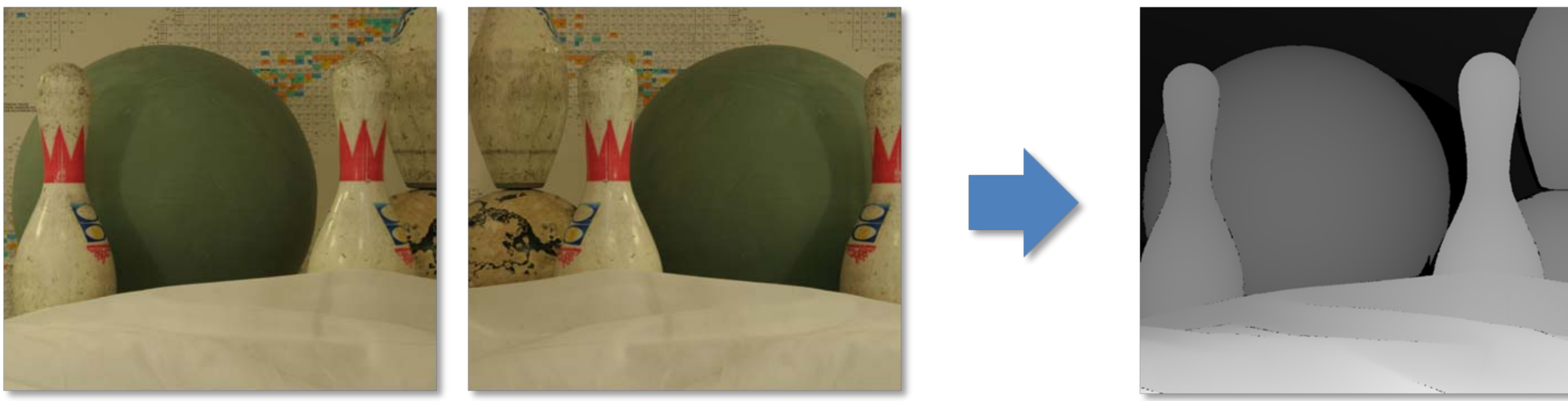
CS6670: Computer Vision

Noah Snavely

Lecture 10: Stereo and Graph Cuts



Announcements

- Project 2 out, due Wednesday, October 14
  - Artifact due Friday, October 16

- Questions?

Readings

- Szeliski, Chapter 11.2 – 11.5

What we've learned so far

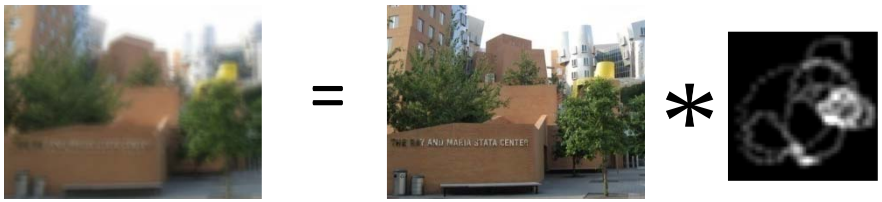


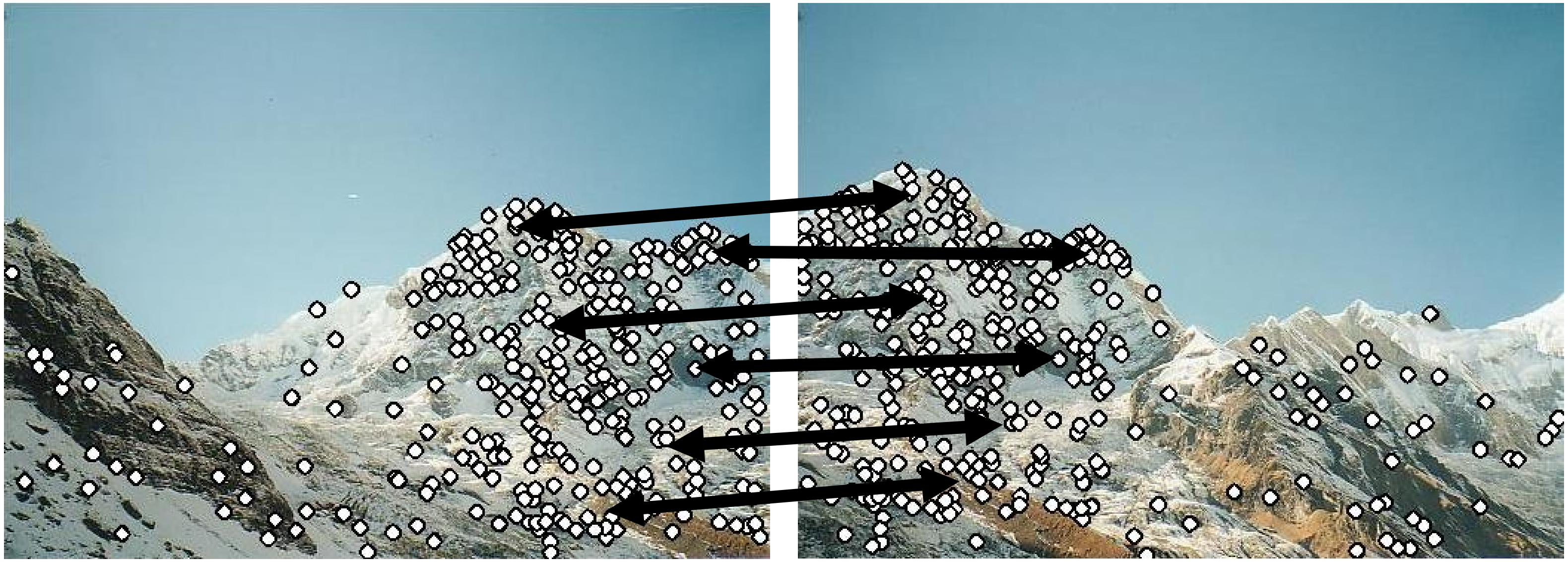
Image filtering

Cameras

Edge detection

Image transformations

Feature detection and matching



Panoramas

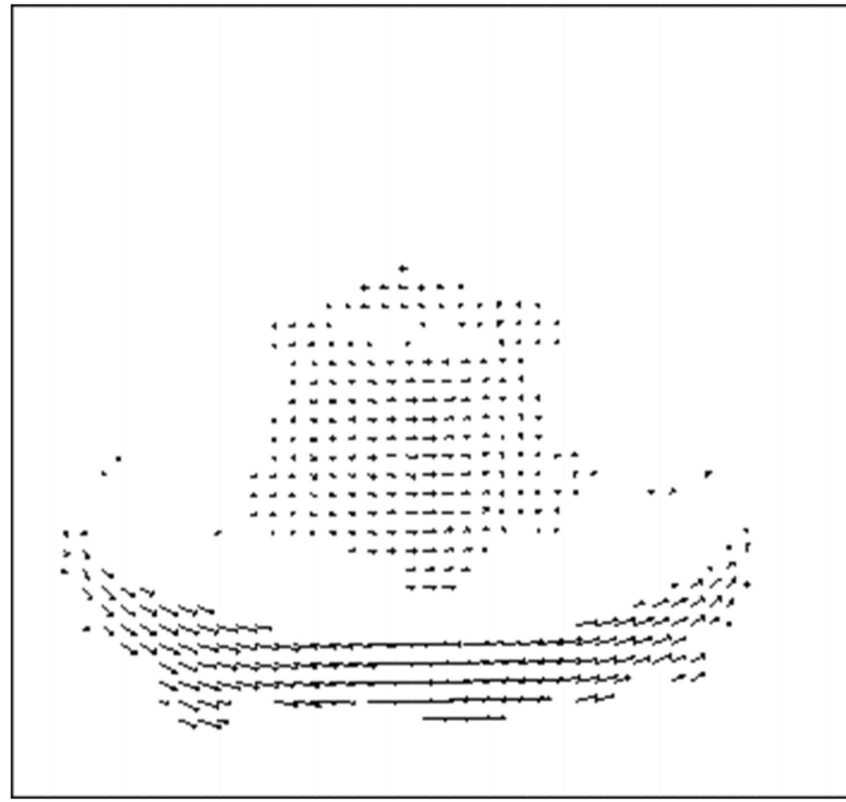
Magic: ghost removal

M. Uyttendaele, A. Eden, and R. Szeliski.
***Eliminating ghosting and exposure artifacts in image mosaics*.**
In Proceedings of the Interational Conference on Computer Vision and Pattern Recognition, volume 2, pages 509--516, Kauai, Hawaii, December 2001.

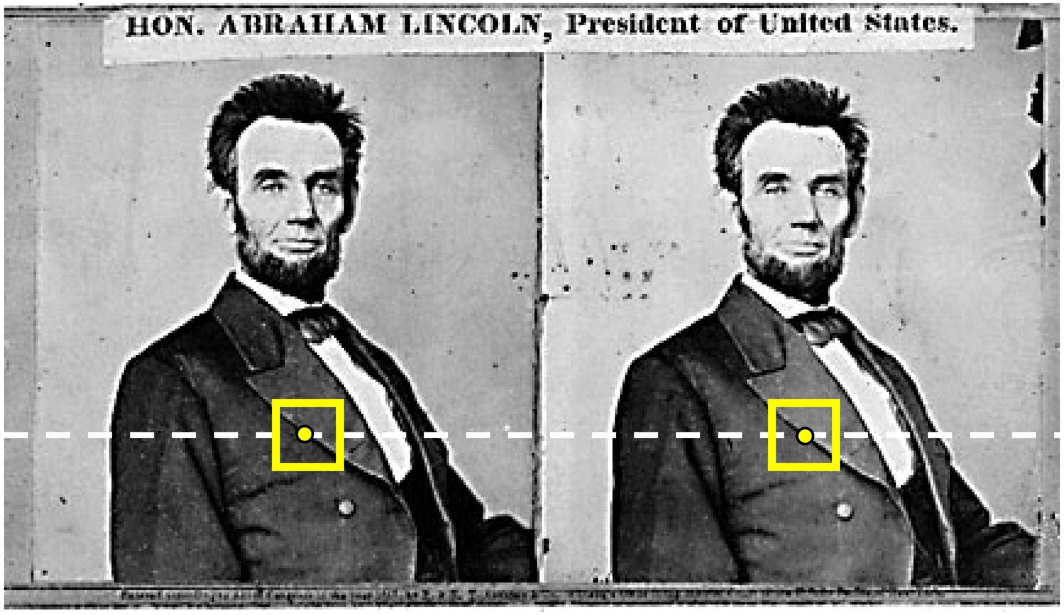
Magic: ghost removal

M. Uyttendaele, A. Eden, and R. Szeliski.
***Eliminating ghosting and exposure artifacts in image mosaics*.**
In Proceedings of the Interational Conference on Computer Vision and Pattern Recognition, volume 2, pages 509--516, Kauai, Hawaii, December 2001.

Optical flow



Your basic stereo algorithm



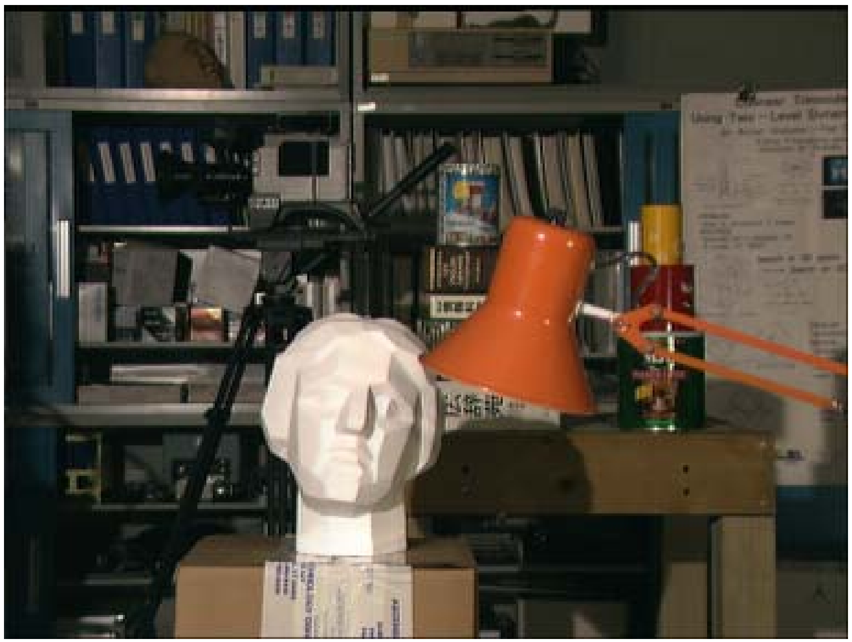
For each epipolar line

For each pixel in the left image

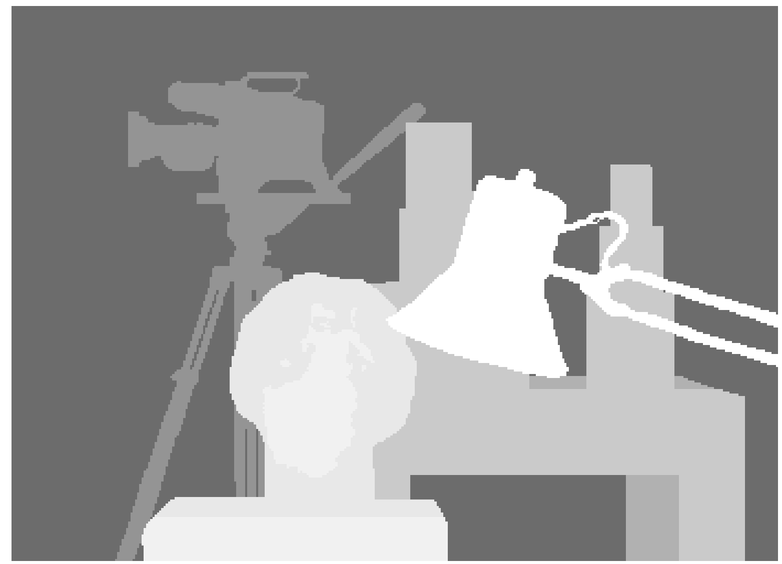
- compare with every pixel on same epipolar line in right image
- pick pixel with minimum match cost

Improvement: match ***windows***

Stereo results



Scene



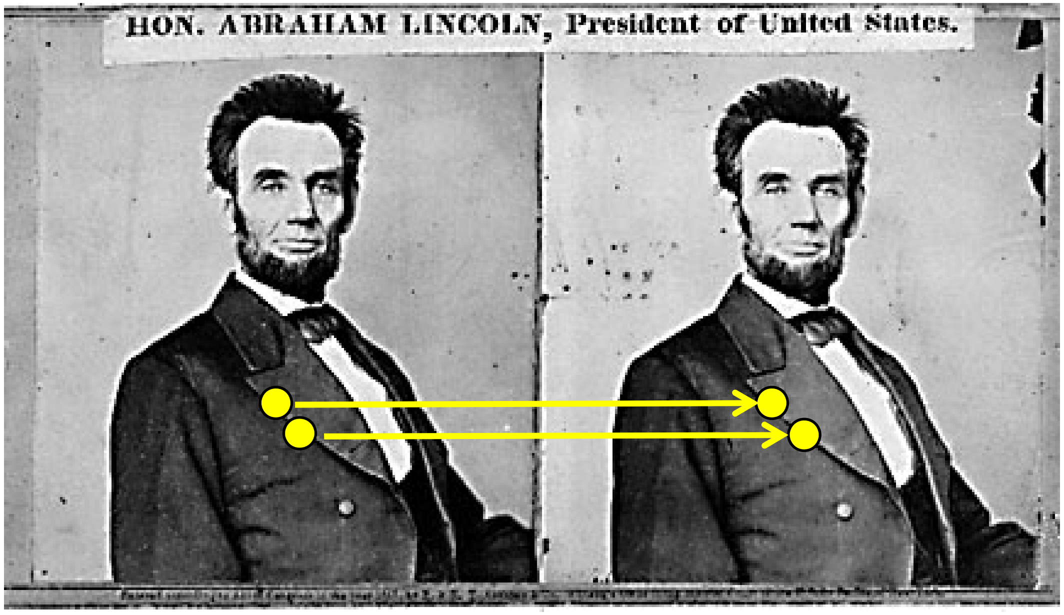
Ground truth

Results with window search

Window-based matching
(best window size)

Ground truth

Can we do better?



- What defines a good stereo correspondence?
  1. Match quality
     - Want each pixel to find a good match in the other image
  2. *Smoothness*
     - two adjacent pixels should (usually) move about the same amount

Stereo as energy minimization

- Find disparities *d* that minimize 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*, *y* + *d*(*x*,*y*))

Stereo as energy minimization

$I(x, y)$

$J(x, y)$

$C(x, y, d)$; 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

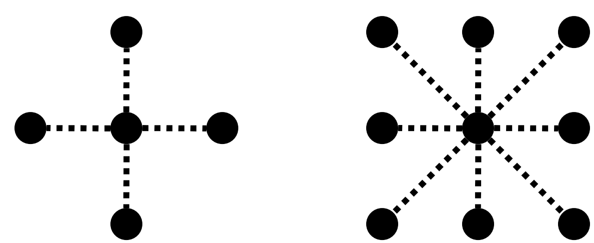
Stereo as energy minimization

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

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

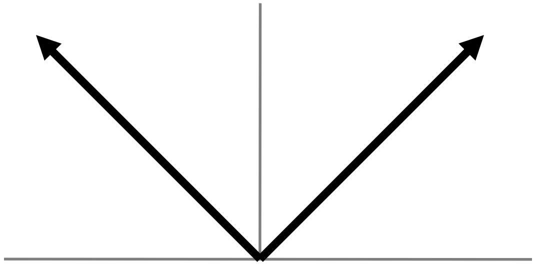
Smoothness cost

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

last time: looked at quadratic and truncated quadratic models for *V*

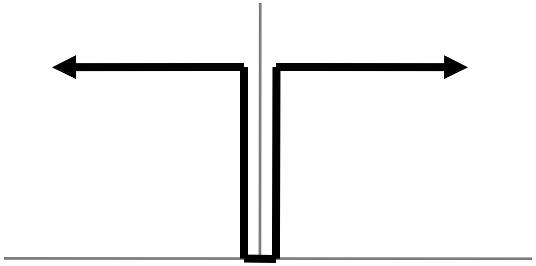
$$V(d_p, d_q) = |d_p - d_q|$$

L_1 distance



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

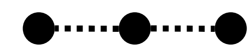
"Potts model"



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*

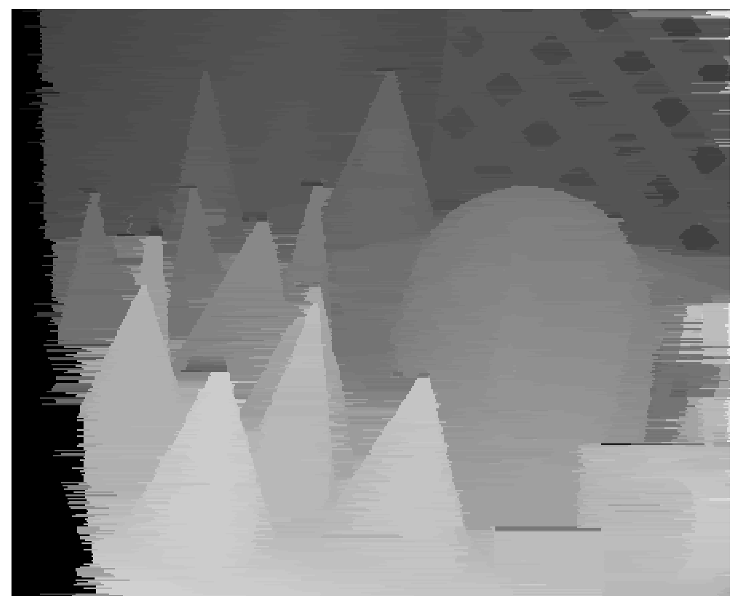
$$D(x, y, d) = C(x, y, d) + \min_{d'} \{D(x-1, y, d') + \lambda\, |d - d'|\}$$

Dynamic programming

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

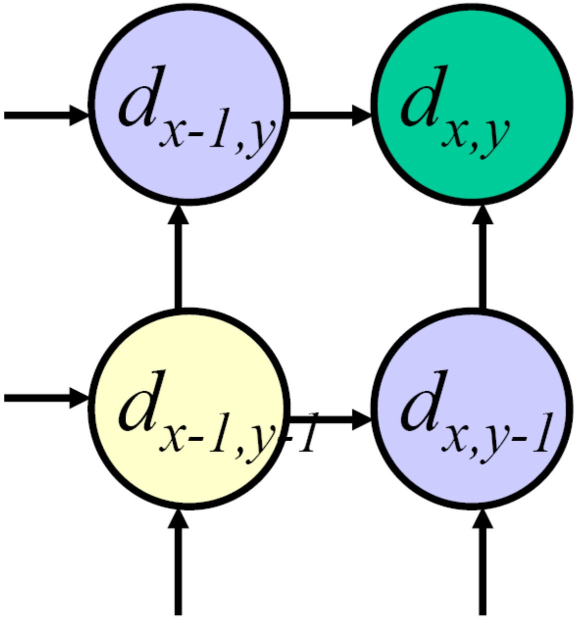
Dynamic Programming





Dynamic programming

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



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

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
  - Simulated annealing works a little better
- And a large search space
  - n x m image w/ k disparities has k^{nm} possible solutions
  - Finding the global minimum is NP-hard
- Good approximations exist…

Interlude: binary segmentation

- Suppose we want to segment an image into foreground and background



Interlude: binary segmentation

- Suppose we want to segment an image into foreground and background

User sketches out a few strokes on foreground and background…

How do we classify the rest of the pixels?

Binary segmentation as energy minimization

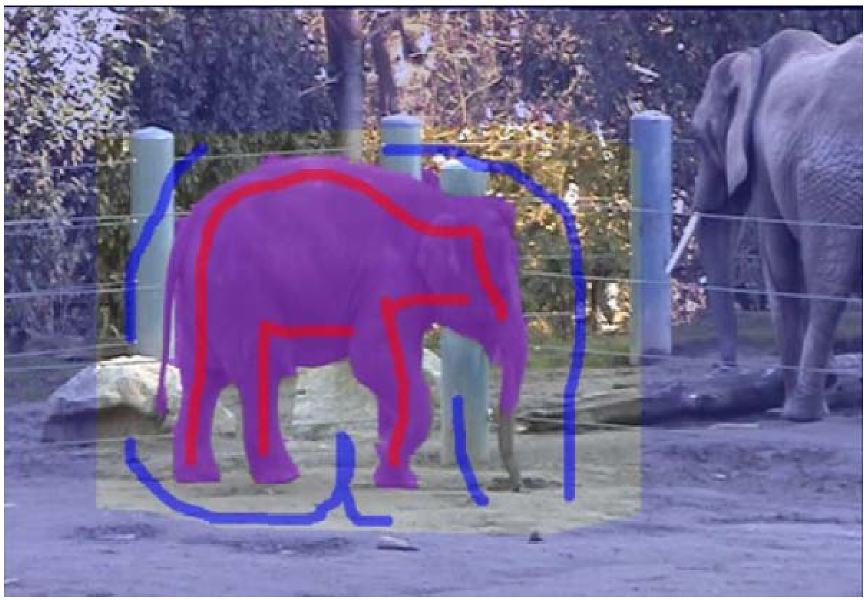
- Define a labeling *L* as an assignment of each pixel with a 0-1 label (background or foreground)

- Problem statement: find the labeling *L* that minimizes

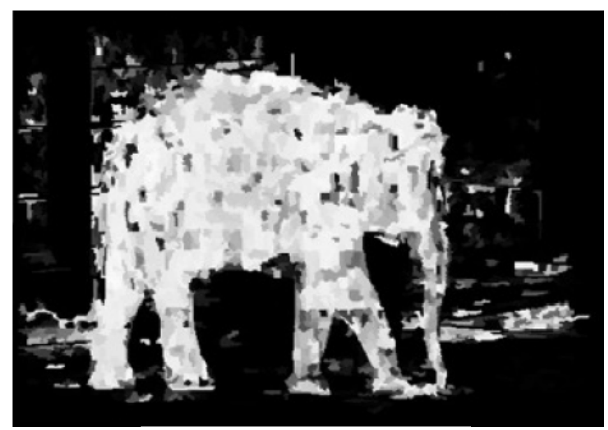
$$E(L) = \underbrace{E_d(L)}_{\text{match cost}} + \lambda \underbrace{E_s(L)}_{\text{smoothness cost}}$$

("how similar is each labeled pixel to the foreground / background?")

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



$C'(x, y, 0)$



$C'(x, y, 1)$

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

- Neighboring pixels should generally have the same labels
  - Unless the pixels have very different intensities

w_{pq} = 10.0

w_{pq} = 0.1

$$E_s(L) = \sum_{\text{neighbors } (p,q)} w_{pq} |L(p) - L(q)|$$

w_{pq} : similarity in intensity of *p* and *q*

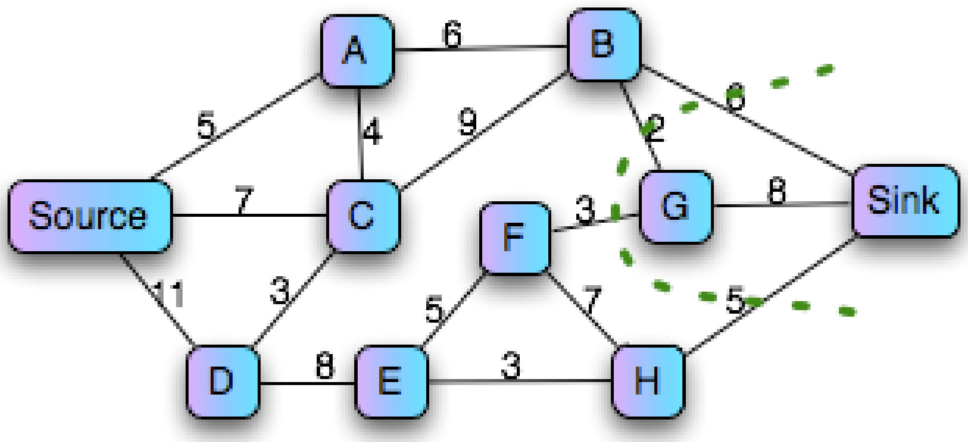
(can use the same trick for stereo)

Binary segmentation as energy minimization

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

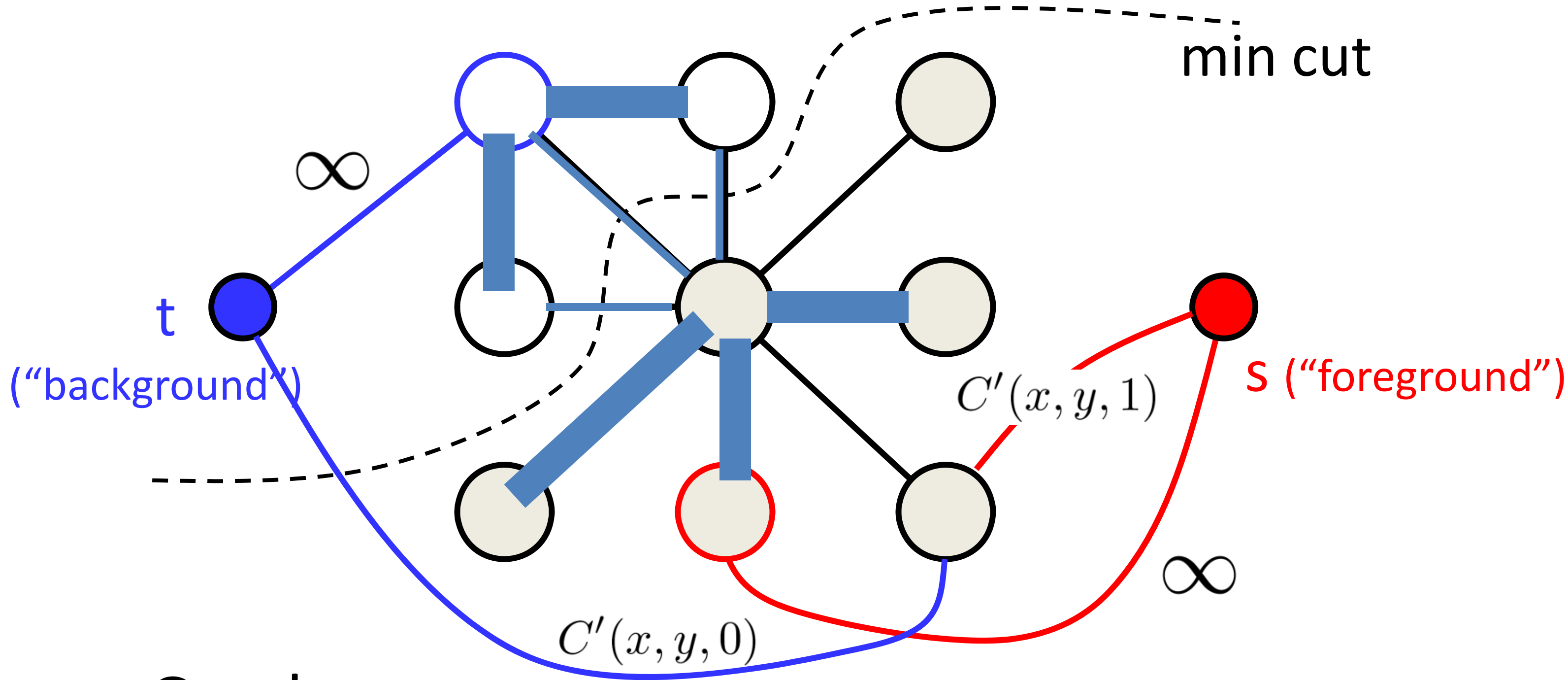
- For this problem, we can easily find the global minimum!
- Use max flow / min cut algorithm

Graph min cut problem



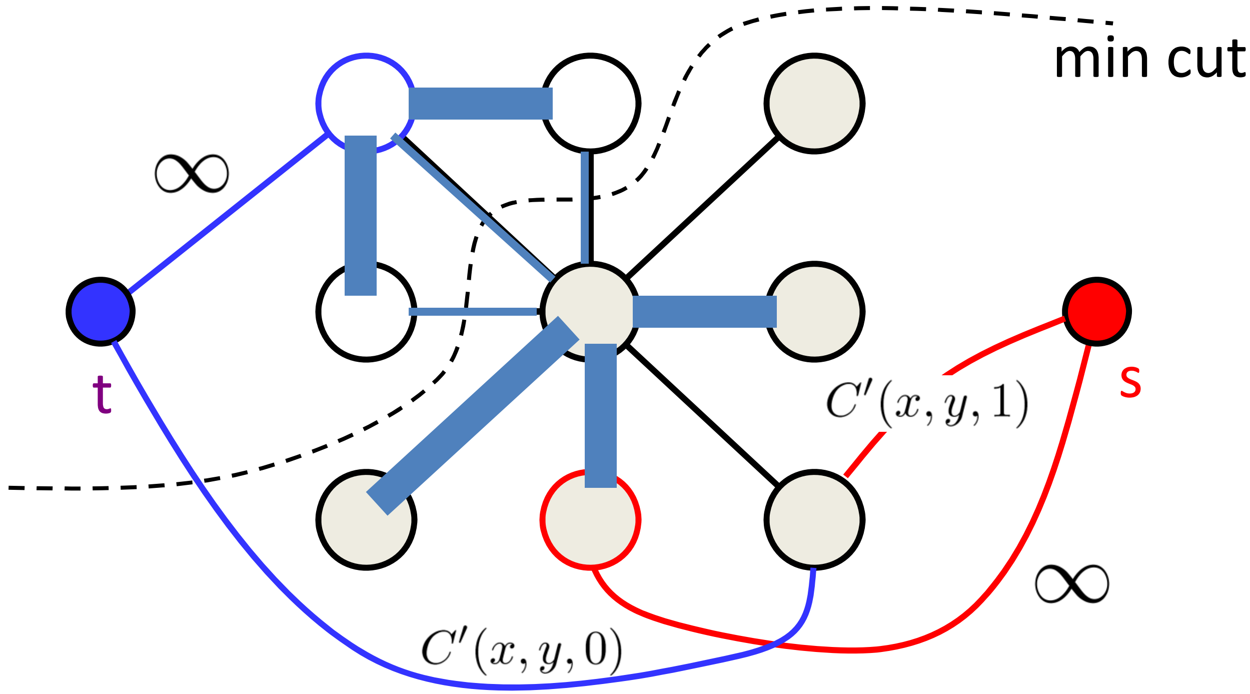
- Given a weighted graph *G* with source and sink nodes (*s* and *t*), partition the nodes into two sets, *S* and *T* such that the sum of edge weights spanning the partition is minimized
  - and $s \in S$ and $t \in T$

Segmentation by min cut



- Graph
  - node for each pixel, link between adjacent pixels
  - specify a few pixels as foreground and background
    - create an infinite cost link from each bg pixel to the *t* node
    - create an infinite cost link from each fg pixel to the *s* node
    - create finite cost links from *s* and *t* to each other node
  - compute min cut that separates s from t
    - The min-cut max-flow theorem [Ford and Fulkerson 1956]

Segmentation by min cut



- The partitions *S* and *T* formed by the min cut give the optimal foreground and background segmentation
- I.e., the resulting labels minimize

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

GrabCut

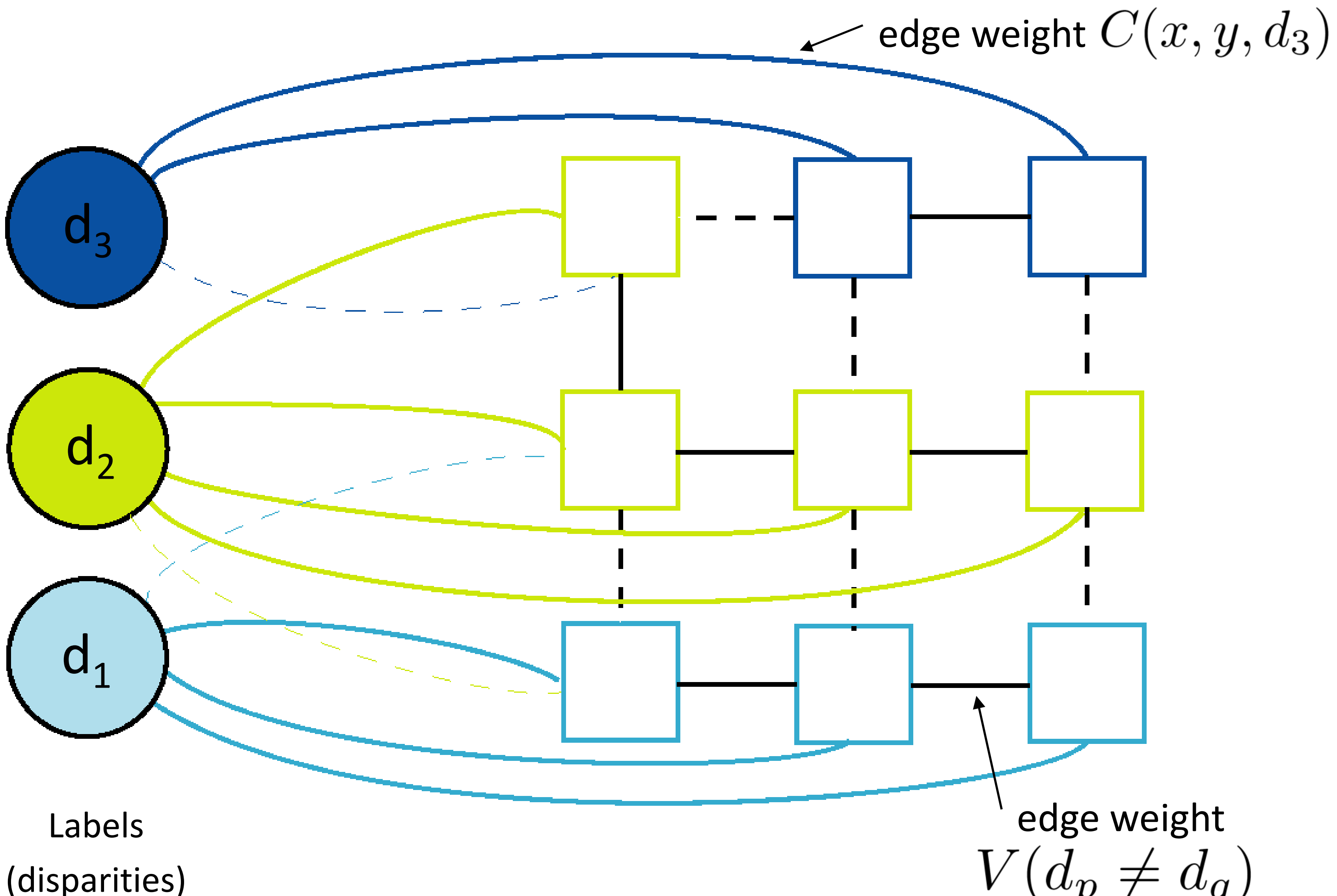
Grabcut [Rother et al., SIGGRAPH 2004]

Back to stereo

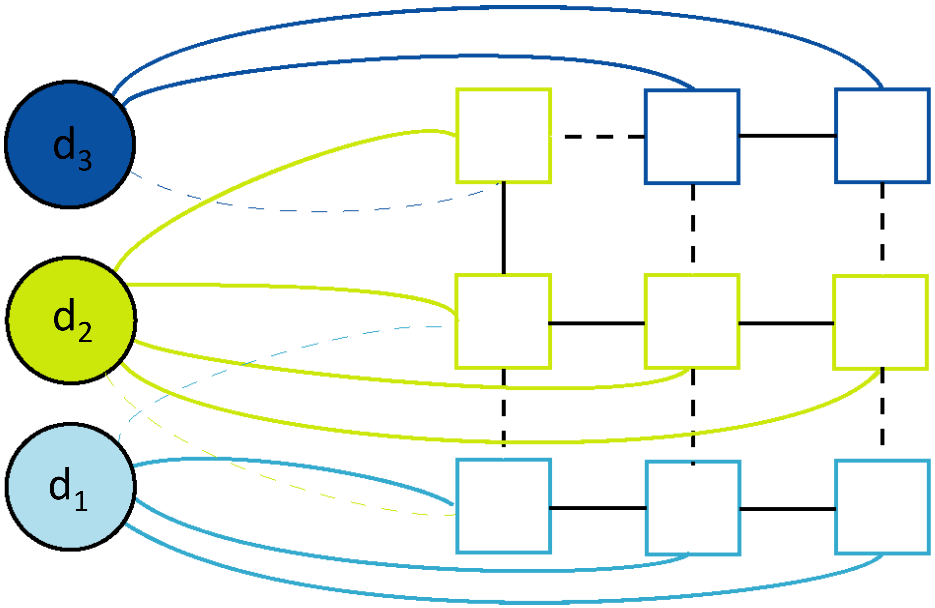
- Can formulate as a (no longer binary) labeling problem
  - with one label per disparity
- Can create similar setup as with segmentation, but with *k* source/sink nodes
  - *k* = number of disparities
  - Using the Potts model, the setup is straightforward

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

Energy minimization via graph cuts



Energy minimization via graph cuts



- Graph Cut
  - Delete enough edges so that
    - each pixel is connected to exactly one label node
  - Cost of a cut: sum of deleted edge weights
  - Finding min cost cut equivalent to finding global minimum of energy function

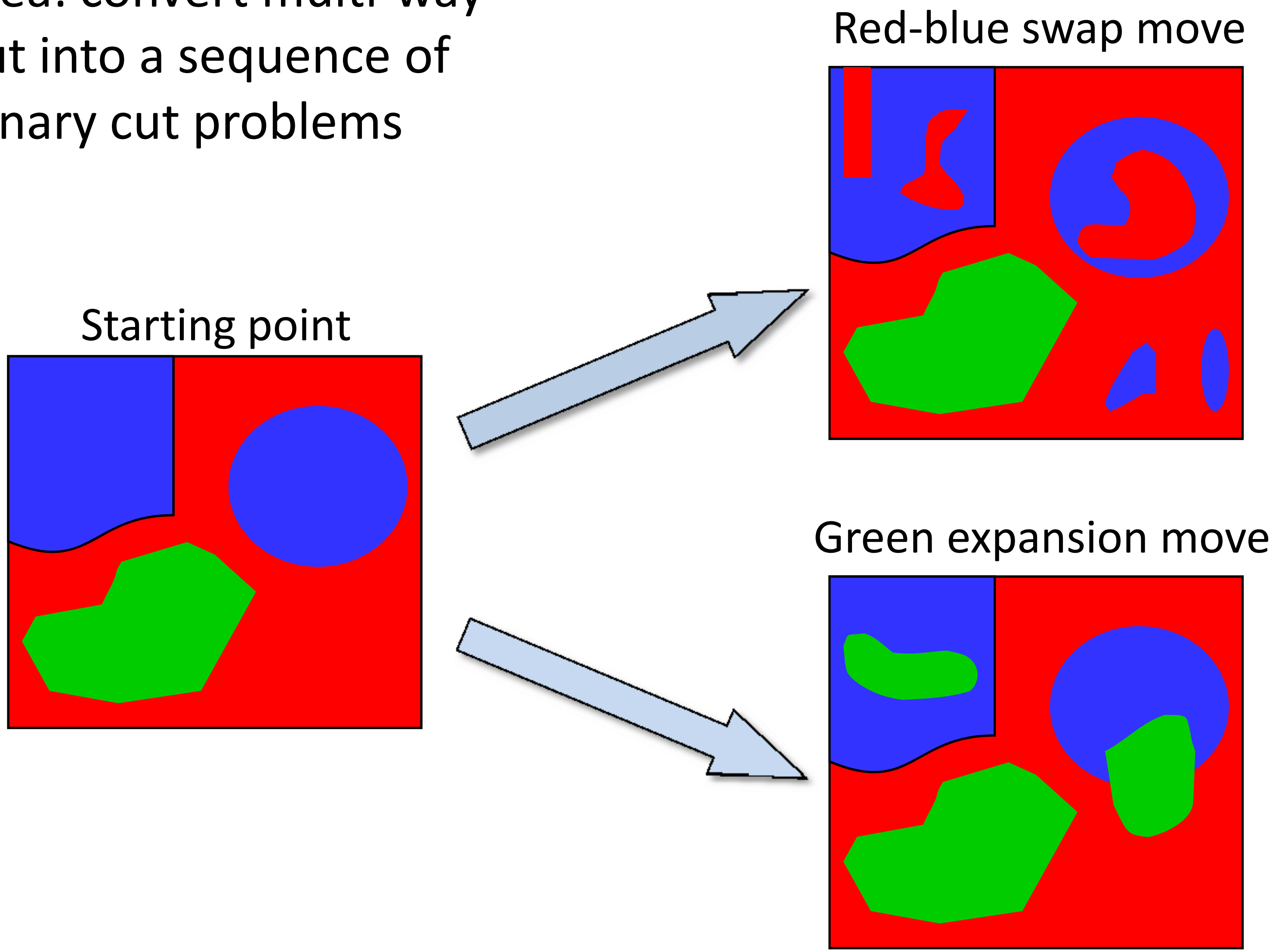
Computing a multiway cut

- With 2 labels: classical min-cut problem
  - Solvable by standard flow algorithms
    - polynomial time in theory, nearly linear in practice
  - More than 2 terminals: NP-hard
    [Dahlhaus *et al.*, STOC '92]

- Efficient approximation algorithms exist
  - Boykov, Veksler and Zabih, Fast Approximate Energy Minimization via Graph Cuts, ICCV 1999.
  - Within a factor of 2 of optimal
  - Computes local minimum in a strong sense
    - even very large moves will not improve the energy

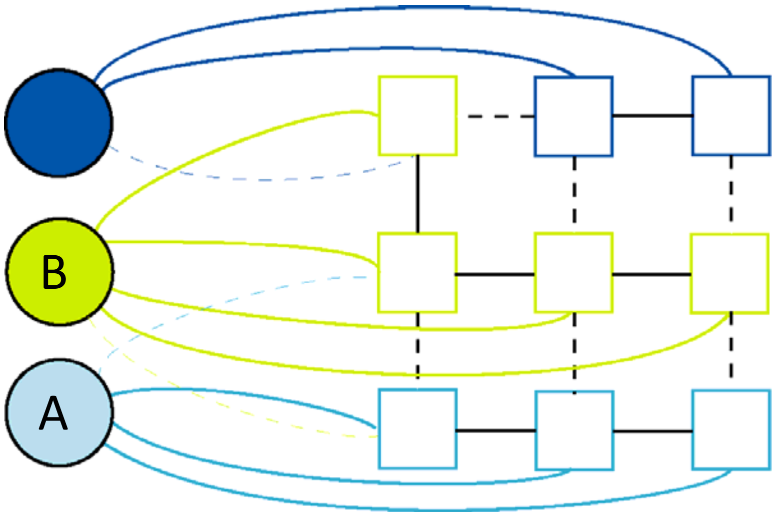
Move examples

Idea: convert multi-way cut into a sequence of binary cut problems

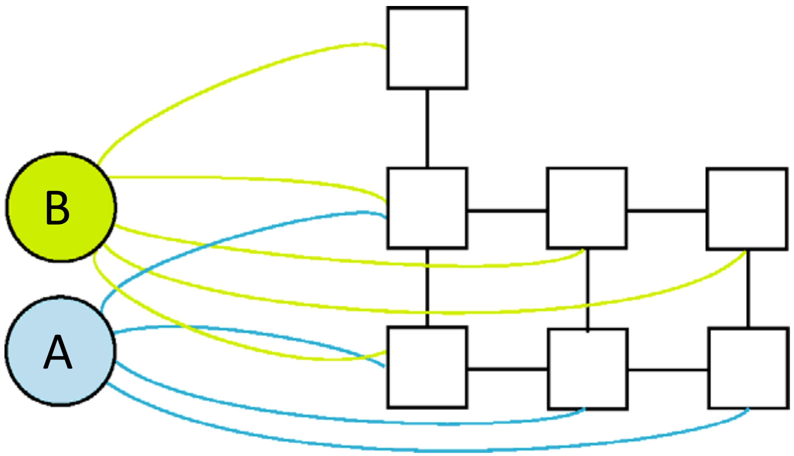


The swap move algorithm

1. Start with an arbitrary labeling
2. Cycle through every label pair *(A,B)* in some order
   2.1 Find the lowest *E* labeling within a single *AB*-swap
   2.2 Go there if it's lower *E* than the current labeling
3. If *E* did not decrease in the cycle, we're done
   Otherwise, go to step 2

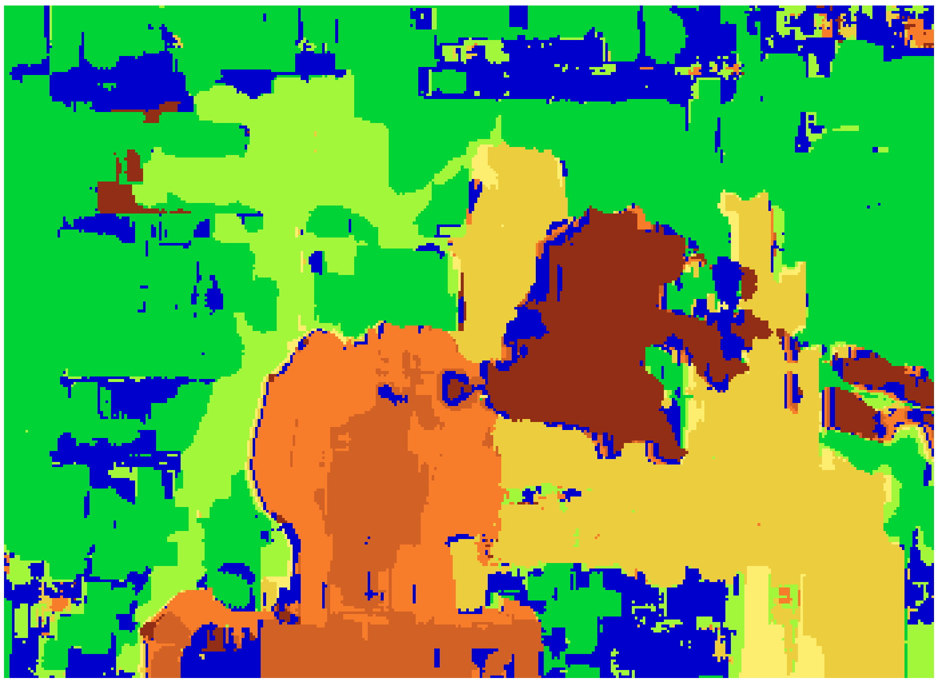


Original graph



AB subgraph
(run min-cut on this graph)

Results with window correlation



normalized correlation
(best window size)

ground truth

Results with graph cuts

graph cuts
(Potts model,
expansion move algorithm)

ground truth

Other energy functions

- Can optimize other functions (exactly or approximately) with graph cuts

$$V(d_p, d_q) = (d_p - d_q)^2$$

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

Questions?

Real-time stereo

Nomad robot 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 exist (most based on simple search)

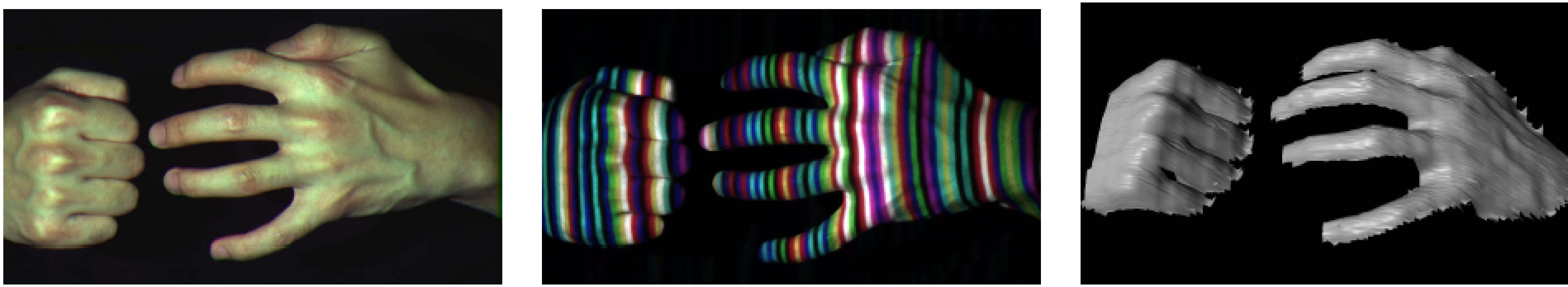
Stereo reconstruction pipeline

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

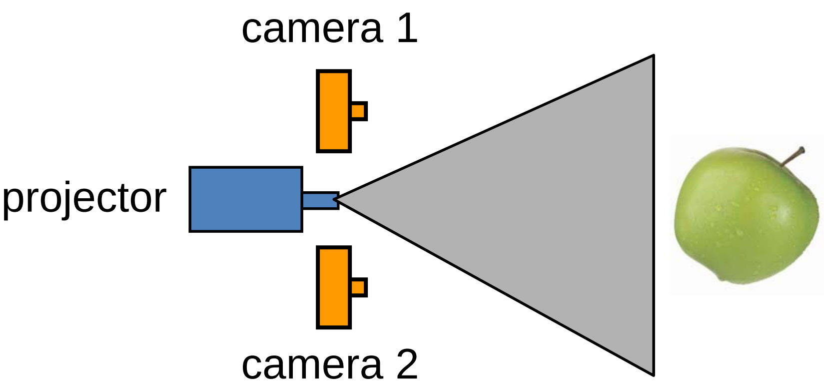
What will cause errors?

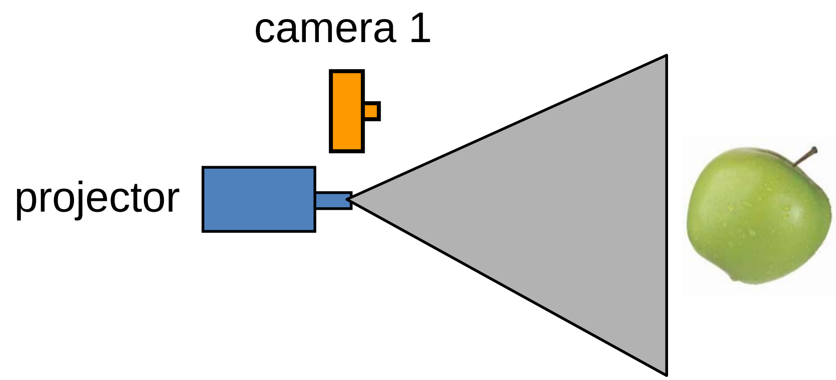
- Camera calibration errors
- Poor image resolution
- Occlusions
- Violations of brightness constancy (specular reflections)
- Large motions
- Low-contrast image regions

Active stereo with structured light



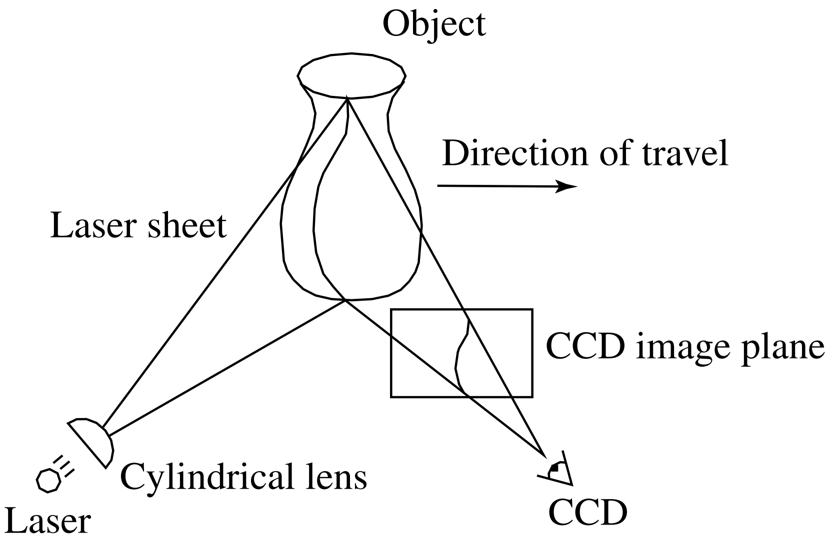
Li Zhang's one-shot stereo

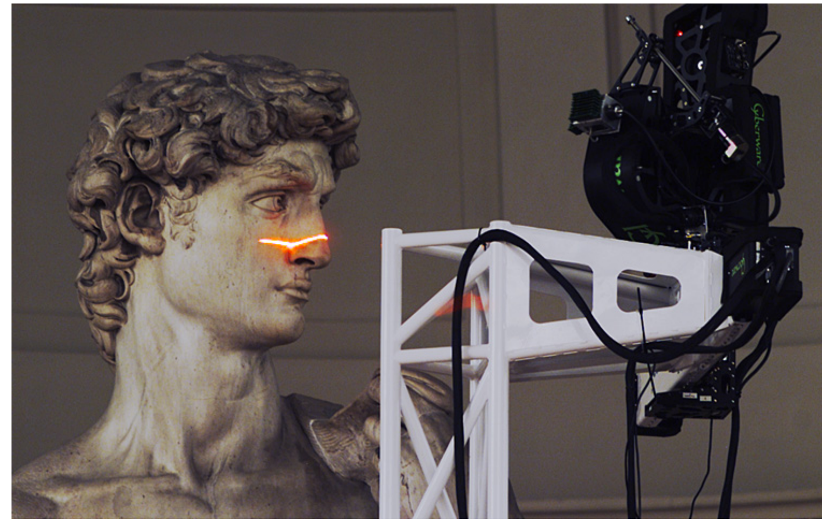




- Project "structured" light patterns onto the object
  - simplifies the correspondence problem

Laser scanning





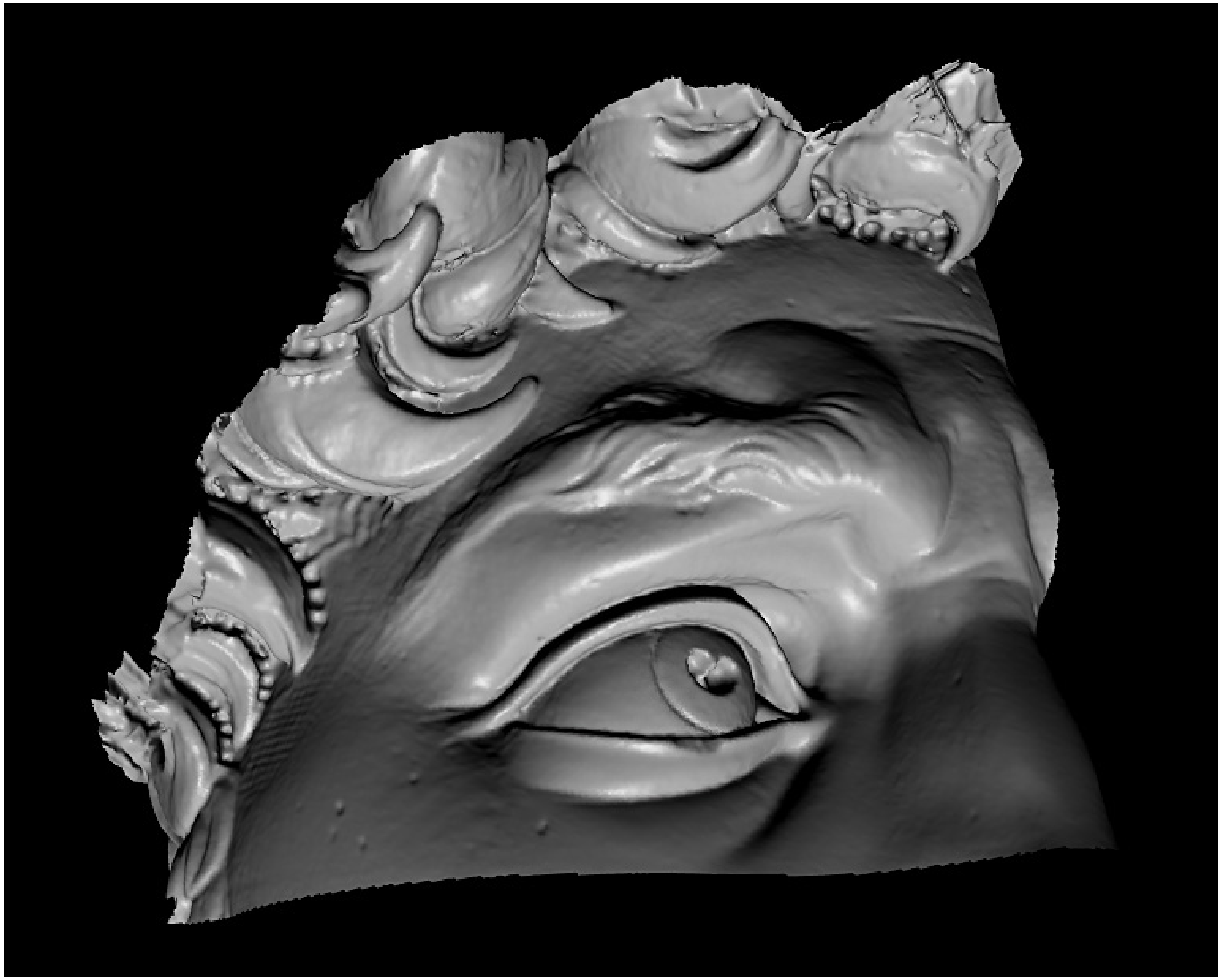
Digital Michelangelo Project
http://graphics.stanford.edu/projects/mich/

- 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

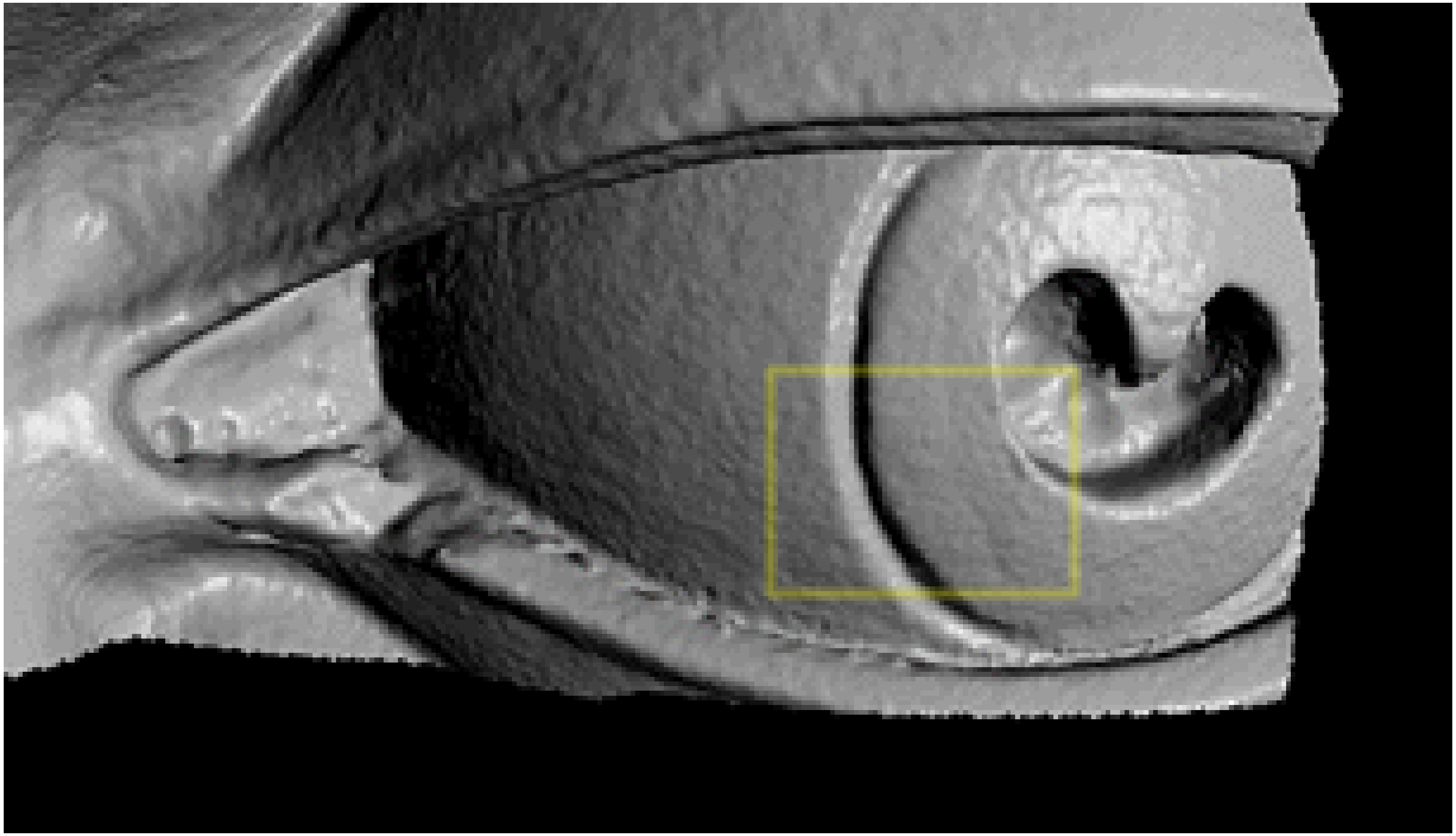
The Digital Michelangelo Project, Levoy et al.

Laser scanned models



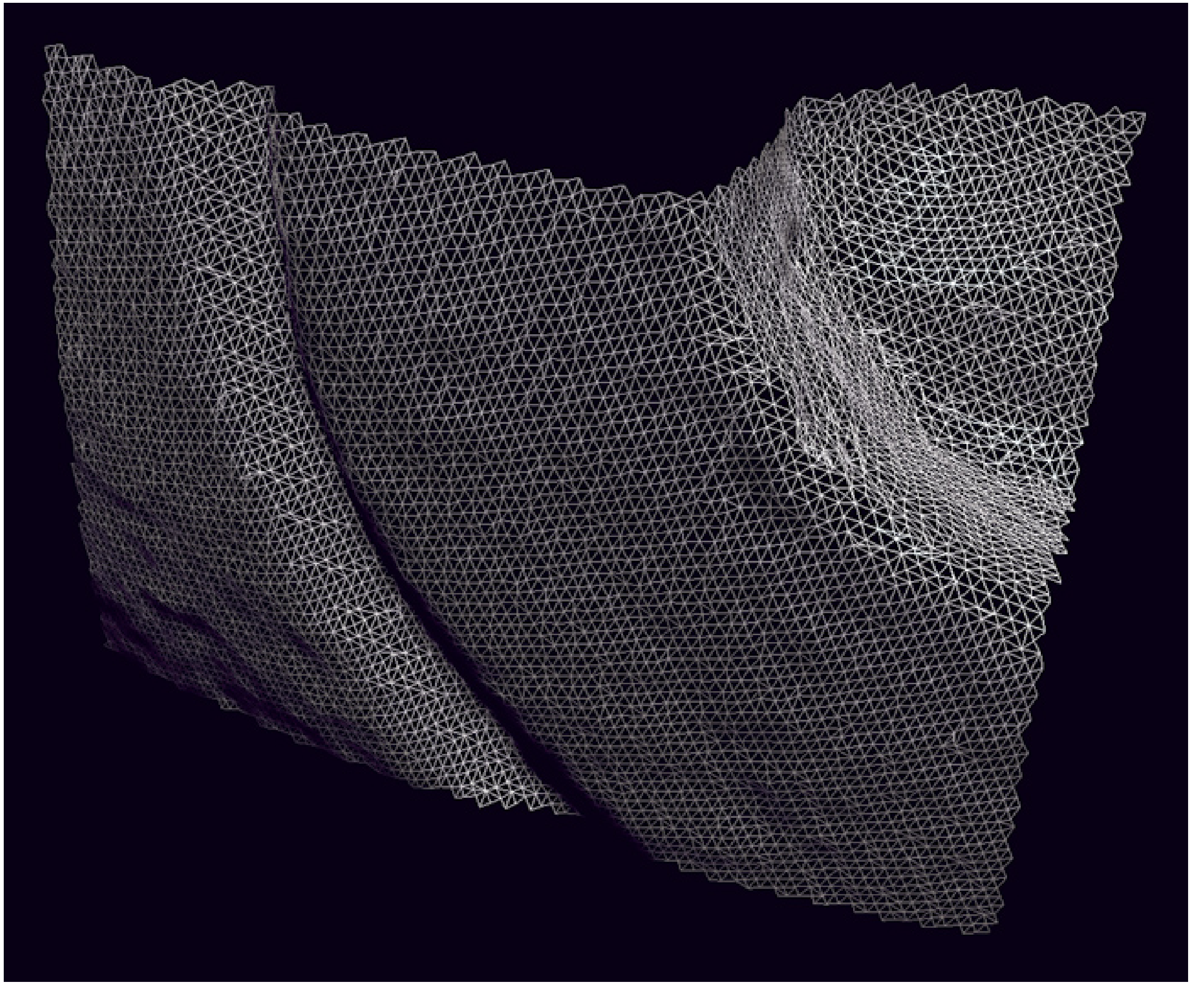
The Digital Michelangelo Project, Levoy et al.

Laser scanned models



The Digital Michelangelo Project, Levoy et al.

Laser scanned models



The Digital Michelangelo Project, Levoy et al.

Spacetime Stereo

http://grail.cs.washington.edu/projects/stfaces/

Li Zhang, Noah Snavely,
Brian Curless, Steve Seitz