— CM50248 — 2017/2018 —

Visual Understanding 1

Dr Christian Richardt

Today's roadmap

1. Optical flow
2. Stereo matching
3. Structure-from-motion + bundle adjustment
4. Introduction to recognition for vision

Optical flow

Why estimate visual motion?

- Visual motion can be annoying
  - Camera instabilities, jitter
  - Measure it; remove it (stabilize)
- Visual motion indicates dynamics in the scene
  - Moving objects, behavior
  - Track objects and analyze trajectories
- Visual motion reveals spatial layout
  - Motion parallax

Slide by Richard Szeliski

Classes of techniques

- Feature-based methods
  - Extract visual features (corners, textured areas) and track them
  - Sparse motion fields, but possibly robust tracking
  - Suitable especially when image motion is large (10s of pixels)
- Direct-methods
  - Directly recover image motion from spatio-temporal image brightness variations
  - Global motion parameters directly recovered without an intermediate feature motion calculation
  - Dense motion fields, but more sensitive to appearance variations
  - Suitable for video and when image motion is small (< 10 pixels)

Slide by Richard Szeliski

Patch matching (revisited)

- How do we determine correspondences?
  - block matching or SSD (sum squared differences)

$$E(x, y; d) = \sum_{(x',y') \in N(x,y)} [I_L(x'+d, y') - I_R(x', y')]^2$$

Slide by Richard Szeliski

The brightness constraint

Brightness constancy equation:

$$J(x,y) \approx \boxed{I(x+u(x,y), y+v(x,y))}$$

Or, equivalently, minimize :

$$E(u,v) = (J(x,y) - I(x+u, y+v))^2$$

Linearizing (assuming small *u*, *v*)
using Taylor series expansion:

$$J(x,y) \approx \boxed{I(x,y) + I_x(x,y) \cdot u(x,y) + I_y(x,y) \cdot v(x,y)}$$

Slide by Richard Szeliski

Gradient constraint (or the optical flow constraint)

$$E(u,v) = (I_x \cdot u + I_y \cdot v + I_t)^2$$

Minimizing:

$$\frac{\partial E}{du} = \frac{\partial E}{dv} = 0$$

$$I_x(I_x u + I_y v + I_t) = 0$$

$$I_y(I_x u + I_y v + I_t) = 0$$

In general $I_x, I_y \neq 0$

Hence, $I_x \cdot u + I_y \cdot v + I_t \approx 0$

Slide by Richard Szeliski

Patch translation [Lucas-Kanade]

Assume a single velocity for all pixels within an image patch

$$E(u,v) = \sum_{x,y \in \Omega} \left(I_x(x,y)u + I_y(x,y)v + I_t \right)^2$$

Minimizing

$$\begin{bmatrix} \sum I_x^2 & \sum I_x I_y \\ \sum I_x I_y & \sum I_y^2 \end{bmatrix} \begin{pmatrix} u \\ v \end{pmatrix} = -\begin{pmatrix} \sum I_x I_t \\ \sum I_y I_t \end{pmatrix}$$

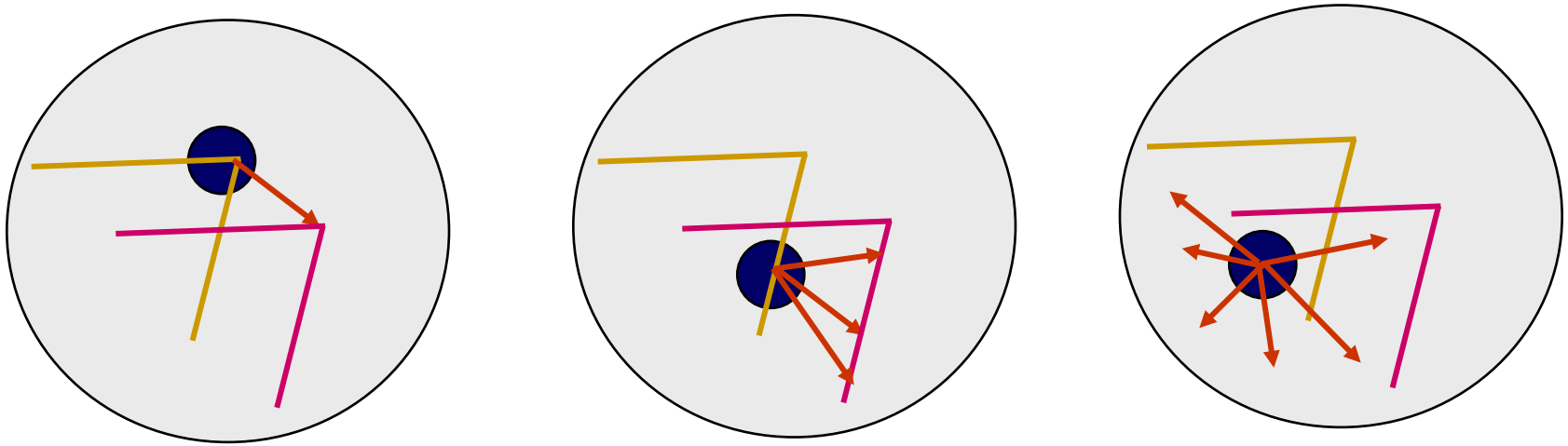
$$\left(\sum \nabla I \nabla I^T \right) \vec{U} = -\sum \nabla I I_t$$

LHS: sum of the 2×2 outer product of the gradient vector

Slide by Richard Szeliski

Local patch analysis

- How *certain* are the motion estimates?



Slide by Richard Szeliski

The aperture problem

Let $M = \sum (\nabla I)(\nabla I)^T$ and $b = \begin{bmatrix} -\sum I_x I_t \\ -\sum I_y I_t \end{bmatrix}$

- Algorithm: At each pixel compute U by solving $MU = b$

- M is singular if all gradient vectors point in the same direction
  - e.g., along an edge
  - of course, trivially singular if the summation is over a single pixel or there is no texture
  - i.e., only *normal flow* is available (aperture problem)

- Corners and textured areas are OK

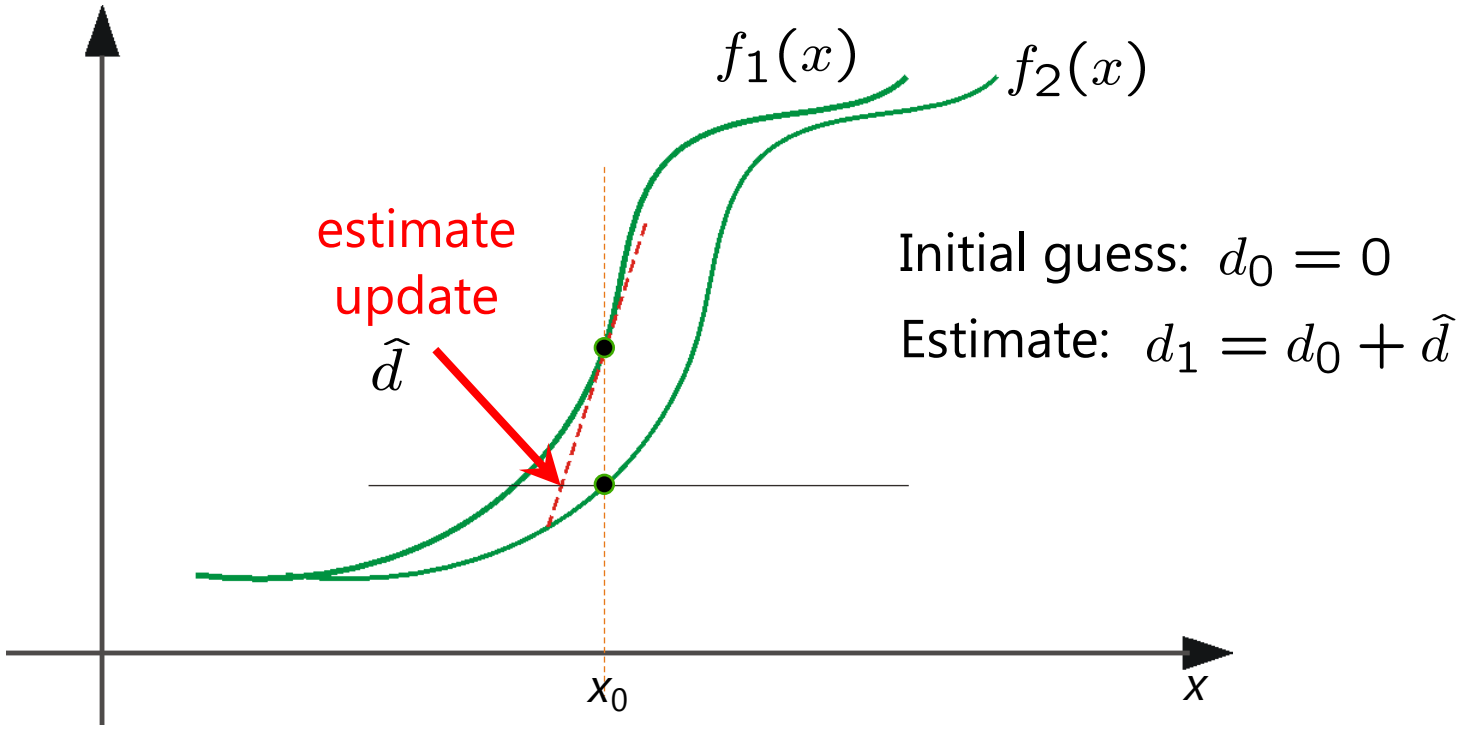
Slide by Richard Szeliski

Iterative refinement

- Estimate velocity at each pixel using one iteration of Lucas–Kanade estimation
- Warp one image toward the other using the estimated flow field
  - (easier said than done)
- Refine estimate by repeating the process

Slide by Richard Szeliski

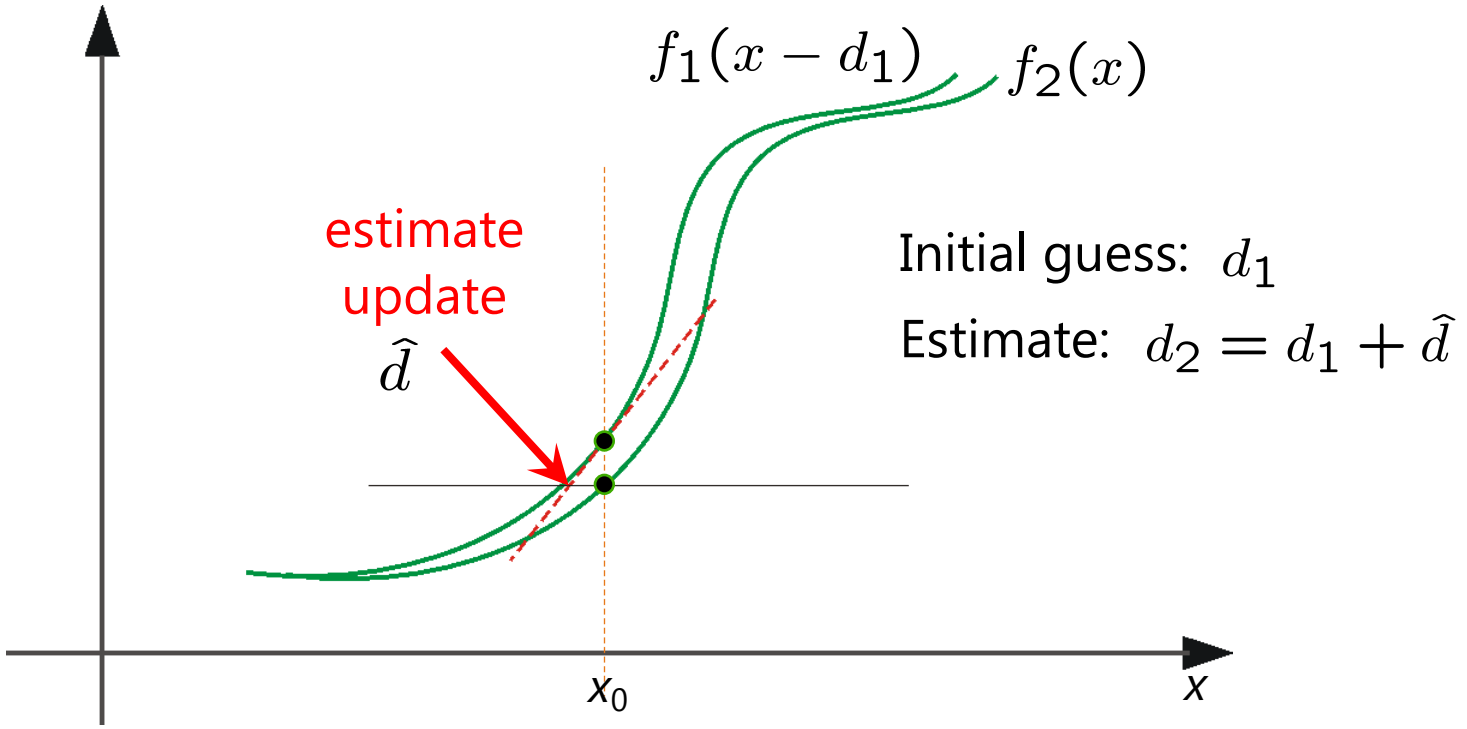
Optical flow: iterative estimation



(using *d* for *displacement* here instead of *u*)**)**

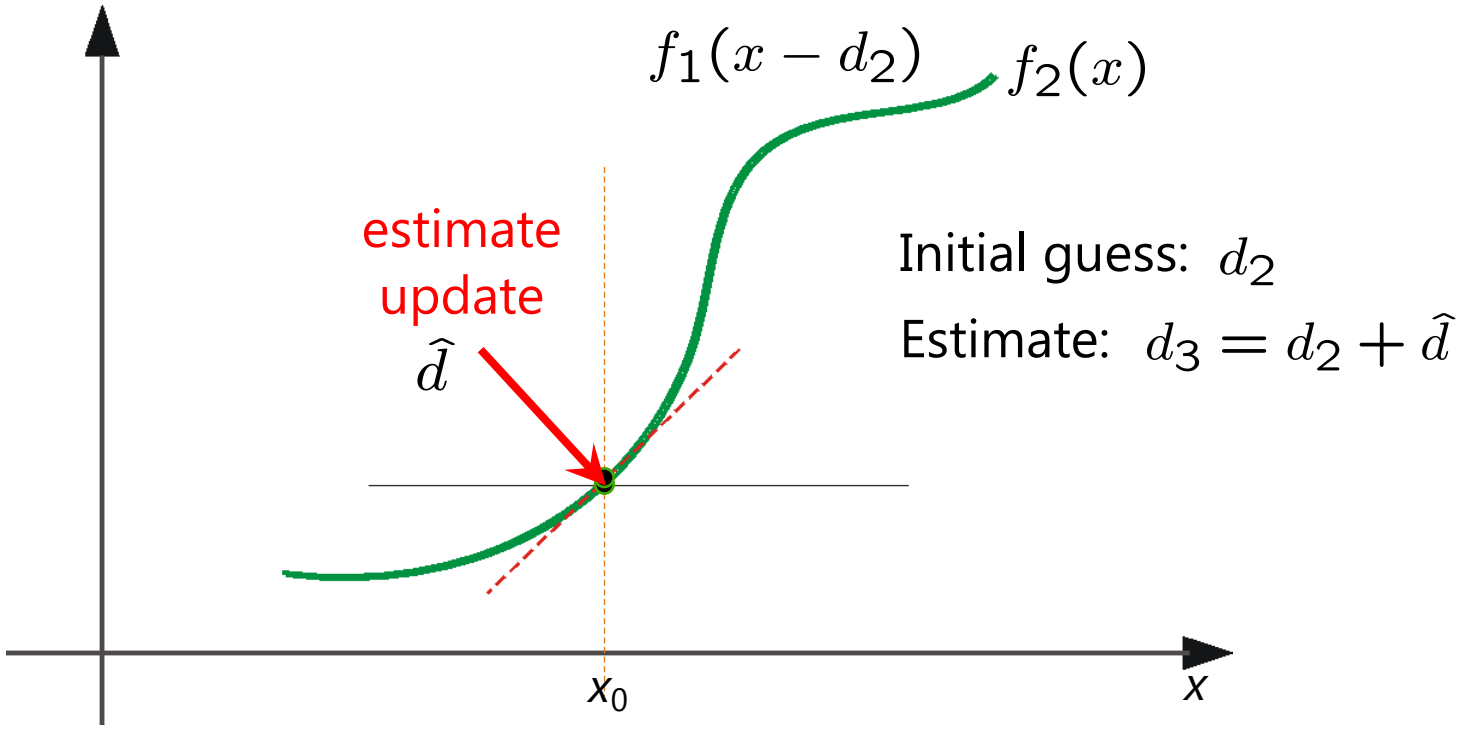
Slide by Richard Szeliski

Optical flow: iterative estimation



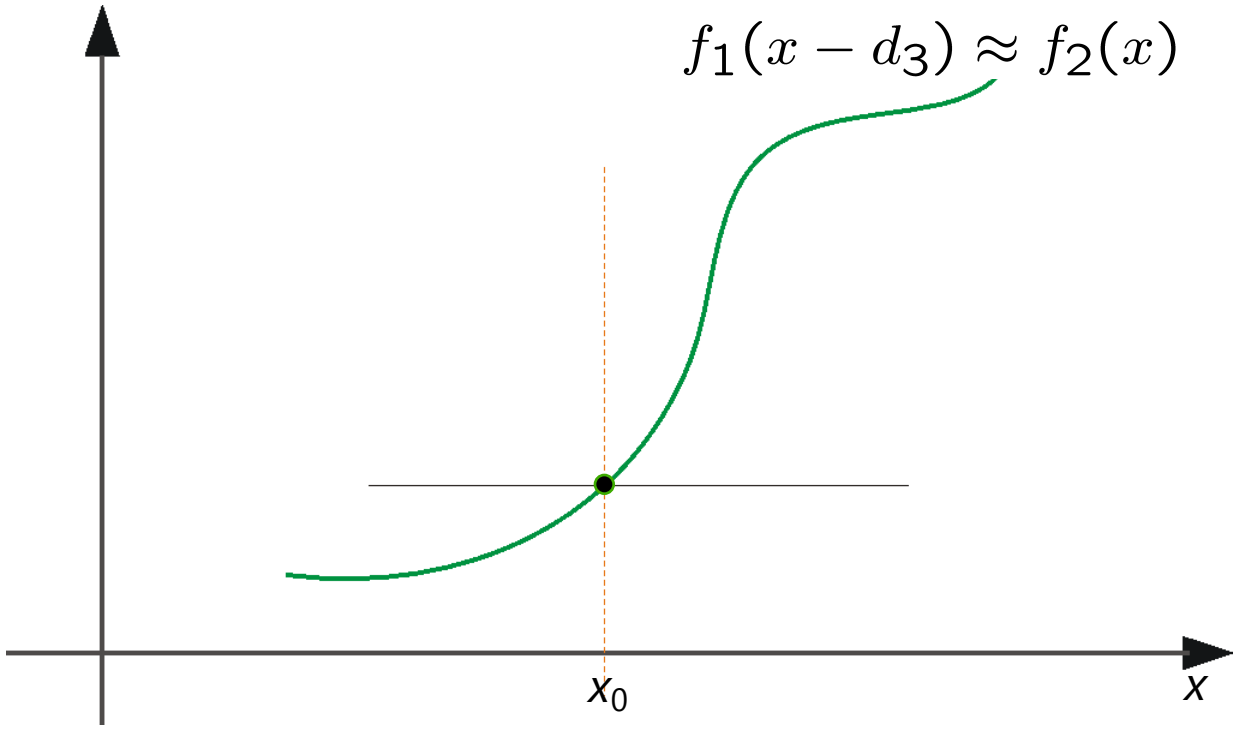
Slide by Richard Szeliski

Optical flow: iterative estimation



Slide by Richard Szeliski

Optical flow: iterative estimation



Slide by Richard Szeliski

Optical flow: iterative estimation

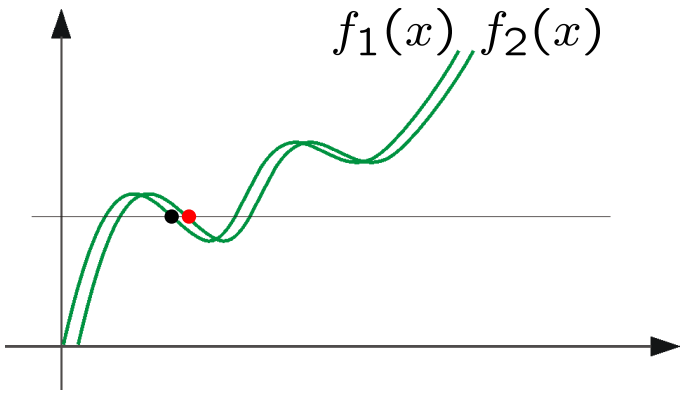
- Some implementation issues:
  - Warping is not easy (ensure that errors in warping are smaller than the estimate refinement)
  - Warp one image, take derivatives of the other so you don't need to re-compute the gradient after each iteration
  - Often useful to low-pass filter the images before motion estimation (for better derivative estimation, and linear approximations to image intensity)

Slide by Richard Szeliski

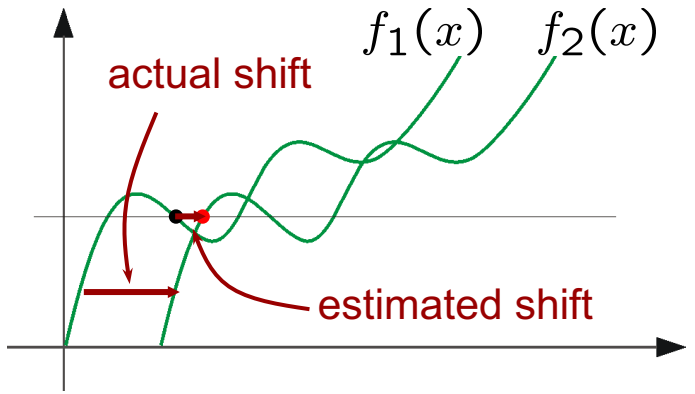
Optical flow: aliasing

Temporal aliasing causes ambiguities in optical flow because images can have many pixels with the same intensity.

How do we know which 'correspondence' is correct?



nearest match is correct
(no aliasing)



nearest match is incorrect
(aliasing)

To overcome aliasing: coarse-to-fine estimation.

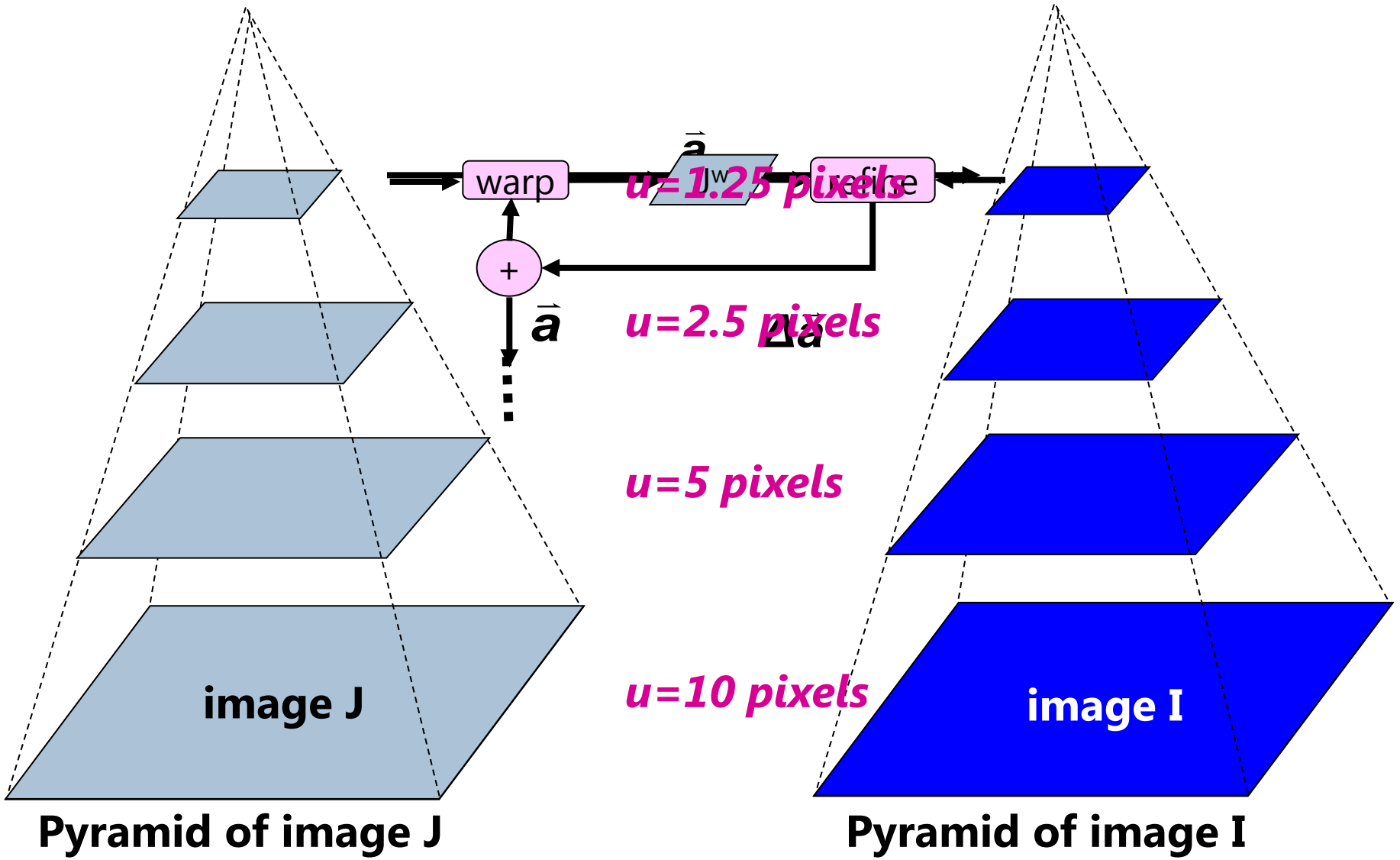
Slide by Richard Szeliski

Limits of the gradient method

- Fails when intensity structure in window is poor
- Fails when the displacement is large (typical operating range is motion of 1 pixel)
  - Linearization of brightness is suitable only for small displacements
- Also, brightness is not strictly constant in images
  - actually less problematic than it appears, since we can pre-filter images to make them look similar

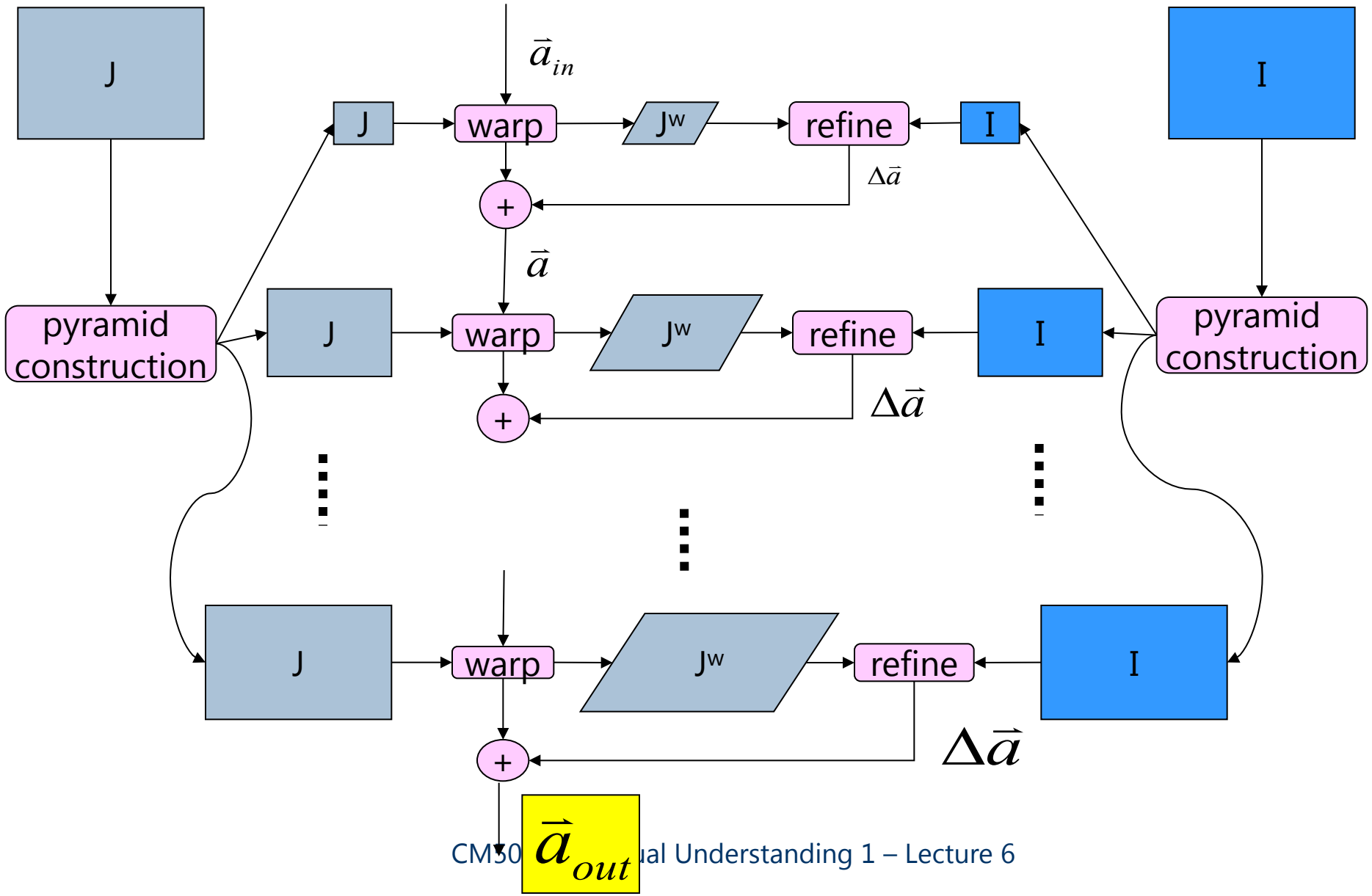
Slide by Richard Szeliski

Coarse-to-fine estimation



Slide by Richard Szeliski

Coarse-to-fine estimation



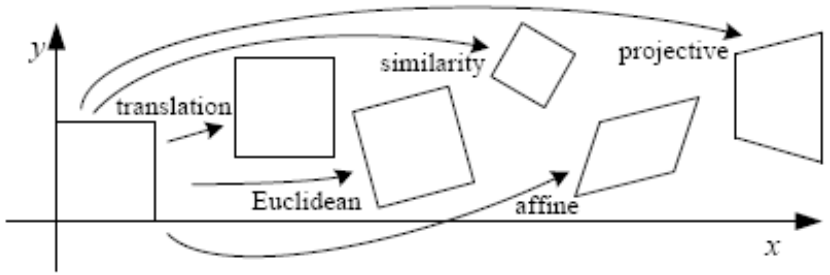
Slide by Richard Szeliski

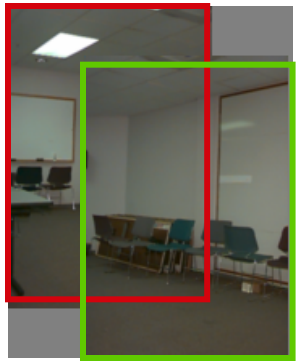
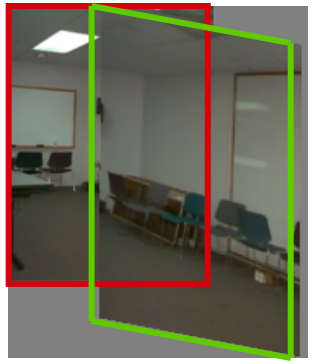
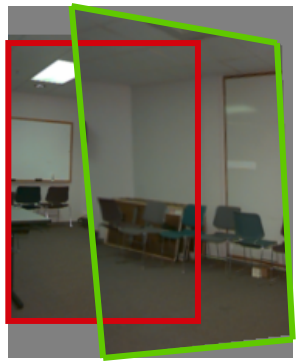
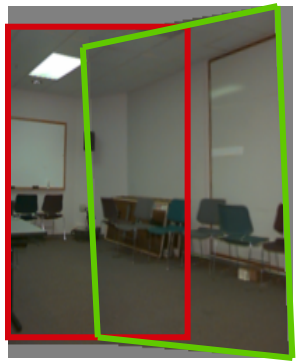
Global (parametric) motion models

- 2D models:
  - Affine
  - Quadratic
  - Planar projective transform (homography)
- 3D models:
  - Instantaneous camera motion models
  - Homography + epipole
  - Plane + Parallax

Slide by Richard Szeliski

Motion models



Translation	Affine	Perspective	3D rotation
			
2 unknowns	6 unknowns	8 unknowns	3 unknowns

Slide by Richard Szeliski

Example: affine motion

$$u(x,y) = a_1 + a_2 x + a_3 y$$
$$v(x,y) = a_4 + a_5 x + a_6 y$$

Substituting into the brightness constancy equation:

$$I_x(a_1 + a_2 x + a_3 y) + I_y(a_4 + a_5 x + a_6 y) + I_t \approx 0$$

Each pixel provides 1 linear constraint in 6 *global* unknowns

Least Square Minimization (over all pixels):

$$Err(\vec{a}) = \sum \left[I_x(a_1 + a_2 x + a_3 y) + I_y(a_4 + a_5 x + a_6 y) + I_t \right]^2$$

Slide by Richard Szeliski

Other 2D motion models

Quadratic – instantaneous approximation to planar motion

$$u = q_1 + q_2 x + q_3 y + q_7 x^2 + q_8 xy$$
$$v = q_4 + q_5 x + q_6 y + q_7 xy + q_8 y^2$$

Projective – exact planar motion

$$x' = \frac{h_1 + h_2 x + h_3 y}{h_7 + h_8 x + h_9 y}$$
$$y' = \frac{h_4 + h_5 x + h_6 y}{h_7 + h_8 x + h_9 y}$$
and
$$u = x' - x, \quad v = y' - y$$

Slide by Richard Szeliski

3D motion models

Instantaneous camera motion:

Global parameters: $\Omega_X, \Omega_Y, \Omega_Z, T_X, T_Y, T_Z$

Local Parameter: $Z(x, y)$

$$u = -xy\Omega_X + (1+x^2)\Omega_Y - y\Omega_Z + (T_X - T_Z x)/Z$$
$$v = -(1+y^2)\Omega_X + xy\Omega_Y - x\Omega_Z + (T_Y - T_Z x)/Z$$

Homography+Epipole

Global parameters: $h_1, \ldots, h_9, t_1, t_2, t_3$

Local Parameter: $\gamma(x, y)$

$$x' = \frac{h_1 x + h_2 y + h_3 + \gamma t_1}{h_7 x + h_8 y + h_9 + \gamma t_3}$$
$$y' = \frac{h_4 x + h_5 y + h_6 + \gamma t_1}{h_7 x + h_8 y + h_9 + \gamma t_3}$$
$$\text{and:} \quad u = x' - x, \quad v = y' - y$$

Residual Planar Parallax Motion

Global parameters: t_1, t_2, t_3

Local Parameter: $\gamma(x, y)$

$$u = x^w - x = \frac{\gamma}{1+\gamma t_3}(t_3 x - t_1)$$
$$v = y^w - x = \frac{\gamma}{1+\gamma t_3}(t_3 y - t_2)$$

Slide by Richard Szeliski

How well do these techniques work?

- A database and evaluation methodology for optical flow

  Simon Baker, Daniel Scharstein, JP Lewis, Stefan Roth, Michael Black and Richard Szeliski

- ICCV 2007

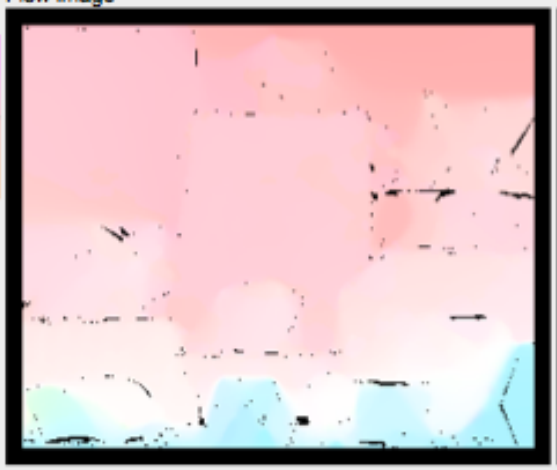
  http://vision.middlebury.edu/flow/

Optical flow evaluation results Choose error measures: Average SD R1.0 R3.0 R5.0 A50 A75 A95

Average angle error	avg.	Dimetrodon (Hidden texture) GT im0 im1			Seashell (Hidden texture) GT im0 im1			Rock (Synthetic) GT im0 im1			Grove (Synthetic) GT im0 im1			Yosemite (Synthetic) GT im0 im1			Venus (Stereo) GT im0 im1			Moebius (Stereo) GT im0 im1		
	rank	all	disc	untext	all	disc	untext	all	disc	untext	all	disc	untext	all	disc	untext	all	disc	untext	all	disc	untext
Bruhn et al.	1.6	10.99 3	**9.41** 1	14.22 3	11.09 2	19.48 2	16.21 2	**6.14** 1	**17.41** 1	12.86 2	**6.32** 1	**12.41** 1	**10.98** 1	**1.69** 1	**2.86** 1	**1.05** 1	8.73 2	31.46 2	8.15 2	**5.85** 1	10.12 2	8.80 2
Black and Anandan	2.1	**9.26** 1	10.11 3	**12.08** 1	11.20 3	19.83 3	17.01 3	7.67 3	18.44 3	16.80 4	7.89 2	13.55 2	13.96 4	2.65 2	4.18 2	1.88 2	**7.64** 1	**30.13** 1	**7.31** 1	7.05 2	**10.02** 1	**8.41** 1
Pyramid LK	2.8	10.27 2	9.71 2	13.63 2	**9.46** 1	**18.62** 1	**12.07** 1	6.53 2	18.43 2	**10.95** 1	8.14 3	15.08 3	12.78 2	5.22 3	6.64 3	4.29 3	14.61 4	36.18 4	24.67 5	12.96 5	13.85 4	20.61 5
MediaPlayer™	4.1	15.82 4	26.42 4	16.96 4	23.18 4	27.71 5	21.78 4	9.44 4	22.25 4	15.03 3	10.99 4	18.15 5	13.64 3	11.09 4	17.16 4	10.66 5	15.46 5	43.56 5	15.09 4	9.98 4	15.04 5	9.47 3
Zitnick et al.	4.2	30.10 5	34.27 5	31.58 5	29.07 5	27.55 4	21.78 4	12.38 5	23.93 5	17.59 5	12.55 5	15.56 4	17.35 5	18.50 5	28.00 5	9.41 4	11.42 3	31.46 2	11.12 3	9.88 3	12.83 3	11.28 4

Color encoding of flow vectors

Flow image



Error image

Slide by Richard Szeliski

Conclusions about optical flow

- Difficulty:
  - substantially more challenging than previous datasets
- Diversity:
  - Substantial variation in difficulty across the datasets
- Motion GT vs Interpolation:
  - Best algorithms for one are not the best for the other
- Comparison with Stereo:
  - Performance of existing flow algorithms appears weak

Slide by Richard Szeliski

Stereo matching

Stereo matching

- Given two or more images of the same scene or object, compute a representation of its shape
- What are some possible representations?
  - depth maps
  - volumetric models
  - 3D surface models
  - planar (or offset) layers

Slide by Richard Szeliski

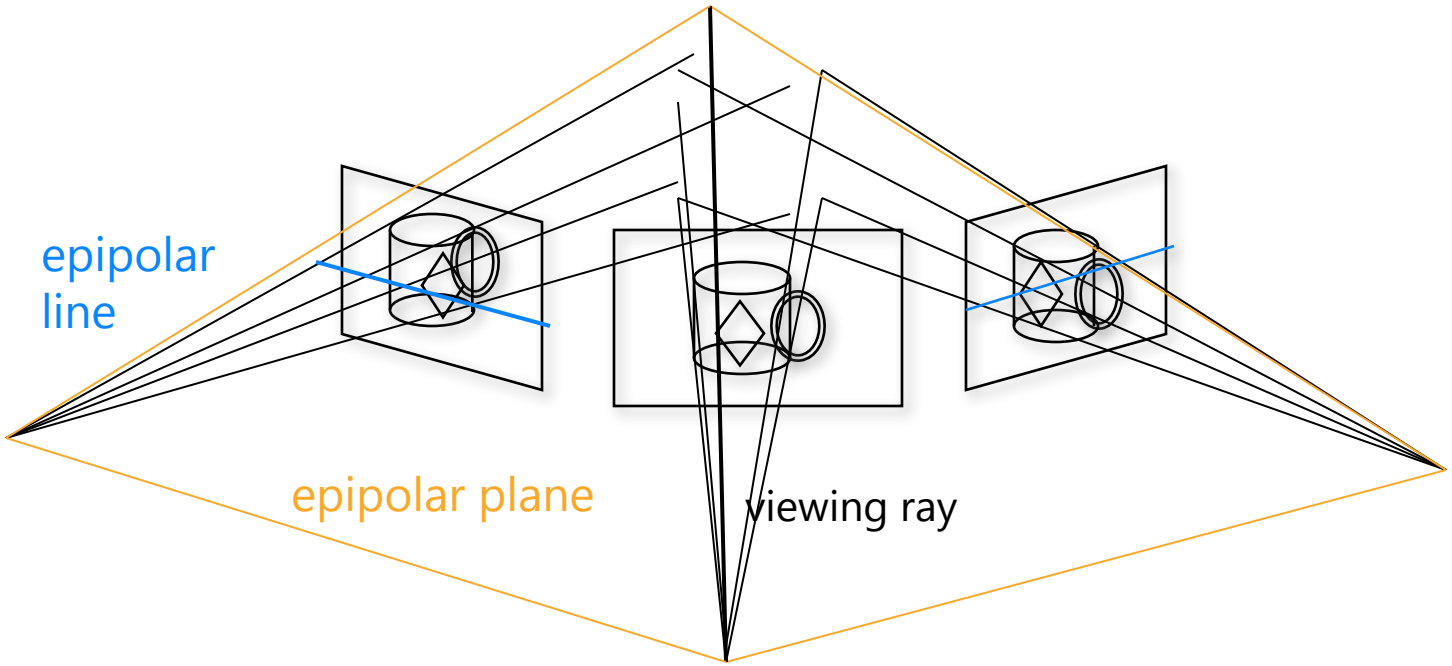
Stereo matching

- What are some possible algorithms?
  - match “features” and interpolate
  - match edges and interpolate
  - match all pixels with windows (coarse-fine)
  - use optimization:
    - iterative updating
    - dynamic programming
    - energy minimization (regularization, stochastic)
    - graph algorithms

Slide by Richard Szeliski

Stereo: epipolar geometry

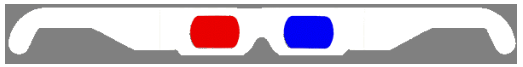
- Match features along epipolar lines



Slide by Richard Szeliski

Stereo image pair

Anaglyphs



http://www.rainbowsymphony.com/freestuff.html

(Wikipedia for images)

Public Library, Stereoscopic Looking Room, Chicago, by Phillips, 1923

Slide by Richard Szeliski

Stereo: epipolar geometry

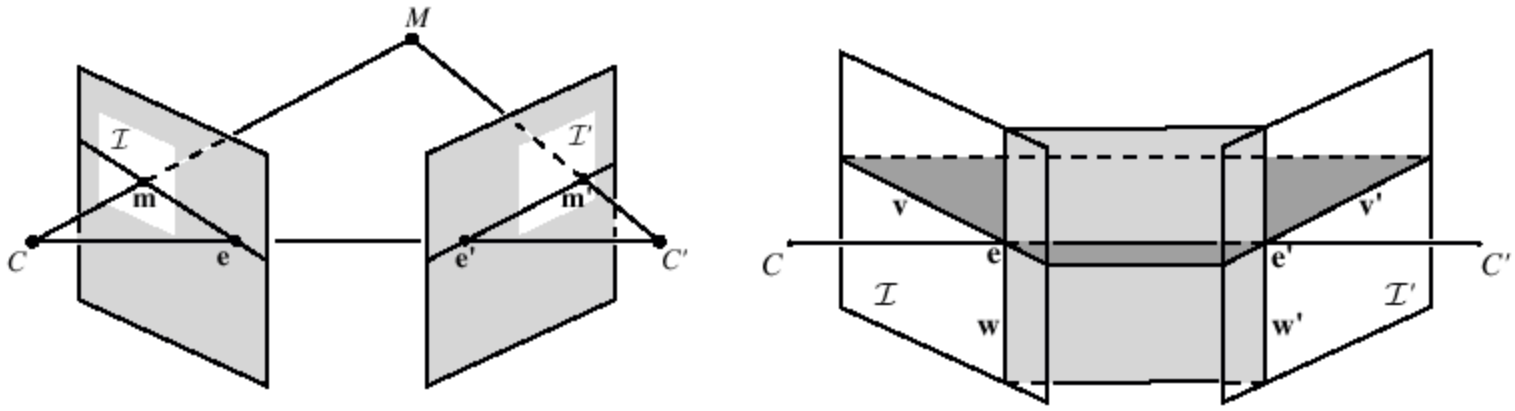
- for two images (or images with collinear camera centers), can find epipolar lines
- epipolar lines are the projection of the pencil of planes passing through the centers

- **Rectification:**
  warping the input images (perspective transformation)
  so that epipolar lines are horizontal

Slide by Richard Szeliski

Rectification

- Project each image onto same plane, which is parallel to the epipole
- Resample lines (and shear/stretch) to place lines in correspondence, and minimize distortion

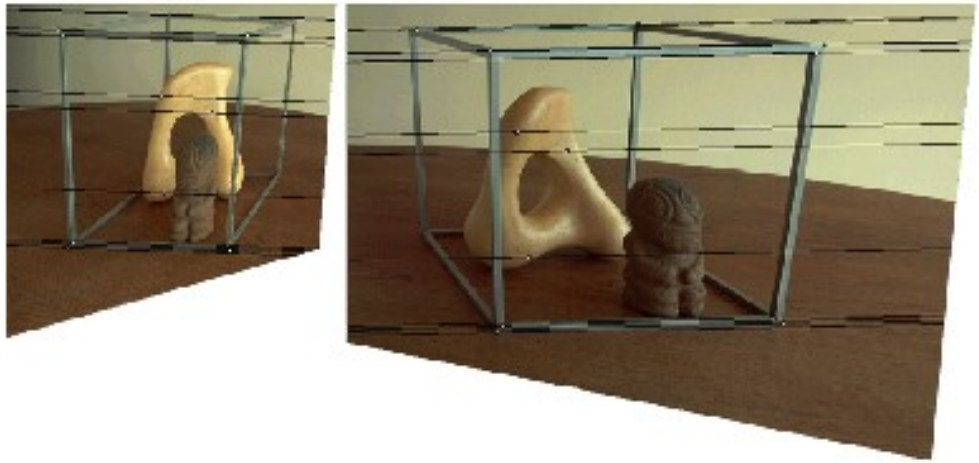


- [Loop and Zhang, CVPR'99]

Slide by Richard Szeliski

Rectification

(a) Original image pair overlayed with several epipolar lines.

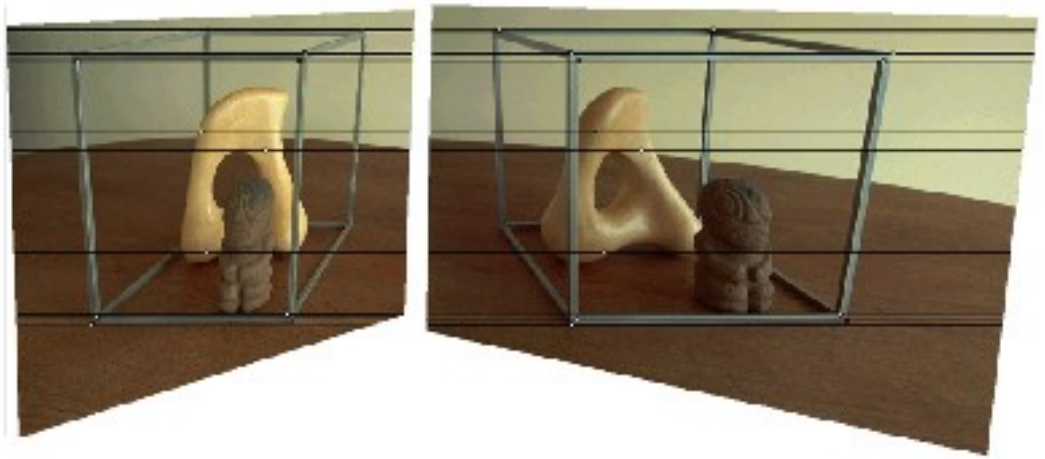


(b) Image pair transformed by the specialized projective mapping $\mathbf{H}_p$ and $\mathbf{H}'_p$. Note that the epipolar lines are now parallel to each other in each image.

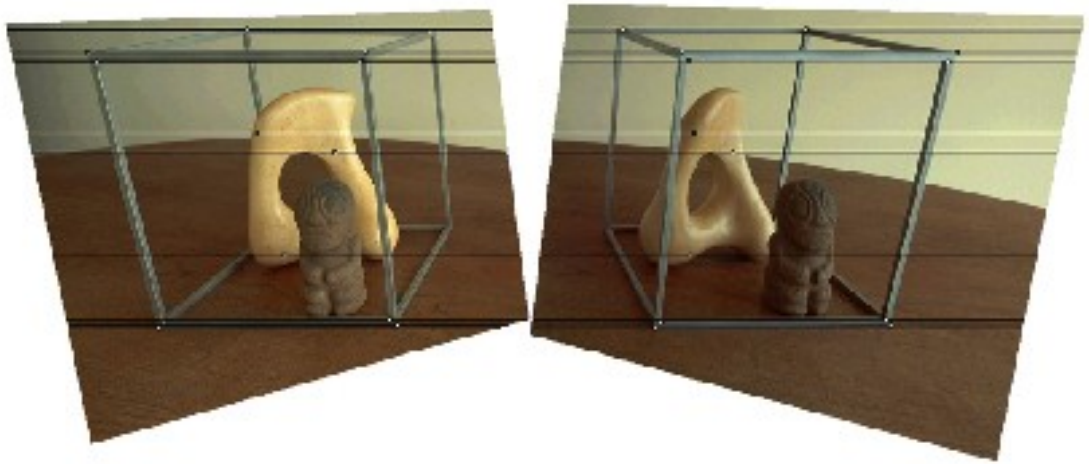
BAD!

Slide by Richard Szeliski

Rectification



(c) Image pair transformed by the similarity $\mathbf{H}_r$ and $\mathbf{H}'_r$. Note that the image pair is now rectified (the epipolar lines are horizontally aligned).



(d) Final image rectification after shearing transform $\mathbf{H}_s$ and $\mathbf{H}'_s$. Note that the image pair remains rectified, but the horizontal distortion is reduced.

GOOD!

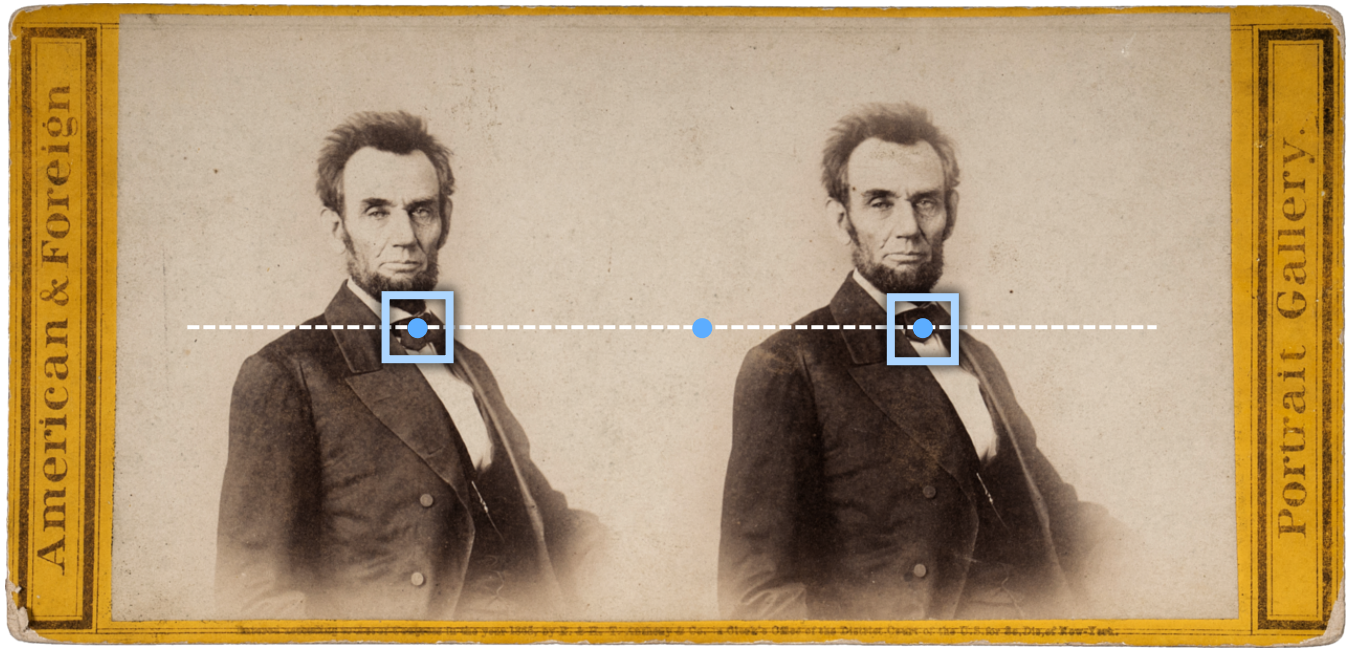
Slide by Richard Szeliski

Finding correspondences

- apply feature matching criterion (e.g., correlation or Lucas-Kanade) at all pixels simultaneously
- search only over epipolar lines (many fewer candidate positions)

Slide by Richard Szeliski

Your basic stereo algorithm



For each epipolar line

For each pixel in the left image

- compare with every pixel on same epipolar line in right image
- pick pixel with minimum match cost

Improvement: match **windows**

- This should look familiar …

Slide by Richard Szeliski

Image registration

- How do we determine correspondences?
  - block matching or SSD (sum squared differences)

$$E(x, y; d) = \sum_{(x',y') \in N(x,y)} [I_L(x'+d, y') - I_R(x', y')]^2$$

d is the disparity (horizontal motion)

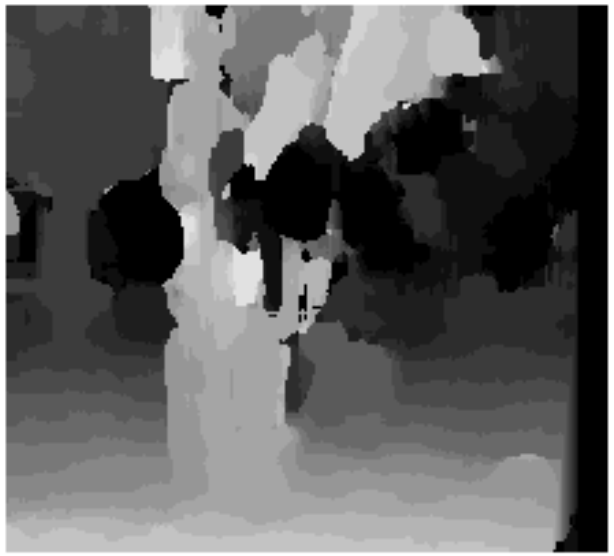
- How big should the neighborhood be?

Slide by Richard Szeliski

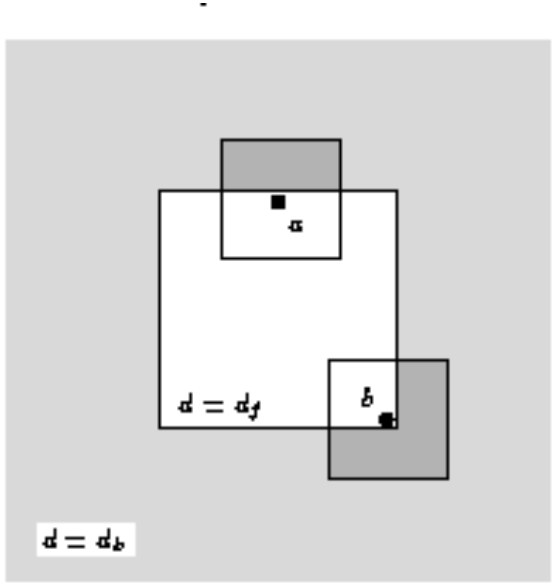
Neighborhood size

- Smaller neighborhood: more details
- Larger neighborhood: fewer isolated mistakes

w = 3



w = 20



Slide by Richard Szeliski

Matching criteria

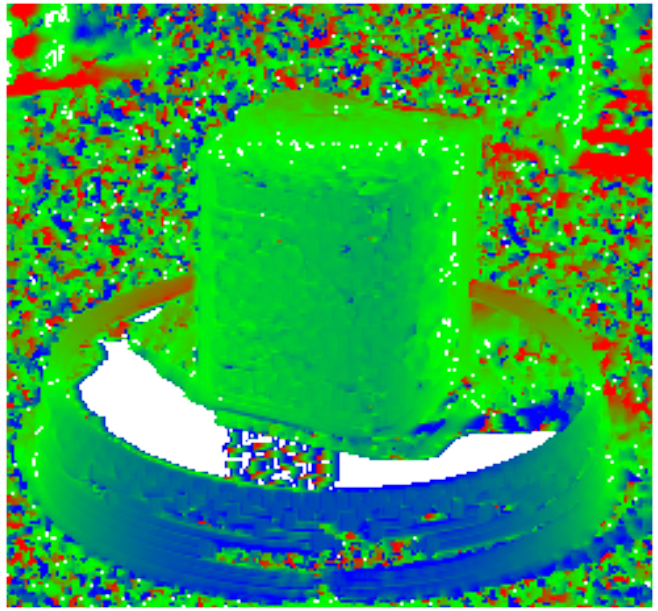
- Raw pixel values (correlation)
- Band-pass filtered images [Jones & Malik 92]
- “Corner” like features [Zhang, …]
- Edges [many people…]
- Gradients [Seitz 89; Scharstein 94]
- Rank statistics [Zabih & Woodfill 94]

Slide by Richard Szeliski

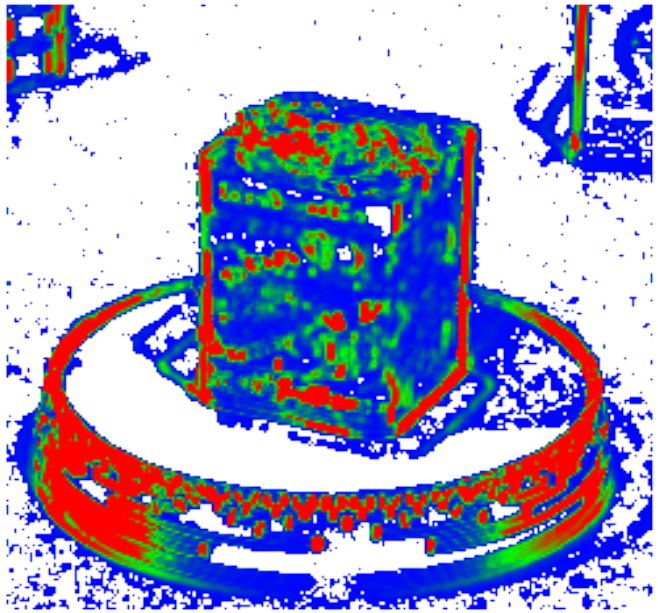
Stereo: certainty modeling

- Compute certainty map from correlations

input



depth map

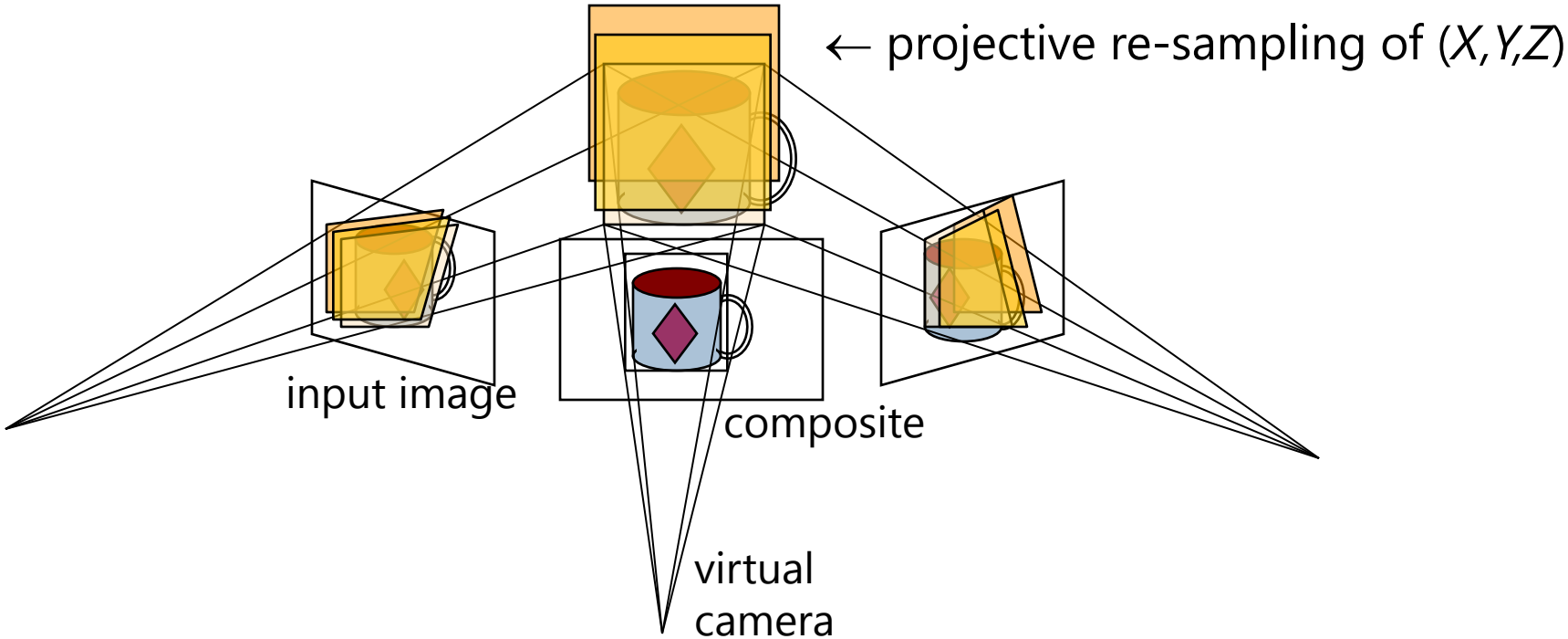


certainty map

Slide by Richard Szeliski

Plane sweep stereo

- Sweep family of planes through volume

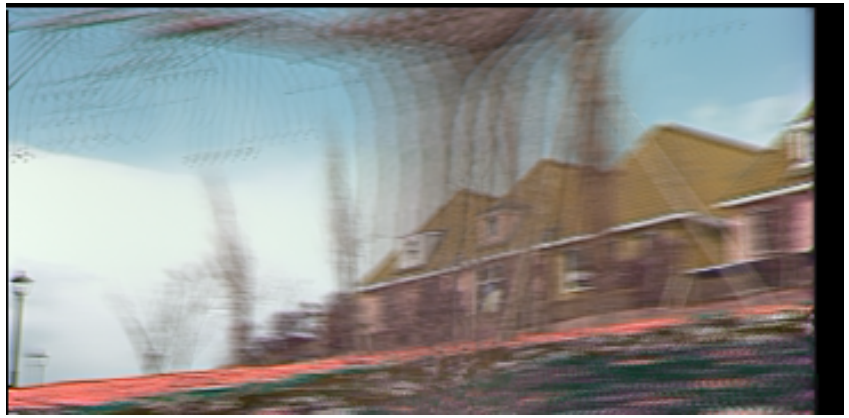


- each plane defines an image ⇒ composite homography

Slide by Richard Szeliski

Plane sweep stereo

- For each depth plane
  - compute composite (mosaic) image — mean

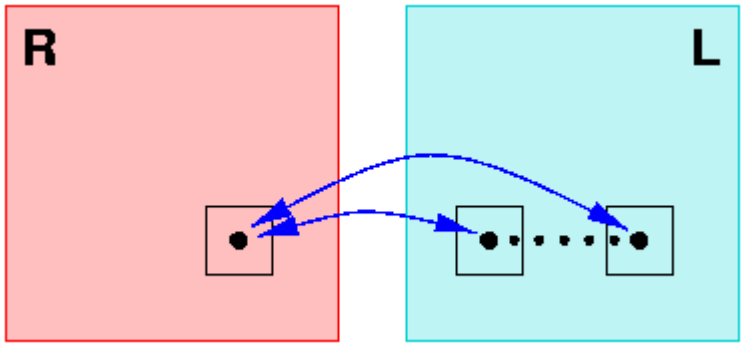


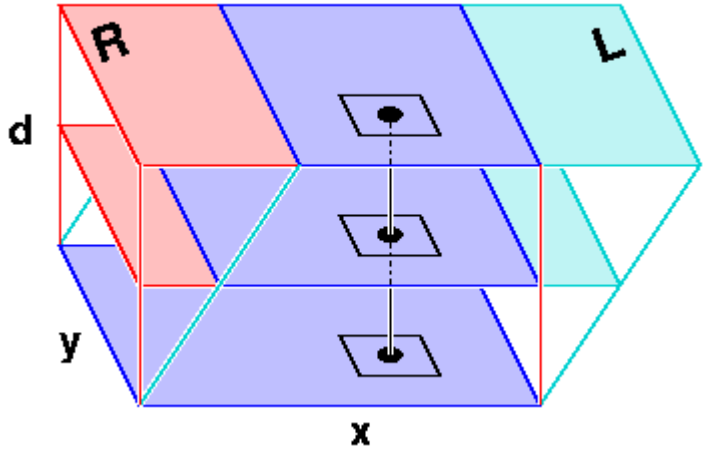
  - compute error image — variance
  - convert to confidence and aggregate spatially
- Select winning depth at each pixel

Slide by Richard Szeliski

Plane sweep stereo

- Re-order (pixel / disparity) evaluation loops





for every pixel,
  for every disparity
    compute cost

for every disparity
  for every pixel
    compute cost

Slide by Richard Szeliski

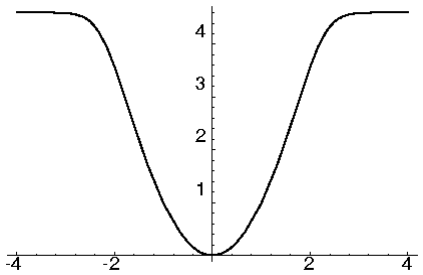
Stereo matching framework

1. For every disparity, compute raw matching costs

$$E_0(x, y; d) = \rho(I_L(x' + d, y') - I_R(x', y'))$$

Why use a robust function?

- occlusions, other outliers



- Can also use alternative match criteria

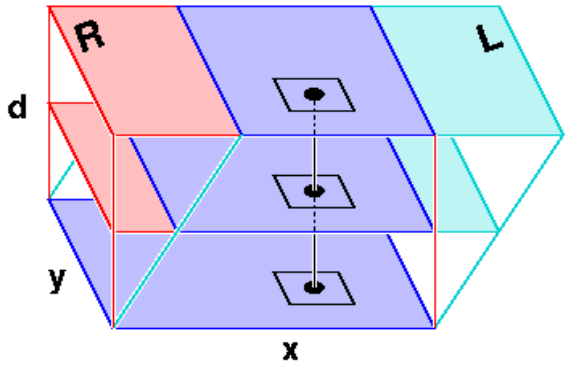
Slide by Richard Szeliski

Stereo matching framework

2. Aggregate costs spatially

$$E(x, y; d) = \sum_{(x',y') \in N(x,y)} E_0(x', y', d)$$

- Here, we are using a box filter (efficient moving average implementation)
- Can also use weighted average, [non-linear] diffusion…



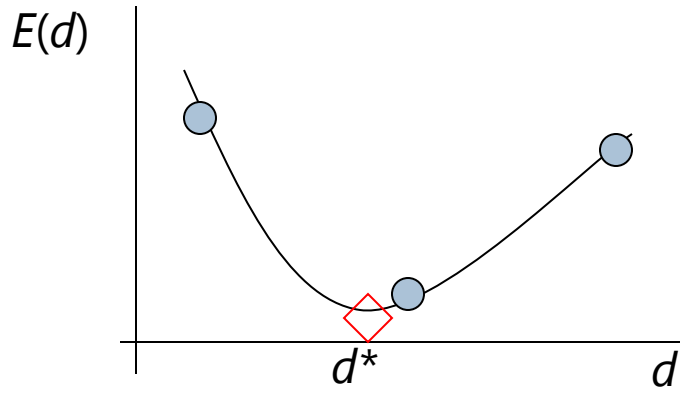
Slide by Richard Szeliski

Stereo matching framework

3. Choose winning disparity at each pixel

$$d(x, y) = \arg\min_{d} E(x, y; d)$$

4. Interpolate to sub-pixel accuracy



Slide by Richard Szeliski

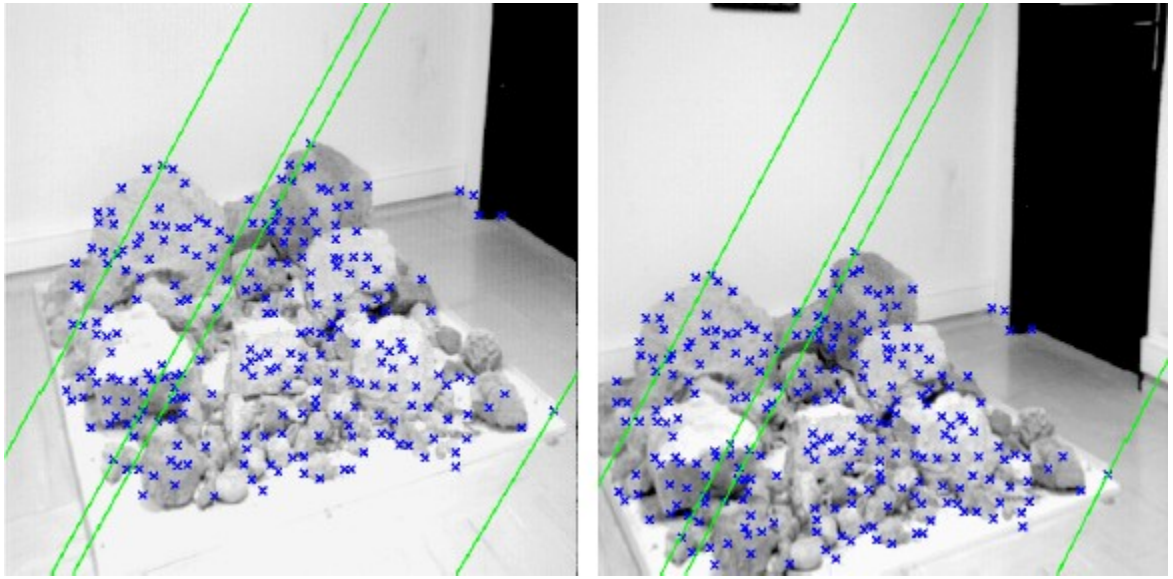
Traditional stereo matching

- Advantages:
  - gives detailed surface estimates
  - fast algorithms based on moving averages
  - sub-pixel disparity estimates and confidence
- Limitations:
  - narrow baseline $\Rightarrow$ noisy estimates
  - fails in textureless areas
  - gets confused near occlusion boundaries

Slide by Richard Szeliski

Feature-based stereo

- Match “corner” (interest) points



- Interpolate complete solution

Slide by Richard Szeliski

Data interpolation

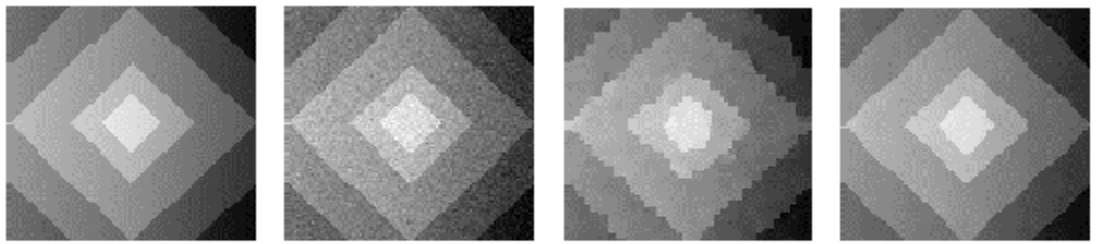
- Given a sparse set of 3D points, how do we interpolate to a full 3D surface?
  - Scattered data interpolation [Nielson93]
  - triangulate
  - put onto a grid and fill (use pyramid?)
  - place a kernel function over each data point
  - minimize an energy function

Slide by Richard Szeliski

Graph cuts

- Solution technique for general 2D problem

$$
\begin{aligned}
E_{\text{total}}(\mathbf{d}) &= E_{\text{data}}(\mathbf{d}) + \lambda E_{\text{smoothness}}(\mathbf{d}) \\
E_{\text{data}}(\mathbf{d}) &= \sum_{x,y} f_{x,y}(d_{x,y}) \\
E_{\text{smoothness}}(\mathbf{d}) &= \sum_{x,y} \rho(d_{x,y} - d_{x-1,y}) \\
&\quad + \sum_{x,y} \rho(d_{x,y} - d_{x,y-1})
\end{aligned}
$$



(a) original image (b) observed image (c) local min w.r.t. standard moves (d) local min w.r.t. α-expansion moves

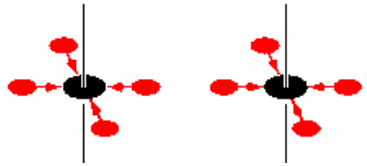
Slide by Richard Szeliski

Markov random field (MRF)

- Probability distribution on disparity field $d(x, y)$

$$p_P(d_{x,y}|\mathbf{d}) = p_P(d_{x,y}|\{d_{x',y'}, (x', y') \in \mathcal{N}(x, y)\})$$

$$p_P(\mathbf{d}) = \frac{1}{Z_P} e^{-E_P(\mathbf{d})}$$



$$E_P(\mathbf{d}) = \sum_{x,y} \rho_P(d_{x+1,y} - d_{x,y}) + \rho_P(d_{x,y+1} - d_{x,y})$$

- Enforces *smoothness* or coherence on field

Slide by Richard Szeliski

MAP estimate

- Maximize posterior likelihood

$$
\begin{aligned}
E(\mathbf{d}) &= -\log p(\mathbf{d}|I_L, I_R) \\
&= \sum_{x,y} \rho_P(d_{x+1,y} - d_{x,y}) + \rho_P(d_{x,y+1} - d_{x,y}) \\
&\quad + \sum_{x,y} \rho_M(I_L(x + d_{x,y}, y) - I_R(x, y))
\end{aligned}
$$

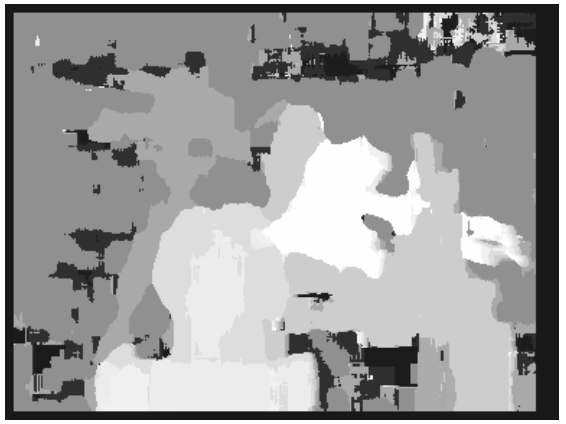
- Equivalent to *regularization* (energy minimization with smoothness constraints)

Slide by Richard Szeliski

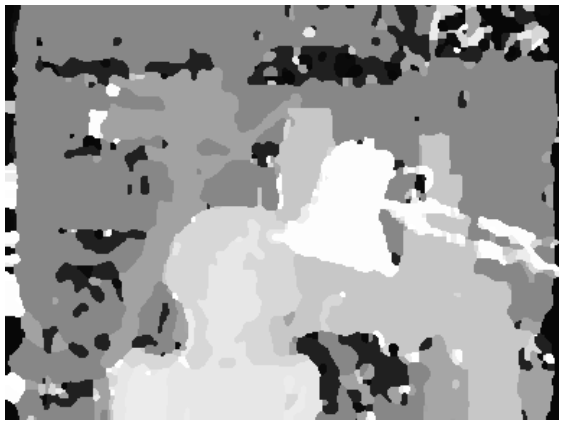
Depth map results

Input image

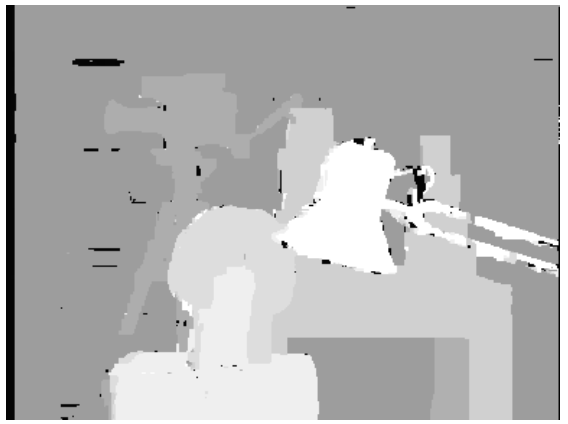
Sum of abs. diff. (SAD)



Mean field



Graph cuts



Slide by Richard Szeliski

Middlebury stereo benchmark

Error Threshold = 1 — Error Threshold... — Sort by nonocc — Sort by all — Sort by disc

Algorithm	Avg. Rank	Tsukuba ground truth nonocc	Tsukuba ground truth all	Tsukuba ground truth disc	Venus ground truth nonocc	Venus ground truth all	Venus ground truth disc	Teddy ground truth nonocc	Teddy ground truth all	Teddy ground truth disc	Cones ground truth nonocc	Cones ground truth all	Cones ground truth disc
AdaptingBP [17]	2.8	1.11 6	1.37 3	5.79 7	**0.10** 1	0.21 2	**1.44** 1	4.22 4	7.06 2	11.8 4	**2.48** 1	7.92 2	**7.32** 1
DoubleBP2 [35]	2.9	**0.88** 1	**1.29** 1	**4.76** 1	0.13 3	0.45 5	1.87 5	3.53 2	8.30 3	**9.63** 1	2.90 3	8.78 8	7.79 2
DoubleBP [15]	4.9	0.88 2	1.29 2	4.76 2	0.14 5	0.60 13	2.00 7	3.55 3	8.71 5	9.70 2	2.90 4	9.24 11	7.80 3
SubPixDoubleBP [30]	5.6	1.24 10	1.76 13	5.98 8	0.12 2	0.46 6	1.74 4	**3.45** 1	8.38 4	10.0 3	2.93 5	8.73 7	7.91 4
AdaptOvrSegBP [33]	9.9	1.69 22	2.04 21	5.64 6	0.14 4	**0.20** 1	1.47 2	7.04 14	11.1 7	16.4 11	3.60 11	8.96 10	8.84 10
SymBP+occ [7]	10.8	0.97 4	1.75 12	5.09 4	0.16 6	0.33 3	2.19 8	6.47 8	10.7 6	17.0 14	4.79 24	10.7 21	10.9 20
PlaneFitBP [32]	10.8	0.97 5	1.83 14	5.26 5	0.17 7	0.51 8	1.71 3	6.65 9	12.1 13	14.7 7	4.17 20	10.7 20	10.6 19
AdaptDispCalib [36]	11.8	1.19 8	1.42 4	6.15 9	0.23 9	0.34 4	2.50 11	7.80 19	13.6 21	17.3 17	3.62 12	9.33 12	9.72 15
Segm+visib [4]	12.2	1.30 15	1.57 5	6.92 18	0.79 21	1.06 18	6.76 22	5.00 5	**6.54** 1	12.3 5	3.72 13	8.62 6	10.2 17
C-SemiGlob [19]	12.3	2.61 29	3.29 24	9.89 27	0.25 12	0.57 10	3.24 15	5.14 6	11.8 8	13.0 6	2.77 2	8.35 4	8.20 5
SO+borders [29]	12.8	1.29 14	1.71 9	6.83 15	0.25 13	0.53 9	2.26 9	7.02 13	12.2 14	16.3 9	3.90 15	9.85 16	10.2 18
DistinctSM [27]	14.1	1.21 9	1.75 11	6.39 11	0.35 14	0.69 16	2.63 13	7.45 18	13.0 17	18.1 19	3.91 16	9.91 18	8.32 7
CostAggr+occ [39]	14.3	1.38 17	1.96 17	7.14 19	0.44 16	1.13 19	4.87 19	6.80 11	11.9 10	17.3 16	3.60 10	8.57 5	9.36 13
OverSegmBP [26]	14.5	1.69 23	1.97 18	8.47 24	0.51 18	0.68 15	4.69 18	6.74 10	11.9 12	15.8 8	3.19 8	8.81 9	8.89 11
SegmentSupport [28]	15.1	1.25 11	1.62 7	6.68 13	0.25 11	0.64 14	2.59 12	8.43 24	14.2 22	18.2 20	3.77 14	9.87 17	9.77 16
RegionTreeDP [18]	15.7	1.39 19	1.64 8	6.85 16	0.22 8	0.57 10	1.93 6	7.42 17	11.9 11	16.8 13	6.31 30	11.9 27	11.8 23
EnhancedBP [24]	16.6	0.94 3	1.74 10	5.05 3	0.35 15	0.86 17	4.34 17	8.11 22	13.3 19	18.5 22	5.09 27	11.1 23	11.0 21

Slide by Richard Szeliski

Traditional stereo

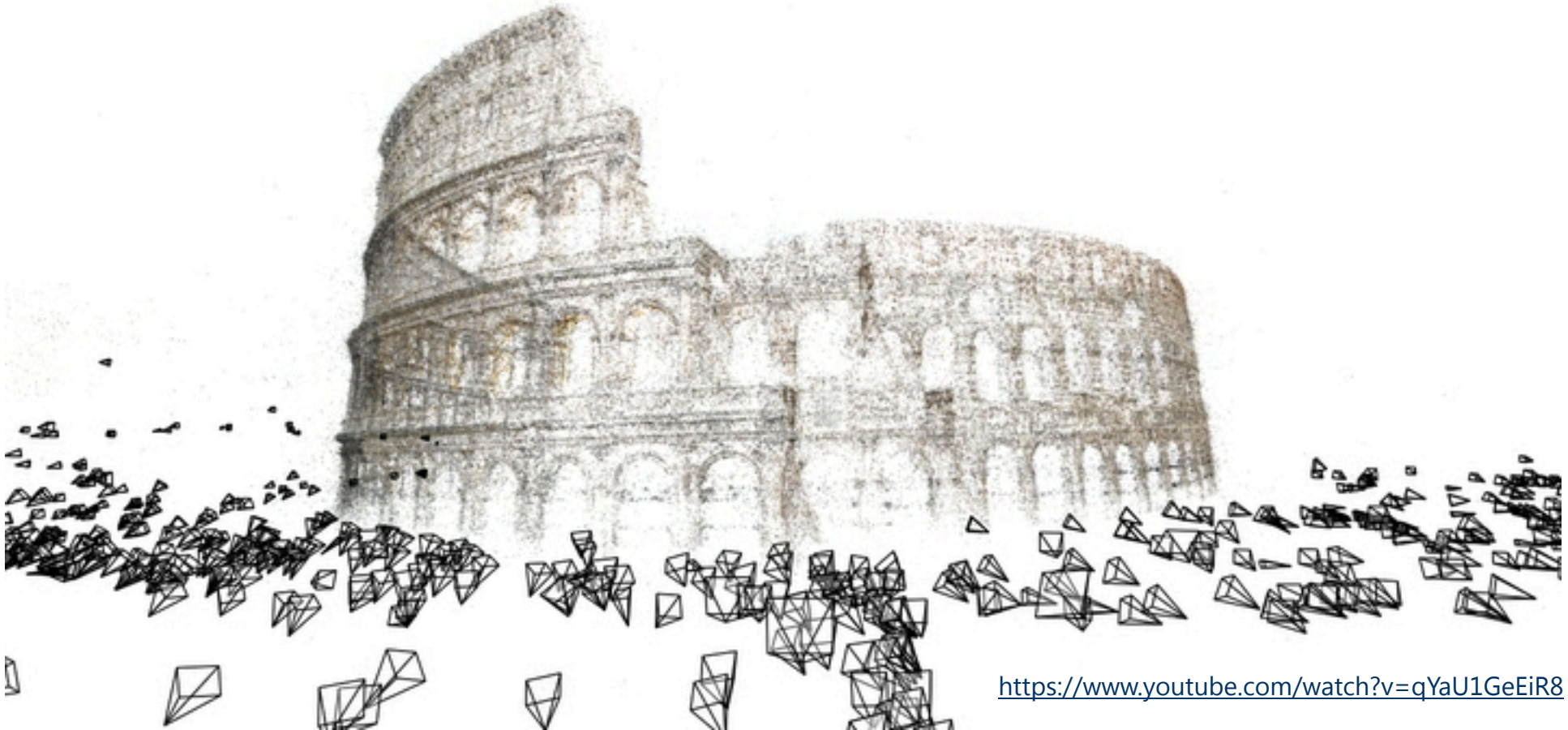
- Advantages:
  - works very well in non-occluded regions
- Disadvantages:
  - restricted to two images (not)
  - gets confused in occluded regions
  - can't handle *mixed* pixels

Slide by Richard Szeliski

Slides by Szymon Rusinkiewicz and Jianxiong Xiao

Structure-from-motion & bundle adjustment

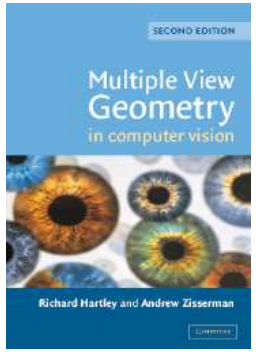
Building Rome in a Day



https://www.youtube.com/watch?v=qYaU1GeEiR8

Slide by Szymon Rusinkiewicz and Jianxiong Xiao

How?



Reading: the MVG bible
(need ~2 years)

5-6 years

Yes

Coding

PhD?

No

Crying: Not Working At All

Slide by Szymon Rusinkiewicz and Jianxiong 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 by Szymon Rusinkiewicz and Jianxiong 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 by Szymon Rusinkiewicz and Jianxiong 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 by Szymon Rusinkiewicz and Jianxiong Xiao

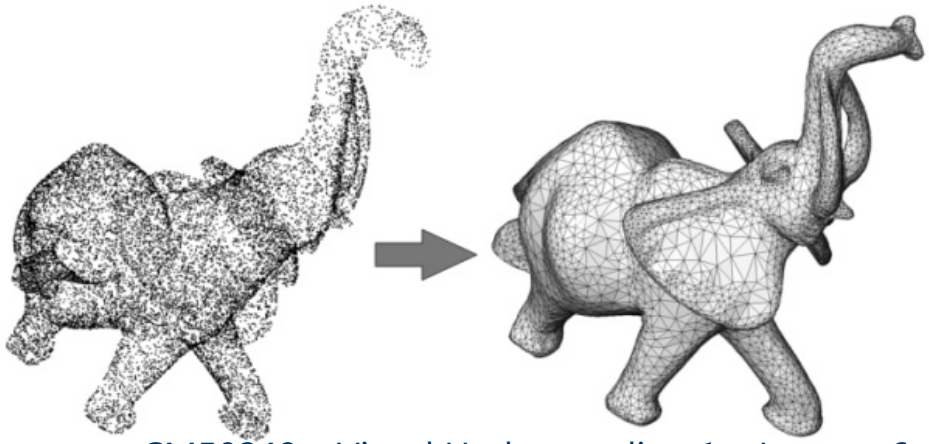
Steps

	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 by Szymon Rusinkiewicz and Jianxiong Xiao

Steps

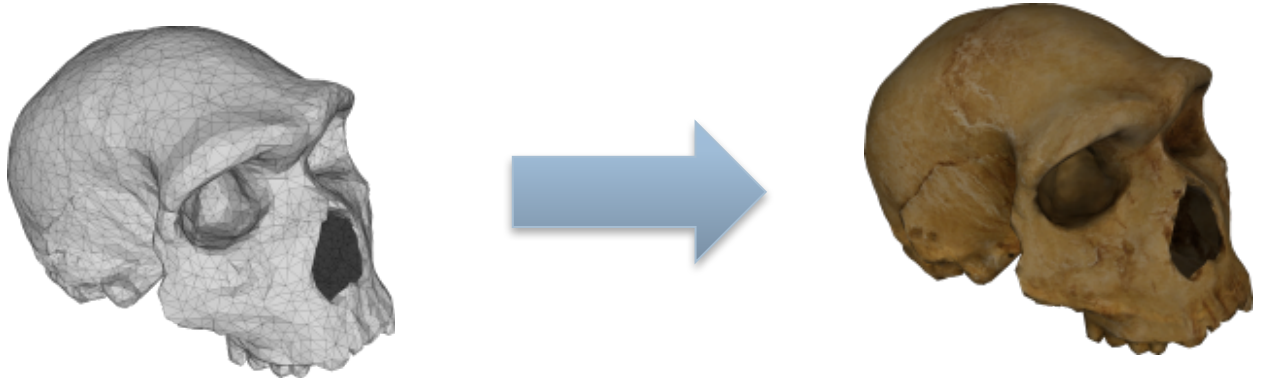
	Images	→ Points:	Structure from Motion
	Points	→ More points:	Multiple View Stereo
	Points	→ Meshes:	Model Fitting
+	Meshes	→ Models:	Texture Mapping
=	Images	→ Models:	Image-based Modeling



Slide by Szymon Rusinkiewicz and Jianxiong 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 by Szymon Rusinkiewicz and Jianxiong 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 by Szymon Rusinkiewicz and Jianxiong Xiao

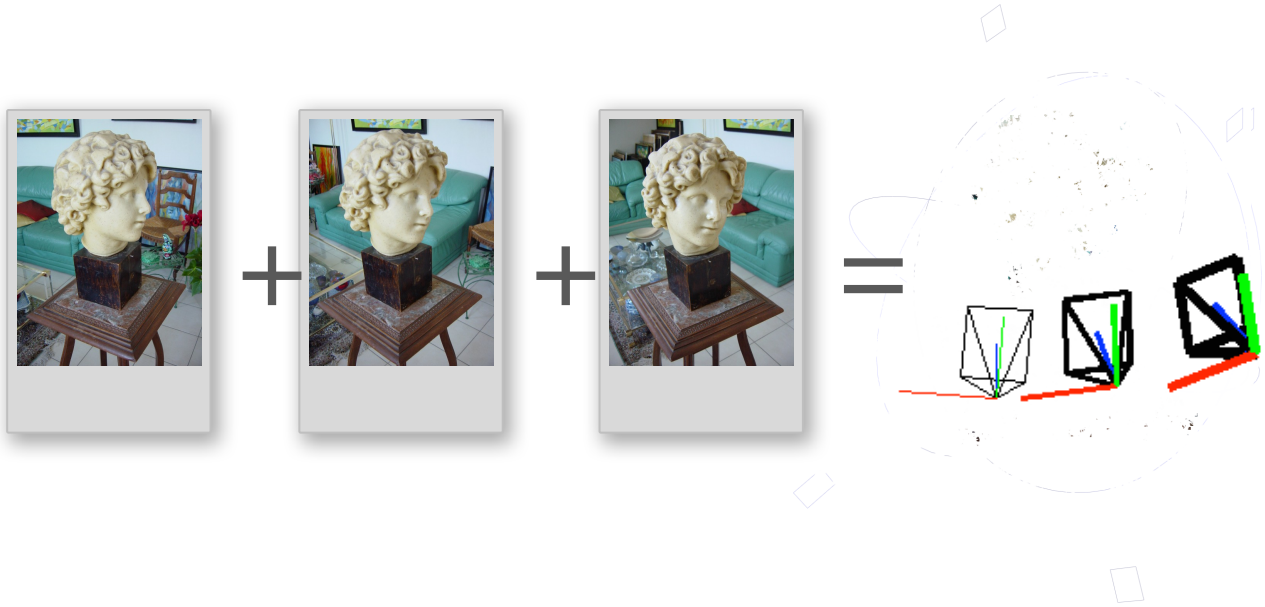
Steps

	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 by Szymon Rusinkiewicz and Jianxiong Xiao

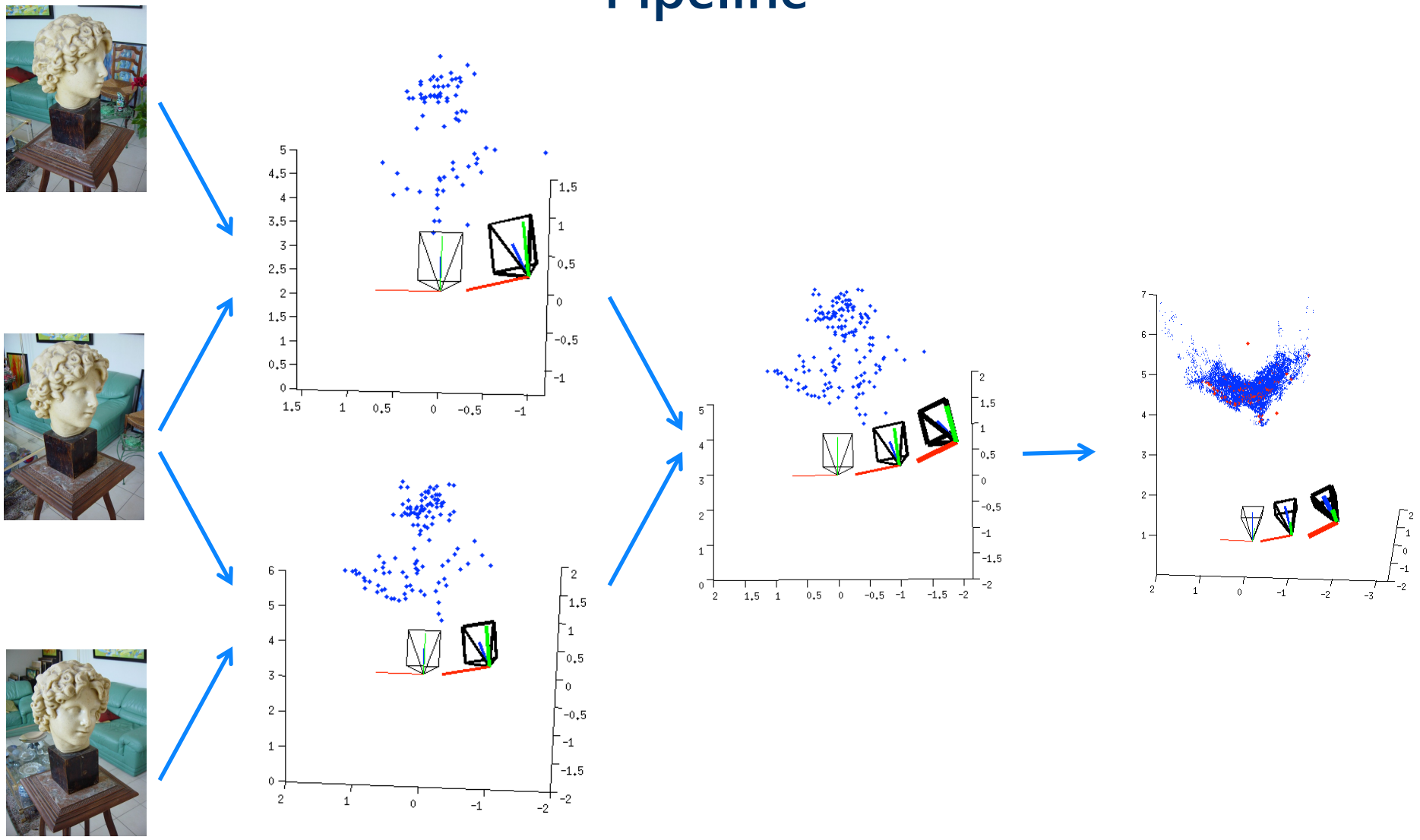
Structure from motion

- Structure = 3D point cloud of the scene
- Motion = camera location and orientation
- SFM = get the point cloud from moving cameras
- Structure and motion: joint problems to solve



Slide by Szymon Rusinkiewicz and Jianxiong Xiao

Pipeline

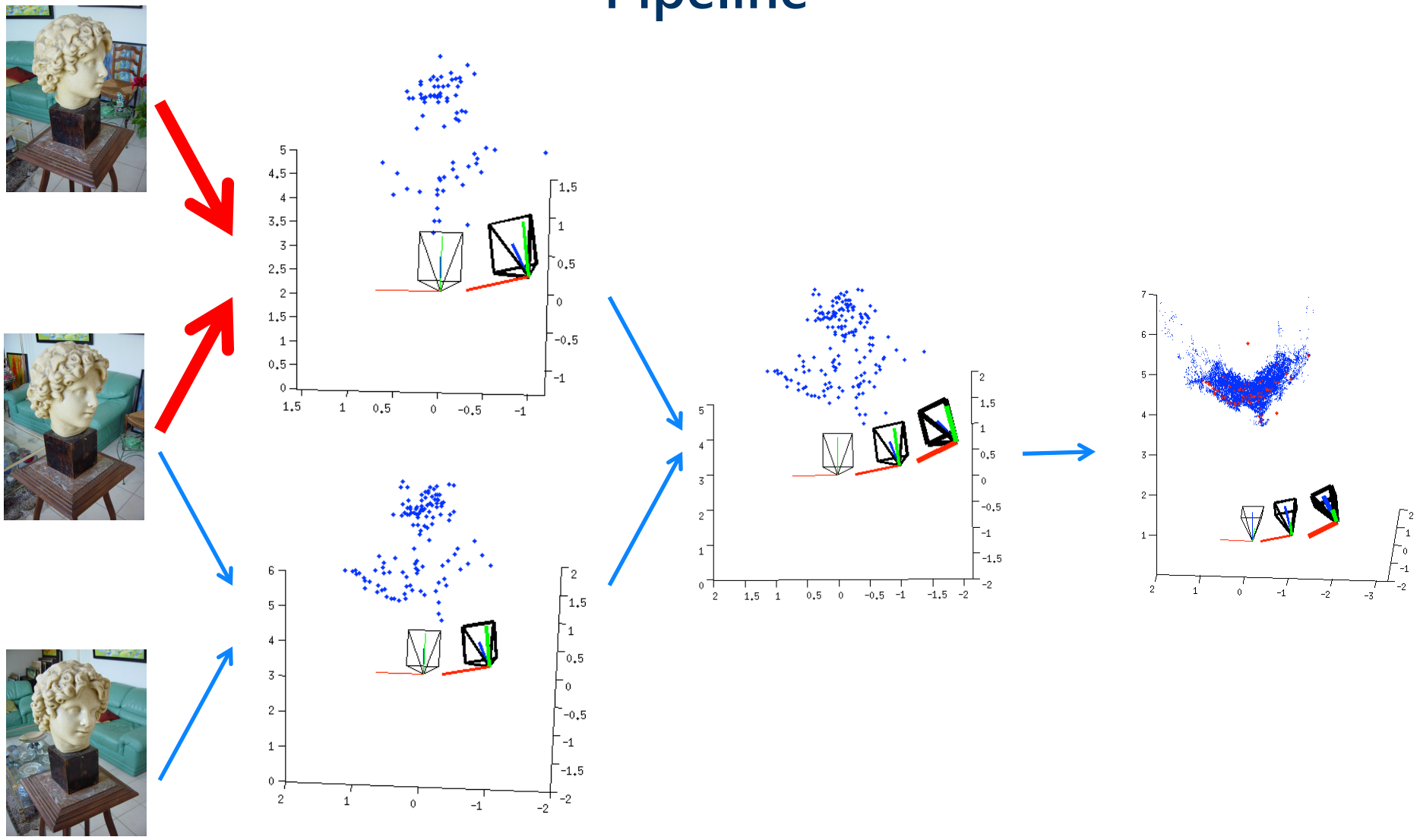


Structure from Motion (SFM)

Multi-view Stereo (MVS)

Slide by Szymon Rusinkiewicz and Jianxiong Xiao

Pipeline

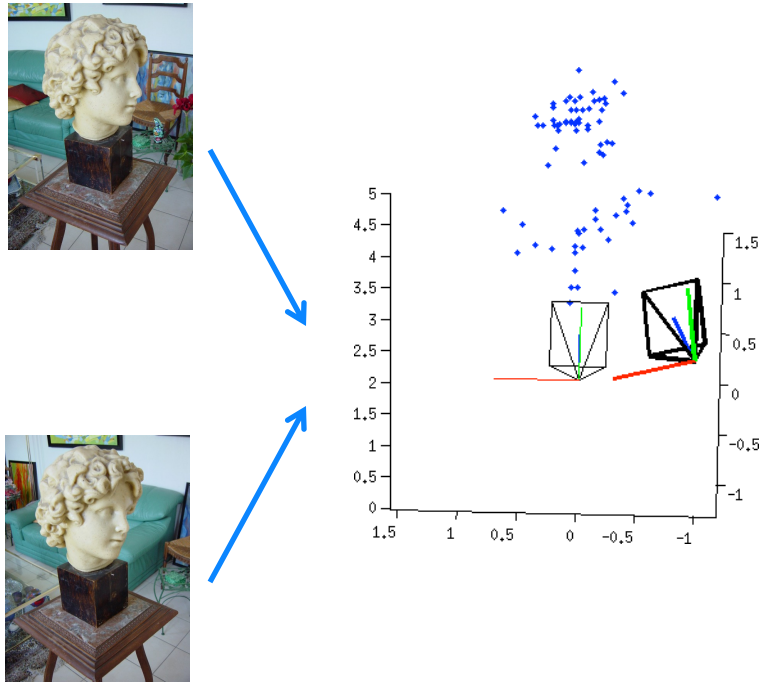


Structure from Motion (SFM)

Multi-view Stereo (MVS)

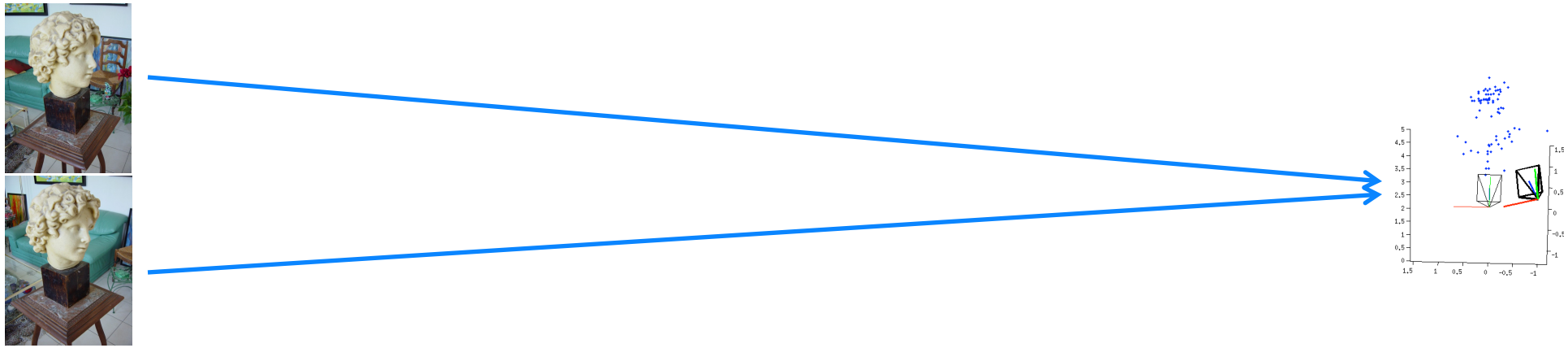
Slide by Szymon Rusinkiewicz and Jianxiong Xiao

Two-view reconstruction



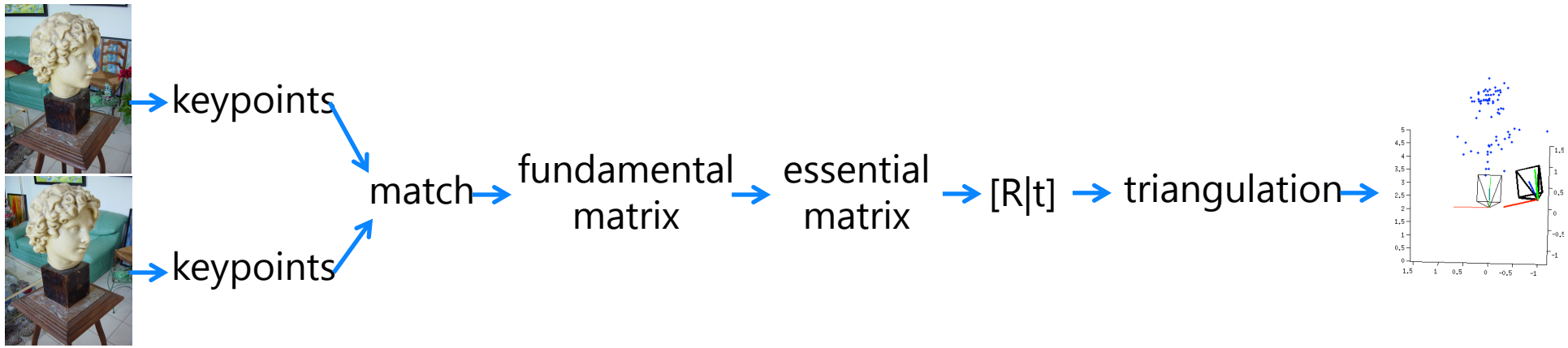
Slide by Szymon Rusinkiewicz and Jianxiong Xiao

Two-view reconstruction



Slide by Szymon Rusinkiewicz and Jianxiong Xiao

Two-view reconstruction



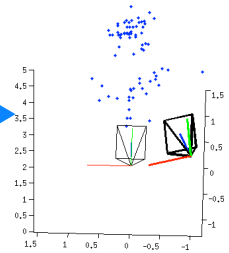
Slide by Szymon Rusinkiewicz and Jianxiong Xiao

Keypoint detection

keypoints

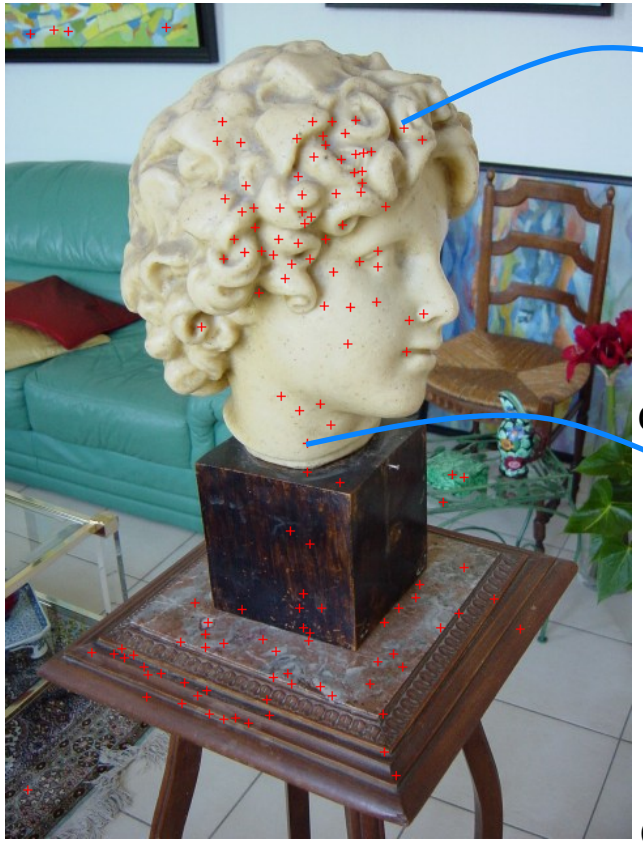
keypoints

match → fundamental matrix → essential matrix → [R|t] → triangulation



Slide by Szymon Rusinkiewicz and Jianxiong Xiao

Descriptor for each point



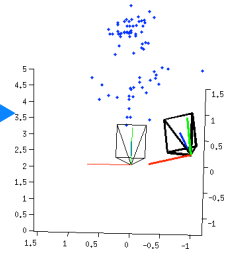
SIFT descriptor

SIFT descriptor

keypoints

keypoints

match → fundamental matrix → essential matrix → [R|t] → triangulation

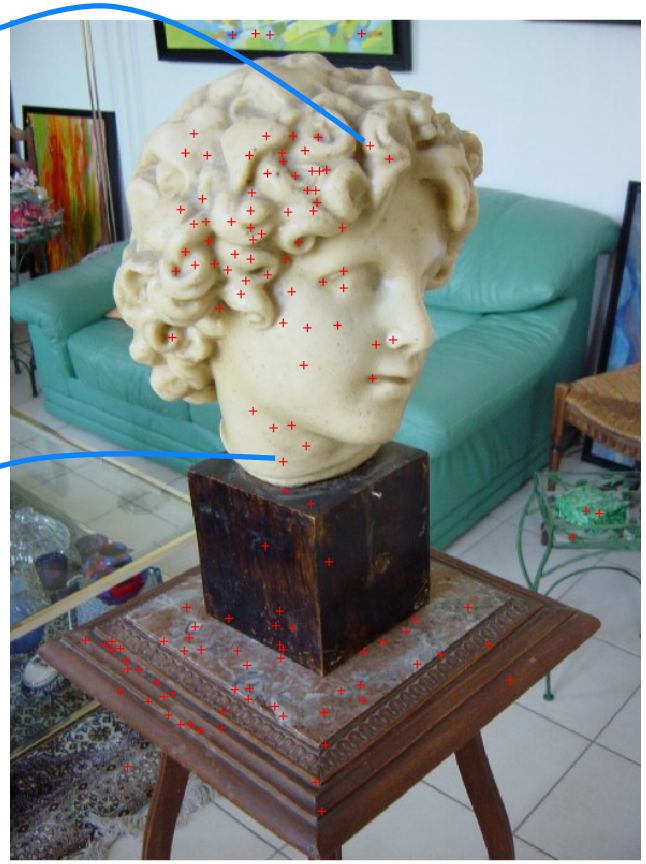


Slide by Szymon Rusinkiewicz and Jianxiong Xiao

Same for the other images

SIFT descriptor

SIFT descriptor



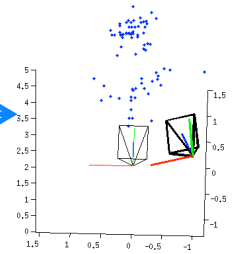
SIFT descriptor

SIFT descriptor

keypoints

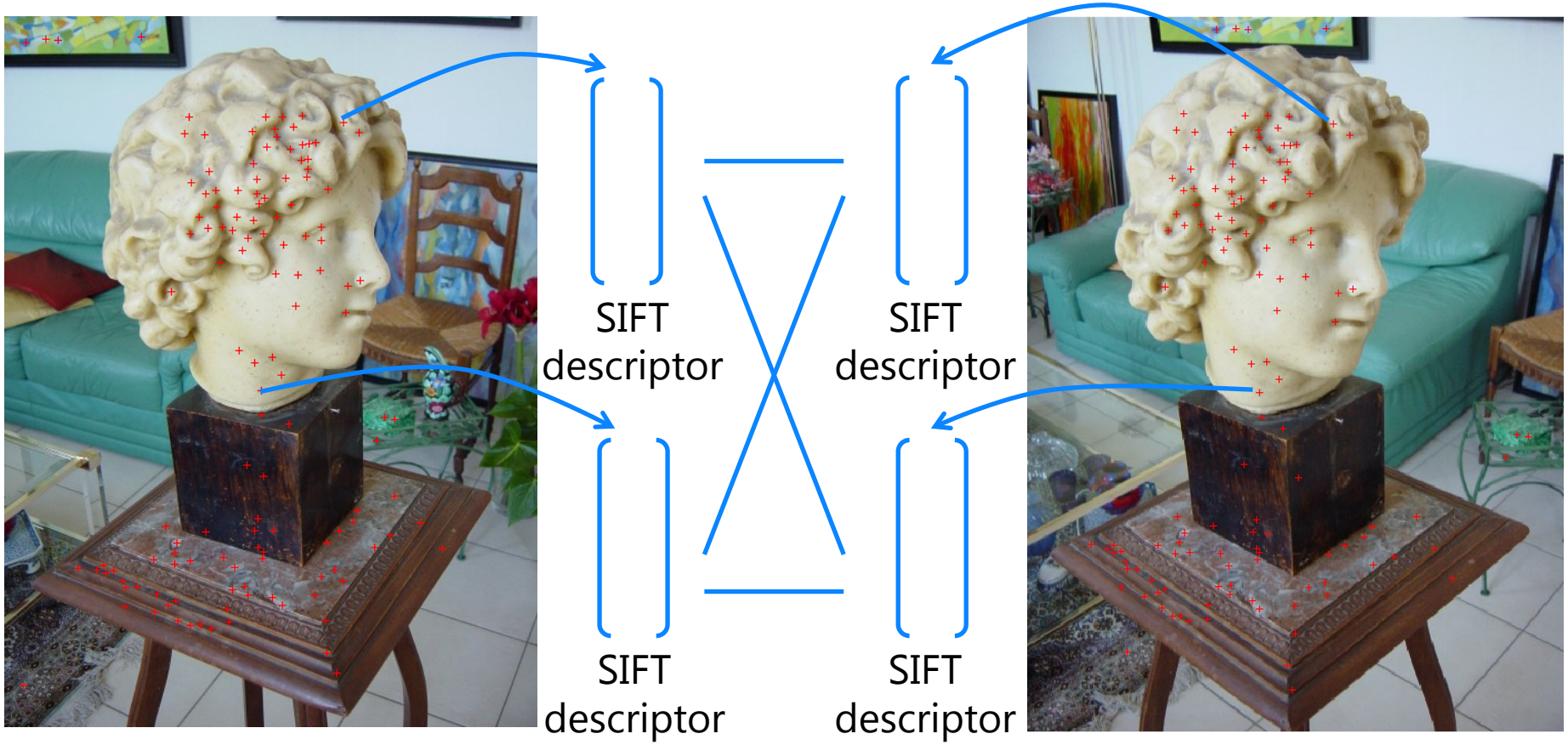
keypoints

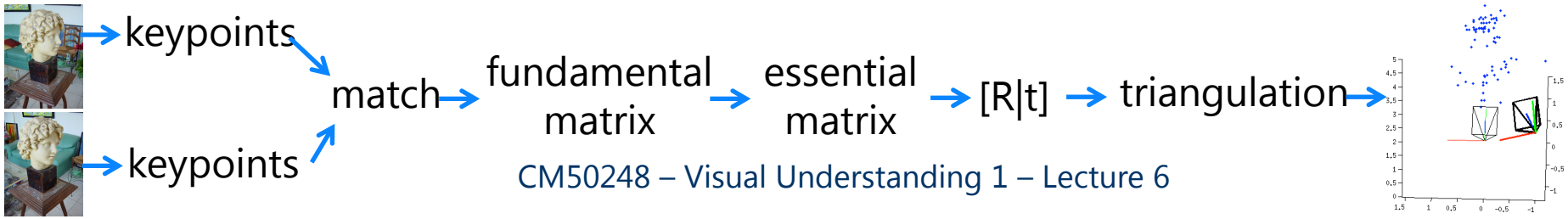
match → fundamental matrix → essential matrix → [R|t] → triangulation



Slide by Szymon Rusinkiewicz and Jianxiong Xiao

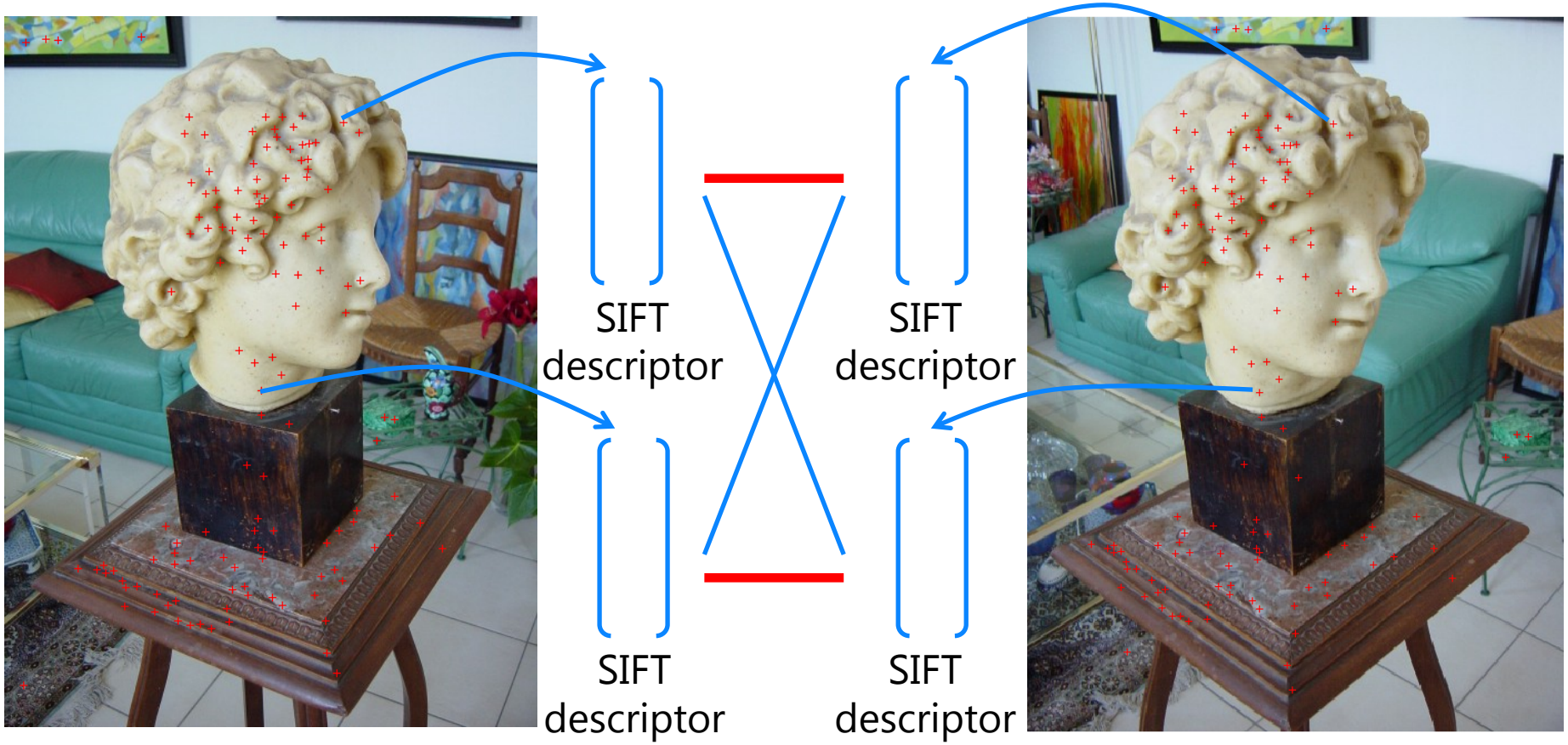
Point match for correspondences



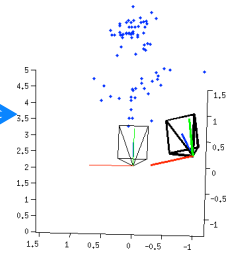


Slide by Szymon Rusinkiewicz and Jianxiong Xiao

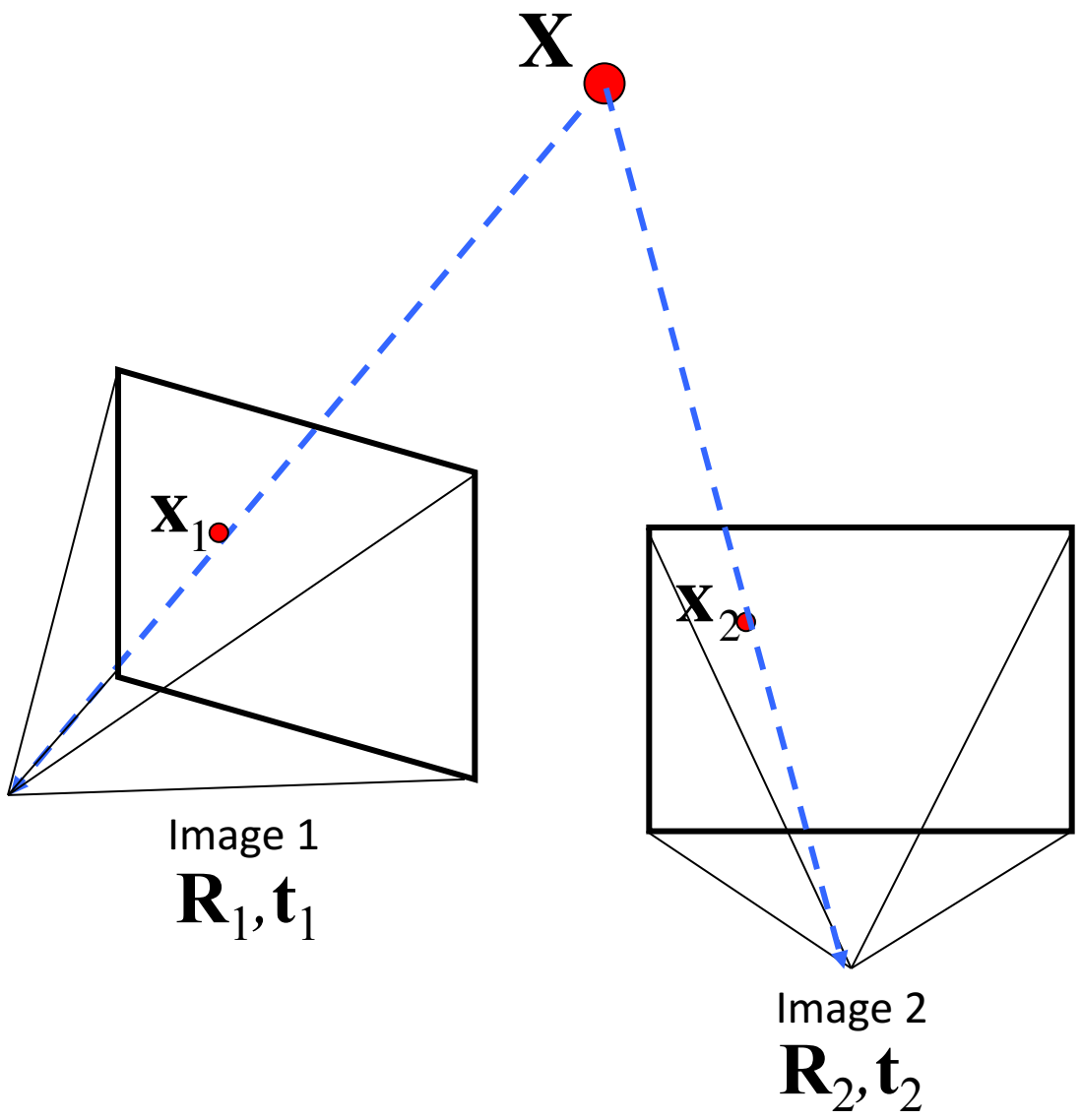
Point match for correspondences





keypoints
keypoints
match
fundamental matrix
essential matrix
[R|t]
triangulation


Fundamental Matrix



$$\mathbf{x}_1 \leftrightarrow \mathbf{x}_2$$

$$\mathbf{x}_1^T \mathbf{F} \mathbf{x}_2 = 0$$

Exercise

What is the difference between a fundamental matrix and a homography?

(Both of them are to explain 2D point to 2D point correspondences.)

Slide by Szymon Rusinkiewicz and Jianxiong Xiao

Estimating the fundamental matrix

- Given a correspondence
- The basic incidence relation is $\mathbf{x}_1 \leftrightarrow \mathbf{x}_2$

$$\mathbf{x}_1^T \mathbf{F} \mathbf{x}_2 = 0 \quad \Longleftrightarrow \quad [x_1x_2, x_1y_2, x_1, y_1x_2, y_1y_2, y_1, x_2, y_2, 1] \begin{bmatrix} f_{11} \\ f_{12} \\ f_{13} \\ f_{21} \\ f_{22} \\ f_{23} \\ f_{31} \\ f_{32} \\ f_{33} \end{bmatrix} = 0$$

Need 8 points

Slide by Szymon Rusinkiewicz and Jianxiong Xiao

Estimating the fundamental matrix

$\mathbf{x}_1^T \mathbf{F} \mathbf{x}_2 = 0$ for 8 point correspondences:

$$\mathbf{x}_1^1 \leftrightarrow \mathbf{x}_2^1, \mathbf{x}_1^2 \leftrightarrow \mathbf{x}_2^2, \mathbf{x}_1^3 \leftrightarrow \mathbf{x}_2^3, \mathbf{x}_1^4 \leftrightarrow \mathbf{x}_2^4, \mathbf{x}_1^5 \leftrightarrow \mathbf{x}_2^5, \mathbf{x}_1^6 \leftrightarrow \mathbf{x}_2^6, \mathbf{x}_1^7 \leftrightarrow \mathbf{x}_2^7, \mathbf{x}_1^8 \leftrightarrow \mathbf{x}_2^8$$

$$\begin{bmatrix} x_1^1x_2^1 & x_1^1y_2^1 & x_1^1 & y_1^1x_2^1 & y_1^1y_2^1 & y_1^1 & x_2^1 & y_2^1 & 1 \\ x_1^2x_2^2 & x_1^2y_2^2 & x_1^2 & y_1^2x_2^2 & y_1^2y_2^2 & y_1^2 & x_2^2 & y_2^2 & 1 \\ x_1^3x_2^3 & x_1^3y_2^3 & x_1^3 & y_1^3x_2^3 & y_1^3y_2^3 & y_1^3 & x_2^3 & y_2^3 & 1 \\ x_1^4x_2^4 & x_1^4y_2^4 & x_1^4 & y_1^4x_2^4 & y_1^4y_2^4 & y_1^4 & x_2^4 & y_2^4 & 1 \\ x_1^5x_2^5 & x_1^5y_2^5 & x_1^5 & y_1^5x_2^5 & y_1^5y_2^5 & y_1^5 & x_2^5 & y_2^5 & 1 \\ x_1^6x_2^6 & x_1^6y_2^6 & x_1^6 & y_1^6x_2^6 & y_1^6y_2^6 & y_1^6 & x_2^6 & y_2^6 & 1 \\ x_1^7x_2^7 & x_1^7y_2^7 & x_1^7 & y_1^7x_2^7 & y_1^7y_2^7 & y_1^7 & x_2^7 & y_2^7 & 1 \\ x_1^8x_2^8 & x_1^8y_2^8 & x_1^8 & y_1^8x_2^8 & y_1^8y_2^8 & y_1^8 & x_2^8 & y_2^8 & 1 \end{bmatrix} \begin{bmatrix} f_{11} \\ f_{12} \\ f_{13} \\ f_{21} \\ f_{22} \\ f_{23} \\ f_{31} \\ f_{32} \\ f_{33} \end{bmatrix} = 0 \quad \Rightarrow \quad \mathbf{Ax} = \mathbf{b}$$

$$\mathbf{Af} = \mathbf{0}$$

Direct Linear Transformation (DLT)

Slide by Szymon Rusinkiewicz and Jianxiong Xiao

Algebraic error vs. geometric error

- Algebraic Error

$$\min \|\mathbf{A}\mathbf{f}\|$$

- Geometric Error (better)

$$\min \sum_j d\left(\mathbf{x}_1^j, \mathbf{F}\mathbf{x}_2^j\right)^2 + d\left(\mathbf{x}_2^j, \mathbf{F}^T\mathbf{x}_1^j\right)^2$$

Unit: pixel

Solved by (non-linear) least square solver (e.g. Ceres)

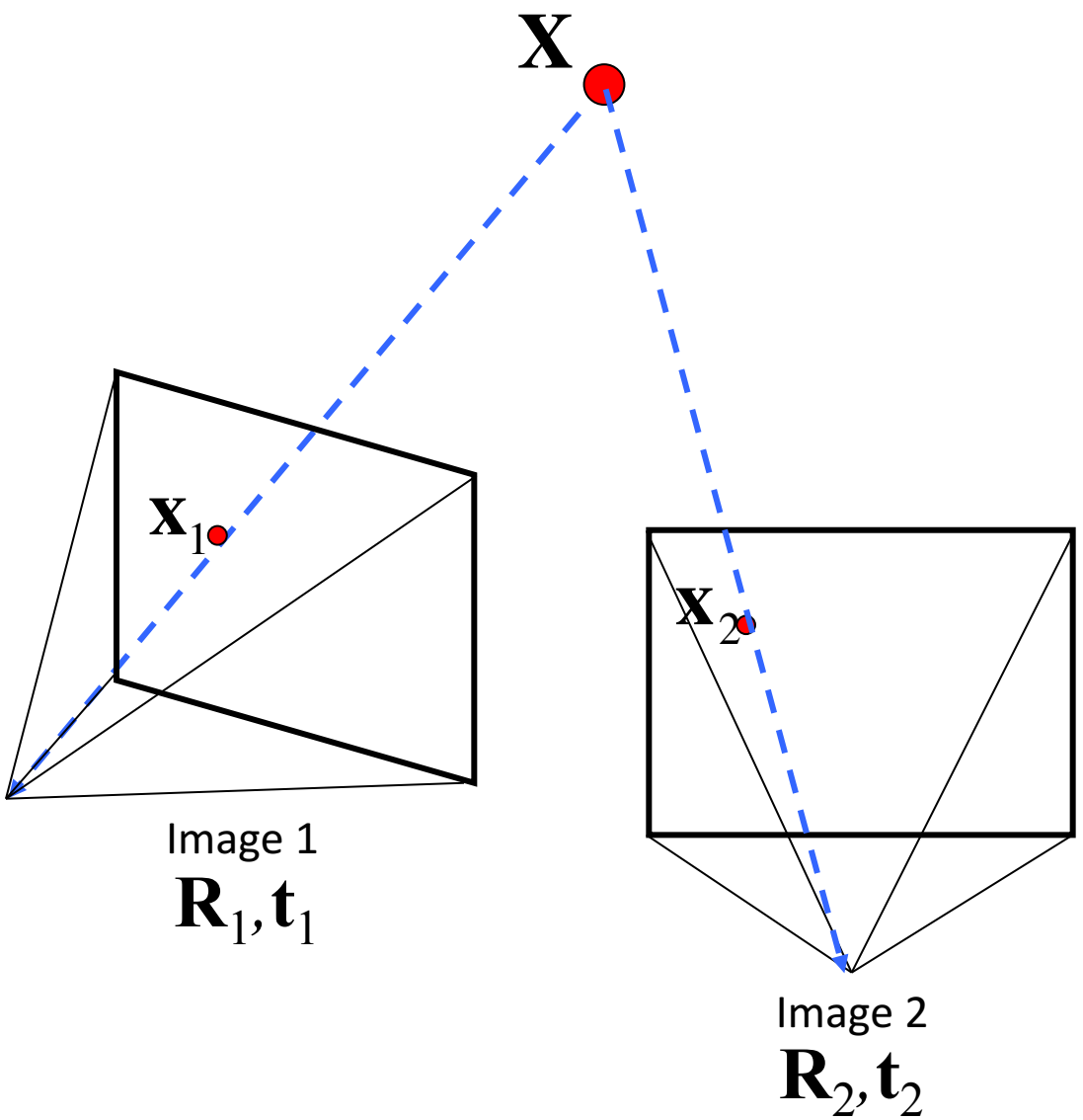
Slide by Szymon Rusinkiewicz and Jianxiong Xiao

RANSAC to estimate the fundamental matrix

- For many times
  - Pick 8 points
  - Compute a solution for $\mathbf{F}$ using these 8 points
  - Count number of inliers that with $\mathbf{x}_1^T \mathbf{F} \mathbf{x}_2$ close to 0
- Pick the one with the largest number of inliers

Slide by Szymon Rusinkiewicz and Jianxiong Xiao

Fundamental matrix → essential matrix

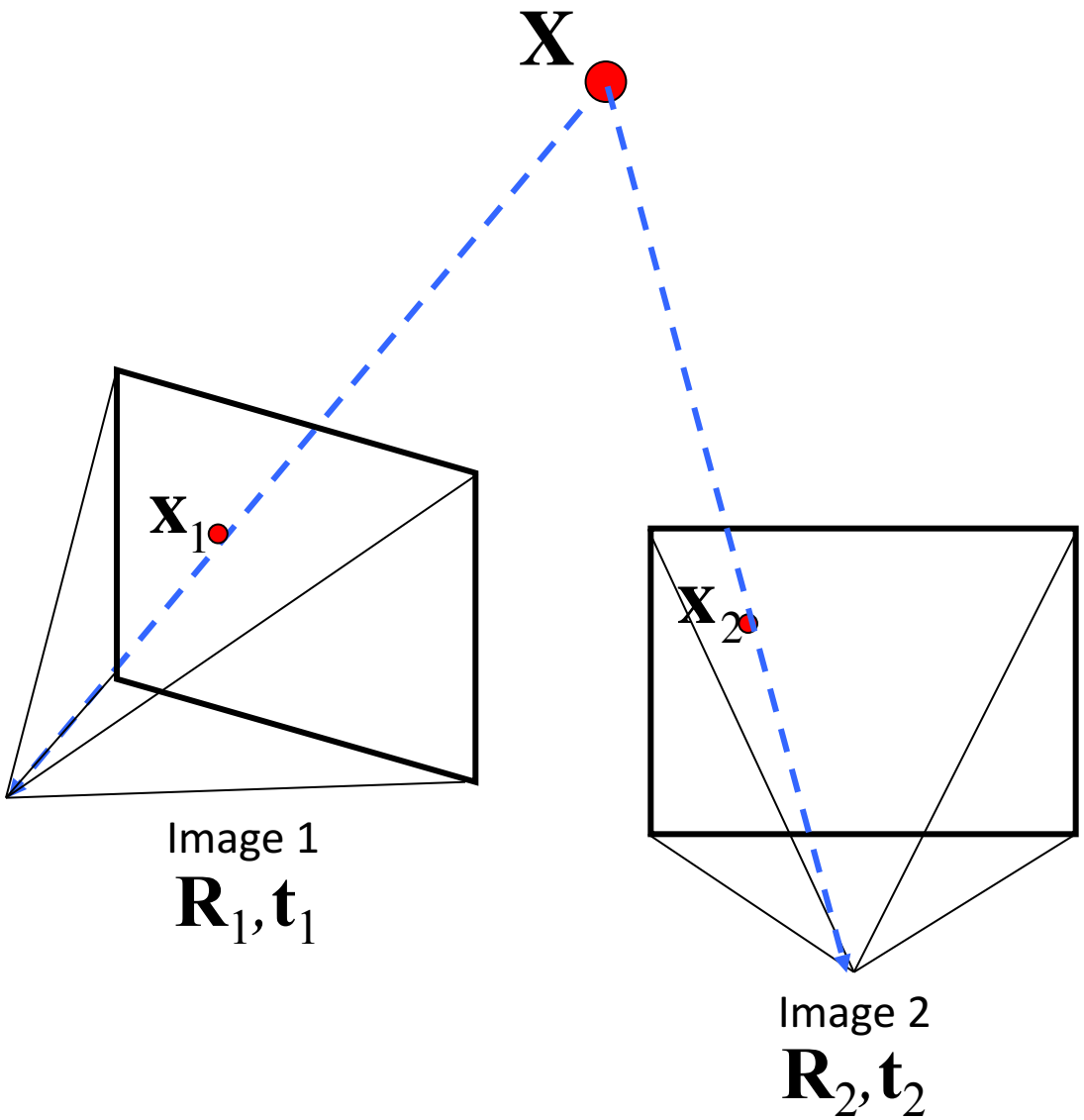


$$\mathbf{x}_1^T \mathbf{F} \mathbf{x}_2 = 0$$

$$\mathbf{E} = \mathbf{K}_1^T \mathbf{F} \mathbf{K}_2$$

Slide by Szymon Rusinkiewicz and Jianxiong Xiao

Essential matrix → $[\mathbf{R}|\mathbf{t}]$



$$\mathbf{x}_1^T \mathbf{F} \mathbf{x}_2 = 0$$

$$\mathbf{E} = \mathbf{K}_1^T \mathbf{F} \mathbf{K}_2$$

Slide by Szymon Rusinkiewicz and Jianxiong Xiao

Essential matrix → $[\mathbf{R}|\mathbf{t}]$

Result 9.19. For a given essential matrix

$$\mathbf{E} = \mathbf{U}\mathrm{diag}(1,1,0)\mathbf{V}^T,$$

and the first camera matrix $\mathbf{P}_1 = [\mathbf{I}|\mathbf{0}]$, there are four possible choices for the second camera matrix $\mathbf{P}_2$:

$$\mathbf{P}_2 = [\mathbf{U}\mathbf{W}\mathbf{V}^T|+\mathbf{u}_3]$$
$$\mathbf{P}_2 = [\mathbf{U}\mathbf{W}\mathbf{V}^T|-\mathbf{u}_3]$$
$$\mathbf{P}_2 = [\mathbf{U}\mathbf{W}^T\mathbf{V}^T|+\mathbf{u}_3]$$
$$\mathbf{P}_2 = [\mathbf{U}\mathbf{W}^T\mathbf{V}^T|-\mathbf{u}_3]$$

$$\mathbf{W} = \begin{bmatrix} 0 & -1 & 0 \\ 1 & 0 & 0 \\ 0 & 0 & 1 \end{bmatrix}$$

Page 259 of the bible (Multiple View Geometry, 2nd Ed)

Four Possible Solutions

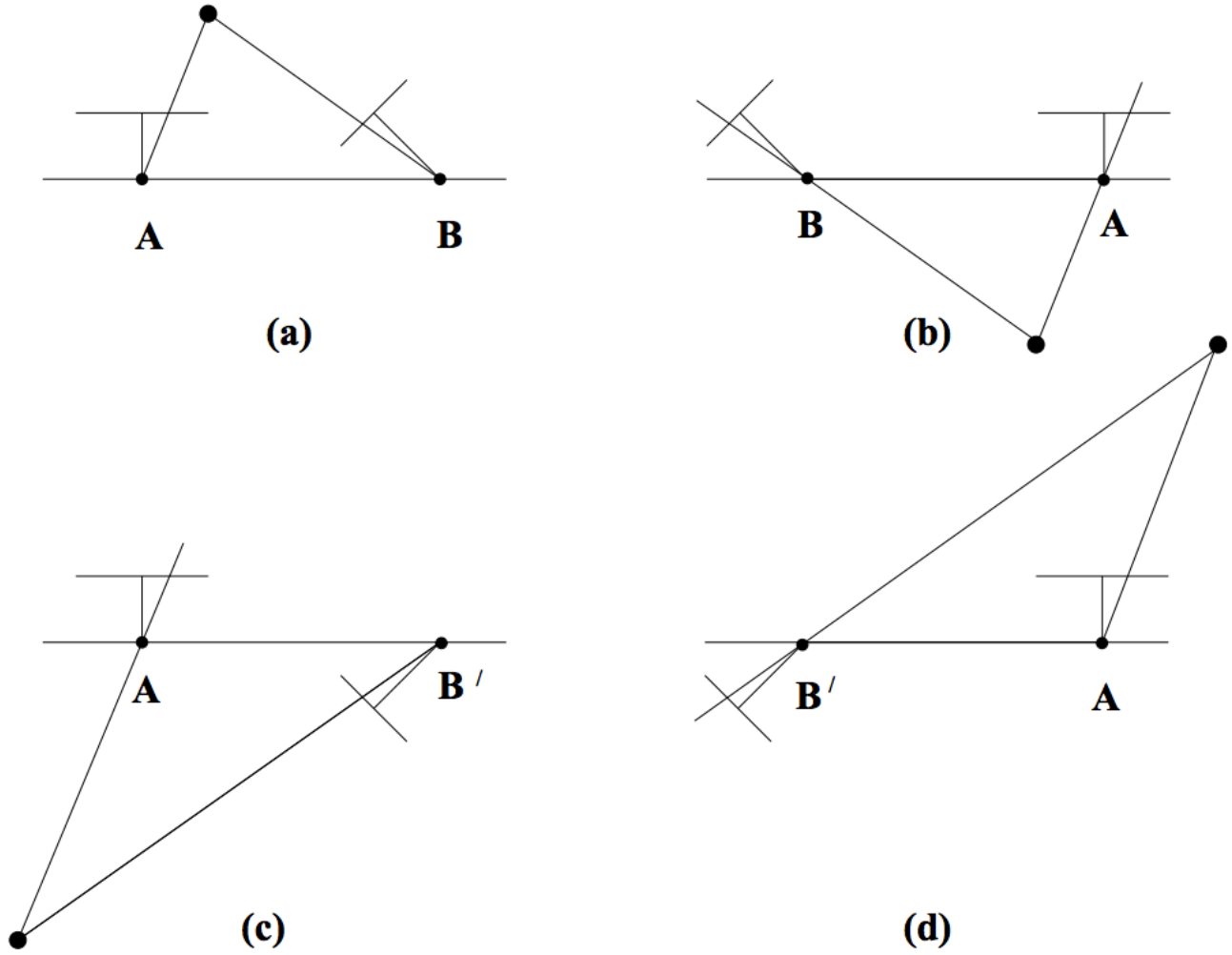
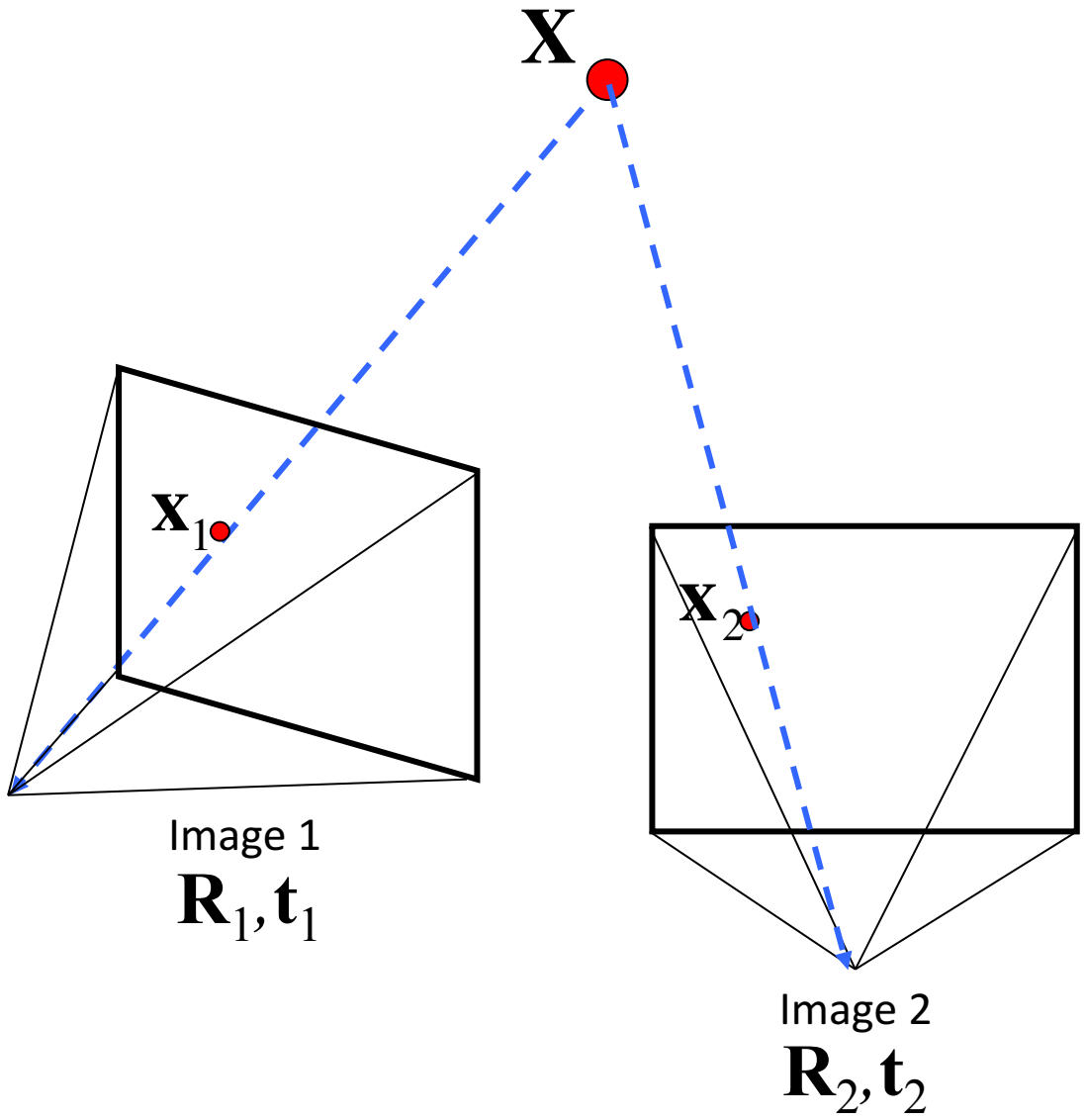


Fig. 9.12. **The four possible solutions for calibrated reconstruction from E.** *Between the left and right sides there is a baseline reversal. Between the top and bottom rows camera B rotates* 180° *about the baseline. Note, only in (a) is the reconstructed point in front of both cameras.*

Page 260 of the bible (Multiple View Geometry, 2nd Ed)

Slide by Szymon Rusinkiewicz and Jianxiong Xiao

Triangulation



Slide by Szymon Rusinkiewicz and Jianxiong Xiao

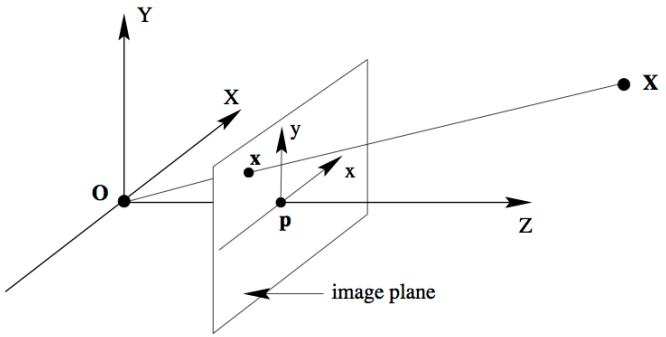
In front of the camera?

- Camera Extrinsic $[\mathbf{R}|\mathbf{t}]$

$$\begin{bmatrix} X_{cam} \\ Y_{cam} \\ Z_{cam} \end{bmatrix} = \mathbf{R}\begin{bmatrix} X_{world} \\ Y_{world} \\ Z_{world} \end{bmatrix} + \mathbf{t} \quad \Leftrightarrow \quad \begin{bmatrix} X_{world} \\ Y_{world} \\ Z_{world} \end{bmatrix} = \mathbf{R}^{-1}\left(\begin{bmatrix} X_{cam} \\ Y_{cam} \\ Z_{cam} \end{bmatrix} - \mathbf{t}\right) = \mathbf{R}^{T}\begin{bmatrix} X_{cam} \\ Y_{cam} \\ Z_{cam} \end{bmatrix} - \mathbf{R}^{T}\mathbf{t}$$

- Camera Center

$$\begin{bmatrix} X_{cam} \\ Y_{cam} \\ Z_{cam} \end{bmatrix} = \begin{bmatrix} 0 \\ 0 \\ 0 \end{bmatrix} \Rightarrow \mathbf{C} = \begin{bmatrix} X_{world} \\ Y_{world} \\ Z_{world} \end{bmatrix} = \mathbf{R}^{T}\begin{bmatrix} 0 \\ 0 \\ 0 \end{bmatrix} - \mathbf{R}^{T}\mathbf{t} = -\mathbf{R}^{T}\mathbf{t}$$



- View Direction

$$\begin{bmatrix} 0 \\ 0 \\ 1 \end{bmatrix} - \begin{bmatrix} 0 \\ 0 \\ 0 \end{bmatrix} \Rightarrow \left(\mathbf{R}^{T}\begin{bmatrix} 0 \\ 0 \\ 1 \end{bmatrix} - \mathbf{R}^{T}\mathbf{t}\right) - (\mathbf{C}) = \left(\mathbf{R}(3,:)^{T} - \mathbf{R}^{T}\mathbf{t}\right) - \left(-\mathbf{R}^{T}\mathbf{t}\right) = \mathbf{R}(3,:)^{T}$$

Camera Coordinate System World Coordinate System

Slide by Szymon Rusinkiewicz and Jianxiong Xiao

In front of the camera?

- A point $\mathbf{X}$
- Direction from camera center to point $\mathbf{X}-\mathbf{C}$
- Angle between two vectors

$$\mathbf{A}\cdot\mathbf{B}=\|\mathbf{A}\|\|\mathbf{B}\|\cos\theta$$

- Angle between $\mathbf{X}-\mathbf{C}$ and view virection
- Just need to test $(\mathbf{X}-\mathbf{C})\cdot\mathbf{R}(3,:)^T>0$?

Slide by Szymon Rusinkiewicz and Jianxiong Xiao

Pick the solution

with maximal number of points in front of both cameras

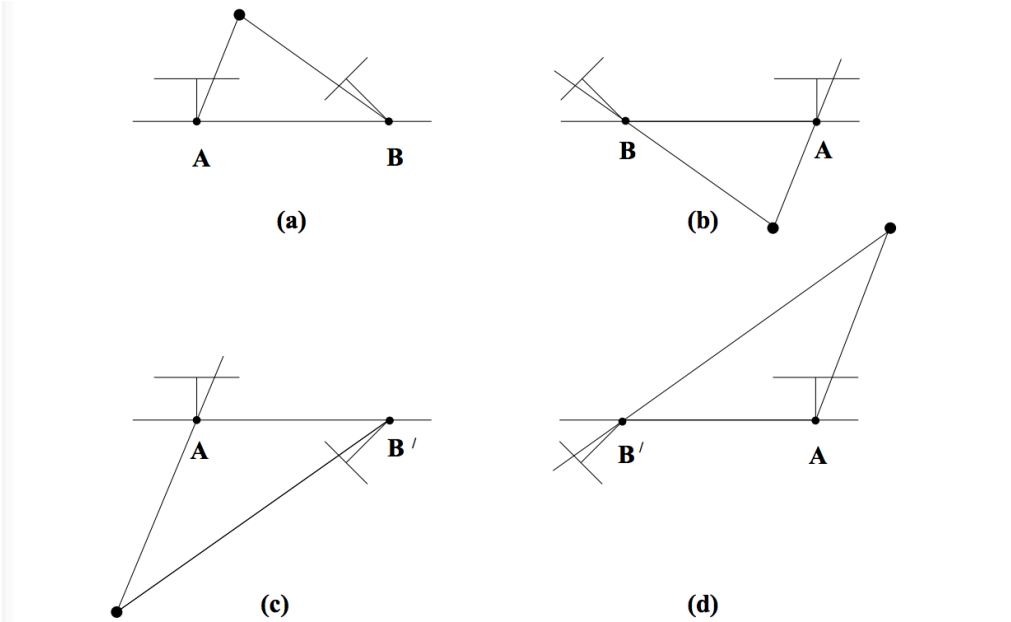
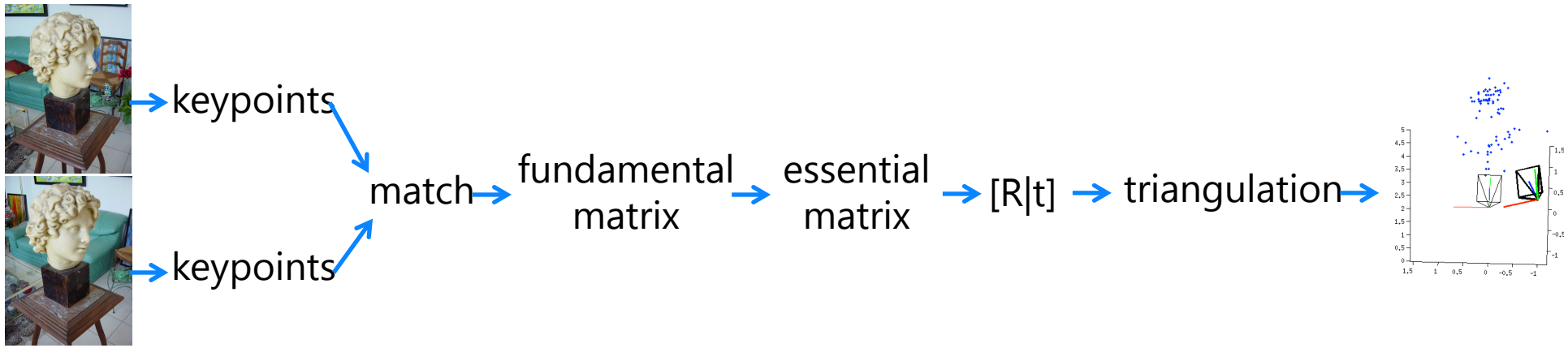


Fig. 9.12. **The four possible solutions for calibrated reconstruction from E.** *Between the left and right sides there is a baseline reversal. Between the top and bottom rows camera B rotates* 180° *about the baseline. Note, only in (a) is the reconstructed point in front of both cameras.*

Page 260 of the bible (Multiple View Geometry, 2nd Ed)

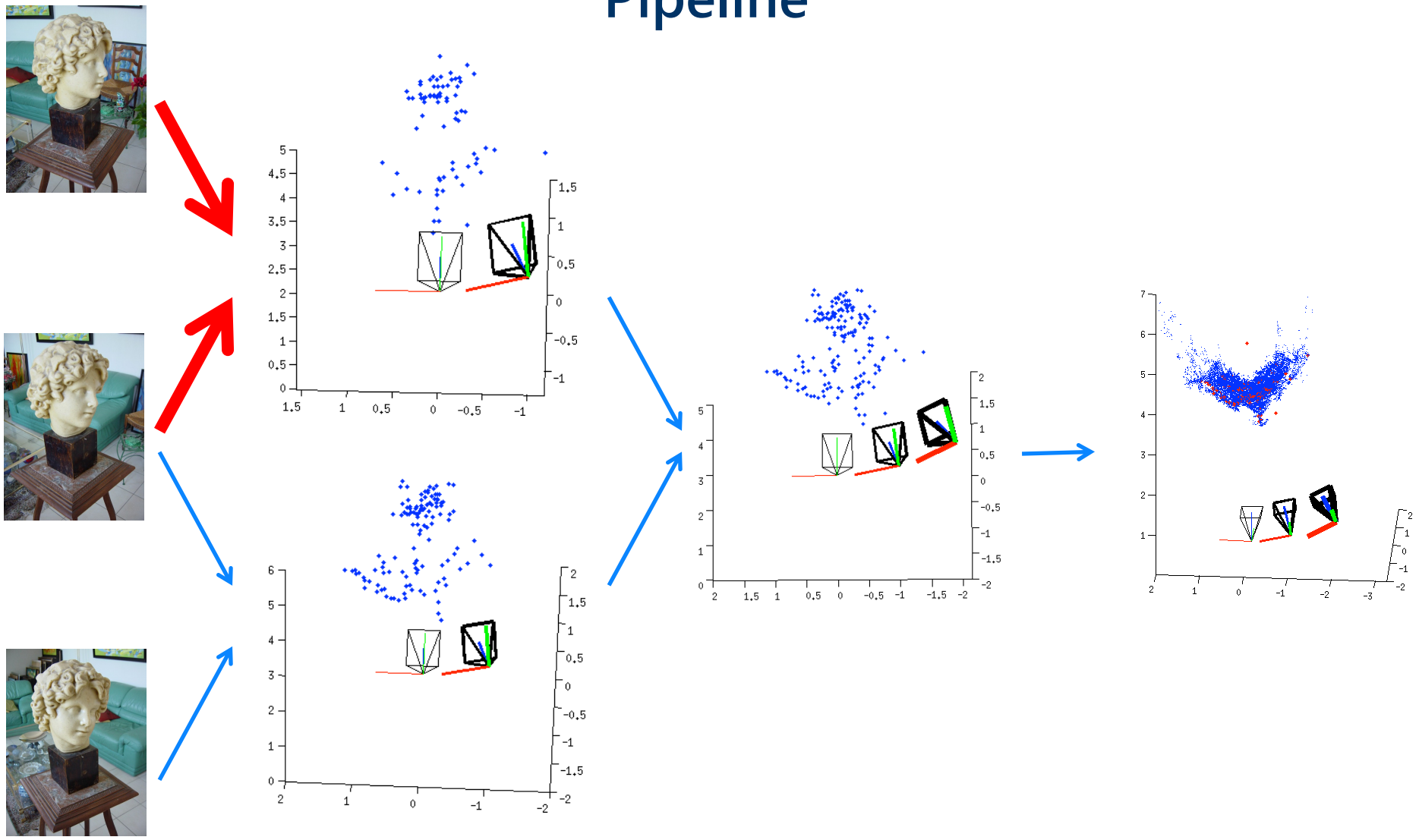
Slide by Szymon Rusinkiewicz and Jianxiong Xiao

Two-view reconstruction



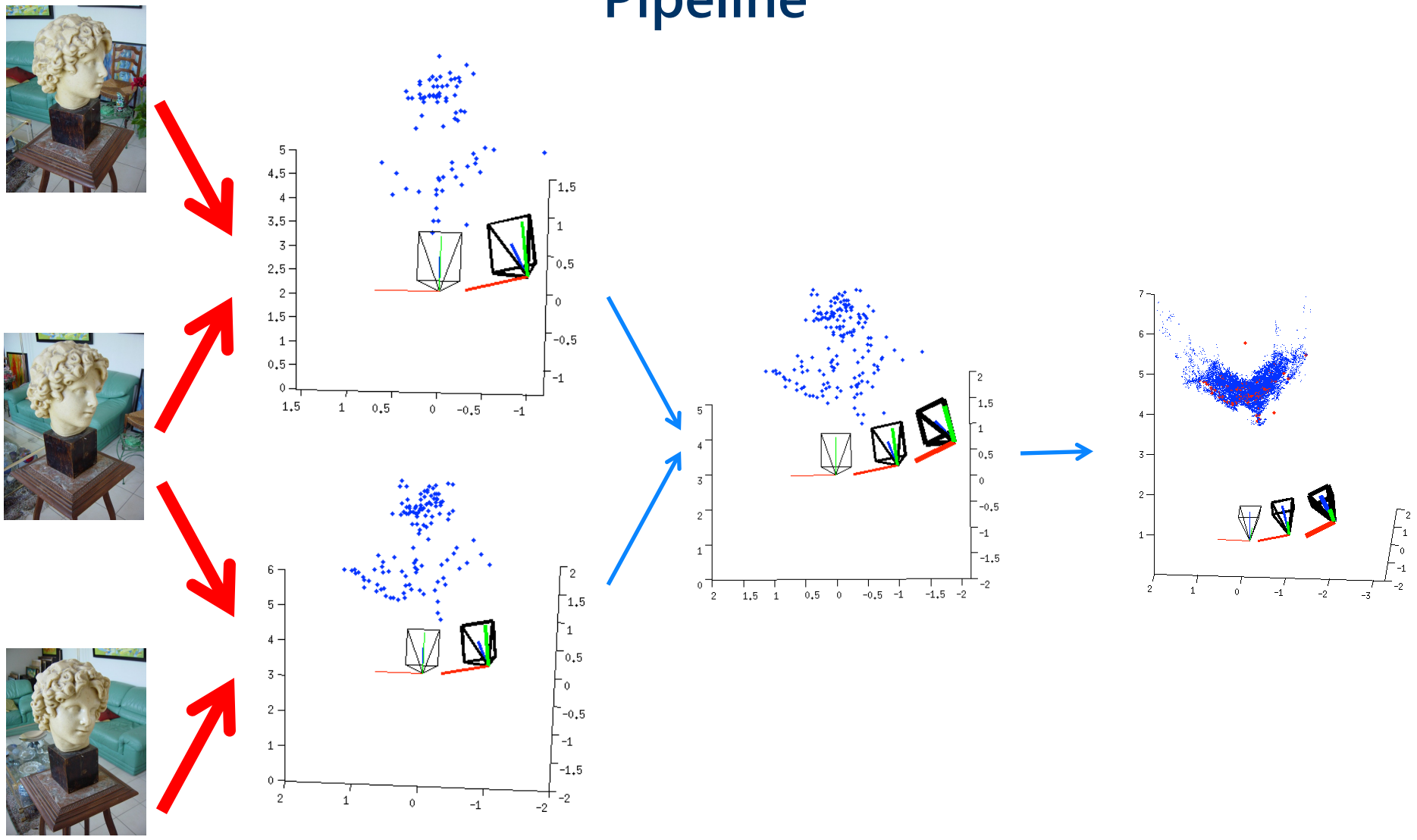
Slide by Szymon Rusinkiewicz and Jianxiong Xiao

Pipeline



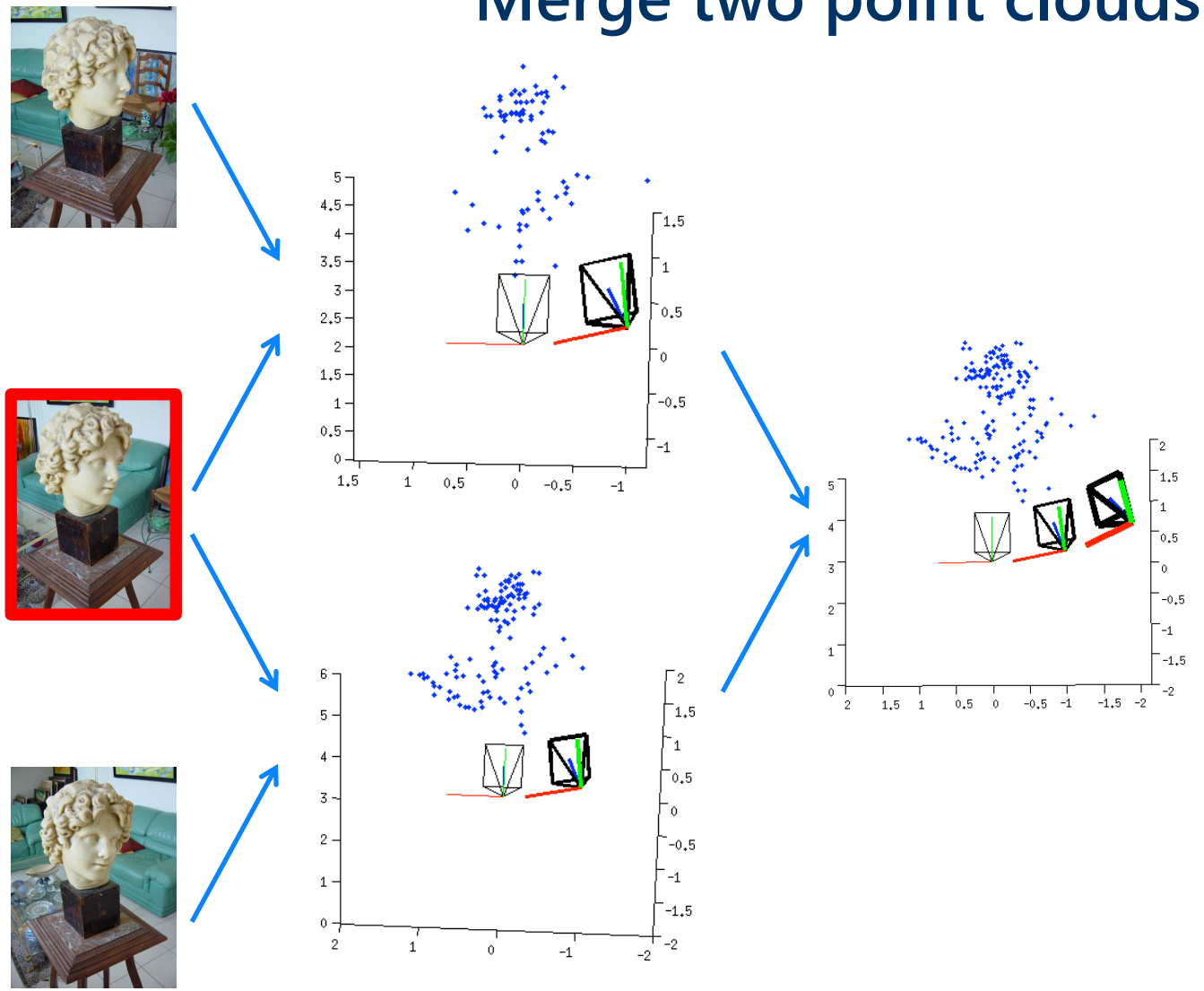
Slide by Szymon Rusinkiewicz and Jianxiong Xiao

Pipeline



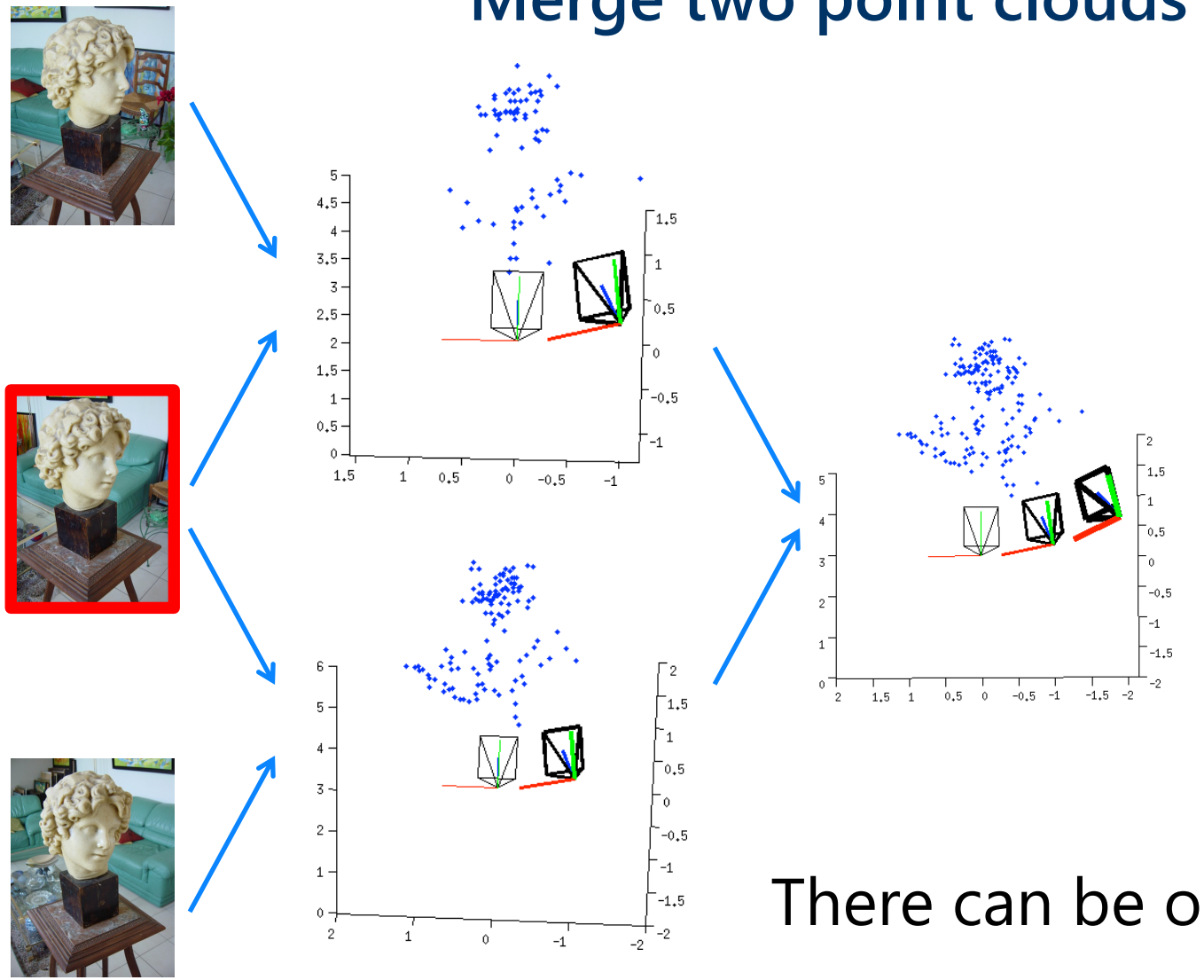
Slide by Szymon Rusinkiewicz and Jianxiong Xiao

Merge two point clouds



Slide by Szymon Rusinkiewicz and Jianxiong Xiao

Merge two point clouds



There can be only one $[\mathbf{R}_2|\mathbf{t}_2]$

Slide by Szymon Rusinkiewicz and Jianxiong Xiao

Merge two point clouds

- From the 1st and 2nd images, we have

$$[\mathbf{R}_1|\mathbf{t}_1] \text{ and } [\mathbf{R}_2|\mathbf{t}_2]$$

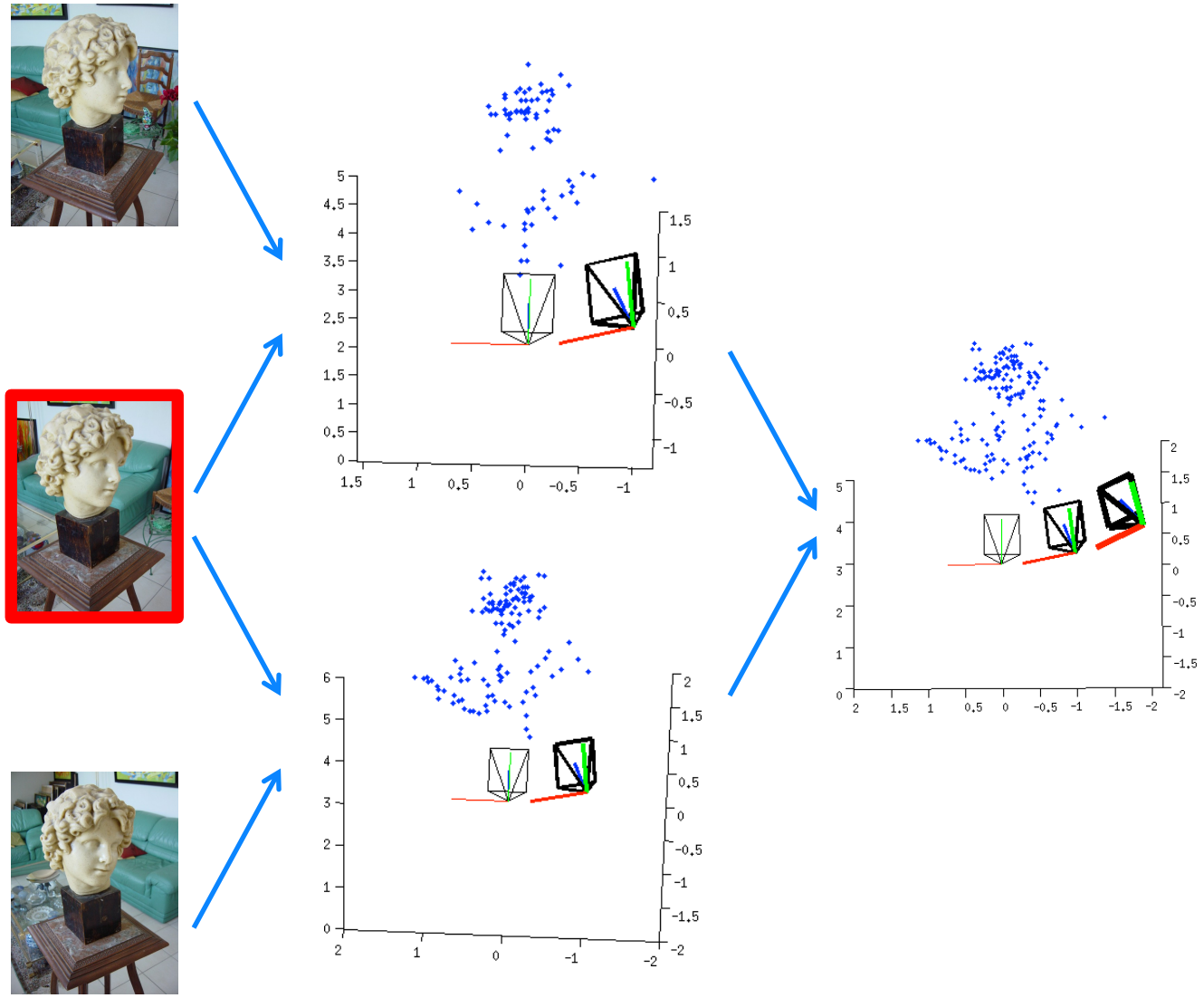
- From the 2nd and 3rd images, we have

$$[\mathbf{R}_2|\mathbf{t}_2] \text{ and } [\mathbf{R}_3|\mathbf{t}_3]$$

- **Exercise**: How to transform the coordinate system of the second point cloud to align with the first point cloud so that there is only one $[\mathbf{R}_2|\mathbf{t}_2]$?

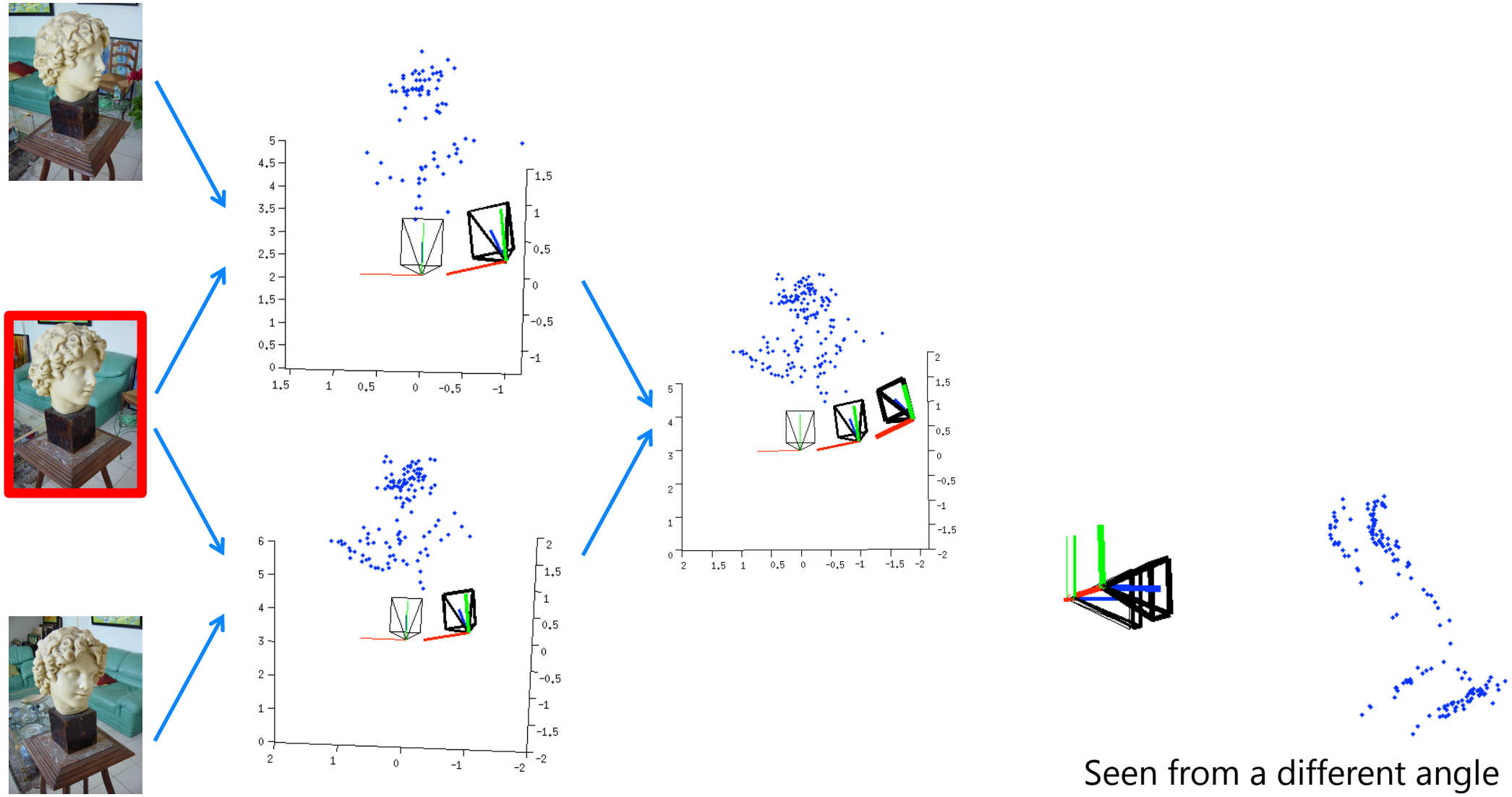
Slide by Szymon Rusinkiewicz and Jianxiong Xiao

Merge two point clouds



Slide by Szymon Rusinkiewicz and Jianxiong Xiao

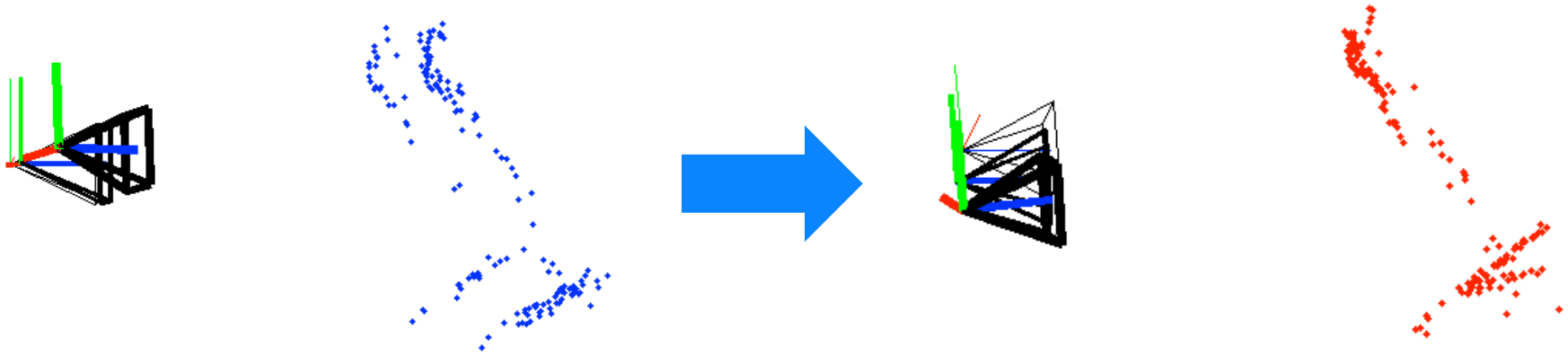
Oops



Seen from a different angle

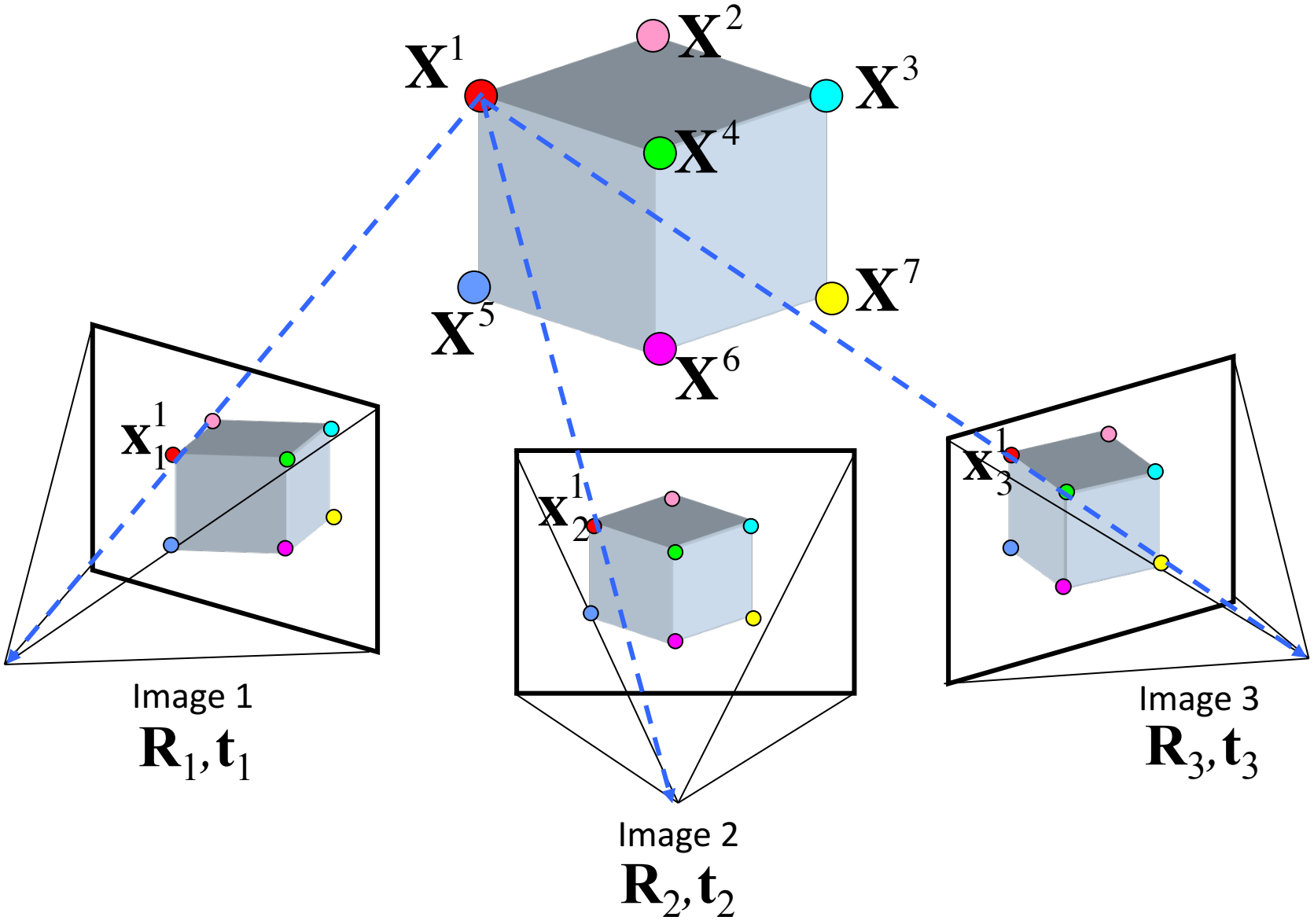
Slide by Szymon Rusinkiewicz and Jianxiong Xiao

Bundle adjustment



Slide by Szymon Rusinkiewicz and Jianxiong Xiao

“America's Next Top Model”



Slide by Szymon Rusinkiewicz and Jianxiong Xiao

“America's Next Top Model”

	Point 1	Point 2	Point 3
Image 1	$\mathbf{x}_1^1 = \mathbf{K}\left[\mathbf{R}_1 \middle| \mathbf{t}_1\right]\mathbf{X}^1$	$\mathbf{x}_1^2 = \mathbf{K}\left[\mathbf{R}_1 \middle| \mathbf{t}_1\right]\mathbf{X}^2$	
Image 2	$\mathbf{x}_2^1 = \mathbf{K}\left[\mathbf{R}_2 \middle| \mathbf{t}_2\right]\mathbf{X}^1$	$\mathbf{x}_2^2 = \mathbf{K}\left[\mathbf{R}_2 \middle| \mathbf{t}_2\right]\mathbf{X}^2$	$\mathbf{x}_2^3 = \mathbf{K}\left[\mathbf{R}_2 \middle| \mathbf{t}_2\right]\mathbf{X}^3$
Image 3	$\mathbf{x}_3^1 = \mathbf{K}\left[\mathbf{R}_3 \middle| \mathbf{t}_3\right]\mathbf{X}^1$		$\mathbf{x}_3^3 = \mathbf{K}\left[\mathbf{R}_3 \middle| \mathbf{t}_3\right]\mathbf{X}^3$

Slide by Szymon Rusinkiewicz and Jianxiong Xiao

Rethinking the SFM problem

- Input: observed 2D image position

$$\begin{matrix} \tilde{\mathbf{x}}_1^1 & \tilde{\mathbf{x}}_1^2 & \\ \tilde{\mathbf{x}}_2^1 & \tilde{\mathbf{x}}_2^2 & \tilde{\mathbf{x}}_2^3 \\ \tilde{\mathbf{x}}_3^1 & & \tilde{\mathbf{x}}_3^3 \end{matrix}$$

- Output:

Unknown camera parameters (with some guess)

$$\left[\mathbf{R}_1|\mathbf{t}_1\right],\left[\mathbf{R}_2|\mathbf{t}_2\right],\left[\mathbf{R}_3|\mathbf{t}_3\right]$$

Unknown point 3D coordinate (with some guess)

$$\mathbf{X}^1,\mathbf{X}^2,\mathbf{X}^3,\ldots$$

Slide by Szymon Rusinkiewicz and Jianxiong Xiao

Bundle adjustment

A valid solution $[\mathbf{R}_1|\mathbf{t}_1],[\mathbf{R}_2|\mathbf{t}_2],[\mathbf{R}_3|\mathbf{t}_3]$ and $\mathbf{X}^1,\mathbf{X}^2,\mathbf{X}^3,\cdots$

must let

Re-projection

$$\begin{cases} \mathbf{x}_1^1 = \mathbf{K}[\mathbf{R}_1|\mathbf{t}_1]\mathbf{X}^1 & \mathbf{x}_1^2 = \mathbf{K}[\mathbf{R}_1|\mathbf{t}_1]\mathbf{X}^2 & \\ \mathbf{x}_2^1 = \mathbf{K}[\mathbf{R}_2|\mathbf{t}_2]\mathbf{X}^1 & \mathbf{x}_2^2 = \mathbf{K}[\mathbf{R}_2|\mathbf{t}_2]\mathbf{X}^2 & \mathbf{x}_2^3 = \mathbf{K}[\mathbf{R}_2|\mathbf{t}_2]\mathbf{X}^3 \\ \mathbf{x}_3^1 = \mathbf{K}[\mathbf{R}_3|\mathbf{t}_3]\mathbf{X}^1 & & \mathbf{x}_3^3 = \mathbf{K}[\mathbf{R}_3|\mathbf{t}_3]\mathbf{X}^3 \end{cases}$$

=

Observation

$$\begin{cases} \tilde{\mathbf{x}}_1^1 & \tilde{\mathbf{x}}_1^2 & \\ \tilde{\mathbf{x}}_2^1 & \tilde{\mathbf{x}}_2^2 & \tilde{\mathbf{x}}_2^3 \\ \tilde{\mathbf{x}}_3^1 & & \tilde{\mathbf{x}}_3^3 \end{cases}$$

Slide by Szymon Rusinkiewicz and Jianxiong Xiao

Bundle adjustment

A valid solution $[\mathbf{R}_1|\mathbf{t}_1],[\mathbf{R}_2|\mathbf{t}_2],[\mathbf{R}_3|\mathbf{t}_3]$ and $\mathbf{X}^1,\mathbf{X}^2,\mathbf{X}^3,\cdots$

must let the reprojection close to the observation,

i.e. to minimize the reprojection error

$$\min \sum_i \sum_j \left(\tilde{\mathbf{x}}_i^j - \mathbf{K}[\mathbf{R}_i|\mathbf{t}_i]\mathbf{X}^j \right)^2$$

Slide by Szymon Rusinkiewicz and Jianxiong Xiao

Bundle adjustment

A valid solution $[\mathbf{R}_1|\mathbf{t}_1],[\mathbf{R}_2|\mathbf{t}_2],[\mathbf{R}_3|\mathbf{t}_3]$ and $\mathbf{X}^1,\mathbf{X}^2,\mathbf{X}^3,\cdots$

must let the reprojection close to the observation,

i.e. to minimize the reprojection error

$$\min \sum_i \sum_j \left(\tilde{\mathbf{x}}_i^j - \mathbf{K}[\mathbf{R}_i|\mathbf{t}_i]\mathbf{X}^j \right)^2$$

Question: What is the unit of this objective function?

Slide by Szymon Rusinkiewicz and Jianxiong Xiao

Solving this optimization problem

- Theory:

  The Levenberg–Marquardt algorithm

  http://en.wikipedia.org/wiki/Levenberg-Marquardt_algorithm

- Practice:

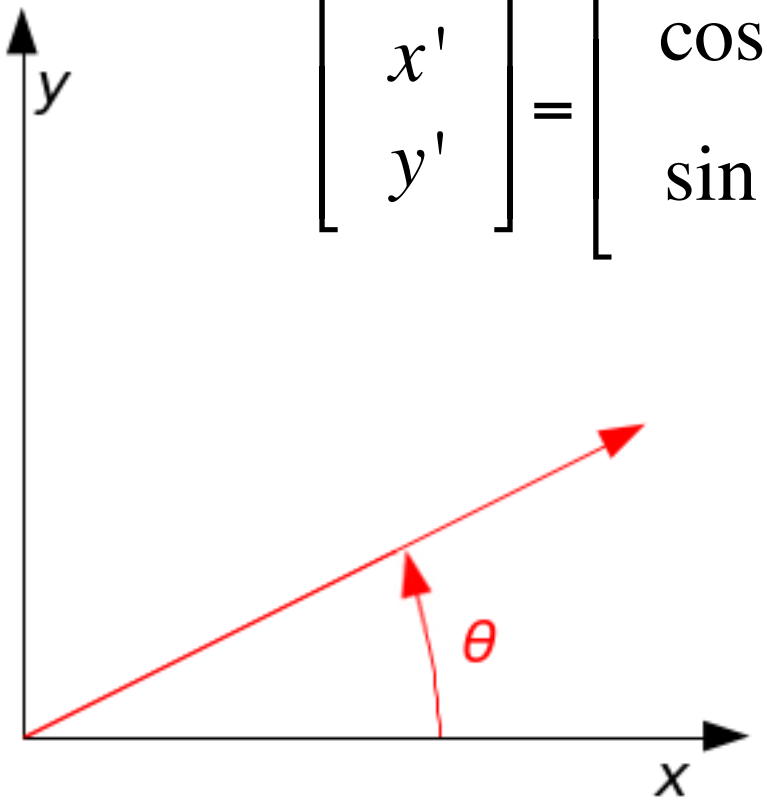
  The Ceres-Solver from Google

  http://code.google.com/p/ceres-solver/

Slide by Szymon Rusinkiewicz and Jianxiong Xiao

Parameterizing rotation matrix

- 2D Rotation



$$\begin{bmatrix} x' \\ y' \end{bmatrix} = \begin{bmatrix} \cos(\theta) & -\sin(\theta) \\ \sin(\theta) & \cos(\theta) \end{bmatrix} \begin{bmatrix} x \\ y \end{bmatrix}$$

$$\mathbf{R}^T = \mathbf{R}^{-1}, \det \mathbf{R} = 1$$

Slide by Szymon Rusinkiewicz and Jianxiong Xiao

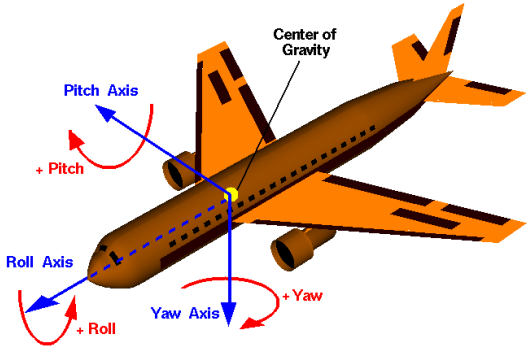
3D rotation

$$\mathbf{R}_x(\theta) = \begin{bmatrix} 1 & 0 & 0 \\ 0 & \cos(\theta) & -\sin(\theta) \\ 0 & \sin(\theta) & \cos(\theta) \end{bmatrix}$$

$$\mathbf{R}_y(\theta) = \begin{bmatrix} \cos(\theta) & 0 & \sin(\theta) \\ 0 & 1 & 0 \\ -\sin(\theta) & 0 & \cos(\theta) \end{bmatrix}$$

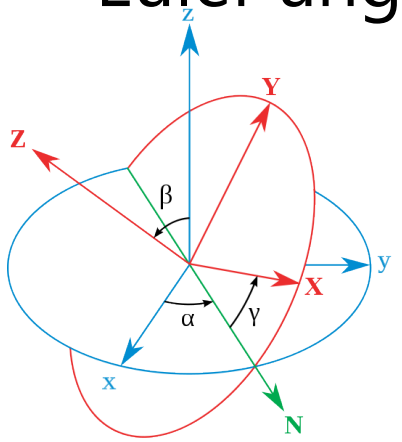
$$\mathbf{R}_z(\theta) = \begin{bmatrix} \cos(\theta) & -\sin(\theta) & 0 \\ \sin(\theta) & \cos(\theta) & 0 \\ 0 & 0 & 1 \end{bmatrix}$$

Yaw, pitch and roll are α, β, γ



$$\mathbf{R}_x(\gamma)\mathbf{R}_y(\beta)\mathbf{R}_z(\alpha)$$

Euler angles are α, β, γ



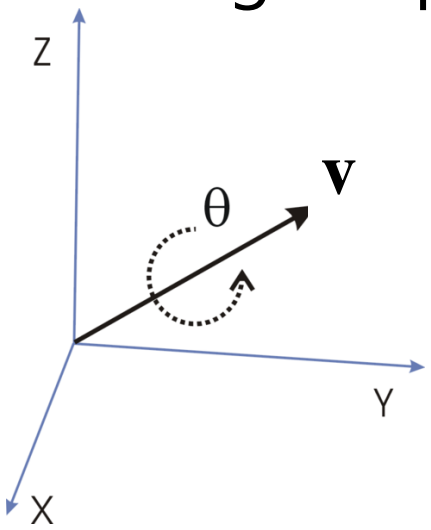
$$\mathbf{R}_z(\gamma)\mathbf{R}_x(\beta)\mathbf{R}_y(\alpha)$$

http://en.wikipedia.org/wiki/Rotation_matrix

Slide by Szymon Rusinkiewicz and Jianxiong Xiao

3D rotation

Axis-angle representation



Quaternions

$$\mathbf{q} = \left(\mathbf{v} \sin\left(\frac{\theta}{2}\right), \cos\left(\frac{\theta}{2}\right) \right)^T$$

Avoid Gimbal Lock!

Recommendation!

Triplet Representation $\mathbf{v}\theta$ (3 dof) ← Not over-parameterized

Rodrigues' rotation formula

$$\mathbf{k}_{rot} = \mathbf{k}\cos\theta + (\mathbf{v} \times \mathbf{k})\sin\theta + \mathbf{v}(\mathbf{v} \cdot \mathbf{k})(1 - \cos\theta)$$

http://en.wikipedia.org/wiki/Axis-angle_representation
http://en.wikipedia.org/wiki/Quaternions_and_spatial_rotation
http://en.wikipedia.org/wiki/Rodrigues%27_rotation_formula

Slide by Szymon Rusinkiewicz and Jianxiong Xiao

Initialization matters

- Input: Observed 2D image position

$$\begin{matrix} \tilde{\mathbf{x}}_1^1 & \tilde{\mathbf{x}}_1^2 & \\ \tilde{\mathbf{x}}_2^1 & \tilde{\mathbf{x}}_2^2 & \tilde{\mathbf{x}}_2^3 \\ \tilde{\mathbf{x}}_3^1 & & \tilde{\mathbf{x}}_3^3 \end{matrix}$$

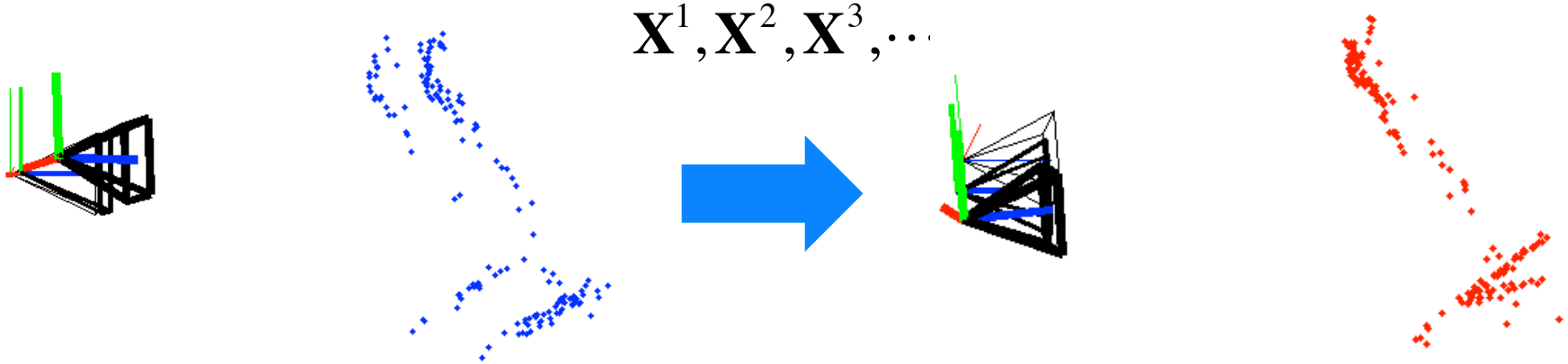
- Output:

Unknown camera parameters (with some guess)

$$[\mathbf{R}_1|\mathbf{t}_1],[\mathbf{R}_2|\mathbf{t}_2],[\mathbf{R}_3|\mathbf{t}_3]$$

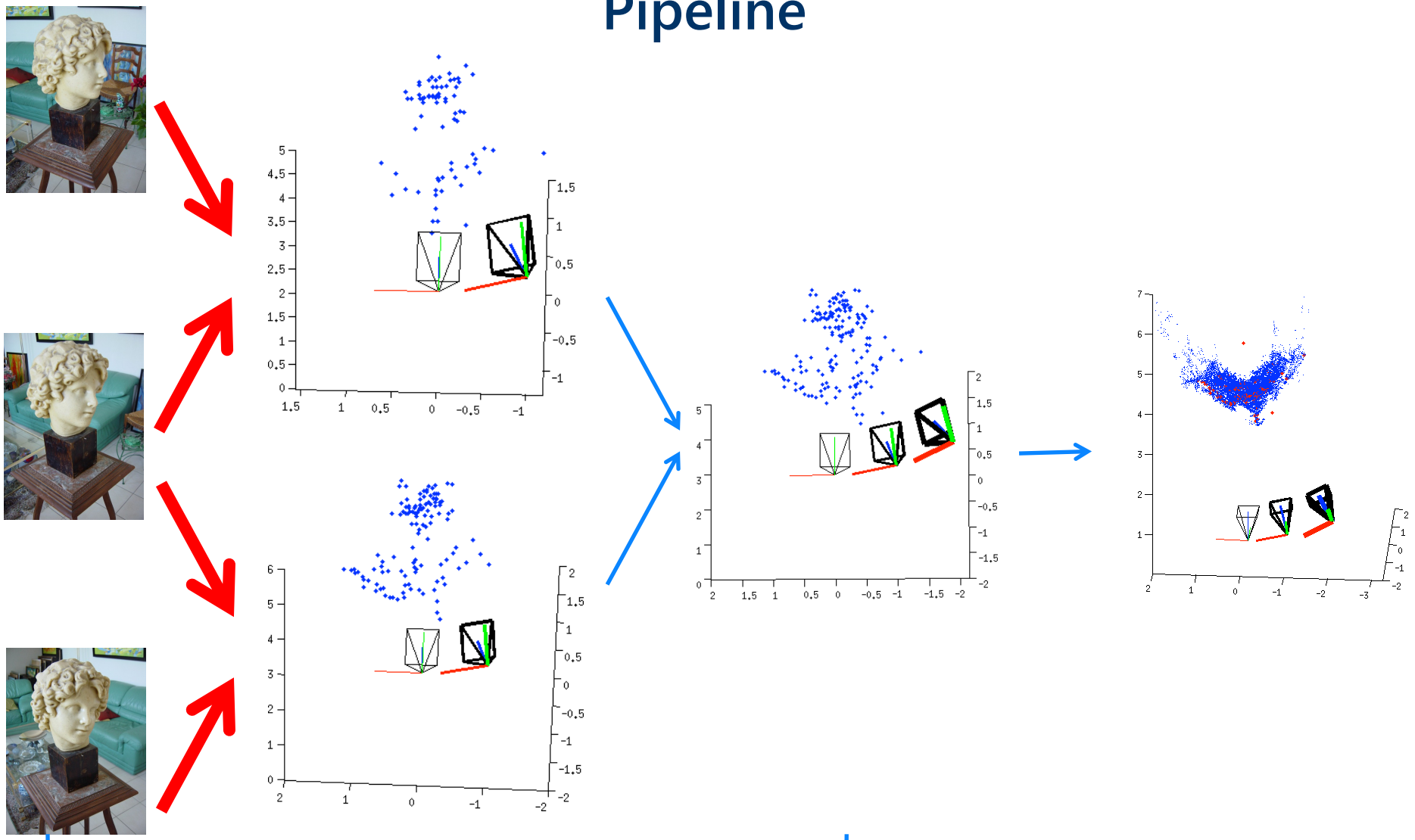
Unknown point 3D coordinate (with some guess)

$$\mathbf{X}^1,\mathbf{X}^2,\mathbf{X}^3,\cdots$$



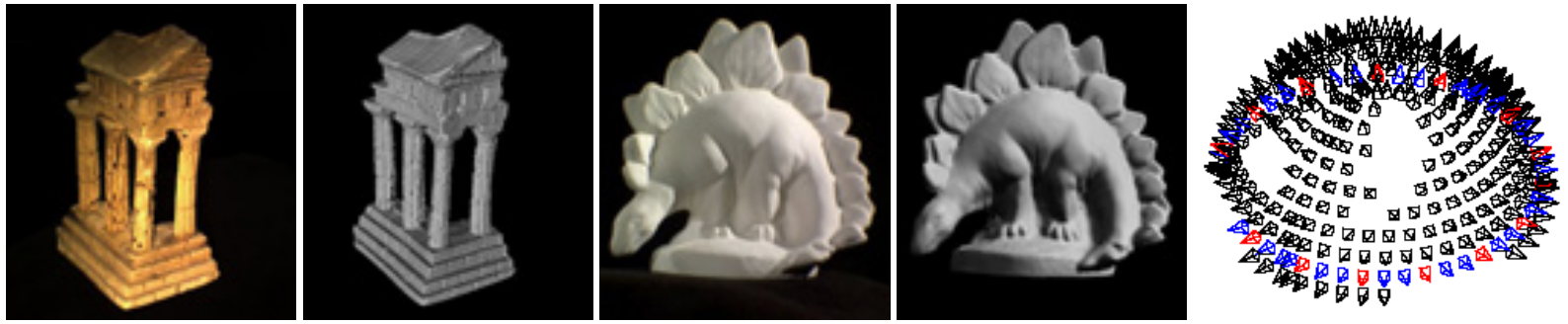
Slide by Szymon Rusinkiewicz and Jianxiong Xiao

Pipeline



Slide by Szymon Rusinkiewicz and Jianxiong Xiao

Multi-view stereo



State-of-the-art:

COLMAP: https://colmap.github.io/
Pixelwise View Selection for Unstructured Multi-View Stereo
JL Schönberger, E Zheng, M Pollefeys, J-M Frahm, ECCV 2016.

Benchmark:

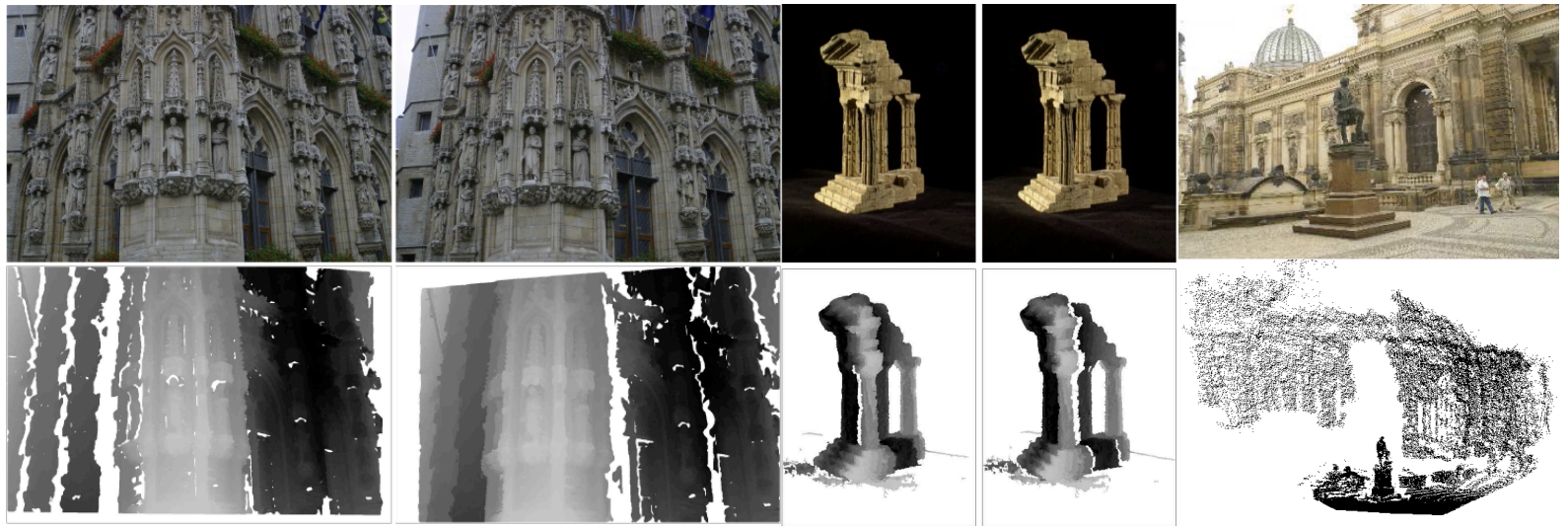
http://vision.middlebury.edu/mview/
A Comparison and Evaluation of Multi-View Stereo Reconstruction Algorithms.
SM Seitz, B Curless, J Diebel, D Scharstein, R Szeliski. CVPR 2006.

Baseline:

Multi-view stereo revisited. M Goesele, B Curless, SM Seitz. CVPR 2006.

Slide by Szymon Rusinkiewicz and Jianxiong Xiao

Key idea: matching propagation



[1] Learning Two-view Stereo Matching, J Xiao, J Chen, DY Yeung, and L Quan, 2008.

[2] Accurate, Dense, and Robust Multi-View Stereopsis, Y Furukawa and J Ponce, 2007.

[3] Multi-view stereo revisited. M Goesele, B Curless, SM Seitz. 2006.

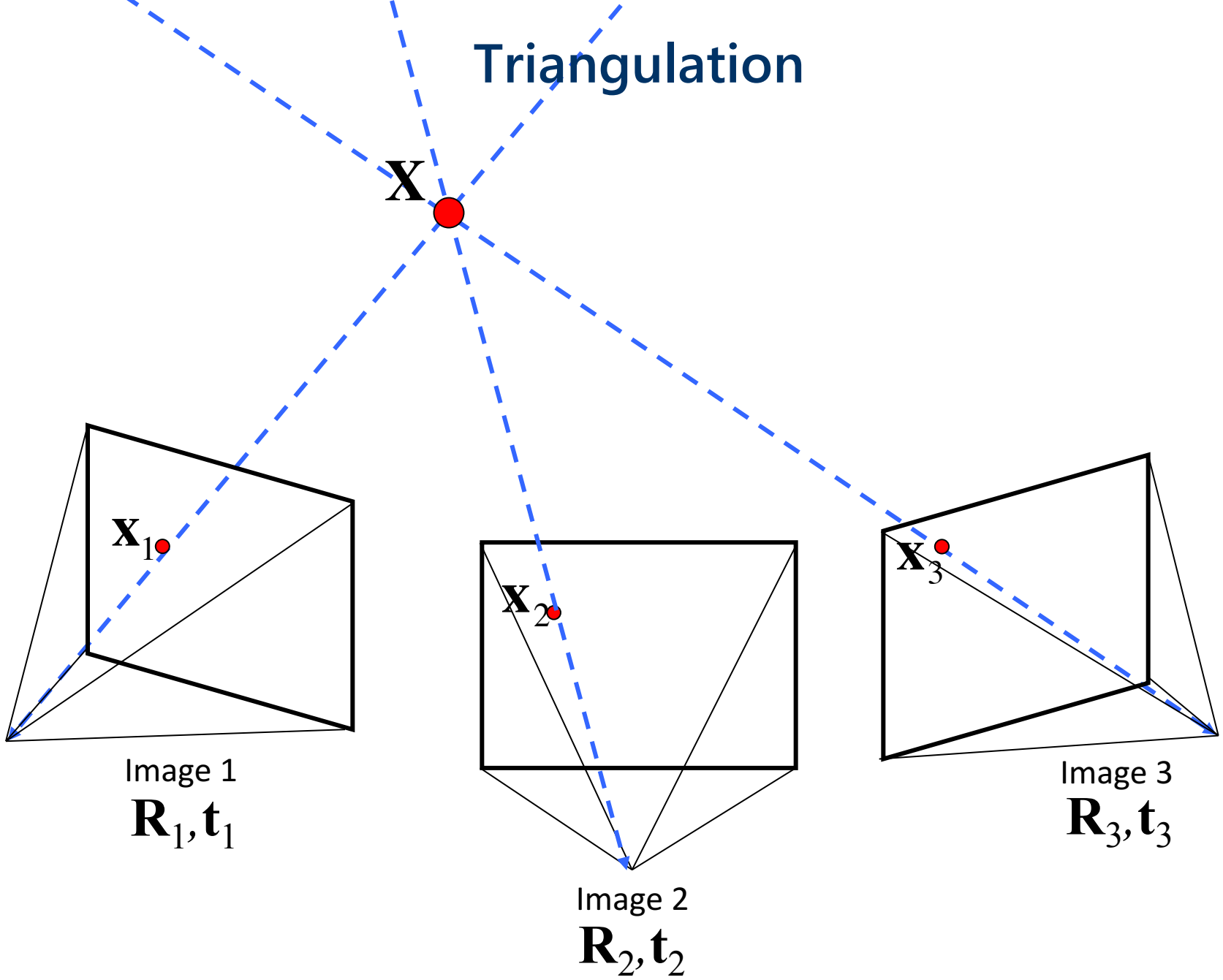
[4] Robust Dense Matching Using Local and Global Geometric Constraints, M Lhuillier & L Quan, 2000.

In another context:

[5] PatchMatch: A Randomized Correspondence Algorithm for Structural Image Editing, C Barnes, E Shechtman, A Finkelstein, and DB Goldman, 2009

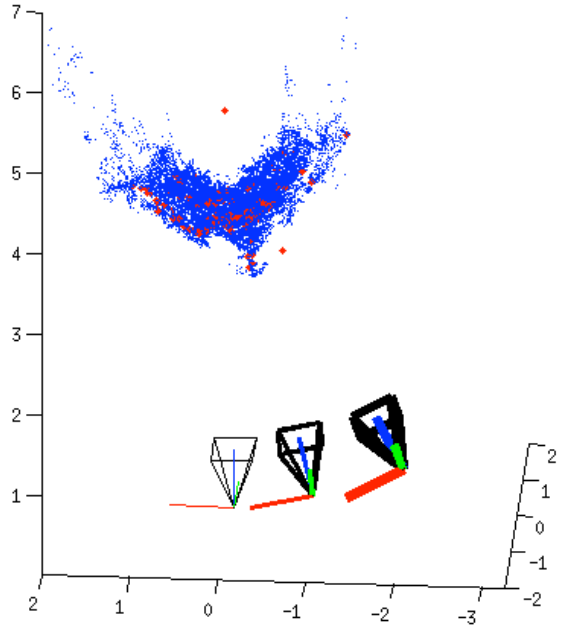
Slide by Szymon Rusinkiewicz and Jianxiong Xiao

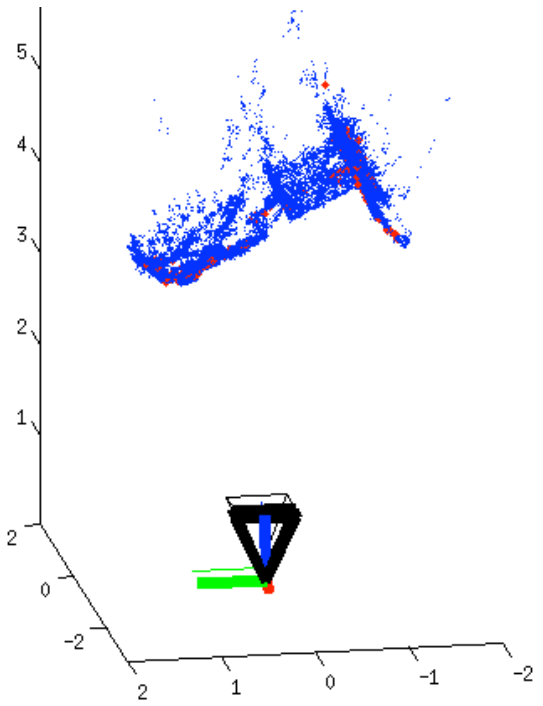
Triangulation

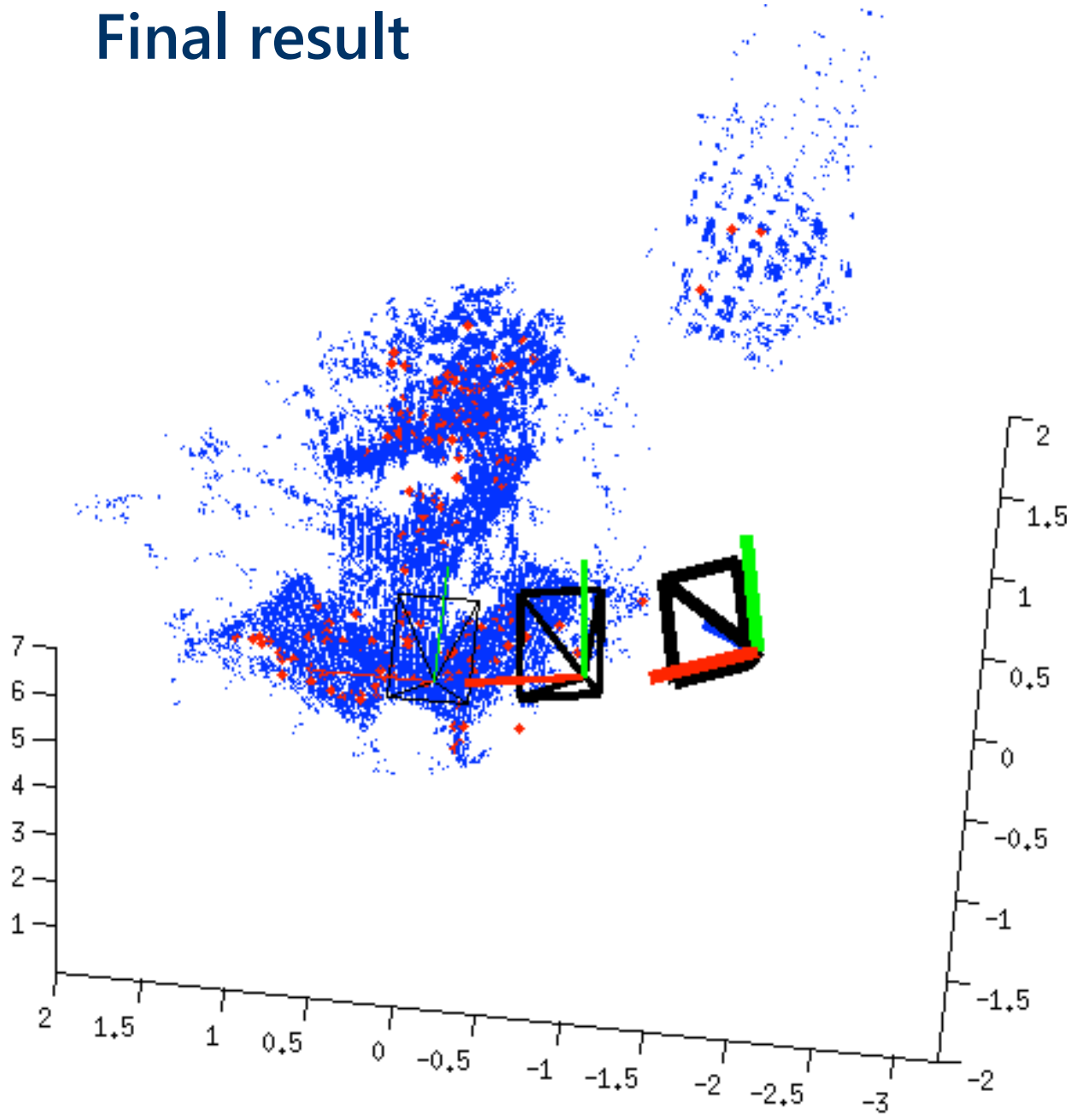


Slide by Szymon Rusinkiewicz and Jianxiong Xiao

Final result

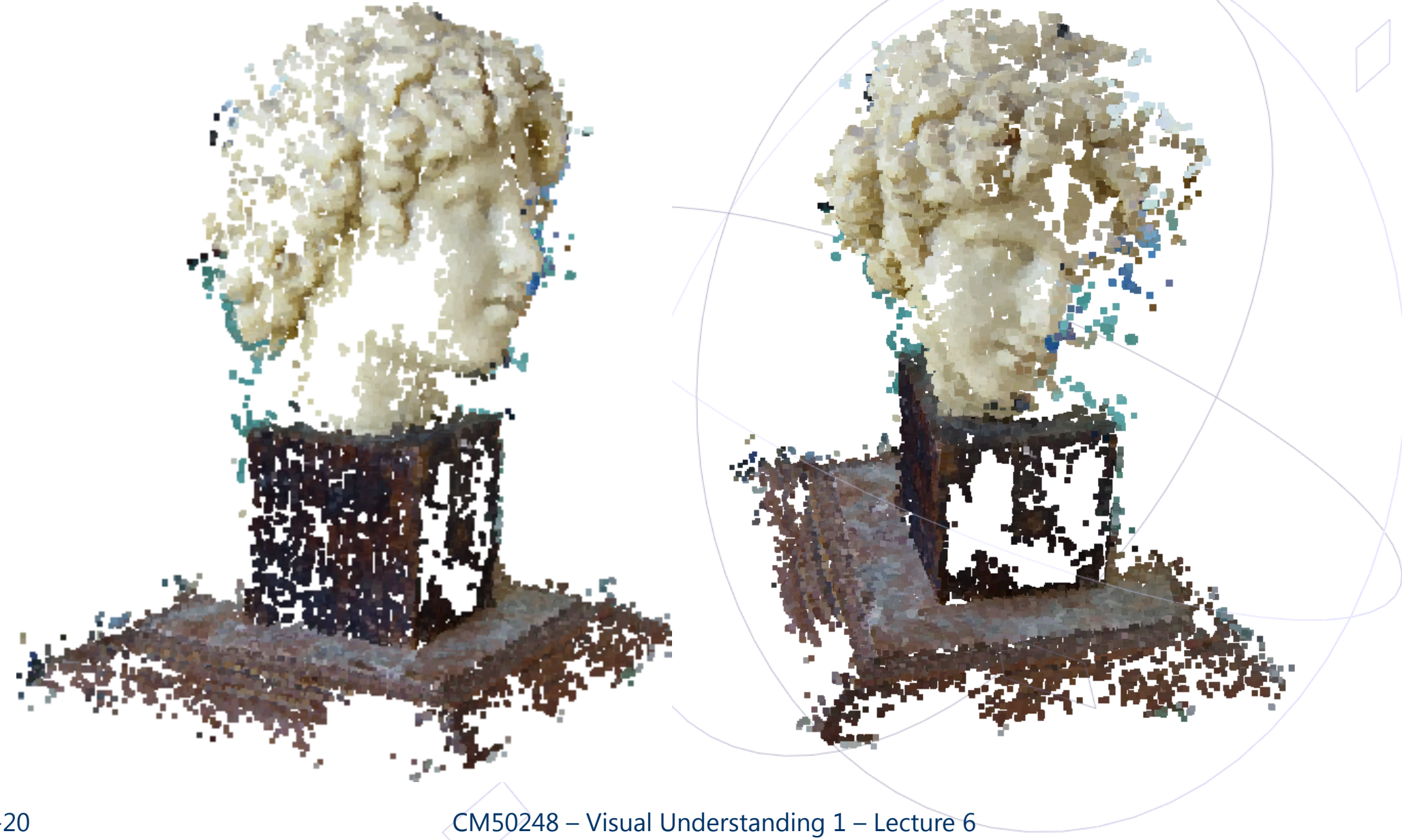






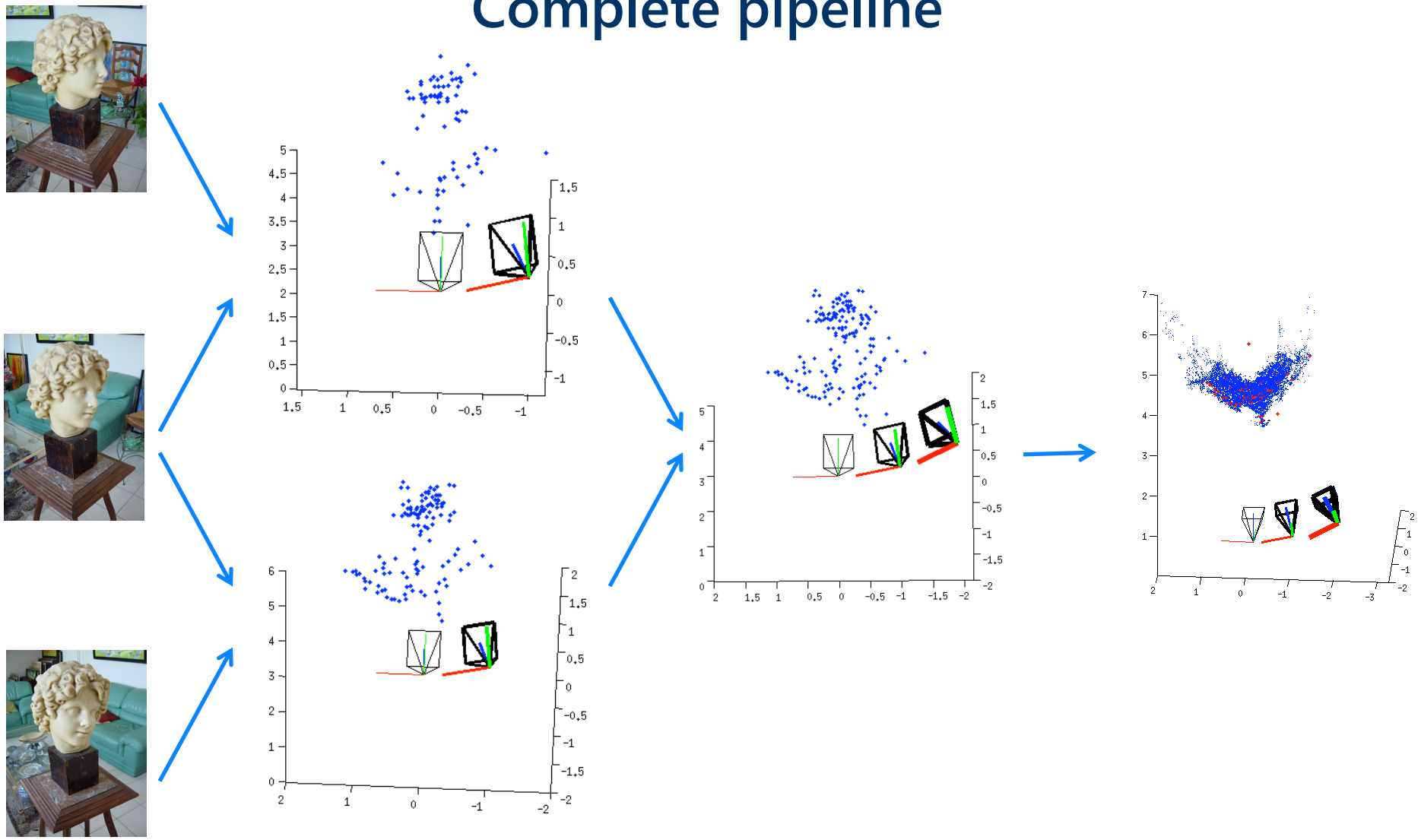
Slide by Szymon Rusinkiewicz and Jianxiong Xiao

Colorize the point cloud



Slide by Szymon Rusinkiewicz and Jianxiong Xiao

Complete pipeline



Structure from Motion (SFM)

Multi-view Stereo (MVS)

Slide by Szymon Rusinkiewicz and Jianxiong Xiao

Wait — How to get the focal length?

- Auto-calibration
  - Self-calibration and metric reconstruction inspite of varying and unknown intrinsic camera parameters, M Pollefeys, R Koch and L Van Gool, IJCV 1999
- Grid search to look for the solution with minimal reprojection error
  - for f=min_f:max_f
  - do everything, then obtain reprojection error after bundle adjustment
- Optimize for this value in bundle adjustment
- Camera calibration (with checkerboard)
- EXIF of JPEG file recorded from digital camera
  - Read Bundler code to understand how to convert EXIF into focal length value http://phototour.cs.washington.edu/bundler/

Slide by Szymon Rusinkiewicz and Jianxiong 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 by Szymon Rusinkiewicz and Jianxiong 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 by Szymon Rusinkiewicz and Jianxiong Xiao

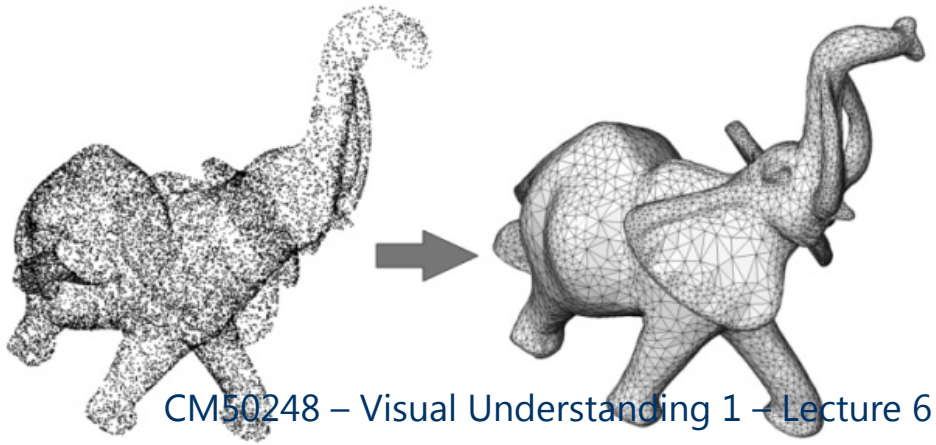
Real-world applications

- Streetview Reconstruction and Recognition
  - http://vision.princeton.edu/projects/2009/ICCV/
  - http://vision.princeton.edu/projects/2009/TOG/
- Photo Tourism
  - http://phototour.cs.washington.edu/
- Microsoft Photosynth
  - http://photosynth.net/
- 2d3, boujou (Matchmovers) and movies
  - http://www.2d3.com/
  - http://www.vicon.com/boujou/
- Robotics: SLAM (Simultaneous Localization And Mapping)
  - http://openslam.org/

Slide by Szymon Rusinkiewicz and Jianxiong 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



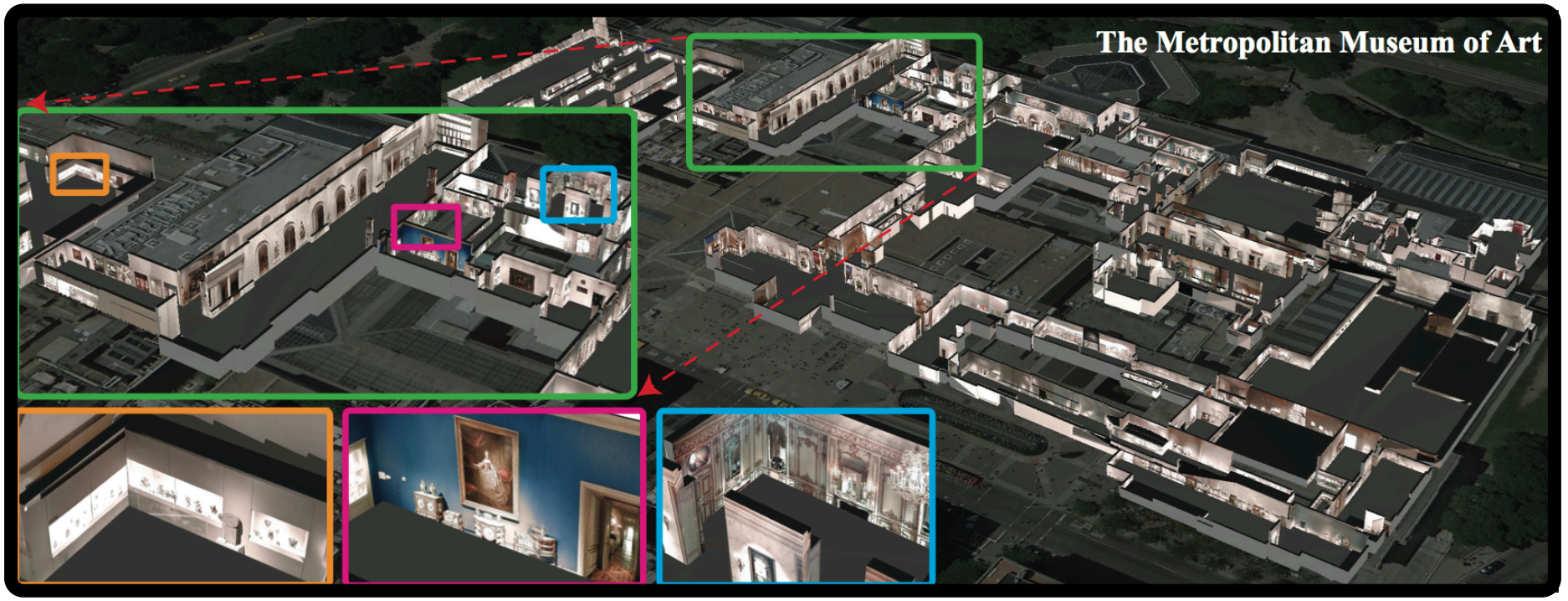
Slide by Szymon Rusinkiewicz and Jianxiong Xiao

Point cloud → 3D mesh model

- Surface reconstruction:
  - Marching cube
    http://en.wikipedia.org/wiki/Marching_cubes
  - Poisson surface reconstruction
    http://research.microsoft.com/en-us/um/people/hoppe/proj/poissonrecon/
- Model fitting:
  - RANSAC & J-linkage
    http://www.diegm.uniud.it/fusiello/demo/jlk/
  - InverseCSG
    http://research.google.com/pubs/pub39963.html
  - GlobFit
    http://code.google.com/p/globfit/

Slide by Szymon Rusinkiewicz and Jianxiong Xiao

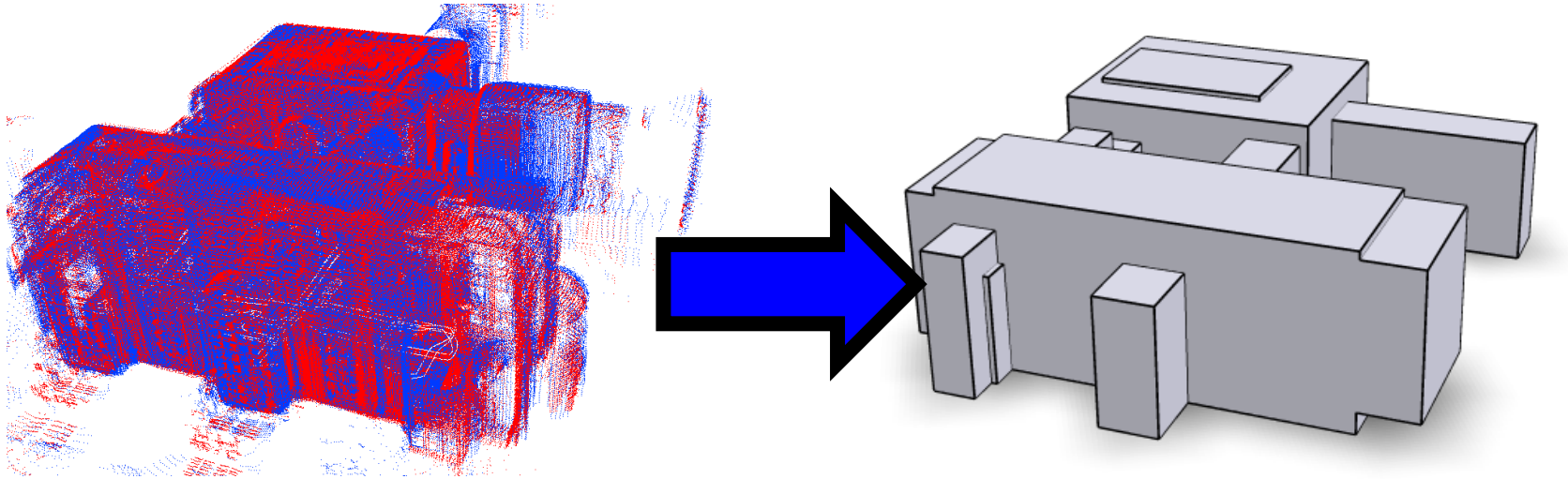
InverseCSG Algorithm



Reconstructing the World's Museums, J. Xiao and Y. Furukawa, 2012.

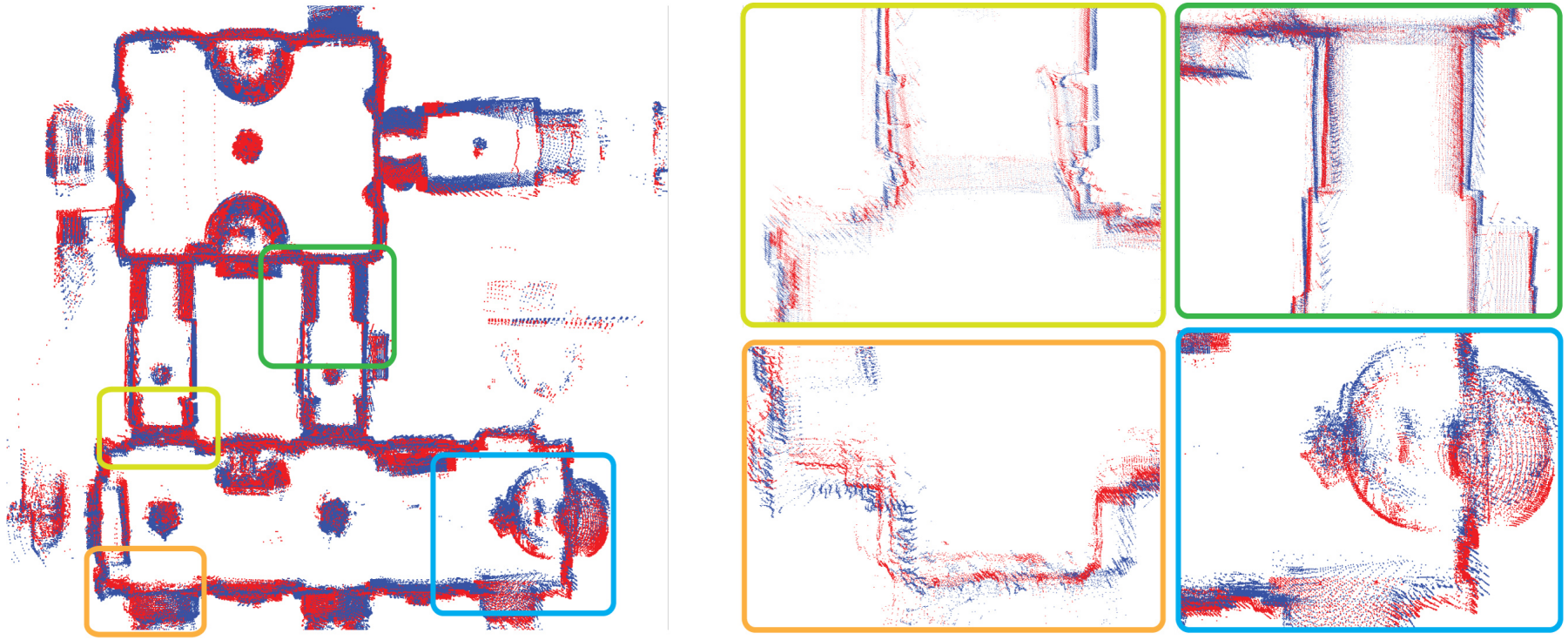
Slide by Szymon Rusinkiewicz and Jianxiong Xiao

InverseCSG Algorithm



Slide by Szymon Rusinkiewicz and Jianxiong Xiao

Noisy Points

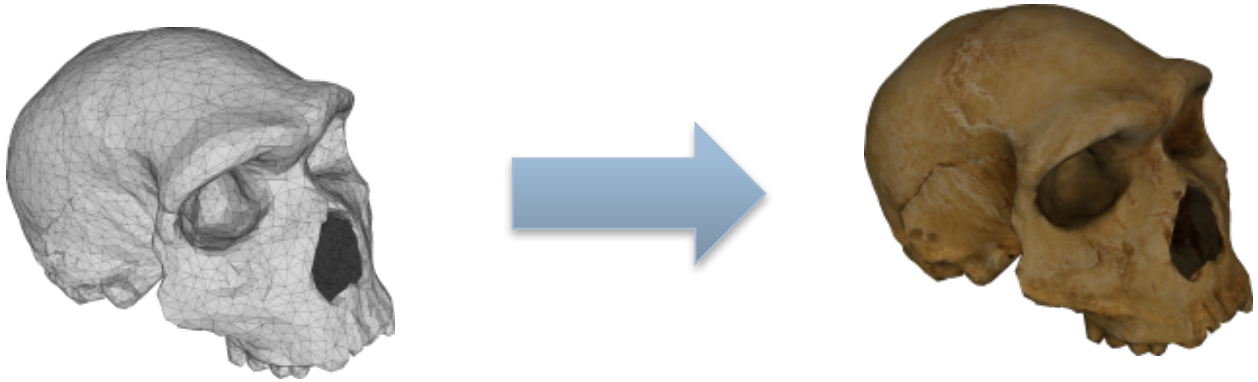


top-down view of input points

Slide by Szymon Rusinkiewicz and Jianxiong 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



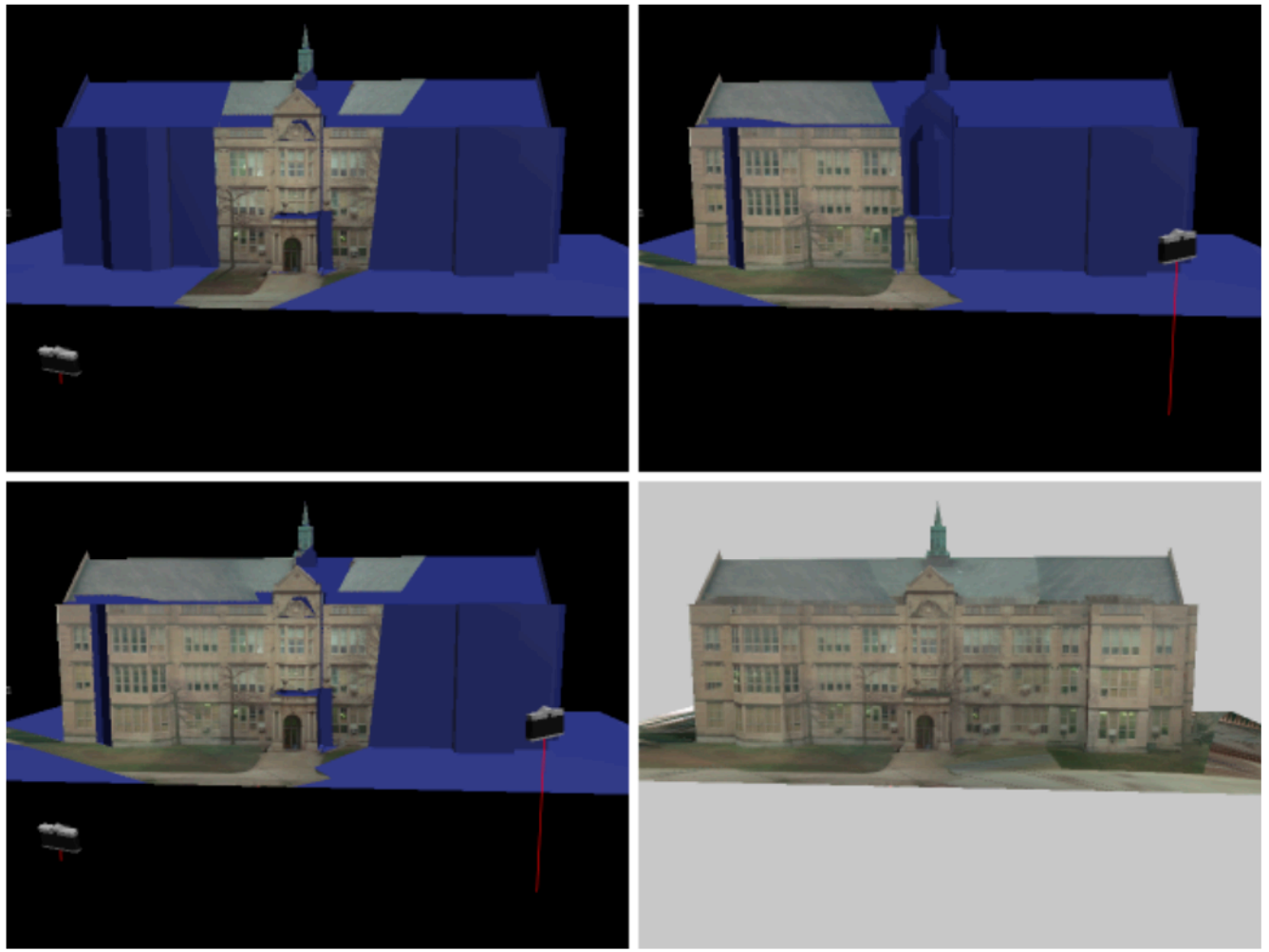
Slide by Szymon Rusinkiewicz and Jianxiong Xiao

Texture mapping

- Texture stitching
- View-based texture mapping
- View interpolation and warping
- Interactive visualization

Slide by Szymon Rusinkiewicz and Jianxiong Xiao

Texture stitching

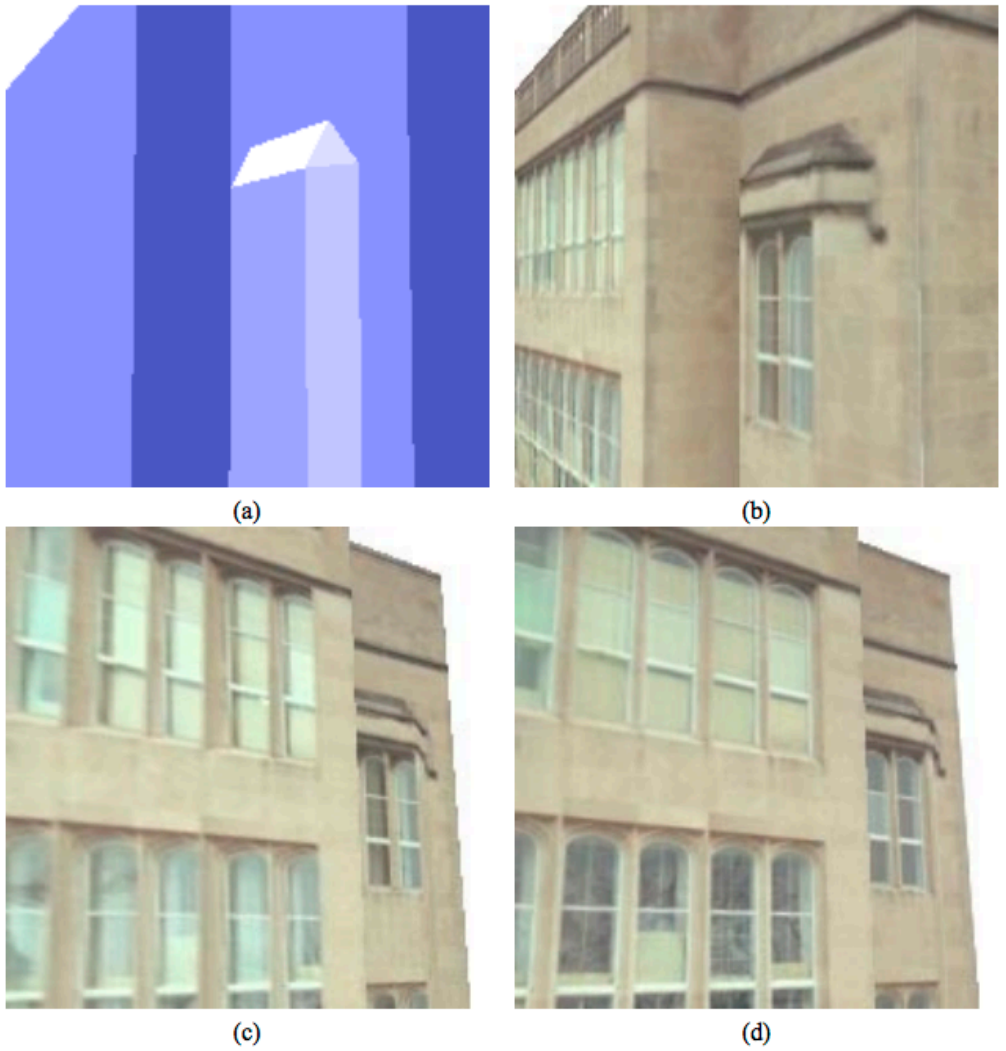


The process of assembling projected images to form a composite rendering

Image from Paul Debevec's Thesis

Slide by Szymon Rusinkiewicz and Jianxiong Xiao

View-Dependent Texture Mapping



Modeling and Rendering Architecture from Photographs,
Paul Debevec, PhD Thesis, UC Berkeley, 1996

Slide by Szymon Rusinkiewicz and Jianxiong Xiao

Q & A

Slide by Szymon Rusinkiewicz and Jianxiong Xiao

Recognition

- How about something completely different?

Bag of words

- Which is the vision text book?

Document A

13747	the
12479	and
7765	a
7423	of
…	
2456	image
…	
1711	vision
…	
919	algorithms
…	
773	ieee
…	
699	recognition
…	

Document B

18062	the
8969	and
7144	a
6695	to
…	
1532	lord
…	
1311	ser
…	
800	jon
…	
745	ned
…	
590	men
579	like
572	tyrion

http://www.writewords.org.uk/word_count.asp

Bag of words

- Two-stage technique:
  - Learning/Training
  - Testing/Application

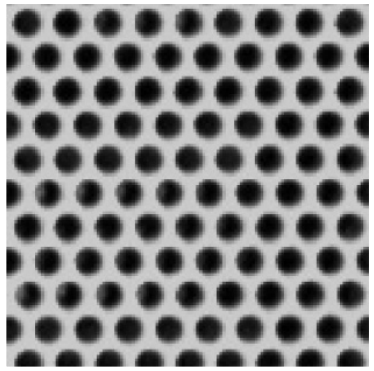
- Databases of classification test data (with ground truth)

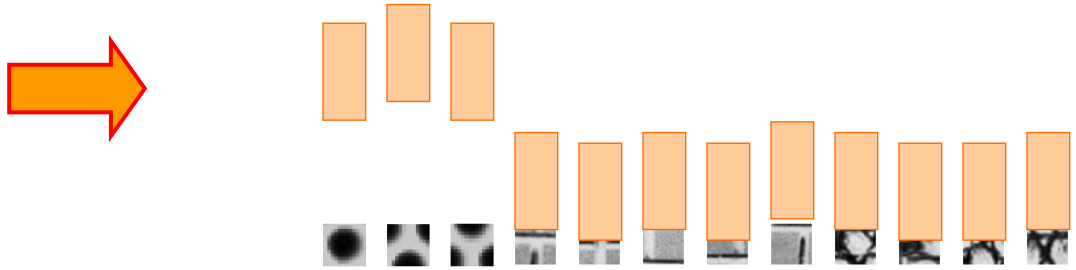
Bag of words

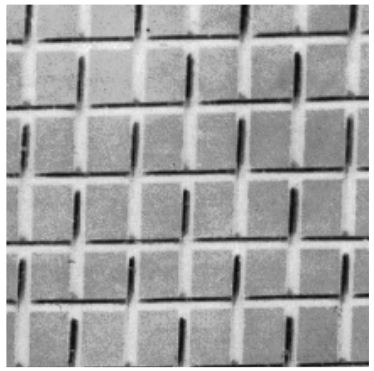
- Create a shared dictionary (English language)
- Find histogram of word counts
  - Including words that aren't present
- Classify based on histogram

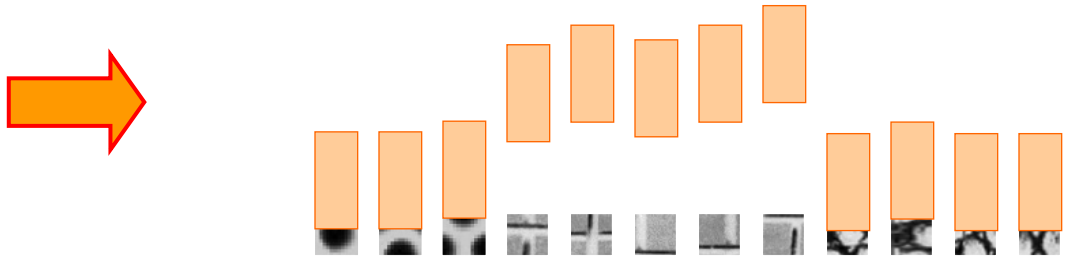
- "ser" → Game of Thrones
- "recognition" → Vision text-book

Bag of features









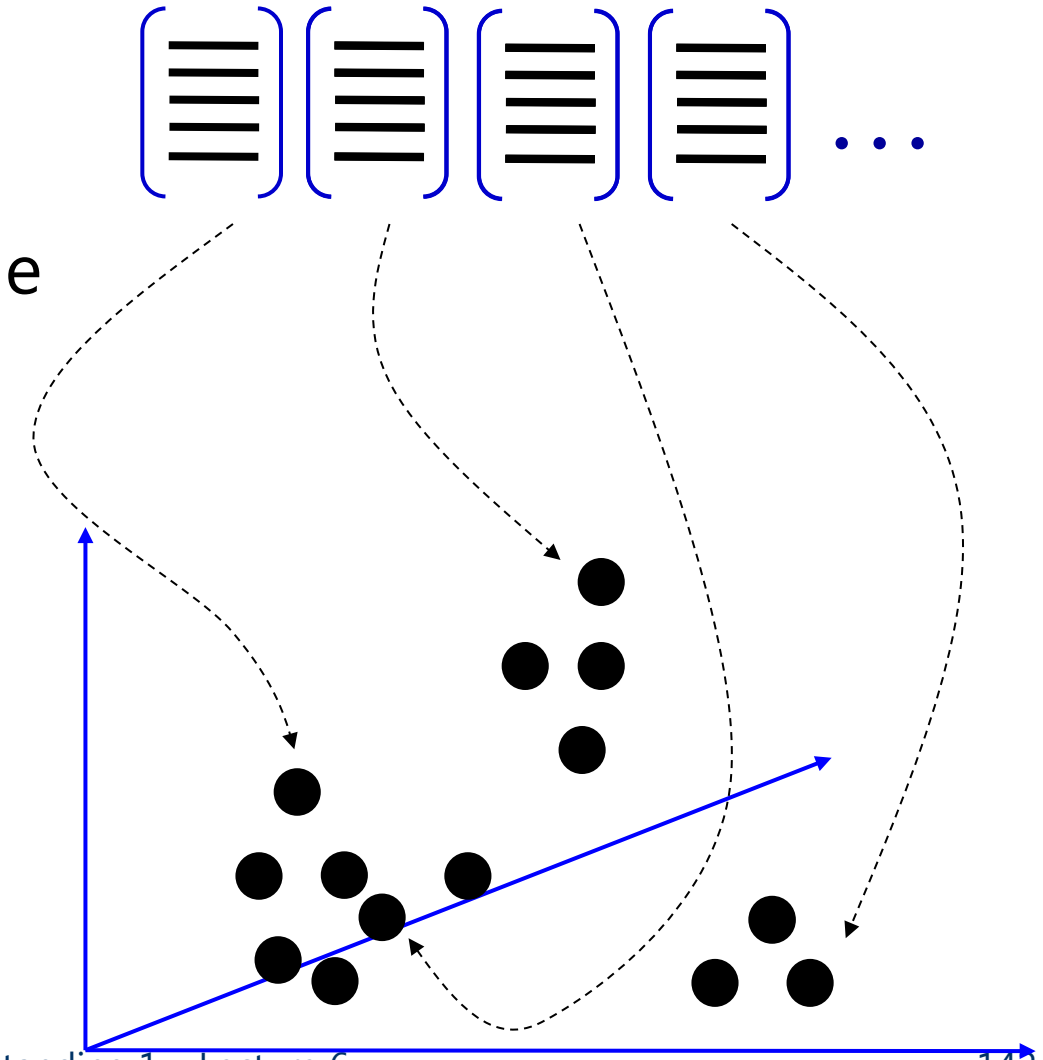
Slide by Cordelia Schmid

Bag of features

Imagine features in n-dimensional space, one dimension per element in the vector.

SIFT features have 128 elements.

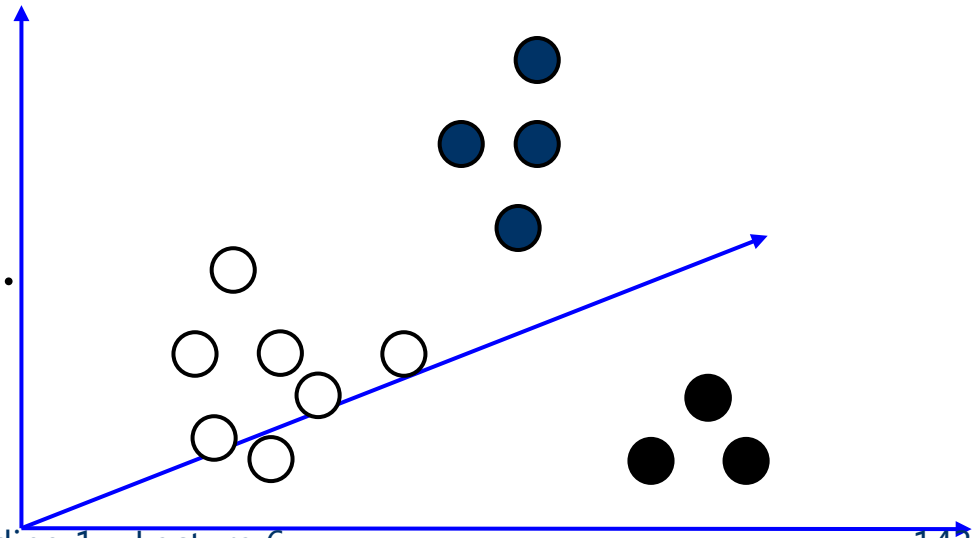
Higher dimensional representations are very useful in vision.



Bag of features

Cluster the points in n-space. This is the dictionary equivalent. Each cluster represents a word.

For each image, work out the number of instances of each word (features that belong to the given cluster). Use this to build the histogram describing the image.



Bag of features

- Now your image is represented by a histogram of words, another vector, probably 1000+ elements.
- Train a Support Vector Machine to classify.

Questions?

Well done!

You've made it to the end!