Computer Vision
Week 3

Binary Image Analysis
(Low-level operations)
Binarization: Thresholding
Enhancement
Connected Components
Feature Extraction

Binary Image Analysis

- Thresholding: Separate objects from background (binarization)
- Enhancement: Remove noise; improve appearance
- Identify individual objects: Connected Components Algorithms
- Compute features for each object

Example

- Find the turkeys in the picture

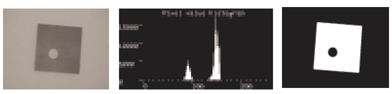
Thresholding

<u>Assumptions</u>:

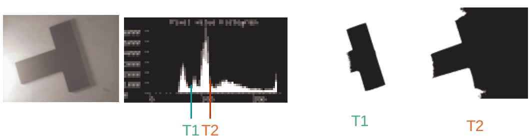
- Object region of interest has intensity distribution different from background
- Region pixel likely to be identified by intensity alone: intensity > a or intensity < b or a < intensity < b

Thresholding- Difficulties

- Works OK with flat-shaded scenes or engineered scenes.

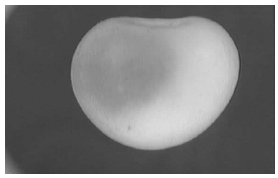


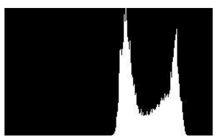
- Does not work well with nonuniform illumination.



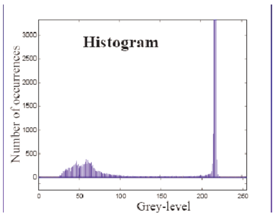
Cherry image shows 3 regions

- Background is black
- Healthy cherry is bright
- Bruise is medium dark
- Histogram shows two cherry regions (black background has been removed)

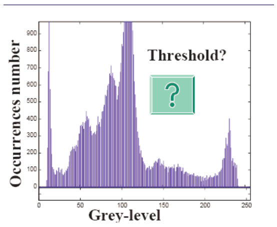




How easy is it?



How many objects?



Choosing a threshold

- Common to find the deepest valley between two modes of bimodal histogram
- Can fit two or more Gaussian curves to the histogram
- Can do optimization of some performance metric

Parametric vs. Non-parametric

Parametric techniques:
– Estimate parameters of two distributions from a given histogram.
– Difficult or impossible to establish a reliable model.

Non-parametric techniques:
– Separate the two gray level classes in an optimum manner according to a criterion:
  – between-class variance
  – Divergence
  – Entropy
  – conservation of moments

Non-parametric methods are more robust, and usually faster.

Automatic vs interactive

- Automatic means that the user does not have to specify any parameters.
- There are no truly automatic methods, always built-in parameters (but they can be estimated automatically).
- Distinction between automatic and interactive methods.
- Distinction between supervised (with training) and unsupervised (clustering).

Global or local (adaptive)

- Global methods use a single threshold for the entire image.
- Local methods optimize new thresholds for a number of sub-images or blocks.
- Global methods put severe restrictions on:
  – the gray level characteristics of objects and background
  – the uniformity in lighting and detection
- Non-contextual methods rely on the gray-level histogram of the image.
- Contextual methods make use of the geometrical relations between pixels

Bayesian decision-making

- classify into class ω_i that is most likely based on observations $\mathbf{X}$
- In order to compute the likelihoods given the measurement $\mathbf{X}$, the following distributions are needed.

class conditional distribution : $p(x|\omega_i)$ *for each class omega_i* (1)

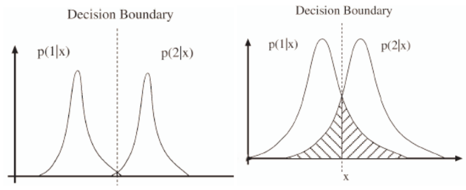
a priori probability : $P(\omega_i)$ *for each class* ω_i (2)

unconditional distribution : $p(x)$ (3)

- use Bayes rule if all of the classes ω_i are disjoint

$$P(\omega_i|x) = \frac{p(x|\omega_i)P(\omega_i)}{p(x)} = \frac{p(x|\omega_i)P(\omega_i)}{\Sigma_{i=1,m}\, p(x|\omega_i)P(\omega_i)} \qquad (4)$$

Some loss may be inevitable: the minimum risk (shaded area) is called the Bayes risk



Bi-level thresholding

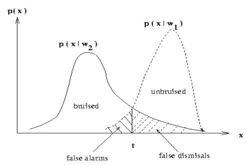
- Histogram is assumed to have two peaks. Let P_1 and P_2 be the a priori probabilities for background and foreground. Two distributions given by b(z) and f(z). The complete histogram is given by:
  $$p(z) = P_1 \cdot b(z) + P_2 \cdot f(z)$$
- The probabilities of misclassifying a pixel given a threshold t:
  $$E_1(t) = \int_{-\infty}^{t} f(z)dz$$
  $$E_2(t) = \int_{t}^{\infty} b(z)dz$$
- The total error is:
  $$E(t) = P_1 \cdot \int_{t}^{\infty} b(z)dz + P_2 \cdot \int_{-\infty}^{t} f(z)dz$$
- Differentiate with respect to the threshold t:
  $$\frac{\partial E}{\partial t} = 0 \Rightarrow \underline{P_1 \cdot b(T) = P_2 \cdot f(T)}$$

Parametric Models can be used

Parametric Models for Distributions

A **normal distribution** characterized by mean μ and standard deviation σ is defined as follows.

$$p(x) = N(\mu,\sigma)(x) = \frac{1}{\sqrt{2\pi}\sigma} exp[-\frac{1}{2}(\frac{x-\mu}{\sigma})^2]$$



Distributions for intensity measurement x conditioned on whether x is taken from an unbruised or bruised cherry.

Bi-level thresholding

- For Gaussian distributions
  $$\frac{P_1}{\sqrt{2\pi}\sigma_1} e^{-\frac{(T-\mu_1)^2}{2\sigma_1^2}} = \frac{P_2}{\sqrt{2\pi}\sigma_2} e^{-\frac{(T-\mu_2)^2}{2\sigma_2^2}}$$
- We get a quadratic equation:
  $$(\sigma_1^2 - \sigma_2^2)\cdot T^2$$
  $$+2(\mu_1\sigma_2^2 - \mu_2\sigma_1^2)\cdot T$$
  $$+\sigma_1^2\mu_2^2 - \sigma_2^2\mu_1^2 + 2\sigma_1^2\sigma_2^2 \ln\left(\frac{P_1\sigma_2}{P_2\sigma_1}\right) = 0$$
- If the two variances are equal
  $$\sigma_B^2 = \sigma_F^2 = \sigma^2$$
  $$T = \frac{(\mu_1+\mu_2)}{2} + \frac{\sigma^2}{(\mu_1-\mu_2)} \ln\left(\frac{P_2}{P_1}\right)$$
- If the *a priori* probabilities P_1 og P_2 are equal
  $$T = \frac{(\mu_1+\mu_2)}{2}$$

Kittler and Illingworth

- Kittler and Illingworth(1985) assume a mixture of two Gaussian distributions (five unknown parameters).
- Find T that minimizes the KL (Kullback-Leibler) distance between the observed histogram and model distribution.

$$J(t) = 1 + 2\left[P_1(t) ln\sigma_1(t) + P_2(t) ln\sigma_2(t)\right]$$
$$-2\left[P_1(t) lnP_1(t) + P_2(t) lnP_2(t)\right].$$

Kapur's entropy based method

- Maximize total entropy. Backgrund and object entropies are defined as:

$$H_t = -\sum_{z=0}^{t} p(z)ln(p(z))$$

$$H_G = -\sum_{z=0}^{G-1} p(z)ln(p(z))$$

- Maximize their sum:

$$\psi(t) = ln\,[P_1(t)(1-P_1(t))] + \frac{H_t}{P_1(t)} + \frac{H_G - H_t}{1-P_1(t)}.$$

- The value of T that maximizes this is Kapur threshold.

Otsu's Method (1979)

- Based on a very simple idea: Find the threshold that *minimizes the weighted within-class variance.*
- This turns out to be the same as *maximizing the between-class variance.*
- Operates directly on the gray level histogram [*e.g.* 256 numbers, P(i)], so it's fast (once the histogram is computed).

Automatic Thresholding: Otsu

- Assume two modes separated by t are present and minimize within group variance:

$$\sigma_W^2(t) = q_1(t)\sigma_1^2(t) + q_2(t)\sigma_2^2(t)$$

where q_i is the a priori probability and σ_i is the variance of group i

- This is equivalent to maximizing between group variance:

$$\sigma_B^2(t) = q_1(t)[1-q_1(t)][\mu_1(t) - \mu_2(t)]^2$$

The *weighted within-class variance* is:

$$\sigma_w^2(t) = q_1(t)\sigma_1^2(t) + q_2(t)\sigma_2^2(t)$$

Where the class probabilities are estimated as:

$$q_1(t) = \sum_{i=1}^{t} P(i) \qquad q_2(t) = \sum_{i=t+1}^{I} P(i)$$

And the class means are given by:

$$\mu_1(t) = \sum_{i=1}^{t} \frac{iP(i)}{q_1(t)} \qquad \mu_2(t) = \sum_{i=t+1}^{I} \frac{iP(i)}{q_2(t)}$$

Finally, the individual class variances are:

$$\sigma_1^2(t) = \sum_{i=1}^{t} [i-\mu_1(t)]^2 \frac{P(i)}{q_1(t)}$$

$$\sigma_2^2(t) = \sum_{i=t+1}^{I} [i-\mu_2(t)]^2 \frac{P(i)}{q_2(t)}$$

Now, we could actually stop here. All we need to do is just run through the full range of *t* values [1,256] and pick the value that minimizes $\sigma_w^2(t)$.

But the relationship between the within-class and between-class variances can be exploited to generate a recursion relation that permits a much faster calculation.

After some algebra, we can express the total variance as...

$$\sigma^2 = \underbrace{\sigma_w^2(t)}_{\text{Within-class, from before}} + \underbrace{q_1(t)[1-q_1(t)][\mu_1(t)-\mu_2(t)]^2}_{\text{Between-class, } \sigma_B^2(t)}$$

Since the total is constant and independent of *t*, the effect of changing the threshold is merely to move the contributions of the two terms back and forth.

So, *minimizing the within-class variance is the same as maximizing the between-class variance*.

The nice thing about this is that we can compute the quantities in $\sigma_B^2(t)$ *recursively* as we run through the range of *t* values.

Finally...

Initialization... $q_1(1) = P(1)\,;\ \mu_1(0) = 0$

Recursion...

$$q_1(t+1) = q_1(t) + P(t+1)$$

$$\mu_1(t+1) = \frac{q_1(t)\mu_1(t) + (t+1)P(t+1)}{q_1(t+1)}$$

$$\mu_2(t+1) = \frac{\mu - q_1(t+1)\mu_1(t+1)}{1 - q_1(t+1)}$$

Adaptive Thresholding

- All the methods presented so far are originally global methods.
- They can also be used locally in a subimage.
- A threshold is computed in a local window of size w centered around a pixel.
- The window is moved and a new threshold can be computed
  - – overlapping windows
  - – non-overlapping windows
- Local changes in background and foreground contrast is handled better than in global thresholding.

Adaptive Thresholding

- Bernsen's method: For each pixel (x,y)
  $t(x,y)=(Z_{low}+Z_{high})/2$
- Z_{low} is the smallest and Z_{high} is the highest pixel value in the local window of size r×r.

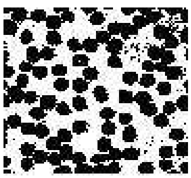
- Niblack's method: The threshold at pixel (x,y) is
  $t(x,y)=m(x,y)+k \cdot s(x,y)$
- m(x,y) is the mean and s(x,y) the standard deviation in the
- local window of size r×r.
- Parameters: contrast ratio k and window size r

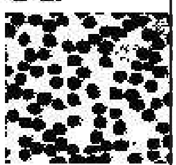
Example: Cleaning up thresholding results

- Can delete object pixels on boundary to better separate parts.
- Can fill small holes
- Can delete tiny objects

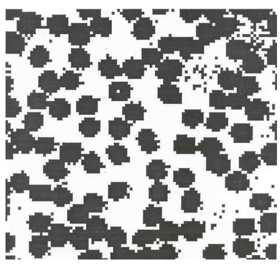
Removing salt-and-pepper

- Change pixels all of whose neighbors are different (coercion!): see hole filled at right
- Delete objects that are tiny relative to targets: see some islands removed at right





Example: red blood cell image



- Many blood cells are separate objects
- Many touch – bad!
- Salt and pepper noise from thresholding

Enhancement

- Linear Filtering: low-pass filter masks
- Nonlinear filtering: median, max, min, and other order-statistic operators
- Morphological filters

Linear filtering

- Convolution with a filter mask is a weighted sum of pixel intensities

5	5	5	5	5	5	5
5	5	5	5	5	5	5
5	5	5	5	5	5	5
0	0	0	0	5	5	5
0	5	0	0	0	5	0
5	0	0	5	0	0	0
0	0	0	0	0	0	0

Input

1/5 x

0	1	0
1	1	1
0	1	0

Mask
(Kernel)

Mean Filter

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

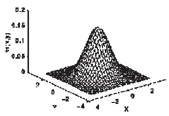
Output

Linear Filtering: Lowpass Filters

- Lowpass filters reduce noise,…
- But smoothes edges
- Mean filter is a LPF
- Gaussian Filter is a better LPF

LP Filters: Gaussian

$$G(x,y) = \frac{1}{2\pi\sigma^2} e^{-\frac{x^2+y^2}{2\sigma^2}}$$



Gaussian Filters are separable: you can filter rows then columns…

.006	.061	.242	.383	.242	.061	.006

or, apply 2D convolution:

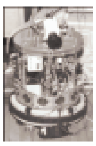
$\frac{1}{273}$

1	4	7	4	1
4	16	26	16	4
7	26	41	26	7
4	16	26	16	4
1	4	7	4	1

LP Filters: Gaussian

- Effects of sigma:

Original



sigma=1
5 x 5

sigma=2
9 x 9

sigma=4
15 x 15

Types of Noise

- Noise is often modeled as:
- Gaussian (mean, sigma)
- Salt-and pepper (0 or 255 with prob p)

Original

Gaussian
sigma=20
mean=0

Salt-and-pepper
p=0.3

Nonlinear Filters

- Linear Filters are good for removing gaussian noise, but not good for salt-and-pepper (impulsive) noise
- Median filter, alpha-trimmed-mean filter better for impulsive noise

Median, Max, Mean Filters

123	125	126	130	140
122	124	126	127	135
118	120	150	125	134
119	115	119	123	133
111	116	110	120	130

Neighbourhood values:

115, 119, 120, 123, 124, 125, 126, 127, 150

Median value: 124

Alpha-trimmed-mean

- Commonly used in figure skating: Remove the highest and lowest scores, then take average of the rest

123	125	126	130	140
122	124	126	127	135
118	120	150	125	134
119	115	119	123	133
111	116	110	120	130

Neighbourhood values:

115, 119, 120, 123, 124, 125, 126, 127, 150

Avg: 123.6

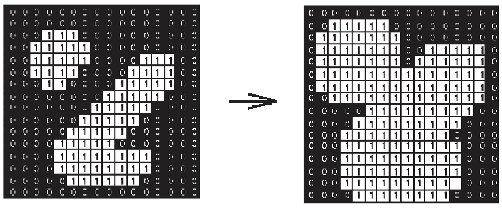
Median value: 124

Binary Morphology

- Treat an object within a binary image as the set of '1's: set A
- Instead of a kernel window, here, we have a "structuring element": set B
- Define the following operations based on set intersection, union, difference:
- Dilation: $A \oplus B = \cup \{ t \in I^2 : t = a+b, a \in A \}$
- Erosion: $A \theta B = \cap \{ t \in I^2 : t = a+b, a \in A \}$

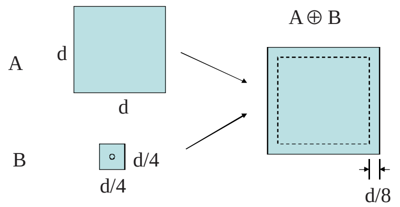
Dilation

- Dilation: $A \oplus B = \cup \{ t \in I^2 : t = a+b, a \in A \}$
- Example: 3 x 3 square structuring element



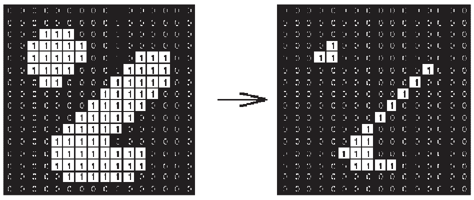
Dilation

-- Example of dilation



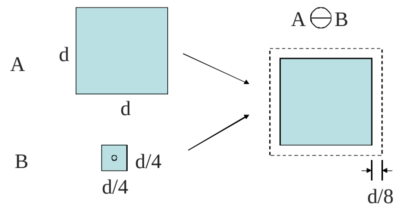
Erosion

- Erosion: $A \theta B = \cap \{ t \in I^2 : t = a+b, a \in A \}$
- Example: 3 x 3 square structuring element



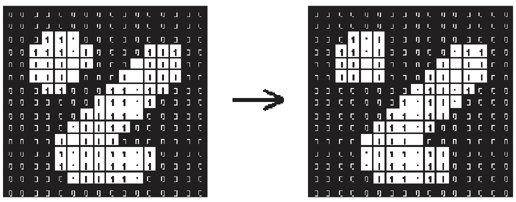
Erosion

-- Example of erosion



Opening

- Erosion followed by dilation



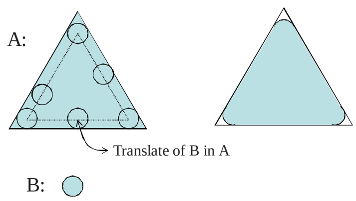
- Fit the structuring element inside the object

Opening

- Example

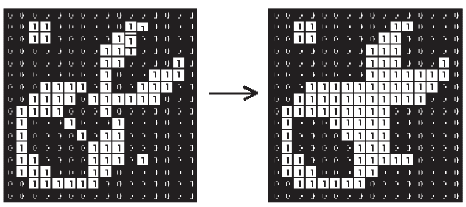
Opening:

$A \circ B = (A \ominus B) \oplus B$



Closing

- Dilation followed by erosion



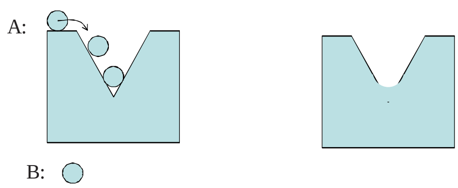
- Fit the structuring element in the background

Closing

- Example

Closing:

$A \bullet B = (A \oplus B) \ominus B$

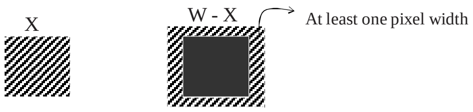


Other morphological operations

- Hit-or-miss transform
- Thinning
- Thickening
- Skeletonization

Hit-or-miss transformation

-- Shape detection
-- Using two structure elements



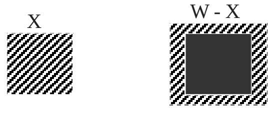
Structure element I

Structure element II:
complement of X with respect to W

Hit-or-miss transformation

-- The match (or fit) of B in A is called hit-or-miss transform, denoted $A \circledast B$
-- B is composed of X and W-X

$$A \circledast B = (A \ominus X) \cap [A^c \ominus (W\text{-}X)]$$



Hit-or-miss transformation

- General notation

structure element: $B = (B_1, B_2)$

e.g., $B_1 = X$ (object);
$B_2 = W\text{-}X$ (background)

$$A \circledast B = (A \ominus B_1) \cap [A^c \ominus B_2]$$

This set contains all the (origin) points, at which, B_1 found a match ("hit") in A and B_2 found a match in A^c, simultaneously.

Hit-or-miss transformation

- Hit-or-miss definition by set difference

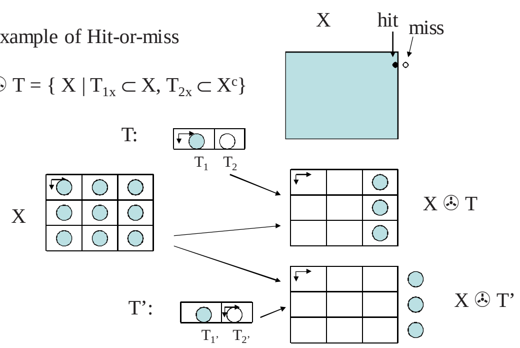
$$A \circledast B = (A \ominus B_1) - [A \oplus B_2\hat{}\,]$$

Note:
Hit-or-miss is the object match plus background match

Hit-or-miss transformation

- Example of Hit-or-miss

$X \circledast T = \{ X \mid T_{1x} \subset X, T_{2x} \subset X^c \}$



Other Applications

- Boundary extraction

  $$\text{Boundary}(A) = A - (A \ominus B)$$

- Region filling
  - given a set A which defines a region boundary
  - start with a non-boundary point P within the region
  - let $X_0 = P$
  - $X_k = (X_{k-1} \oplus B) \cap A^c, \quad k = 1,2,3,\ldots$
  - iteration at each step k
  - terminate if $X_k = X_{k-1}$

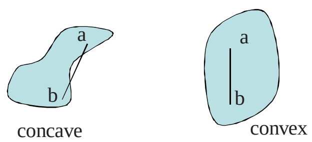
Note: $A \cup X_k$ includes the filled set and the boundary

Other Applications - CC

- Connected component extraction
  - similar to the region filling
  - start with a point P which is contained in A
  - let $X_0 = P$
  - $X_k = (X_{k-1} \oplus B) \cap A, \quad k = 1,2,3,\ldots$
  - iteration at each step k
  - terminate if $X_k = X_{k-1}$

Other Apps – Convex Hull

- Convex hull extraction
  - set A is convex if any line $ab \subset A$ $(a \in A,\ b \in A)$



  - convex hull H if an arbitrary set S is the smallest convex set which contains S

Other Apps – Convex Hull

- Convex hull extraction (cont'd)
  - Example of detection of convex hull of set A

    Given a set A and four structure elements B^1, B^2, B^3, B^4
    calculate the convex hull region: $C(A) = D^1 \cup D^2 \cup D^3 \cup D^4$

    where:
    D^i is derived from: $X^i_k = (X^i_{k-1} \circledast B^i) \cup A$
    $(i=1,2,3,4), (k=1,2,\ldots)$

    $D^i = X^i_k$ when $X^i_k = X^i_{k-1}$

    Initial $X^i_0 = A$

Other Applications - Thinning

- Thinning
  - "peel" from "outside" into "inside", which is defined in terms of the hit-or-miss transform:

    $A \otimes B = A - (A \circledast B)$
    $B = \{B^1, B^2, \ldots, B^n\}$
    $A \circledast \{B\} = (((A \circledast B^1) \circledast B^2)\ldots) \circledast B^n)$

Other Applications - Thickening

- Thickening

  - The structure element B is similar to the structure element for thinning, except that 1's and 0's are exchanged.
  - morphological dual of thinning

    $A \odot B = A \cup (A \circledast B)$
    $B = \{B^1, B^2, \ldots, B^n\}$

  - Alternative algorithm:
    To thicken a set A, we can also
    - apply "thinning" algorithm on A^c,
    - obtain region R
    - then take R^c as the thickening result

Other Applications - Skeletons

- Skeletons
  - skeletons can be implemented by the operations of erosions and openings

$S(A) = \cup_{k=0}^{K}(S_k(A))$

$S_k(A) = (A \ominus kB) - (A \ominus kB) \circ B$

$A \ominus kB = (((A \ominus B) \ominus B)\ldots) \ominus B)$

$K = \max\{k \mid (A \ominus kB) \neq \varnothing\}$

Other Applications - Pruning

- Pruning
  - it is complement to thinning and skeletonizing algorithm
  - example: hand-writing recognition

Connected Components: Object coloring

- Each object is a connected set of pixels
- Object label is "color"
- How is this done?

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

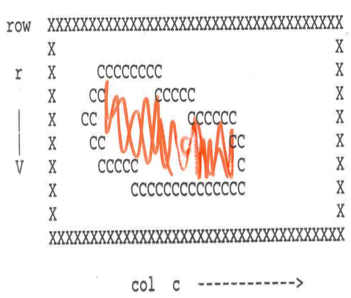
b) 5 components

Extracting components

- Collect foreground pixels into separate objects – label pixels with same color
- Can then compute many features from each set of pixels
- A) collect by "random access" of pixels using "paint" or "fill" algorithm
- B) collect by "raster" (row-by-row) scanning all pixels

paint/fill algorithm

- Obj region must be bounded by background
- Start at any pixel [r,c] inside obj
- Recursively color neighbors



Connected Components Labeling

- Recursive algorithm
  - For each object pixel {assign label L
    recursively assign label L to all neighbors
    stop if there are no more unlabeled 1's}
- Sequential algorithm
  - Scan the image top to right, top to bottom
  - For each pixel that is 1 {
    if only one of u or l has a label, copy it
    if both have the same label, copy it
    if different, copy l and enter labels as equivalent
    otherwise, assign a new label
  - Find the lowest label for each equivalence set; relabel entries

Merging connecting regions

Detect and record merges while raster scanning.

Use equiv. Table to recode

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

1	1	0	2	2	2	0	3
1	1	0	2	0	2	0	3
1	1	1	1	0	0	0	3
0	0	0	0	0	0	0	3
4	4	4	4	0	5	0	3
0	0	0	4	0	5	0	3
6	6	0	4	0	0	0	3
6	6	0	4	0	7	7	3

1	1	0	1	1	1	0	3
1	1	0	1	0	1	0	3
1	1	1	1	0	0	0	3
0	0	0	0	0	0	0	3
4	4	4	4	0	5	0	3
0	0	0	4	0	5	0	3
6	6	0	4	0	0	0	3
6	6	0	4	0	3	3	3

PARENT

1	2	3	4	5	6	7
0	1	0	0	0	0	3

2 → 1   7 → 3

Example revisited

- Find the turkeys in the picture

Binary features

- Shape descriptors: features that describe a shape specified as a binary mask
- Boundary descriptors: We will talk about boundary descriptors later, after we discuss boundary extraction

Shape descriptors

- Simple geometric features
- Projections
- Moments

- How good is a feature? It should be:
  - Translation invariant
  - Scale invariant
  - Rotation invariant

Simple Geometric Features

- Area: A=ΣΣB[i,j]
- Center of mass: $\bar{x} = \frac{\sum\sum jB[i,j]}{A}$ $\quad \bar{y} = \frac{\sum\sum iB[i,j]}{A}$

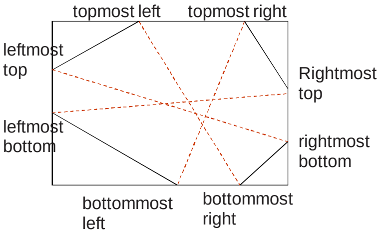
- Perimeter length
- Circularity: $C = \frac{|P|^2}{\mathrm{A}}$

- Better measure for circularity (Haralick):

$$C_2 = \frac{\text{mean radial distance}}{\text{std. dev. of mean radial distance}}$$

Simple Geometric Features

- Bounding box and extremal points



4 extremal axes
maximal axis
minimal axis
elongation: |max axis| / |min axis|

Projections

- Horizontal projection: $H[i]=\sum_{j=1}^{m} B[i,j]$
- Vertical projection $V[i]=\sum_{i=1}^{m} B[i,j]$

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

Projections at arbitrary directions may also be defined

Moments

- Central moments: $\mu_{jk}=\sum\sum(x-\bar{x})^j(y-\bar{y})^k B[x,y]$

  $\mu_{10}=\mu_{01}=0$
- Central moments are translation invariant; but not rotation and scale invariant
- Normalized central moments: $\eta_{jk}=\dfrac{\mu_{jk}}{\mu_{00}^{\gamma}}$ $\quad \gamma=\dfrac{j+k}{2}+1$
- Normalized central moments are scale and translation invariant

Interpretation of central moments

Central moment	interpretation
μ_{20}	horizontal centralness
μ_{02}	vertical centralness
μ_{11}	diagonality
μ_{12}	horizontal divergence (relative extent of right wrt left)
μ_{21}	vertical divergence (relative extent of bottom wrt top)
μ_{30}	horizontal imbalance
μ_{03}	Vertical imbalance

Moment Invariants

$$\phi_1=\eta_{20}+\eta_{02}$$
$$\phi_2=(\eta_{20}-\eta_{02})^2+4\eta_{11}^2$$
$$\phi_3=(\eta_{30}-3\eta_{12})^2+(3\eta_{21}-\eta_{03})^2$$
$$\phi_4=(\eta_{30}+\eta_{12})^2+(\eta_{21}+\eta_{03})^2$$
$$\phi_5=(\eta_{30}-3\eta_{12})(\eta_{30}+\eta_{12})[(\eta_{30}+\eta_{12})^2-3(\eta_{21}+\eta_{03})^2] +(3\eta_{21}-\eta_{03})(\eta_{21}+\eta_{03})[3(\eta_{30}+\eta_{12})^2-(\eta_{21}+\eta_{03})^2]$$
$$\phi_6=(\eta_{20}-\eta_{02})[(\eta_{30}+\eta_{12})^2-(\eta_{21}+\eta_{03})^2] +4\eta_{11}(\eta_{30}+\eta_{12})(\eta_{21}+\eta_{03})$$
$$\phi_7=(3\eta_{21}-\eta_{03})(\eta_{30}+\eta_{12})[(\eta_{30}+\eta_{12})^2-3(\eta_{21}+\eta_{03})^2] -(\eta_{30}-3\eta_{12})(\eta_{21}+\eta_{03})[3(\eta_{30}+\eta_{12})^2-(\eta_{21}+\eta_{03})^2]$$

Project I – Binarization & Connected Components

- due: November 5th

Task: Threshold a given gray-scale image to produce a binary image. Use the Otsu threshold first. Then, use another adaptive thresholding method from literature. For a good survey of thresholding methods, read Trier and Jain, 1995. This paper lists several adaptive thresholding techniques, such as Bernsen's method, Chow and Kaneko's method, Eikvil, Taxt and Moen's method, Mardia and Hainsworth's method, Niblack's method, Taxt, Jain and Flynn's method, Yanowitz and Bruckstein's method, and White and Rohrer's method. Read and implement one. Describe the method in your report.

Display the intermediate image. Then, apply connected components labeling to the binary image. Also display the output image.

Input: The file letters.bmp

Output:

- Thresholding: Your output pixels are 1's and 0's. However, 0 and 1 correspond to very close colors on the palette, so you will not be able to see a difference. Replace the 1's by 255. Call the two files otsu.bmp and thresh.bmp
- Connected components: To display the result of the connected component algorithm, show each connected component with a different color. You can accomplish creating color images by changing the palette. Call the two files compotsu.bmp and compthresh.bmp.