Computer Vision

Dr. Danna Gurari
September 8, 2015

1

Today's Outline

- Aligning two images
  - Chamfer distance
  - Applications
- Analyze binary images
  - Thresholding
  - Morphological operators
  - Connected components
  - Region properties
  - Applications

Today's Outline

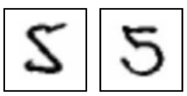
- Aligning two images
  - Chamfer distance
  - Applications
- Analyze binary images
  - Thresholding
  - Morphological operators
  - Connected components
  - Region properties
  - Applications

Aligning Two Images: Idea

- Pixel level similarity? Different.
- Shape similarity? Similar!

- Figure from Belongie et al.

Chamfer distance



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

$\boldsymbol{I}$ = Set of points in image

$\boldsymbol{T}$ = Set of points on template

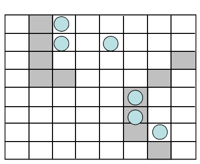
$\boldsymbol{d_I(t)}$ = Minimum distance between point t and some point in I

5

Chamfer distance

- Average matching distance to nearest feature

I:



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

Minimum distance between point t and some point in I

Chamfer distance

- Average matching distance to nearest feature

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

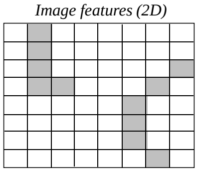
How is the measure different than just filtering with a mask having the shape points?

How expensive is a naïve implementation?

Edge image

Distance transform

Image features (2D)



Distance Transform

1	0	1	2	3	4	3	2
1	0	1	2	3	3	2	1
1	0	1	2	3	2	1	0
1	0	0	1	2	1	0	1
2	1	1	2	1	0	1	2
3	2	2	2	1	0	1	2
4	3	3	2	1	0	1	2
5	4	4	3	2	1	0	1

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

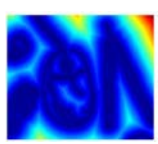
Features could be edge points, foreground points,…

Source: Yuri Boykov

Distance transform

original

edges



distance transform

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

```
>> help bwdist
```

Distance transform (1D)

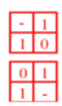
Two pass O(n) algorithm for 1D L_1 norm

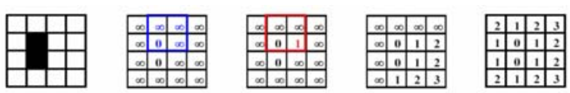
1. Initialize: For all j
   $D[j] \leftarrow 1_P[j]$ // 0 if j is in **P**, infinity otherwise

Adapted from D. Huttenlocher

Distance Transform (2D)

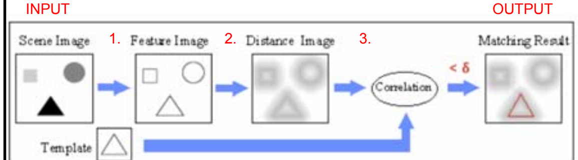
- 2D case analogous to 1D
  - Initialization
  - Forward and backward pass
    - Fwd pass finds closest above and to left
    - Bwd pass finds closest below and to right





Adapted from D. Huttenlocher

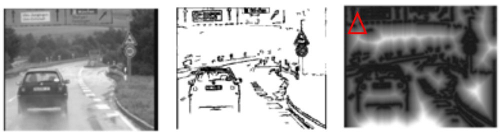
Chamfer distance: Example



Chamfer distance: Example

- Average distance to nearest feature

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



Edge image   Distance transform image

Chamfer distance: Recognition

Edge image   Distance transform image

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

Chamfer matching system

QuickTime™ and a Microsoft Video 1 decompressor are needed to see this picture.

QuickTime™ and a Microsoft Video 1 decompressor are needed to see this picture.

- Gavrila et al. http://gavrila.net/Research/Chamfer_System/chamfer_system.html

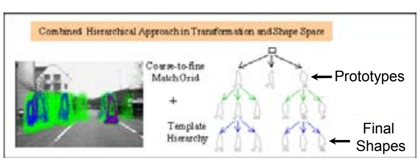
Chamfer matching system

QuickTime™ and a Microsoft Video 1 decompressor are needed to see this picture.

QuickTime™ and a Microsoft Video 1 decompressor are needed to see this picture.

- Gavrila et al. http://gavrila.net/Research/Chamfer_System/chamfer_system.html

Chamfer matching system



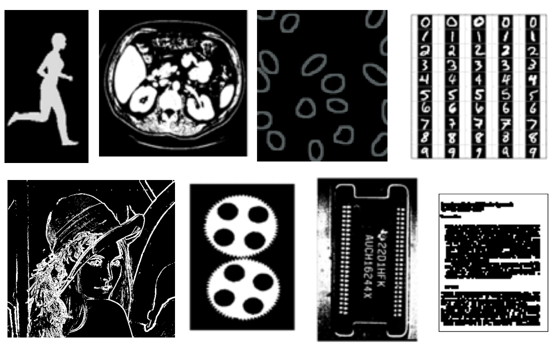
QuickTime™ and a Microsoft Video 1 decompressor are needed to see this picture.

- Gavrila et al. http://gavrila.net/Research/Chamfer_System/chamfer_system.html

Today's Outline

- Aligning two images
  - Chamfer distance
  - Applications
- Analyze binary images
  - Thresholding
  - Morphological operators
  - Connected components
  - Region properties
  - Applications

Binary images



Kristen Grauman, UT-Austin

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

Kristen Grauman, UT-Austin

Binary images

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

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

Thresholding

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

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

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

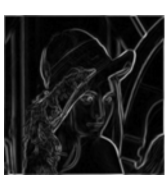
$$F_T[i,j] = \begin{cases} 1 & \text{if } F[i,j] \in Z \\ 0 & otherwise. \end{cases}$$

- Example
  http://homepages.inf.ed.ac.uk/rbf/CVonline/LOCAL_COPIES/FITZGIBBON/simplebinary.html

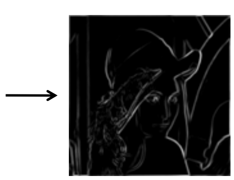
Thresholding

- 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);`

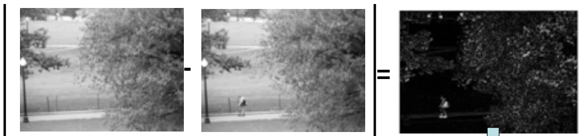
Looking for pixels where gradient is strong.

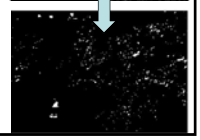
Kristen Grauman, UT-Austin

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.

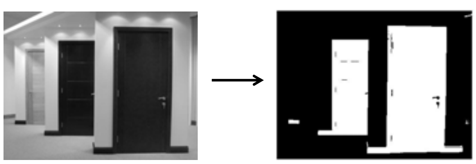
`fg_pix = find(diff > t);`

Kristen Grauman, UT-Austin

Thresholding

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

Example: intensity-based detection



`fg_pix = find(im < 65);`

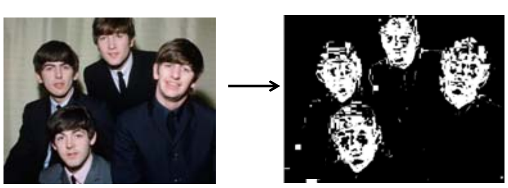
Looking for dark pixels

Kristen Grauman, UT-Austin

Thresholding

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

Example: color-based detection

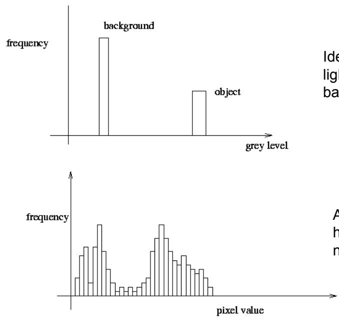


`fg_pix = find(hue > t1 & hue < t2);`

Looking for pixels within a certain hue range.

Kristen Grauman, UT-Austin

A nice case: bimodal intensity histograms

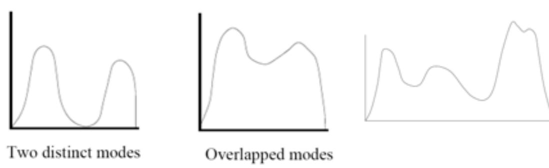


Ideal histogram, light object on dark background

Actual observed histogram with noise

Images: http://homepages.inf.ed.ac.uk/rbf/CVonline/LOCAL_COPIES/OWENS/LECT2/node3.html

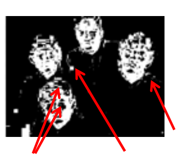
Not so nice cases

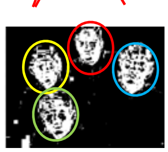


Shapiro and Stockman

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





Kristen Grauman, UT-Austin

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

Kristen Grauman, UT-Austin

Erosion

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

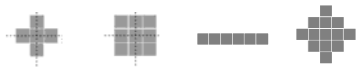
Before erosion

After erosion

Kristen Grauman, UT-Austin

Structuring elements

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



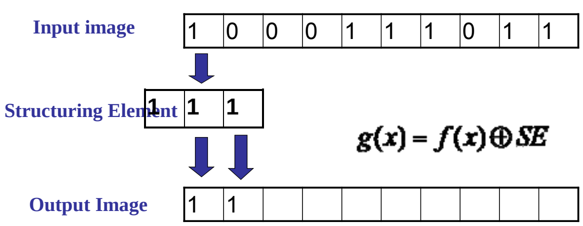
- Scan mask across foreground pixels to transform the binary image
- `>> help strel`

Kristen Grauman, UT-Austin

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

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

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	0						

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**

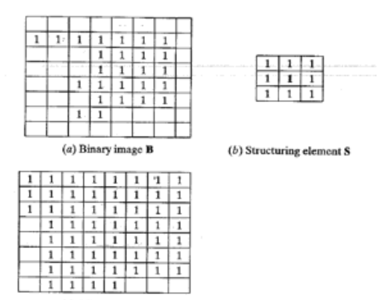
Output Image

1	1	0	1	1	1	1	1	1	1

Note that the object gets bigger and holes are filled.

```
>> help imdilate
```

2D example for dilation



Shapiro & Stockman

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)

Input image

1	0	0	0	1	1	1	0	1	1

Structuring Element

1	**1**	**1**

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

Output Image

0									

Example for Erosion (1D)

Input image

1	0	0	0	1	1	1	0	1	1

Structuring Element

1	**1**	**1**

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

Output Image

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							

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						

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	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	0	1				

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	0	1	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	0	1	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	0	1	0	0	0	

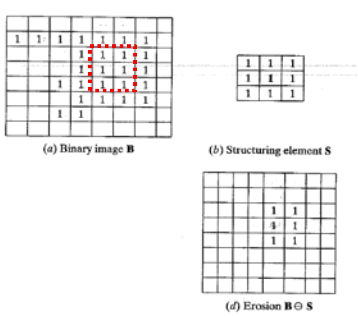
Example for Erosion

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

Note that the object gets smaller

```
>> help imerode
```

2D example for erosion



Shapiro & Stockman

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

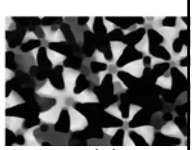
Applet: http://bigwww.epfl.ch/demo/jmorpho/start.php

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



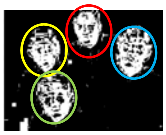
eroded

Kristen Grauman, UT-Austin

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



Kristen Grauman, UT-Austin

Connected components

- Various algorithms to compute
  - Recursive (in memory)
  - Two rows at a time (image not necessarily in memory)
  - Parallel propagation strategy

Recursive connected components

- Find an unlabeled pixel, assign it a new label
- Search to find its neighbors, and recursively repeat to find their neighbors til there are no more
- Repeat

- Demo http://www.cosc.canterbury.ac.nz/mukundan/covn/Label.html

Connected components

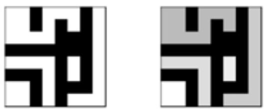
- Identify distinct regions of "connected pixels"

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

a) binary image

1	1	0	1	1	1	0	2
1	1	0	1	0	1	0	2
1	1	1	1	0	0	0	2
0	0	0	0	0	0	0	2
3	3	3	3	0	4	0	2
0	0	0	3	0	4	0	2
5	5	0	3	0	0	0	2
5	5	0	3	0	2	2	2

b) connected components labeling

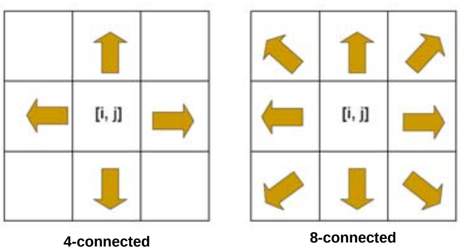


c) binary image and labeling, expanded for viewing

Shapiro and Stockman

Connectedness

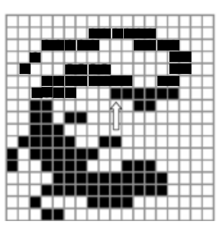
- Defining which pixels are considered neighbors



Source: Chaitanya Chandra

Connected components

- We'll consider a sequential algorithm that requires only 2 passes over the image.

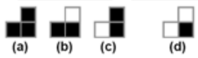


- **Input**: binary image
- **Output**: "label" image, where pixels are numbered per their

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



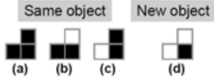
What happens in these cases?

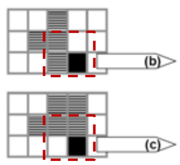
Adapted from J. Neira

Sequential connected components

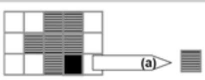
- Labeling a pixel only requires to consider its prior and superior neighbors.
- It depends on the type of connectivity used for foreground (4-connectivity here).

Same object | New object





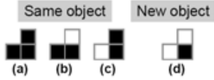
What happens in these cases?

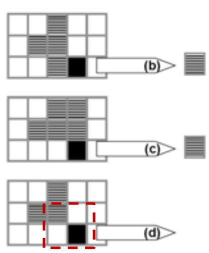


Sequential connected components

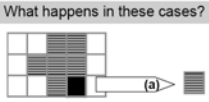
- Labeling a pixel only requires to consider its prior and superior neighbors.
- It depends on the type of connectivity used for foreground (4-connectivity here).

Same object | New object





What happens in these cases?



Sequential connected components

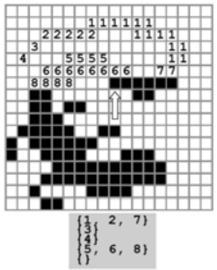
- Process the image from left to right, top to bottom.
  1. If the next pixel to process is 1-pixel:

     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?

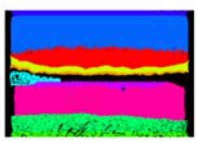
        1. Copy the label from the prior.
        2. Reflect the change in the table of equivalences.
     4. Otw, assign a new label.
  2. More pixels? Go to step 1.



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

Connected components


connected components of 1's from thresholded image

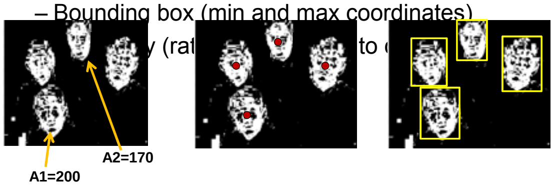



connected components of cluster labels

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)
  - ...y (rat... to ...



Kristen Grauman, UT-Austin

Circularity

- a second measure uses variation off of a circle

circularity(2):

$$C_2 = \frac{\mu_R}{\sigma_R}$$

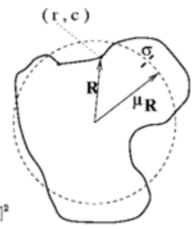
where μ_R and $\sigma_R{}^2$ are the mean and variance of the distance from the centroid of the shape to the boundary pixels (r_k, c_k).

mean radial distance:

$$\mu_R = \frac{1}{K} \sum_{k=0}^{K-1} \|(r_k, c_k) - (\bar{r}, \bar{c})\|$$

variance of radial distance:

$$\sigma_R^2 = \frac{1}{K} \sum_{k=0}^{K-1} [\|(r_k, c_k) - (\bar{r}, \bar{c})\| - \mu_R]^2$$



[Haralick]

Shapiro & Stockman

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

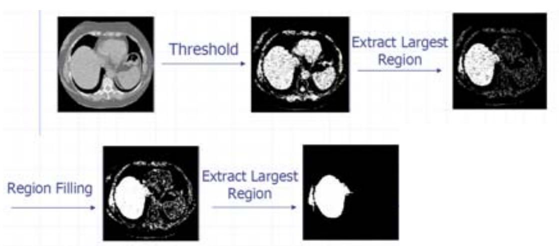
Matlab

- **`N = 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: OCR

[Luis von Ahn et al. http://recaptcha.net/learnmore.html]

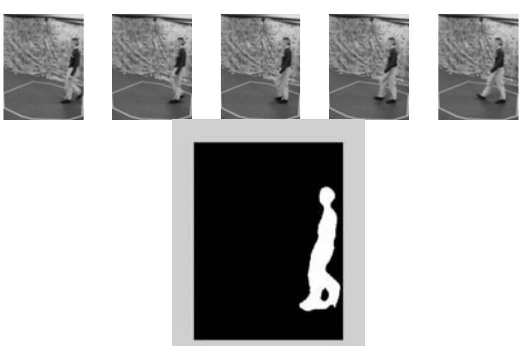
Example using binary image analysis: segmentation of a liver



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

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

Kristen Grauman, UT-Austin

Today’s Outline (Completed)

- Aligning two images
  - Chamfer distance
  - Applications
- Analyze binary images
  - Thresholding
  - Morphological operators
  - Connected components
  - Region properties
  - Applications

Background Subtraction

- Given an image (mostly likely to be a video frame), we want to identify the **foreground objects** in that image!

⇒

Motivation

- In most cases, objects are of interest, not the scene.
- Makes our life easier: less processing costs, and less room for error.

Slide credit: Birgi Tamersoy

Background subtraction

- Simple techniques can do ok with static camera
- …But hard to do perfectly

- Widely used:
  - Traffic monitoring (counting vehicles, detecting & tracking vehicles, pedestrians),
  - Human action recognition (run, walk, jump, squat),
  - Human-computer interaction
  - Object tracking

Simple Approach

Image at time t: $I(x, y, t)$ ⇓

Background at time t: $B(x, y, t)$ ⇓

|  −  | $> Th$

1. Estimate the background for time t.
2. Subtract the estimated background from the input frame.
3. Apply a threshold, Th, to the absolute difference to get the **foreground mask**.

Slide credit: Birgi Tamersoy

Frame Differencing

- Background is estimated to be the previous frame. Background subtraction equation then becomes:

$$B(x, y, t) = I(x, y, t-1)$$
$$\Downarrow$$
$$|I(x, y, t) - I(x, y, t-1)| > Th$$

- Depending on the object structure, speed, frame rate and global threshold, this approach may or may **not** be useful (usually **not**).

|  −  | $> Th$

Slide credit: Birgi Tamersoy

Frame Differencing

$Th = 25$

$Th = 50$

$Th = 100$

$Th = 200$

Slide credit: Birgi Tamersoy

Mean Filter

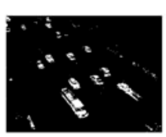
- In this case the background is the mean of the previous n frames:

$$B(x, y, t) = \frac{1}{n}\sum_{i=0}^{n-1} I(x, y, t-i)$$
$$\Downarrow$$
$$|I(x, y, t) - \frac{1}{n}\sum_{i=0}^{n-1} I(x, y, t-i)| > Th$$

- For $n = 10$:

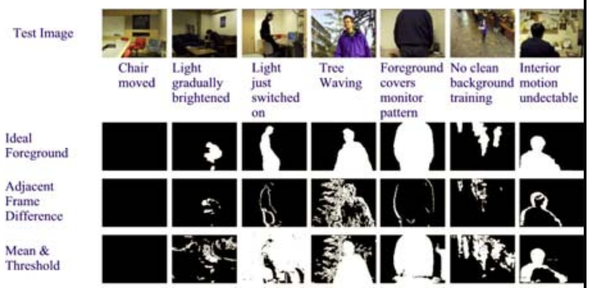
Estimated Background

Foreground Mask



Slide credit: Birgi Tamersoy

Frame differences vs. background subtraction



- Toyama et al. 1999

Median Filter

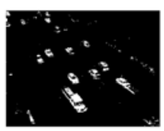
- Assuming that the background is more likely to appear in a scene, we can use the median of the previous n frames as the background model:

$$B(x,y,t) = median\{I(x,y,t-i)\}$$
$$\Downarrow$$
$$|I(x,y,t) - median\{I(x,y,t-i)\}| > Th \text{ where } i \in \{0,\ldots,n-1\}.$$

- For $n = 10$:

Estimated Background

Foreground Mask

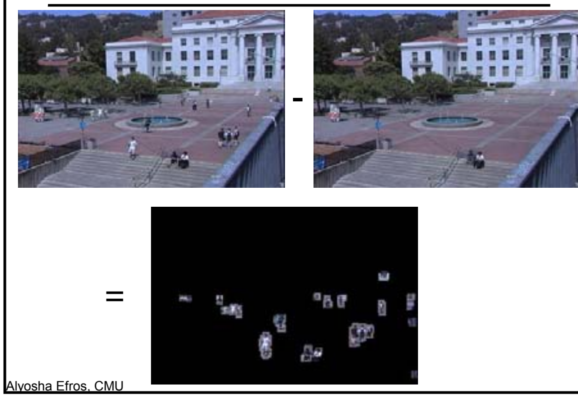


Slide credit: Birgi Tamersoy

Average/Median Image

Alyosha Efros, CMU

Background Subtraction



Alyosha Efros, CMU

Pros and cons

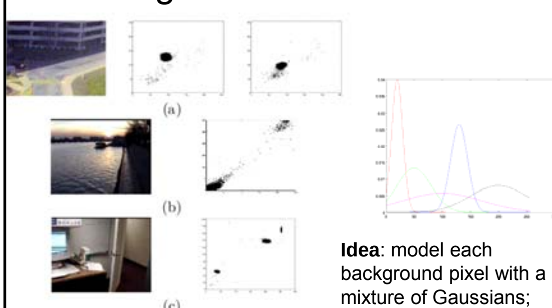
- **Advantages:**
- Extremely easy to implement and use!
- All pretty fast.
- Corresponding background models need not be constant, they change over time.

- **Disadvantages:**
- Accuracy of frame differencing depends on object speed and frame rate
- Median background model: relatively high memory requirements.

When will this basic approach fail?

Slide credit: Birgi Tamersoy

Background mixture models



Idea: model each background pixel with a mixture of Gaussians; update its parameters over time.

- Adaptive Background Mixture Models for Real-Time Tracking, Chris Stauer & W.E.L. Gr

Background subtraction with depth

How can we select foreground pixels based on depth information?