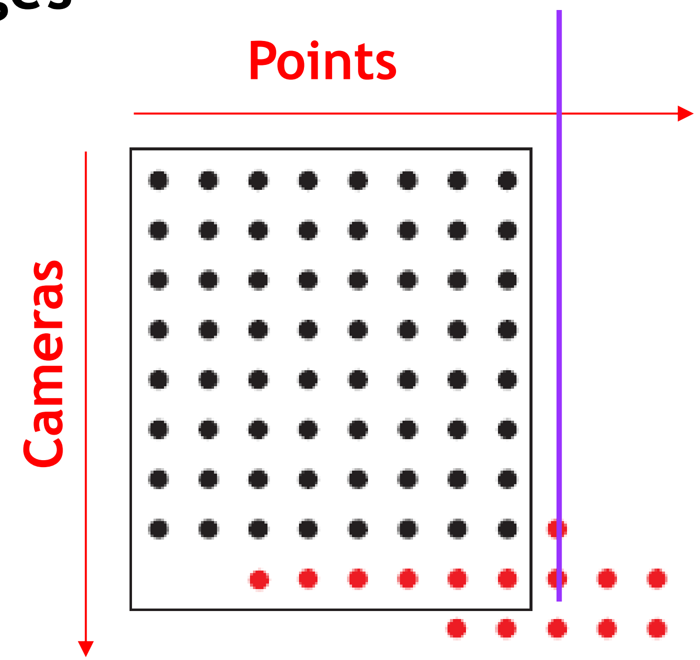
- Initialize motion from two images using fundamental matrix
- Initialize structure
- For each additional view:
  - Determine projection matrix of new camera using all the known 3D points that are visible in its image - *calibration*



Slide credit: Svetlana Lazebnik

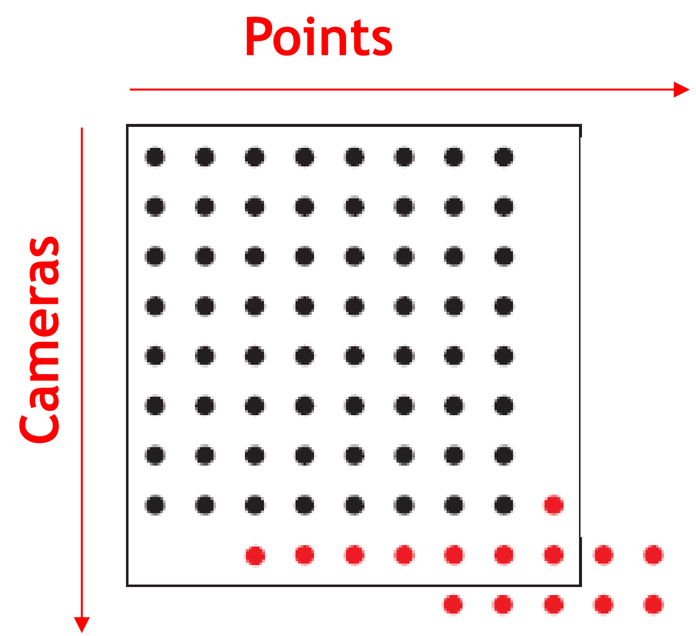
Sequential Structure from Motion

- Initialize motion from two images using fundamental matrix
- Initialize structure
- 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
- 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*

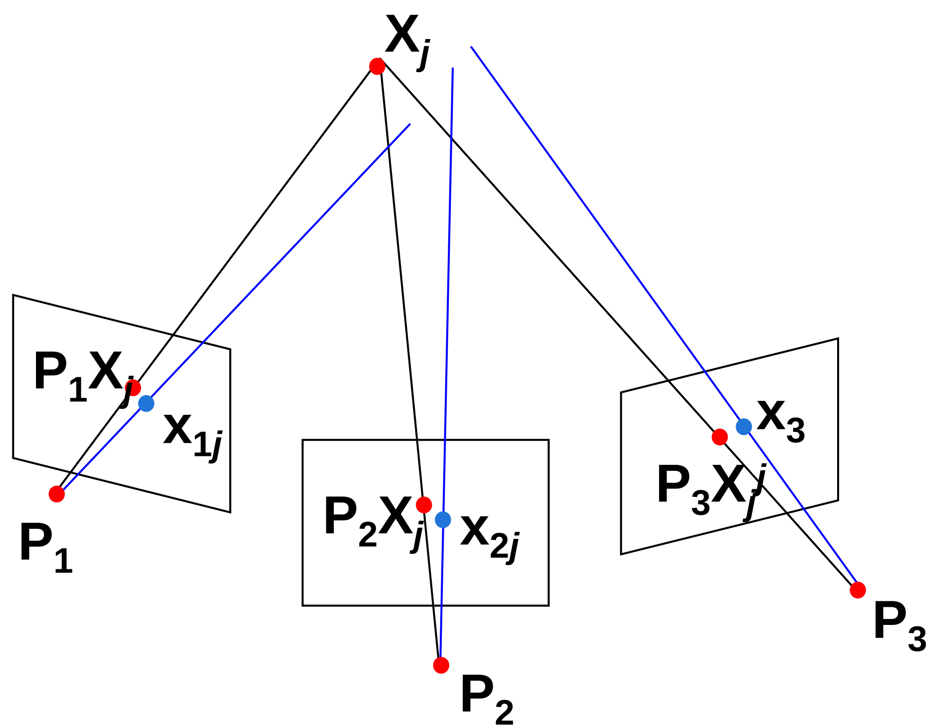


Slide credit: Svetlana Lazebnik

Bundle Adjustment

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

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



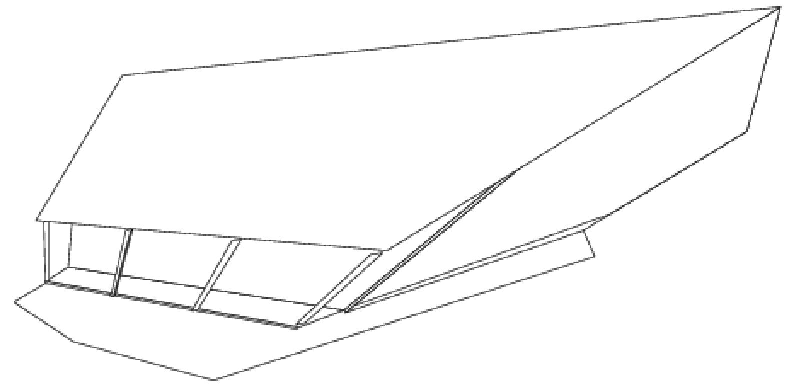
Bundle Adjustment

- Seeks the Maximum Likelihood (ML) solution assuming the measurement noise is Gaussian.
- It involves adjusting the bundle of rays between each camera center and the set of 3D points.
- Bundle adjustment should generally be used as the final step of any multi-view reconstruction algorithm.
  - Considerably improves the results.
  - Allows assignment of individual covariances to each measurement.
- However…
  - It needs a good initialization.
  - It can become an extremely large minimization problem.
- Very efficient algorithms available.

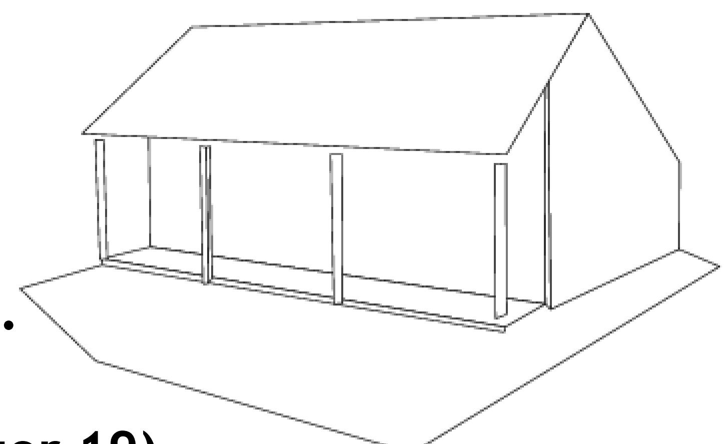
RWTH AACHEN UNIVERSITY

Projective Ambiguity

- If we don't know anything about the camera or the scene, the best we can get with this is a reconstruction up to a projective ambiguity Q.
  - This can already be useful.
  - E.g. we can answer questions like "at what point does a line intersect a plane"?



- If we want to convert this to a "true" reconstruction, we need a *Euclidean upgrade*.
  - Need to put in additional knowledge about the camera (calibration) or about the scene (e.g. from markers).
  - Several methods available (see F&P Chapter 13.5 or H&Z Chapter 19)

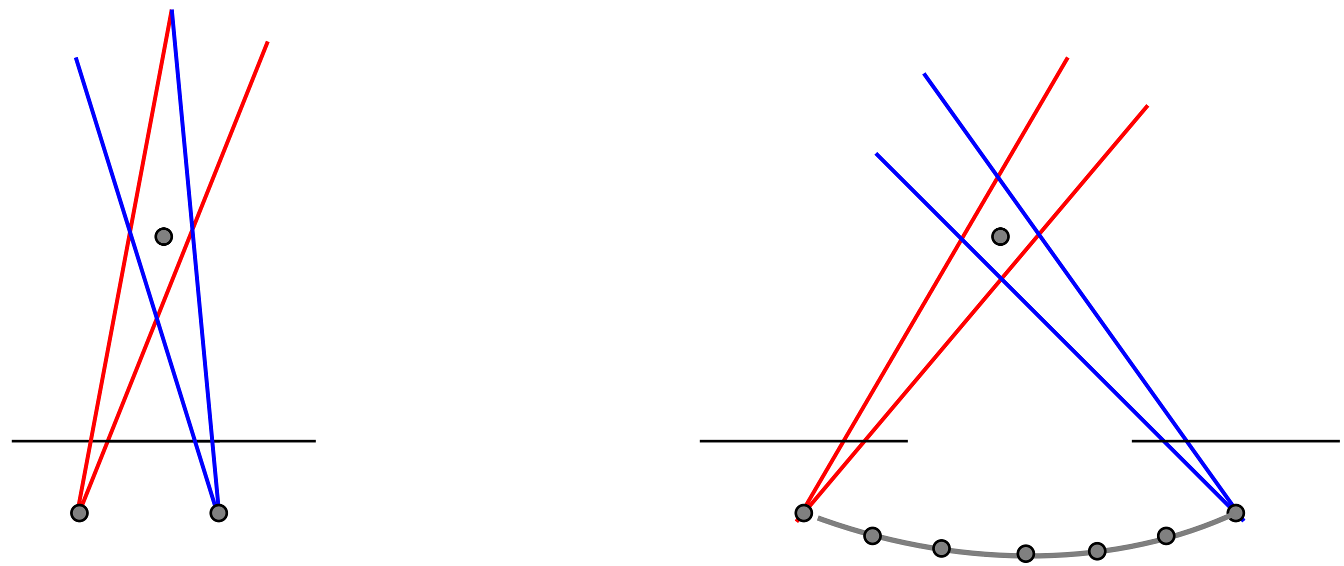


Images from Hartley & Zisserman

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 Q such that all camera matrices are in the form $P_i = K\,[R_i \mid t_i]$.
- Can use constraints on the form of the calibration matrix: square pixels, zero skew, fixed focal length, etc.

Practical Considerations (1)



Small Baseline

Large Baseline

1. **Role of the baseline**
   - Small baseline: large depth error
   - Large baseline: difficult search problem

- **Solution**
  - Track features between frames until baseline is sufficient.

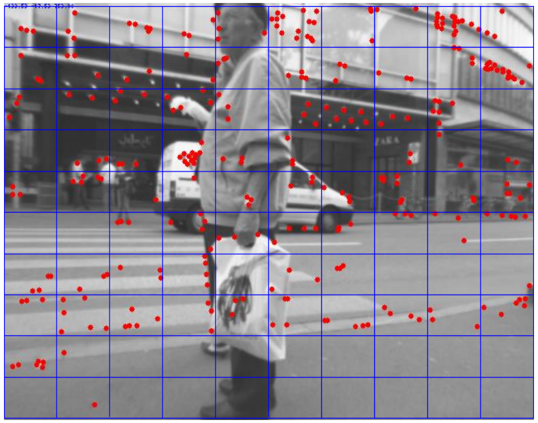
Practical Considerations (2)

2. There will still be many outliers
   - Incorrect feature matches
   - Moving objects

⇒ Apply RANSAC to get robust estimates based on the inlier points.

3. Estimation quality depends on the point configuration
   - Points that are close together in the image produce less stable solutions.

⇒ Subdivide image into a grid and try to extract about the same number of features per grid cell.

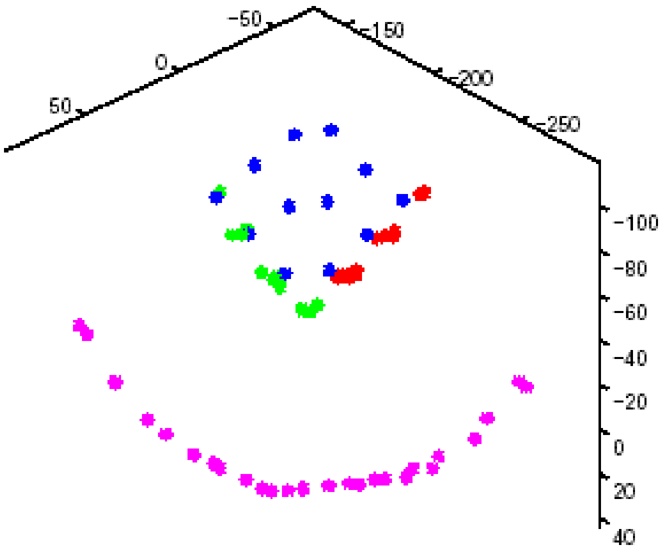


General Guidelines

- Use calibrated cameras wherever possible.
  - It makes life so much easier, especially for SfM.
- SfM with 2 cameras is *far* more robust than with a single camera.
  - Triangulate feature points in 3D using stereo.
  - Perform 2D-3D matching to recover the motion.
  - More robust to loss of scale (main problem of 1-camera SfM).
- Any constraint on the setup can be useful
  - E.g. square pixels, zero skew, fixed focal length in each camera
  - E.g. fixed baseline in stereo SfM setup
  - E.g. constrained camera motion on a ground plane
  - Making best use of those constraints may require adapting the algorithms (some known results are described in H&Z).

Structure-from-Motion: Limitations

- Very difficult to reliably estimate metric SfM unless
  - Large (x or y) motion *or*
  - Large field-of-view and depth variation
- Camera calibration important for Euclidean reconstruction
- Need good feature tracker



Topics of This Lecture

- Structure from Motion (SfM)
  - Motivation
  - Ambiguity
- Affine SfM
  - Affine cameras
  - Affine factorization
  - Euclidean upgrade
  - Dealing with missing data
- Projective SfM
  - Two-camera case
  - Projective factorization
  - Bundle adjustment
  - Practical considerations
- Applications

Commercial Software Packages

- boujou (http://www.2d3.com/)
- PFTrack (http://www.thepixelfarm.co.uk/)
- MatchMover (http://www.realviz.com/)
- SynthEyes (http://www.ssontech.com/)
- Icarus (http://aig.cs.man.ac.uk/research/reveal/icarus/)
- Voodoo Camera Tracker (http://www.digilab.uni-hannover.de/)

Applications: Matchmoving

- Putting virtual objects into real-world videos

[Original sequence](#) [Tracked features](#)

[SfM results](#) [Final video](#)

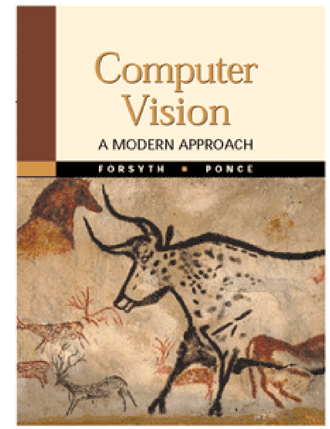
Videos from Stefan Hafeneger

Applications: Large-Scale SfM from Flickr

S. Agarwal, N. Snavely, I. Simon, S.M. Seitz, R. Szeliski, Building Rome in a Day, ICCV'09, 2009. (Video from http://grail.cs.washington.edu/rome/)

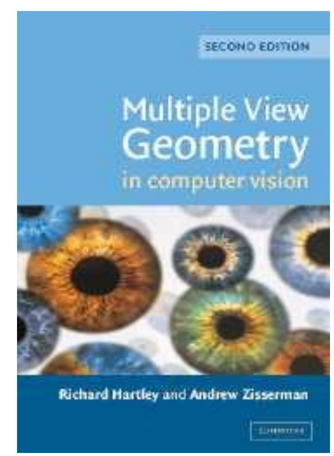
References and Further Reading

- A (relatively short) treatment of affine and projective SfM and the basic ideas and algorithms can be found in Chapters 12 and 13 of



D. Forsyth, J. Ponce,
Computer Vision – A Modern Approach.
Prentice Hall, 2003

- More detailed information (if you really want to implement this) and better explanations can be found in Chapters 10, 18 (factorization) and 19 (self-calibration) of



R. Hartley, A. Zisserman
Multiple View Geometry in Computer Vision
2nd Ed., Cambridge Univ. Press, 2004