


Edges and Binary Images

Tuesday, Sept 16


Review

- Thought question: how could we compute a *temporal gradient* from video data?
- What filter is likely to have produced this image output?

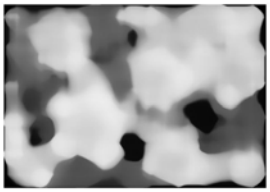

original


filtered output



5 x 5 median filter


25 x 25 median filter


50 x 50 median filter

Edge detection

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

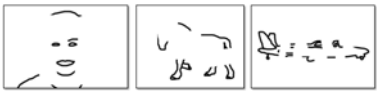


Figure from J. Shotton et al., PAMI 2007

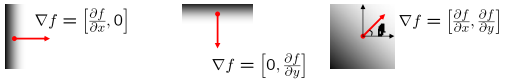
- **Main idea**: look for strong **gradients**, post-process

Image gradient

The gradient of an image:

$$\nabla f = \left[\frac{\partial f}{\partial x}, \frac{\partial f}{\partial y}\right]$$

The gradient points in the direction of most rapid change in intensity



The gradient direction (orientation of edge normal) is given by:

$$\theta = \tan^{-1}\left(\frac{\partial f}{\partial y} / \frac{\partial f}{\partial x}\right)$$

The *edge strength* is given by the gradient magnitude

$$\|\nabla f\| = \sqrt{\left(\frac{\partial f}{\partial x}\right)^2 + \left(\frac{\partial f}{\partial y}\right)^2}$$

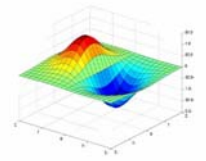
Slide credit S. Seitz

Derivative of Gaussian filter

$$(I \otimes g) \otimes h = I \otimes (g \otimes h)$$

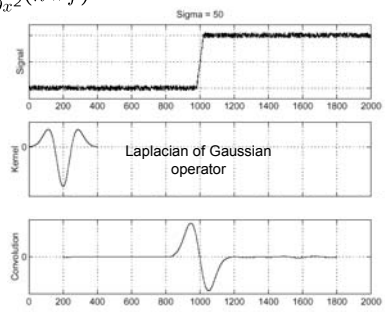
$$\begin{bmatrix} 0.0030 & 0.0133 & 0.0219 & 0.0133 & 0.0030 \\ 0.0133 & 0.0596 & 0.0983 & 0.0596 & 0.0133 \\ 0.0219 & 0.0983 & 0.1621 & 0.0983 & 0.0219 \\ 0.0133 & 0.0596 & 0.0983 & 0.0596 & 0.0133 \\ 0.0030 & 0.0133 & 0.0219 & 0.0133 & 0.0030 \end{bmatrix} \otimes \begin{bmatrix} 1 & -1 \end{bmatrix}$$

We can smooth and take the derivative with one filter, that is, with one convolution pass.



Laplacian of Gaussian

Consider $\frac{\partial^2}{\partial x^2}(h \star f)$



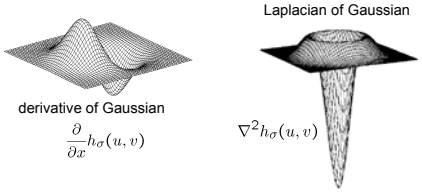
f

$\frac{\partial^2}{\partial x^2}h$

$(\frac{\partial^2}{\partial x^2}h) \star f$

Where is the edge? Zero-crossings of bottom graph

2D edge detection filters



derivative of Gaussian $\frac{\partial}{\partial x}h_\sigma(u,v)$

$\nabla^2 h_\sigma(u,v)$

- ∇^2 is the **Laplacian** operator:

$$\nabla^2 f = \frac{\partial^2 f}{\partial x^2} + \frac{\partial^2 f}{\partial y^2}$$

Gradients -> edges

Primary edge detection steps:

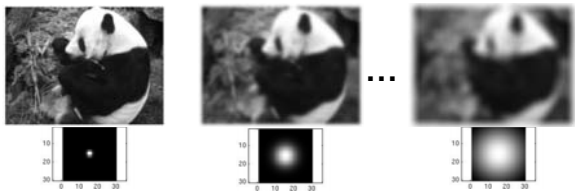
1. Smoothing: suppress noise
2. Edge enhancement: filter for contrast
3. Edge localization

   Determine which local maxima from filter output are actually edges vs. noise

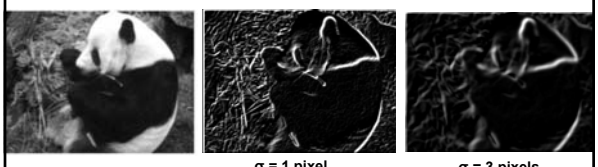
   - Threshold, Thin

Effect of σ on Gaussian smoothing

Recall: parameter σ is the "scale" / "width" / "spread" of the Gaussian kernel, and controls the amount of smoothing.



Effect of σ on derivatives



σ = 1 pixel σ = 3 pixels

The apparent structures differ depending on Gaussian's scale parameter.

Larger values: larger scale edges detected
Smaller values: finer features detected

So, what scale to choose?

It depends what we're looking for.

Often we may want to analyze at multiple scales.

Too fine of a scale…can't see the forest for the trees.
Too coarse of a scale…can't tell the maple grain from the cherry.

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)

Original image

Gradient magnitude image

Thresholding gradient with a lower threshold

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 multi-pixel 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`

Source: D. Lowe, L. Fei-Fei

The Canny edge detector

original image (Lena)

Source: S. Seitz

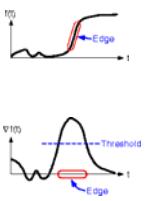
The Canny edge detector

norm of the gradient

The Canny edge detector

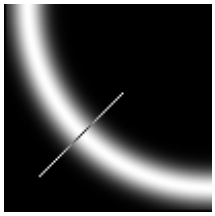
thresholding

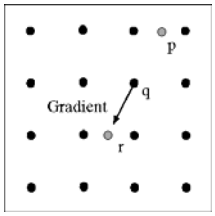
The Canny edge detector



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

thinning
(non-maximum suppression)

Hysteresis thresholding

- Check that maximum value of gradient value is sufficiently large
  - drop-outs? use **hysteresis**
    - use a high threshold to start edge curves and a low threshold to continue them.

Source: S. Seitz

Hysteresis thresholding

original image

high threshold
(strong edges)

low threshold
(weak edges)

hysteresis threshold

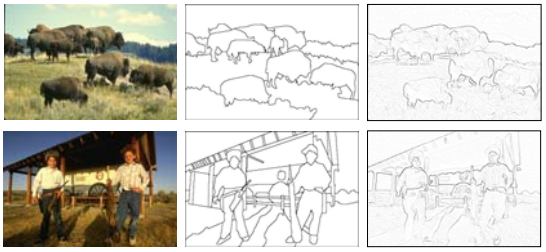
Source: L. Fei-Fei

Object boundaries vs. edges

Background **Texture** **Shadows**

Edge detection is just the beginning…

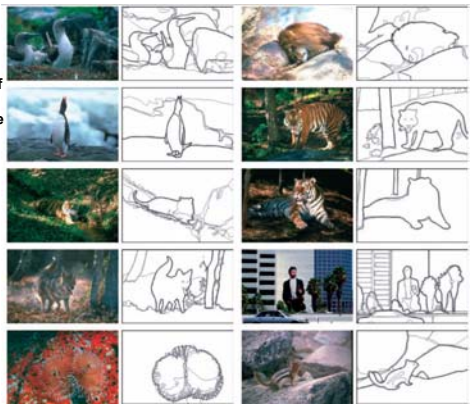
image human segmentation gradient magnitude



Berkeley segmentation database:
http://www.eecs.berkeley.edu/Research/Projects/CS/vision/grouping/segbench/

Source: L. Lazebnik

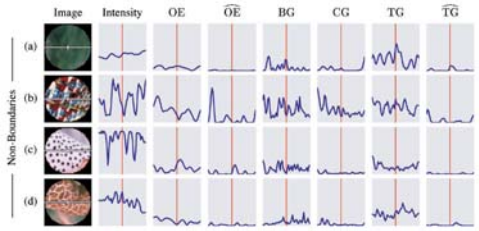
Possible to learn from humans which combination of features is most indicative of a "good" contour?



[D. Martin et al. PAMI 2004]

Human-marked segment boundaries

What features are responsible for perceived edges?

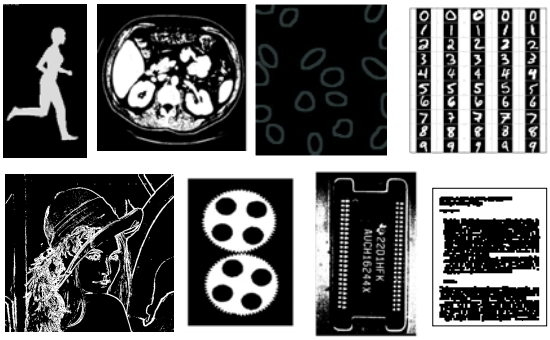


Feature profiles (oriented energy, brightness, color, and texture gradients) along the patch's horizontal diameter

[D. Martin et al. PAMI 2004]

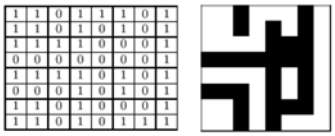
[D. Martin et al. PAMI 2004]

Binary images



Binary images

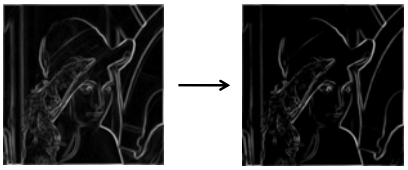
- Two pixel values
  - Foreground and background
  - Mark region(s) of interest



Thresholding

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

Example: edge detection



Gradient magnitude

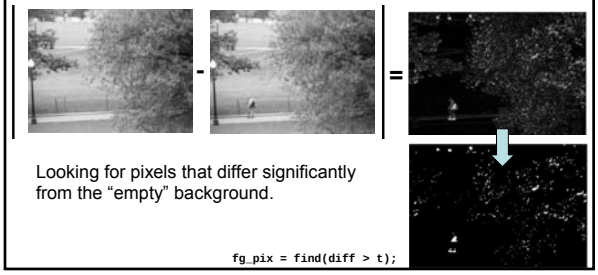
```
fg_pix = find(gradient_mag > t);
```

Looking for pixels where gradient is strong.

Thresholding

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

Example: background subtraction



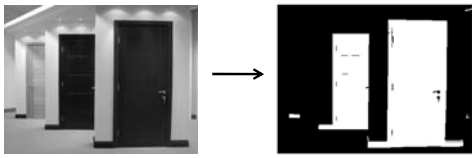
Looking for pixels that differ significantly from the “empty” background.

```
fg_pix = find(diff > t);
```

Thresholding

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

Example: intensity-based detection



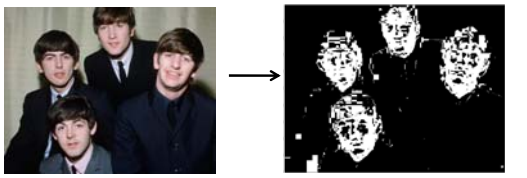
```
fg_pix = find(im < 65);
```

Looking for dark pixels

Thresholding

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

Example: color-based detection

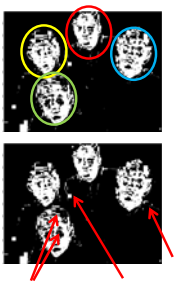


```
fg_pix = find(hue > t1 & hue < t2);
```

Looking for pixels within a certain hue range.

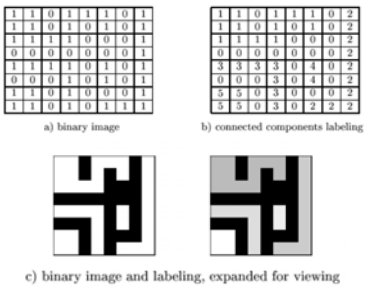
Issues

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



Connected components

- Identify distinct regions of "connected pixels"



a) binary image

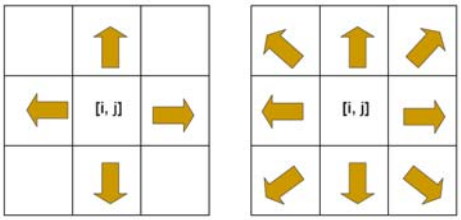
b) connected components labeling

c) binary image and labeling, expanded for viewing

Shapiro and Stockman

Connectedness

- Defining which pixels are considered neighbors

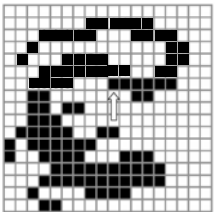


4-connected

8-connected

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, in this example, "foreground" is denoted with black.]



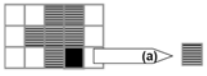
Sequential connected components

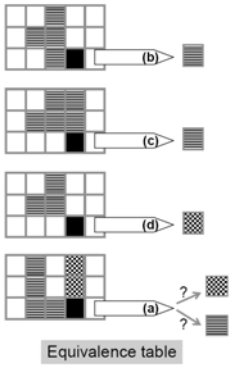
- Labeling a pixel only requires to consider its prior and superior neighbors.
- It depends on the type of connectivity used for foreground (4-connectivity here).

Same object | New object

(a) (b) (c) (d)

What happens in these cases?



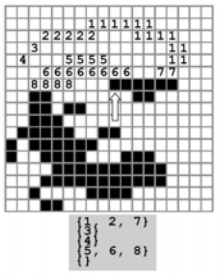


Equivalence table

Slide from J. Neira

Sequential connected components

- Process the image from left to right, top to bottom.
  1. If the next pixel to process is 1-pixel:
     Already processed
     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:
        superior? smallest?
        1. Copy the label from the prior.
        2. Reflect the change in the table of equivalences.
     4. Otw, assign a new label.
  2. More pixels? Go to step 1.



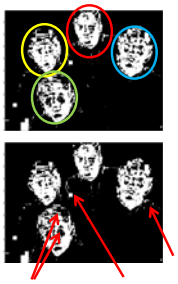
- Re-label with the smallest of equivalent labels.
- Pixels of the same segment always have the same label.

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)
- How could such features be useful?

Issues

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

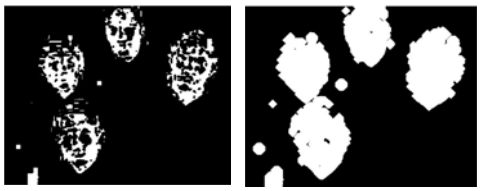


Morphological operators

- Change the shape of the foreground regions/ objects.
- Useful to clean up result from thresholding
- Basic operators are:
  - Dilation
  - Erosion

Dilation

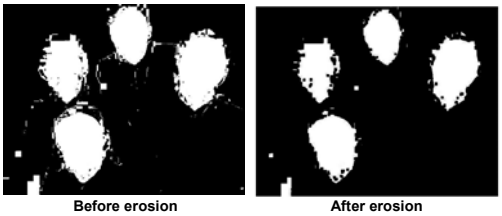
- Expands connected components
- Grow features
- Fill holes



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

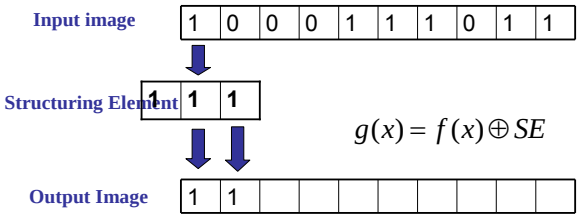
```
>> help strel
```

Dilation vs. Erosion

At each position:

- **Dilation**: if current pixel is foreground, OR the structuring element with the input image.

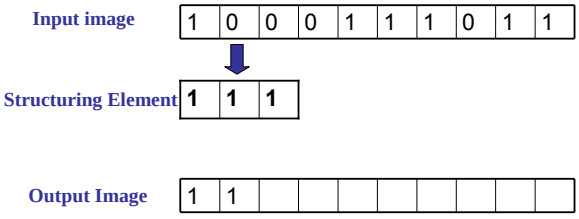
Example for Dilation (1D)



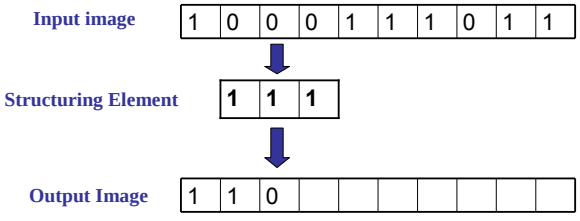
$$g(x) = f(x) \oplus SE$$

Adapted from T. Moeslund

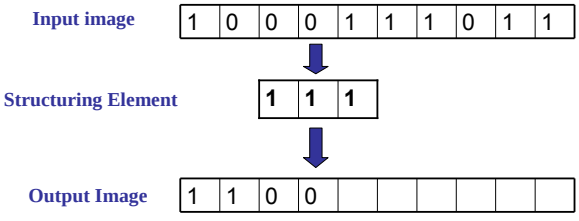
Example for Dilation



Example for Dilation



Example for Dilation



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				

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			

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		

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		

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

Note that the object gets bigger and holes are filled.

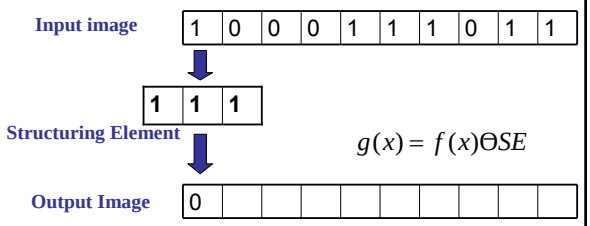
```
>> help imdilate
```

Dilation vs. Erosion

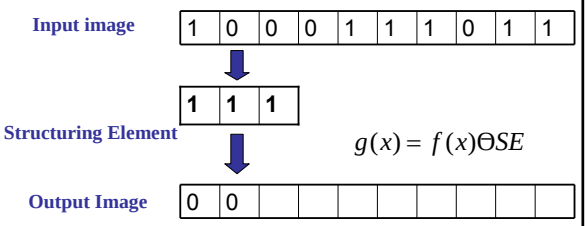
At each position:

- **Dilation**: if current pixel is foreground, OR the structuring element with the input image.
- **Erosion**: if every pixel under the structuring element's nonzero entries is foreground, OR the current pixel with S.

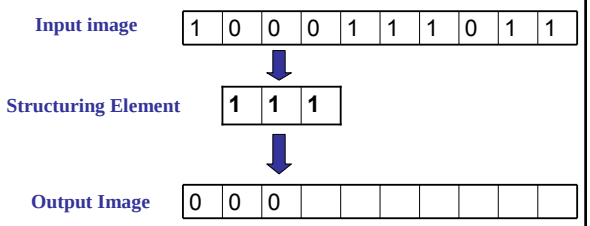
Example for Erosion (1D)



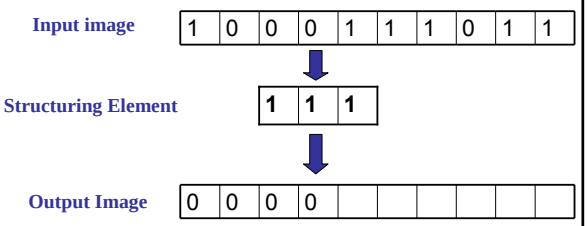
Example for Erosion (1D)



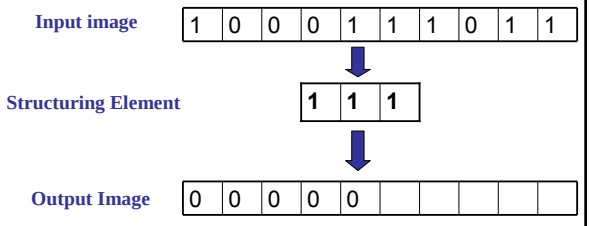
Example for Erosion



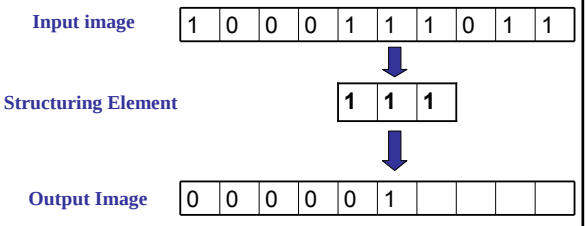
Example for Erosion



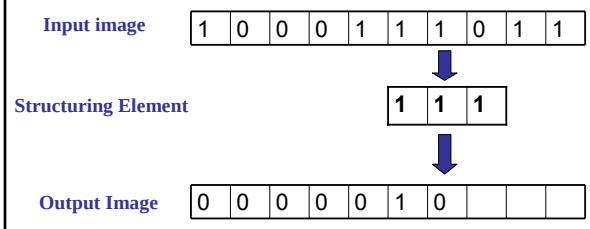
Example for Erosion



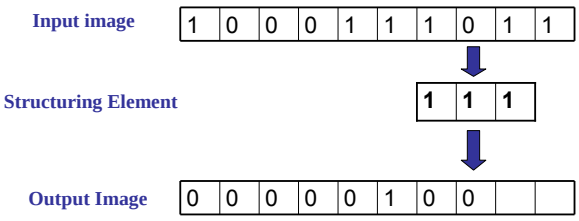
Example for Erosion



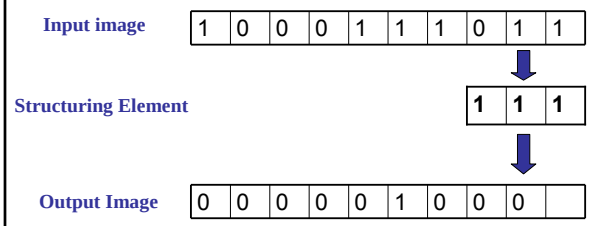
Example for Erosion



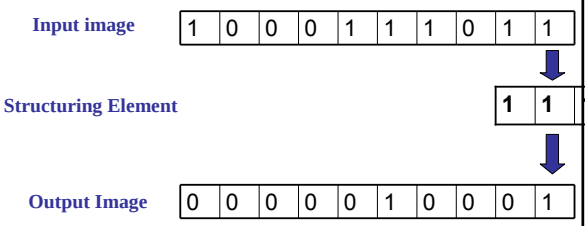
Example for Erosion



Example for Erosion



Example for Erosion



Note that the object gets smaller

```
>> help imerode
```

Opening

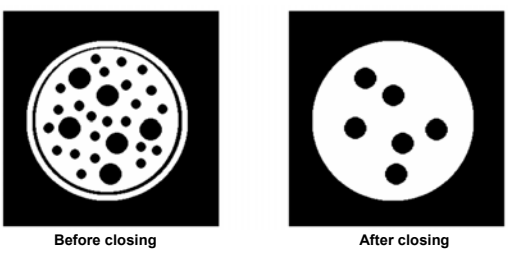
- Erode, then dilate
- Remove small objects, keep original shape

Before opening

After opening

Closing

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



Before closing

After closing

Morphology operators on grayscale images

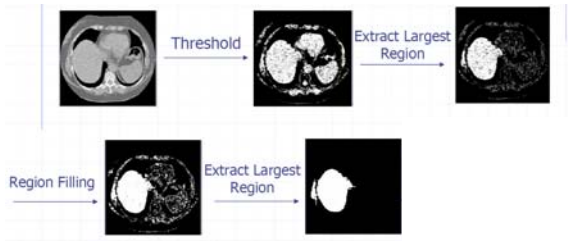
- Dilation and erosion typically performed on binary images.
- If image is grayscale: for dilation take the neighborhood max, for erosion take the min.

original dilated eroded

Matlab

- `N = hist(Y,M)`
- `L = bwlabel (BW,N);`
- `STATS = regionprops(L,PROPERTIES) ;`
  - `'Area'`
  - `'Centroid'`
  - `'BoundingBox'`
  - `'Orientation', …`
- `IM2 = imerode(IM,SE);`
- `IM2 = imdilate(IM,SE);`
- `IM2 = imclose(IM, SE);`
- `IM2 = imopen(IM, SE);`

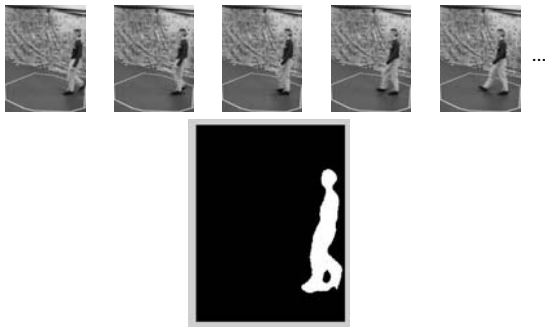
Example using binary image analysis: segmentation of a liver



Slide credit: Li Shen

Application by Jie Zhu, Cornell University

Example using binary image analysis: Bg subtraction + blob detection



Example using binary image analysis: Bg subtraction + blob detection

University of Southern California
http://iris.usc.edu/~icohen/projects/vace/detection.htm

Summary

- **Filters** allow local image neighborhood to influence our description and features
  - Smoothing to reduce noise
  - Derivatives to locate contrast, gradient
  - Templates, matched filters to find designated pattern.
- **Edge detection** processes the image gradient to find curves, or chains of edgels.
- **Binary image analysis** useful to manipulate regions of interest
  - Connected components
  - Morphological operators

Summary

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

⇩⇩⇩⇩⇩⇩⇩⇩

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

Next

- Texture: read F&P Ch 9, Sections 9.1, 9.3