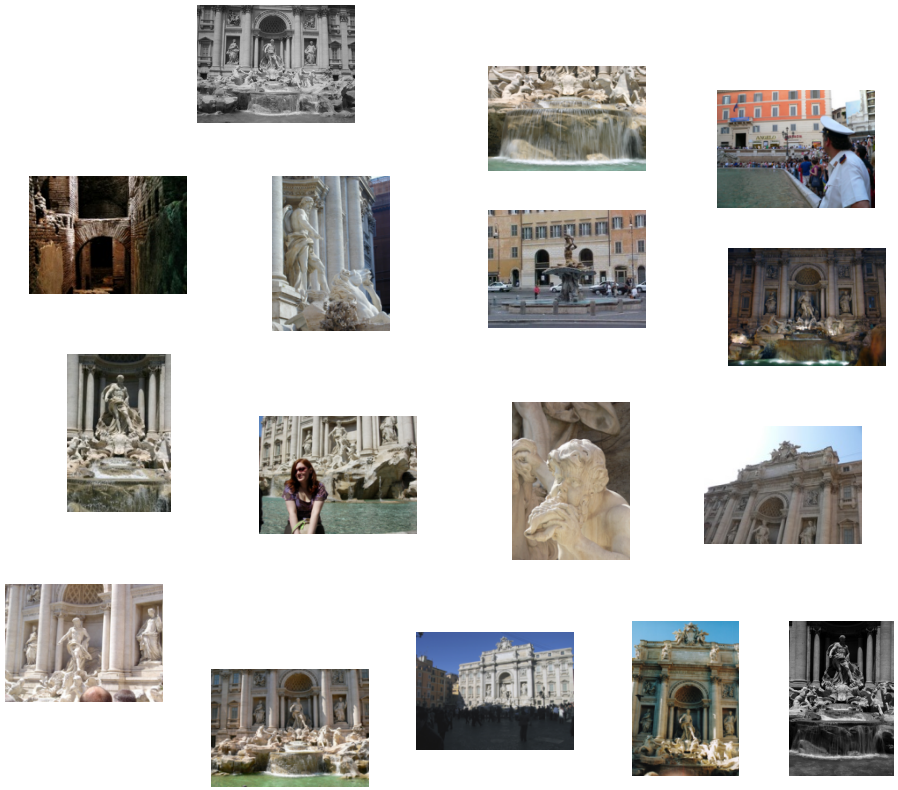
Stereo

Slides by courtesy to Manmohan Chandraker

Recap

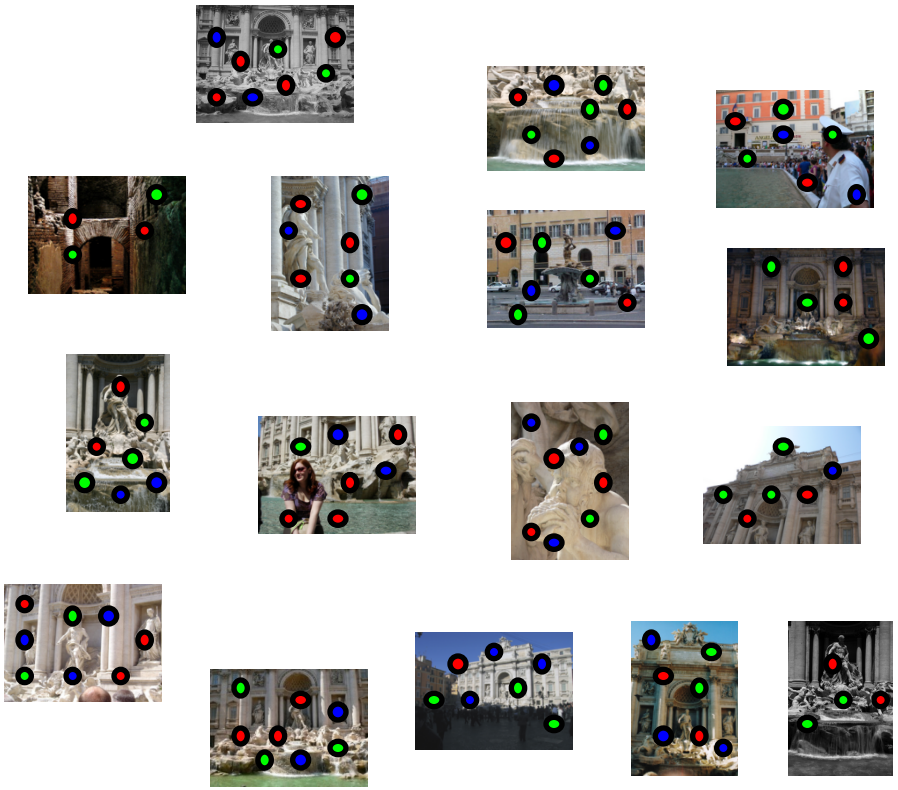
Structure from Motion

Several images observe a scene from different viewpoints



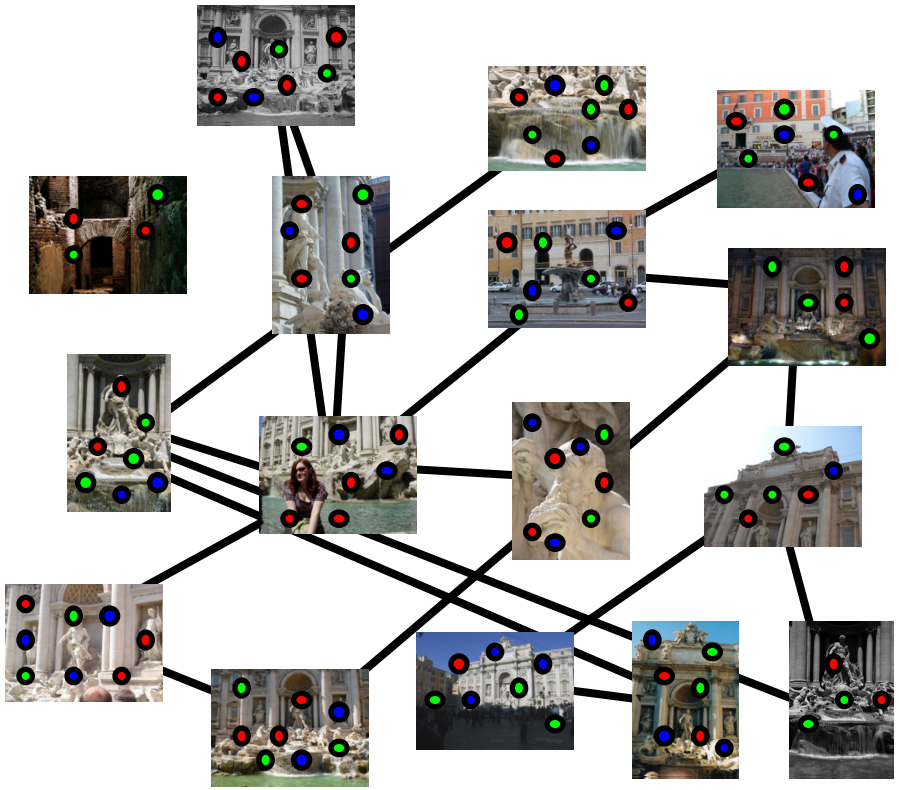
Feature detection

Detect features using, for example, SIFT [Lowe, IJCV 2004]

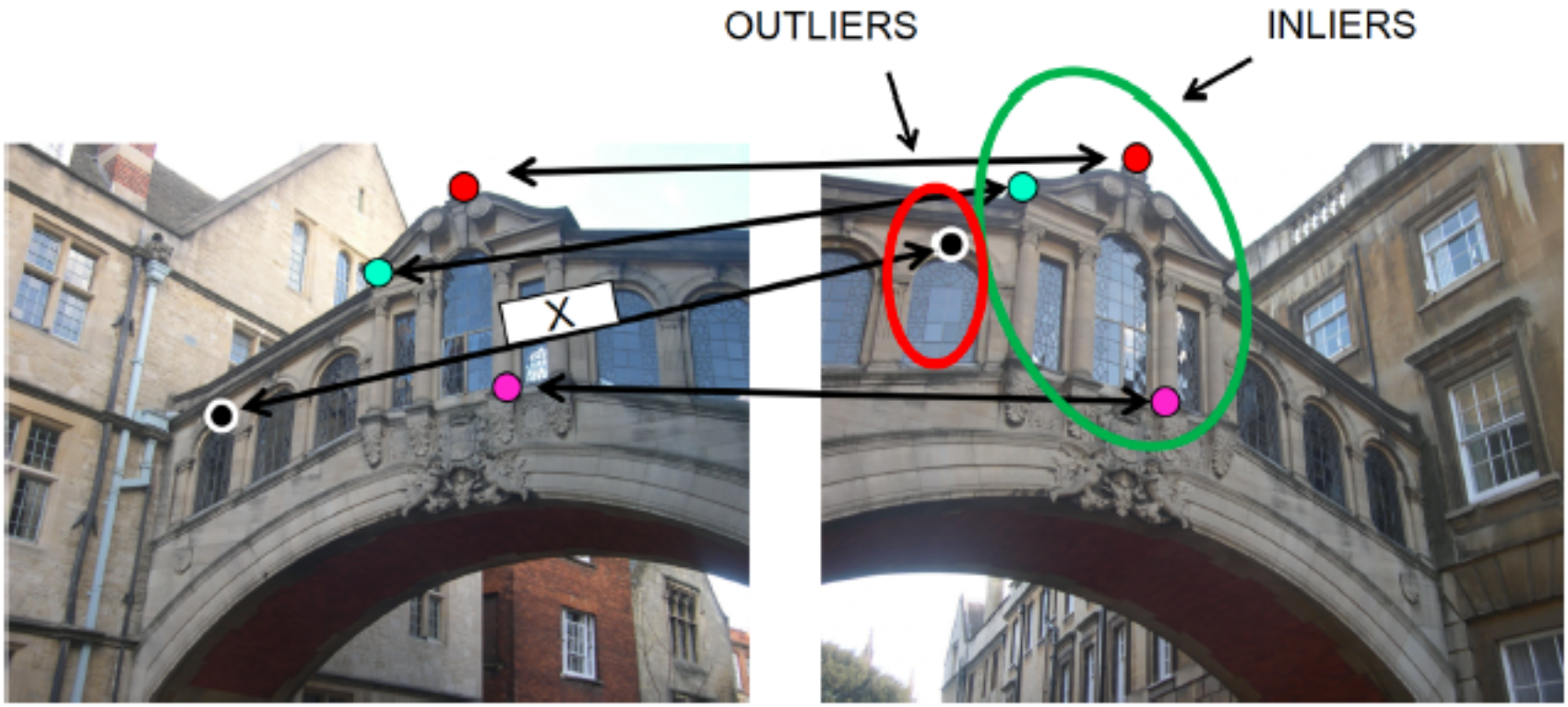


Feature matching

Match features between each pair of images

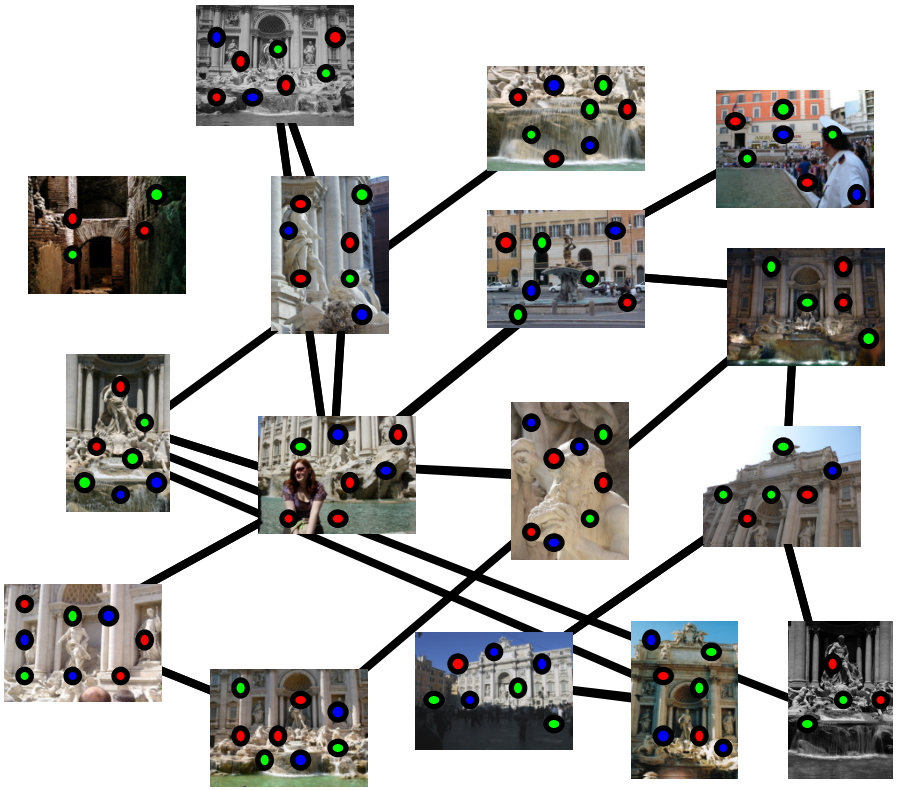


Feature matching

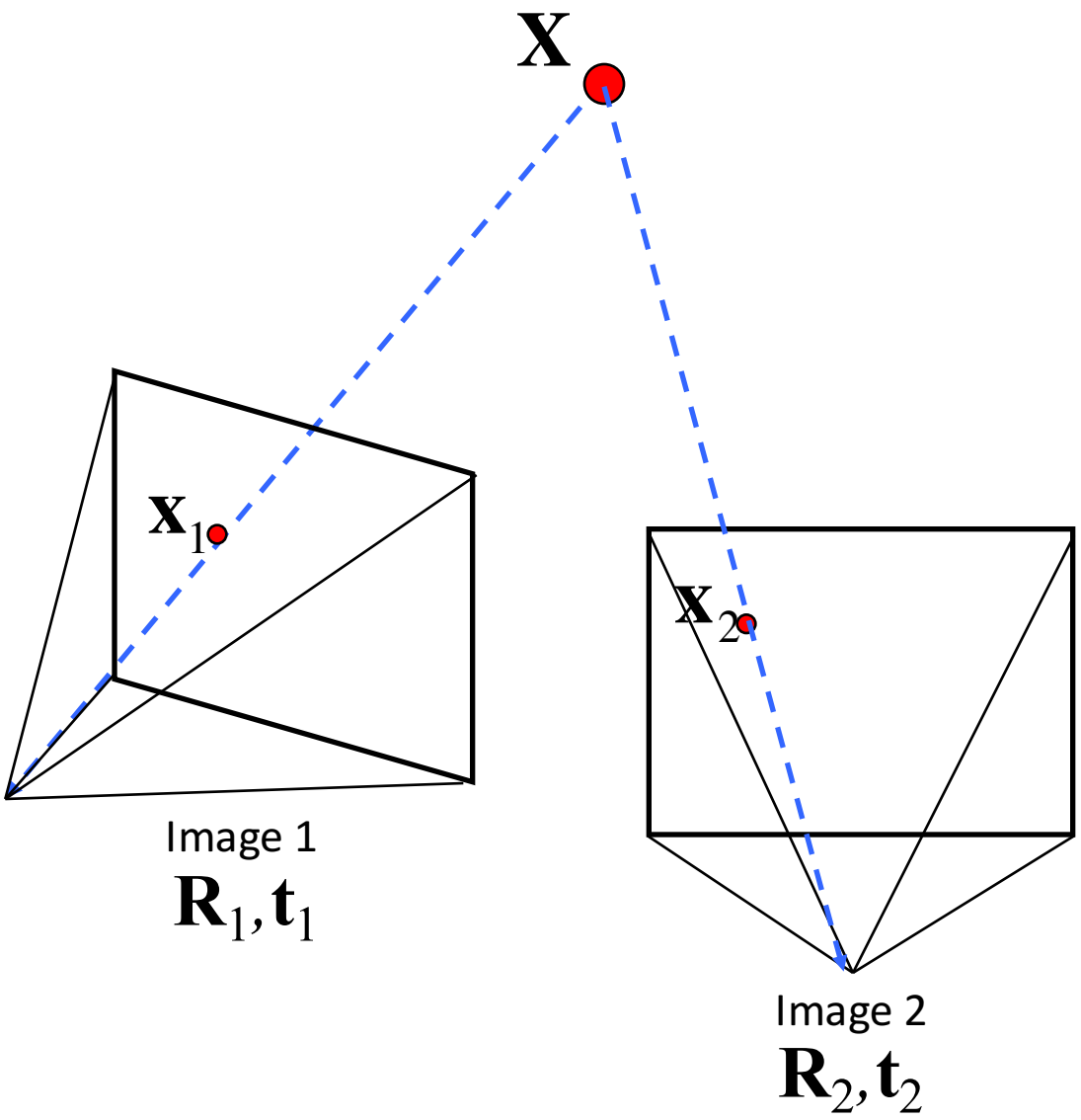


Feature matching

Refine matching using RANSAC to estimate fundamental matrix between each pair



Fundamental Matrix



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

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

8-point algorithm

Given *n* point correspondences, set up a system of equations:

$$\begin{pmatrix} u_1u_1' & v_1u_1' & u_1' & u_1v_1' & v_1v_1' & v_1' & u_1 & v_1 & 1 \\ u_2u_2' & v_2u_2' & u_2' & u_2v_2' & v_2v_2' & v_2' & u_2 & v_2 & 1 \\ \vdots & \vdots & \vdots & \vdots & \vdots & \vdots & \vdots & \vdots & \vdots \\ u_nu_n' & v_nu_n' & u_n' & u_nv_n' & v_nv_n' & v_n' & u_n & v_n & 1 \end{pmatrix} \begin{pmatrix} f_{11} \\ f_{12} \\ f_{13} \\ f_{21} \\ f_{22} \\ f_{23} \\ f_{31} \\ f_{32} \\ f_{33} \end{pmatrix} = 0$$

- In reality, instead of solving $\mathbf{Af} = 0$, we seek $\mathbf{f}$ to minimize $\|\mathbf{Af}\|$.

8-point algorithm – Problem?

- **F** should have rank 2
- To enforce that **F** is of rank 2, **F** is replaced by **F'** that minimizes $\|\mathbf{F}^\top \mathbf{F}'\|$ subject to the rank constraint.

- This is achieved by SVD. Let $\mathbf{F} = \mathbf{U}\Sigma\mathbf{V}$, where

$$\Sigma = \begin{pmatrix} s_1 & 0 & 0 \\ 0 & s_2 & 0 \\ 0 & 0 & s_3 \end{pmatrix}. \text{ Let } \quad \Sigma' = \begin{pmatrix} s_1 & 0 & 0 \\ 0 & s_2 & 0 \\ 0 & 0 & 0 \end{pmatrix}$$

then $\mathbf{F}' = \mathbf{U}\Sigma'\mathbf{V}$ is the solution.

RANSAC

1. Randomly choose *s* samples (correspondences)
   - For fundamental matrix, what is the size of s?
2. Fit the model (fundamental matrix) to those samples
   - How would you fit the model to s correspondences?
3. Count the number of inliers among all other correspondences
   - How can you determine which points are inliers?
4. Repeat *N* times
   - How do you choose *N*?
5. Choose the model with the largest set of inliers

Estimate camera motion from F

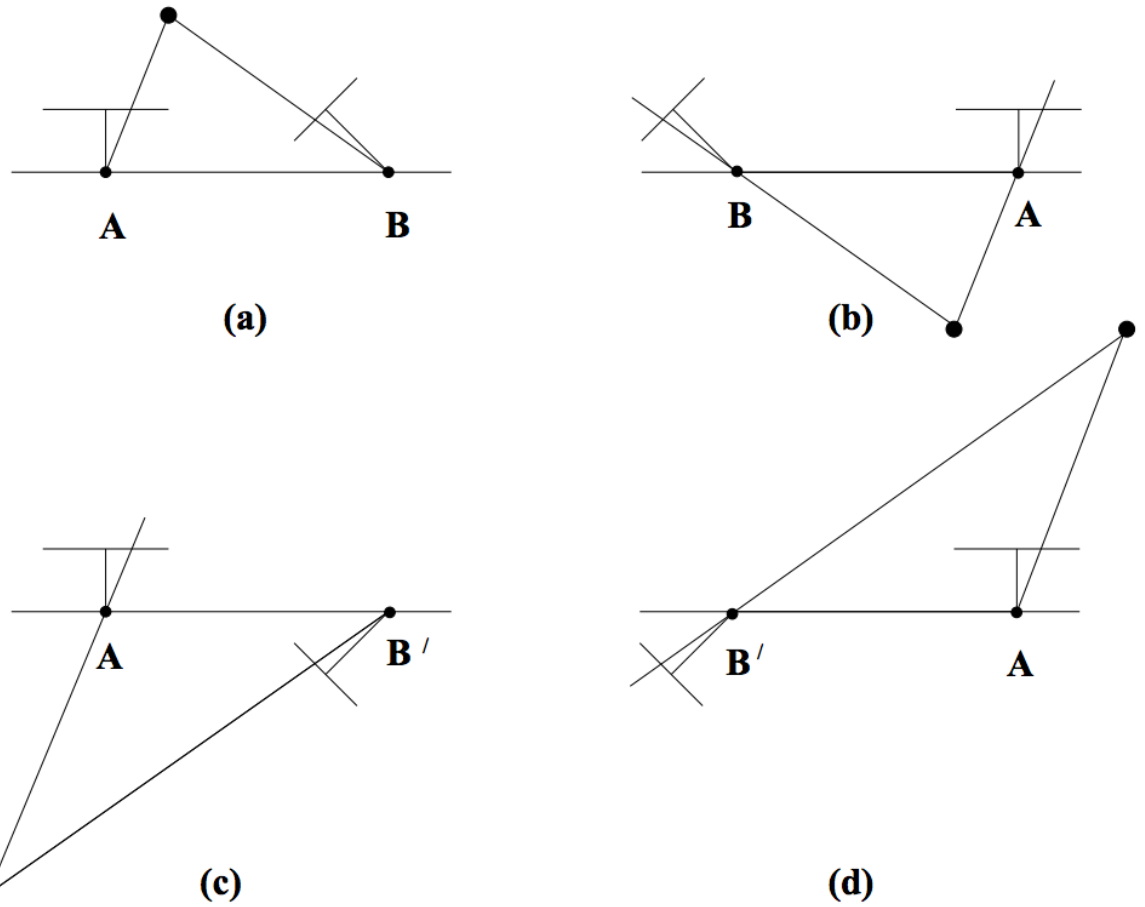
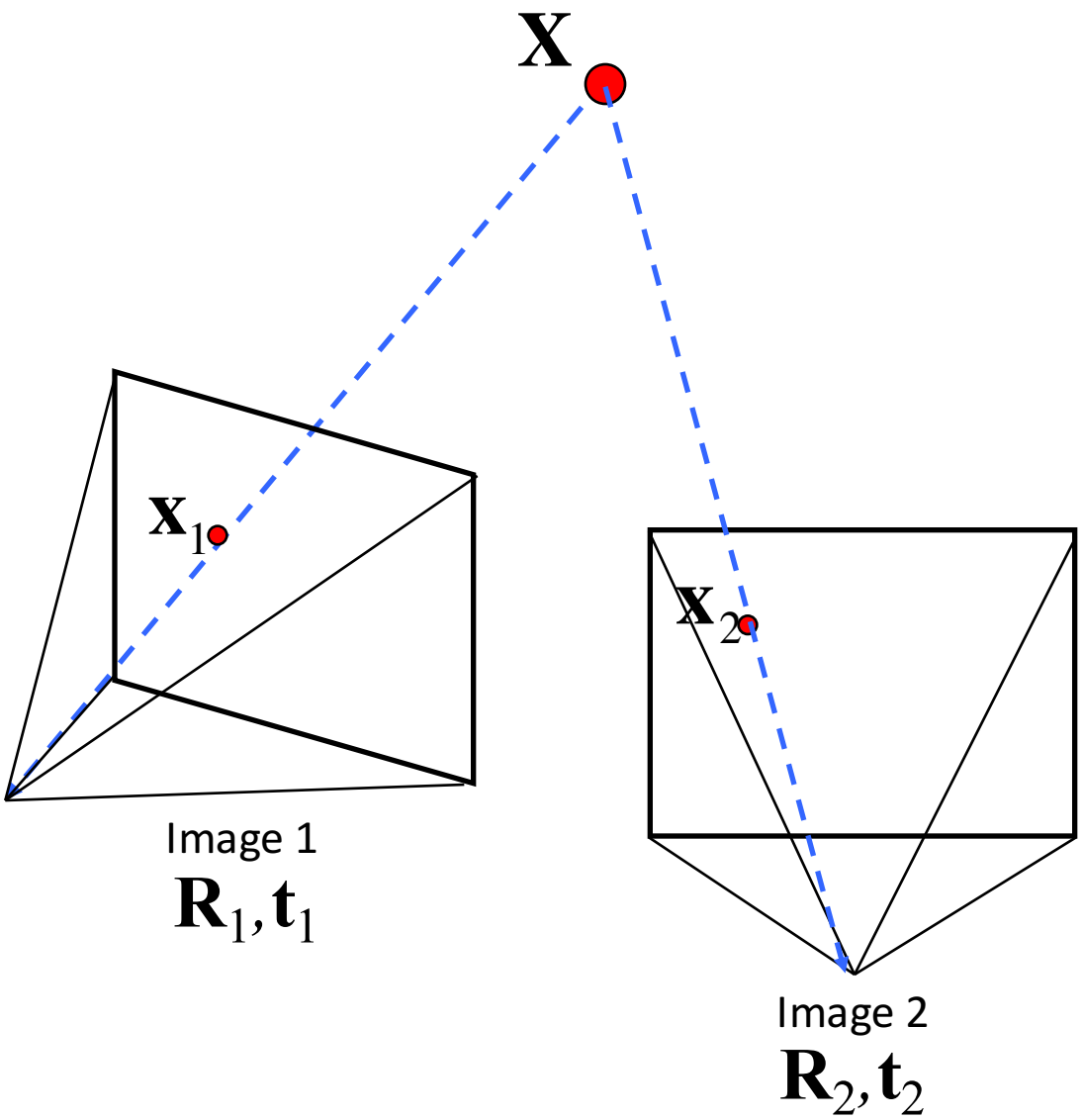
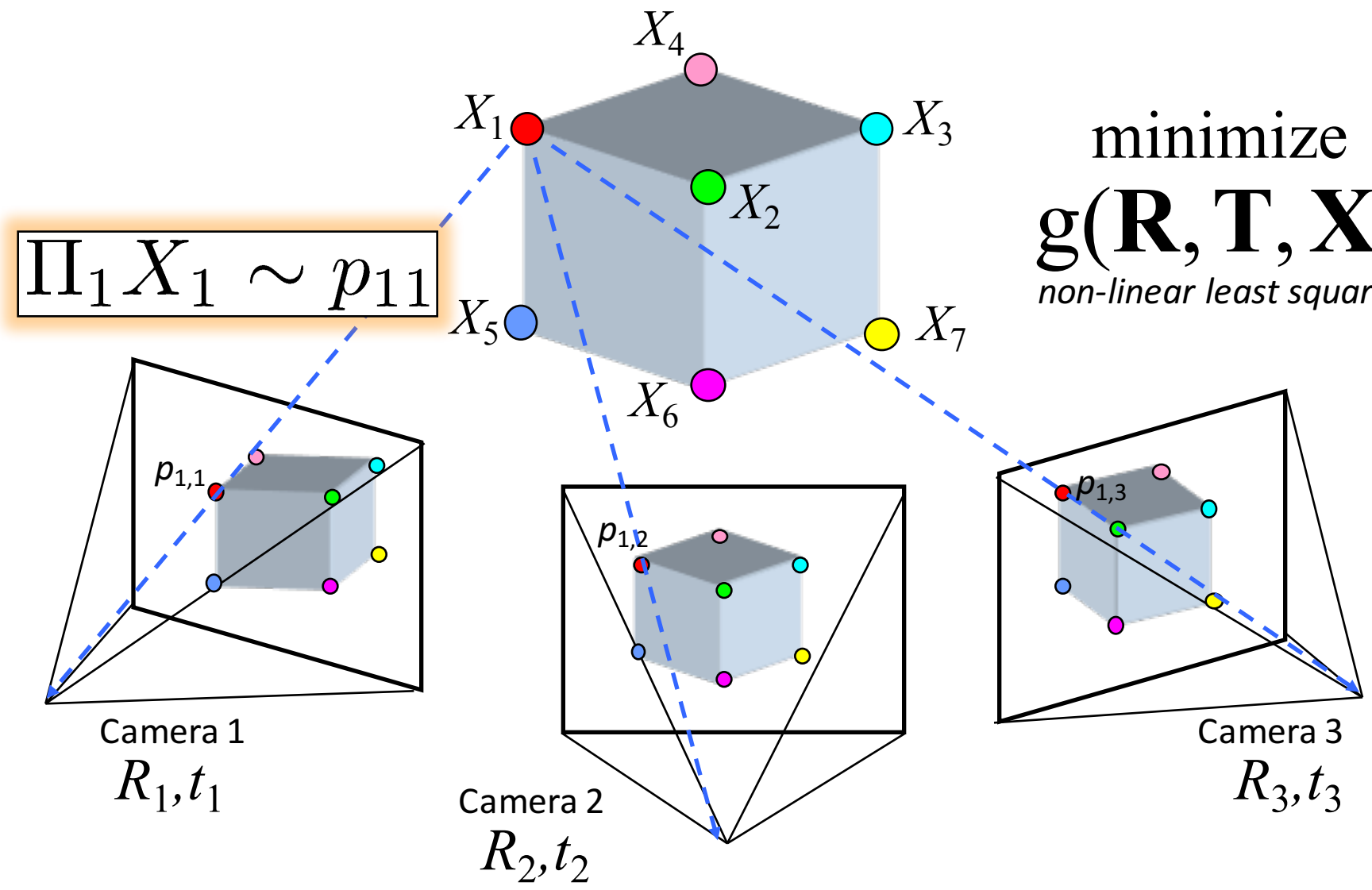


Fig. 9.12. **The four possible solutions for calibrated reconstruction from E.** *Between the left and right sides there is a baseline reversal. Between the top and bottom rows camera B rotates* 180° *about the baseline. Note, only in (a) is the reconstructed point in front of both cameras.*

Triangulation to get 3D points



Structure from motion



Bundle adjustment

- Minimize sum of squared reprojection errors:

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

indicator variable: whether point *i* visible in image *j*

predicted image location

observed image location

  - Optimized with non-linear least squares
  - Levenberg-Marquardt is a popular choice
- Practical challenges?
  - Initialization
  - Outliers

Benchmark datasets

The KITTI Vision Benchmark Suite

A project of Karlsruhe Institute of Technology and Toyota Technological Institute at Chicago

Karlsruhe Institute of Technology

home | setup | stereo | flow | scene flow | odometry | object | tracking | road | semantics | raw data | submit results | jobs

Andreas Geiger (MPI Tübingen) | Philip Lenz (KIT) | Christoph Stiller (KIT) | Raquel Urtasun (University of Toronto)

Visual Odometry / SLAM Evaluation 2012

The odometry benchmark consists of 22 stereo sequences, saved in loss less png format: We provide 11 sequences (00-10) with ground truth trajectories for training and 11 sequences (11-21) without ground truth for evaluation. For this benchmark you may provide results using monocular or stereo visual odometry, laser-based SLAM or algorithms that combine visual and LIDAR information. The only restriction we impose is that your method is fully automatic (e.g., no manual loop-closure tagging is allowed) and that the same parameter set is used for all sequences. A development kit provides details about the data format.

- Download odometry data set (grayscale, 22 GB)
- Download odometry data set (color, 65 GB)
- Download odometry data set (velodyne laser data, 80 GB)
- Download odometry data set (calibration files, 1 MB)
- Download odometry ground truth poses (4 MB)
- Download odometry development kit (1 MB)

Benchmark datasets

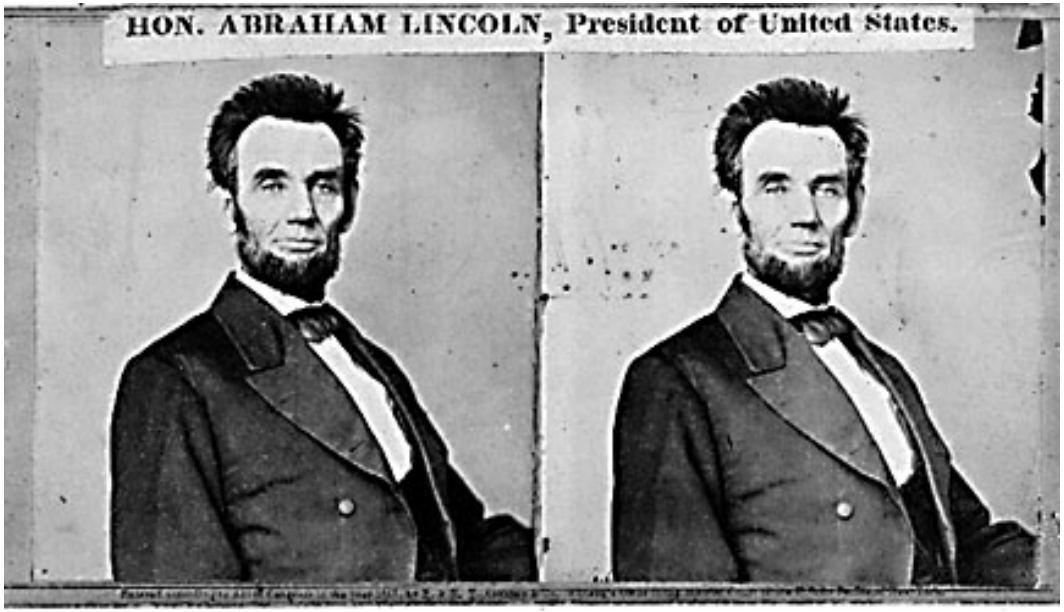
Additional information used by the methods

- Stereo: Method uses left and right (stereo) images
- Laser Points: Method uses point clouds from Velodyne laser scanner
- Loop Closure Detection: This method is a SLAM method that detects loop closures
- Additional training data: Use of additional data sources for training (see details)

	Method	Setting	Code	Translation	Rotation	Runtime	Environment	Compare
1	V-LOAM			**0.68 %**	**0.0015 [deg/m]**	0.1 s	2 cores @ 2.5 Ghz (C/C++)	
J. Zhang and S. Singh: Visual-lidar Odometry and Mapping: Low drift, Robust, and Fast. IEEE International Conference on Robotics and Automation(ICRA) 2015.								
2	LOAM			0.70 %	0.0017 [deg/m]	0.1 s	2 cores @ 2.5 Ghz (C/C++)	
J. Zhang and S. Singh: LOAM: Lidar Odometry and Mapping in Real-time. Robotics: Science and Systems Conference (RSS) 2014.								
3	SOFT2			0.72 %	0.0016 [deg/m]	0.1 s	2 cores @ 2.5 Ghz (C/C++)	
4	LG-SLAM			0.82 %	0.0020 [deg/m]	0.2 s	2 cores @ 2.5 Ghz (C/C++)	
5	GDVO			0.86 %	0.0031 [deg/m]	0.09 s	1 core @ >3.5 Ghz (C/C++)	
6	HypERROCC			0.88 %	0.0027 [deg/m]	0.25 s	2 cores @ 2.0 Ghz (C/C++)	
7	SOFT			0.88 %	0.0022 [deg/m]	0.1 s	2 cores @ 2.5 Ghz (C/C++)	
I. Cvišić and I. Petrović: Stereo odometry based on careful feature selection and tracking. European Conference on Mobile Robots (ECMR) 2015.								
8	RotRocc			0.88 %	0.0025 [deg/m]	0.3 s	2 cores @ 2.0 Ghz (C/C++)	
M. Buczko and V. Willert: Flow-Decoupled Normalized Reprojection Error for Visual Odometry. 19th IEEE Intelligent Transportation Systems Conference (ITSC) 2016.								
9	EDVO			0.89 %	0.0030 [deg/m]	0.1 s	1 core @ 2.5 Ghz (C/C++)	
10	svo2			0.94 %	0.0021 [deg/m]	0.2 s	1 core @ 2.5 Ghz (C/C++)	

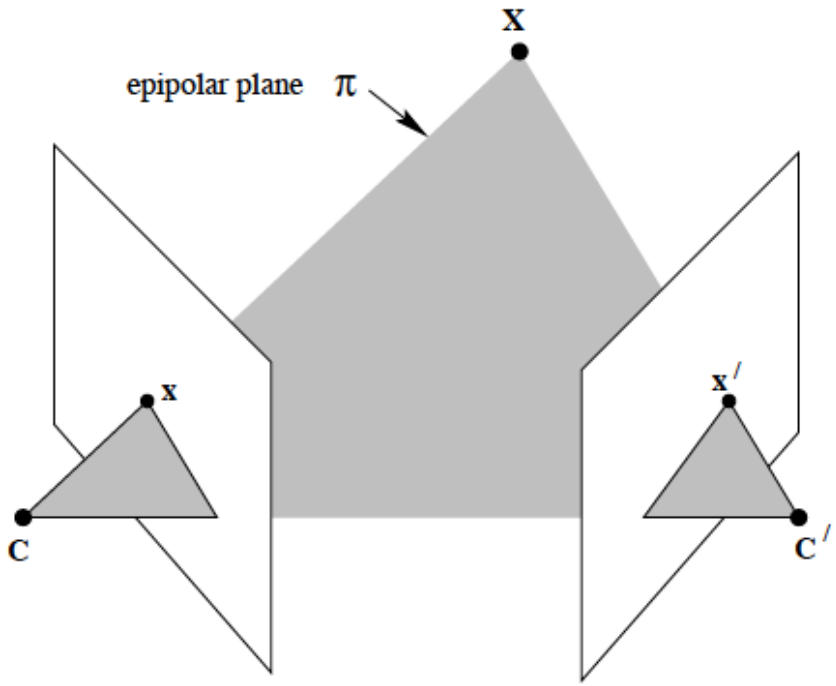
Two-View Stereo

Stereo

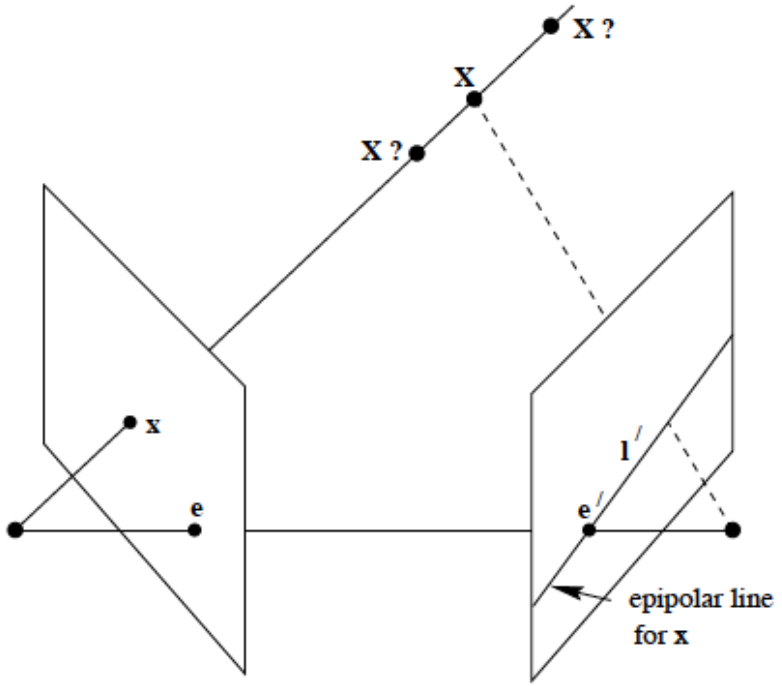


- Given two images from different viewpoints
  - How can we compute the depth of each point in the image?
  - Based on *how much each pixel moves* between the two images
  - We already saw that correspondence allows estimating depth.

Epipolar Geometry

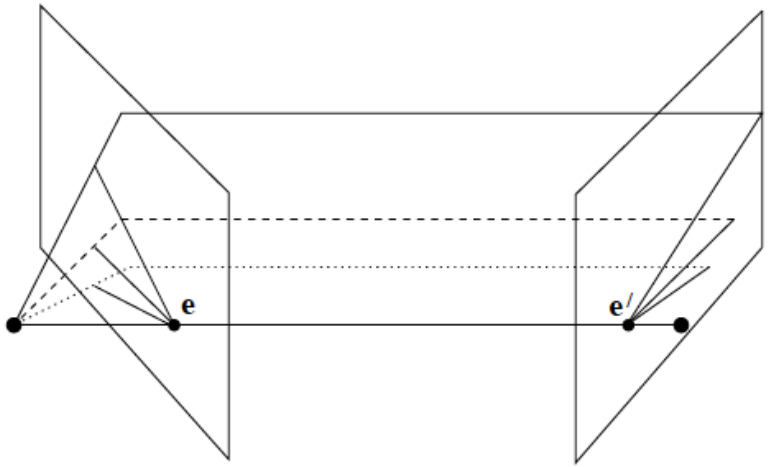


a

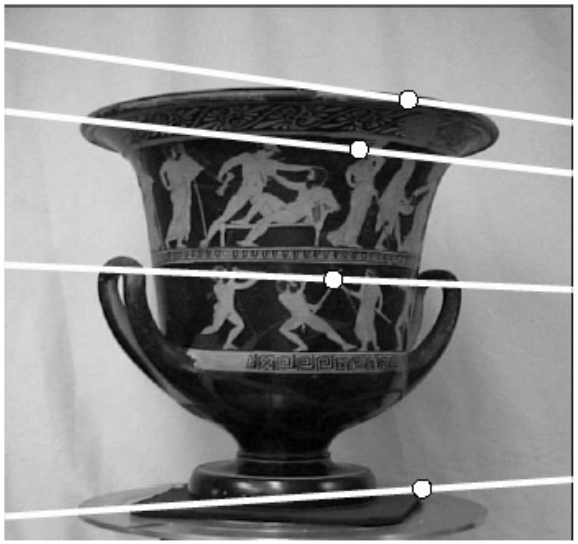


b

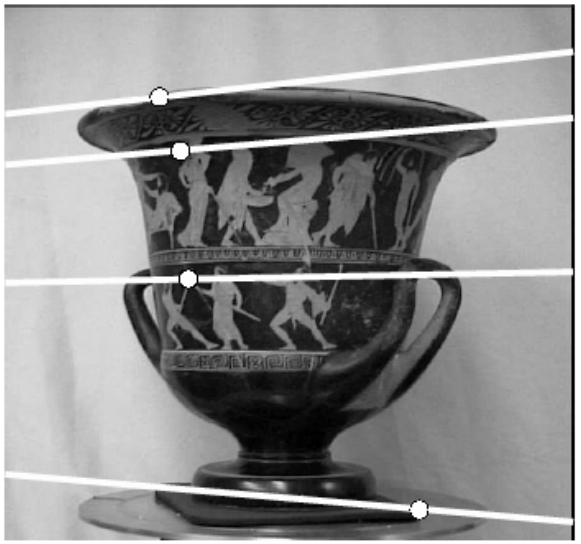
Epipolar Geometry



a



b

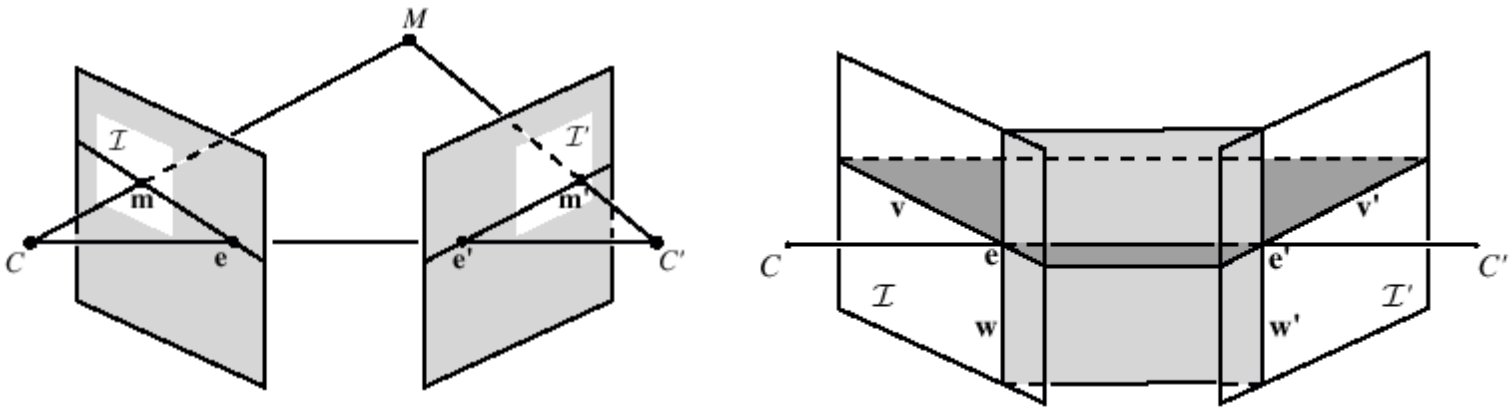


c

Rectification

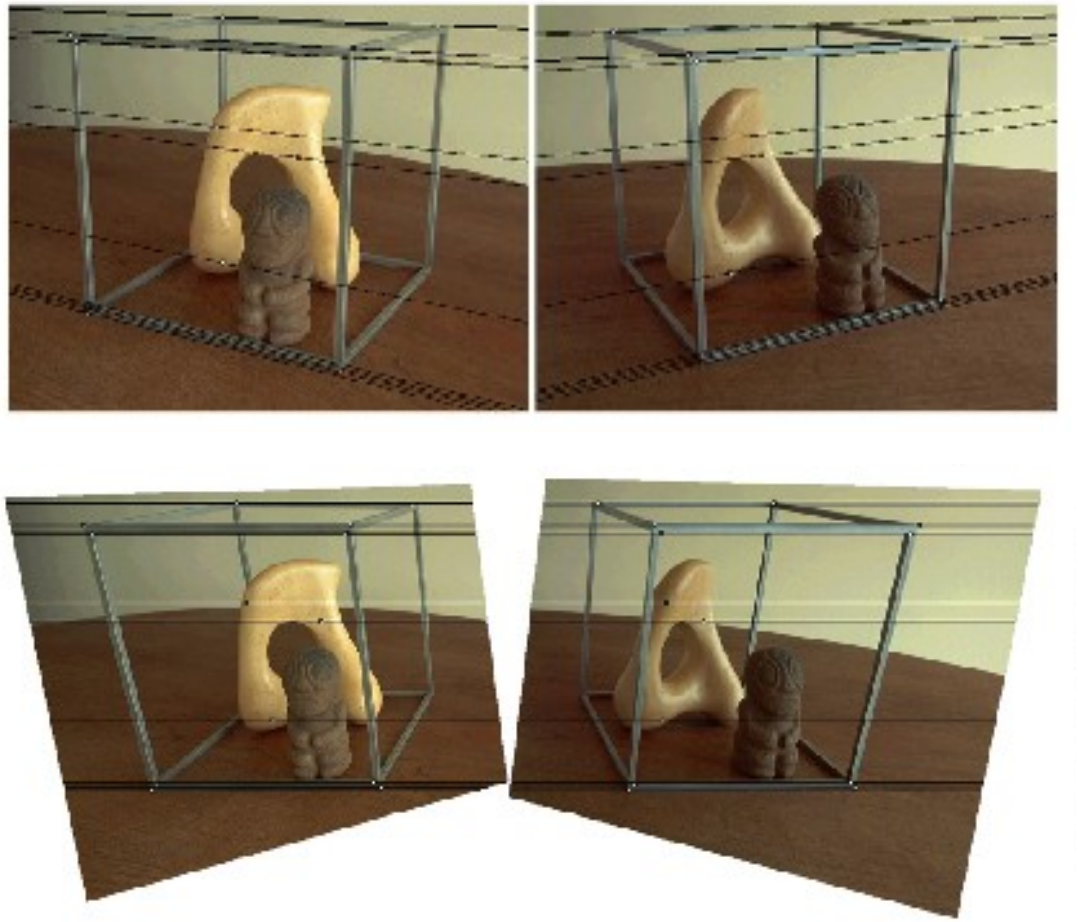
Project each image onto same plane, which is parallel to the epipoles

Resample lines (and shear or stretch) to place lines in correspondence and minimize distortion



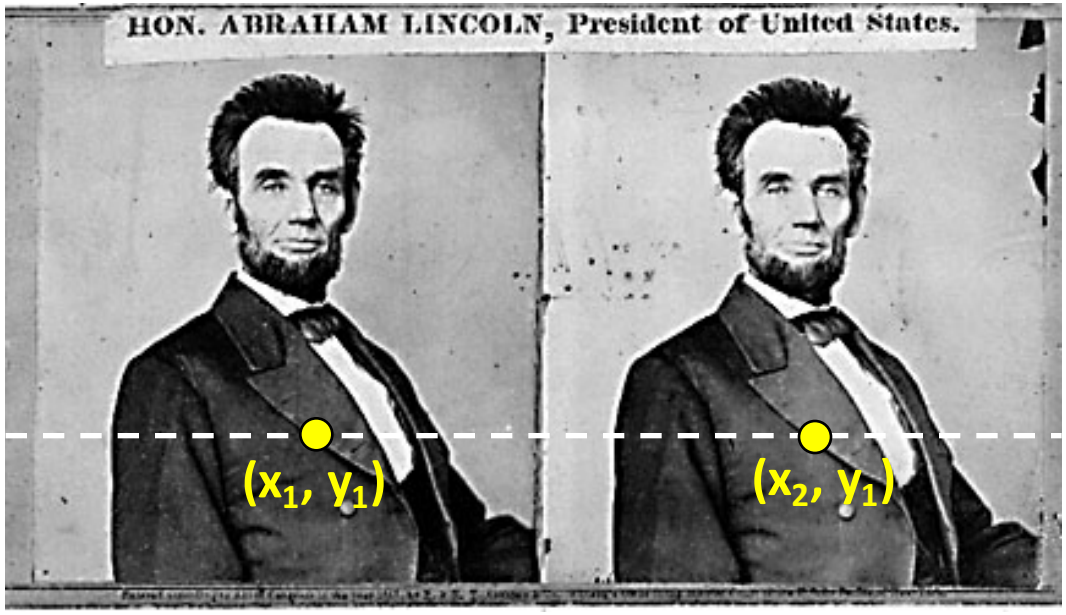
[Loop and Zhang, CVPR'99]

Rectification



Disparity

epipolar lines

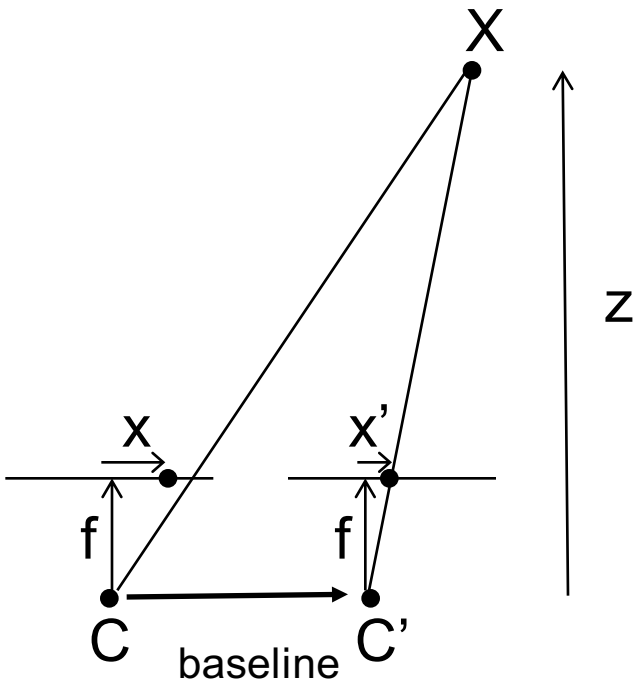


Two images captured by a purely horizontal translating camera (*rectified* stereo pair)

$x_2 - x_1$ = the *disparity* of pixel (x_1, y_1)

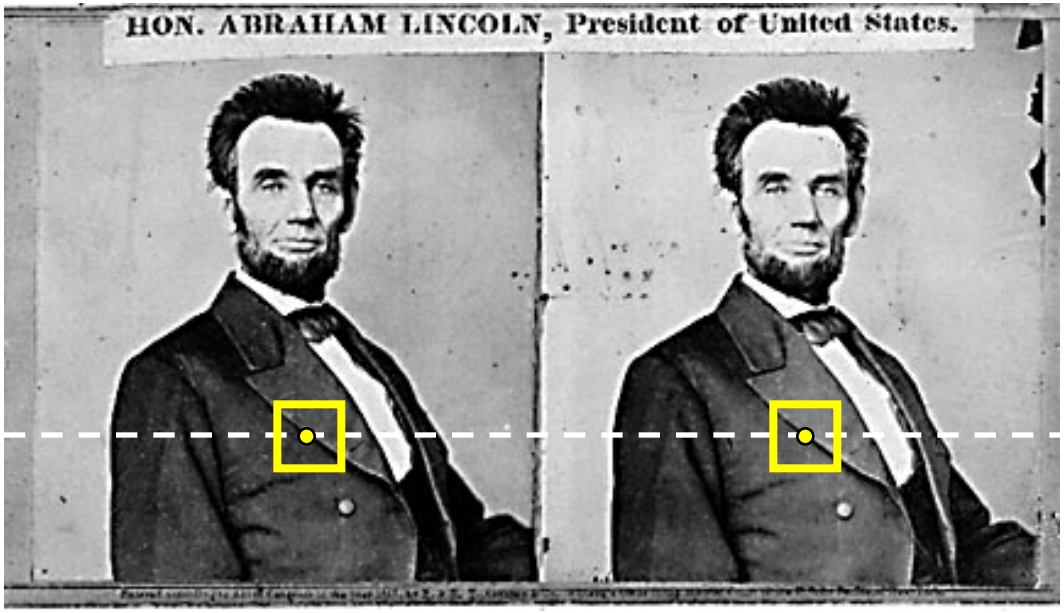
Estimating the disparity is equivalent to estimating depth.

Depth from disparity



$$disparity = x - x' = \frac{baseline * f}{z}$$

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***

Popular matching scores

- SSD (Sum Squared Distance)

$$\sum_{x,y} |W_1(x,y) - W_2(x,y)|^2$$

- NCC (Normalized Cross Correlation)

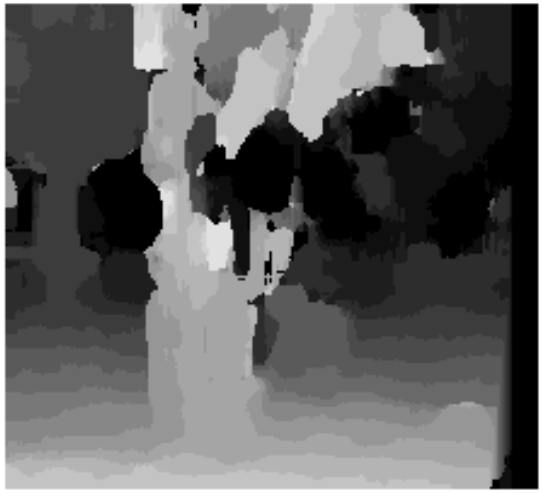
$$\frac{\sum_{x,y}(W_1(x,y) - \overline{W_1})(W_2(x,y) - \overline{W_2})}{\sigma_{W_1}\sigma_{W_2}}$$

where $\overline{W_i} = \frac{1}{n}\sum_{x,y} W_i$ $\quad$ $\sigma_{W_i} = \sqrt{\frac{1}{n}\sum_{x,y}(W_i - \overline{W_i})^2}$

- What advantages might NCC have over SSD?

Window size

W = 3



W = 20

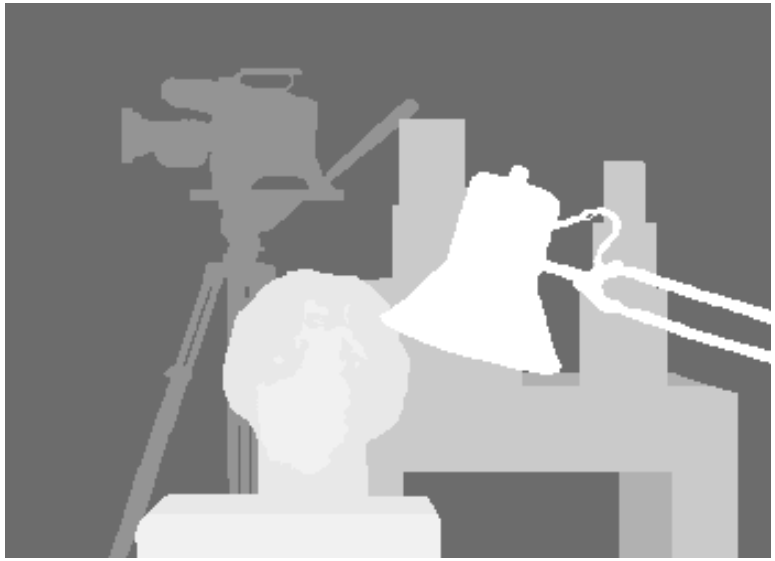
Effect of window size

- Smaller window
  - \+
  - –
- Larger window
  - \+
  - –

Stereo results

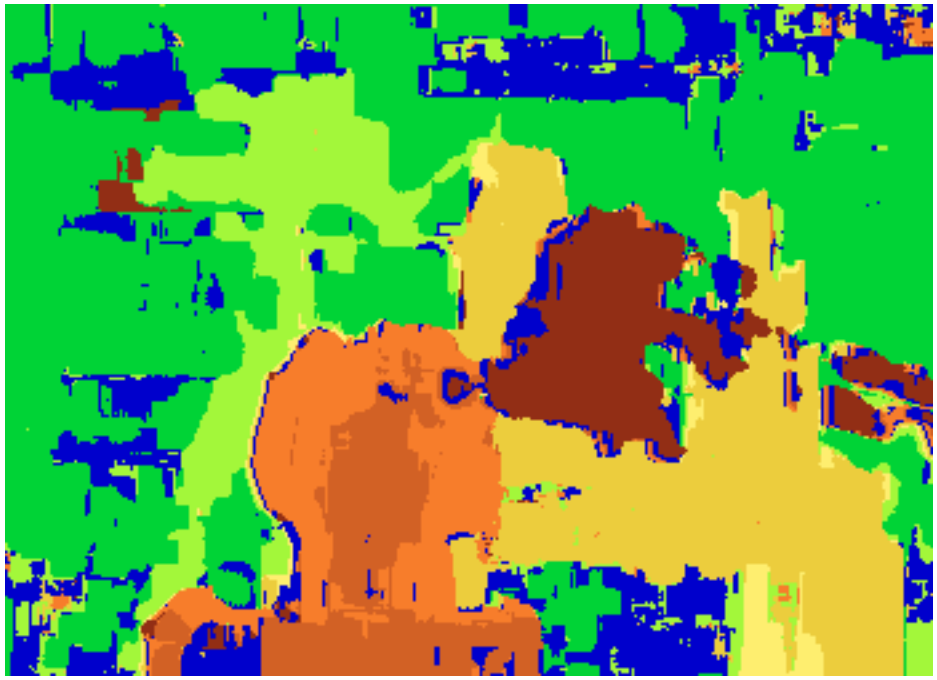
- Data from University of Tsukuba
- Similar results on other images without ground truth

Scene



Ground truth

Results with window search



Window-based matching
(best window size)

Ground truth

Better methods exist...

State of the art method

Ground truth

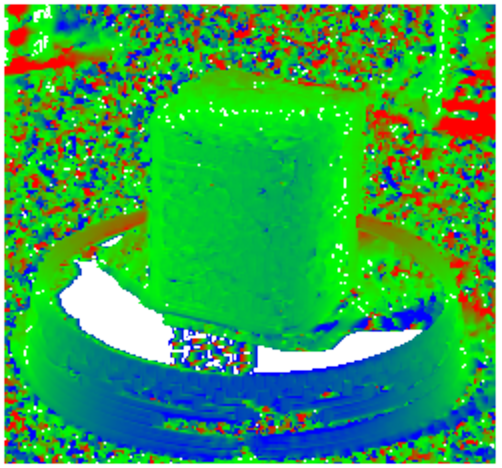
Boykov et al., Fast Approximate Energy Minimization via Graph Cuts, International Conference on Computer Vision, September 1999.

For the latest and greatest: http://www.middlebury.edu/stereo/

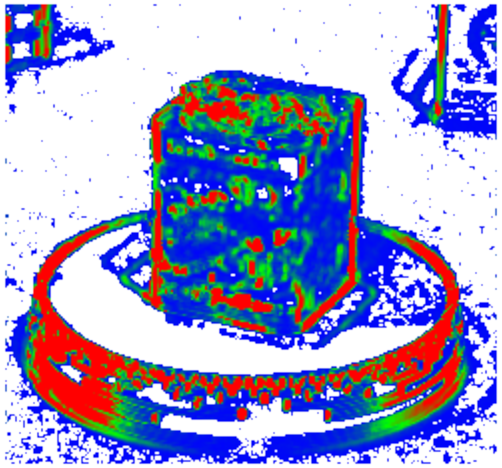
Stereo: certainty modeling

Compute certainty map from correlations

input



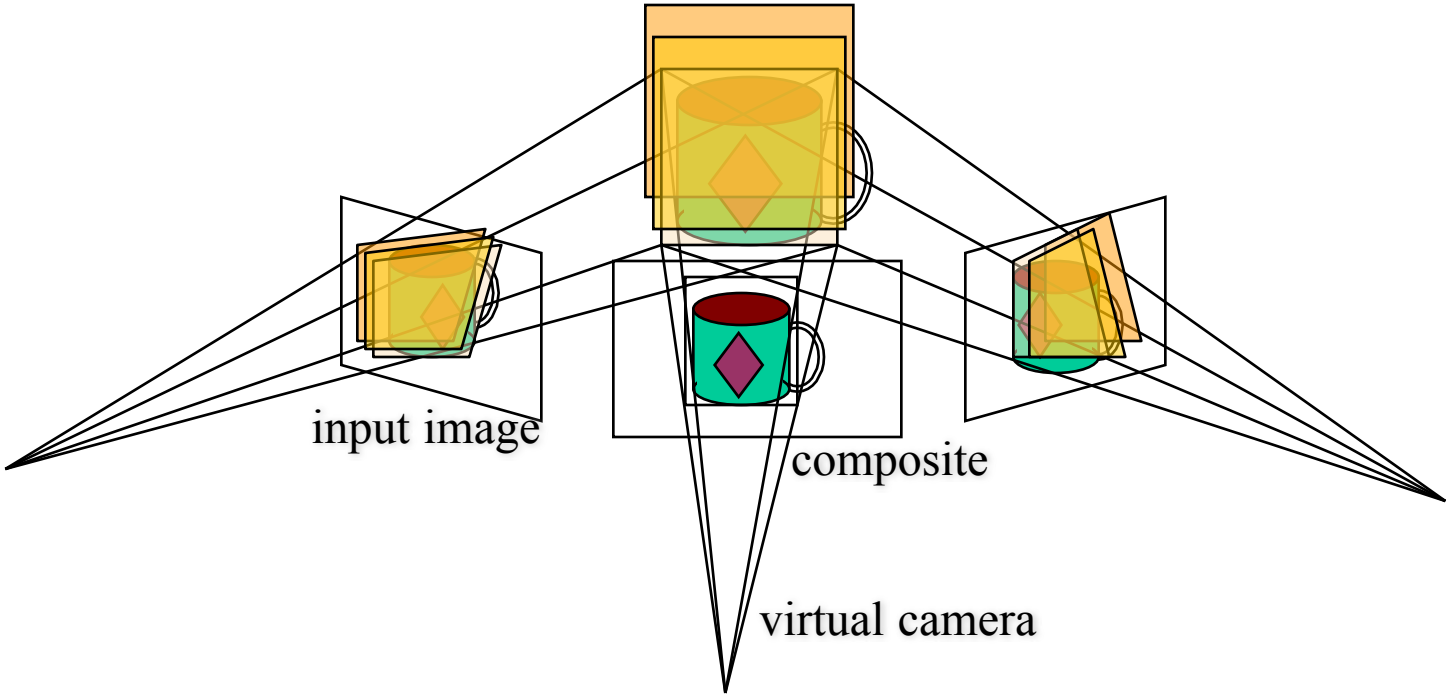
depth map



certainty map

Plane Sweep Stereo

Sweep family of planes through volume

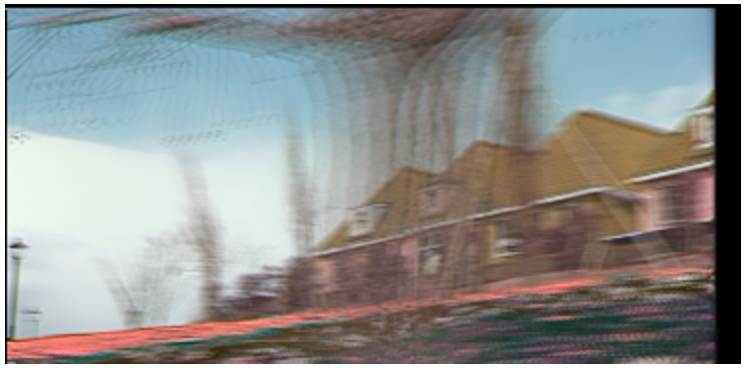


- each plane defines an image

Plane Sweep Stereo

For each depth plane

- compute composite (mosaic) image — *mean*

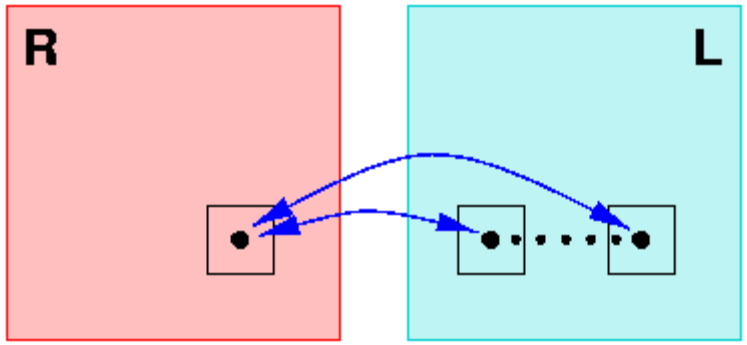


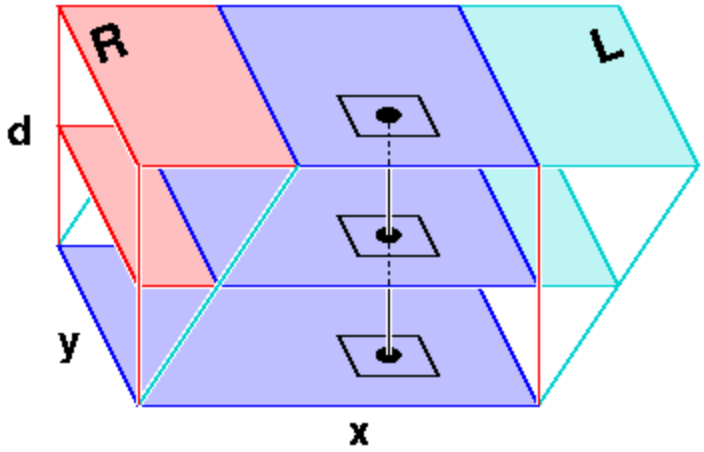
- compute error image — *variance*
- convert to confidence and aggregate spatially

Select winning depth at each pixel

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

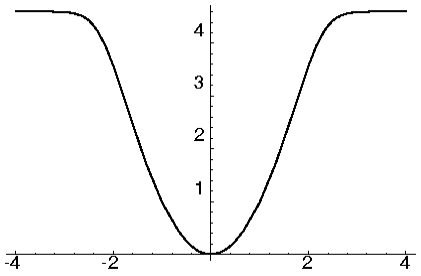
Stereo matching framework

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

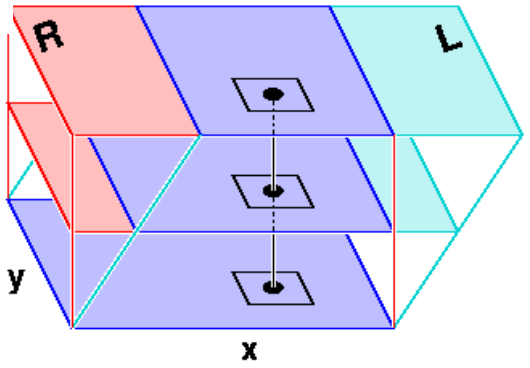


Can also use alternative match criteria

Stereo matching framework

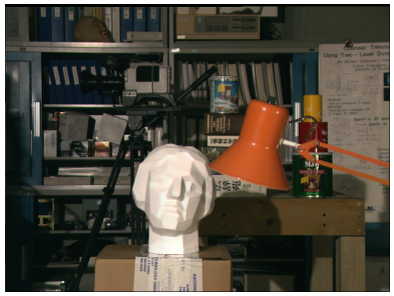
2. Aggregate costs spatially

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



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

Stereo matching



$I(x, y)$

$J(x, y)$

$E(x, y, d)$ the *disparity space image* (DSI)

Stereo matching framework

3. Choose winning disparity at each pixel

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

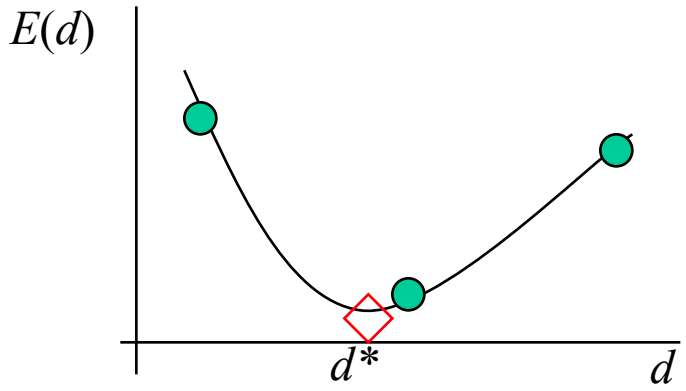
E(*x*, *y*, *d*) the *disparity space image* (DSI)

Stereo matching framework

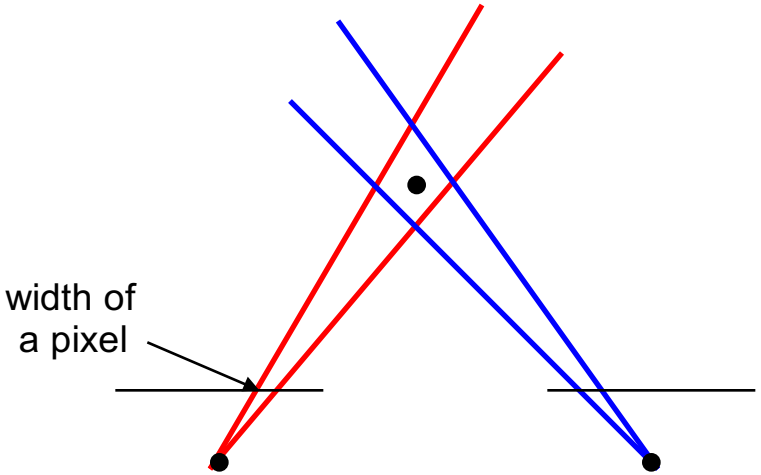
3. Choose winning disparity at each pixel

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

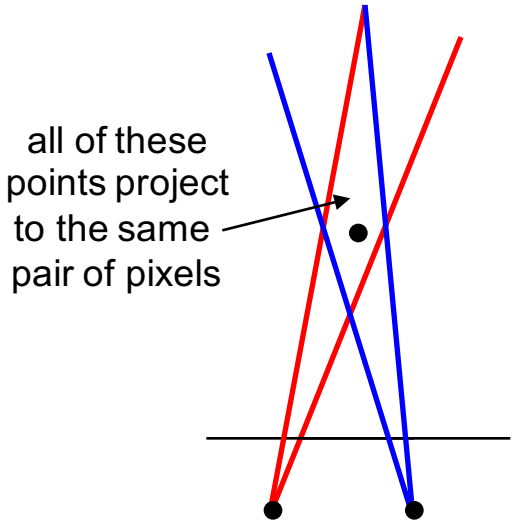
4. Interpolate to *sub-pixel* accuracy



Choosing the stereo baseline



Large Baseline



Small Baseline

What's the optimal baseline?

- Too small: large depth error
- Too large: difficult search problem

Traditional Stereo Matching

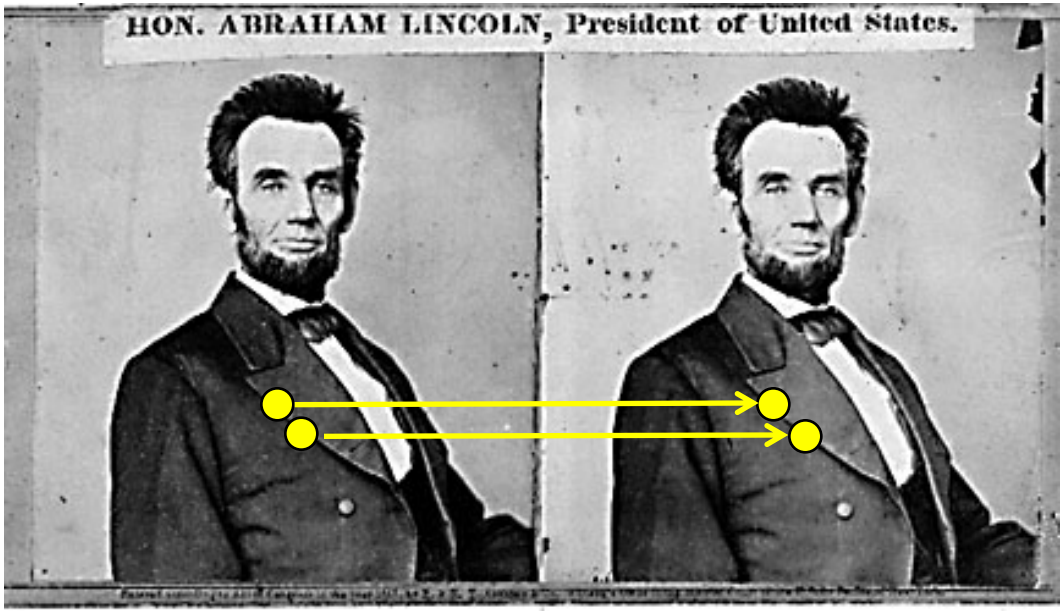
Advantages:

- 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

Stereo as energy minimization



What defines a good stereo correspondence?

1. Match quality
   - Want each pixel to find a good match in the other image
2. Smoothness
   - If two pixels are adjacent, they should (usually) move about the same amount

Stereo as energy minimization

- Find disparity map *d* that minimizes an energy function $E(d)$

- Simple pixel or window matching

$$E(d) = \sum_{(x,y)\in I} C(x, y, d(x, y))$$

$C(x, y, d(x, y))$ = Match distance between windows $I(x, y)$ and $J(x + d(x,y), y)$

Stereo as energy minimization

Better objective function

$$E(d) = E_d(d) + \lambda E_s(d)$$

match cost

Want each pixel to find a good match in the other image

smoothness cost

Adjacent pixels should (usually) move about the same amount