CSCI 497P/597P: Computer Vision

Scott Wehrwein

Depth From Disparity, Stereo Matching

CMV: Panorama Stitching is a Solved Problem

Goals

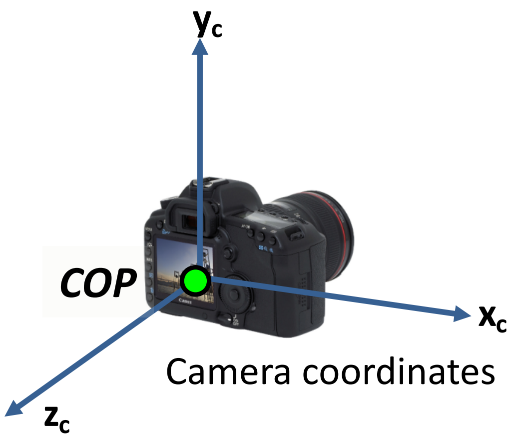
- Know how to calculate depth from disparity in a rectified stereo setup
- Understand why stereo matching is the hard part of stereo vision.
- Know the definition and formation of the stereo cost volume.
- Understand the basic metrics used to compare patches (SSD, SAD, NCC)

Announcements

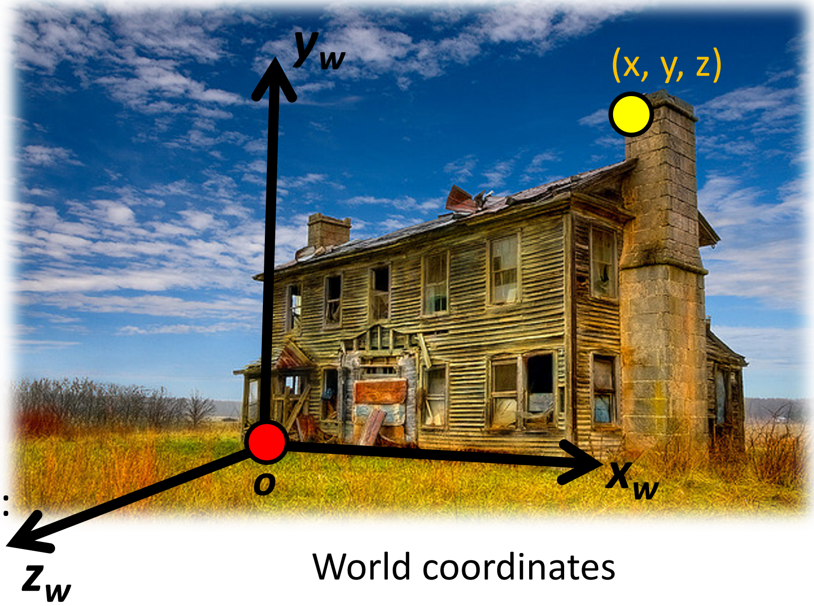
- Exam (out Monday, due Tuesday)
- P2 (due Monday)
  - don’t forget to fill out the P2 Survey
  - no need to email me for slip days
  - artifacts due Tuesday

It’s not always about you(r camera).

- We’ve assumed that 3D points are represented in “camera coordinates” (i.e., origin = COP, -z = optical axis).
- How can we model the geometry of a camera in a separate world coordinate system?



Camera coordinates



World coordinates

Three important coordinate systems:

1. *World* coordinates
2. *Camera* coordinates
3. *Pixel* coordinates

How do we project a given point (x, y, z) in *world* coordinates?

Intrinsic Camera Parameters

Everything you need to get from **camera** coordinates to **pixel** coordinates:

$$\underbrace{\begin{bmatrix} -f & 0 & 0 \\ 0 & -f & 0 \\ 0 & 0 & 1 \end{bmatrix}}_{\mathbf{K}} \begin{bmatrix} 1 & 0 & 0 & 0 \\ 0 & 1 & 0 & 0 \\ 0 & 0 & 1 & 0 \end{bmatrix}$$

(intrinsics)

(converts from 3D rays in camera coordinate system to pixel coordinates)

more generally, $\mathbf{K} = \begin{bmatrix} -f & s & c_x \\ 0 & -\alpha f & c_y \\ 0 & 0 & 1 \end{bmatrix}$ (upper triangular matrix)

α : **aspect ratio** (1 unless pixels are not square)

s : **skew** (0 unless pixels are shaped like rhombi/parallelograms)

(c_x, c_y) : **principal point** ((0,0) unless optical axis doesn’t intersect projection plane at origin)

Extrinsic Camera Parameters

- Everything you need to get from **world** coordinates to **camera** coordinates

$K[R;t]$

$$\begin{bmatrix} & \mathbf{R} & & \begin{matrix}0\\0\\0\end{matrix} \\ 0 & 0 & 0 & 1 \end{bmatrix} \begin{bmatrix} \mathbf{I}_{3\times 3} & -\mathbf{c} \\ 0 \quad 0 \quad 0 & 1 \end{bmatrix}$$

Step 2: Rotate by **R** Step 1: Translate by -**c**

$[R \mid t]$

$[R ; t]$

$$\begin{matrix} R & t \\ 0\,0\,0 & 1 \end{matrix}$$

Careful!

$$\cancel{\begin{bmatrix} R & -c \\ 0\,0\,0 & 1 \end{bmatrix}} \leftarrow$$

$R\vec{x} + \vec{c}$?

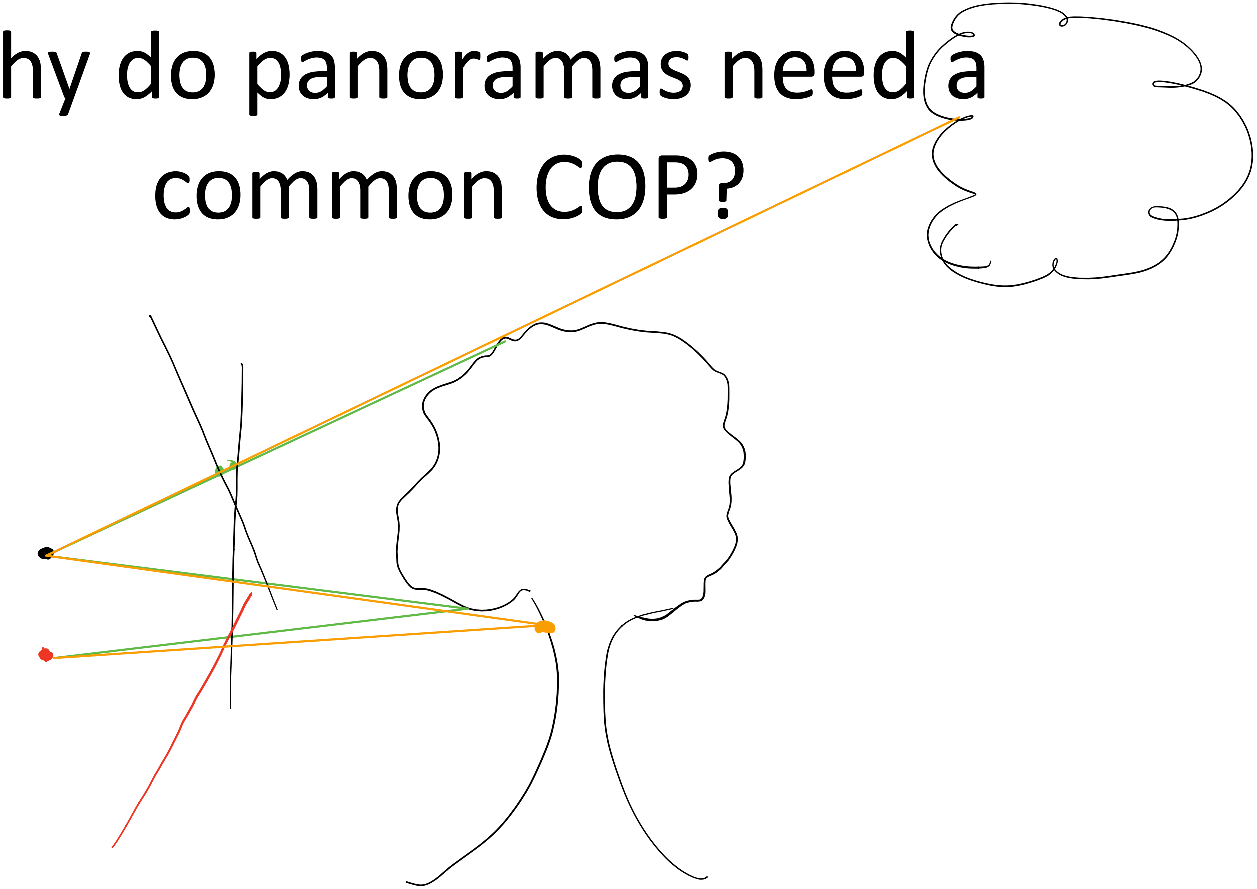
Projection matrix: Putting it all together

$$\boldsymbol{\Pi} = \underset{\text{intrinsics}}{\overset{\downarrow}{\mathbf{K}}} \underbrace{\underset{\text{projection}}{\overset{\downarrow}{\begin{bmatrix} 1 & 0 & 0 & 0 \\ 0 & 1 & 0 & 0 \\ 0 & 0 & 1 & 0 \end{bmatrix}}} \underset{\text{rotation}}{\overset{\downarrow}{\begin{bmatrix} \mathbf{R} & \begin{matrix} 0 \\ 0 \\ 0 \end{matrix} \\ \begin{matrix} 0 & 0 & 0 \end{matrix} & 1 \end{bmatrix}}} \underset{\text{translation}}{\overset{\downarrow}{\begin{bmatrix} \mathbf{I}_{3\times3} & -\mathbf{c} \\ \begin{matrix} 0 & 0 & 0 \end{matrix} & 1 \end{bmatrix}}}}$$

The **K** matrix converts 3D rays in the camera's coordinate system to 2D image points in image (pixel) coordinates.

This part converts 3D points in world coordinates to 3D rays in the camera's coordinate system. There are 6 parameters represented (3 for position/translation, 3 for rotation).

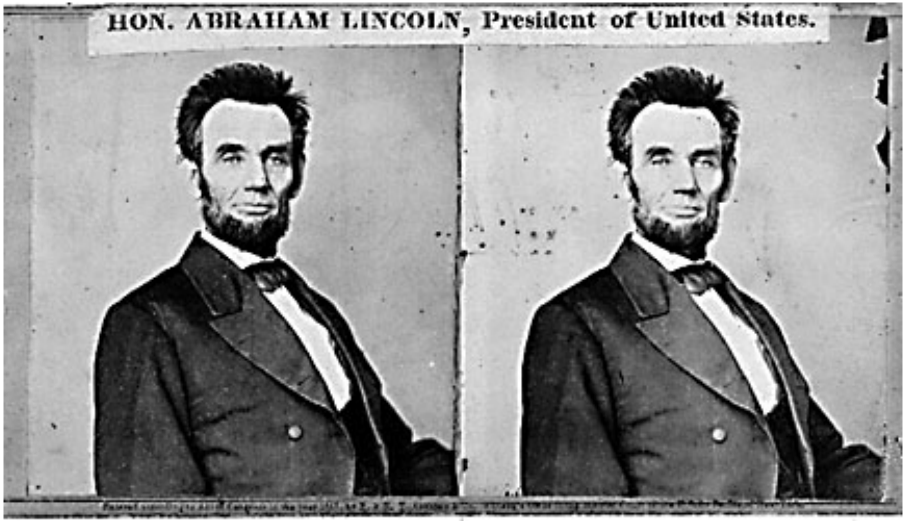
Why do panoramas need a common COP?



~~Bad~~ news: If the COPs are different, the fate of a ray **depends on its depth**.

Good?

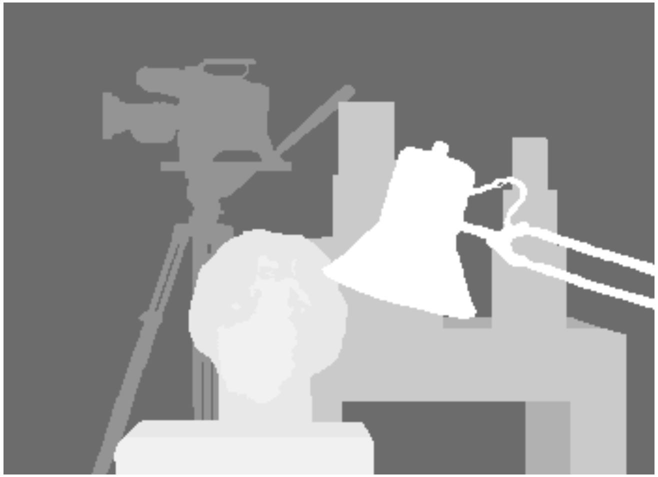
Stereo



- Given two images from different viewpoints (COPs)
  - How can we compute the depth of each point in the image?
  - Based on *how much each pixel moves* between the two images

Stereo

Left Image



Ground truth disparity map

- Given two images from different viewpoints (COPs)
  - How can we compute the depth of each point in the image?
  - Based on *how much each pixel moves* between the two images

Hypothesis generation time: what relationship do you expect to find between **depth** and **how much a pixel moves**?

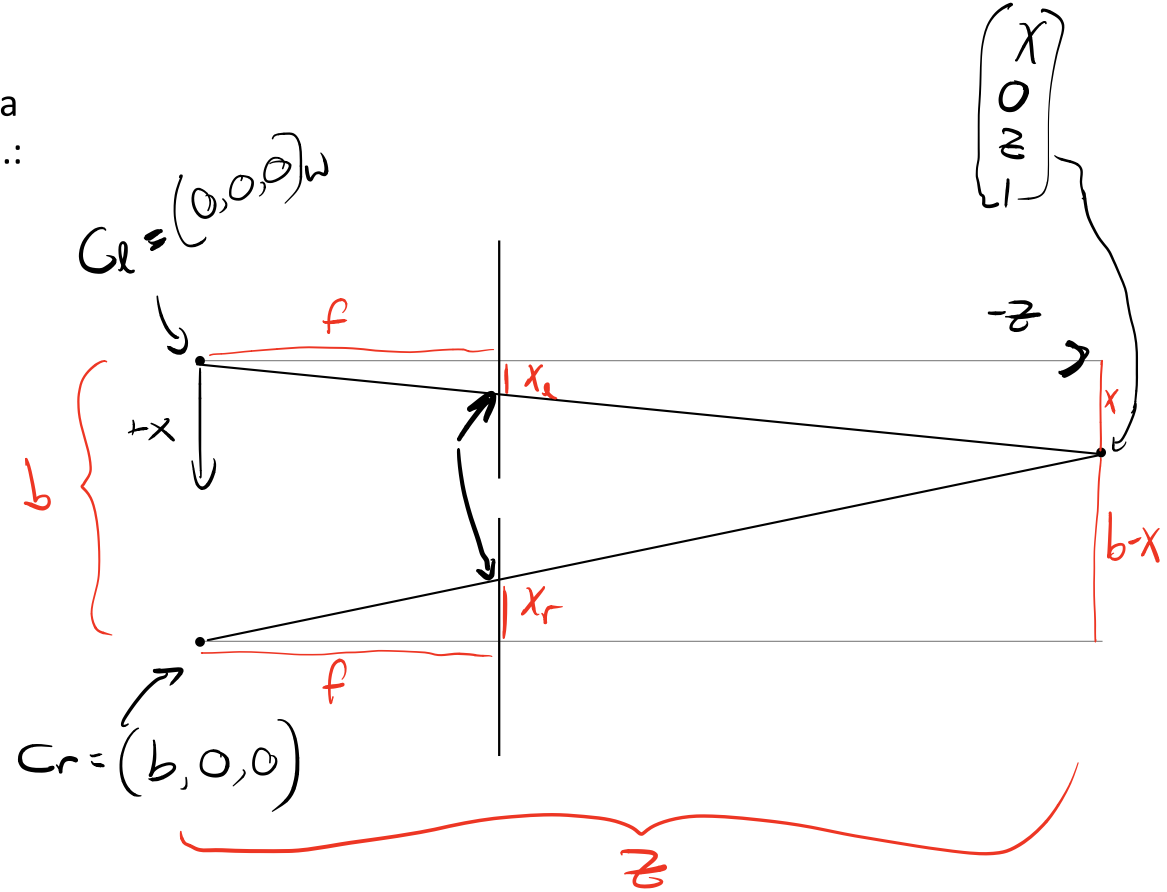
$$\text{depth} \propto \frac{1}{\text{disparity}}$$

Depth from Disparity

Assumption - we have a rectified stereo pair, i.e.:

- 2 cameras
- same f
- same PP
- COP off by b in x

↑ known



$$\frac{b-x}{x_r} = \frac{z}{f}$$

$$\frac{z}{f} = \frac{x}{x_\ell}$$

$$x = b - \frac{z x_r}{f}$$

$$\frac{z x_\ell}{f} = x$$

$$\frac{z x_\ell}{f} = b - \frac{z x_r}{f}$$

$$\frac{z x_\ell}{f} + \frac{z x_r}{f} = b \rightarrow \frac{z(x_\ell + x_r)}{f} = b$$

Local

baseline

depth

$$\boxed{z = \frac{fb}{(x_\ell + x_r)}}$$

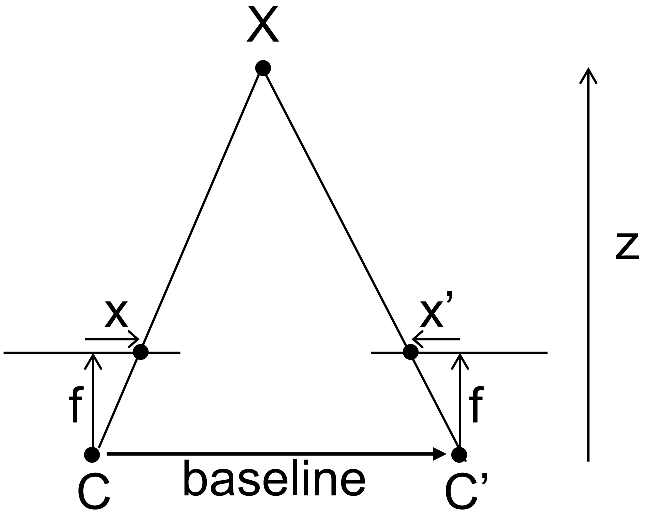
disparity

$$z \propto \frac{1}{\text{disparity}}$$

Note: if x_ℓ, x_r signed, disparity is

$$\underline{x_\ell - x_r}$$

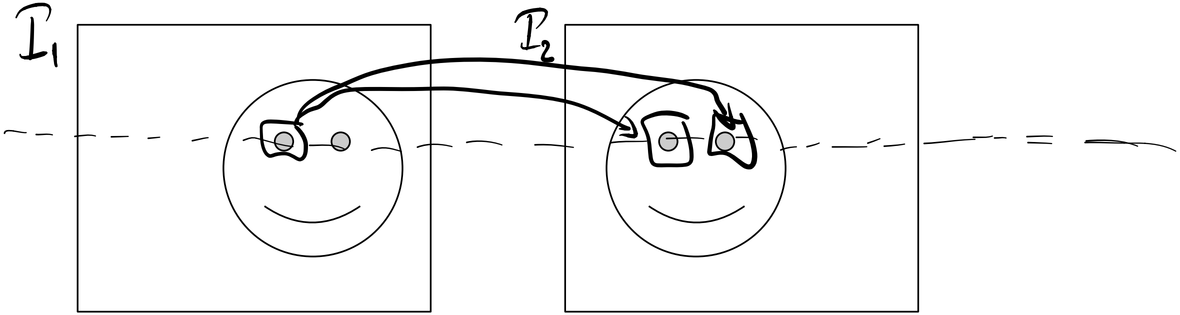
Depth from disparity



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

Stereo Depth Reconstruction: Approach

If we have a rectified stereo pair,



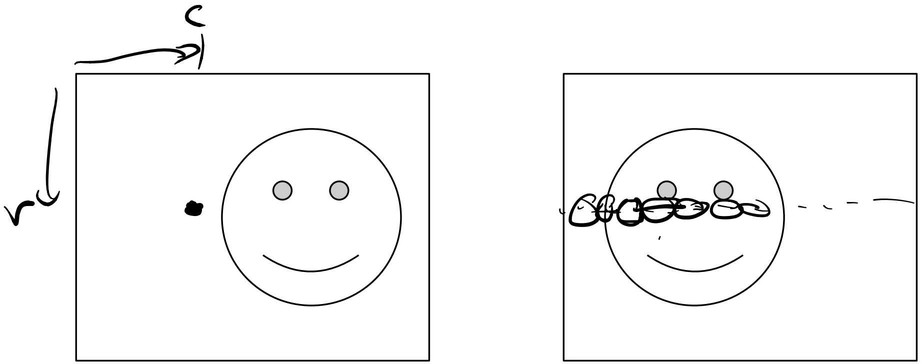
I can calculate depth if I know correspondence.

Good news: I only need to search a row!

Bad news: Ambiguity abounds!

Matching is the hard part of Stereo.

Stereo Depth Reconstruction: Algorithm



# of possible disparities

```
C = np.zeros(h, w, D)
for each row r:
    for each col c:
        for each disparity d:
```

$$C[r, c, \underset{\uparrow}{d}] = \text{match-cost}(I_l[r, c], I_r[r, c+d])$$

disparity = np.argmin(C, axis=2)

depth = f * b / disparity

Notes:

- C is the Cost Volume
- Looks at <u>windows</u> around pixels

The Cost Volume

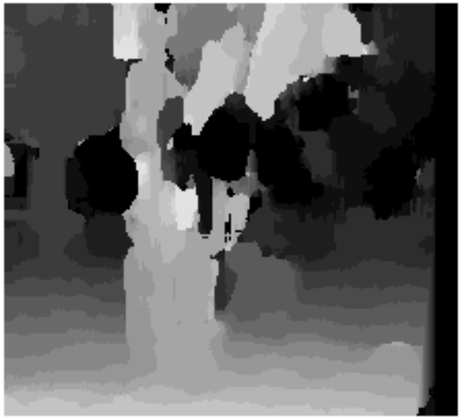
I(*x*, *y*)

J(*x*, *y*)

C(*x*, *y*, *d*); the cost volume (aka *disparity space image* (DSI))

Window size

W = 3



W = 20

Effect of window size

- Smaller window
  + detail
  – more noise
- Larger window
  + less noise
  • less detail

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

Metrics for Stereo Matching

- SSD = sum of squared differences
- SAD = sum of absolute differences
- NCC = normalized cross-correlation
  - (more ~~convolution~~cross correlation!)

Un-Normalized Cross Correlation

Insight: a cross-correlation filter is good at finding patches that **look like itself**.

Normalized Cross Correlation

Approach: apply a **patch** from one image as a **filter** across the other.
Trick: normalize patches before computing product to add invariance.

regions $\mathtt{A}, \mathtt{B}$, write as vectors $\mathbf{a}, \mathbf{b}$

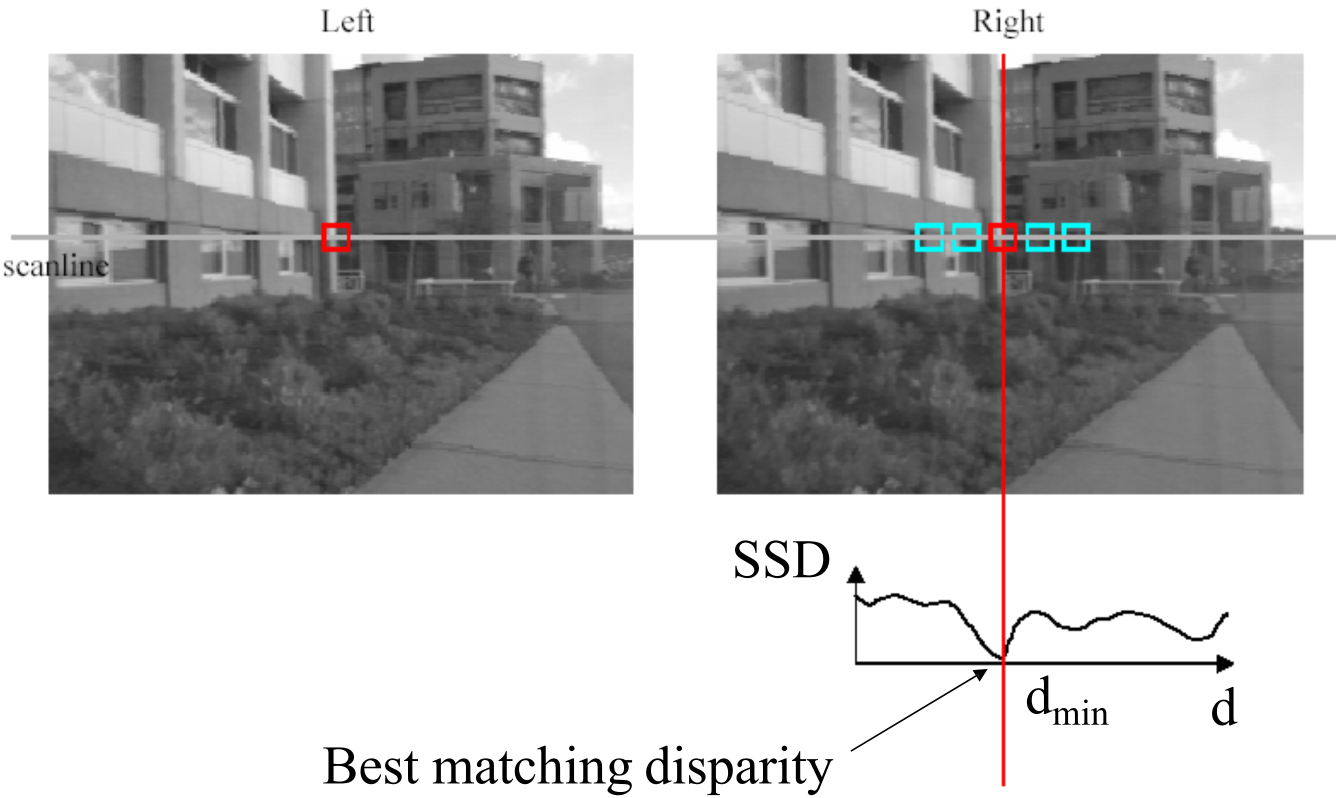
subtract the mean of each vector:

$$\mathtt{a} \rightarrow \mathtt{a} - \langle \mathbf{a} \rangle, \quad \mathtt{b} \rightarrow \mathtt{b} - \langle \mathbf{b} \rangle$$

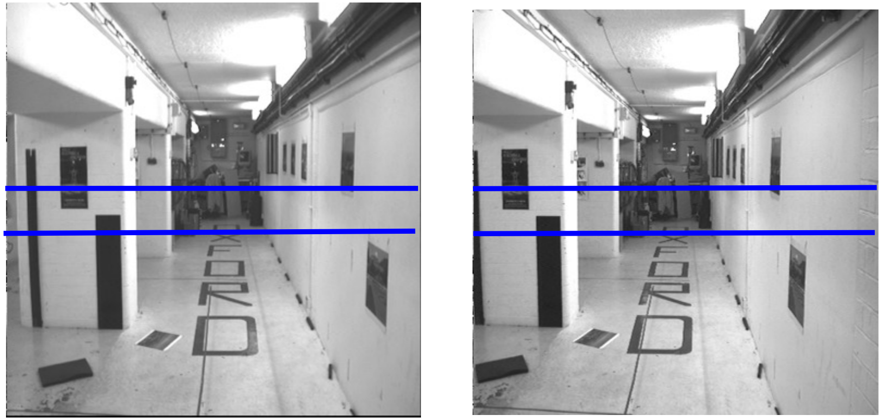
$$\text{cross correlation} = \frac{\mathbf{a}.\mathbf{b}}{|\mathbf{a}||\mathbf{b}|}$$

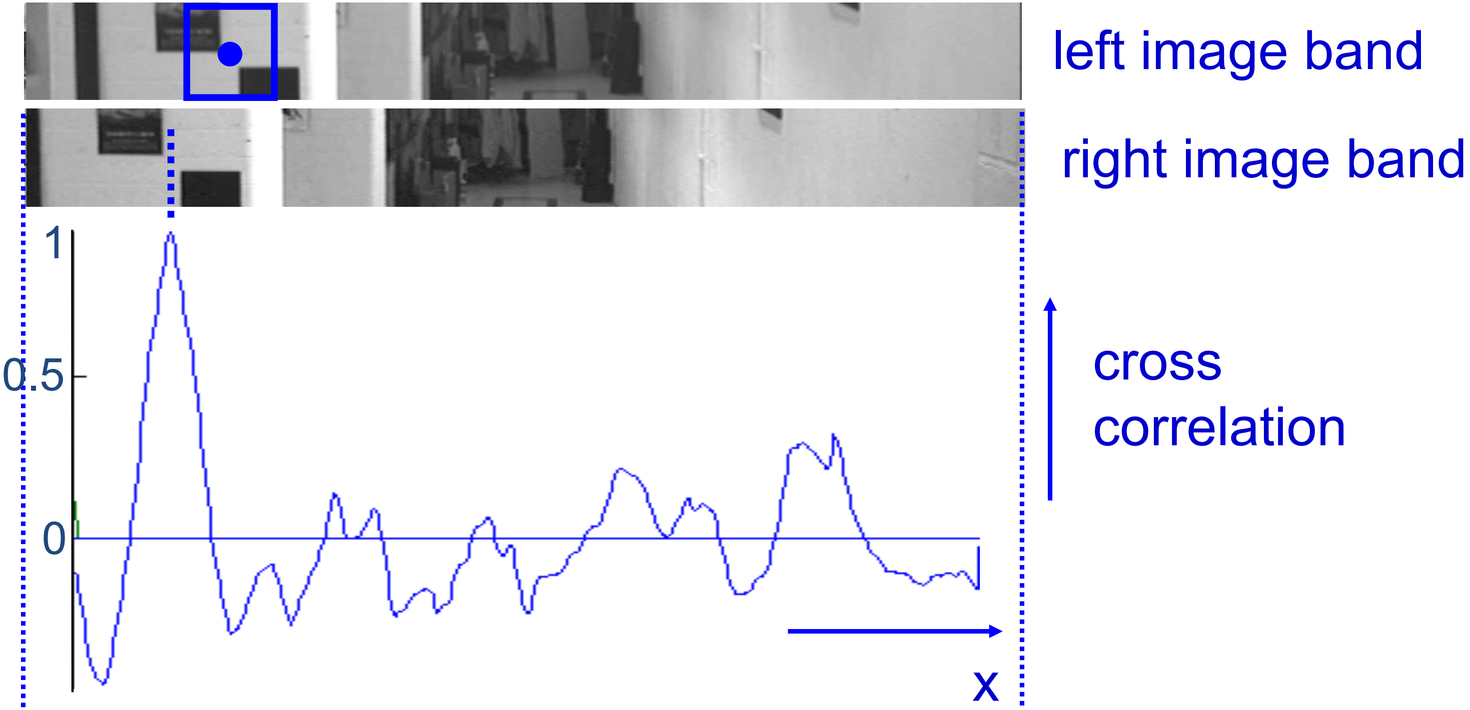
Invariant to $I \rightarrow \alpha I + \beta$

Stereo matching based on SSD

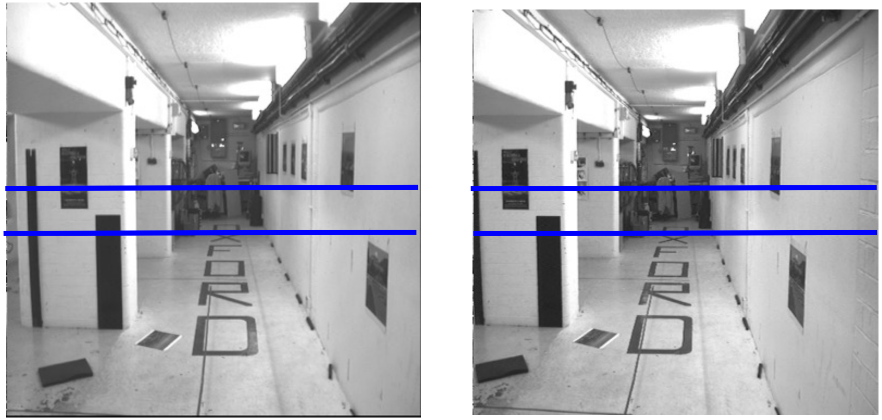


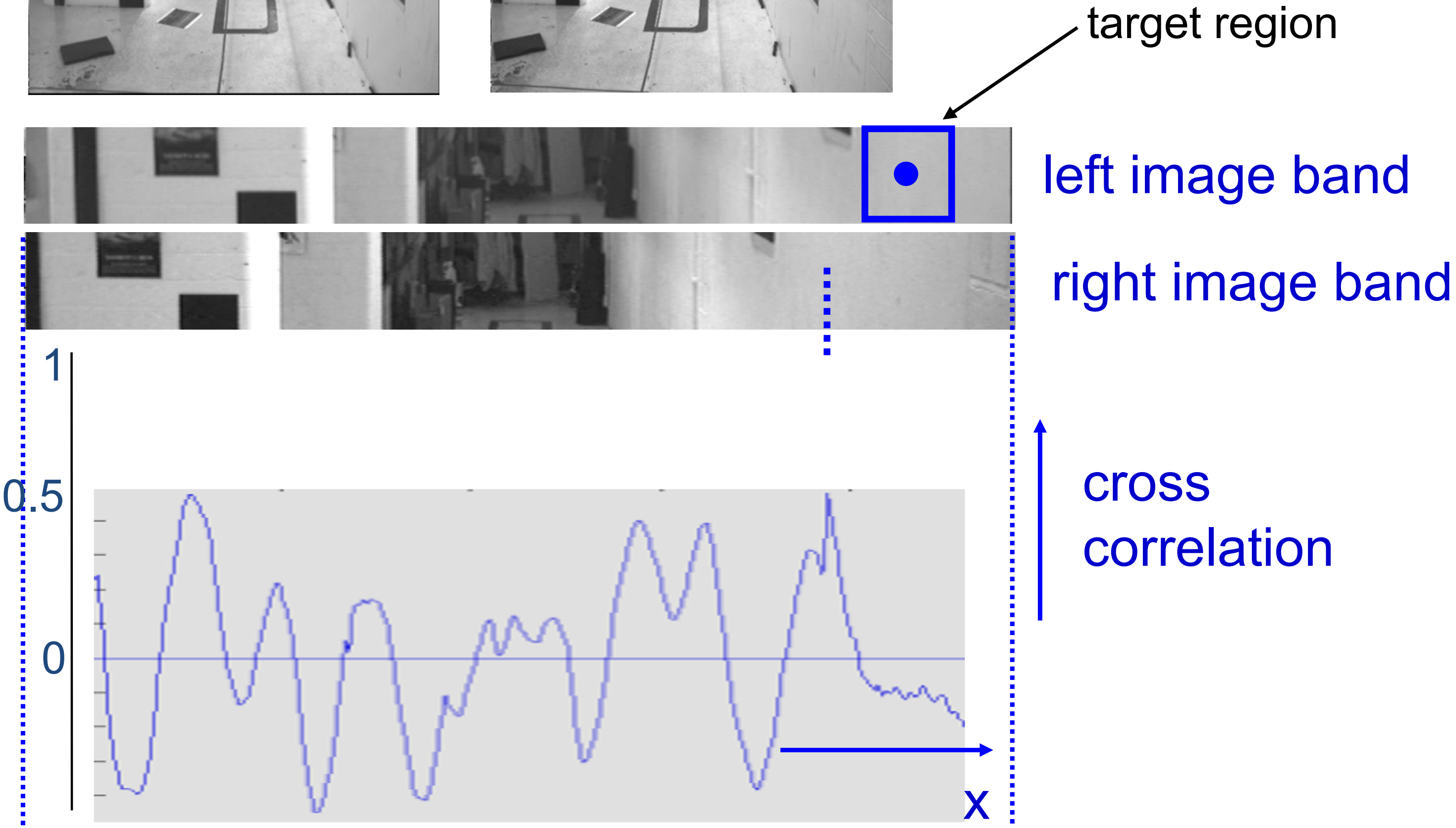
Stereo with NCC: The Good Case





Stereo with NCC: The Bad Case

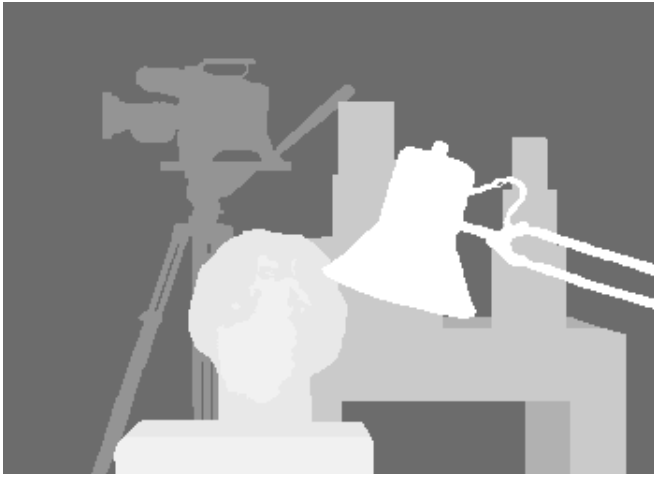




Stereo results

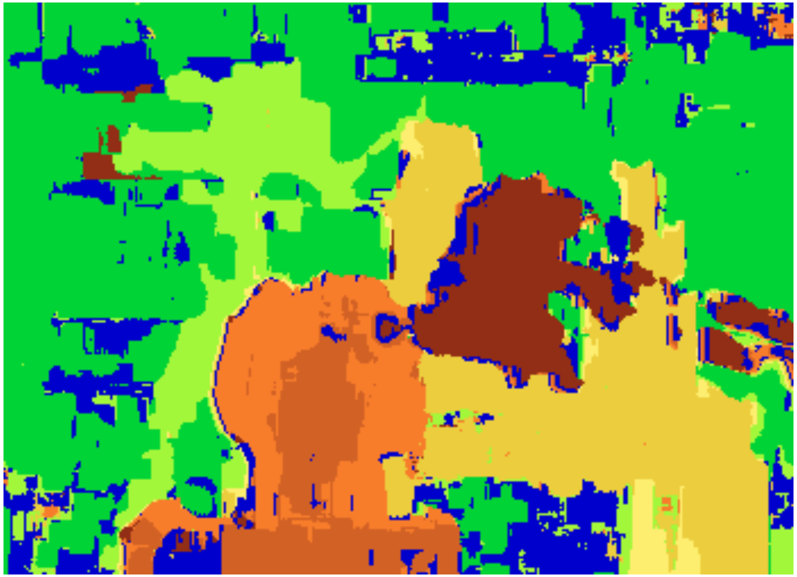
- Data from University of Tsukuba
- Similar results on other images without ground truth

Scene



Ground truth

Results with window search



Window-based matching
(best window size)

Ground truth

Better methods exist...

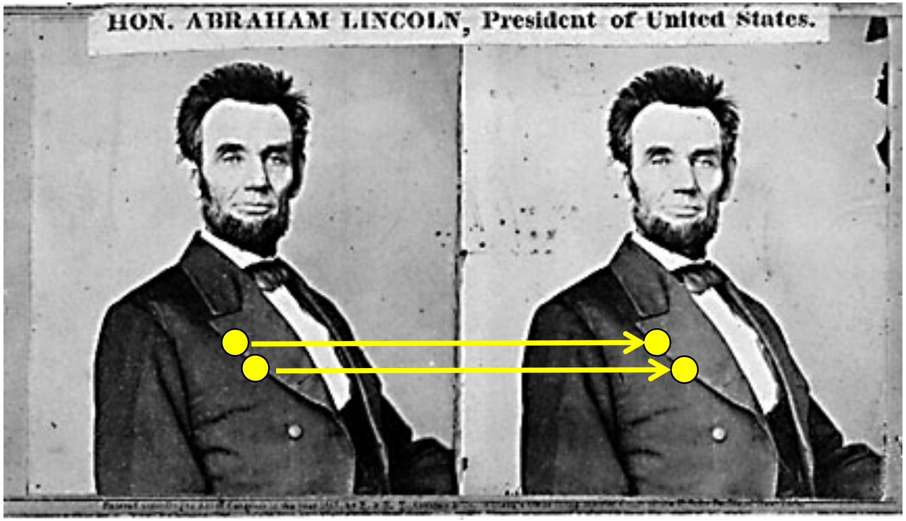
Fancier method

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

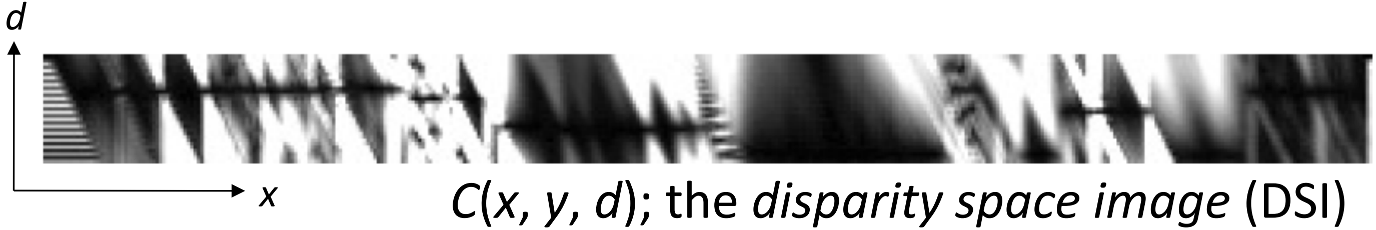


- What defines a good stereo correspondence?
  1. Match quality
     - Want each pixel to find a good match in the other image
  2. Smoothness
     - If two pixels are adjacent, they should (usually) move about the same amount

Stereo as energy minimization

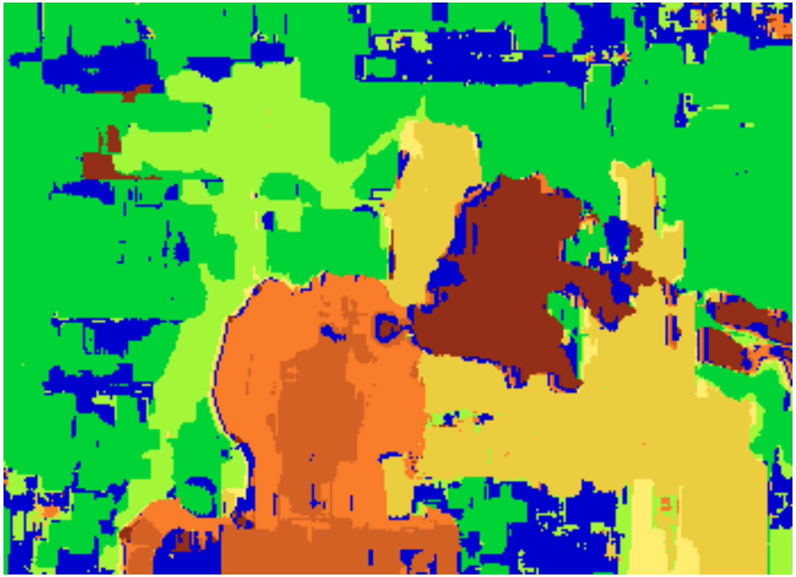
$I(x, y)$

$J(x, y)$



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

Greedy selection of best match



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

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 real-time stereo techniques have been developed (most based on simple discrete search)

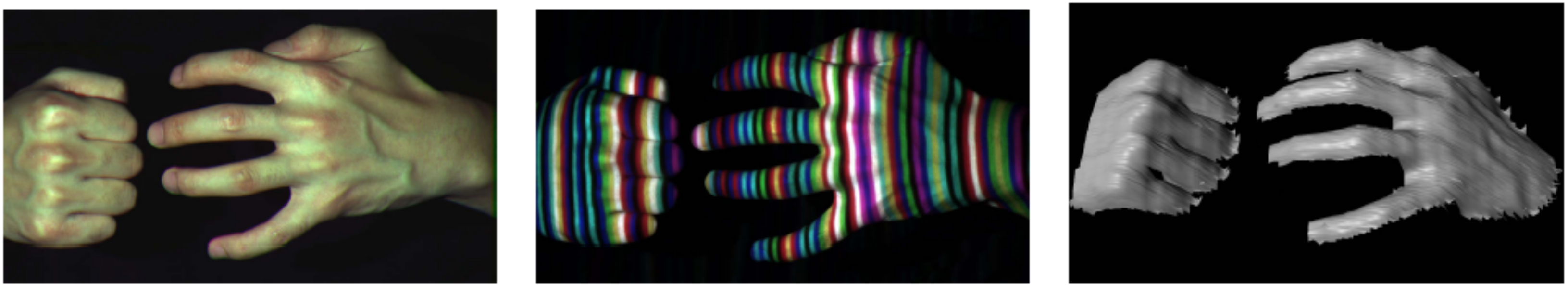
Stereo reconstruction pipeline

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

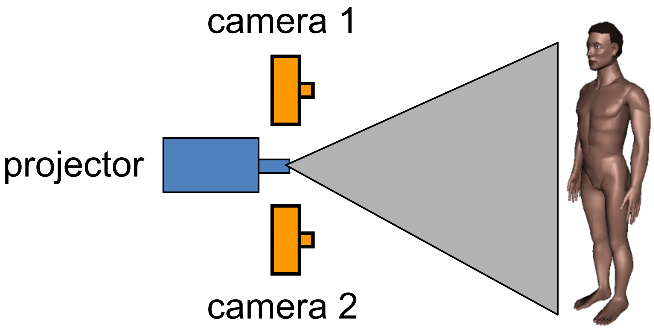
What will cause errors?

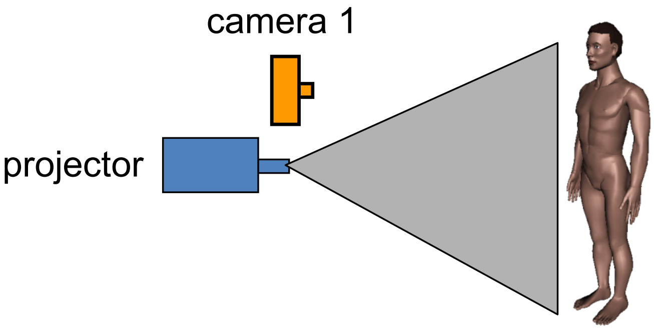
- Camera calibration errors
- Poor image resolution
- Occlusions
- Violations of brightness constancy (specular reflections)
- Large motions
- **Low-contrast image regions**

Active stereo with structured light



Li Zhang's one-shot stereo



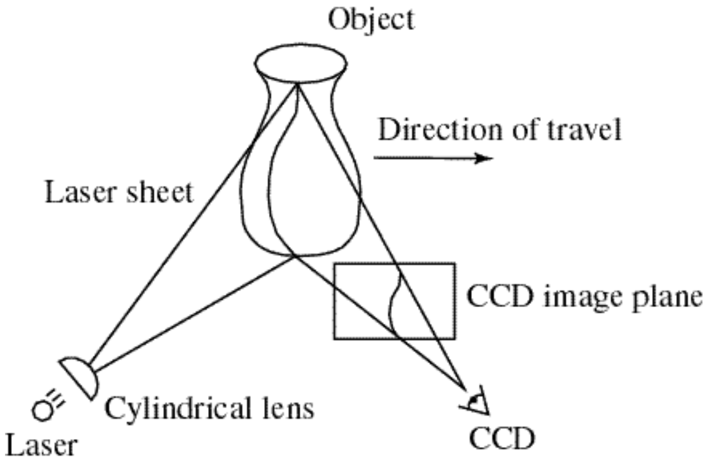


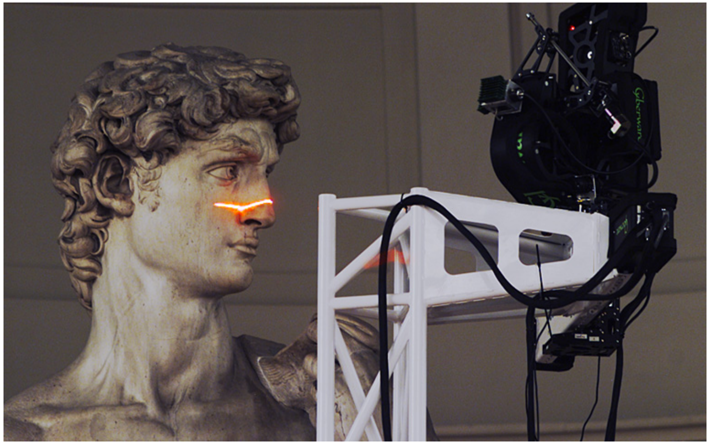
- Project "structured" light patterns onto the object
  - simplifies the correspondence problem
  - basis for active depth sensors, such as Kinect and iPhone X (using IR)

Active stereo with structured light

https://ios.gadgethacks.com/news/watch-iphone-xs-30k-ir-dots-scan-your-face-0180944/

Laser scanning





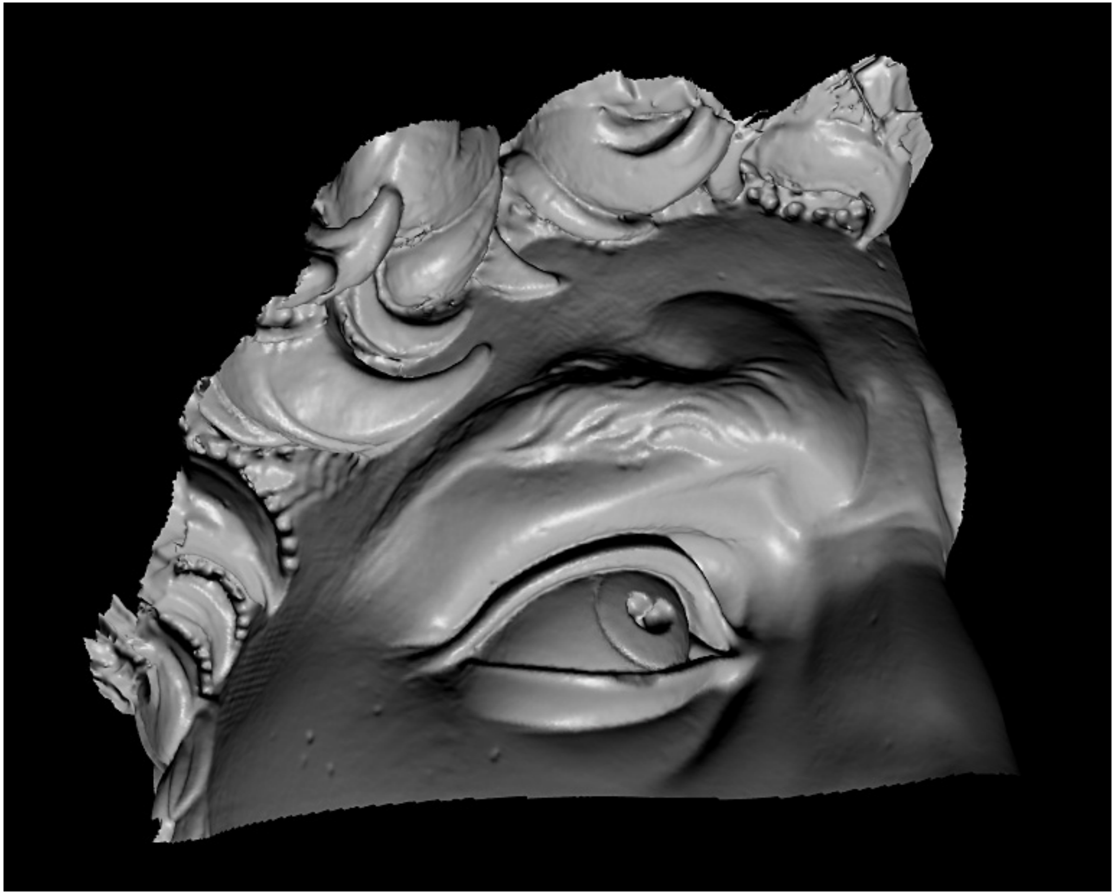
Digital Michelangelo Project
http://graphics.stanford.edu/projects/mich/

- 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

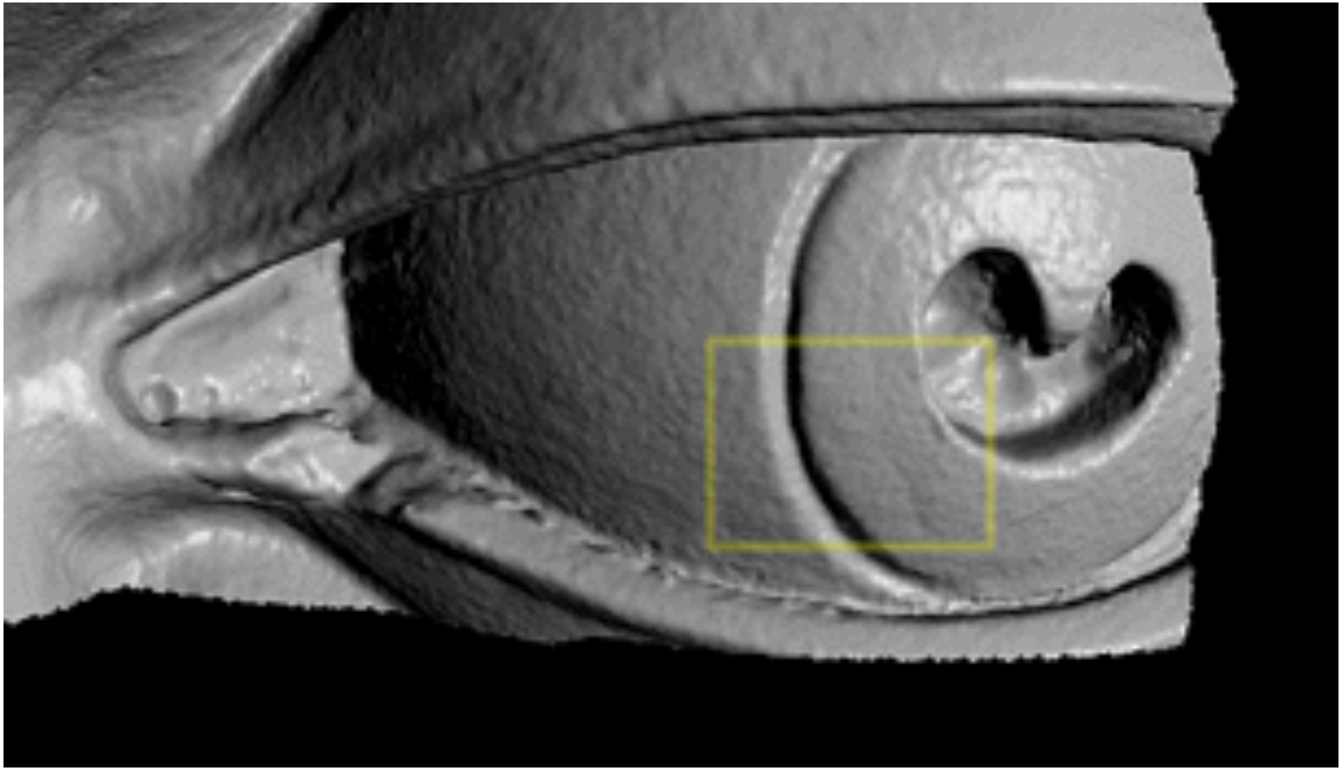
The Digital Michelangelo Project, Levoy et al.

Laser scanned models



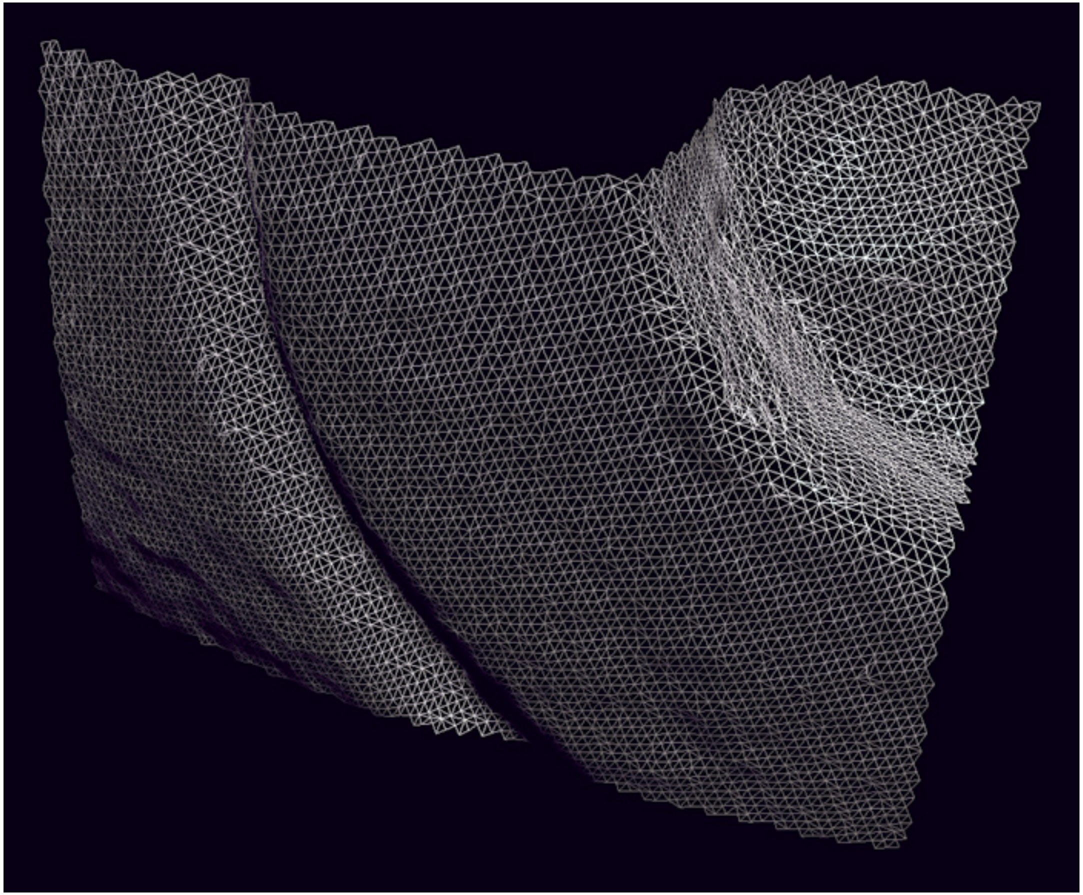
The Digital Michelangelo Project, Levoy et al.

Laser scanned models



The Digital Michelangelo Project, Levoy et al.

Laser scanned models



The Digital Michelangelo Project, Levoy et al.

Microsoft Kinect