What is this?

Single image stereogram,
https://en.wikipedia.org/wiki/Autostereogram
Announcements

• Project 3 due this Friday, March 17 at 8pm (code), Monday, March 20 at 8pm (artifact)

• Project 4 (Stereo) to be released on Tuesday, March 21, due Friday, April 31, by 8pm
  – To be done in groups of two
From last time: 3D modeling from a photograph

video by Antonio Criminisi
3D modeling from a photograph

*Flagellation.* Piero della Francesca. c1453.
Related problem: camera calibration

• Goal: estimate the camera parameters
  – Version 1: solve for 3x4 projection matrix

\[ \mathbf{X} = \begin{bmatrix} wx \\ wy \\ w \end{bmatrix} = \begin{bmatrix} * & * & * \\ * & * & * \\ * & * & * \end{bmatrix} \mathbf{X}^T = \Pi \mathbf{X} \]

  – Version 2: solve for camera parameters separately
    • intrinsics (focal length, principal point, pixel size)
    • extrinsics (rotation angles, translation)
    • radial distortion
Vanishing points and projection matrix

\[ \Pi = \begin{bmatrix} \pi_1 & \pi_2 & \pi_3 & \pi_4 \end{bmatrix} \]

- \( \pi_1 = \Pi \begin{bmatrix} 1 & 0 & 0 & 0 \end{bmatrix}^T = \mathbf{v}_x \) (X vanishing point)
- similarly, \( \pi_2 = \mathbf{v}_y, \pi_3 = \mathbf{v}_z \)
- \( \pi_4 = \Pi \begin{bmatrix} 0 & 0 & 0 & 1 \end{bmatrix}^T = \) projection of world origin

\[ \Pi = \begin{bmatrix} \mathbf{v}_X & \mathbf{v}_Y & \mathbf{v}_Z & \mathbf{0} \end{bmatrix} \]

Not So Fast! We only know \( \mathbf{v}'s \) up to a scale factor

\[ \Pi = \begin{bmatrix} a \mathbf{v}_X & b \mathbf{v}_Y & c \mathbf{v}_Z & \mathbf{0} \end{bmatrix} \]

- Can fully specify by providing 3 reference points with known coordinates
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
AR codes
Estimating the projection matrix

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

\[
\begin{bmatrix}
  u_i \\
  v_i \\
  1
\end{bmatrix} \mapsto \begin{bmatrix}
  m_{00} & m_{01} & m_{02} & m_{03} \\
  m_{10} & m_{11} & m_{12} & m_{13} \\
  m_{20} & m_{21} & m_{22} & m_{23}
\end{bmatrix} \begin{bmatrix}
  X_i \\
  Y_i \\
  Z_i \\
  1
\end{bmatrix}
\]
Alternative: multi-plane calibration

Advantage

- Only requires a plane
- Don’t have to know positions/orientations
- Good code available online! (including in OpenCV)
  - Amy Tabb’s camera calibration software: [https://github.com/amy-tabb/basic-camera-calibration](https://github.com/amy-tabb/basic-camera-calibration)

Images courtesy Jean-Yves Bouguet
Single-image depth prediction using deep learning

MiDaS depth prediction


https://gradio.app/g/AK391/MiDaS

https://github.com/intel-isl/MiDaS
Single-image depth prediction


Boosting Monocular Depth Estimation Models to High-Resolution via Content-Adaptive Multi-Resolution Merging.

CVPR 2021.

Picture credit: Magritte, The Treachery of Images, and the Berkeley Computer Vision Group
Deep geometry prediction

• More on this topic later!
Questions?
“Mark Twain at Pool Table”, no date, UCR Museum of Photography
Stereo Vision as Localizing Points in 3D

- An object point will project to some point in our image
- That image point corresponds to a ray in the world
- Two rays intersect at a single point, so if we want to localize points in 3D we need 2 eyes
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
Epipolar geometry

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

\[ x_2 - x_1 = \text{the disparity of pixel } (x_1, y_1) \]
Disparity = inverse depth

http://stereo.nypl.org/view/41729

(Or, hold a finger in front of your face and wink each eye in succession.)
Your basic stereo matching algorithm

• **Match Pixels in Conjugate Epipolar Lines**
  – Assume brightness constancy
  – This is a challenging problem
  – Hundreds of approaches
    • A good survey and evaluation: [http://www.middlebury.edu/stereo/](http://www.middlebury.edu/stereo/)
Your basic stereo matching 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 matching based on SSD

Best matching disparity
Window size

Effect of window size

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

Better results with *adaptive 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...

Graph cuts-based method


Ground truth

For the latest and greatest: [http://www.middlebury.edu/stereo/](http://www.middlebury.edu/stereo/)
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 / window matching

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

$$C(x, y, d(x, y)) = \text{SSD distance between windows } I(x, y) \text{ and } J(x + d(x,y), y)$$
Stereo as energy minimization

\[ y = 141 \]

The disparity space image (DSI)

\[ C(x, y, d); \text{the disparity space image (DSI)} \]
Stereo as energy minimization

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

\[ d(x, y) = \arg \min_{d'} C(x, y, d') \]
Greedy selection of best match
Stereo as energy minimization

- Better objective function

\[ E(d) = E_d(d) + \lambda E_s(d) \]

- \( E_d(d) \): Match cost
  - Want each pixel to find a good match in the other image

- \( \lambda E_s(d) \): Smoothness cost
  - 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| \quad \text{\( L_1 \) distance}
\]

\[
V(d_p, d_q) = \begin{cases} 
0 & \text{if } d_p = d_q \\
1 & \text{if } d_p \neq d_q 
\end{cases} \quad \text{“Potts model”}
\]
Smoothness cost

\[ E(d) = E_d(d) + \lambda E_s(d) \]

- If \( \lambda = \text{infinity} \), then we only consider smoothness.
- Optimal solution is a surface of constant depth/disparity
  – Fronto-parallel surface
- In practice, want to balance data term with smoothness term.
Dynamic programming

\[ E(d) = E_d(d) + \lambda E_s(d) \]

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

- Finds “smooth”, low-cost path through DPI from left to right
- Visiting a node incurs its data cost, switching disparities from one column to the next also incurs a (smoothness) cost
Dynamic Programming
Dynamic programming

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

• No: the shortest path trick only works to find a 1D path

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
  – \( n \times m \) image w/ \( k \) disparities has \( k^{nm} \) possible solutions
  – Finding the global minimum is NP-hard in general

• Good approximations exist (e.g., graph cuts algorithms)
Questions?
Depth from disparity

\[
disparity = x - x' = \frac{\text{baseline} \times f}{z}
\]
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
Variants of stereo
Real-time stereo

Nomad robot searches for meteorites in Antartica

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

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

Li Zhang’s one-shot stereo
Active stereo with structured light

Laser scanning

• 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

Digital Michelangelo Project
http://graphics.stanford.edu/projects/mich/
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.
3D Photography on your Desk

The idea

[Image: Diagram showing a camera, desk, lamp, stick or pencil, and a 3D rendering of a figure]

http://www.vision.caltech.edu/bouguetj/ICCV98/
Questions?