Computer Vision

Howie Choset
http://www.cs.cmu.edu/~choset

Introduction to Robotics
http://generalrobotics.org
How can we use Computer Vision?

- Understand the world from images!

**WHO**
Facial recognition, object classification,

**WHAT**
action recognition

**WHEN**
Object tracking, image labeling, scene reconstruction

**WHERE**
Scene understanding, image alteration

**WHY**
Crowd Tracking

Image Dehazing

Scene Reconstruction

Scene Understanding

Massive Data

Scene Alteration

Goal of Vision: Recover Projection

\( \Pi: \mathbb{R}^3 \rightarrow \mathbb{R}^2 \) or \( \Pi: \mathbb{R}^3 \rightarrow \mathbb{Z}^2 \)

Recover third dimension or just infer stuff
Projection on the image plane

- Size of an image on the image plane is inversely proportional to the distance from the focal point

\[ \frac{h}{d} = \frac{-h'}{f} \]

By conceptually moving the image plane, we can eliminate the negative sign

\[ \frac{h}{d} = \frac{h'}{f} \]
Grayscale vs. Binary image

<table>
<thead>
<tr>
<th>Grayscale</th>
<th>Binary threshold</th>
</tr>
</thead>
<tbody>
<tr>
<td><img src="image1.png" alt="Grayscale Image" /></td>
<td><img src="image2.png" alt="Binary Threshold" /></td>
</tr>
</tbody>
</table>
Mona’s Histogram
(a) An image with 256 gray levels. (b) Thresholded version of the image in (a). (c) Histogram for the image shown in 11.2a (d) Within-group variance for the image shown in 11.2a
Connectivity

- Two conventions on considering two pixels next to each other

  8 point connectivity
  All pixels sharing a side or corner are considered adjacent

  4 point connectivity
  Only pixels sharing a side are considered adjacent

- To eliminate the ambiguity, we could define the shape of a pixel to be a hexagon
Segmentation: Wavefront

Assume a binary image with values of 0 or 1

1. Choose 1\textsuperscript{st} pixel with value 1, make it a 2
2. For each neighbor, if it is also a 1, make it a 2 as well
3. Repeat step two for each neighbor until there are no neighbors with value 1
4. All pixels with a value 2 are are a continuous object

Or look for pixels below a threshold (instead of a 0) and otherwise above (instead of 1)
Input and Output

Connected Components

Labeled Connected Components
Segmentation: Double Raster

Assume a binary image with values of 0 or 1
Initialize cntr to 0

• 1. Perform a raster scan – across and down
  – a. Encounter a pixel with a 1
  – b. Look up, look left
    • If both 0
      – Increment cntr by 1
      – assign pixel P a value cnt
    • If either is 1, assign P the label of the 1
    • If both are 1
      – Note equivalence
      – Assign P’s label as minimum of 2

• 2. Perform second raster scan to align equivalences

Or look for pixels below a threshold (instead of a 0) and otherwise above (instead of 1)
Double Raster Example

Copied from Robot Motion and Control
By Spong, Hutchinson, Vidyasagar
Centroids

- Use the region filled image from above
- Compute the area of the region
  - Number of pixels with the same number value (n)
- Sum all of the x coords with the same pixel value. Do the same for y coords
- Divide each sum by n and the resulting x, y coord is the centroid
Stereo Vision

• Way of calculating depth from two dimensional images using two cameras

\[
\alpha \quad \beta \\
\begin{align*}
\tan \alpha &= \frac{d}{c - y} \\
\tan \beta &= \frac{d}{y} \\
y &= \frac{c \tan \alpha}{\tan \alpha - \tan \beta} \\
d &= y \tan \beta
\end{align*}
\]
Edge detection

- Scanline: one row of pixels in an image
- Compute $B[m+1] - B[m]$
Edge detection

• Scanline: one row of pixels in an image
• Take the first derivative of a scanline

• The derivative becomes nonzero when an edge (pixels change values) is encountered
Implementing $1^{st}$ derivative edge detection digitally

- Derivative is defined as $\lim_{x \to c} \frac{f(x) - f(c)}{x - c}$
Implementing 1st derivative edge detection digitally

• Derivative is defined as \[ \lim_{x \to c} \frac{f(x) - f(c)}{x - c} \]

• With a scan line, the run \((x - c)\) is 1, and the rise \((f(x) - f(c))\) is \(B[m+1] - B[m]\)
Implementing 1\textsuperscript{st} derivative edge detection digitally

- Derivative is defined as \( \lim_{x \to c} \frac{f(x) - f(c)}{x - c} \)
- With a scan line, the run \((x - c)\) is 1, and the rise \((f(x) - f(c))\) is \(B[m+1] - B[m]\)
- This becomes \( I[m] = 1 \cdot B[m+1] + -1B[m] \)
Implementing 1\textsuperscript{st} derivative edge detection digitally

- Derivative is defined as
  \[ \lim_{x \to c} \frac{f(x) - f(c)}{x - c} \]
- With a scan line, the run \((x - c)\) is 1, and the rise \((f(x) - f(c))\) is \(B[m+1] - B[m]\)
- This becomes
  \[ I[m] = 1 \cdot B[m+1] + -1B[m] \]
- This is really just a dot product of the vector \([-1 \ 1]\) repeated each pixel in the resulting image

\[ I[m] = [B[m] \ B[m+1]] \cdot [-1 \ 1] \]
Implementing 1\textsuperscript{st} derivative edge detection digitally

- Derivative is defined as
  \[ \lim_{x \to c} \frac{f(x) - f(c)}{x - c} \]
- With a scan line, the run \((x - c)\) is 1, and the rise \((f(x) - f(c))\) is \(B[m+1] - B[m]\)
- This becomes
  \[ I[m] = 1 \cdot B[m+1] + -1B[m] \]
- This is really just a dot product of the vector \([-1\ 1]\) repeated each pixel in the resulting image

\[
I[m] = [B[m] \quad B[m+1]] \bullet [-1\ 1]
\]
Convolution

• This operation of moving a mask across an image has a name, called convolution

• In order to mathematically apply a filter to a signal, we must use convolution
  – If you know Laplace transforms, this is a multiplication in the Laplace domain
Convolution: Analog

\[ y(t) = \int_{-\infty}^{\infty} x(\tau) h(t - \tau) d\tau \]
Convolution: Analog

\[ y(t) = \int_{-\infty}^{\infty} x(\tau)h(t - \tau)d\tau \]

Given a symmetric h (common in image processing), simplifies to

\[ y(t) = \int_{-\infty}^{\infty} x(\tau)h(\tau)d\tau \]
Convolution: Analog

\[ y(t) = \int_{-\infty}^{\infty} x(\tau)h(t - \tau)d\tau \]

Given a symmetric \( h \) (common in image processing), simplifies to

\[ y(t) = \int_{-\infty}^{\infty} x(\tau)h(\tau)d\tau \]

\[
\begin{bmatrix}
-1 & 1
\end{bmatrix}
\]

Move across the signal \( x \) (possibly a scanline in an image)
Convolution: Digital

\[ y[n] = \sum_{k=-\infty}^{\infty} x[n+k]h[k] \]

\[ h(t) = [-1 1] \]

Move across the signal \( x \) (possibly a scanline in an image)
Convolution: Digital

\[ y[n] = \sum_{k=-\infty}^{\infty} x[n+k]h[k] \]

More useful in image processing on a digital computer, \( x[n] \) is a pixel in an image, \( y[n] \) is the resulting pixel

1. \[
\begin{array}{cccccc}
0 & 1 & 1 & 4 & 4 & 2 \\
0 & 2 & 2 & 0 & 1 & 1 & 3
\end{array}
\]

2. \[
\begin{array}{cccccc}
0 & 1 & 1 & 4 & 4 & 2 \\
0 & 2 & 2 & 0 & 1 & 1 & 3
\end{array}
\]
### Convolution example, cont

#### 3.
```
<table>
<thead>
<tr>
<th>0</th>
<th>1</th>
<th>1</th>
</tr>
</thead>
<tbody>
<tr>
<td>0</td>
<td>2</td>
<td>2</td>
</tr>
</tbody>
</table>
```
```
| 4 | 2 | 1 |
```

#### 4.
```
<table>
<thead>
<tr>
<th>0</th>
<th>1</th>
<th>1</th>
</tr>
</thead>
<tbody>
<tr>
<td>0</td>
<td>2</td>
<td>2</td>
</tr>
</tbody>
</table>
```
```
| 4 | 2 | 1 | 2 |
```

#### 5.
```
<table>
<thead>
<tr>
<th>0</th>
<th>1</th>
<th>1</th>
</tr>
</thead>
<tbody>
<tr>
<td>0</td>
<td>2</td>
<td>2</td>
</tr>
</tbody>
</table>
```
```
| 4 | 2 | 1 | 2 | 4 |
```
Second Derivative

\[
\begin{bmatrix}
0 & -1 & 1
\end{bmatrix}
\begin{bmatrix}
0 & -1 & 1
\end{bmatrix}
= \begin{bmatrix}
-1 & 2 & -1
\end{bmatrix}
\]
Convolution: Two-dimensional

\[ y(m, n) = \sum_{m_0} \sum_{n_0} x(m_0 + m, n_0 + n) h(m_0, n_0) \]

- Rotate your mask 180 degrees about the origin (if you were doing “correct” convolution, but since we are doing the “other” convolution, you can skip this step.
- Do the same dot product operation, this time using matrices instead of vectors.
- Repeat the dot product for every pixel in the resulting image.
- In the boundary case around the edges of the image there are two options:
  - extend the original image out using the pixel values at the edge
  - Make the resulting image \( y \) smaller than the original and don’t compute pixels where the mask would extend beyond the edge of the original.
Convolution: Old and New

- Analog
  \[ y(t) = \int_{-\infty}^{\infty} x(\tau)h(t-\tau)d\tau \]

- Digital
  \[ y[n] = \sum_{k=-\infty}^{\infty} x[k]h[n-k] \]

- Two dimensional, digital
  \[ y(m,n) = \sum_{m_0} \sum_{n_0} x(m_0,n_0)h(m-m_0,n-n_0) \]
  \[ y(m,n) = \sum_{m_0} \sum_{n_0} x(m+m_0,n+n_0)h(m_0,n_0) \]
Filters, Masks, Transforms

• Smoothing / Averaging / Blurring

• Edge detection
  – Wide masks

• Object detection
Gaussian Masks

- Used to smooth images and for noise reduction
- Use before edge detection to avoid spurious edges

Johann Carl Friedrich Gauss

April 30, 1777 – Feb 23, 1855
number theory, statistics, analysis, differential geometry, geodesy, electrostatics, astronomy, and optics.

17 heptadecagon
Wide Masks

- Wider masks lead to uncertainty about location of edge
- Can detect more gradual edges

Convolution with Gaussian distribution
Wide Masks

- Wider masks lead to uncertainty about location of edge
- Can detect more gradual edges

Convolution with Gaussian distribution

Edge detection on a signal that was convolved with a Gaussian
Wide Masks
Cat Video
Interest Points

• How would we find “interesting” parts of the image?
• Let’s say corners are “interesting”
• Convolve with Gaussians, take difference:

\[ G_1 \quad \sigma = 1.2 \]

\[ G_2 \quad \sigma = 1.0 \]

\[ G_1 - G_2 \]

Difference of Gaussians (DoG)
Difference of Gaussians

Original Howie

Howie with less blur

Howie with more blur

Difference of Howies
Scale Stability

- Interest points should be stable to scaling
- Can calculate DoG at many scales, find strong edges in space and scale

Poorly localized

Well localized
Scale Space Example
Keypoint Example

Original Keypoints

Scaled 110%
Keypoint Orientation

• Local gradient magnitude, orientation calculated
• Keypoint scale affects blur, sampling used

\[ m(x, y) = \sqrt{(L(x+1, y) - L(x-1, y))^2 + (L(x, y+1) - L(x, y-1))^2} \]

\[ \theta(x, y) = \tan^{-1}((L(x, y+1) - L(x, y-1))/(L(x+1, y) - L(x-1, y))) \]

• Magnitude and Gaussian weighted orientations binned, max taken as keypoint orientation

Figure from David G. Lowe
Local Descriptor

• Surrounding gradients summarized by smaller regional histograms
• Descriptor normalized to account for illumination

Figure from David G. Lowe
Rotational Invariance

- Descriptor orientations calculated relative to keypoint orientation
Histogram of Oriented Gradients (HOG)

- What if we calculated descriptors at every pixel?
  - "Dense features"
- Can be used as different "image" with many channels