Edges and Binary Image Analysis
April 9th, 2015

Yong Jae Lee
UC Davis
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
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?**

![Image from J. Shotton et al., PAMI 2007](image1)

Figure from D. Lowe

- **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

Slide credit: Kristen Grauman
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)
Gradient magnitude image
Thresholding gradient with a lower threshold
Thresholding gradient with a higher threshold

Slide credit: Kristen Grauman
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

gradient magnitude
Compute Gradients (DoG)

X-Derivative of Gaussian  Y-Derivative of Gaussian  Gradient Magnitude

Slide credit: Svetlana Lazebnik
The Canny edge detector

gradient magnitude
The Canny edge detector

thresholding
The Canny edge detector

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

Slide credit: Kristen Grauman
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 $r$

Slide credit: Kristen Grauman
The Canny edge detector

Problem: pixels along this edge didn’t survive the thresholding

thinning
(non-maximum suppression)

Slide credit: Kristen Grauman
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

Slide credit: Fei-Fei Li
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

Background
Texture
Shadows

Slide credit: Kristen Grauman
Low-level edges vs. perceived contours

- Berkeley segmentation database:
  [http://www.eecs.berkeley.edu/Research/Projects/CS/vision/grouping/segbench/](http://www.eecs.berkeley.edu/Research/Projects/CS/vision/grouping/segbench/)

Slide credit: Svetlana Lazebnik
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…

Figure from Fowlkes
pB Boundary Detector

Figure from Fowlkes
State-of-the-Art in Contour Detection

Canny+opt thresholds

Canny

Prewitt, Sobel, Roberts

Learned with combined features

Human agreement

Source: Jitendra Malik: http://www.cs.berkeley.edu/~malik/malik-talks-ptrs.html

Slide credit: Adapted from Kristen Grauman
Global pB boundary detector

Figure from Fowlkes
Edge Detection with Structured Random Forests (Dollar Zitnick ICCV 2013)

• Goal: quickly predict whether each pixel is an edge

• Insights
  – Predictions can be learned from training data
  – Predictions for nearby pixels should not be independent

• Solution
  – Train structured random forests to split data into patches with similar boundaries based on features
  – Predict boundaries at patch level, rather than pixel level, and aggregate (average votes)

Slide credit: Derek Hoiem

Edge Detection with Structured Random Forests

Training data

Input images

Ground truth

Output

P. Dollar and L. Zitnick, Structured forests for fast edge detection, ICCV 2013

Slide credit: Svetlana Lazebnik
State of edge detection

• Local edge detection is mostly solved
  – Intensity gradient, color, texture

• Work on RGB-D edge detection is currently more active

• Often used in combination with object detectors or region classifiers
Today

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

• Binary image analysis
  – blobs and regions
Fig. 1. Examples of two handwritten digits. In terms of pixel-to-pixel comparisons, these two images are quite different, but to the human observer, the shapes appear to be similar.
Chamfer distance

- Average distance to nearest feature

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

\[I = \text{Set of points in image}\]

\[T = \text{Set of points on (shifted) template}\]

\[d_I(t) = \text{Minimum distance between point } t \text{ and some point in } I\]
Chamfer distance

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

- Average distance to nearest feature

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

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

How expensive is a naïve implementation?
Distance Transform is a function $D(\cdot)$ 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 $I$.

Features could be edge points, foreground points,…
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 $j$
   
   $D[j] \leftarrow 1_P[j]$  
   
   // 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

$$D_{chamfer}(T, I) \equiv \frac{1}{|T|} \sum_{t \in T} d_I(t)$$

Slide credit: Kristen Grauman
Chamfer distance

Fig from D. Gavrila, DAGM 1999
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

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

\[ F_T[i, j] = \begin{cases} 
1 & \text{if } T_1 \leq F[i, j] \leq T_2 \\
0 & \text{otherwise.} 
\end{cases} \]

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

• Example
Thresholding

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

Example: edge detection

Gradient magnitude

$\text{fg\_pix} = \text{find} (\text{gradient\_mag} > t)$;

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.

\[ fg\_pix = \text{find(diff > t)}; \]
Thresholding

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

Example: intensity-based detection

Looking for dark pixels

\[
\text{fg\_pix} = \text{find}(\text{im} < 65);
\]
Thresholding

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

Example: color-based detection

Looking for pixels within a certain hue range.

\[ \text{fg_pix} = \text{find(hue} > t1 \text{ & hue} < t2); \]
A nice case: bimodal intensity histograms

Ideal histogram, light object on dark background

Actual observed histogram with noise


Slide credit: Kristen Grauman
Not so nice cases

Two distinct modes

Overlapped modes

Slide credit: Shapiro and Stockman
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

Slide credit: Kristen Grauman
Erosion

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

Before erosion

After erosion

Slide credit: Kristen Grauman
Structuring elements

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

  ![Masks Diagram]

  - 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

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

Structuring Element: 1 1 1

Output Image: 1 1
Example for Dilation

Input image

$$\begin{array}{cccccccccc}
1 & 0 & 0 & 0 & 1 & 1 & 1 & 1 & 0 & 1 & 1 \\
\end{array}$$

Structuring Element

$$\begin{array}{c}
1 & 1 & 1 \\
\end{array}$$

Output Image

$$\begin{array}{c}
1 & 1 \\
\end{array}$$

Slide credit: Kristen Grauman
Example for Dilation

Input image

| 1 | 0 | 0 | 0 | 1 | 1 | 1 | 1 | 0 | 1 | 1 |

Structuring Element

| 1 | 1 | 1 | 1 |

Output Image

| 1 | 1 | 1 | 0 | 0 |
Example for Dilation

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

Structuring Element: 1 1 1 1

Output Image: 1 1 0 1 1 1 1 1

Slide credit: Kristen Grauman
Example for Dilation

**Input image**

```
1 0 0 0 0 1 1 1 1 1 0 1 1 1
```

**Structuring Element**

```
1 1 1 1
```

**Output Image**

```
1 1 0 1 1 1 1 1
```
Example for Dilation

Input image

| 1 | 0 | 0 | 0 | 1 | 1 | 1 | 1 | 0 | 1 | 1 |

Structuring Element

| 1 | 1 | 1 | 1 |

Output Image

| 1 | 1 | 0 | 1 | 1 | 1 | 1 | 1 | 1 | 1 |
Example for Dilation

Input image

Structuring Element

Output Image

Slide credit: Kristen Grauman
Example for Dilation

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

Structuring Element
1 1 1 1

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

Note that the object gets bigger and holes are filled.

>> help imdilate
2D example for dilation

(a) Binary image $B$

(b) Structuring element $S$

(c) Dilation $B \oplus S$

Slide credit: Shapiro & Stockman
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 1 1 0 1 1

Structuring Element: 1 1 1

Output Image: 0
Example for Erosion (1D)

**Input image**

```
1 0 0 0 1 1 1 1 0 1 1 1
```

**Structuring Element**

```
1 1 1
```

**Output Image**

```
0 0
```
Example for Erosion

Input image:

```
1 0 0 0 1 1 1 1 0 1 1 1
```

Structuring Element:

```
1 1 1 1
```

Output Image:

```
0 0 0 0
```
Example for Erosion

Input image:

```
1 0 0 0 1 1 1 1 0 1 1 1
```

Structuring Element:

```
1 1 1 1
```

Output Image:

```
0 0 0 0 0
```
Example for Erosion

Input image:

```
1 0 0 0 1 1 1 1 0 1 1
```

Structuring Element:

```
1 1 1 1
```

Output Image:

```
0 0 0 0 0 0 0
```
Example for Erosion

Input image

1 0 0 0 1 1 1 1 0 1 1

Structuring Element

1 1 1 1

Output Image

0 0 0 0 0 0 0 1
Example for Erosion

Input image

Structuring Element

Output Image

Slide credit: Kristen Grauman
Example for Erosion

Input image

[1 0 0 0 1 1 1 1 1 0 1 1 1]

Structuring Element

[1 1 1 1]

Output Image

[0 0 0 0 0 0 1 0 0 0]
Example for Erosion

**Input image**

```
1 0 0 0 1 1 1 1 0 1 1 1
```

**Structuring Element**

```
1 1 1 1
```

**Output Image**

```
0 0 0 0 0 0 1 0 0 0 0
```
Example for Erosion

Input image

1 0 0 0 1 1 1 1 0 1 1

Structuring Element

1 1 1

Output Image

0 0 0 0 0 0 1 1 0 0 0 1

Note that the object gets smaller

>> help imerode

Slide credit: Kristen Grauman
2D example for erosion

(a) Binary image \( B \)

(b) Structuring element \( S \)

(d) Erosion \( B \ominus S \)

Slide credit: Shapiro & Stockman
Opening

• Erode, then dilate
• Remove small objects, keep original shape

Before opening

After opening

Slide credit: Kristen Grauman
Closing

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

Applet: [http://bigwww.epfl.ch/demo/jmorpho/start.php](http://bigwww.epfl.ch/demo/jmorpho/start.php)

Slide credit: Kristen Grauman
Morphology operators on grayscale images

- Dilation and erosion typically performed on binary images.
- If image is grayscale: for dilation take the neighborhood $\text{max}$, for erosion take the $\text{min}$.  

original  
dilated  
eroded  

Slide credit: Kristen Grauman
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

Slide credit: Kristen Grauman
Connected components

- Identify distinct regions of “connected pixels”

![Binary Image](image1)

![Connected Components Labeling](image2)

![Binary Image and Labeling, Expanded for Viewing](image3)
Connectedness

- Defining which pixels are considered neighbors

4-connected

8-connected

[i, j]

Slide credit: Chaitanya Chandra
Connected components

• We’ll consider a sequential algorithm that requires only 2 passes over the image.

• **Input**: binary image

• **Output**: “label” image, where pixels are numbered per their component

• Note: foreground here is denoted with black pixels.

Slide credit: Kristen Grauman
Sequential connected components

- Labeling a pixel only requires to consider its prior and superior neighbors.
- It depends on the type of connectivity used for foreground (4-connectivity here).

<table>
<thead>
<tr>
<th>Same object</th>
<th>New object</th>
</tr>
</thead>
<tbody>
<tr>
<td>(a)</td>
<td>(b)</td>
</tr>
<tr>
<td>(c)</td>
<td>(d)</td>
</tr>
</tbody>
</table>

What happens in these cases?
Sequential connected components

- Labeling a pixel only requires to consider its prior and superior neighbors.
- It depends on the type of connectivity used for foreground (4-connectivity here).

What happens in these cases?

Slide credit: Kristen Grauman
Sequential connected components

- Labeling a pixel only requires to consider its prior and superior neighbors.
- It depends on the type of connectivity used for foreground (4-connectivity here).

![Same object](a) ![New object](b) ![New object](c) ![New object](d)

What happens in these cases?

Slide credit: Kristen Grauman
Sequential connected components

- Process the image from left to right, top to bottom.

1. If the next pixel to process is 1-pixel:
   1. If only one of its neighbors (superior or left) is 1-pixel, copy its label.
   2. If both are, and have the same label, copy it.
   3. If they have different labels:
      1. Copy the label from the prior.
      2. Reflect the change in the table of equivalences.
   4. Otw, assign a new label.


Slide credit: Kristen Grauman
Connected components

connected components of 1’s from thresholded image

connected components of cluster labels

Slide credit: Pinar Duygulu
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)
  – Circularity (ratio of mean dist. to centroid over std)

\[ A_1 = 200 \]
\[ A_2 = 170 \]

Slide credit: Kristen Grauman
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

- \( N = \text{hist}(Y,M) \)
- \( L = \text{bwlabel}(BW,N) \)
- \( \text{STATS} = \text{regionprops}(L,\text{PROPERTIES}) \)
  - 'Area'
  - 'Centroid'
  - 'BoundingBox'
  - 'Orientation', ...
- \( \text{IM2} = \text{imerode}(IM,SE) \)
- \( \text{IM2} = \text{imdilate}(IM,SE) \)
- \( \text{IM2} = \text{imclose}(IM,SE) \)
- \( \text{IM2} = \text{imopen}(IM,SE) \)
Example using binary image analysis: OCR

Digitizing Books One Word at a Time

reCAPTCHA is a free CAPTCHA service that helps to digitize books, newspapers and old time radio shows. Check out our paper in Science about it (or read more below).

A CAPTCHA is a program that can tell whether its user is a human or a computer. You've probably seen them — colorful images with distorted text at the bottom of Web registration forms. CAPTCHAs are used by many websites to prevent abuse from “bots,” or automated programs usually written to generate spam. No computer program can read distorted text as well as humans can, so bots cannot navigate sites protected by CAPTCHAs.

Example using binary image analysis: segmentation of a liver

Slide credit: Li Shen

Application by Jie Zhu, Cornell University
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
  – Not 3d
## Summary

<table>
<thead>
<tr>
<th>Operations, tools</th>
<th>Features, representations</th>
</tr>
</thead>
<tbody>
<tr>
<td>Derivative filters</td>
<td>Edges, gradients</td>
</tr>
<tr>
<td>Smoothing, morphology</td>
<td>Blobs/regions</td>
</tr>
<tr>
<td>Thresholding</td>
<td>Local patterns</td>
</tr>
<tr>
<td>Connected components</td>
<td>Textures (next)</td>
</tr>
<tr>
<td>Connected components</td>
<td>Color distributions</td>
</tr>
<tr>
<td>Matched filters</td>
<td></td>
</tr>
</tbody>
</table>
Coming up

• Texture
  – Read Szeliski 10.5
Questions?

See you Tuesday!