Computer Vision

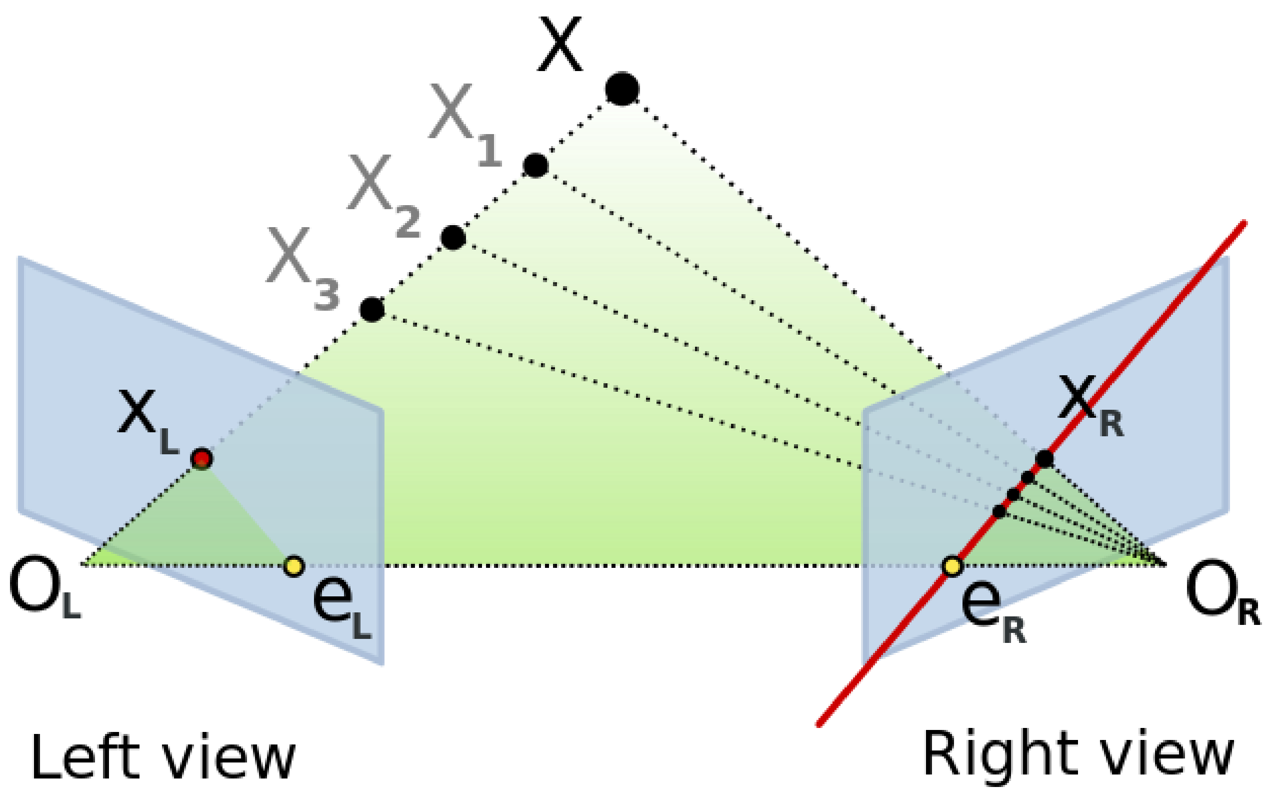
Structure From Motion

Dr. Mrinmoy Ghorai

Indian Institute of Information Technology
Sri City, Chittoor

Recap: Epipoles

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



Recap: Fundamental Matrix

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

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

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

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

Recap: Automatic Estimation of F

Assume we have matched points x ⇔ x' with outliers

8-Point Algorithm for Recovering F

- Correspondence Relation

$$\mathbf{x}'^T \mathbf{F} \mathbf{x} = 0$$

1. Normalize image coordinates

$$\tilde{\mathbf{x}} = \mathbf{T}\mathbf{x} \quad \tilde{\mathbf{x}}' = \mathbf{T}'\mathbf{x}'$$

2. RANSAC with 8 points
    - Randomly sample 8 points
    - Compute F via least squares
    - Enforce $\det(\tilde{\mathbf{F}}) = 0$ by SVD
    - Repeat and choose F with most inliers
3. De-normalize: $\mathbf{F} = \mathbf{T}'^T \tilde{\mathbf{F}} \mathbf{T}$

Perspective and 3D Geometry

- **Camera models and Projective geometry**
  - What's the mapping between image and world coordiantes?
- **Projection Matrix and Camera calibration**
  - What's the projection matrix between scene and image coordiantes?
  - How to calibrate the projection matrix?
- **Single view metrology and Camera properties**
  - How can we measure the size of 3D objects in an image?
  - What are the important camera properties?
- **Photo stitching**
  - What's the mapping from two images taken without camera translation?
- **Epipolar Geometry and Stereo Vision**
  - What's the mapping from two images taken with camera translation?
- **Structure from motion**
  - How can we recover 3D points from multiple images?

This class: Structure from Motion

- Projective structure from motion
- Affine structure from motion
- Multi-view Stereo

Structure:

3D Point Cloud of the Scene

Structure:

3D Point Cloud of the Scene

Motion:

Camera Location and Orientation

Structure:

3D Point Cloud of the Scene

Motion:

Camera Location and Orientation

Structure from Motion (SfM)

Get the Point Cloud from Moving Cameras

SfM Applications – 3D Modeling

http://www.3dcadbrowser.com/download.aspx?3dmodel=40454

SfM Applications – Surveying cultural heritage structure analysis

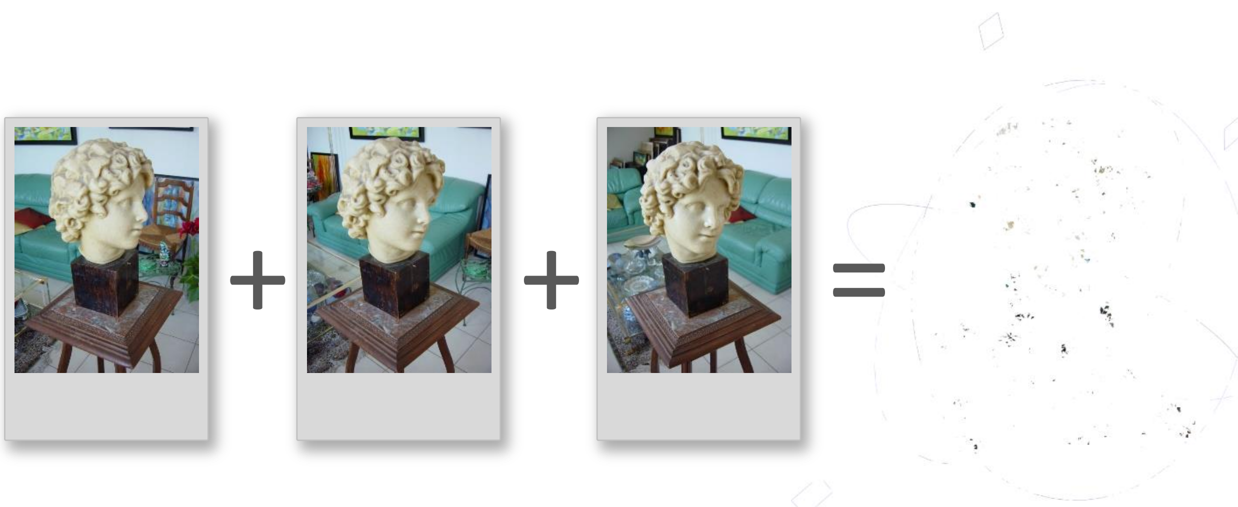
Guidi et al. High-accuracy 3D modeling of cultural heritage, 2004

SfM Applications –
Robot navigation and mapmaking

https://www.youtube.com/watch?v=1HhOmF22oYA

Steps

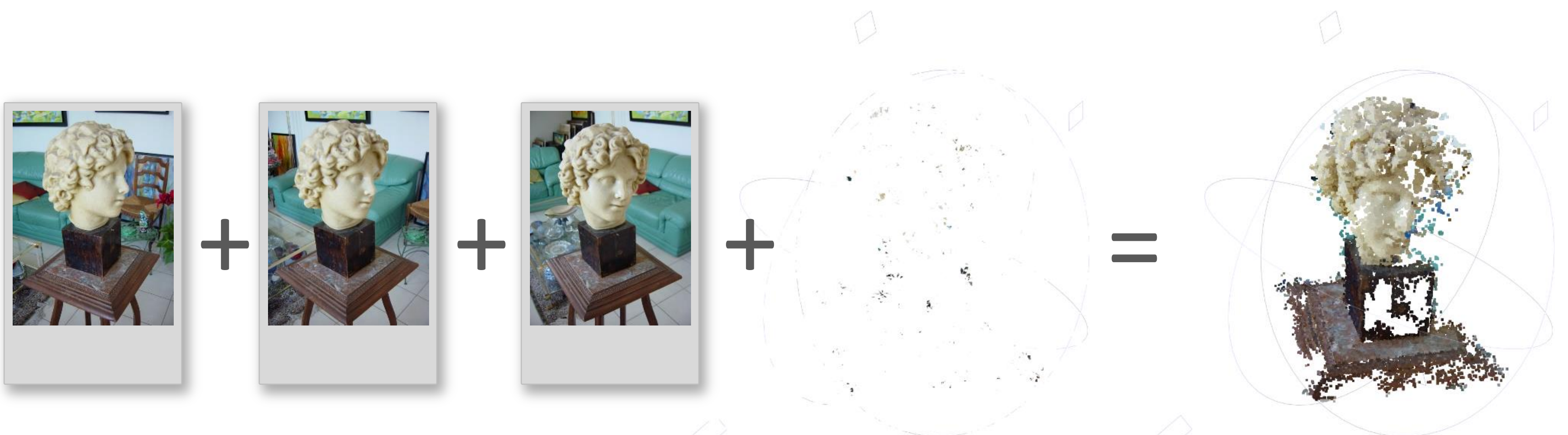
	Images → Points:	Structure from Motion
	Points → More points:	Multiple View Stereo
	Points → Meshes:	Model Fitting
+	Meshes → Models:	Texture Mapping
=	Images → Models:	Image-based Modeling



Slide credit: J. Xiao

Steps

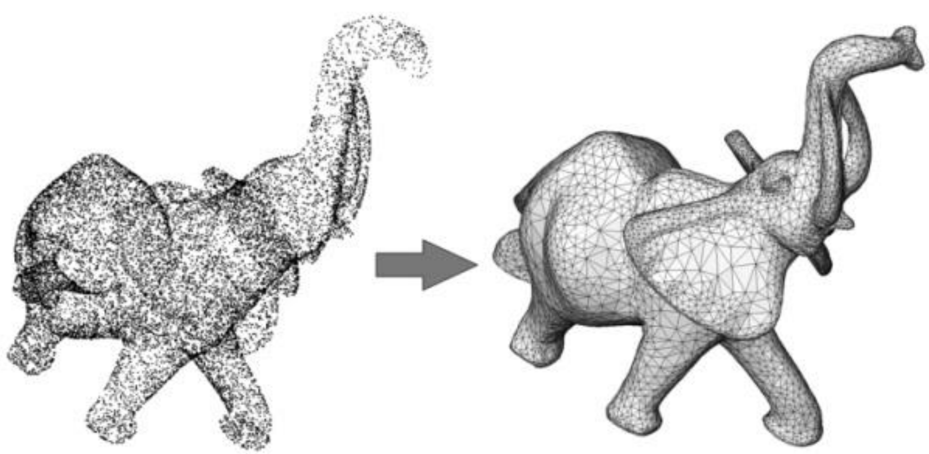
	Images → Points:	Structure from Motion
	Points → More points:	Multiple View Stereo
	Points → Meshes:	Model Fitting
+	Meshes → Models:	Texture Mapping
=	Images → Models:	Image-based Modeling



Slide credit: J. Xiao

Steps

	Images → Points:	Structure from Motion
	Points → More points:	Multiple View Stereo
	Points → Meshes:	Model Fitting
+	Meshes → Models:	Texture Mapping
=	Images → Models:	Image-based Modeling



Slide credit: J. Xiao

Steps

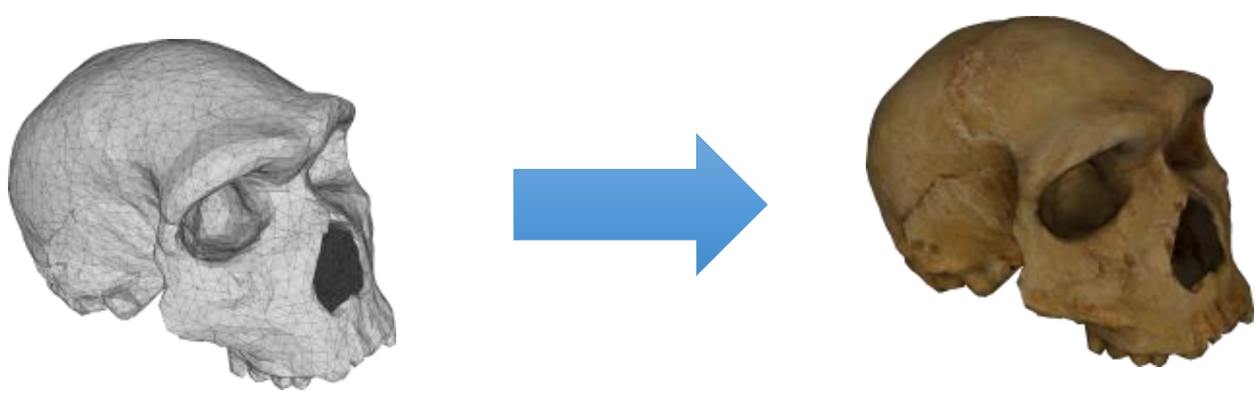
Images → Points: Structure from Motion

Points → More points: Multiple View Stereo

Points → Meshes: Model Fitting

\+ Meshes → Models: Texture Mapping

\= Images → Models: Image-based Modeling



Slide credit: J. Xiao

Steps

	Images → Points:	Structure from Motion
	Points → More points:	Multiple View Stereo
	Points → Meshes:	Model Fitting
+	Meshes → Models:	Texture Mapping
=	Images → Models:	Image-based Modeling

[Image Source: Neill D.F. Campbell University of Bath]

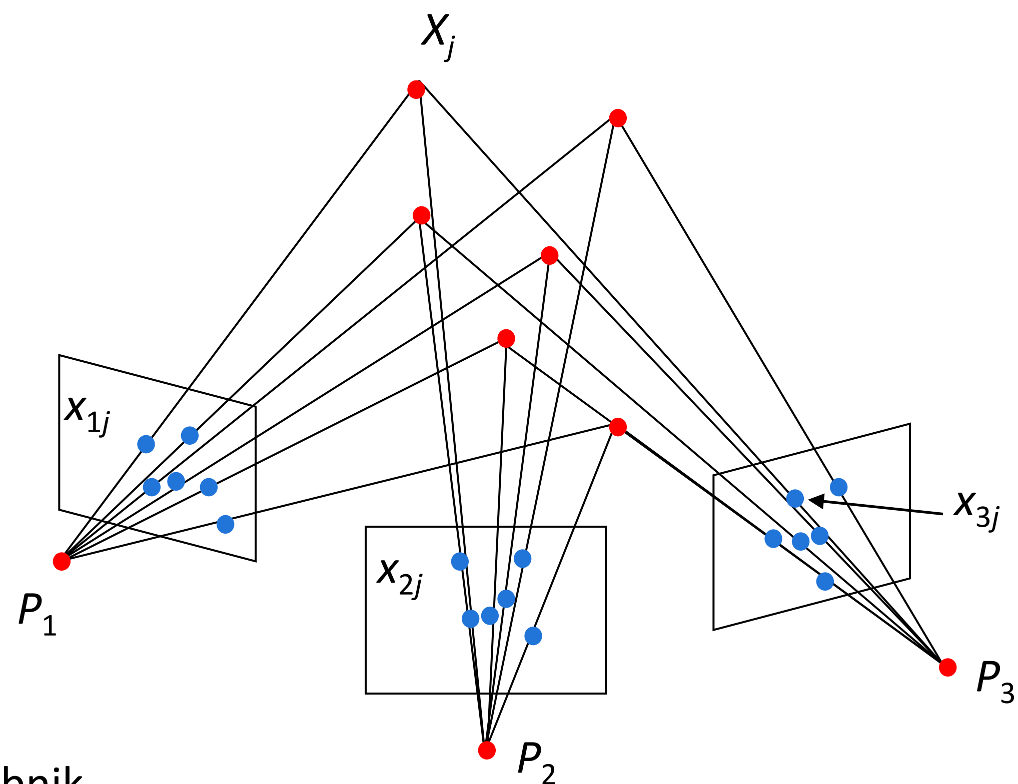
Slide credit: J. Xiao

Projective structure from motion

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

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

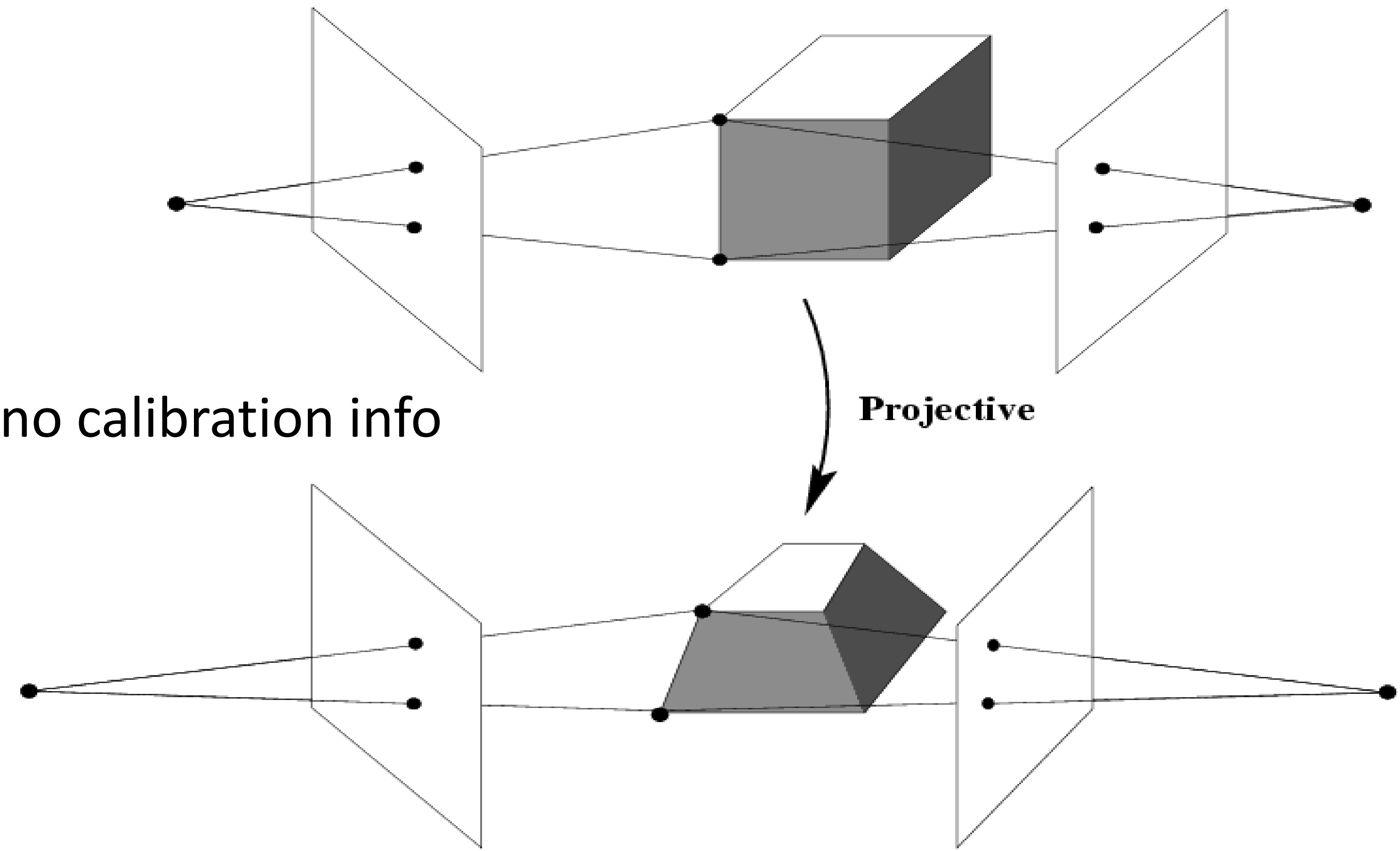
- Problem: estimate *m* projection matrices $\mathbf{P}_i$ and *n* 3D points $\mathbf{X}_j$ from the *mn* corresponding 2D points $\mathbf{x}_{ij}$



Slides from Lana Lazebnik

Projective ambiguity

With no calibration info



Projective structure from motion

- Given: *m* images of *n* fixed 3D points
  - $\mathbf{x}_{ij} = \mathbf{P}_i \mathbf{X}_j, \quad i = 1, \dots, m, \quad j = 1, \dots, n$
- Problem:
  - Estimate unknown *m* projection matrices $\mathbf{P}_i$ and *n* 3D points $\mathbf{X}_j$ from the known *mn* corresponding points $\mathbf{x}_{ij}$

Projective structure from motion

- Given: *m* images of *n* fixed 3D points
  - $\mathbf{x}_{ij} = \mathbf{P}_i \mathbf{X}_j, \quad i = 1, \ldots, m, \quad j = 1, \ldots, n$
- Problem:
  - Estimate unknown *m* projection matrices $\mathbf{P}_i$ and *n* 3D points $\mathbf{X}_j$ from the known *mn* corresponding points $\mathbf{x}_{ij}$
- With no calibration info, cameras and points can only be recovered up to a 4x4 projective transformation **Q**:
  - $\mathbf{X} \rightarrow \mathbf{QX}, \mathbf{P} \rightarrow \mathbf{PQ}^{-1}$

Projective structure from motion

- Given: *m* images of *n* fixed 3D points
  - $\mathbf{x}_{ij} = \mathbf{P}_i \mathbf{X}_j, \quad i = 1, \dots, m, \quad j = 1, \dots, n$
- Problem:
  - Estimate unknown *m* projection matrices $\mathbf{P}_i$ and *n* 3D points $\mathbf{X}_j$ from the known *mn* corresponding points $\mathbf{x}_{ij}$
- 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

$$2mn \geq \underbrace{\boxed{11m}}_{\text{DoF in } \mathbf{P}_i} + \underbrace{\boxed{3n}}_{\text{DoF in } \mathbf{X}_j} - \underbrace{\boxed{15}}_{\text{Up to 4x4 projective tform Q}}$$

Projective structure from motion

- Given: *m* images of *n* fixed 3D points
  - $\mathbf{x}_{ij} = \mathbf{P}_i \mathbf{X}_j, \quad i = 1, \ldots, m, \quad j = 1, \ldots, n$
- Problem:
  - Estimate unknown *m* projection matrices $\mathbf{P}_i$ and *n* 3D points $\mathbf{X}_j$ from the known *mn* corresponding points $\mathbf{x}_{ij}$
- 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

$$2mn >= \underbrace{\boxed{11m}}_{\text{DoF in } \mathbf{P}_i} + \underbrace{\boxed{3n}}_{\text{DoF in } \mathbf{X}_j} - \underbrace{\boxed{15}}_{\text{Up to 4x4 projective tform Q}}$$

- For two cameras, at least 7 points are needed

Sequential structure from motion

Sequential structure from motion

- Initialize motion (calibration) from two images using fundamental matrix

Sequential structure from motion

- Initialize motion (calibration) from two images using fundamental matrix

- Initialize structure by triangulation

Sequential structure from motion

- Initialize motion (calibration) from two images using fundamental matrix

- Initialize structure by triangulation

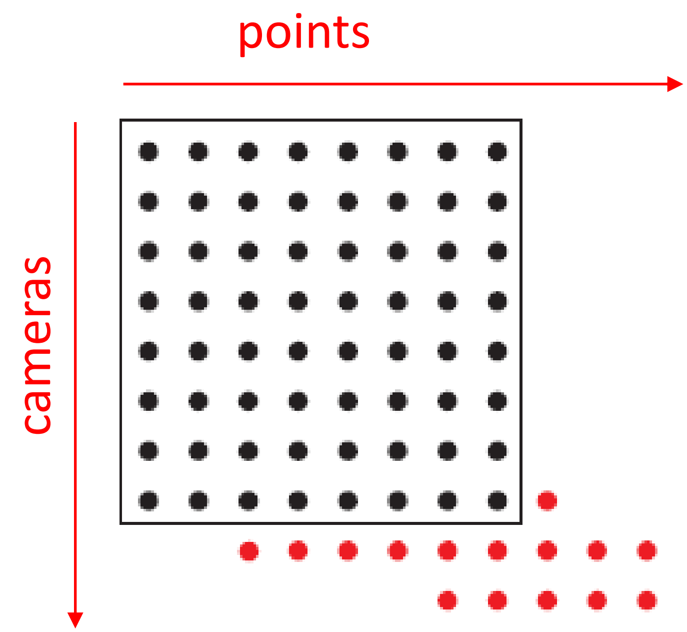
- For each additional view:

Sequential structure from motion

- Initialize motion (calibration) from two images using fundamental matrix

- Initialize structure by triangulation

- For each additional view:

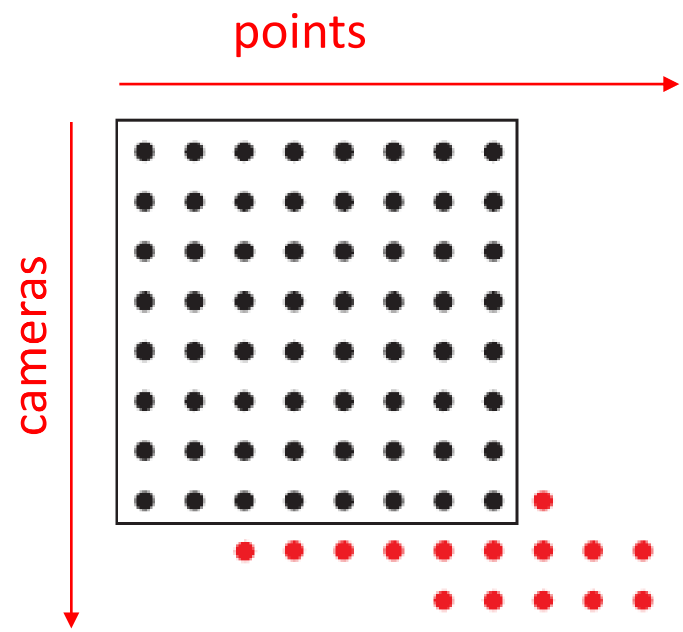


Sequential structure from motion

- Initialize motion (calibration) 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*

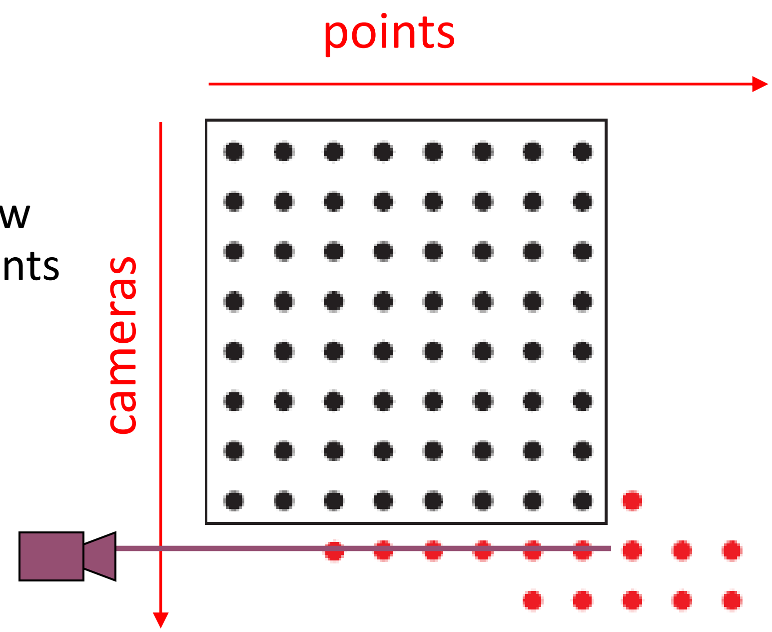


Sequential structure from motion

- Initialize motion (calibration) 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*



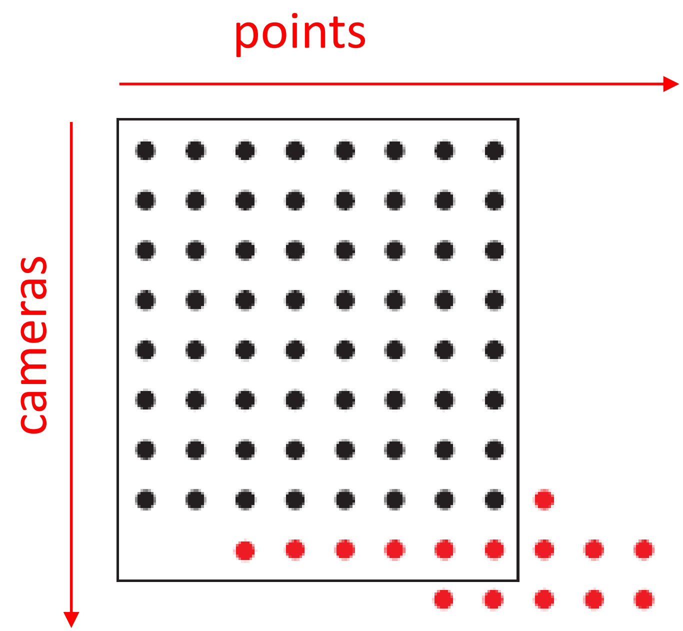
Sequential structure from motion

- Initialize motion (calibration) 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*



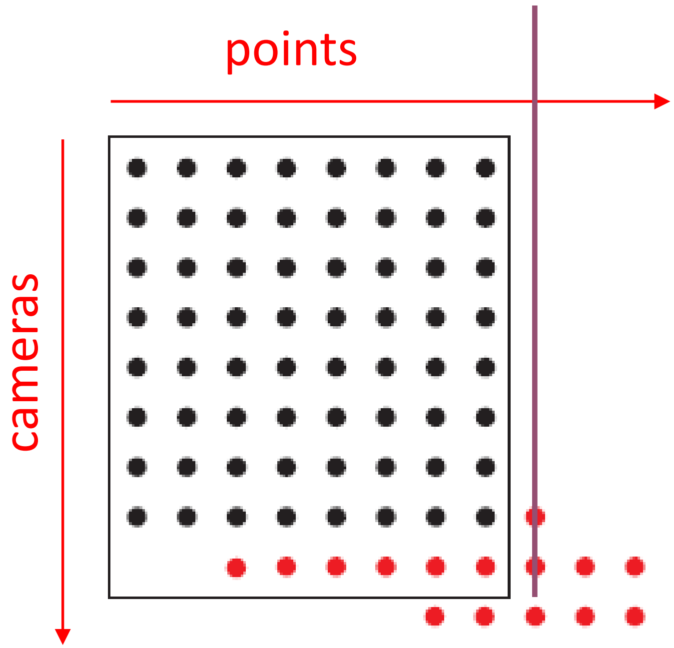
Sequential structure from motion

- Initialize motion (calibration) 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*



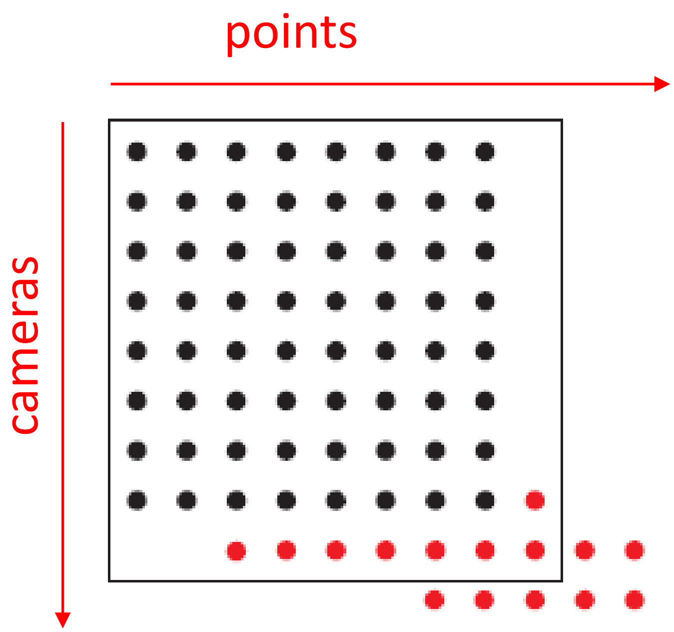
Sequential structure from motion

- Initialize motion (calibration) 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*



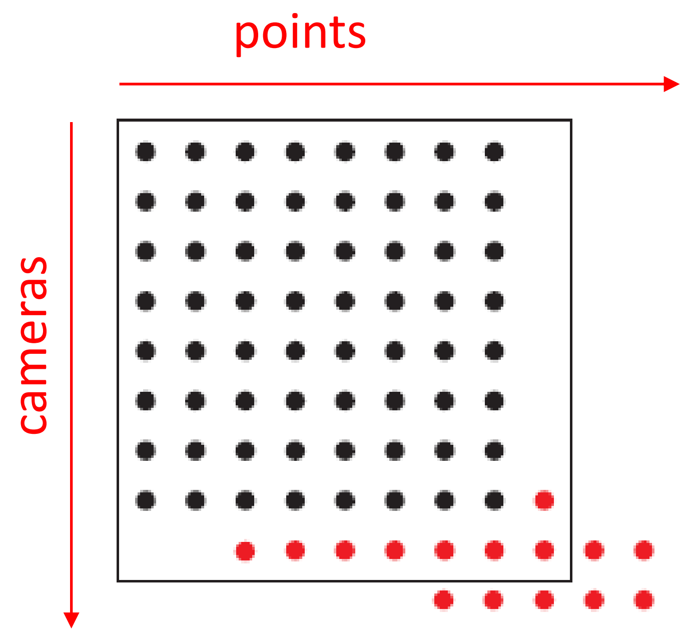
Sequential structure from motion

- Initialize motion (calibration) 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*

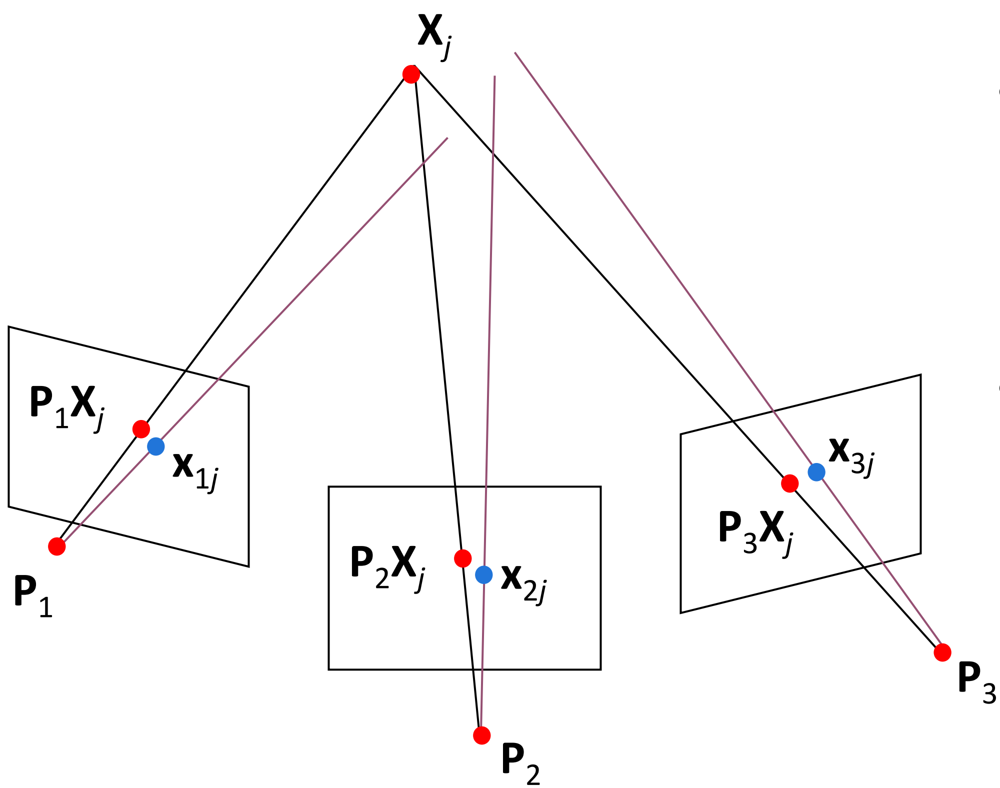


- Refine structure and motion: bundle adjustment

Bundle adjustment

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

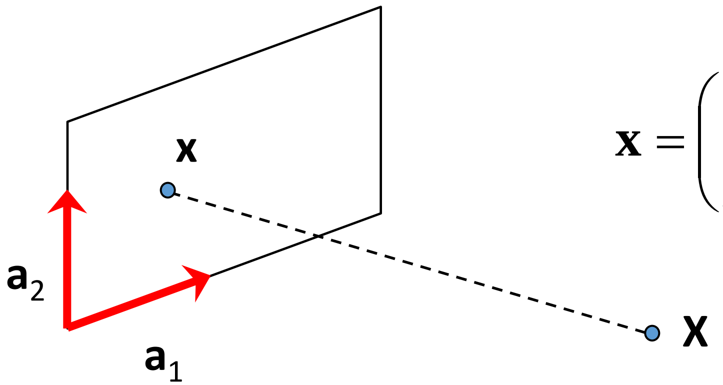
$$E(\mathbf{P},\mathbf{X}) = \sum_{i=1}^{m}\sum_{j=1}^{n} D\left(\mathbf{x}_{ij}, \mathbf{P}_i \mathbf{X}_j\right)^2$$



- Theory:
  The Levenberg–Marquardt algorithm
- Practice:
  The Ceres-Solver from Google

Affine structure from motion

- Affine projection is a linear mapping + translation



$$\mathbf{x} = \begin{pmatrix} x \\ y \end{pmatrix} = \begin{bmatrix} a_{11} & a_{12} & a_{13} \\ a_{21} & a_{22} & a_{23} \end{bmatrix} \begin{pmatrix} X \\ Y \\ Z \end{pmatrix} + \begin{pmatrix} t_x \\ t_y \end{pmatrix} = \mathbf{A}\mathbf{X} + \mathbf{t}$$

Projection of world origin

1. We are given corresponding 2D points (**x**) in several frames
2. We want to estimate the 3D points (**X**) and the affine parameters of each camera (**A**)

Simplify by getting rid of **t**: shift to centroid of points for each camera

$$\mathbf{x}_i = \mathbf{A}_i \mathbf{X} + \mathbf{t}_i \qquad \hat{\mathbf{x}}_{ij} = \mathbf{x}_{ij} - \frac{1}{n}\sum_{k=1}^{n} \mathbf{x}_{ik}$$

Simplify by getting rid of **t**: shift to centroid of points for each camera

$$\mathbf{x}_i = \mathbf{A}_i \mathbf{X} + \mathbf{t}_i \qquad \hat{\mathbf{x}}_{ij} = \mathbf{x}_{ij} - \frac{1}{n}\sum_{k=1}^{n} \mathbf{x}_{ik}$$

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

Simplify by getting rid of **t**: shift to centroid of points for each camera

$$\mathbf{x}_i = \mathbf{A}_i \mathbf{X} + \mathbf{t}_i \qquad\qquad \hat{\mathbf{x}}_{ij} = \mathbf{x}_{ij} - \frac{1}{n}\sum_{k=1}^{n} \mathbf{x}_{ik}$$

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

Simplify by getting rid of **t**: shift to centroid of points for each camera

$$\mathbf{x}_i = \mathbf{A}_i \mathbf{X} + \mathbf{t}_i \qquad \hat{\mathbf{x}}_{ij} = \mathbf{x}_{ij} - \frac{1}{n}\sum_{k=1}^{n} \mathbf{x}_{ik}$$

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

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

2d normalized point (observed)

Linear (affine) mapping

3d normalized point

Suppose we know 3D points and affine camera parameters …

$$
\begin{bmatrix} \mathbf{A}_1 \\ \mathbf{A}_2 \\ \vdots \\ \mathbf{A}_m \end{bmatrix}
\begin{bmatrix} \mathbf{X}_1 & \mathbf{X}_2 & \cdots & \mathbf{X}_n \end{bmatrix}
=
\begin{bmatrix}
\hat{\mathbf{x}}_{11} & \hat{\mathbf{x}}_{12} & \cdots & \hat{\mathbf{x}}_{1n} \\
\hat{\mathbf{x}}_{21} & \hat{\mathbf{x}}_{22} & \cdots & \hat{\mathbf{x}}_{2n} \\
 & & \ddots & \\
\hat{\mathbf{x}}_{m1} & \hat{\mathbf{x}}_{m2} & \cdots & \hat{\mathbf{x}}_{mn}
\end{bmatrix}
$$

3D Points (3xn)

Camera Parameters (2mx3)

2D Image Points (2mxn)

What if we instead observe corresponding 2d image points?

Can we recover the camera parameters and 3d points?

cameras (2 m)

$$\mathbf{D} = \begin{bmatrix} \hat{\mathbf{x}}_{11} & \hat{\mathbf{x}}_{12} & \cdots & \hat{\mathbf{x}}_{1n} \\ \hat{\mathbf{x}}_{21} & \hat{\mathbf{x}}_{22} & \cdots & \hat{\mathbf{x}}_{2n} \\ & & \ddots & \\ \hat{\mathbf{x}}_{m1} & \hat{\mathbf{x}}_{m2} & \cdots & \hat{\mathbf{x}}_{mn} \end{bmatrix} \overset{?}{\Rightarrow} \begin{bmatrix} \mathbf{A}_1 \\ \mathbf{A}_2 \\ \vdots \\ \mathbf{A}_m \end{bmatrix} \begin{bmatrix} \mathbf{X}_1 & \mathbf{X}_2 & \cdots & \mathbf{X}_n \end{bmatrix}$$

points (3 × n)

points (n)

2D Image Points (2mxn)

cameras

(2 m × 3)

What if we instead observe corresponding 2d image points?

Can we recover the camera parameters and 3d points?

cameras ($2m$)

$$\mathbf{D} = \begin{bmatrix} \hat{\mathbf{x}}_{11} & \hat{\mathbf{x}}_{12} & \cdots & \hat{\mathbf{x}}_{1n} \\ \hat{\mathbf{x}}_{21} & \hat{\mathbf{x}}_{22} & \cdots & \hat{\mathbf{x}}_{2n} \\ & & \ddots & \\ \hat{\mathbf{x}}_{m1} & \hat{\mathbf{x}}_{m2} & \cdots & \hat{\mathbf{x}}_{mn} \end{bmatrix} \overset{?}{\Rightarrow} \begin{bmatrix} \mathbf{A}_1 \\ \mathbf{A}_2 \\ \vdots \\ \mathbf{A}_m \end{bmatrix} \begin{bmatrix} \mathbf{X}_1 & \mathbf{X}_2 & \cdots & \mathbf{X}_n \end{bmatrix}$$

points ($3 \times n$)

points (n)

2D Image Points (2mxn)

cameras ($2m \times 3$)

What rank is the matrix of 2D points?

What if we instead observe corresponding 2d image points?

Can we recover the camera parameters and 3d points?

cameras ($2m$)

$$\mathbf{D} = \begin{bmatrix} \hat{\mathbf{x}}_{11} & \hat{\mathbf{x}}_{12} & \cdots & \hat{\mathbf{x}}_{1n} \\ \hat{\mathbf{x}}_{21} & \hat{\mathbf{x}}_{22} & \cdots & \hat{\mathbf{x}}_{2n} \\ & & \ddots & \\ \hat{\mathbf{x}}_{m1} & \hat{\mathbf{x}}_{m2} & \cdots & \hat{\mathbf{x}}_{mn} \end{bmatrix} \overset{?}{\Rightarrow} \begin{bmatrix} \mathbf{A}_1 \\ \mathbf{A}_2 \\ \vdots \\ \mathbf{A}_m \end{bmatrix} \begin{bmatrix} \mathbf{X}_1 & \mathbf{X}_2 & \cdots & \mathbf{X}_n \end{bmatrix}$$

points ($3 \times n$)

cameras ($2m \times 3$)

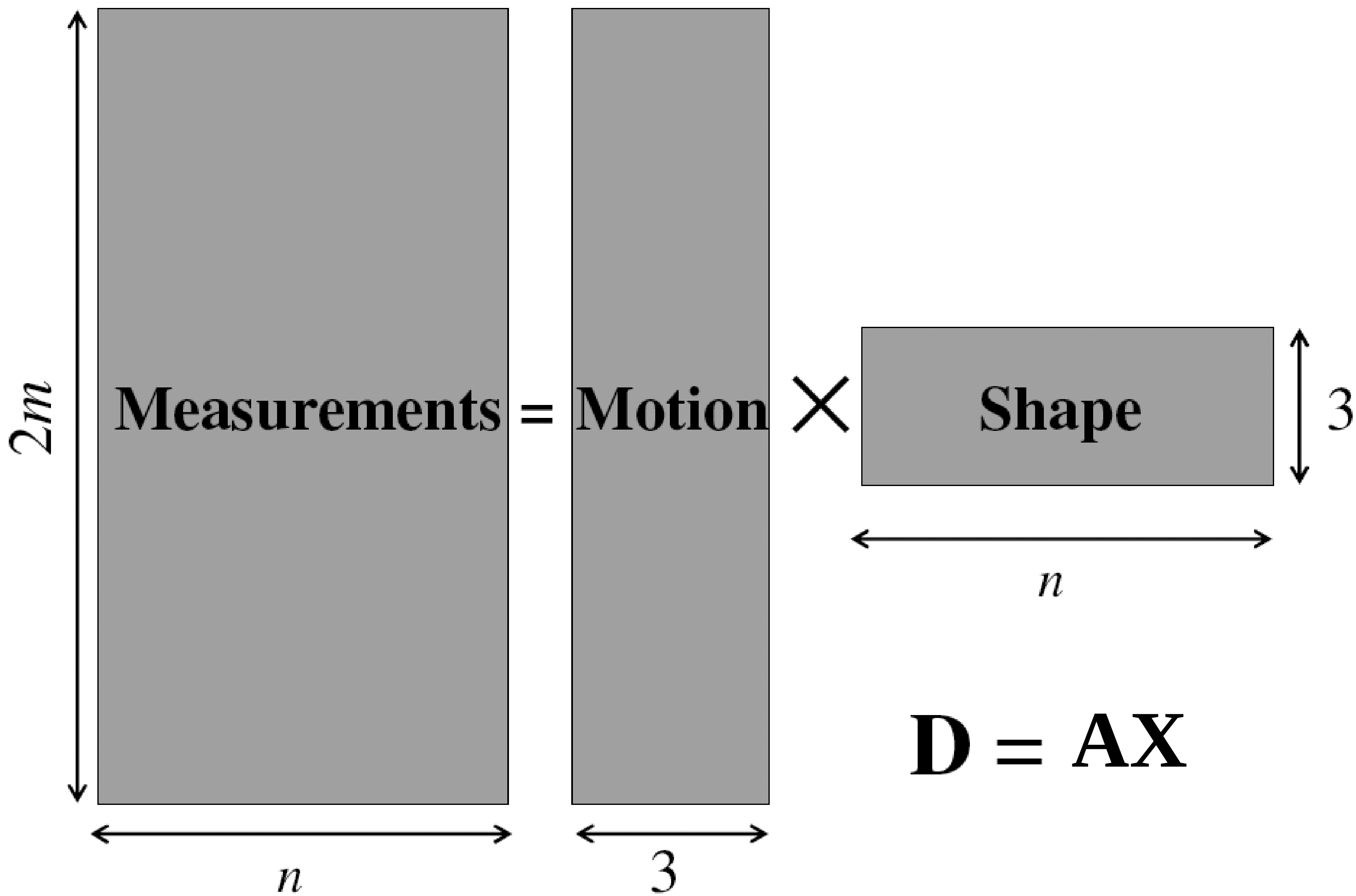
points (n)

2D Image Points (2mxn)

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

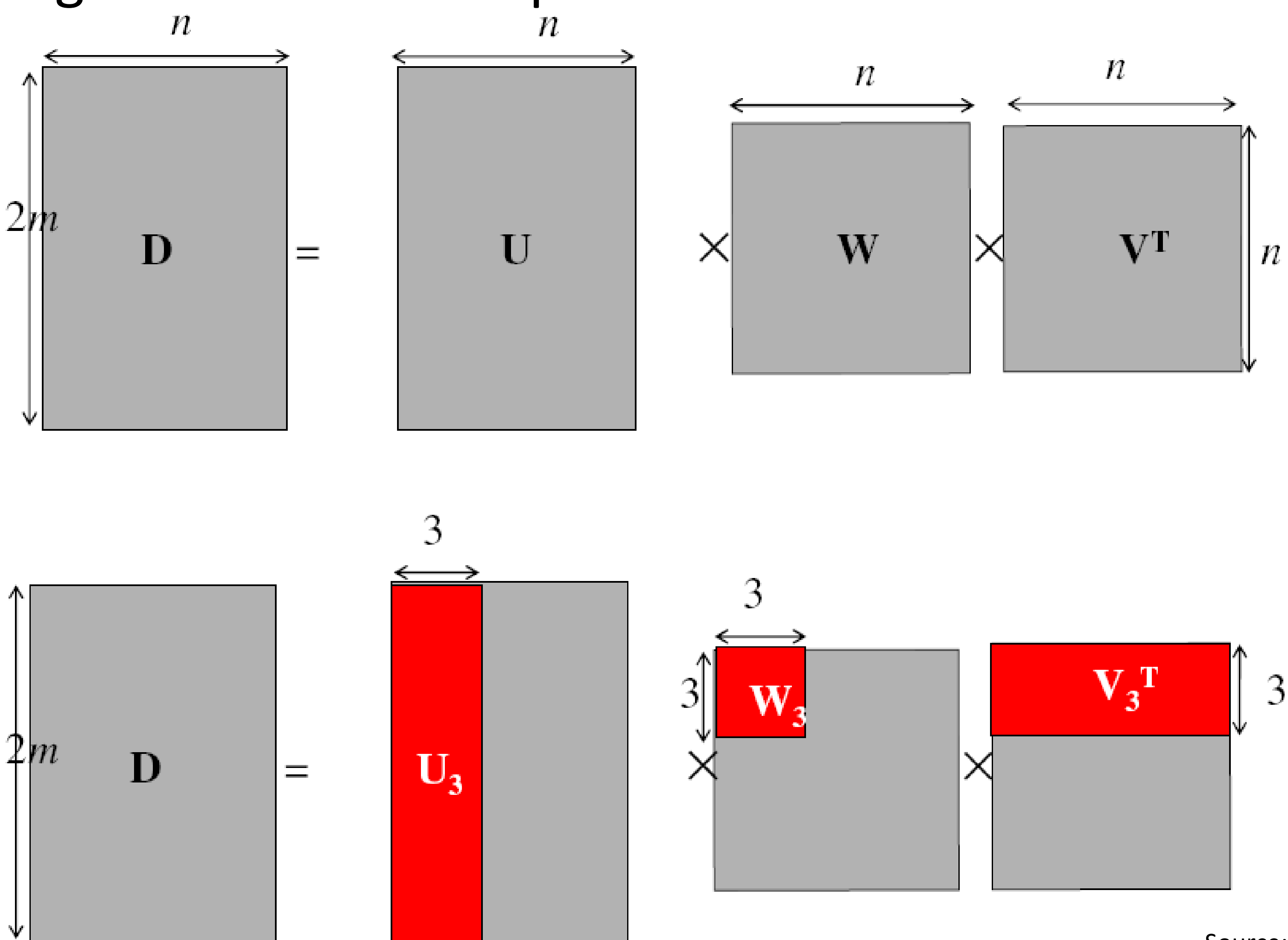


$$\mathbf{D} = \mathbf{AX}$$

Source: M. Hebert

Factorizing the measurement matrix

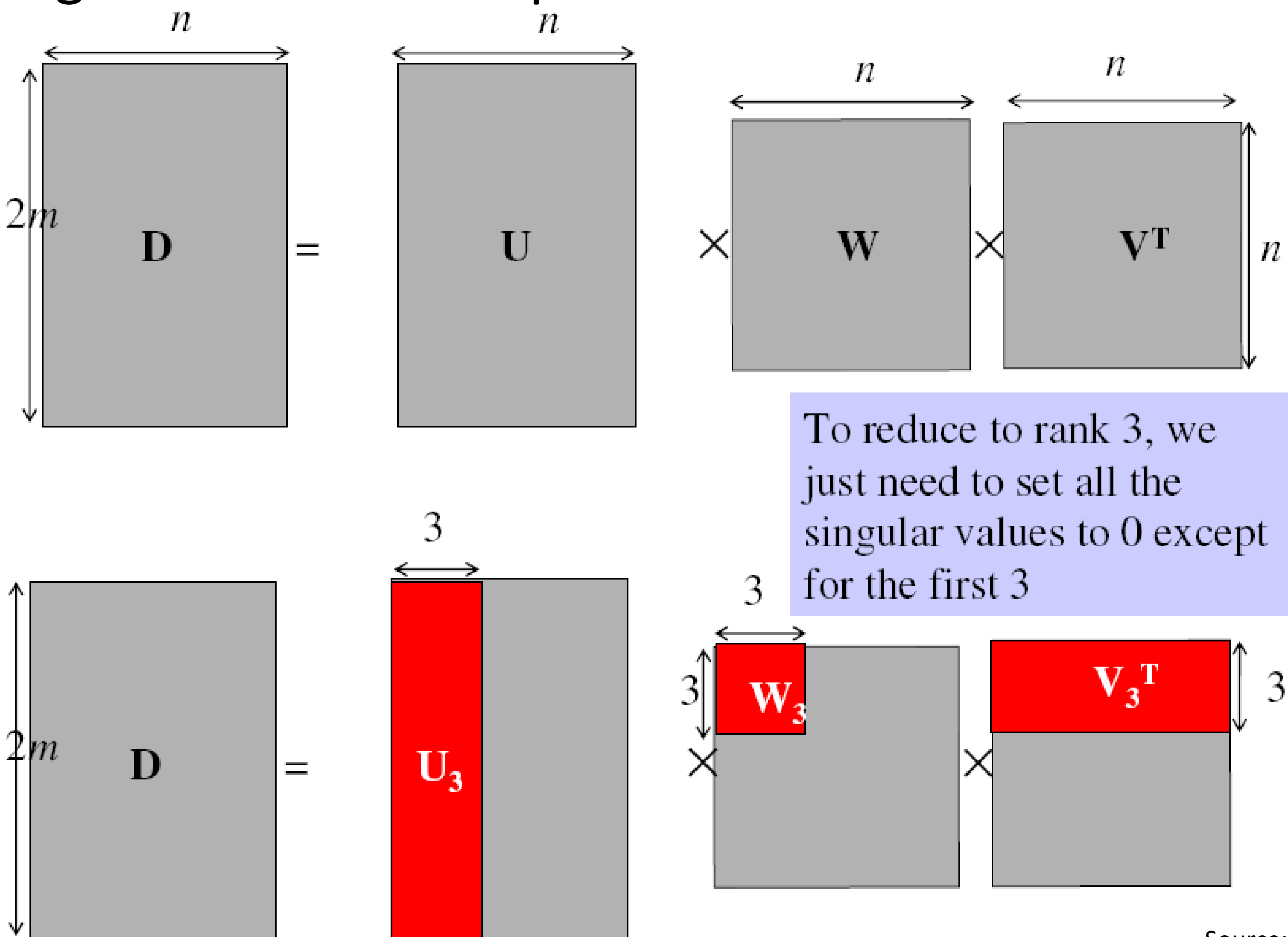
- Singular value decomposition of D:



Source: M. Hebert

Factorizing the measurement matrix

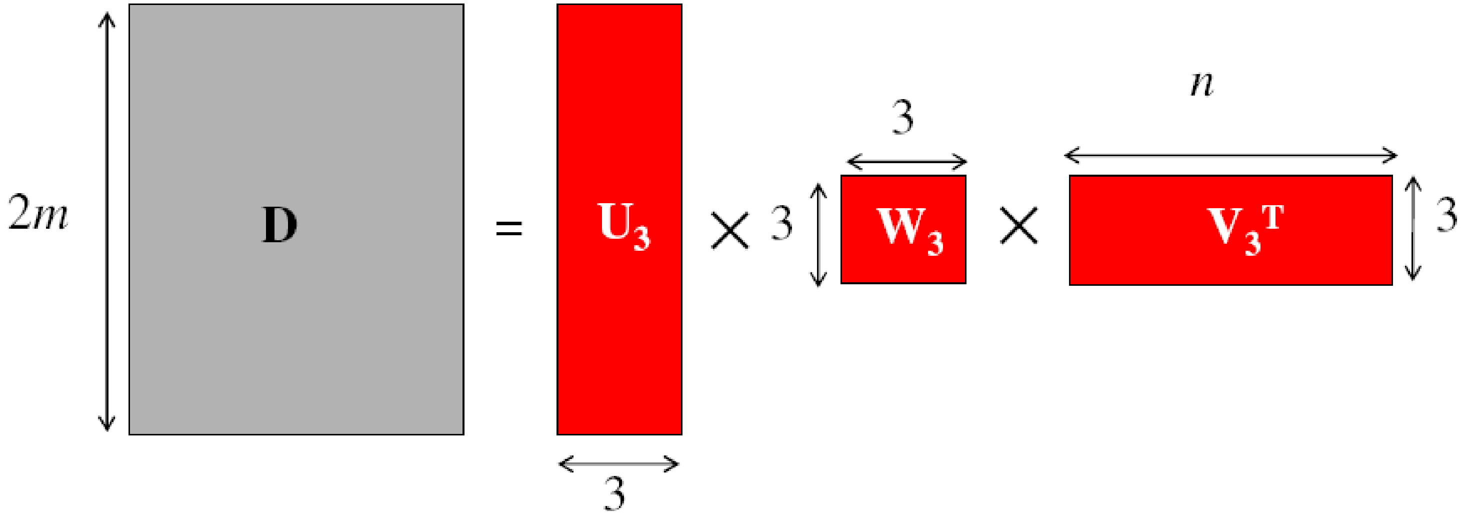
- Singular value decomposition of D:



Source: M. Hebert

Factorizing the measurement matrix

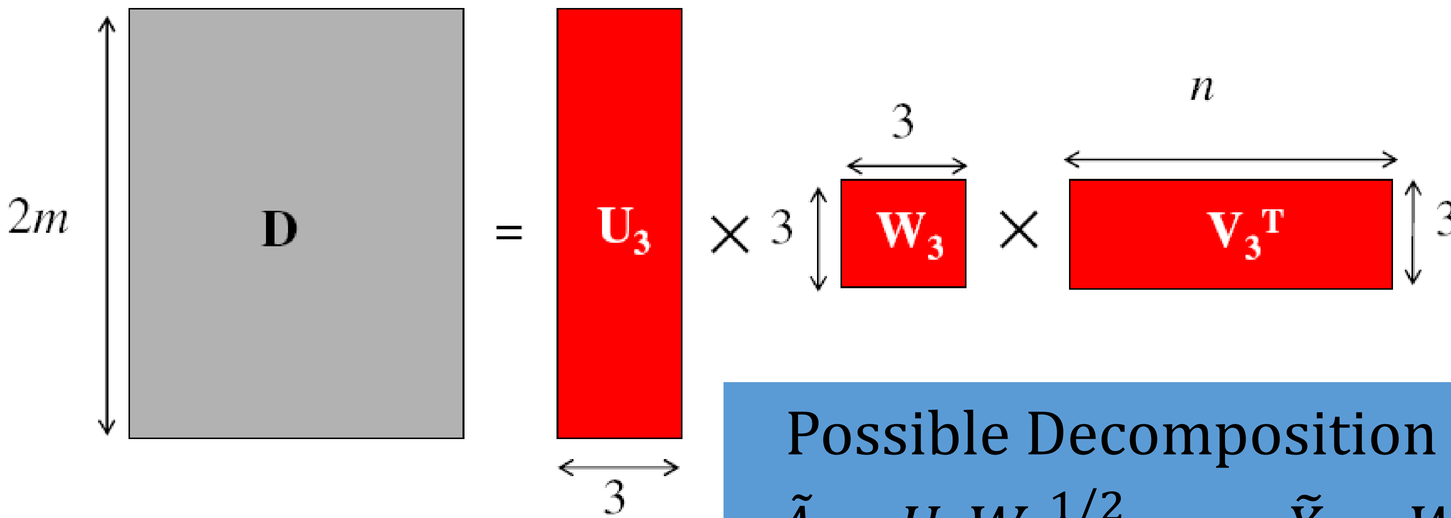
- Obtaining a factorization from SVD:



Source: M. Hebert

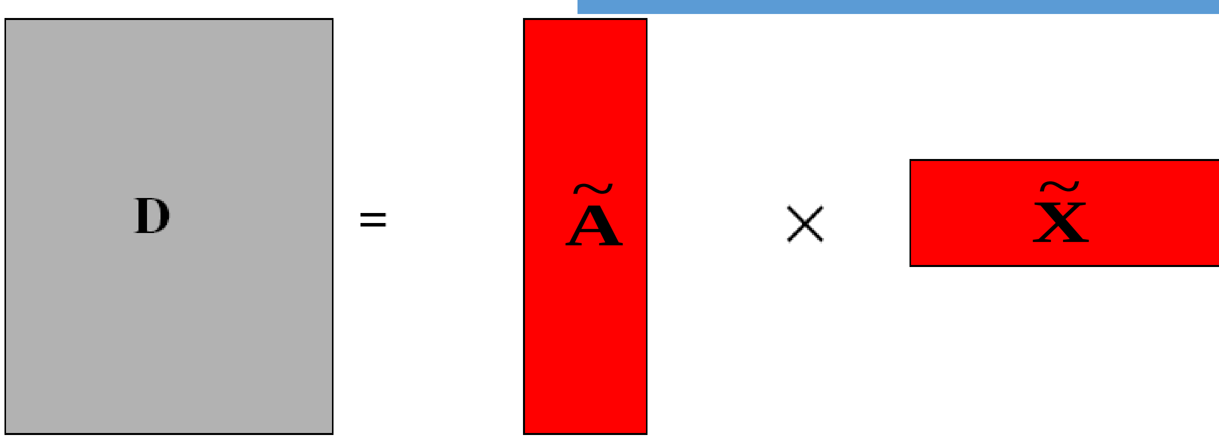
Factorizing the measurement matrix

- Obtaining a factorization from SVD:



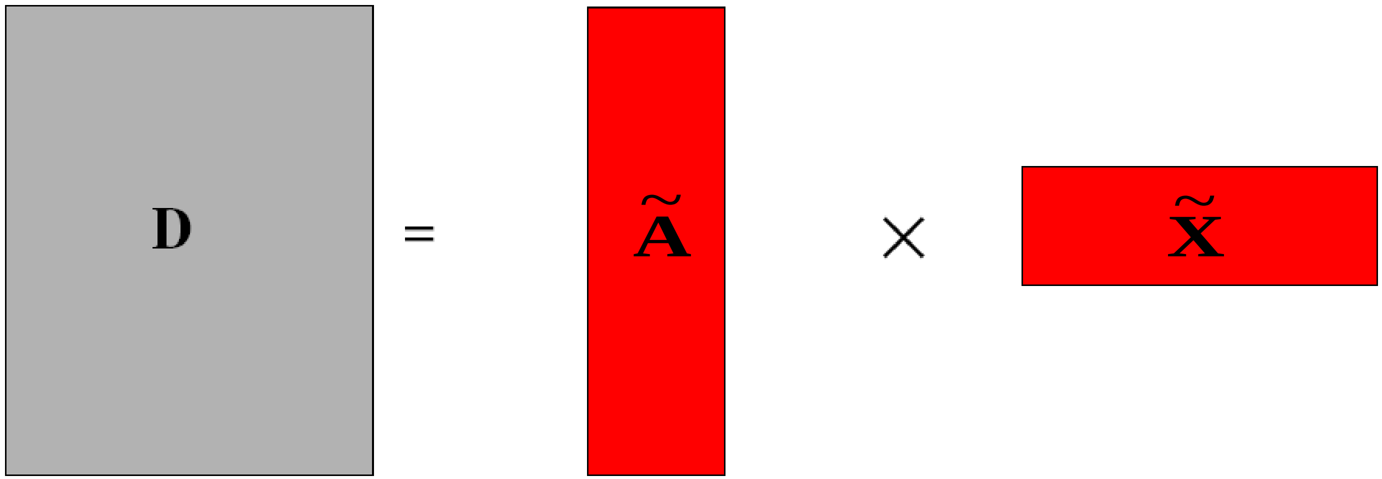
Possible Decomposition

$\tilde{A} = U_3 W_3^{1/2}$ $\qquad$ $\tilde{X} = W_3^{1/2} V_3^T$



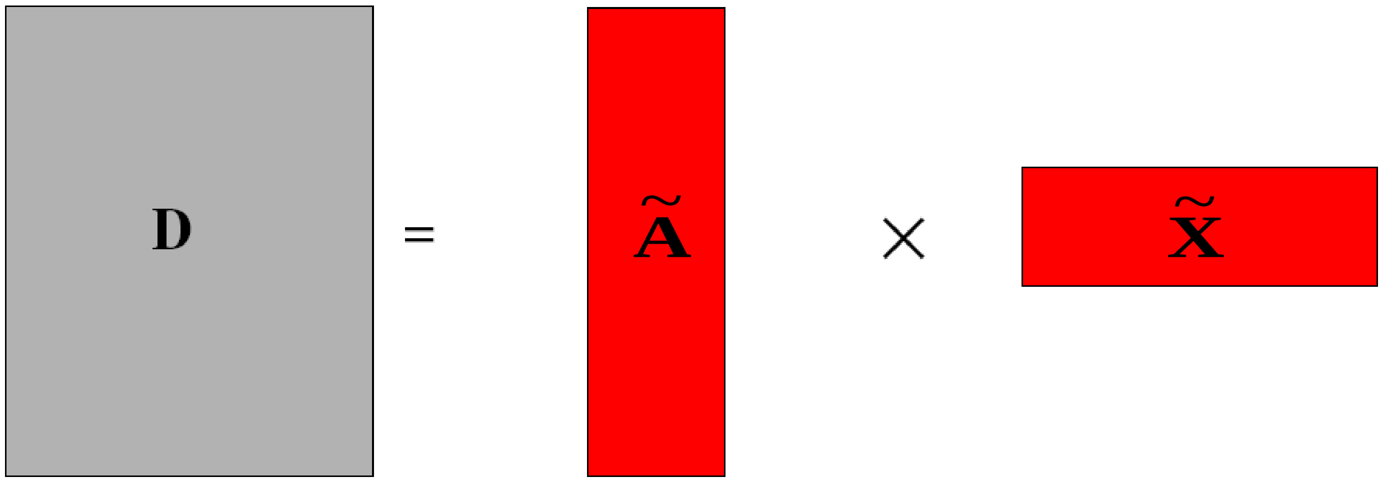
Source: M. Hebert

Affine ambiguity



- The decomposition is not unique.
  We get the same **D** by using any 3×3 matrix **C** and applying the transformations **A → AC, X → $\mathbf{C}^{-1}$X**

Affine ambiguity



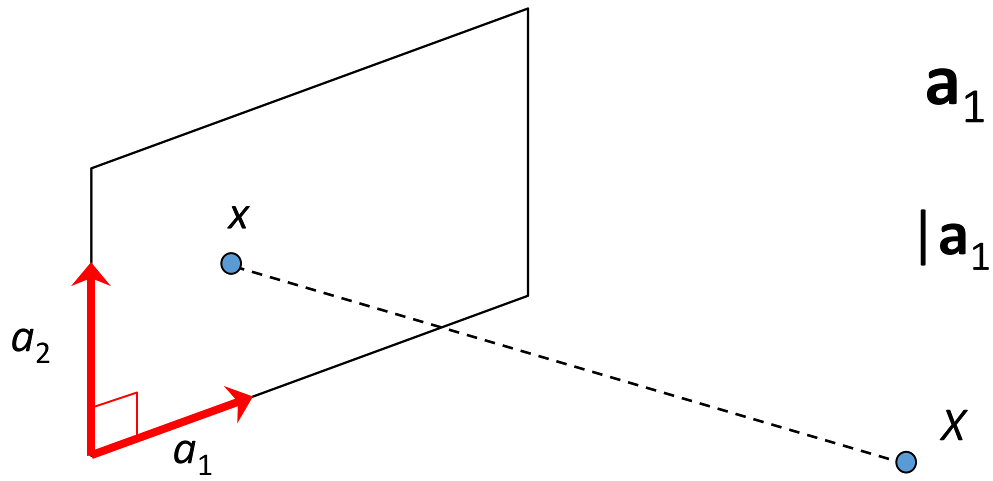
- The decomposition is not unique.
  We get the same **D** by using any 3×3 matrix **C** and applying the transformations **A → AC, X → C^{-1}X**

- Why?
  We have only an affine transformation and we have not enforced any Euclidean constraints
  (e.g., perpendicular image axes)

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

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

Solve for orthographic constraints

Three equations for each image i

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

- Solve for **L = CC$^\mathbf{T}$**

Solve for orthographic constraints

Solve for **L = CC^T**

$$\begin{bmatrix} a & b & c \end{bmatrix} \begin{bmatrix} L_{11} & L_{21} & L_{31} \\ L_{12} & L_{22} & L_{32} \\ L_{13} & L_{23} & L_{33} \end{bmatrix} \begin{bmatrix} d \\ e \\ f \end{bmatrix} = k$$

$$\begin{bmatrix} \qquad\qquad\qquad\qquad\qquad\qquad\qquad\qquad \end{bmatrix} \begin{bmatrix} L_{11} \\ L_{12} \\ L_{13} \\ L_{21} \\ L_{22} \\ L_{23} \\ L_{31} \\ L_{32} \\ L_{33} \end{bmatrix} = \mathrm{k}$$

Solve for orthographic constraints

Solve for **$L = CC^T$**

$$[a \quad b \quad c]\begin{bmatrix} L_{11} & L_{21} & L_{31} \\ L_{12} & L_{22} & L_{32} \\ L_{13} & L_{23} & L_{33} \end{bmatrix}\begin{bmatrix} d \\ e \\ f \end{bmatrix} = k$$

$$[ad \;\; bd \;\; cd \quad ae \;\; be \;\; ce \quad af \;\; bf \;\; cf]\begin{bmatrix} L_{11} \\ L_{12} \\ L_{13} \\ L_{21} \\ L_{22} \\ L_{23} \\ L_{31} \\ L_{32} \\ L_{33} \end{bmatrix} = \mathrm{k}$$

Solve for orthographic constraints

Three equations for each image i

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

- Solve for **L = CC**$^{\mathbf{T}}$

- Recover **C** from **L** by Cholesky decomposition (modified version of Gaussian elimination): **L = CC**$^{\mathbf{T}}$
- Update **A** and **X**: **A = ÃC, X = C**$^{\mathbf{-1}}$**X̃**

Cholesky factorization

every positive definite matrix $A \in \mathbf{R}^{n \times n}$ can be factored as

$$A = R^T R$$

where R is upper triangular with positive diagonal elements

- complexity of computing R is $(1/3)n^3$ flops
- R is called the *Cholesky factor* of A
- can be interpreted as 'square root' of a positive definite matrix

http://www.seas.ucla.edu/~vandenbe/133A/lectures/chol.pdf

Cholesky factorization

$$\begin{bmatrix} A_{11} & A_{1,2:n} \\ A_{2:n,1} & A_{2:n,2:n} \end{bmatrix} = \begin{bmatrix} R_{11} & 0 \\ R_{1,2:n}^T & R_{2:n,2:n}^T \end{bmatrix} \begin{bmatrix} R_{11} & R_{1,2:n} \\ 0 & R_{2:n,2:n} \end{bmatrix}$$

$$= \begin{bmatrix} R_{11}^2 & R_{11}R_{1,2:n} \\ R_{11}R_{1,2:n}^T & R_{1,2:n}^T R_{1,2:n} + R_{2:n,2:n}^T R_{2:n,2:n} \end{bmatrix}$$

1. compute first row of R:

$$R_{11} = \sqrt{A_{11}}, \qquad R_{1,2:n} = \frac{1}{R_{11}} A_{1,2:n}$$

2. compute $2,2$ block $R_{2:n,2:n}$ from

$$A_{2:n,2:n} - R_{1,2:n}^T R_{1,2:n} = R_{2:n,2:n}^T R_{2:n,2:n}$$

   this is a Cholesky factorization of order $n-1$

http://www.seas.ucla.edu/~vandenbe/133A/lectures/chol.pdf

Algorithm summary

- Given: m images and n tracked features $\mathbf{x}_{ij}$
- For each image i, center the feature coordinates
- Construct a $2m \times n$ measurement matrix $\mathbf{D}$:
  - 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 $\mathbf{D}$:
  - Compute SVD: $\mathbf{D} = \mathbf{U}\,\mathbf{W}\,\mathbf{V}^T$
  - Create $\mathbf{U}_3$ by taking the first 3 columns of $\mathbf{U}$
  - Create $\mathbf{V}_3$ by taking the first 3 columns of $\mathbf{V}$
  - Create $\mathbf{W}_3$ by taking the upper left 3×3 block of $\mathbf{W}$
- Create the motion (affine) and shape (3D) matrices:

  $\mathbf{A} = \mathbf{U}_3\mathbf{W}_3^{½}$ and $\mathbf{X} = \mathbf{W}_3^{½}\,\mathbf{V}_3^T$
- Eliminate affine ambiguity
  - Solve $\mathbf{L} = \mathbf{C}\mathbf{C}^T$ using metric constraints
  - Solve $\mathbf{C}$ using Cholesky decomposition
  - Update $\mathbf{A}$ and $\mathbf{X}$: $\mathbf{A} = \mathbf{AC}$, $\mathbf{X} = \mathbf{C}^{-1}\mathbf{X}$

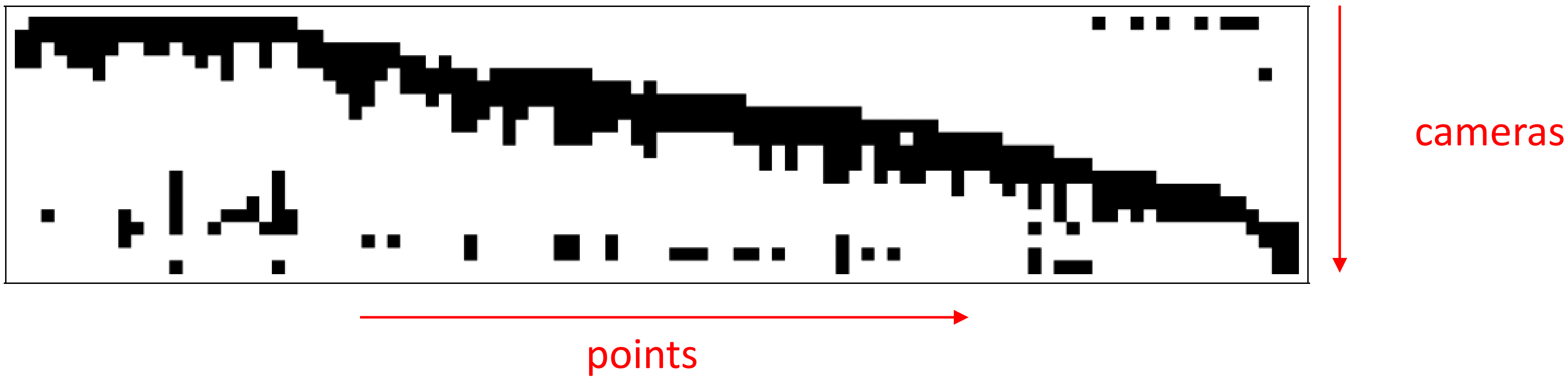
Source: M. Hebert

Dealing with missing data

- So far, we have assumed that all points are visible in all views

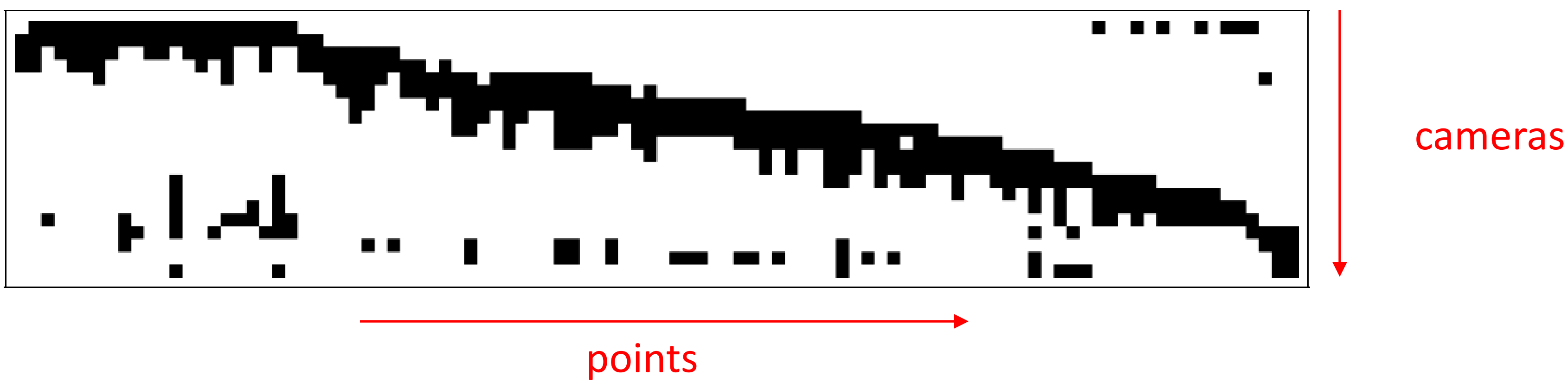
Dealing with missing data

- 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

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



One solution:

- Solve using a dense submatrix of visible points
- Iteratively add new cameras

Further reading

- Short explanation of Affine SfM: class notes from Lischinksi and Gruber

  http://www.cs.huji.ac.il/~csip/sfm.pdf

- Clear explanation of epipolar geometry and projective SfM
  - http://mi.eng.cam.ac.uk/~cipolla/publications/contributionToEditedBook/2008-SFM-chapters.pdf

Important Work in SfM

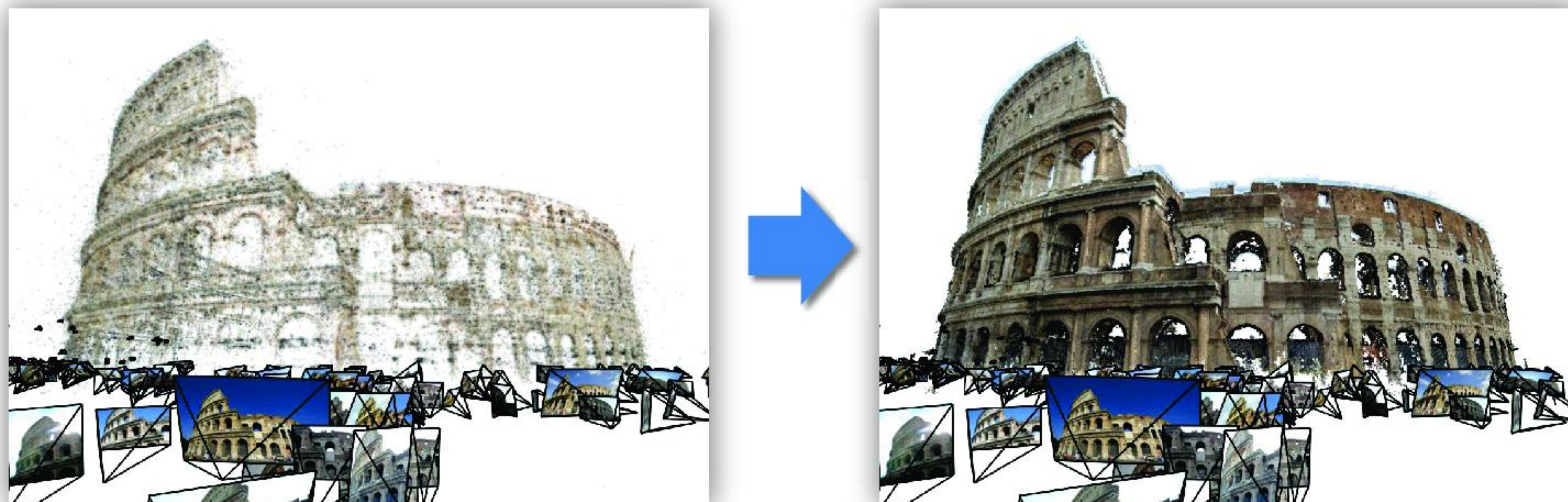
- Reconstruct from many images by efficiently finding subgraphs
  - http://www.cs.cornell.edu/projects/matchminer/ (Lou et al. ECCV 2012)

- Improving efficiency of bundle adjustment or
  - http://vision.soic.indiana.edu/projects/disco/ (Crandall et al. ECCV 2011)
  - http://imagine.enpc.fr/~moulonp/publis/iccv2013/index.html (Moulin et al. ICCV 2013)

(best method with software available; also has good overview of recent methods)

Reconstruction of Cornell (Crandall et al. ECCV 2011)

Multi-view stereo



Multi-view stereo

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 (special rig, camera network or video sequence)
- Calibration may be known or unknown

Multi-view stereo: Basic idea

Source: Y. Furukawa

Multi-view stereo: Basic idea

Source: Y. Furukawa

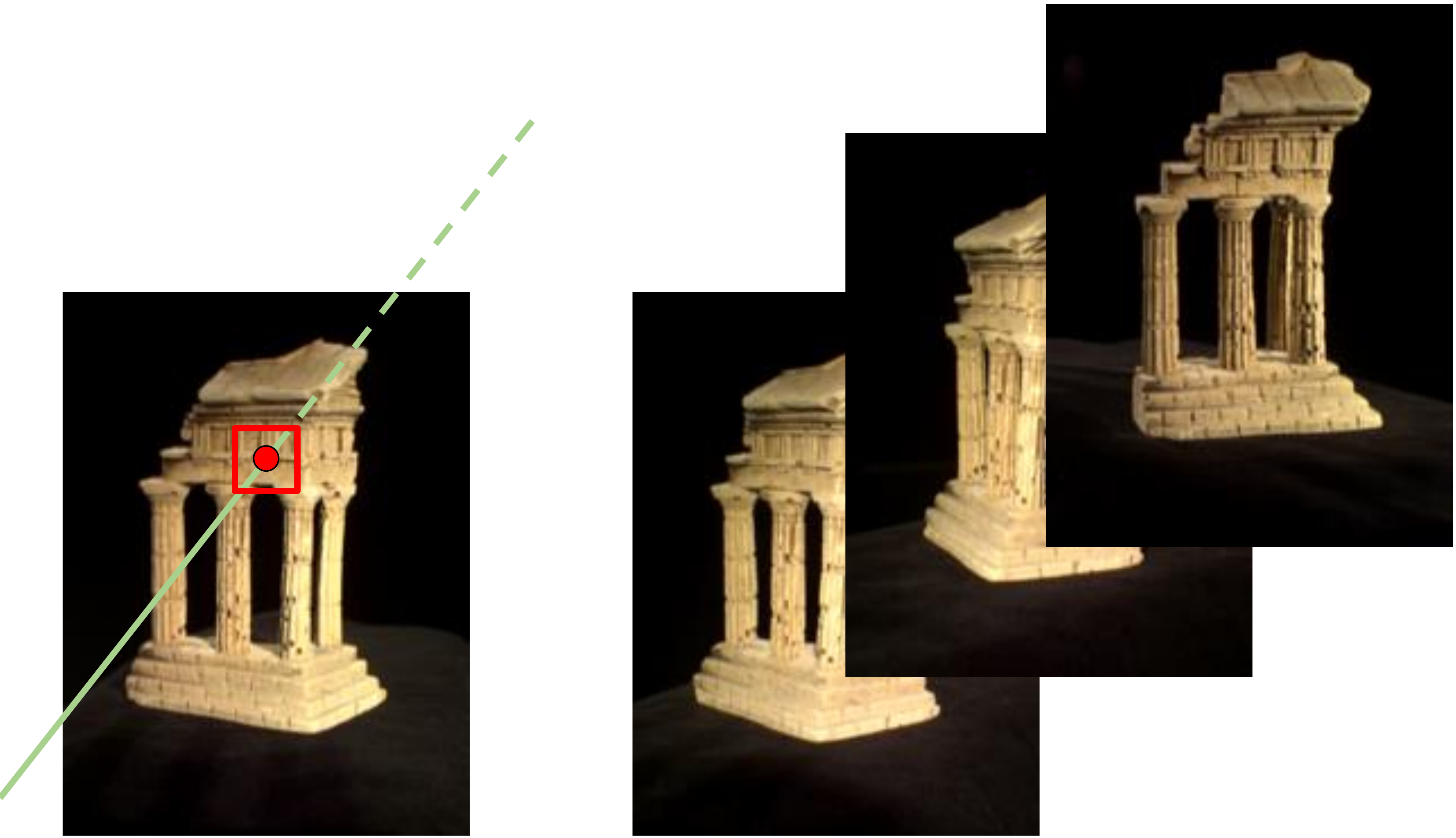
Multi-view stereo: Basic idea

Source: Y. Furukawa

Multi-view stereo: Basic idea

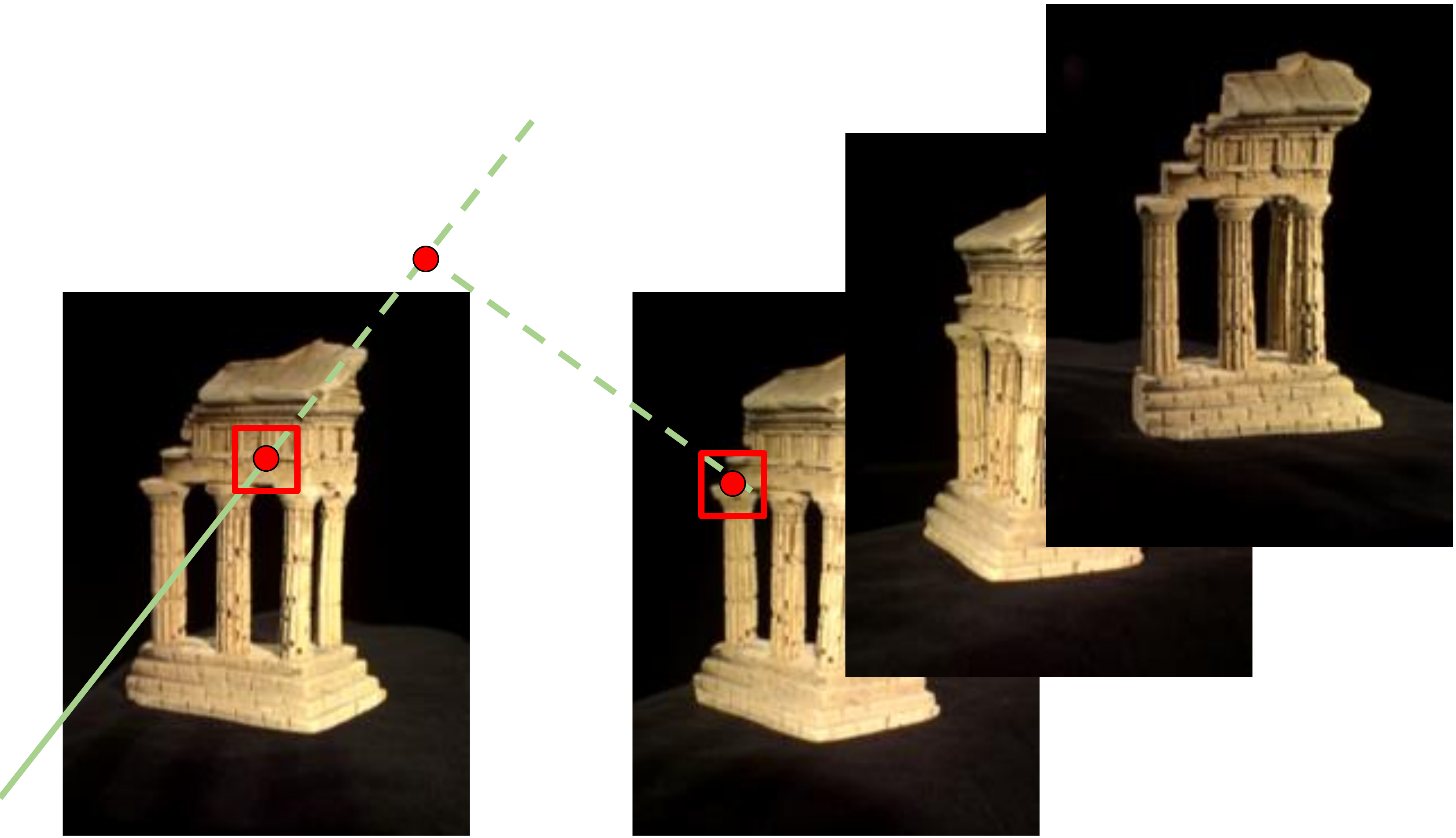
Source: Y. Furukawa

Multi-view stereo: Basic idea



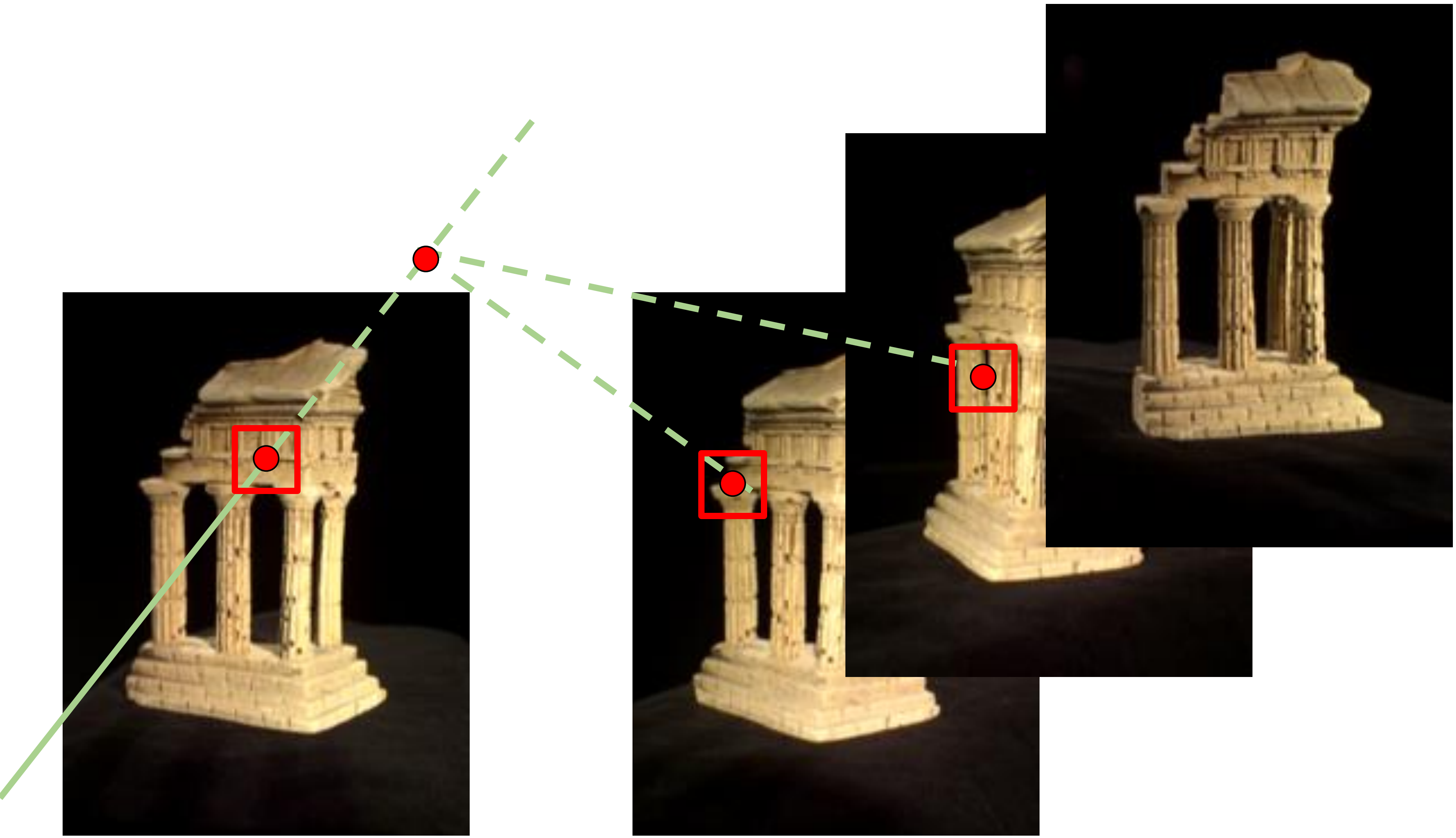
Source: Y. Furukawa

Multi-view stereo: Basic idea



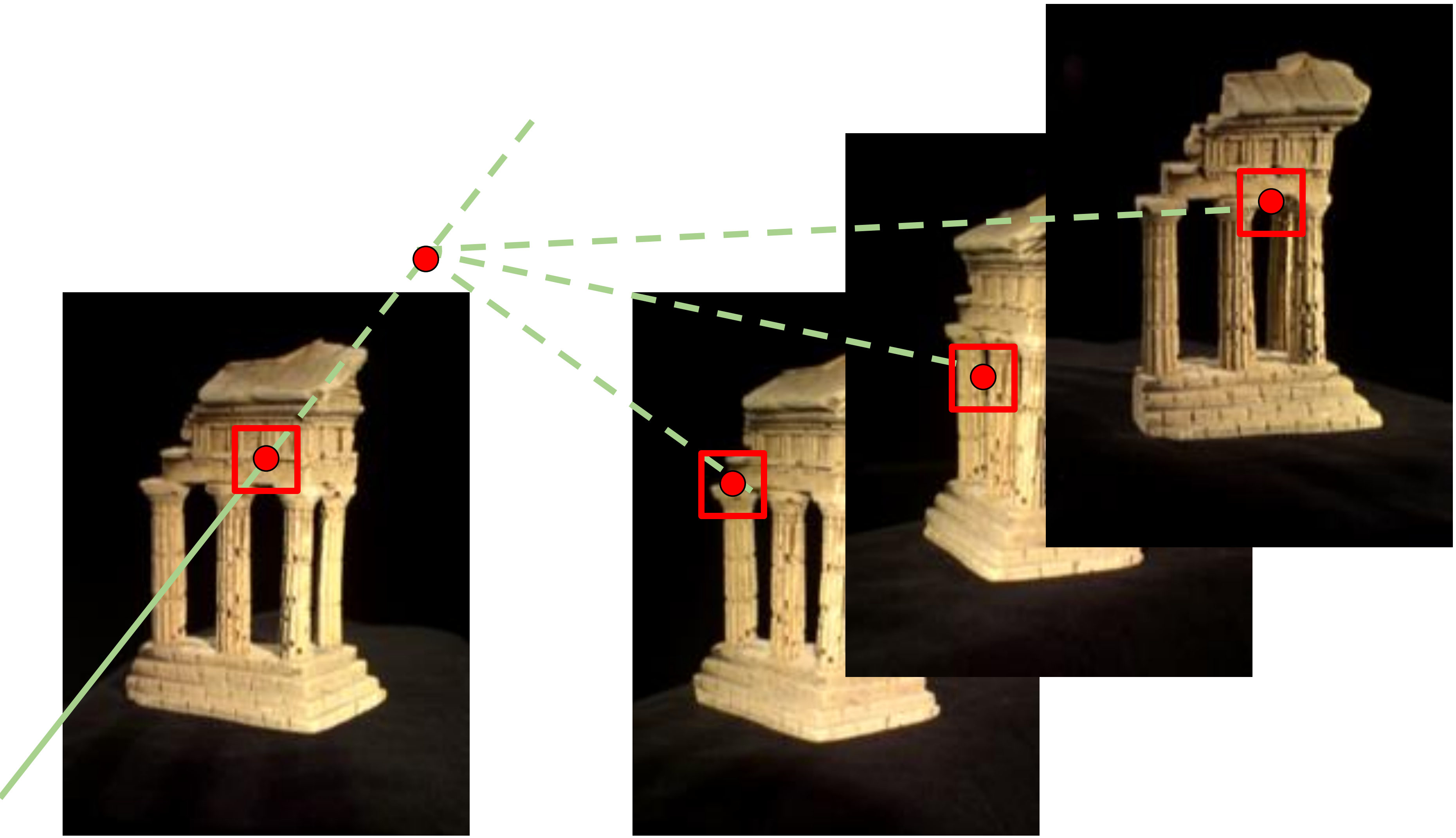
Source: Y. Furukawa

Multi-view stereo: Basic idea



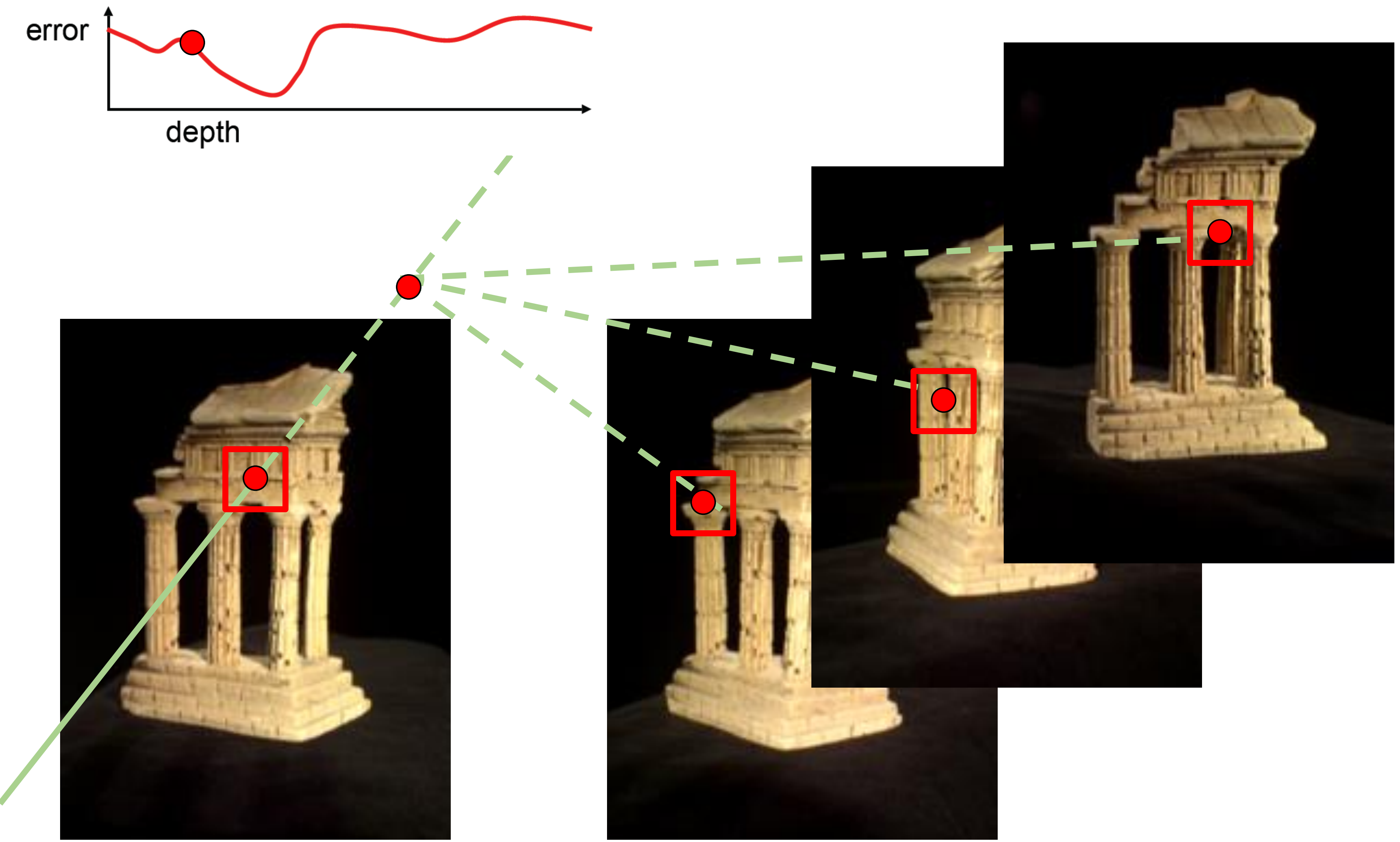
Source: Y. Furukawa

Multi-view stereo: Basic idea



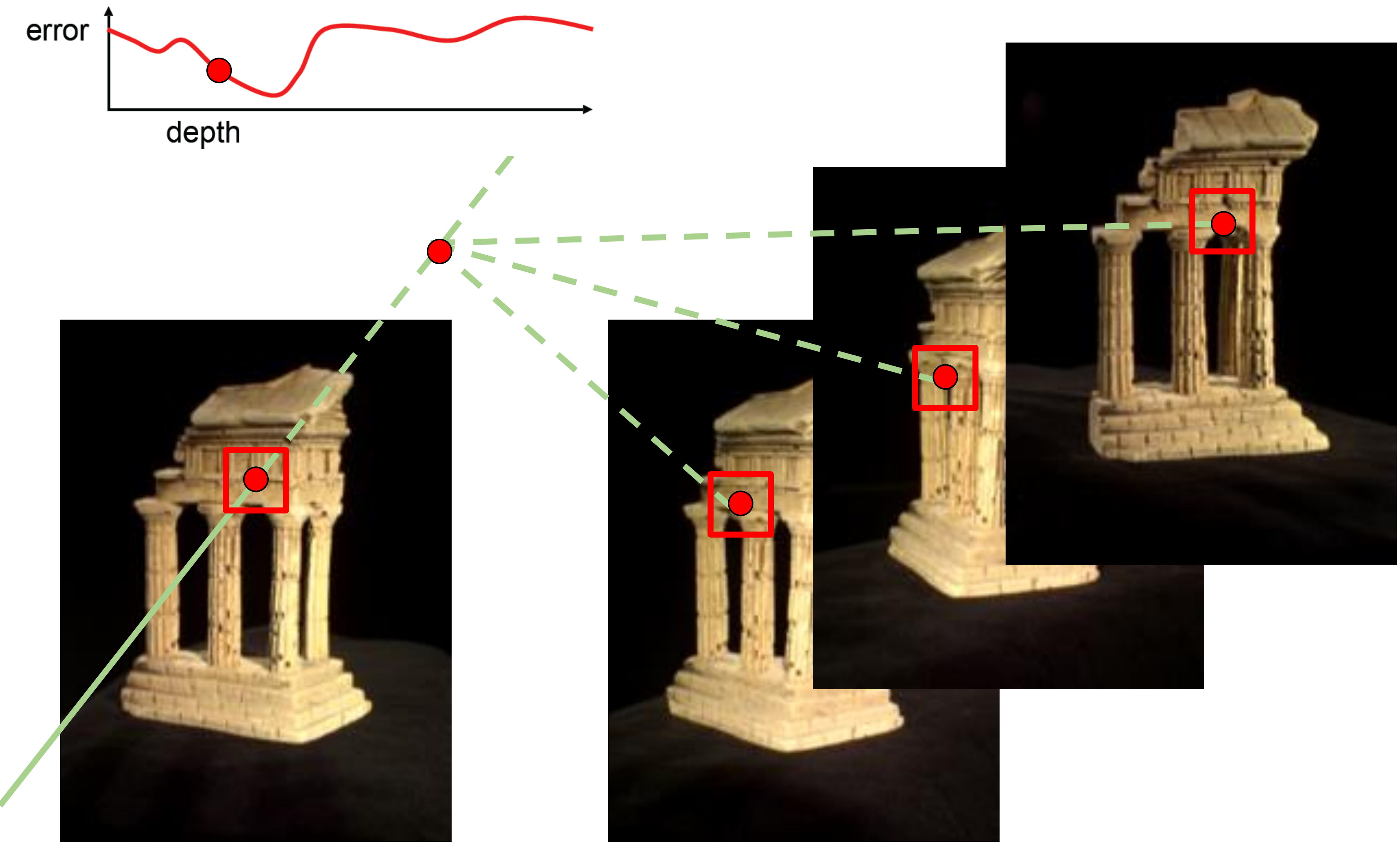
Source: Y. Furukawa

Multi-view stereo: Basic idea



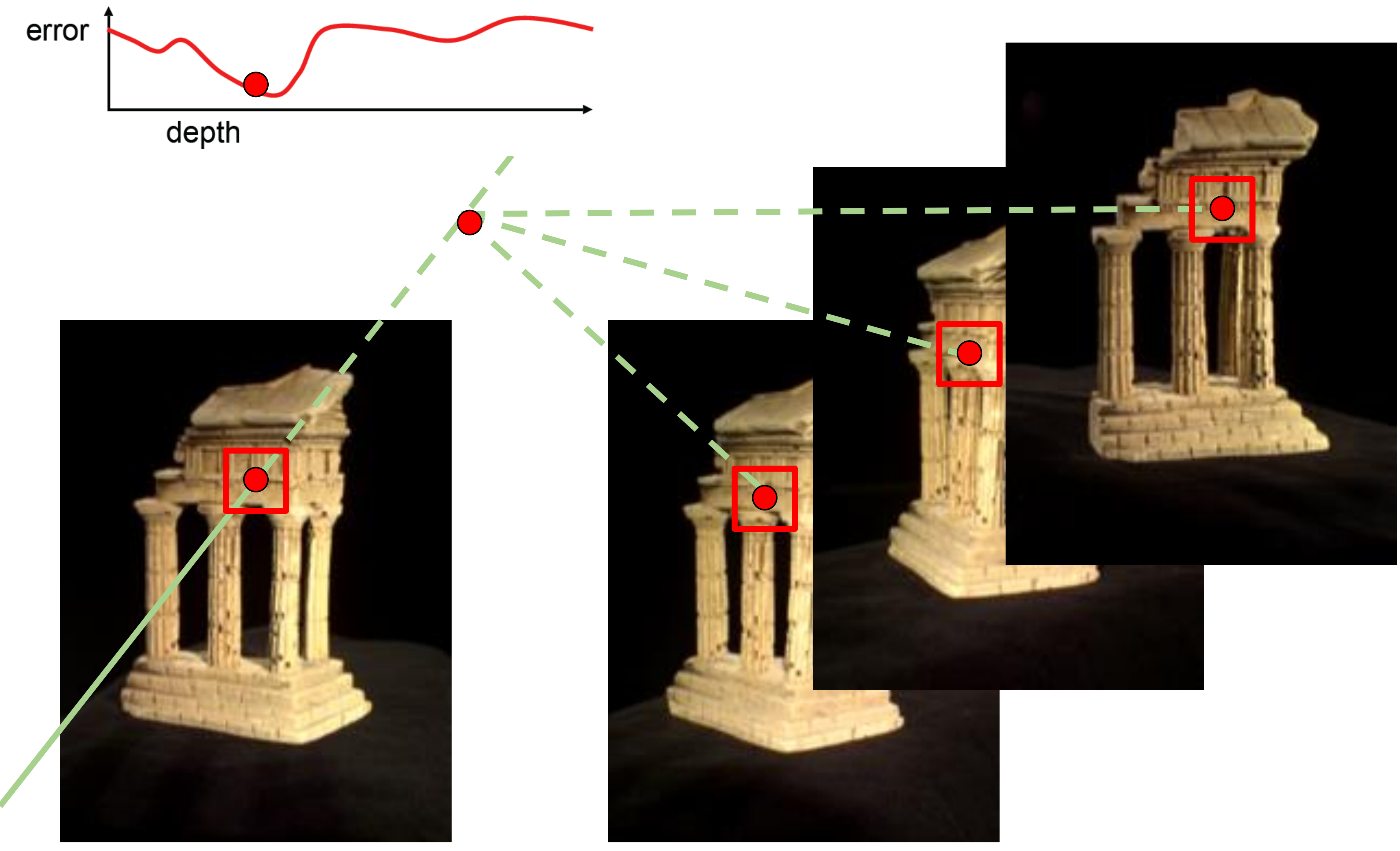
Source: Y. Furukawa

Multi-view stereo: Basic idea



Source: Y. Furukawa

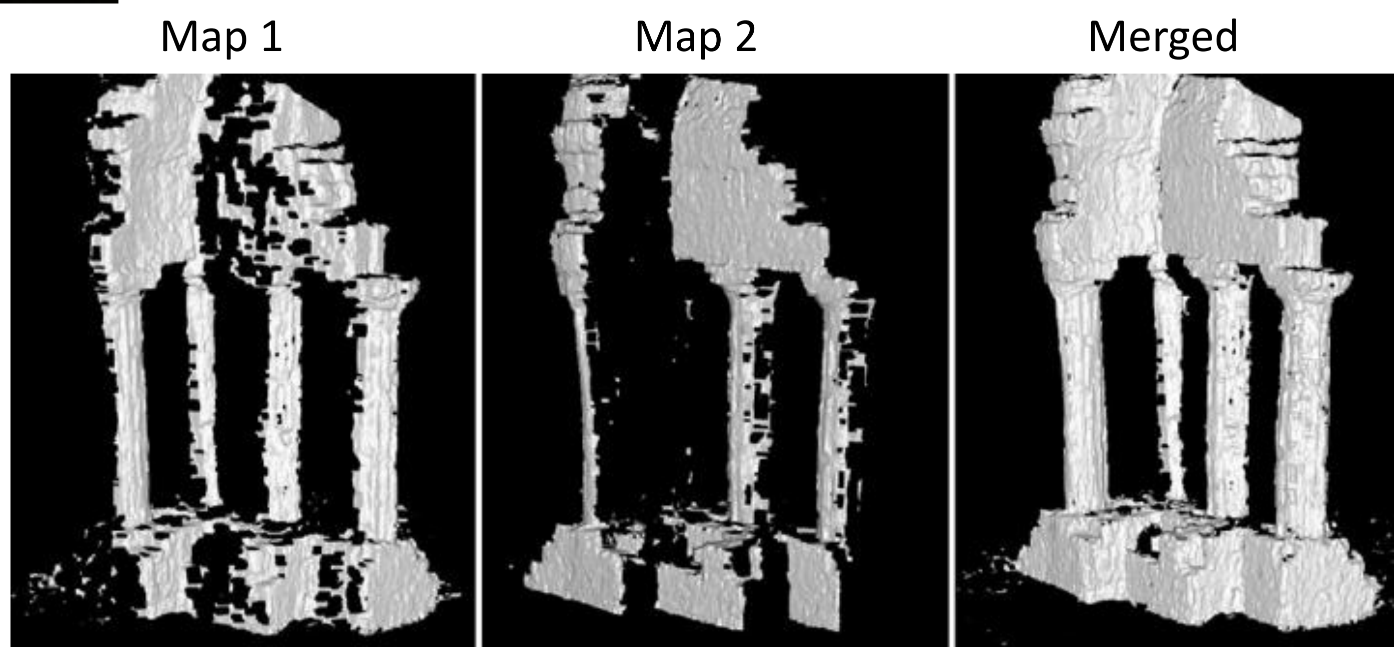
Multi-view stereo: Basic idea



Source: Y. Furukawa

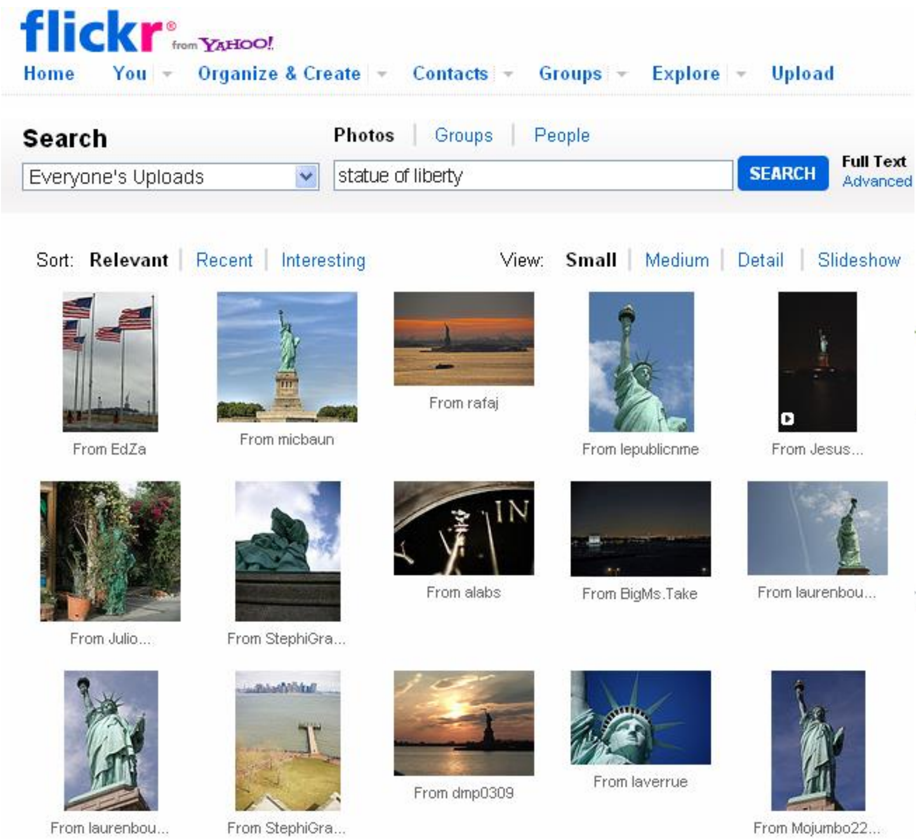
Merging depth maps

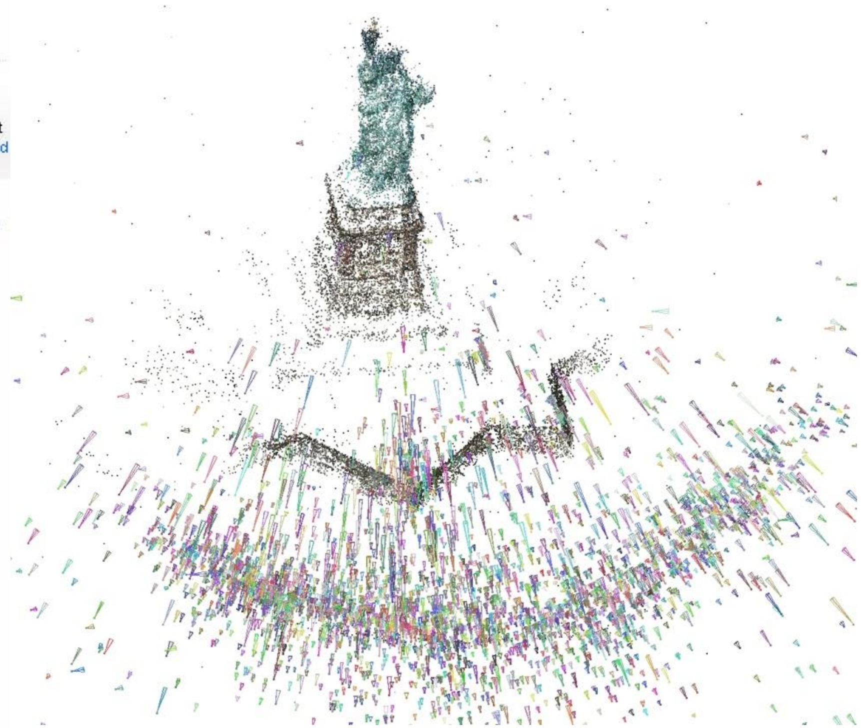
- Given a group of images, choose each one as reference and compute a depth map w.r.t. that view using a multi-baseline approach
- Merge multiple depth maps to a volume or a mesh (see, e.g., Curless and Levoy 96)

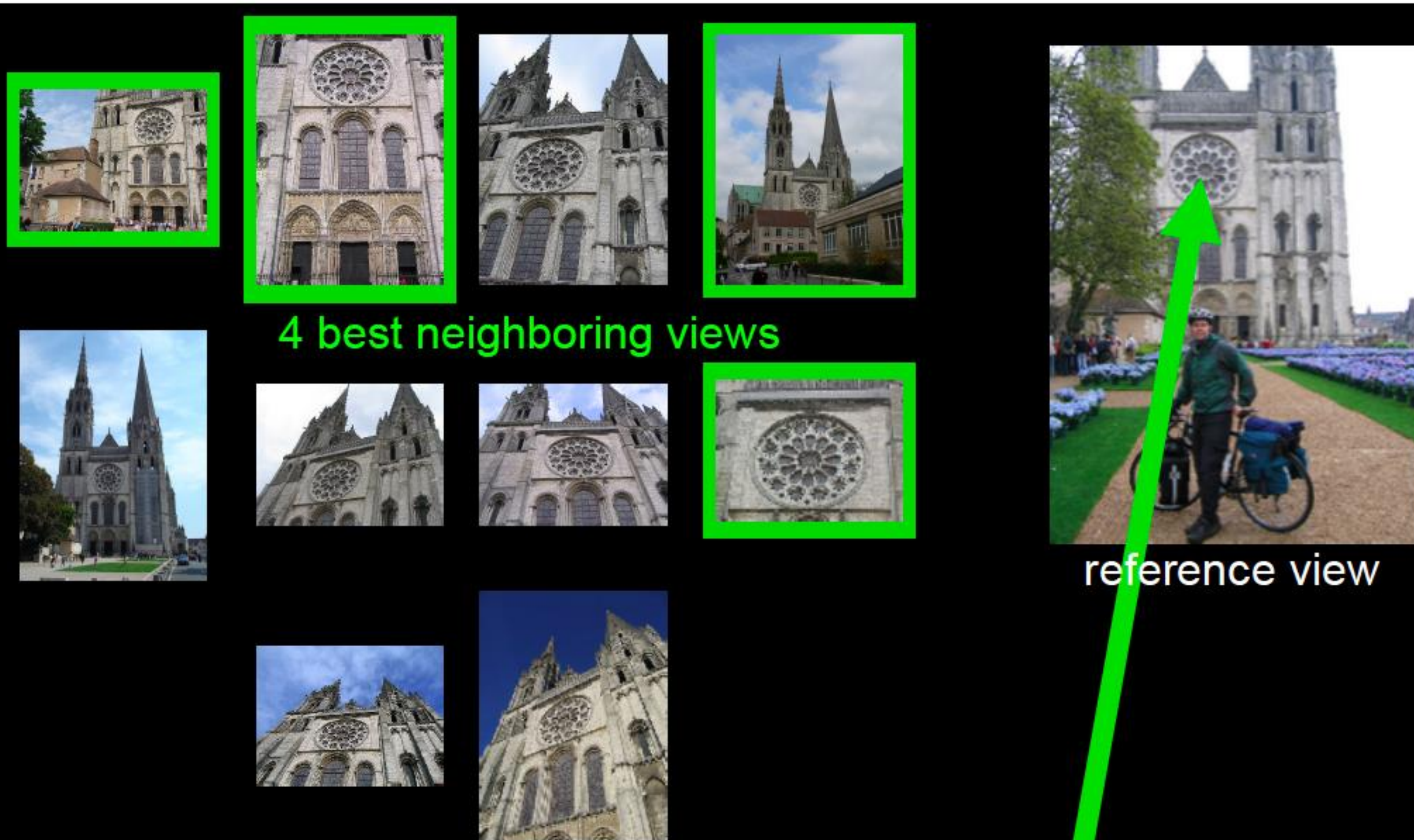


Stereo from community photo collections

- Need *structure from motion* to recover unknown camera parameters
- Need *view selection* to find good groups of images on which to run dense stereo

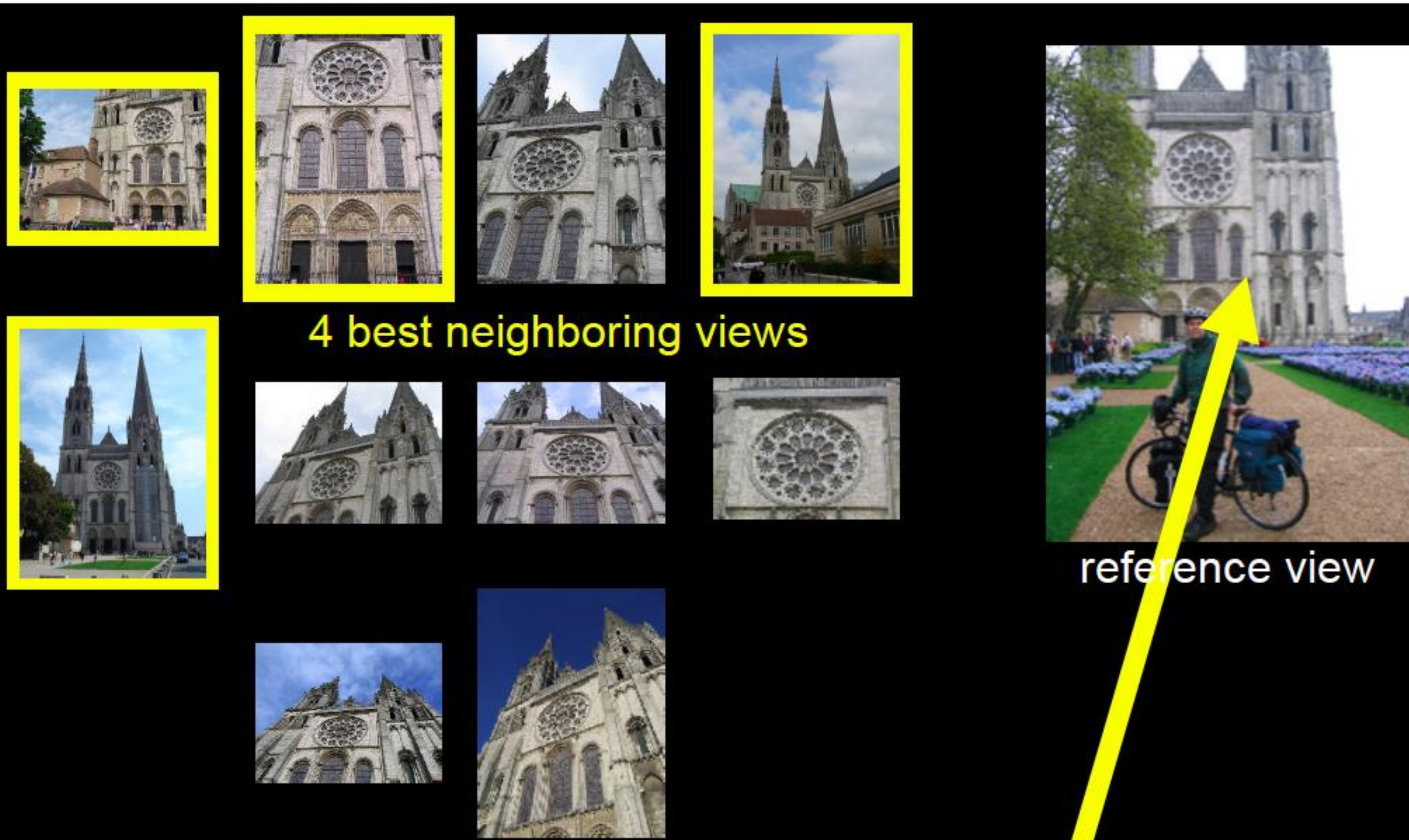






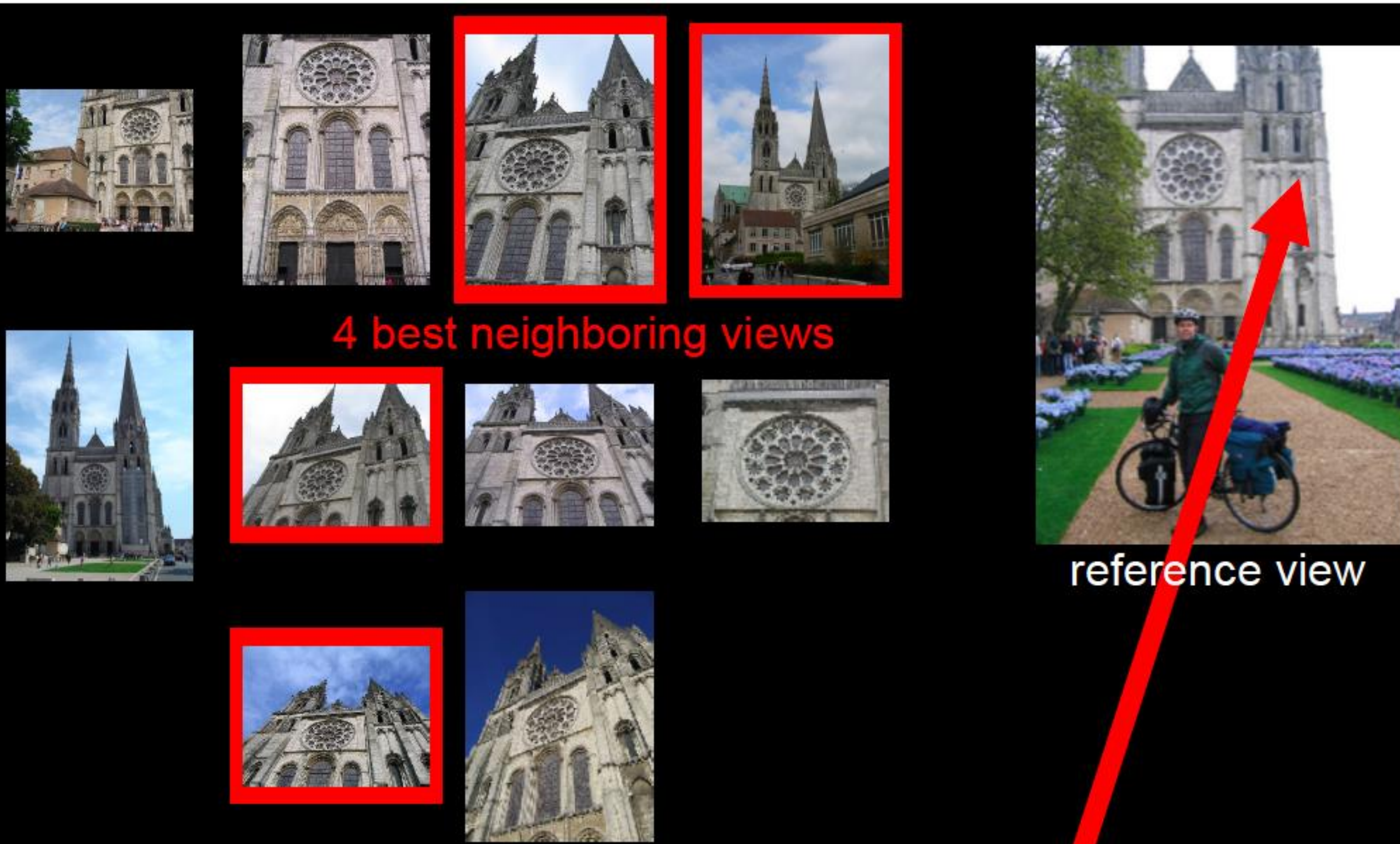
Local view selection

- Automatically select neighboring views for each point in the image
- Desiderata: good matches AND good baselines



Local view selection

- Automatically select neighboring views for each point in the image
- Desiderata: good matches AND good baselines



Local view selection

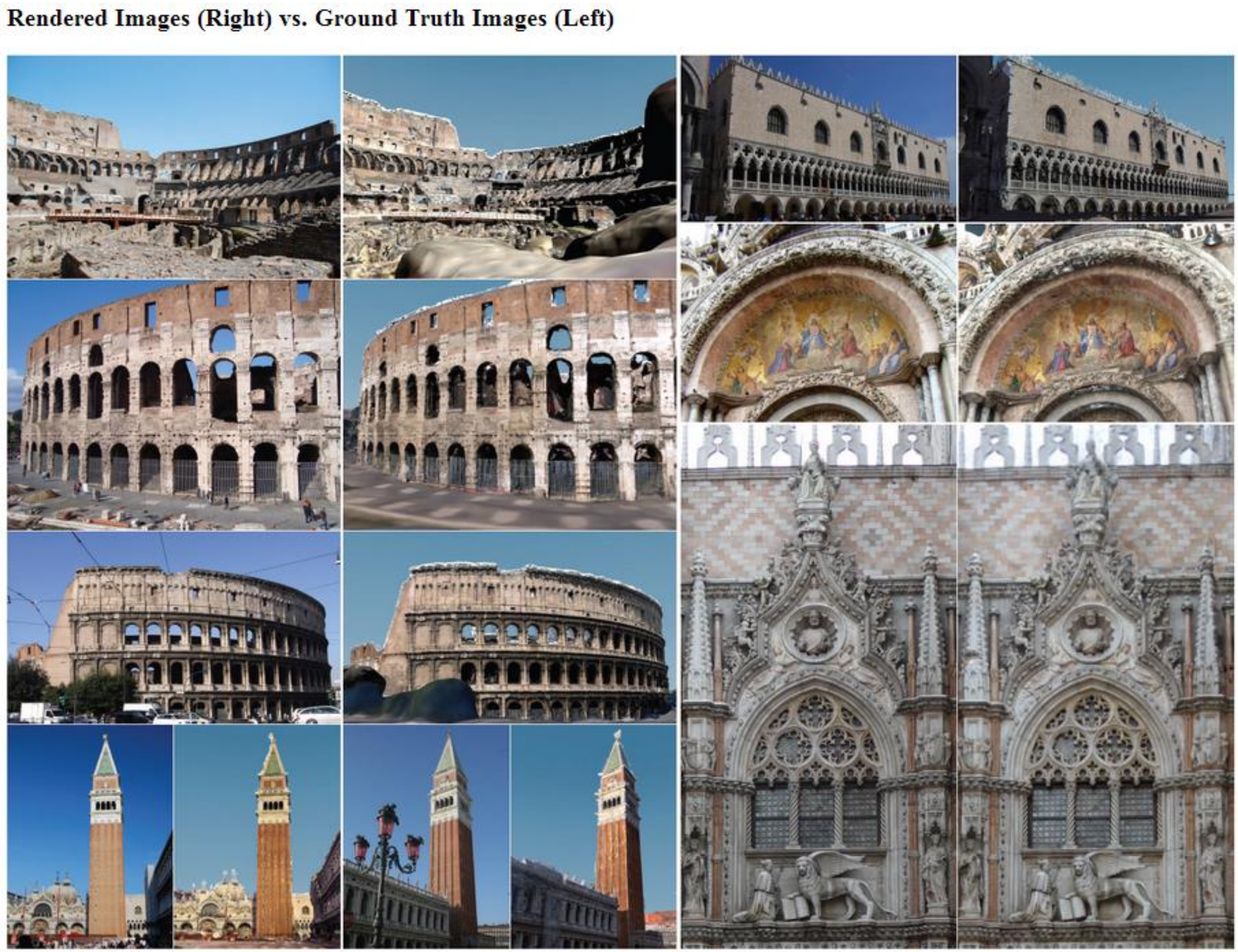
- Automatically select neighboring views for each point in the image
- Desiderata: good matches AND good baselines

Towards Internet-Scale Multi-View Stereo

- YouTube video, high-quality video

Yasutaka Furukawa, Brian Curless, Steven M. Seitz and Richard Szeliski, Towards Internet-scale Multi-view Stereo,CVPR 2010.

The Visual Turing Test for Scene Reconstruction



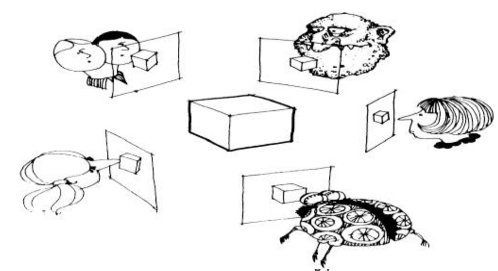
Q. Shan, R. Adams, B. Curless, Y. Furukawa, and S. Seitz, ["The Visual Turing Test for Scene Reconstruction,"]() 3DV 2013.

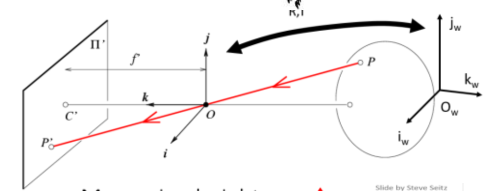
The Reading List

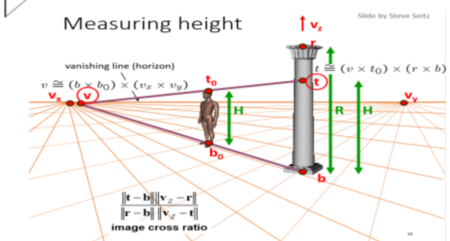
- "A computer algorithm for reconstructing a scene from two images", Longuet-Higgins, Nature 1981
- "Shape and motion from image streams under orthography: A factorization method." C. Tomasi and T. Kanade, *IJCV*, 9(2):137-154, November 1992
- "In defense of the eight-point algorithm", Hartley, PAMI 1997
- "An efficient solution to the five-point relative pose problem", Nister, PAMI 2004
- "Accurate, dense, and robust multiview stereopsis", Furukawa and Ponce, CVPR 2007
- "Photo tourism: exploring image collections in 3d", ACM SIGGRAPH 2006
- "Building Rome in a day", Agarwal et al., ICCV 2009
- https://www.youtube.com/watch?v=kyIzMr917Rc, 3D Computer Vision: Past, Present, and Future

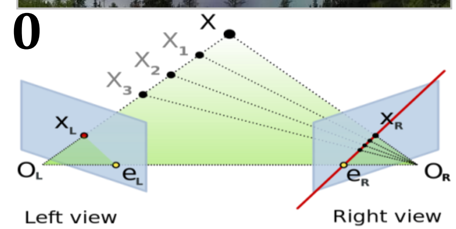
This Module: Perspective and 3D Geometry

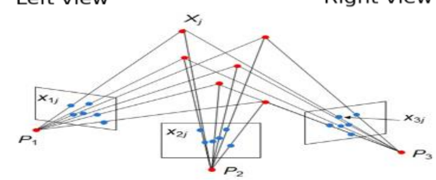
- **Camera Models and Projective Geometry**
  - Perspective projection
  - Vanishing points/lines
- **Projection Matrix and Calibration**
  - $\mathrm{x} = \mathbf{K}[\mathbf{R} \;\; \mathbf{t}]\mathbf{X}$
  - Calibration using known 3D object or vanishing points
- **Single-view Metrology and Camera Properties**
  - Measuring size using perspective cues
  - Focal length, Field of View, etc.
- **Photo stitching**
  - Homography relates rotating cameras $\mathbf{x}' = \mathbf{H}\mathbf{x}$
  - Recover homography using RANSAC
- **Epipolar Geometry and Stereo Vision**
  - Fundamental/essential matrix relates two cameras $\mathbf{x}'\mathbf{F}\mathbf{x} = \mathbf{0}$
  - Recover $\mathbf{F}$ using RANSAC + normalized 8-point algorithm
  - Enforce rank 2 using SVD
- **Structure from motion**
  - How can we recover 3D points from multiple images?











Acknowledgements

- Thanks to the following researchers for making their teaching/research material online
  - Forsyth
  - Steve Seitz
  - Noah Snavely
  - J.B. Huang
  - Derek Hoiem
  - J. Hays
  - J. Johnson
  - R. Girshick
  - S. Lazebnik
  - K. Grauman
  - Antonio Torralba
  - Rob Fergus
  - Leibe
  - And many more ……….

Next Module: Recognition and Learning

- Image Features and Categorization
- Classifiers
- Neural Networks
- Convolutional Neural Networks
- Object Detection
- Segmentation
- Image Generation
- Etc.