RWTH AACHEN UNIVERSITY

Computer Vision - Lecture 2

Binary Image Analysis

26.10.2015

Bastian Leibe
RWTH Aachen
http://www.vision.rwth-aachen.de/

leibe@vision.rwth-aachen.de

RWTH AACHEN UNIVERSITY

Announcements

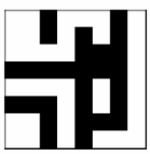
- Course webpage
  - http://www.vision.rwth-aachen.de/teaching/
  - Slides will be made available on the webpage
- L2P electronic repository
  - Exercises and supplementary materials will be posted on the L2P
- Please subscribe to the lecture on the Campus system!
  - Important to get email announcements and L2P access!
  - Bachelor students please also subscribe

RWTH AACHEN UNIVERSITY

Binary Images

- Just two pixel values
- Foreground and background
- Regions of interest

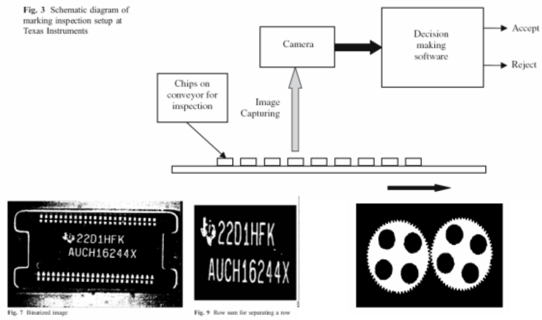
1	1	0	1	1	1	0	1
1	1	0	1	0	1	0	1
1	1	1	1	0	0	0	1
0	0	0	0	0	0	0	1
1	1	1	1	0	1	0	1
0	0	0	1	0	1	0	1
1	1	0	1	0	0	0	1
1	1	0	1	0	1	1	1



Slide credit: Kristen Grauman

RWTH AACHEN UNIVERSITY

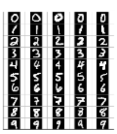
Uses: Industrial Inspection



R. Nagarajan et al. "A real time marking inspection scheme for semiconductor industries", 2006

RWTH AACHEN UNIVERSITY

Uses: Document Analysis, Text Recognition



Handwritten digits

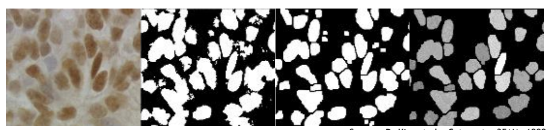
Natural text (after detection)

Scanned documents

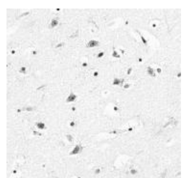
Source: Till Quack, Martin Renold

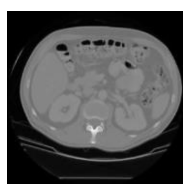
RWTH AACHEN UNIVERSITY

Uses: Medical/Bio Data



Source: D. Kim et al., Cytometry 35(1), 1999





Uses: Blob Tracking & Motion Analysis

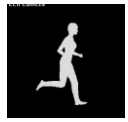
Frame Differencing

Source: Kristen Grauman

Background Subtraction

Source: Tobias Jäggli

Uses: Shape Analysis, Free-Viewpoint Video

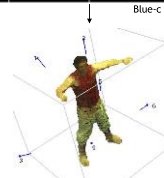


Silhouette

Visual Hull Reconstruction

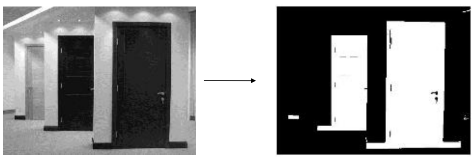
Blue-c project, ETH Zurich

Medial axis



Uses: Intensity Based Detection

- Looking for dark pixels…

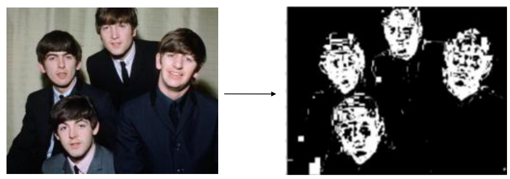


```
fg_pix = find(im < 65);
```

Slide Credit: Kristen Grauman

Uses: Color Based Detection

- Looking for pixels within a certain color range…

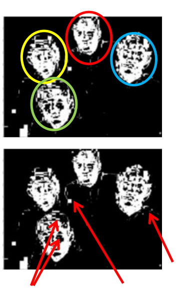


```
fg_pix = find(hue > t1 & hue < t2);
```

Slide Credit: Kristen Grauman

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



Slide Credit: Kristen Grauman

Outline of Today's Lecture

- Convert the image into binary form
  - Thresholding
- Clean up the thresholded image
  - Morphological operators
- Extract individual objects
  - Connected Components Labeling
- Describe the objects
  - Region properties

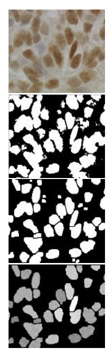
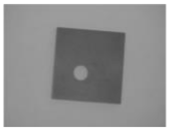


Image Source: D. Kim et al., Cytometry 35(1), 1999

Thresholding

- Grayscale image ⇒ Binary mask
- Different variants
  - One-sided
    $$F_T[i,j] = \begin{cases} 1, & \text{if } F[i,j] \geq T \\ 0, & \text{otherwise} \end{cases}$$
  - Two-sided
    $$F_T[i,j] = \begin{cases} 1, & \text{if } T_1 \leq F[i,j] \leq T_2 \\ 0, & \text{otherwise} \end{cases}$$
  - Set membership
    $$F_T[i,j] = \begin{cases} 1, & \text{if } F[i,j] \in Z \\ 0, & \text{otherwise} \end{cases}$$



↓

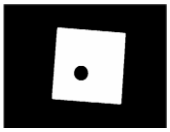
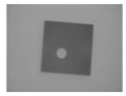


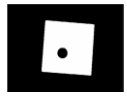
Image Source: http://homepages.inf.ed.ac.uk/rbf/HIPR2/

Selecting Thresholds

- Typical scenario
  - Separate an object from a distinct background

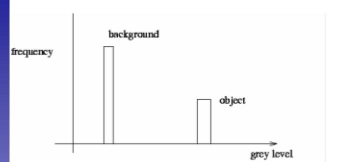


→

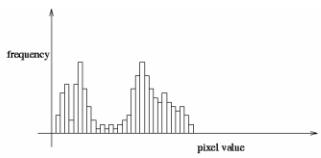


- Try to separate the different grayvalue distributions
  - Partition a bimodal histogram
  - Fit a parametric distribution (e.g. Mixture of Gaussians)
  - Dynamic or local thresholds
- In the following, I will present some simple methods.
  - We will then see some more general methods in Lecture 6…

A Nice Case: Bimodal Intensity Histograms



Ideal histogram, light object on dark background

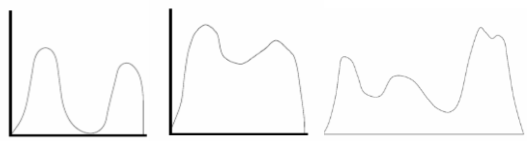


Actual observed histogram with noise

Source: Robyn Owens

Not so Nice Cases…

- How to separate those?



- Threshold selection is difficult in the general case
  - Domain knowledge often helps
  - E.g. Fraction of text on a document page (⇒ histogram quantile)
  - E.g. Size of objects/structure elements

Source: Shapiro & Stockman

Global Binarization [Otsu'79]

- Search for the threshold T that minimizes the within-class variance σ_{within} of the two classes separated by T
  $$\sigma^2_{within}(T) = n_1(T)\sigma_1^2 + n_2(T)\sigma_2^2(T)$$
  where
  $$n_1(T) = |\{I_{(x,y)} < T\}|, \quad n_2(T) = |\{I_{(x,y)} \geq T\}|$$
- This is the same as maximizing the between-class variance $\sigma_{between}$
  $$\begin{aligned} \sigma^2_{between}(T) &= \sigma^2 - \sigma^2_{within}(T) \\ &= n_1(T)n_2(T)\left[\mu_1(T) - \mu_2(T)\right]^2 \end{aligned}$$

Algorithm

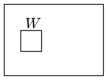
1. Precompute a cumulative grayvalue histogram h.
2. For each potential threshold T
   a) Separate the pixels into two clusters according to T
   b) Look up n_1, n_2 in h and compute both cluster means
   c) Compute $\sigma^2_{between}(T) = n_1(T)n_2(T)\left[\mu_1(T) - \mu_2(T)\right]^2$
3. Choose
   $$T^* = \arg\max_T \left[\sigma^2_{between}(T)\right]$$

→

Local Binarization [Niblack'86]

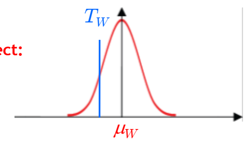
- Estimate a local threshold within a small neighborhood window W

$$T_W = \mu_W + k \cdot \sigma_W$$



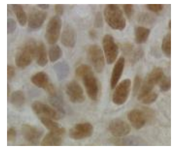
where $k \in [-1,0]$ is a user-defined parameter.

Effect:

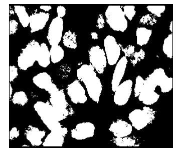


What is the hidden assumption here?

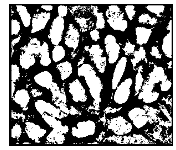
Effects



Original image



Global threshold selection (Otsu)

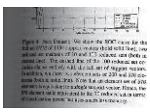


Local threshold selection (Niblack)

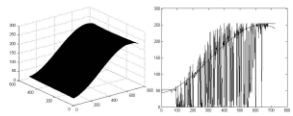
Additional Improvements

- Document images often contain a smooth gradient

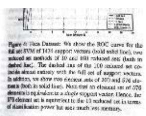
$\Rightarrow$ *Try to fit that gradient with a polynomial function*



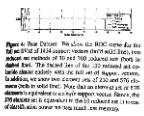
Original image



Fitted surface



Shading compensation



Binarized result

Source: S. Lu & C. Tan, ICDAR'07

Polynomial Surface Fitting

- Polynomial surface of degree d

$$f(x,y) = \sum_{i+j=0}^{d} b_{i,j} x^i y^j$$

- For an image pixel (x_0, y_0) with intensity I_0, this means

$$b_{0,0} + b_{1,0}x_0 + b_{0,1}y_0 + b_{2,0}x_0^2 + b_{1,1,}x_0y_0 + \cdots + b_{0,3}y_0^3 = I_0$$

- Least-squares estimation, e.g. for $d = 3$

$$Ab = I$$

$$\begin{bmatrix} 1 & x_0 & y_0 & x_0^2 & x_0y_0 & \cdots & y_0^3 \\ 1 & x_1 & y_1 & x_1^2 & x_1y_1 & \cdots & y_1^3 \\ \vdots & \vdots & \vdots & \vdots & \vdots & \ddots & \vdots \\ 1 & x_n & y_n & x_n^2 & x_ny_n & \cdots & y_n^3 \end{bmatrix} \begin{bmatrix} b_{0,0} \\ b_{1,0} \\ \vdots \\ b_{0,3} \end{bmatrix} = \begin{bmatrix} I_0 \\ I_1 \\ \vdots \\ I_n \end{bmatrix}$$

Solution with pseudo-inverse:

$$b = (A^T A)^{-1} A^T I$$

Matlab (using SVD):

$$b = I \setminus A$$

Surface Fitting

- Iterative Algorithm
  1.) Fit parametric surface to all points in region.
  2.) Subtract estimated surface.
  3.) Apply global threshold (*e.g.* with Otsu method)

  $\}$ *Initial guess*

  4.) Fit surface to all *background* pixels in original region.
  5.) Subtract estimated surface.
  6.) Apply global threshold (Otsu)

  $\}$ *Refined guess*

  7.) *Iterate further if needed…*

- The first pass also takes foreground pixels into account.
  - This is corrected in the following passes.
  - Basic assumption here: most pixels belong to the background.

Result Comparison

Original image

Global (Otsu)

Local (Niblack)

Polynomial + Global

Source: S. Lu & C. Tan, ICDAR'07

Outline of Today's Lecture

- Convert the image into binary form
  - Thresholding
- Clean up the thresholded image
  - Morphological operators
- Extract individual objects
  - Connected Components Labeling
- Describe the objects
  - Region properties

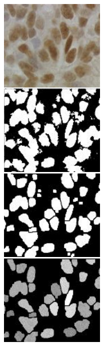
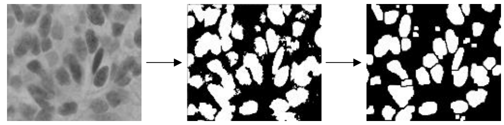


Image Source: D. Kim et al., Cytometry 35(1), 1999

Cleaning the Binarized Results

- Results of thresholding often still contain noise



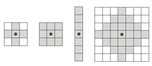
- Necessary cleaning operations
  - Remove isolated points and small structures
  - Fill holes

$\Rightarrow$ Morphological Operators

Image Source: D. Kim et al., Cytometry 35(1), 1999

Morphological Operators

- Basic idea
  - Scan the image with a structuring element
  - Perform set operations (intersection, union) of image content with structuring element



Matlab:
`>> help strel`

- Two basic operations
  - Dilation (Matlab: `imdilate`)
  - Erosion (Matlab: `imerode`)
- Several important combinations
  - Opening (Matlab: `imopen`)
  - Closing (Matlab: `imclose`)
  - Boundary extraction

Image Source: R.C. Gonzales & R.E. Woods

Dilation

- Definition
  - *"The dilation of A by B is the set of all displacements z, such that $(\hat{B})_z$ and A overlap by at least one element".*
  - $((\hat{B})_z$ is the mirrored version of B, shifted by $z)$
- Effects
  - If current pixel z is foreground, set all pixels under $(B)_z$ to foreground.

  $\Rightarrow$ Expand connected components
  $\Rightarrow$ Grow features
  $\Rightarrow$ Fill holes

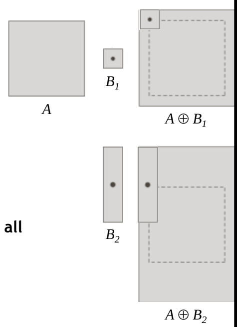


Image Source: R.C. Gonzales & R.E. Woods

Erosion

- Definition
  - *"The erosion of A by B is the set of all displacements z, such that $(B)_z$ is entirely contained in A".*
- Effects
  - If not every pixel under $(B)_z$ is foreground, set the current pixel z to background.

  $\Rightarrow$ Erode connected components
  $\Rightarrow$ Shrink features
  $\Rightarrow$ Remove bridges, branches, noise

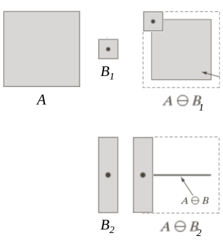
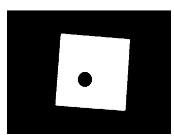
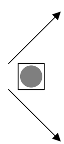


Image Source: R.C. Gonzales & R.E. Woods

Effects



Original image

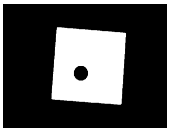


Dilation with circular structuring element

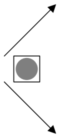
Erosion with circular structuring element

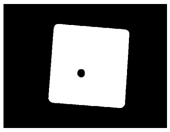
Image Source: http://homepages.inf.ed.ac.uk/rbf/HIPR2/

Effects

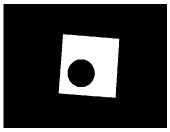


Original image





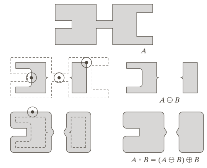
Dilation with circular structuring element



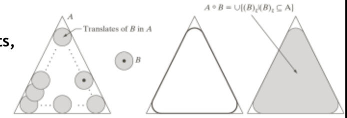
Erosion with circular structuring element

Opening

- Definition
  - Sequence of Erosion and Dilation
  - $A \circ B = (A \ominus B) \oplus B$

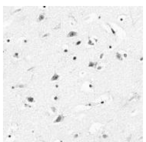


- Effect
  - $A \circ B$ is defined by the points that are reached if B is *rolled around inside* A.
  - ⇒ Remove small objects, keep original shape.

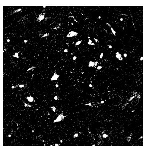


Effect of Opening

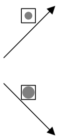
- Feature selection through *size* of structuring element



Original image



Thresholded



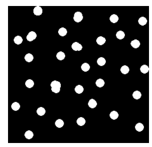
Opening with small structuring element

Opening with larger structuring element

Effect of Opening

- Feature selection through ***shape*** of structuring element

Input Image

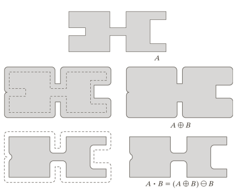


Opening with circular structuring element

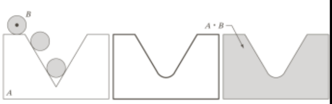
- *How could we have extracted the lines?*

Closing

- Definition
  - Sequence of Dilation and Erosion
  - $A \bullet B = (A \oplus B) \ominus B$



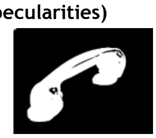
- Effect
  - $A \bullet B$ is defined by the points that are reached if B is *rolled around on the outside* of A.
  - ⇒ Fill holes, keep original shape.



Effect of Closing

- Fill holes in thresholded image (*e.g.* due to specularities)

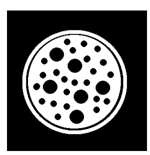
Original image

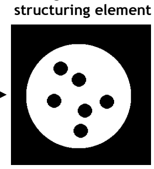


Thresholded

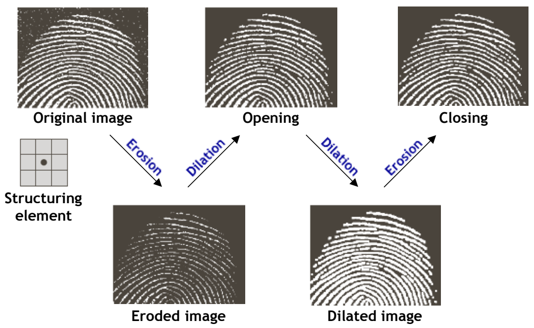
Closing with circular structuring element

Size of structuring element determines which structures are selectively filled.



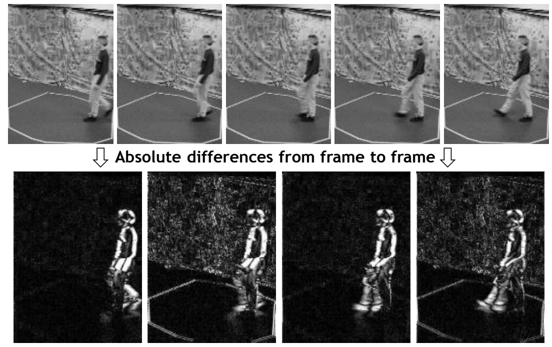


Example Application: Opening + Closing

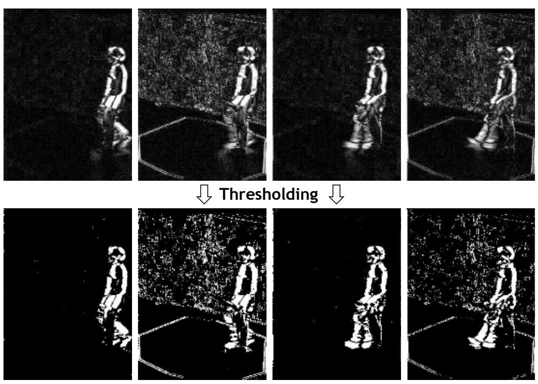


Source: R.C. Gonzales & R.E. Woods

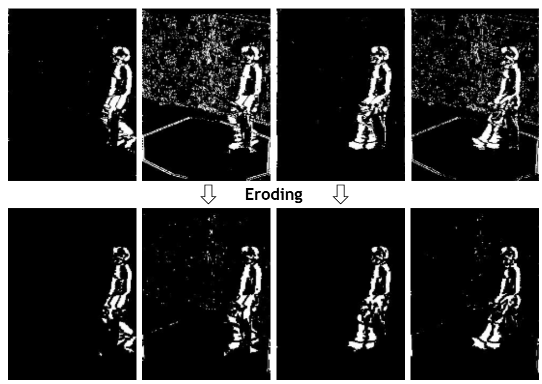
Application: Blob Tracking



Slide credit: K. Grauman



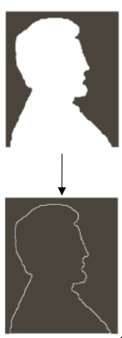
Slide credit: K. Grauman



Slide credit: K. Grauman

Morphological Boundary Extraction

- Definition
  - First erode A by B, then subtract the result from the original A.
    $\beta(A) = A - (A \ominus B)$
- Effects
  - If a 3×3 structuring element is used, this results in a boundary that is exactly 1 pixel thick.



Source: R.C. Gonzales & R.E. Woods

Morphology Operators on Grayscale Images

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

Original　Dilated　Eroded

Slide credit: Kristen Grauman

Outline of Today's Lecture

- Convert the image into binary form
  - Thresholding
- Clean up the thresholded image
  - Morphological operators
- **Extract individual objects**
  - **Connected Components Labeling**
- Describe the objects
  - Region properties

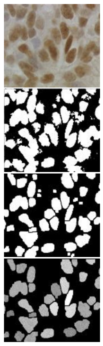
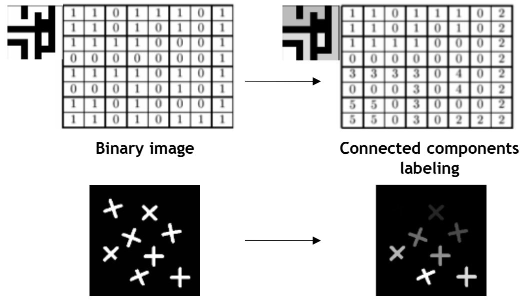


Image Source: D. Kim et al., Cytometry 35(1), 1999

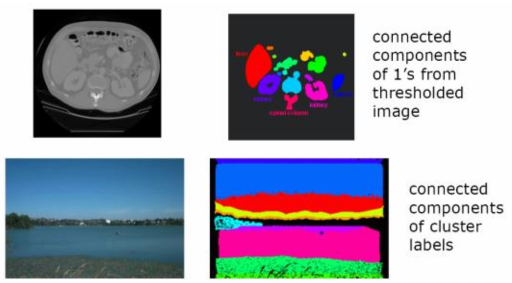
Connected Components Labeling

- Goal: Identify distinct regions



Sources: Shapiro & Stockman, Chandra

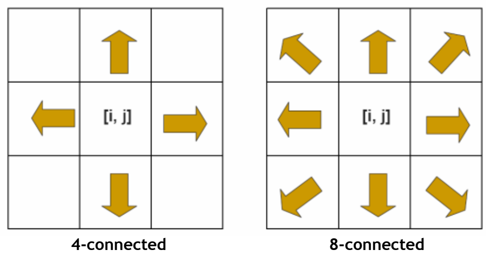
Connected Components Examples



Source: Pinar Duygulu

Connectedness

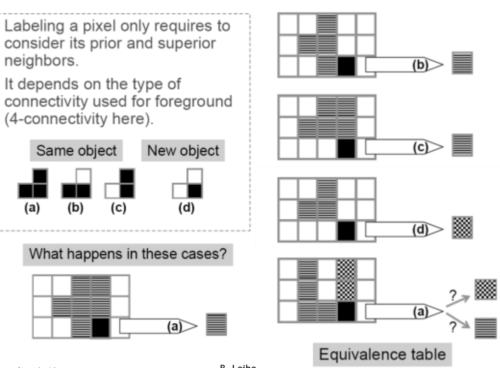
- Which pixels are considered neighbors?



Source: Chaitanya Chandra

Sequential Connected Components

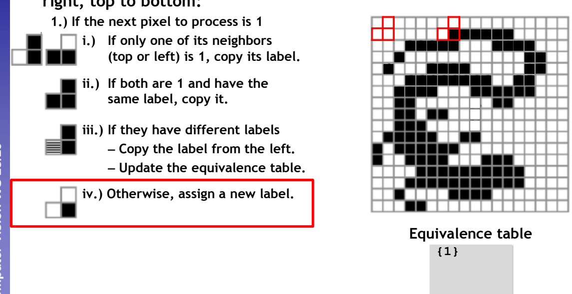
- Labeling a pixel only requires to consider its prior and superior neighbors.
- It depends on the type of connectivity used for foreground (4-connectivity here).



Slide credit: J. Neira

Sequential Connected Components (2)

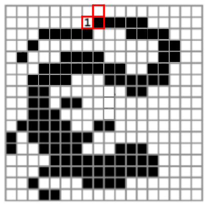
- Process the image from left to right, top to bottom:
  1.) If the next pixel to process is 1
  - i.) If only one of its neighbors (top or left) is 1, copy its label.
  - ii.) If both are 1 and have the same label, copy it.
  - iii.) If they have different labels
    - Copy the label from the left.
    - Update the equivalence table.
  - iv.) Otherwise, assign a new label.



Slide credit: J. Neira

Sequential Connected Components (2)

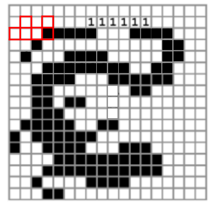
- Process the image from left to right, top to bottom:
  1.) If the next pixel to process is 1
  - i.) If only one of its neighbors (top or left) is 1, copy its label.
  - ii.) If both are 1 and have the same label, copy it.
  - iii.) If they have different labels
    – Copy the label from the left.
    – Update the equivalence table.
  - iv.) Otherwise, assign a new label.



Equivalence table
{1}

Sequential Connected Components (2)

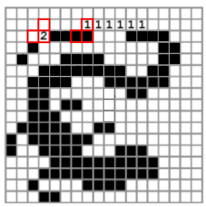
- Process the image from left to right, top to bottom:
  1.) If the next pixel to process is 1
  - i.) If only one of its neighbors (top or left) is 1, copy its label.
  - ii.) If both are 1 and have the same label, copy it.
  - iii.) If they have different labels
    – Copy the label from the left.
    – Update the equivalence table.
  - iv.) Otherwise, assign a new label.



Equivalence table
{1}
{2}

Sequential Connected Components (2)

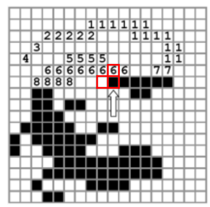
- Process the image from left to right, top to bottom:
  1.) If the next pixel to process is 1
  - i.) If only one of its neighbors (top or left) is 1, copy its label.
  - ii.) If both are 1 and have the same label, copy it.
  - iii.) If they have different labels
    – Copy the label from the left.
    – Update the equivalence table.
  - iv.) Otherwise, assign a new label.



Equivalence table
{1, 2}
{2}

Sequential Connected Components (2)

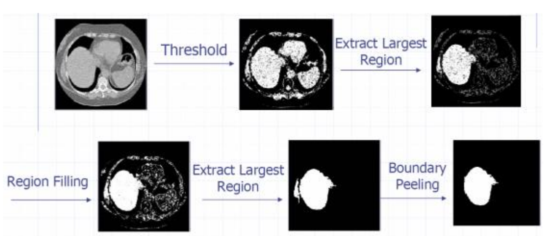
- Process the image from left to right, top to bottom:
  1.) If the next pixel to process is 1
  - i.) If only one of its neighbors (top or left) is 1, copy its label.
  - ii.) If both are 1 and have the same label, copy it.
  - iii.) If they have different labels
    – Copy the label from the left.
    – Update the equivalence table.
  - iv.) Otherwise, assign a new label.



Equivalence table
{1, 3, 4, 2, 7}
{5, 6, 8}
{}

- Re-label with the smallest of equivalent labels

Application: Segmentation of a Liver



Application by Jie Zhu, Cornell University

Outline of Today's Lecture

- Convert the image into binary form
  - Thresholding
- Clean up the thresholded image
  - Morphological operators
- Extract individual objects
  - Connected Components Labeling
- **Describe the objects**
  - Region properties

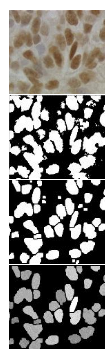


Image Source: D. Kim et al., Cytometry 35(1), 1999

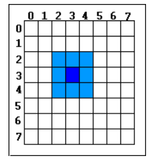
Region Properties

- From the previous steps, we can obtain separated objects.
- Some useful features can be extracted once we have connected components, including
  - Area
  - Centroid
  - Extremal points, bounding box
  - Circularity
  - Spatial moments

Area and Centroid

- We denote the set of pixels in a region by R
- Assuming square pixels, we obtain
  - *Area*: $A = \sum_{(x,y)\in R} 1$
  - *Centroid*: $\bar{x} = \frac{1}{A} \sum_{(x,y)\in R} x$

    $\bar{y} = \frac{1}{A} \sum_{(x,y)\in R} y$



Source: Shapiro & Stockman

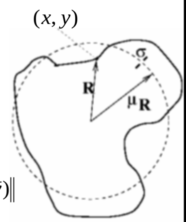
Circularity

- Measure the deviation from a perfect circle
  - *Circularity*: $C = \frac{\mu_R}{\sigma_R}$

    where μ_R and σ_R^2 are the mean and variance of the distance from the centroid of the shape to the boundary pixels (x_k, y_k).
  - *Mean radial distance*:

    $$\mu_R = \frac{1}{K} \sum_{k=0}^{K-1} \left\| (x_k, y_k) - (\bar{x}, \bar{y}) \right\|$$
  - *Variance of radial distance*:

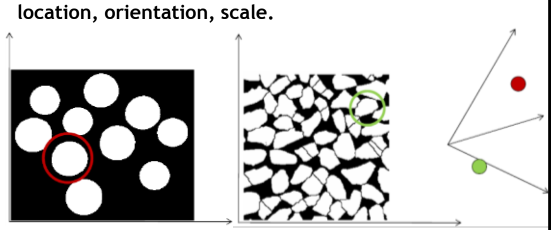
    $$\sigma_R^2 = \frac{1}{K} \sum_{k=0}^{K-1} \left[\left\| (x_k, y_k) - (\bar{x}, \bar{y}) \right\| - \mu_R \right]^2$$



Source: Shapiro & Stockman

Invariant Descriptors

- Often, we want features independent of location, orientation, scale.



Slide credit: Kristen Grauman

Central Moments

- S is a subset of pixels (region).
- Central $(j,k)^{th}$ moment defined as:

  $$\mu_{jk} = \sum_{(x,y)\in S} (x-\bar{x})^j (y-\bar{y})^k$$
- Invariant to translation of S.
- Interpretation:
  - 0th central moment: *area*
  - 2nd central moment: *variance*
  - 3rd central moment: *skewness*
  - 4th central moment: *kurtosis*

Slide credit: Kristen Grauman

Moment Invariants ("Hu Moments")

- Normalized central moments

  $$\eta_{pq} = \frac{\mu_{pq}}{\mu_{00}^{\gamma}}, \qquad \gamma = \frac{p+q}{2} + 1$$
- From those, a set of *invariant moments* can be defined for object description.

  $$\phi_1 = \eta_{20} + \eta_{02}$$
  $$\phi_2 = (\eta_{20} - \eta_{02})^2 + 4\eta_{11}^2$$
  $$\phi_3 = (\eta_{30} - 3\eta_{12})^2 + (3\eta_{21} - \eta_{03})^2$$
  $$\phi_4 = (\eta_{30} + \eta_{12})^2 + (\eta_{21} + \eta_{03})^2$$
- Robust to translation, rotation & scaling, but don't expect wonders (still summary statistics).

Moment Invariants

$$\phi_5 = (\eta_{30}-3\eta_{12})(\eta_{30}+\eta_{12})\left[(\eta_{30}+\eta_{12})^2-3(\eta_{21}+\eta_{03})^2\right] + (3\eta_{21}-\eta_{03})(\eta_{21}+\eta_{03})\left[3(\eta_{30}+\eta_{12})^2-(\eta_{21}+\eta_{03})^2\right]$$

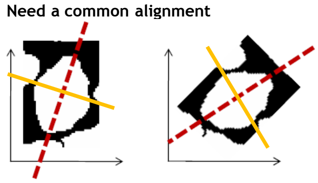
$$\phi_6 = (\eta_{20}-\eta_{02})\left[(\eta_{30}+\eta_{12})^2-(\eta_{21}+\eta_{03})^2\right] + 4\eta_{11}(\eta_{30}+\eta_{12})(\eta_{21}+\eta_{03})$$

$$\phi_7 = (3\eta_{21}-\eta_{03})(\eta_{30}+\eta_{12})\left[(\eta_{30}+\eta_{12})^2-3(\eta_{21}+\eta_{03})^2\right] + (3\eta_{12}-\eta_{30})(\eta_{21}+\eta_{03})\left[3(\eta_{30}+\eta_{12})^2-(\eta_{21}+\eta_{03})^2\right]$$

Often better to use $\log_{10}(\phi_i)$ instead of ϕ_i directly…

Axis of Least Second Moment

- Invariance to orientation?
  - Need a common alignment



Axis for which the squared distance to 2D object points is minimized (maximized).

  - Compute Eigenvectors of 2nd moment matrix (Matlab: eig(A))

$$\begin{bmatrix} \mu_{20} & \mu_{11} \\ \mu_{11} & \mu_{02} \end{bmatrix} = VDV^T = \begin{bmatrix} v_{11} & v_{12} \\ v_{22} & v_{22} \end{bmatrix} \begin{bmatrix} \lambda_1 & 0 \\ 0 & \lambda_2 \end{bmatrix} \begin{bmatrix} \boxed{\begin{matrix} v_{11} \\ v_{21} \end{matrix}} & \boxed{\begin{matrix} v_{12} \\ v_{22} \end{matrix}} \end{bmatrix}^T$$

Summary: Binary Image Processing

- Pros
  - Fast to compute, easy to store
  - Simple processing techniques
  - Can be very useful for constrained scenarios
- Cons
  - Hard to get "clean" silhouettes
  - Noise is common in realistic scenarios
  - Can be too coarse a representation
  - Cannot deal with 3D changes

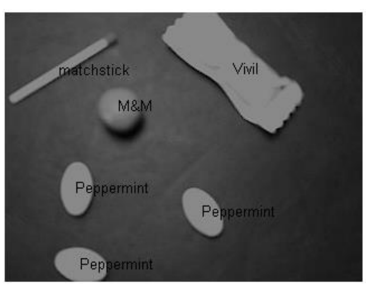
Slide credit: Kristen Grauman

References and Further Reading

- More on morphological operators can be found in
  - R.C. Gonzales, R.E. Woods, *Digital Image Processing*. Prentice Hall, 2001
- Online tutorial and Java demos available on
  - http://homepages.inf.ed.ac.uk/rbf/HIPR2/

Questions ?

Demo "Haribo Classification"



You Can Do It At Home…

Accessing a webcam in Matlab:

```
function out = webcam
% uses "Image Acquisition Toolbox„
adaptorName = 'winvideo';
vidFormat = 'I420_320x240';
vidObj1= videoinput(adaptorName, 1, vidFormat);
set(vidObj1, 'ReturnedColorSpace', 'rgb');
set(vidObj1, 'FramesPerTrigger', 1);
out = vidObj1 ;


cam = webcam();
img=getsnapshot(cam);
```

Questions ?