Correspondence and Stereopsis

Original notes by W. Correa. Figures from [Forsyth & Ponce] and [Trucco & Verri]
Introduction

• Disparity:
  – Informally: difference between two pictures
  – Allows us to gain a strong sense of depth

• Stereopsis:
  – Ability to perceive depth from disparity

• Goal:
  – Design algorithms that mimic stereopsis
Stereo Vision

• Two parts
  – Binocular fusion of features observed by the eyes
  – Reconstruction of their three-dimensional preimage
Stereo Vision – Easy Case

• A single point being observed
  – The preimage can be found at the intersection of the rays from the focal points to the image points
Stereo Vision – Hard Case

- Many points being observed
  - Need some method to establish correspondences
Components of Stereo Vision Systems

- Camera calibration: previous lecture
- Image rectification: simplifies the search for correspondences
- Correspondence: which item in the left image corresponds to which item in the right image
- Reconstruction: recovers 3-D information from the 2-D correspondences
Epipolar Geometry

- Epipolar constraint: corresponding points must lie on conjugate epipolar lines
  - Search for correspondences becomes a 1-D problem
Image Rectification

- Warp images such that conjugate epipolar lines become collinear and parallel to $u$ axis
Image Rectification (cont.)

- Perform by rotating the cameras
- *Not* equivalent to rotating the images
- The lines through the centers become parallel to each other, and the epipoles move to infinity
Image Rectification (cont.)

- Given extrinsic parameters T and R (relative position and orientation of the two cameras)
  - Rotate the left camera about the projection center so that the epipolar lines become parallel to the horizontal axis
  - Apply the same rotation to the right camera
  - Rotate the right camera by R
  - Adjust the scale in both camera reference frames
Disparity

• With rectified images, disparity is just (horizontal) displacement of corresponding features in the two images
  – Disparity = 0 for distant points
  – Larger disparity for closer points
  – Depth of point proportional to 1/disparity
Correspondence

• Given an element in the left image, find the corresponding element in the right image

• Classes of methods
  – Correlation-based
  – Feature-based
Correlation-Based Correspondence

• Input: rectified stereo pair and a point \((u,v)\) in the first image

• Method:
  – Form window of size \((2m+1) \times (2n+1)\) centered at \((u,v)\) and assemble points into the vector \(w\)
  – For each potential match \((u+d,v)\) in the second image, compute \(w'\) and the normalized correlation between \(w\) and \(w'\)
Sum of Squared Differences

• Recall: SSD for image similarity

\[ \psi(u, v) = -(u - v)^2 \]

• Negative sign so that higher values mean greater similarity
Normalized Cross-Correlation

- Normalize to eliminate brightness sensitivity:

\[ \psi(u, v) = \frac{(u - \bar{u})(v - \bar{v})}{\sigma_u \sigma_v} \]

where

\[ \bar{u} = \text{average}(u) \]

\[ \sigma_u = \text{standard deviation}(u) \]

- Can help for non-diffuse scenes, hurts for perfectly diffuse ones
Window-Based Correlation

• For each pixel
  – For each disparity
    • For each pixel in window
      – Compute difference
    – Find disparity with minimum SSD
Reverse Order of Loops

• For each disparity
  – For each pixel
    • For each pixel in window
      – Compute difference

• Find disparity with minimum SSD at each pixel
Incremental Computation

- Given SSD of a window, at some disparity
Incremental Computation

- Want: SSD at next location
Incremental Computation

- Subtract contributions from leftmost column, add contributions from rightmost column
Selecting Window Size

- Small window: more detail, but more noise
- Large window: more robustness, less detail
- Example:
Selecting Window Size

3 pixel window

20 pixel window
Non-Square Windows

• Compromise: have a large window, but higher weight near the center
• Example: Gaussian
• For each disparity
  – For each pixel
    • Compute weighted SSD
Diffusion

- For each disparity
  - For each pixel
    - Compute squared difference in intensities
  - For n iterations:
    - $E_i \leftarrow (1-4\lambda) E_i + \lambda \sum E_j$
- Sum is over four neighbors of each pixel
Non-Linear Diffusion

• To prevent blurring even more, only perform diffusion in ambiguous regions

• For each pixel, compute certainty
  – High certainty iff one disparity has low error, all others have high error

• For each pixel, only perform diffusion if certainty goes up
Certainty Metrics for Non-Linear Diffusion

• Winner margin: normalized difference between lowest and second-lowest error

\[ C(i, j) = \frac{E_{\text{min}} - E_{\text{min}}}{\sum_d E_d} \]

• Entropy:

\[ C(i, j) = -\sum_d p(d) \log p(d), \quad p(d) = \frac{e^{-E_d}}{\sum_d e^{-E_d}} \]
Results

• Scharstein and Szeliski, 1996

3 pixel window  20 pixel window  Nonlinear diffusion
Problems with Correlation-Based Correspondence

• Main problem:
  – Assumes that the observed surface is locally parallel to the two image planes
  – If not, unequal amounts of foreshortening in images
  – Iterate: compute disparities, warp images, repeat

• Other problems:
  – Not robust against noise
  – Similar pixels may not correspond to physical features
Feature-Based Correspondence

• Main idea: significant features should be preferred to matches between raw pixel intensities
• Instead of correlation-like measures, use similarity between feature descriptors
• Typical features: points, lines, and corners
• Example: Marr-Poggio-Grimson algorithm
Marr-Poggio-Grimson Algorithm

- Convolve images with Laplacian of Gaussian filters with decreasing widths
- Find zero crossings of the Laplacian along horizontal scanlines of the filtered images
- For each $\sigma$, match zero crossings with same parity and similar orientations in a $[-w_{\sigma}, w_{\sigma}]$ disparity range, with

$$w_{\sigma} = 2\sqrt{2\sigma}$$
Marr-Poggio-Grimson Algorithm (cont.)

• Use disparities found at larger scales to control eye vergence and cause unmatched regions at smaller scales to come into correspondence
Marr-Poggio-Grimson Algorithm (cont.)

Matching zero-crossings at a single scale

Matching zero-crossings at multiple scales

Scale  \( \varnothing \)  \( \varnothing \)  \( \varnothing \)  \( \varnothing \)  \( \varnothing \)  \( \varnothing \)
Width  \( \varnothing \)  \( \varnothing \)  \( \varnothing \)  \( \varnothing \)  \( \varnothing \)  \( \varnothing \)

Match  \( \varnothing \)  \( \varnothing \)  \( \varnothing \)  \( \varnothing \)  \( \varnothing \)  \( \varnothing \)

Offset  \( \varnothing \)  \( \varnothing \)  \( \varnothing \)  \( \varnothing \)  \( \varnothing \)  \( \varnothing \)

Rematch  \( \varnothing \)  \( \varnothing \)  \( \varnothing \)  \( \varnothing \)  \( \varnothing \)  \( \varnothing \)
Ordering Constraint

• Order of matching features usually the same in both images
• But not always: occlusion
Graph Search

- Treat feature correspondence as graph problem

Cost of edges = similarity of regions between image features
Graph Search

- Find min-cost path through graph
Reconstruction

• Given pair of image points $p$ and $p'$, and focal points $O$ and $O'$, find preimage $P$

• In theory: find $P$ by intersecting the rays $R = Op$ and $R' = Op'$

• In practice: $R$ and $R'$ won't actually intersect due to calibration and feature localization errors
Reconstruction Approaches

• Geometric
  – Construct the line segment perpendicular to R and R' that intersects both rays and take its mid-point
Reconstruction Approaches

• Image-space: find the point $P$ whose projection onto the images minimizes distance to desired correspondences

• Nonlinear optimization