Previously

- Filters allow local image neighborhood to influence our description and features
  - Smoothing to reduce noise
  - Derivatives to locate contrast, gradient
- Seam carving application:
  - Use image gradients to measure “interestingness” or “energy”
  - Remove 8-connected seams so as to preserve image’s energy

Review: Partial derivatives of an image

\[
\begin{align*}
\frac{\partial f(x, y)}{\partial x} & \quad \text{and} \quad \frac{\partial f(x, y)}{\partial y} \\
\begin{bmatrix}
-1 & 1 \\
-1 & 1
\end{bmatrix} & \quad \text{or} \quad \begin{bmatrix}
1 & -1 \\
1 & -1
\end{bmatrix}
\end{align*}
\]

Which shows changes with respect to x?
Today

- Edge detection and matching
  - process the image gradient to find curves/contours
  - comparing contours

- Binary image analysis
  - blobs and regions
Edge detection

- **Goal**: map image from 2d array of pixels to a set of curves or line segments or contours.
- **Why?**

  - **Main idea**: look for strong gradients, post-process

Gradients -> edges

Primary edge detection steps:
1. Smoothing: suppress noise
2. Edge enhancement: filter for contrast
3. Edge localization
   - Determine which local maxima from filter output are actually edges vs. noise
     - Threshold, Thin

Thresholding

- Choose a threshold value \( t \)
- Set any pixels less than \( t \) to zero (off)
- Set any pixels greater than or equal to \( t \) to one (on)
Thresholding gradient with a higher threshold

Canny edge detector

- Filter image with derivative of Gaussian
- Find magnitude and orientation of gradient
- **Non-maximum suppression:**
  - Thin wide “ridges” down to single pixel width
- **Linking and thresholding (hysteresis):**
  - Define two thresholds: low and high
  - Use the high threshold to start edge curves and the low threshold to continue them

- MATLAB: `edge(image, 'canny');`
- `>>help edge`

The Canny edge detector

original image (Lena)
The Canny edge detector

Compute Gradients

The Canny edge detector
The Canny edge detector

thresholding

How to turn these thick regions of the gradient into curves?

Non-maximum suppression

Check if pixel is local maximum along gradient direction
Select single max across width of the edge
Requires checking interpolated pixels p and r
The Canny edge detector

Problem: pixels along this edge didn't survive the thresholding.

Hysteresis thresholding

- Use a high threshold to start edge curves, and a low threshold to continue them.

Hysteresis thresholding

original image

high threshold (strong edges)
low threshold (weak edges)
hysteresis threshold
Hysteresis thresholding

http://users.ecs.soton.ac.uk/msn/book/new_demo/thresholding/

Recap: Canny edge detector

- Filter image with derivative of Gaussian
- Find magnitude and orientation of gradient
- **Non-maximum suppression:**
  - Thin wide “ridges” down to single pixel width
- **Linking and thresholding (hysteresis):**
  - Define two thresholds: low and high
  - Use the high threshold to start edge curves and the low threshold to continue them

- MATLAB: `edge(image, 'canny');`

- `>>help edge`

Low-level edges vs. perceived contours
Low-level edges vs. perceived contours

- Berkeley segmentation database:
  [link]

Learn from humans which combination of features is most indicative of a 'good' contour?

[D. Martin et al. PAMI 2004]

Human-marked segment boundaries

pB boundary detector

Martin, Fowlkes, Malik 2004: Learning to Detection Natural Boundaries...
[link]
pB Boundary Detector

Figure from Fowlkes.

State-of-the-Art in Contour Detection

Source: Jitendra Malik; http://www.cs.berkeley.edu/~malik/malik-talks-ptrs.html
Holistically-Nested Edge Detection (Xie, Tu ICCV 2015)

1. holistic image training and prediction
2. multi-scale and multi-level feature learning
3. Deeply-supervised fully-convolutional network

State-of-the-Art in Contour Detection

Today

- Edge detection and matching
  - process the image gradient to find curves/contours
  - comparing contours
- Binary image analysis
  - blobs and regions
Chamfer distance

- Average distance to nearest feature/edge

\[ D_{\text{cham}}(I, I) = \frac{1}{|T|} \sum_{t \in T} d_t(t) \]

- \( I \) = Set of edge points in image
- \( T \) = Set of edge points on (shifted) template
- \( d_t(t) \) = Minimum distance between point \( t \) and some point in \( I \)
Chamfer distance

- Average distance to nearest feature

\[ D_{\text{chamfer}}(T, I) \equiv \frac{1}{|T|} \sum_{t \in T} d_T(t) \]

How is the measure different than just filtering with a mask having the shape points?

How expensive is a naïve implementation?

Distance transform

Image features (2D)

Distance Transform

Distance Transform is a function \( D() \) that for each image pixel \( p \) assigns a non-negative number \( D(p) \) corresponding to distance from \( p \) to the nearest feature in the image \( I \)

Distance transform

Value at \((x,y)\) tells how far that position is from the nearest edge point (or other binary image structure)

>> help bwdist
Distance transform (1D)

Two pass $O(n)$ algorithm for 1D $L_1$ norm

1. **Initialize**: For all $j$
   \[ D[j] \leftarrow I_p[j] \]
   \[ \text{// 0 if j is in P, infinity otherwise} \]

Distance Transform (2D)

- 2D case analogous to 1D
  - Initialization
  - Forward and backward pass
    - Fwd pass finds closest above and to left
    - Bwd pass finds closest below and to right

Chamfer distance

- Average distance to nearest feature
  \[ D_{\text{cham}}(T, I) \equiv \frac{1}{|T|} \sum_{t \in T} d_f(t) \]
Chamfer distance

Chamfer distance: properties

- Sensitive to scale and rotation
- Tolerant of small shape changes, clutter
- Need large number of template shapes
- Inexpensive way to match shapes

Today

- Edge detection and matching
  - process the image gradient to find curves/contours
  - comparing contours
- Binary image analysis
  - blobs and regions
Binary images

Binary image analysis:
basic steps
- Convert the image into binary form
  - Thresholding
- Clean up the thresholded image
  - Morphological operators
- Extract separate blobs
  - Connected components
- Describe the blobs with region properties

Binary images
- Two pixel values
  - Foreground and background
  - Mark region(s) of interest
Thresholding

- Grayscale -> binary mask
- Useful if object of interest's intensity distribution is distinct from background

\[
F_{1}[i,j] = \begin{cases} 
1 & \text{if } F[i,j] \geq T \\
0 & \text{otherwise},
\end{cases}
\]

\[
F_{2}[i,j] = \begin{cases} 
1 & \text{if } T_{1} \leq F[i,j] \leq T_{2} \\
0 & \text{otherwise},
\end{cases}
\]

\[
F_{3}[i,j] = \begin{cases} 
1 & \text{if } F[i,j] \in Z \\
0 & \text{otherwise}.
\end{cases}
\]

- Example

---

Thresholding

- Given a grayscale image or an intermediate matrix \(\rightarrow\) threshold to create a binary output.

Example: edge detection

Looking for pixels where gradient is strong.

---

Thresholding

- Given a grayscale image or an intermediate matrix \(\rightarrow\) threshold to create a binary output.

Example: background subtraction

Looking for pixels that differ significantly from the “empty” background.
Thresholding

- Given a grayscale image or an intermediate matrix → threshold to create a binary output.

Example: intensity-based detection

Looking for dark pixels

fg_pix = find(im < 65);

Example: color-based detection

Looking for pixels within a certain hue range.

fg_pix = find(hue > t1 & hue < t2);

A nice case: bimodal intensity histograms

- Ideal histogram, light object on dark background
- Actual observed histogram with noise

Not so nice cases

Two distinct modes

Overlapped modes

Issues

• What to do with “noisy” binary outputs?
  – Holes
  – Extra small fragments

• How to demarcate multiple regions of interest?
  – Count objects
  – Compute further features per object

Morphological operators

• Change the shape of the foreground regions via intersection/union operations between a scanning structuring element and binary image

• Useful to clean up result from thresholding

• Basic operators are:
  – Dilation
  – Erosion
**Dilation**
- Expands connected components
- Grow features
- Fill holes

Before dilation

After dilation

**Erosion**
- Erode connected components
- Shrink features
- Remove bridges, branches, noise

Before erosion

After erosion

**Structuring elements**
- **Masks** of varying shapes and sizes used to perform morphology, for example:

```
+    
```

- Scan mask across foreground pixels to transform the binary image

```
>> help strel
```
Dilation vs. Erosion

At each position:
- **Dilation**: if current pixel is 1, then set all the output pixels corresponding to structuring element to 1.

Example for Dilation

<table>
<thead>
<tr>
<th>Input image</th>
<th>1</th>
<th>0</th>
<th>0</th>
<th>1</th>
<th>1</th>
<th>0</th>
<th>1</th>
<th>1</th>
</tr>
</thead>
<tbody>
<tr>
<td>Structuring Element</td>
<td>1</td>
<td>1</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>Output Image</td>
<td></td>
<td></td>
<td></td>
<td>1</td>
<td>1</td>
<td></td>
<td></td>
<td></td>
</tr>
</tbody>
</table>

Example for Dilation

<table>
<thead>
<tr>
<th>Input image</th>
<th>1</th>
<th>0</th>
<th>0</th>
<th>1</th>
<th>1</th>
<th>0</th>
<th>1</th>
<th>1</th>
</tr>
</thead>
<tbody>
<tr>
<td>Structuring Element</td>
<td>1</td>
<td>1</td>
<td>1</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>Output Image</td>
<td></td>
<td></td>
<td></td>
<td>1</td>
<td>1</td>
<td></td>
<td></td>
<td></td>
</tr>
</tbody>
</table>

Slide credit: Kristen Grauman (from T. Moeslund)
Example for Dilation

Input image: 1 0 0 0 1 1 0 1 1
Structuring Element: 1 1 1
Output Image: 1 1 0

Example for Dilation

Input image: 1 0 0 0 1 1 0 1 1
Structuring Element: 1 1 1
Output Image: 1 1 0 0

Example for Dilation

Input image: 1 0 0 0 1 1 0 1 1
Structuring Element: 1 1 1
Output Image: 1 1 0 1 1
Example for Dilation

Input image: 1 0 0 0 1 1 1 0 1 1

Structuring Element: 1 1 1

Output Image: 1 1 0 1 1 1 1 1 1 1

Slide credit: Kristen Grauman
Example for Dilation

<table>
<thead>
<tr>
<th>Input image</th>
<th>1 0 0 0 1 1 0 1 1</th>
</tr>
</thead>
<tbody>
<tr>
<td>Structuring Element</td>
<td>1 1 1</td>
</tr>
<tr>
<td>Output Image</td>
<td>1 1 0 1 1 1 1 1 1</td>
</tr>
</tbody>
</table>

Note that the object gets bigger and holes are filled.

>> helpimdilate

Slide credit: Kristen Grauman

2D example for dilation

Dilation vs. Erosion

At each position:
- **Dilation**: if current pixel is 1, then set all the output pixels corresponding to structuring element to 1.
- **Erosion**: if every pixel under the structuring element is 1, then set the output pixel corresponding to the current pixel to 1.
Example for Erosion (1D)

Input image:

1 0 0 0 1 1 0 1 1

Structuring Element:

1 1 1

Output Image:

0 0 0 0 0
Example for Erosion

Input image
\[ 1 \ 0 \ 0 \ 0 \ 1 \ 1 \ 1 \ 0 \ 1 \ 1 \]

Structuring Element
\[ 1 \ 1 \ 1 \]

Output Image
\[ 0 \ 0 \ 0 \ 0 \]

Slide credit: Kristen Grauman
Example for Erosion

Input image: 1 0 0 0 1 1 0 1 1

Structuring Element: 1 1 1

Output Image: 0 0 0 0 1 0 0 0

Example for Erosion

Input image: 1 0 0 0 1 1 0 1 1

Structuring Element: 1 1 1

Output Image: 0 0 0 0 1 0 0 0

Example for Erosion

Input image: 1 0 0 0 1 1 0 1 1

Structuring Element: 1 1 1

Output Image: 0 0 0 0 1 0 0 0

Slide credit: Kristen Grauman
Example for Erosion

Input image
1 0 0 0 1 1 0 1 1

Structuring Element
1 1

Output Image
0 0 0 0 1 0 0 0 1

Note that the object gets smaller

Help imerode

2D example for erosion

Opening
- Erode, then dilate
- Remove small objects, keep original shape

Before opening

After opening
Closing

- Dilate, then erode
- Fill holes, but keep original shape

Before closing

After closing


Issues

- What to do with “noisy” binary outputs?
  - Holes
  - Extra small fragments
- How to demarcate multiple regions of interest?
  - Count objects
  - Compute further features per object

Connected components

- Identify distinct regions of “connected pixels”
Connectedness

• Defining which pixels are considered neighbors

4-connected 8-connected

Region properties

• Given connected components, can compute simple features per blob, such as:
  – Area (num pixels in the region)
  – Centroid (average x and y position of pixels in the region)
  – Bounding box (min and max coordinates)
Binary image analysis: basic steps (recap)

• Convert the image into binary form
  – Thresholding
• Clean up the thresholded image
  – Morphological operators
• Extract separate blobs
  – Connected components
• Describe the blobs with region properties

Matlab

• L = bwlabel (BW,8);
• STATS = regionprops(L,PROPERTIES) ;
  – 'Area'
  – 'Centroid'
  – 'BoundingBox'
  – 'Orientation', …
• IM2 = imerode(IM,SE);
• IM2 = imdilate(IM,SE);
• IM2 = imclose(IM, SE);
• IM2 = imopen(IM, SE);

Example using binary image analysis: segmentation of a liver
Binary images

- **Pros**
  - Can be fast to compute, easy to store
  - Simple processing techniques available
  - Lead to some useful compact shape descriptors

- **Cons**
  - Hard to get “clean” silhouettes
  - Noise common in realistic scenarios
  - Can be too coarse of a representation

Summary

- **Operations, tools**
  - Derivative filters
  - Smoothing, morphology
  - Thresholding
  - Connected components
  - Matching filters
  - Histograms

- **Features, representations**
  - Edges, gradients
  - Blobs/regions
  - Local patterns
  - Textures (next)
  - Color distributions

Coming up

- **Texture**
  - Read Szeliski 10.5
Questions?

See you Tuesday!