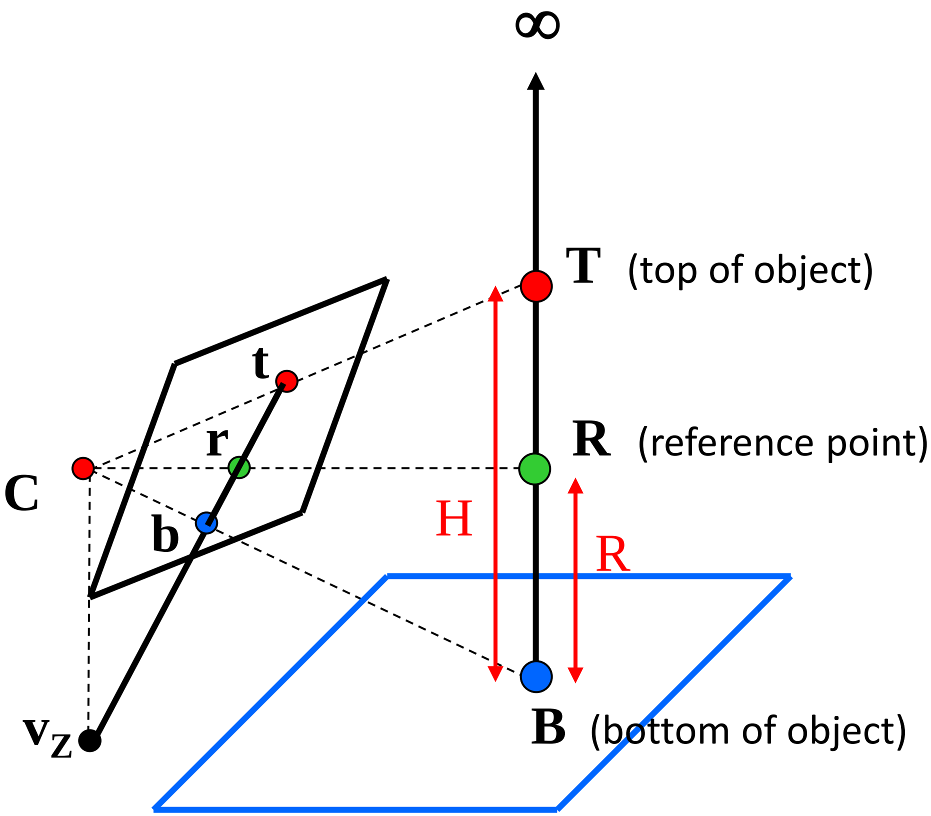
Measuring height



$$\frac{\|\mathbf{T}-\mathbf{B}\|\,\|\infty-\mathbf{R}\|}{\|\mathbf{R}-\mathbf{B}\|\,\|\infty-\mathbf{T}\|} = \frac{H}{R}$$

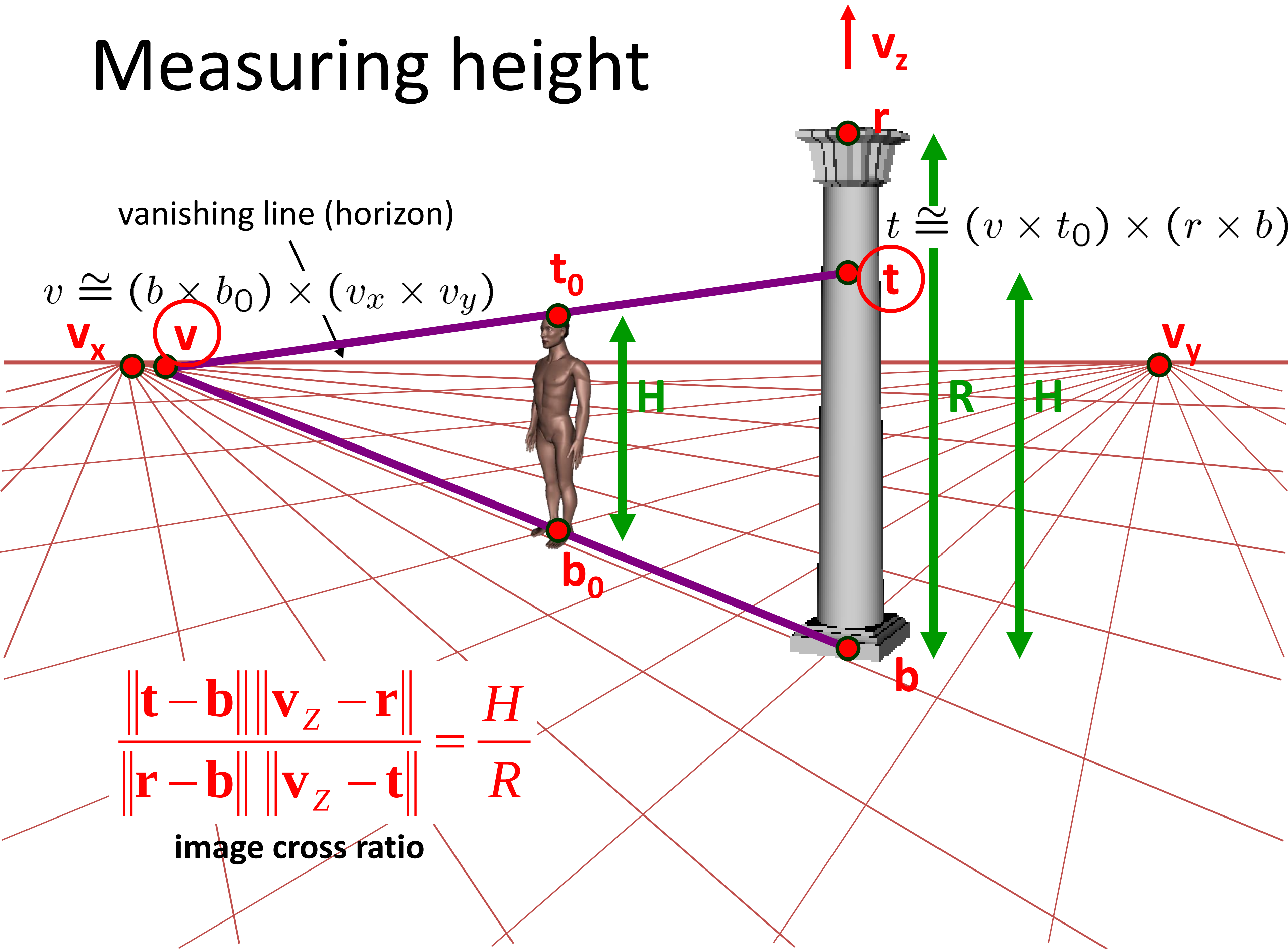
scene cross ratio

$$\frac{\|\mathbf{t}-\mathbf{b}\|\|\mathbf{v}_Z-\mathbf{r}\|}{\|\mathbf{r}-\mathbf{b}\|\,\|\mathbf{v}_Z-\mathbf{t}\|} = \frac{H}{R}$$

image cross ratio

scene points represented as $\mathbf{P} = \begin{bmatrix} X \\ Y \\ Z \\ 1 \end{bmatrix}$ image points as $\mathbf{p} = \begin{bmatrix} x \\ y \\ 1 \end{bmatrix}$

Measuring height

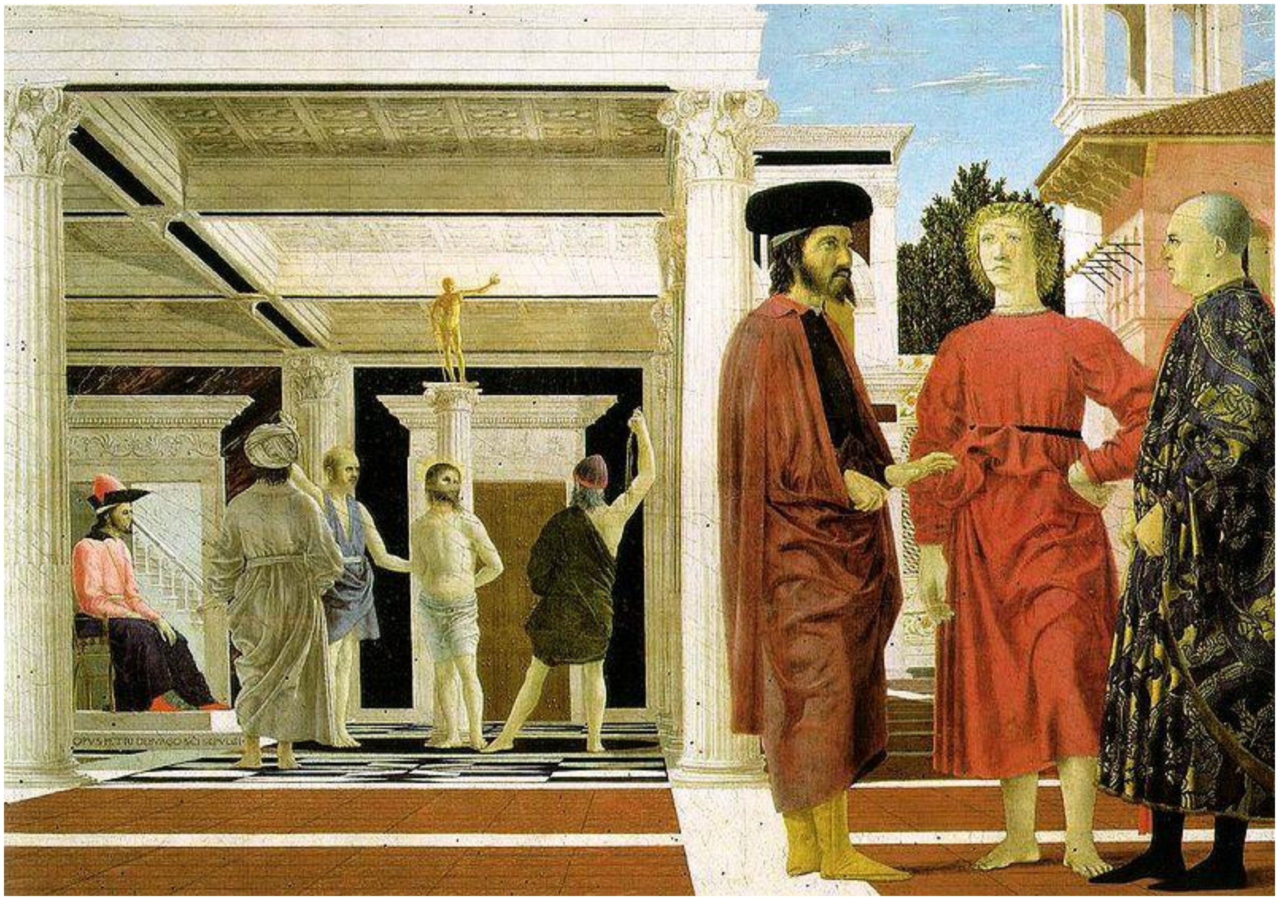


3D Modeling from a photograph

St. Jerome in his Study, H. Steenwick

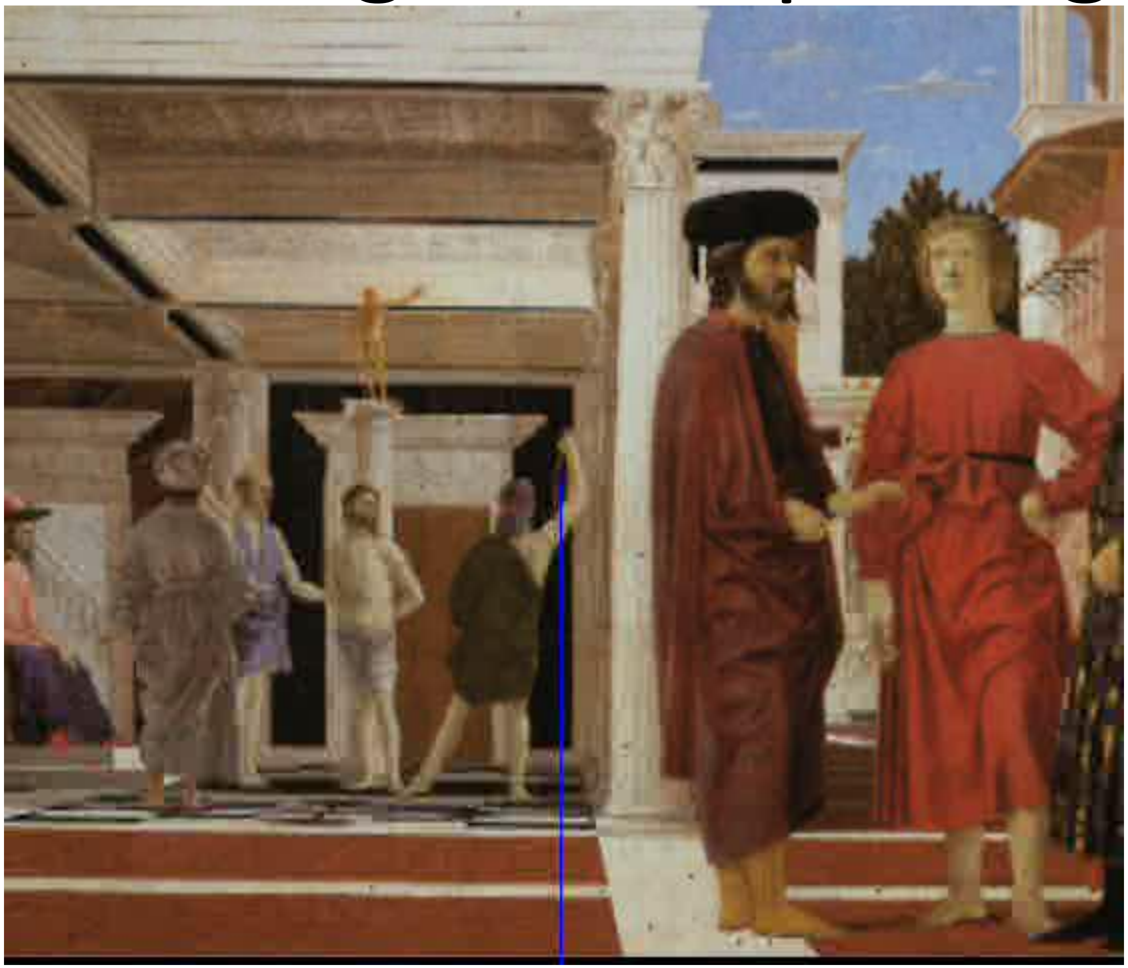
3D Modeling from a photograph

3D Modeling from a photograph



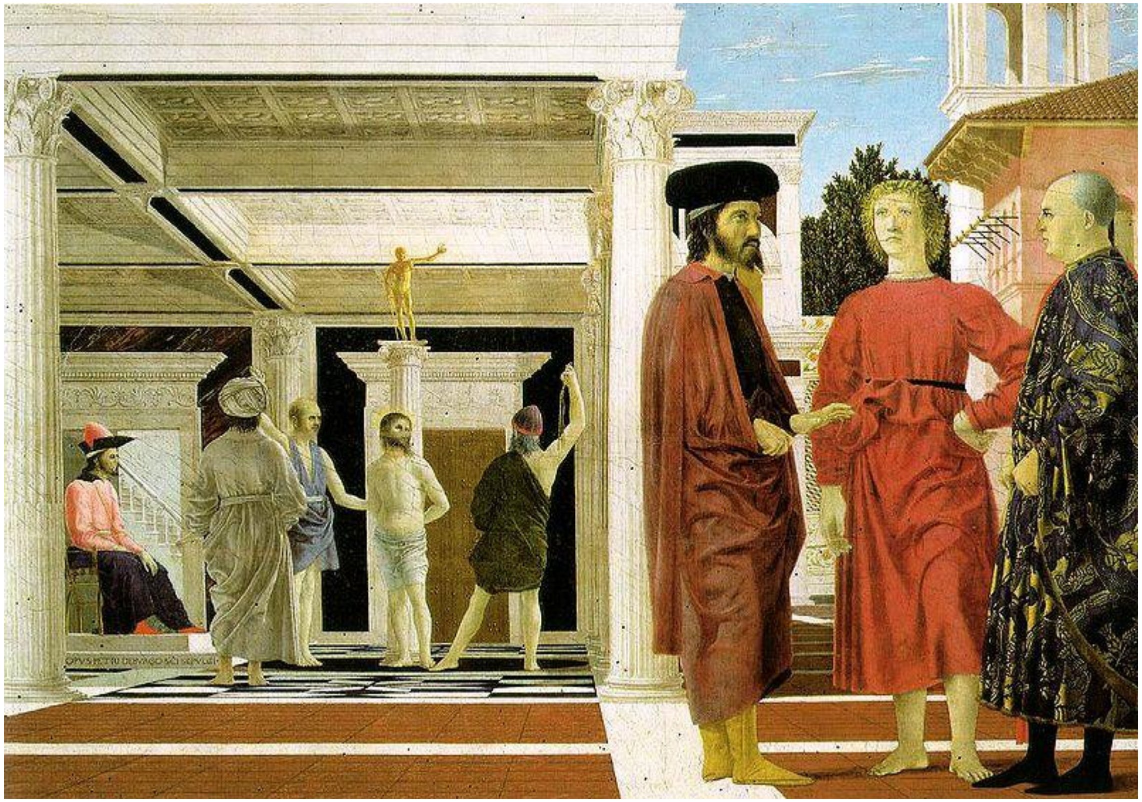
Flagellation, Piero della Francesca

3D Modeling from a photograph



video by Antonio Criminisi

3D Modeling from a photograph



Questions?

Camera calibration

- Goal: estimate the camera parameters
  - Version 1: solve for projection matrix

$$\mathbf{x} = \begin{bmatrix} wx \\ wy \\ w \end{bmatrix} = \begin{bmatrix} * & * & * & * \\ * & * & * & * \\ * & * & * & * \end{bmatrix} \begin{bmatrix} X \\ Y \\ Z \\ 1 \end{bmatrix} = \mathbf{\Pi X}$$

- Version 2: solve for camera parameters separately
  - intrinsics (focal length, principle point, pixel size)
  - extrinsics (rotation angles, translation)
  - radial distortion

Vanishing points and projection matrix

$$\mathbf{\Pi} = \begin{array}{c} \begin{bmatrix} \boxed{\begin{matrix} * \\ * \\ * \end{matrix}} & \boxed{\begin{matrix} * \\ * \\ * \end{matrix}} & \boxed{\begin{matrix} * \\ * \\ * \end{matrix}} & \boxed{\begin{matrix} * \\ * \\ * \end{matrix}} \end{bmatrix} \\ \begin{matrix} \boldsymbol{\pi}_1 & \boldsymbol{\pi}_2 & \boldsymbol{\pi}_3 & \boldsymbol{\pi}_4 \end{matrix} \end{array} = \begin{bmatrix} \boldsymbol{\pi}_1 & \boldsymbol{\pi}_2 & \boldsymbol{\pi}_3 & \boldsymbol{\pi}_4 \end{bmatrix}$$

- $\boldsymbol{\pi}_1 = \mathbf{\Pi}\begin{bmatrix}1 & 0 & 0 & 0\end{bmatrix}^T$ = $\mathbf{v}_x$ (X vanishing point)
- similarly, $\boldsymbol{\pi}_2 = \mathbf{v}_Y, \ \boldsymbol{\pi}_3 = \mathbf{v}_Z$
- $\boldsymbol{\pi}_4 = \mathbf{\Pi}\begin{bmatrix}0 & 0 & 0 & 1\end{bmatrix}^T = \text{projection of world origin}$

$$\mathbf{\Pi} = \begin{bmatrix} \mathbf{v}_X & \mathbf{v}_Y & \mathbf{v}_Z & \mathbf{o} \end{bmatrix}$$

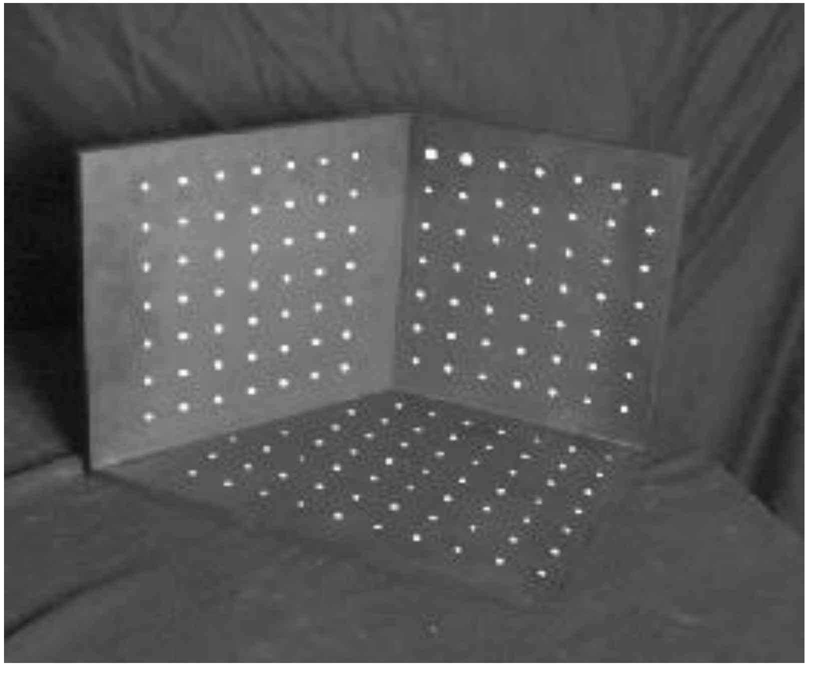
Not So Fast! We only know **v**’s up to a scale factor

$$\mathbf{\Pi} = \begin{bmatrix} a\,\mathbf{v}_X & b\mathbf{v}_Y & c\mathbf{v}_Z & \mathbf{o} \end{bmatrix}$$

- Can fully specify by providing 3 reference points

Calibration using a reference object

- Place a known object in the scene
  - identify correspondence between image and scene
  - compute mapping from scene to image

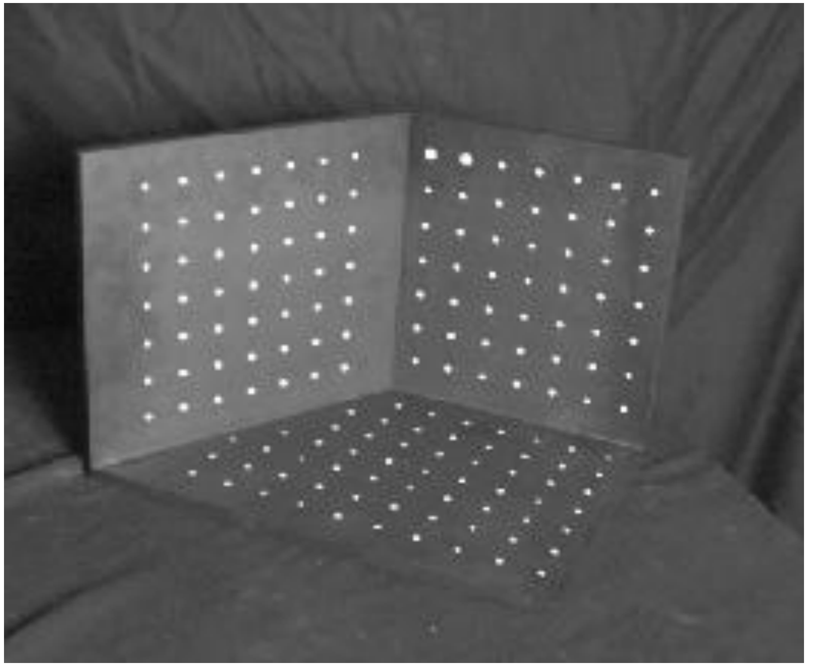


Issues

- must know geometry very accurately
- must know 3D->2D correspondence

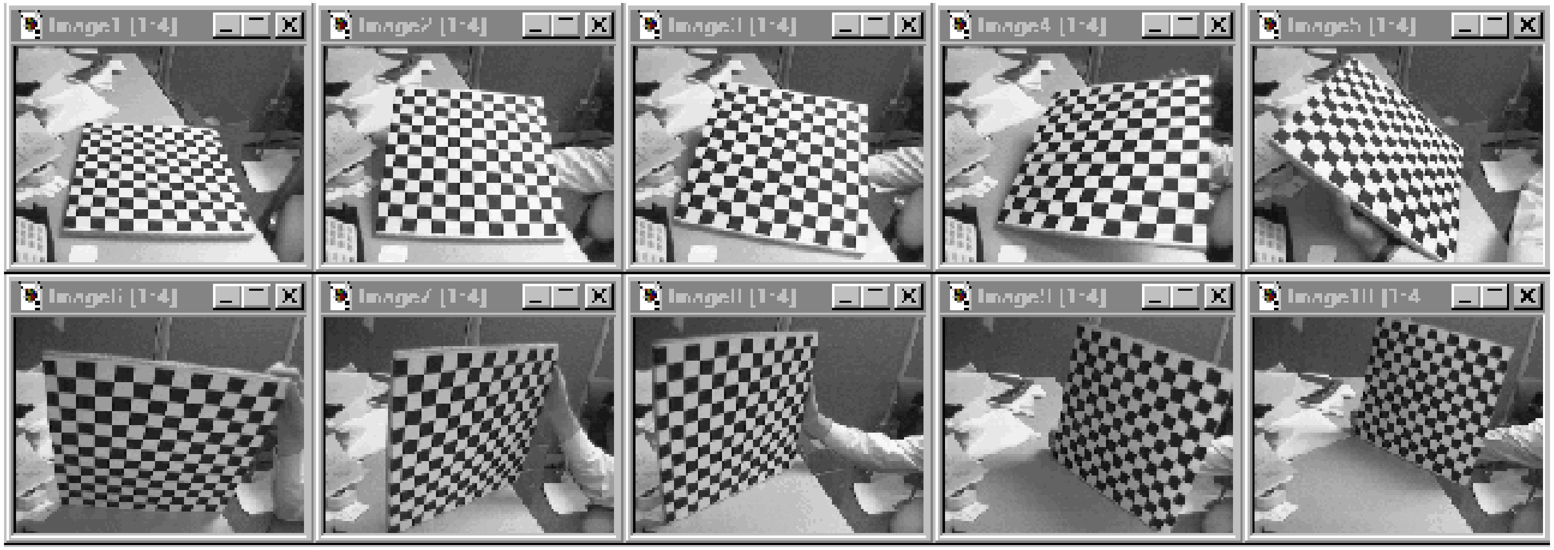
Estimating the projection matrix

- Place a known object in the scene
  - identify correspondence between image and scene
  - compute mapping from scene to image



$$\begin{bmatrix} u_i \\ v_i \\ 1 \end{bmatrix} \cong \begin{bmatrix} m_{00} & m_{01} & m_{02} & m_{03} \\ m_{10} & m_{11} & m_{12} & m_{13} \\ m_{20} & m_{21} & m_{22} & m_{23} \end{bmatrix} \begin{bmatrix} X_i \\ Y_i \\ Z_i \\ 1 \end{bmatrix}$$

Alternative: multi-plane calibration



Images courtesy Jean-Yves Bouguet, Intel Corp.

Advantage

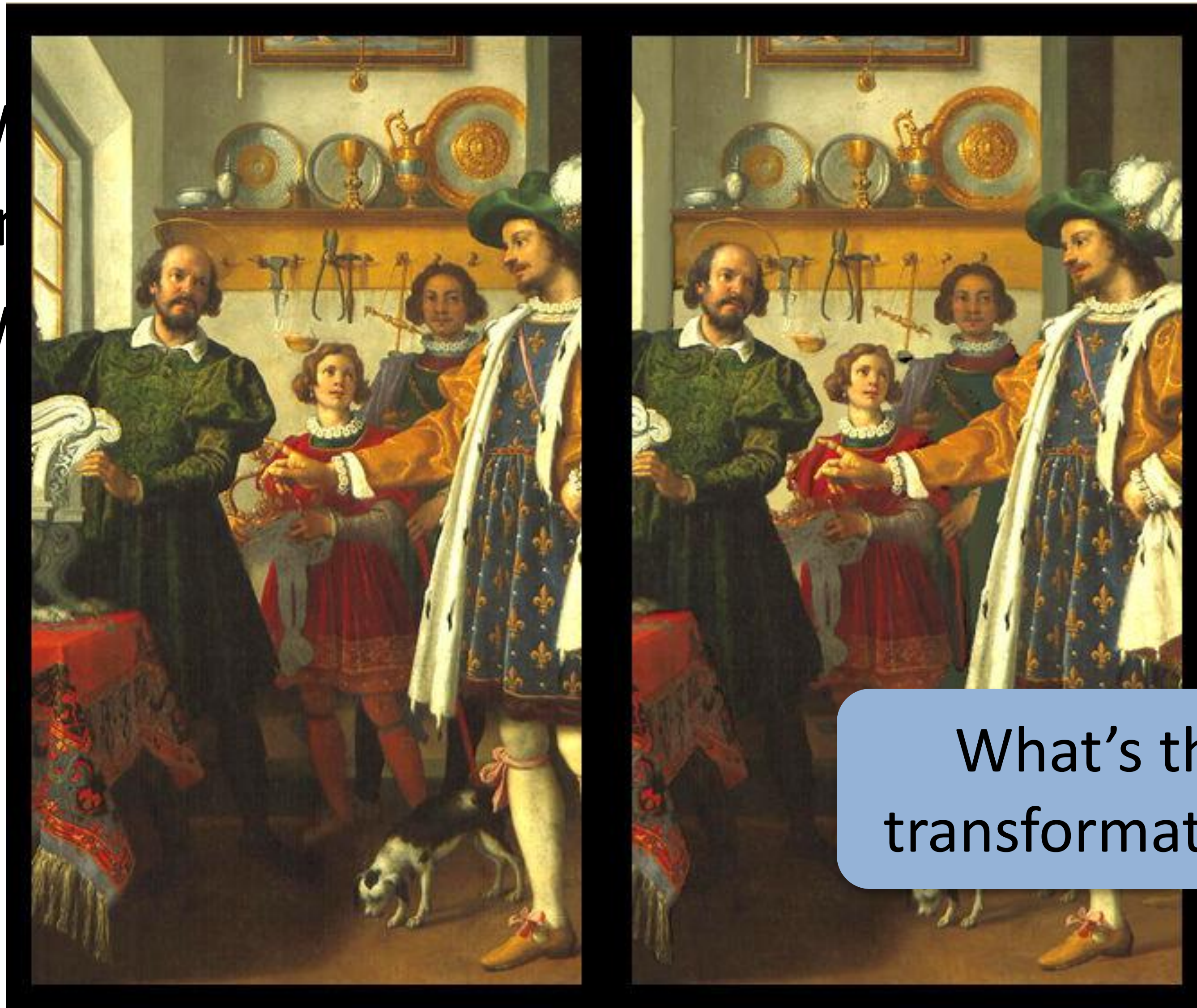
- Only requires a plane
- Don’t have to know positions/orientations
- Good code available online! (including in OpenCV)
  - Matlab version by Jean-Yves Bouget: http://www.vision.caltech.edu/bouguetj/calib_doc/index.html
  - Zhengyou Zhang’s web site: http://research.microsoft.com/~zhang/Calib/

Some Related Techniques

- Image-Based Modeling and Photo Editing
  - Mok et al., SIGGRAPH 2001
  - http://graphics.csail.mit.edu/ibedit/
- Single View Modeling of Free-Form Scenes
  - Zhang et al., CVPR 2001
  - http://grail.cs.washington.edu/projects/svm/
- Tour Into The Picture
  - Anjyo et al., SIGGRAPH 1997
  - http://koigakubo.hitachi.co.jp/little/DL_TipE.html

More than one view?

- W a
  si
- W

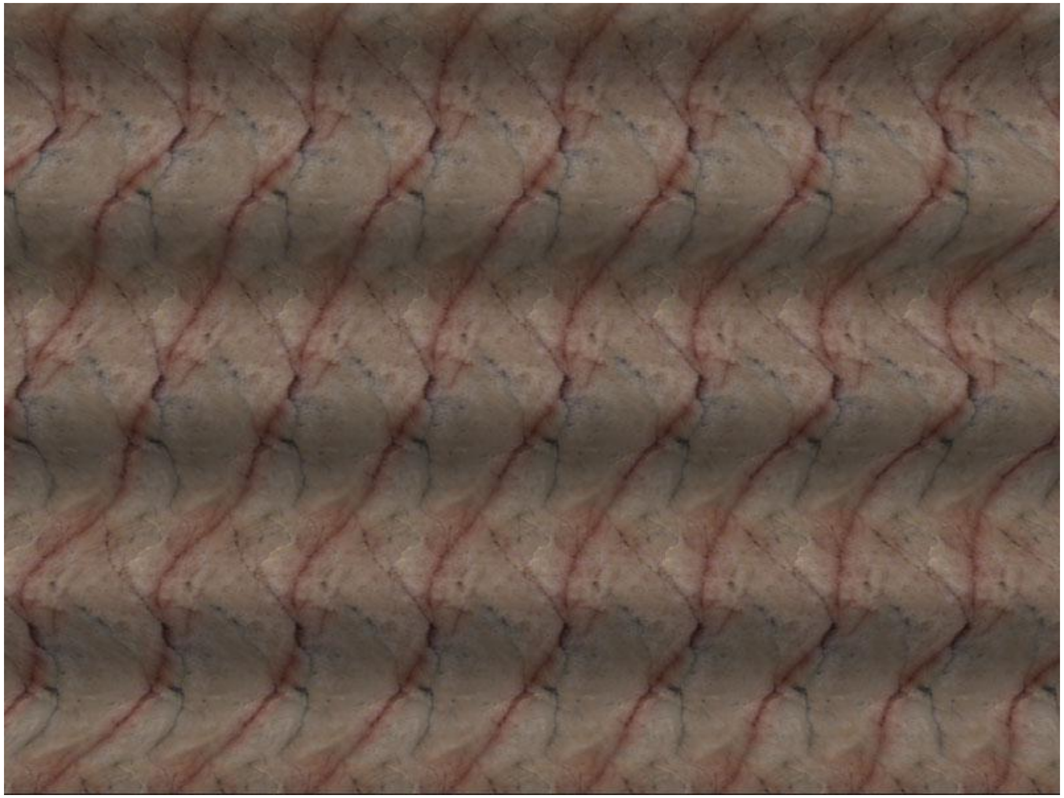


What's the transformation?

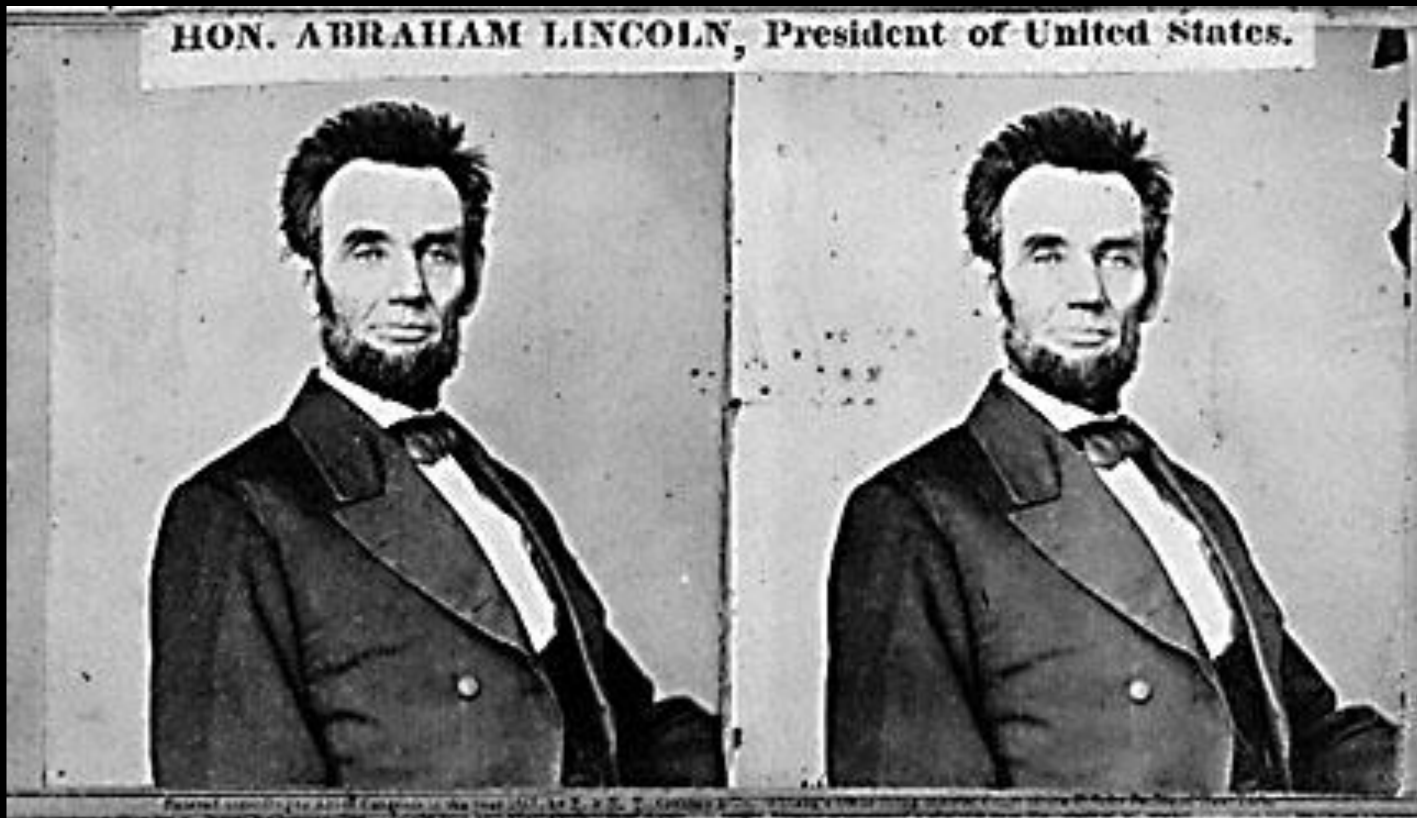
CS6670: Computer Vision

Noah Snavely

Lecture 10: Stereo

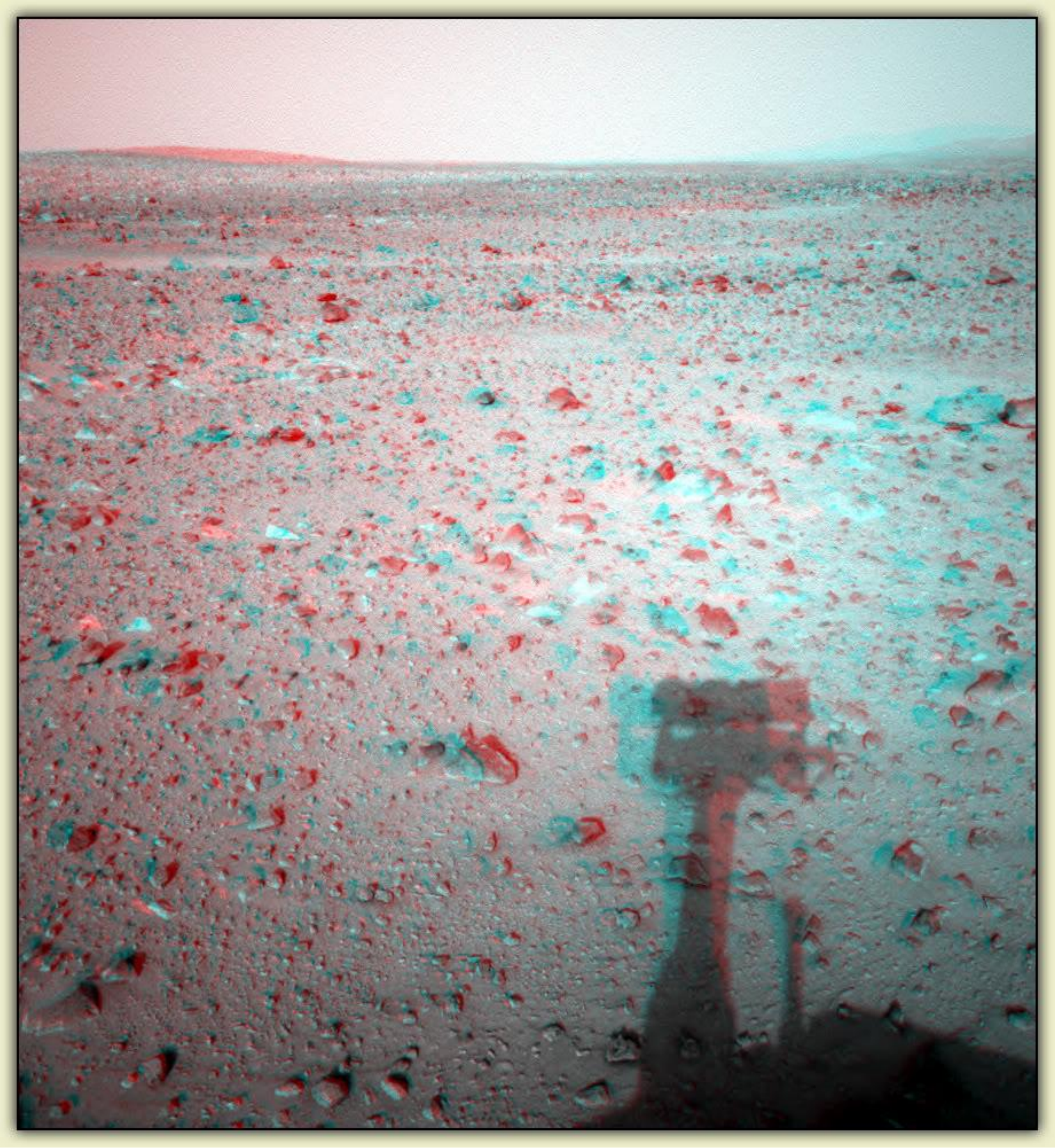


Single image stereogram, by [Niklas Een]


HON. ABRAHAM LINCOLN, President of United States.

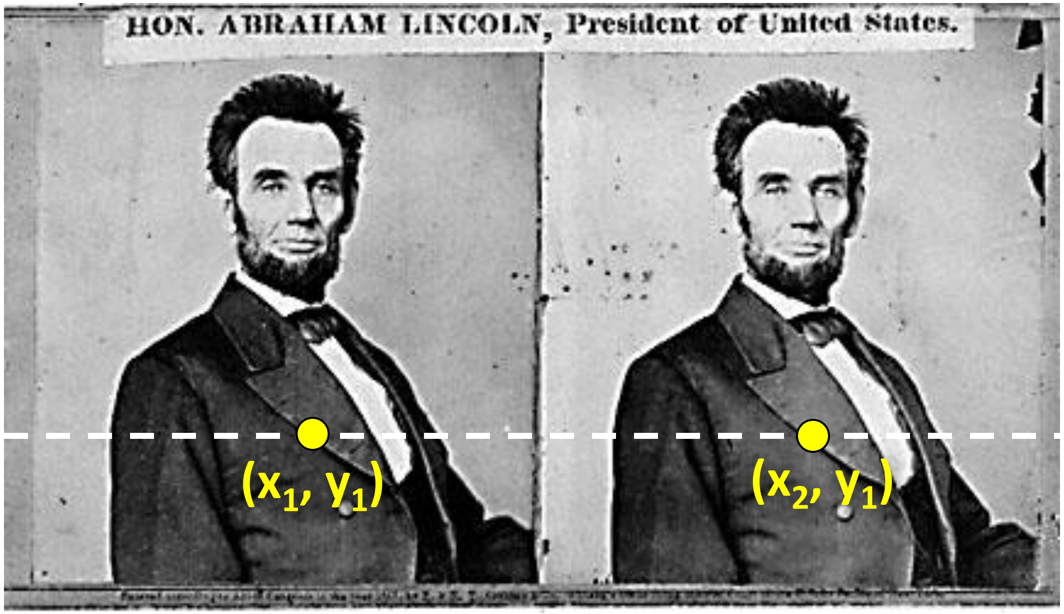
Public Library, Stereoscopic Looking Room, Chicago, by Phillips, 1923

Mark Twain at Pool Table", no date, UCR Museum of Photography



Epipolar geometry

epipolar lines



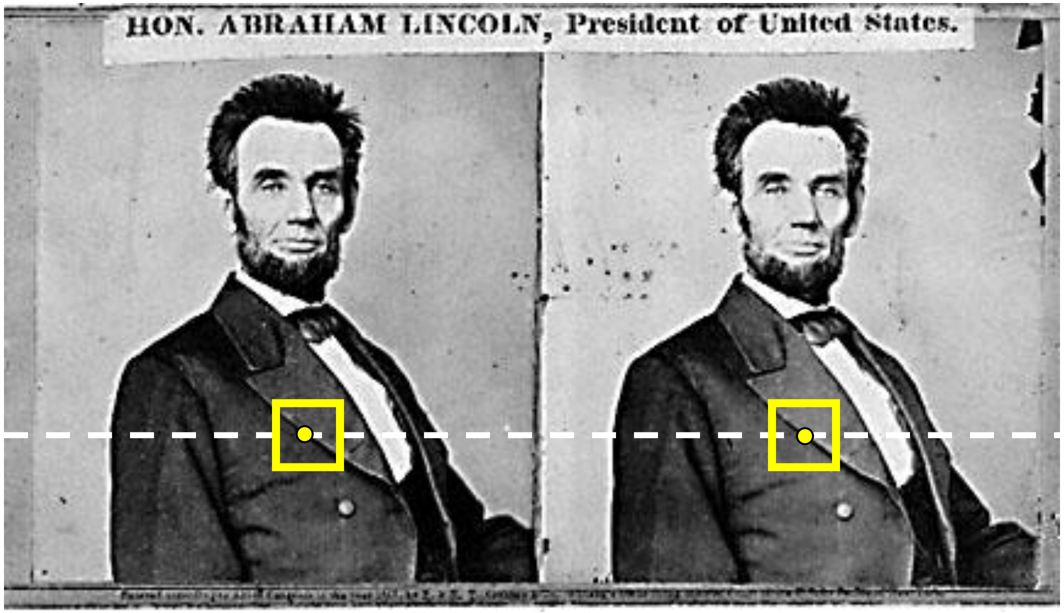
Two images captured by a purely horizontal translating camera
(*rectified* stereo pair)

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

Stereo matching algorithms

- Match Pixels in Conjugate Epipolar Lines
  - Assume brightness constancy
  - This is a tough problem
  - Numerous approaches
    - A good survey and evaluation: http://www.middlebury.edu/stereo/

Your basic stereo algorithm



For each epipolar line

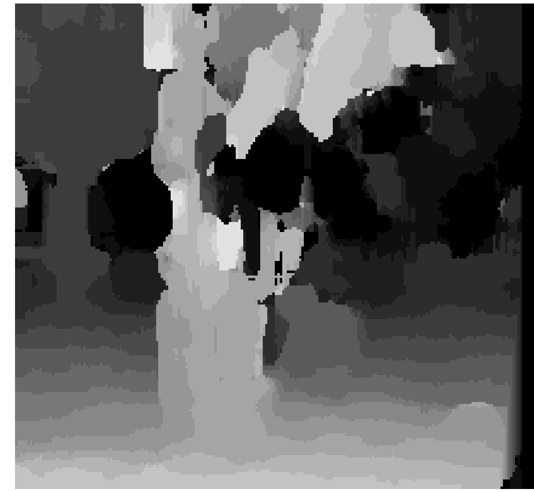
For each pixel in the left image

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

Improvement: match ***windows***

Window size

W = 3



W = 20

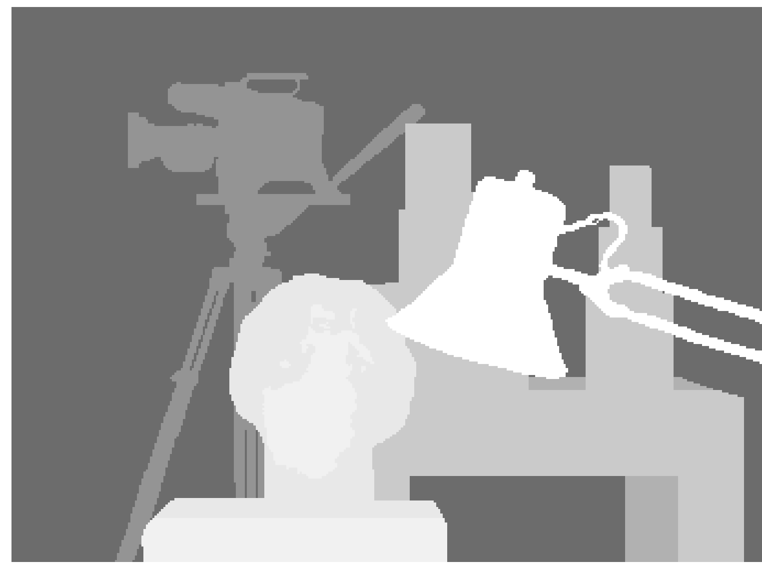
Better results with *adaptive window*

- T. Kanade and M. Okutomi, *A Stereo Matching Algorithm with an Adaptive Window: Theory and Experiment*,, Proc. International Conference on Robotics and Automation, 1991.
- D. Scharstein and R. Szeliski. Stereo matching with nonlinear diffusion. International Journal of Computer Vision, 28(2):155-174, July 1998

Stereo results

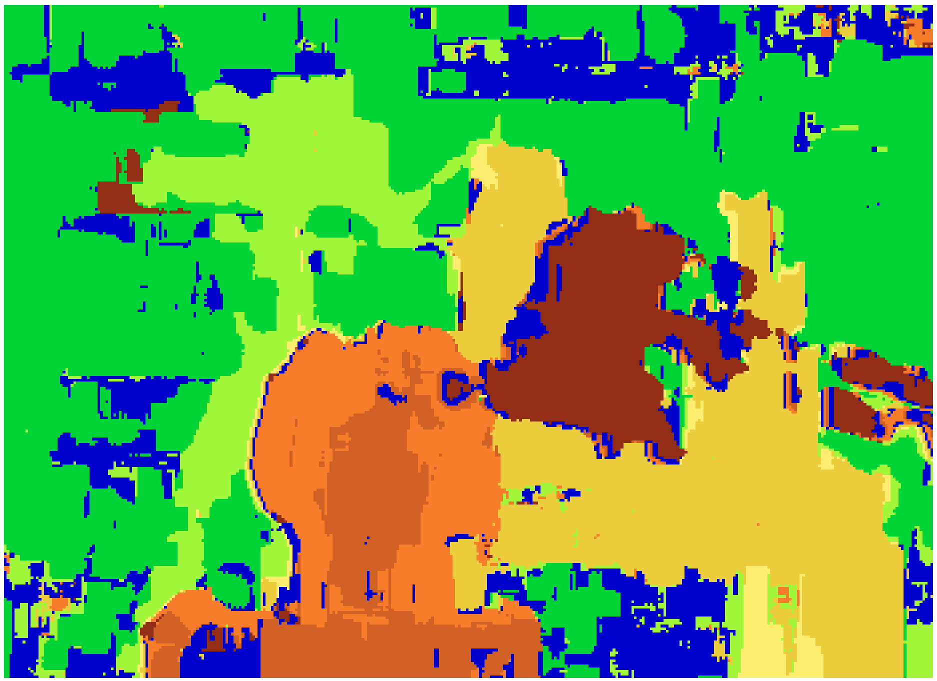
– Data from University of Tsukuba
– Similar results on other images without ground truth

Scene



Ground truth

Results with window search



Window-based matching
(best window size)

Ground truth

Better methods exist...

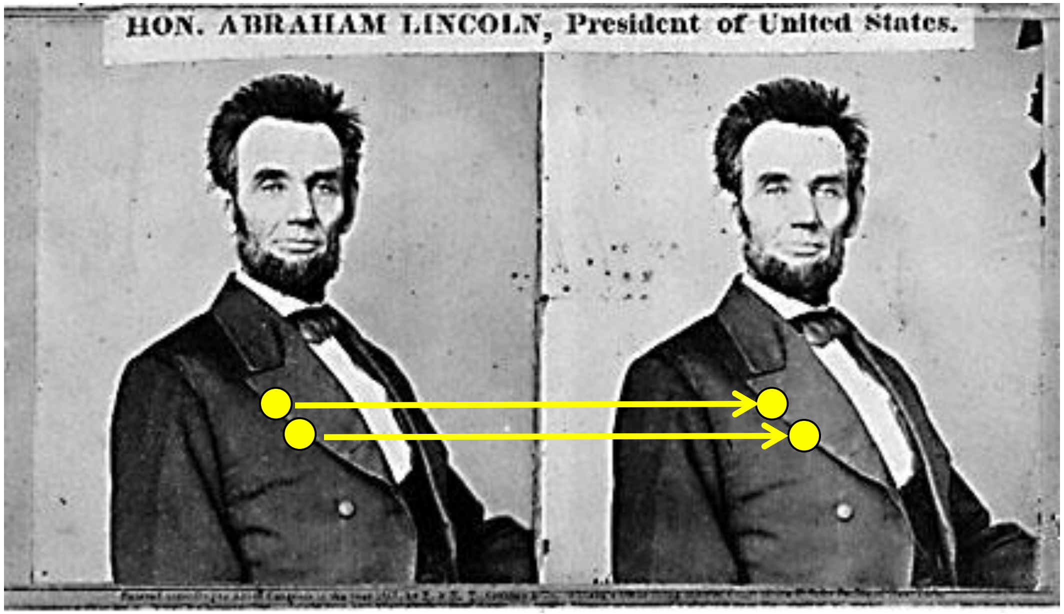
State of the art method

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

Ground truth

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

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

- Expressing this mathematically
  1. Match quality
     - Want each pixel to find a good match in the other image

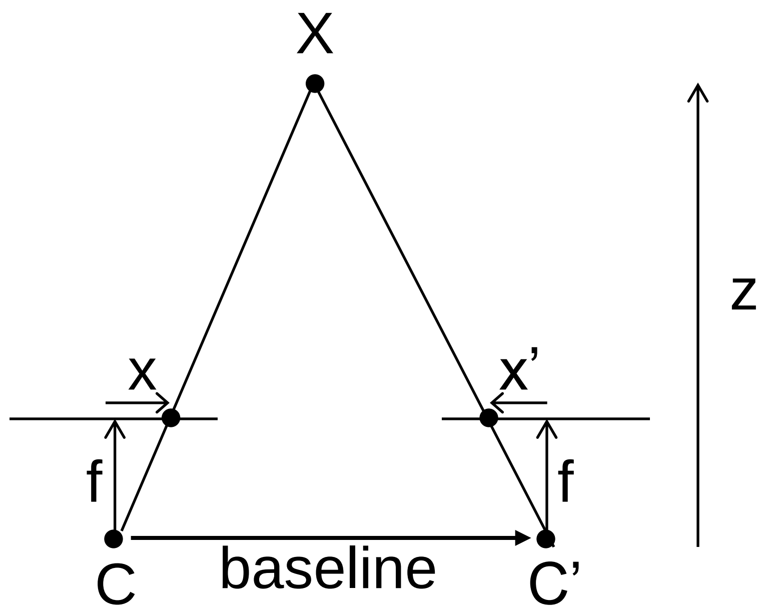
$$matchCost = \sum_{x,y} \|I(x,y) - J(x + d_{xy}, y)\|$$

  2. Smoothness
     - If two pixels are adjacent, they should (usually) move about the same amount

$$smoothnessCost = \sum_{neighbor\ \ pixels\ \ p,q} |d_p - d_q|$$

- We want to minimize $Energy = matchCost + smoothnessCost$
  – This is a special type of energy function known as an MRF (Markov Random Field)
    - Effective and fast algorithms have been recently developed:
      – Graph cuts, belief propagation….
      – for more details (and code): http://vision.middlebury.edu/MRF/
      – Great tutorials available online (including video of talks)

Depth from disparity



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

Real-time stereo

[Nomad robot](http://www.frc.ri.cmu.edu/projects/meteorobot/index.html) searches for meteorites in Antartica
http://www.frc.ri.cmu.edu/projects/meteorobot/index.html

- Used for robot navigation (and other tasks)
  - Several software-based real-time stereo techniques have been developed (most based on simple discrete search)

Stereo reconstruction pipeline

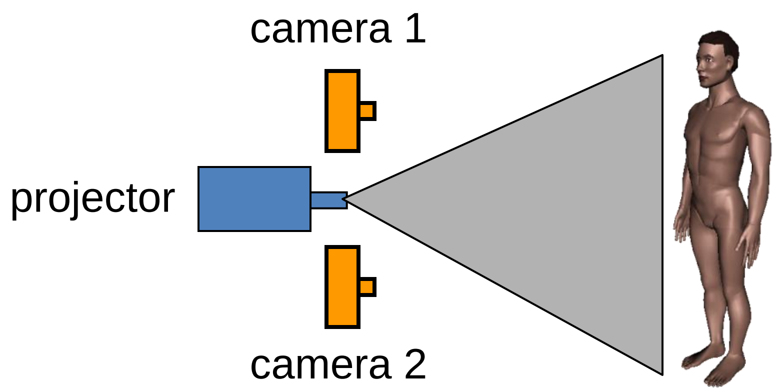
- Steps
  - Calibrate cameras
  - Rectify images
  - Compute disparity
  - Estimate depth

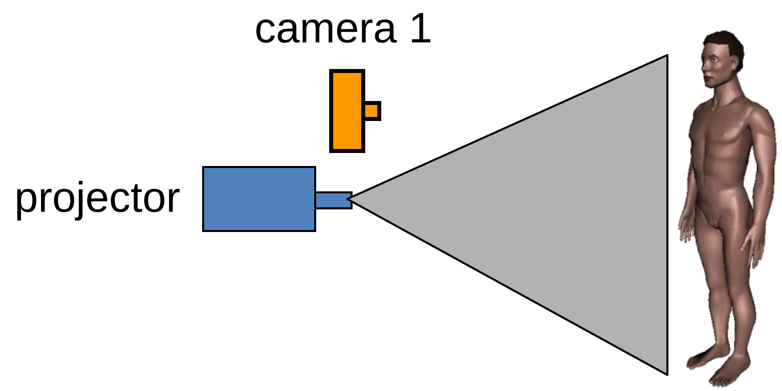
What will cause errors?

- Camera calibration errors
- Poor image resolution
- Occlusions
- Violations of brightness constancy (specular reflections)
- Large motions
- Low-contrast image regions

Active stereo with structured light

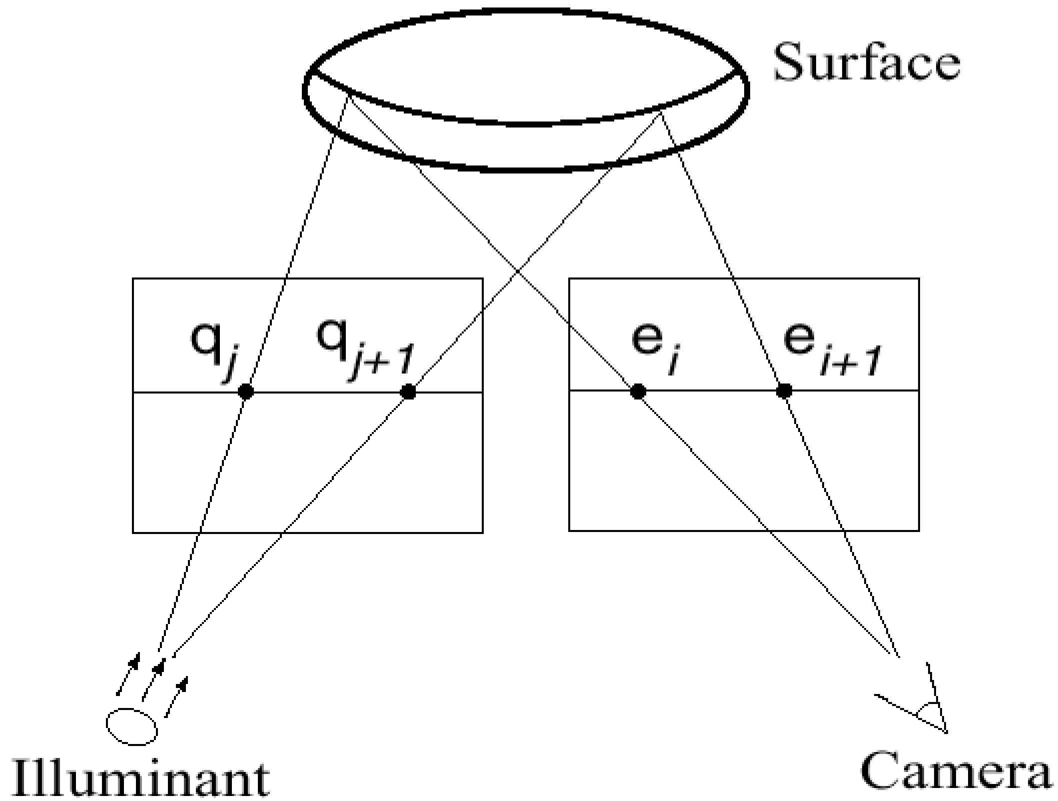
Li Zhang's one-shot stereo



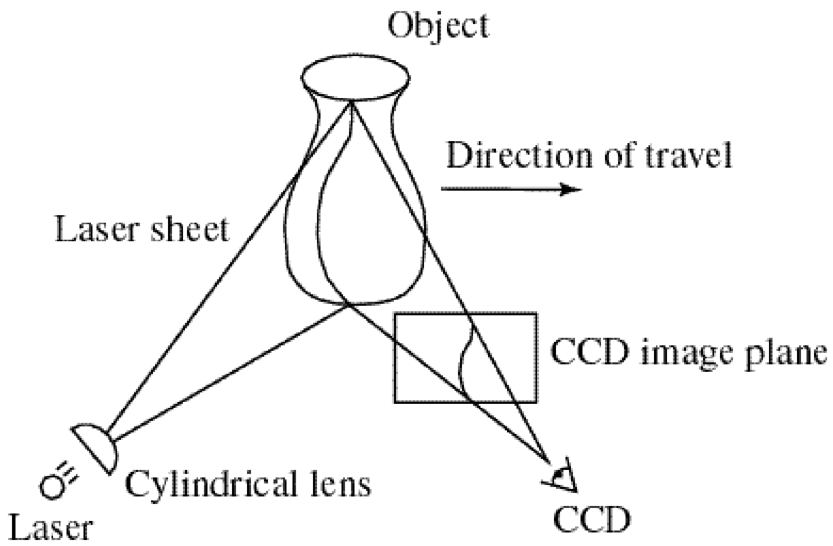


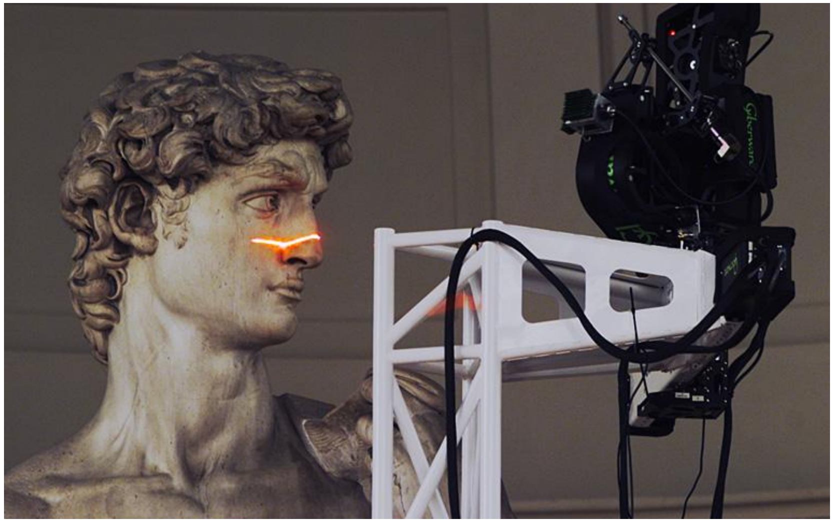
- Project "structured" light patterns onto the object
  - simplifies the correspondence problem

Active stereo with structured light



Laser scanning





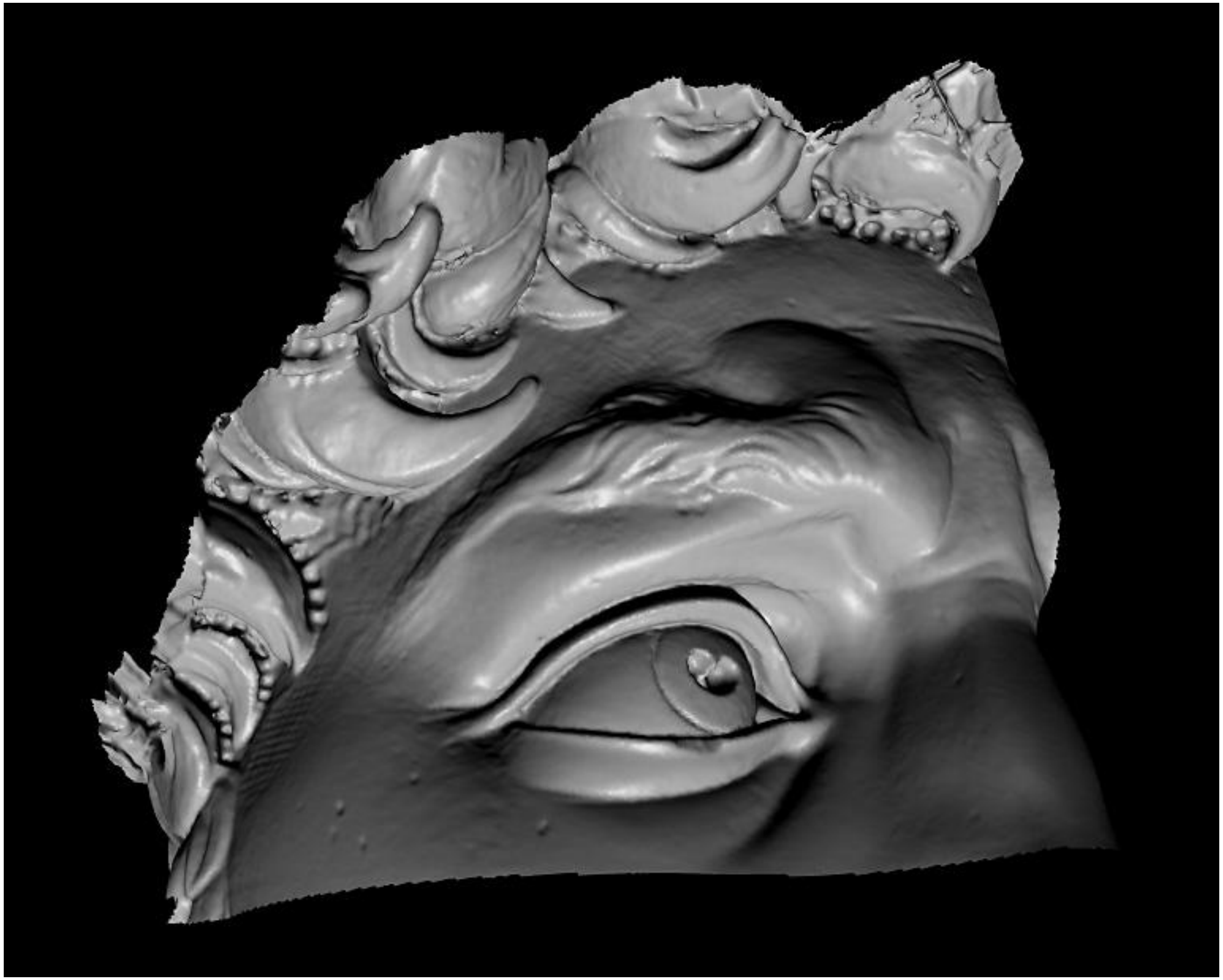
Digital Michelangelo Project
http://graphics.stanford.edu/projects/mich/

- Optical triangulation
  - Project a single stripe of laser light
  - Scan it across the surface of the object
  - This is a very precise version of structured light scanning

Laser scanned models

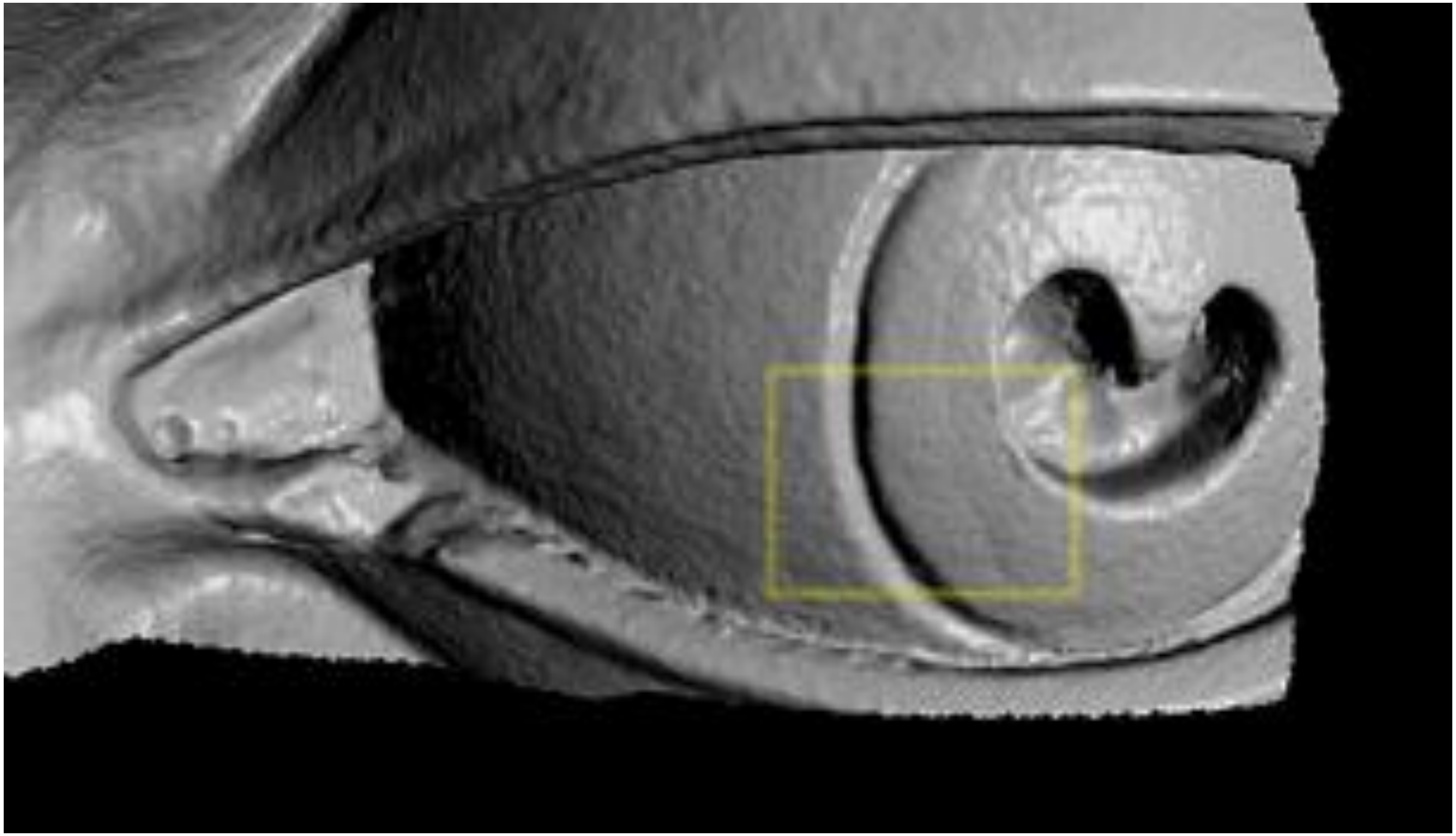
The Digital Michelangelo Project, Levoy et al.

Laser scanned models



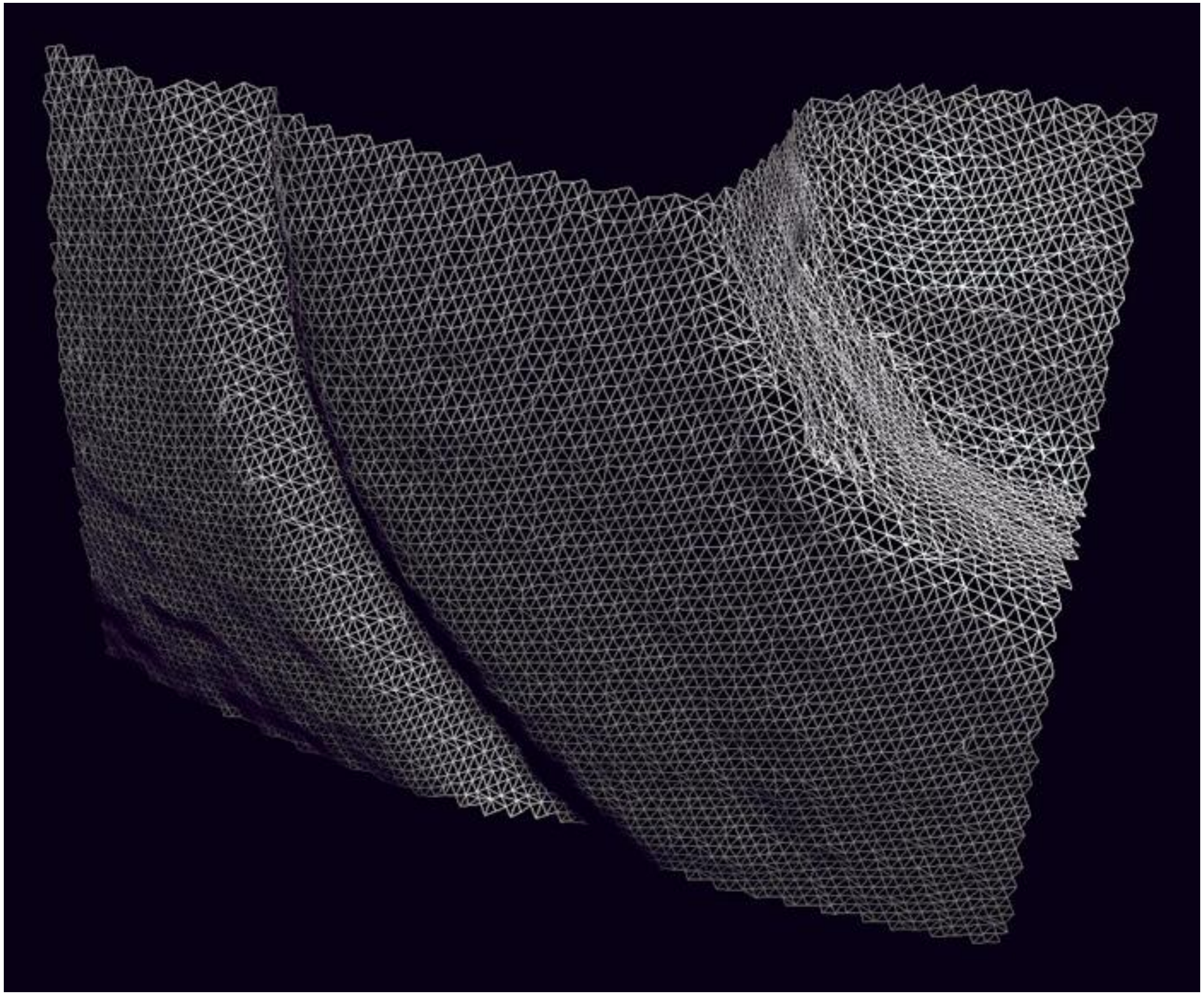
The Digital Michelangelo Project, Levoy et al.

Laser scanned models



The Digital Michelangelo Project, Levoy et al.

Laser scanned models



The Digital Michelangelo Project, Levoy et al.

Stereo as energy minimization

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

- Simple pixel / window matching

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

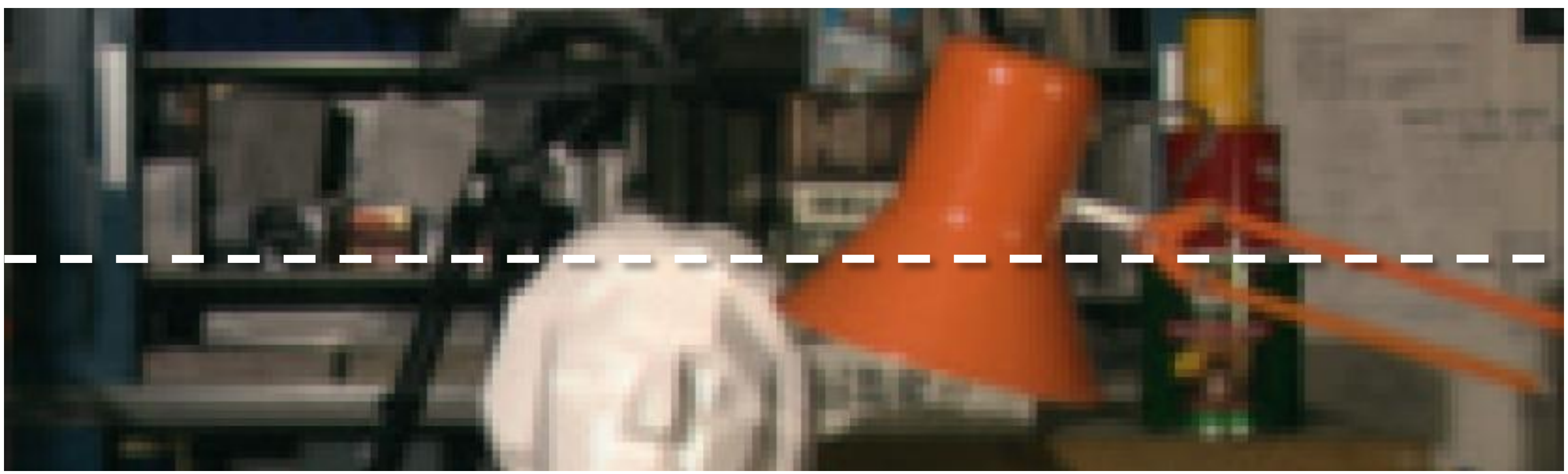
$C(x, y, d(x, y))$ = SSD distance between windows $I(x, y)$ and $J(x + d(x,y), y)$

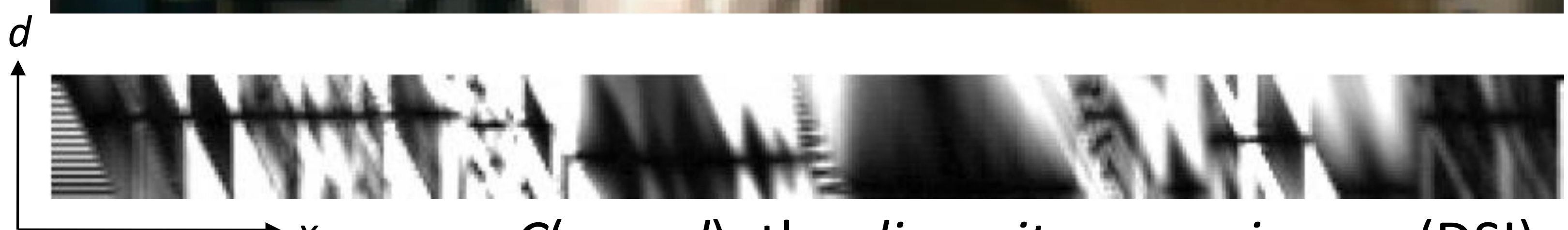
Stereo as energy minimization

$I(x, y)$

$J(x, y)$

y = 141





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

Stereo as energy minimization

Simple pixel / window matching: choose the minimum of each column in the DSI independently:

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

Stereo as energy minimization

- Better objective function

$$E(d) = \underbrace{E_d(d)}_{\text{match cost}} + \lambda \underbrace{E_s(d)}_{\text{smoothness cost}}$$

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

Adjacent pixels should (usually) move about the same amount

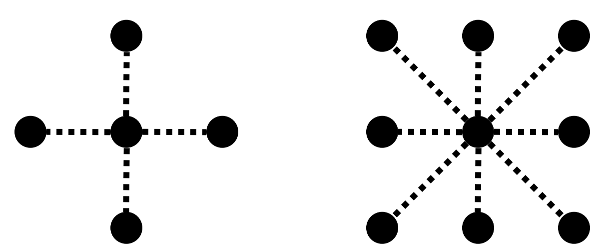
Stereo as energy minimization

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

match cost: $$E_d(d) = \sum_{(x,y)\in I} C(x, y, d(x, y))$$

smoothness cost: $$E_s(d) = \sum_{(p,q)\in \mathcal{E}} V(d_p, d_q)$$

$\mathcal{E}$: set of neighboring pixels



4-connected neighborhood

8-connected neighborhood

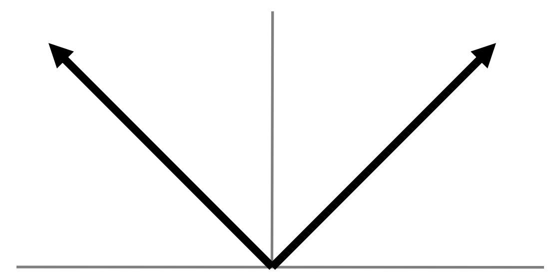
Smoothness cost

$$E_s(d) = \sum_{(p,q)\in\mathcal{E}} V(d_p, d_q)$$

How do we choose *V*?

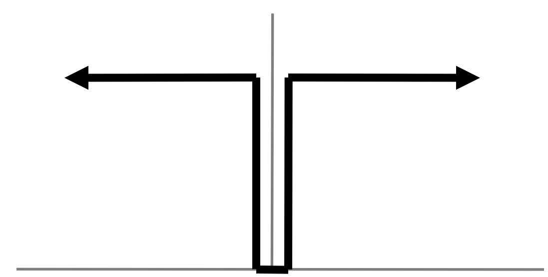
$$V(d_p, d_q) = |d_p - d_q|$$

L_1 distance



$$V(d_p, d_q) = \begin{cases} 0 & \text{if } d_p = d_q \\ 1 & \text{if } d_p \neq d_q \end{cases}$$

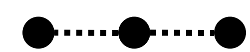
“Potts model”



Dynamic programming

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

- Can minimize this independently per scanline using dynamic programming (DP)



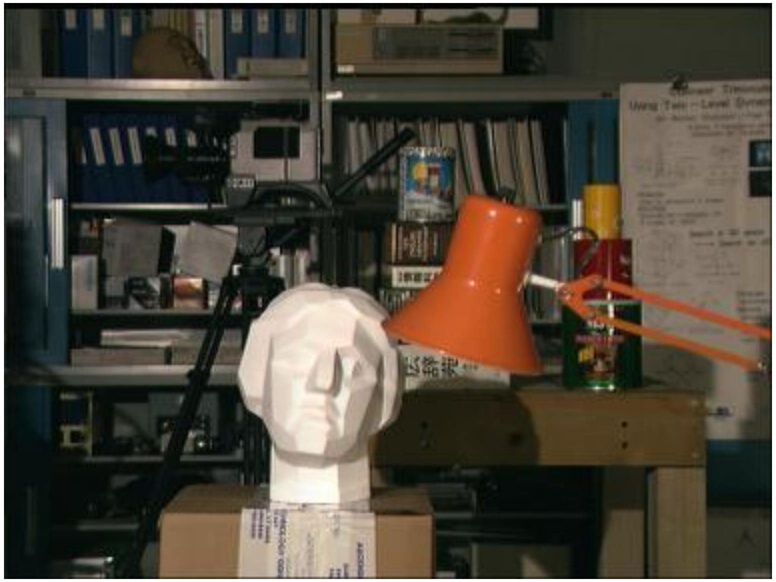
$D(x, y, d)$: minimum cost of solution such that *d*(*x*,*y*) = *d*

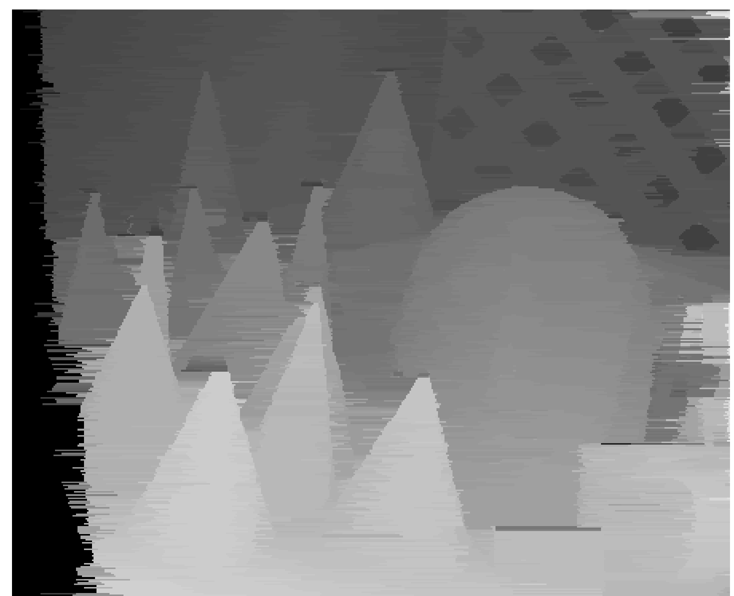
$$D(x,y,d) = C(x,y,d) + \min_{d'} \{D(x-1,y,d') + \lambda \left|d - d'\right|\}$$

Dynamic programming

- Finds “smooth” path through DPI from left to right

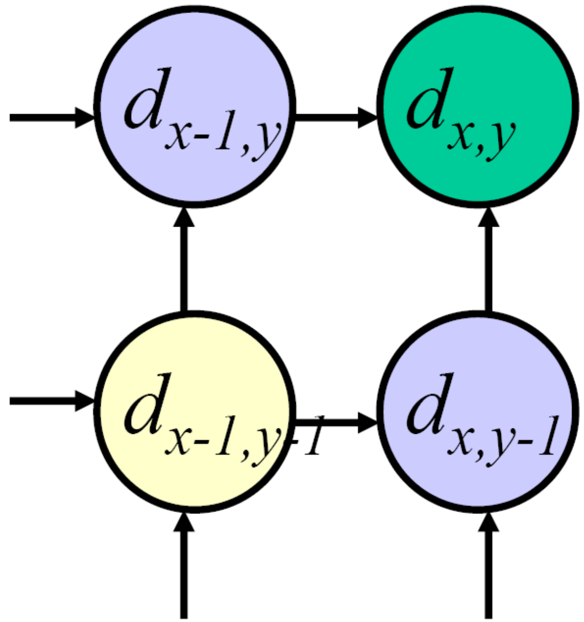
Dynamic Programming





Dynamic programming

- Can we apply this trick in 2D as well?



- No: $d_{x,y-1}$ and $d_{x-1,y}$ may depend on different values of $d_{x-1,y-1}$

Slide credit: D. Huttenlocher

Stereo as a minimization problem

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

- The 2D problem has many local minima
  - Gradient descent doesn't work well

- And a large search space
  - n x m image w/ k disparities has k^{nm} possible solutions
  - Finding the global minimum is NP-hard in general

- Good approximations exist… we'll see this soon

Questions?