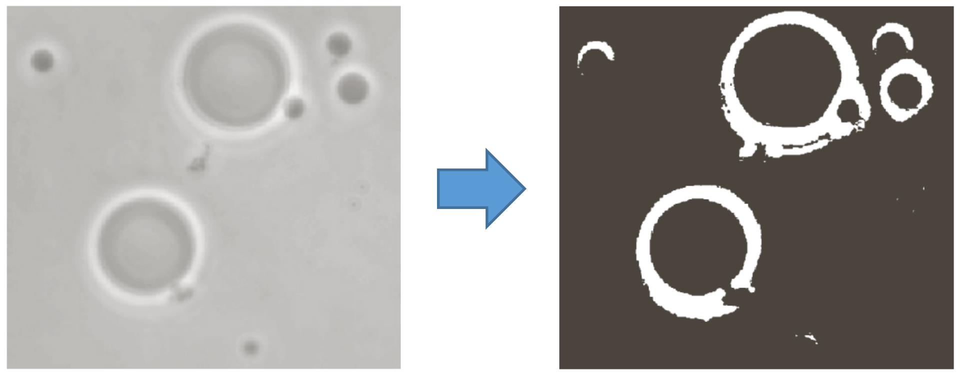
Image Segmentation

Sung Soo Hwang

Introduction

- What is image/video segmentation?
  - Process of partitioning a digital image into multiple regions
  - Application
    - Object classification



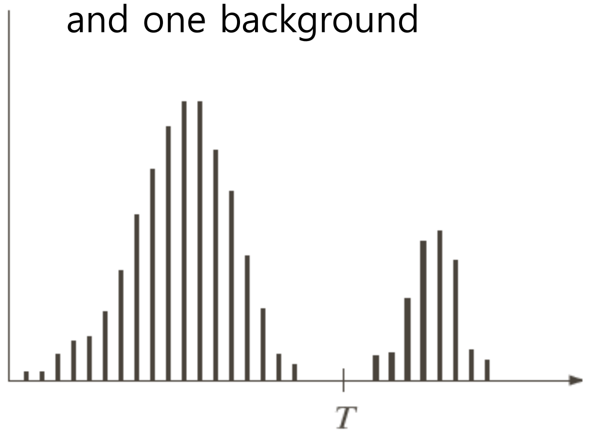
Introduction

- What is image/video segmentation?
  - Input: gray-scale image or color image
  - Output(for two-class problems): binary image (images with 0 and 255 (or 0 and 1) only)

Thresholding

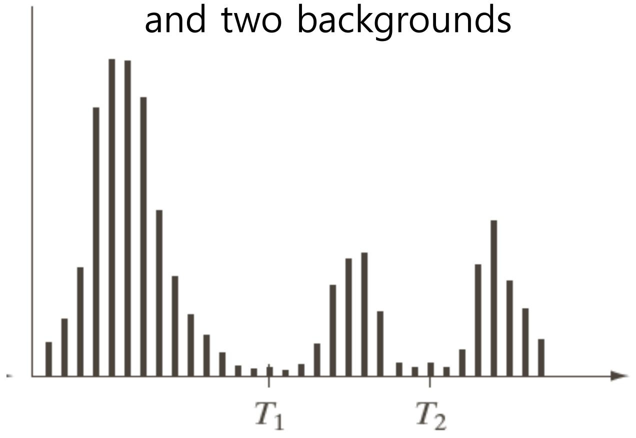
- Basic concepts
  - Assumption
    - Intensity of background and object is different
    - Background and object are homogenous

Images with one object and one background



$$g(x,y) = \begin{cases} 1 & if\ f(x,y) > T \\ 0 & otherwise \end{cases}$$

Images with one object and two backgrounds

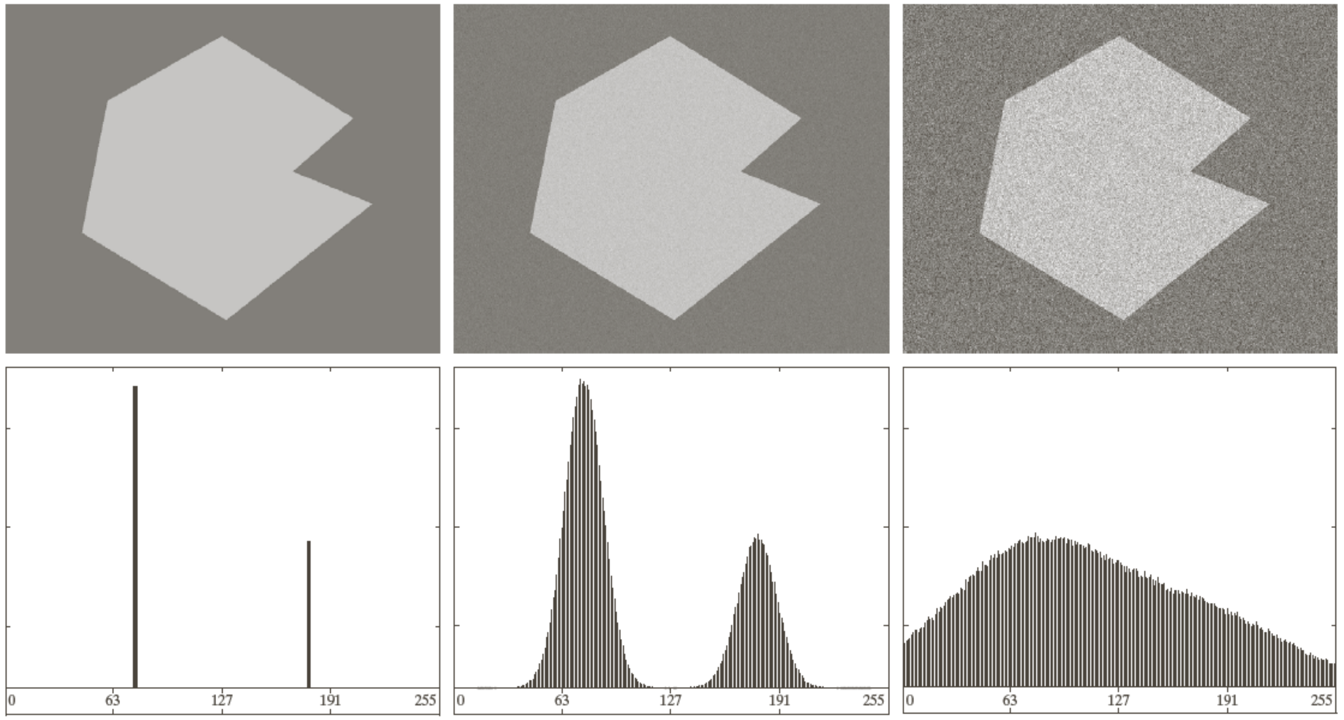


$$g(x,y) = \begin{cases} a & if\ f(x,y) > T_2 \\ b & if\ T_1 < f(x,y) \le T_2 \\ c & otherwise \end{cases}$$

- Finding the proper threshold is important

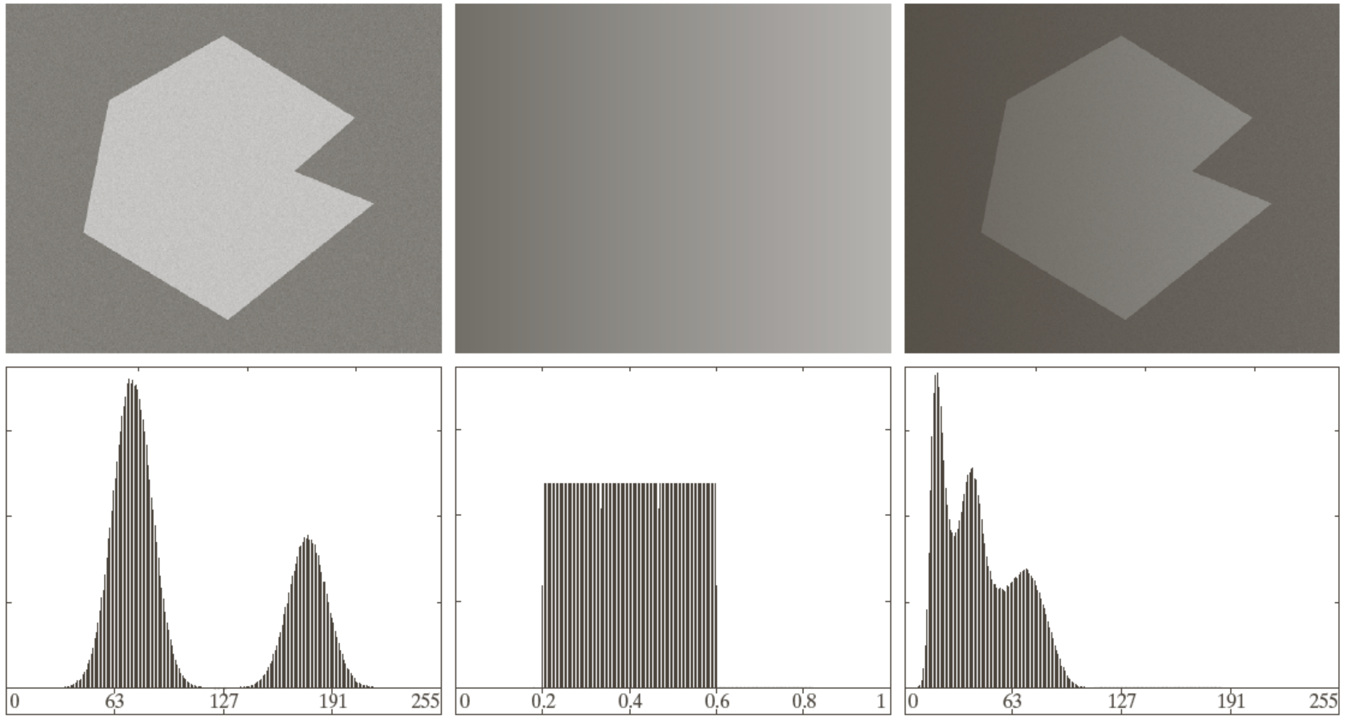
Thresholding

- Challenges
  - Noise



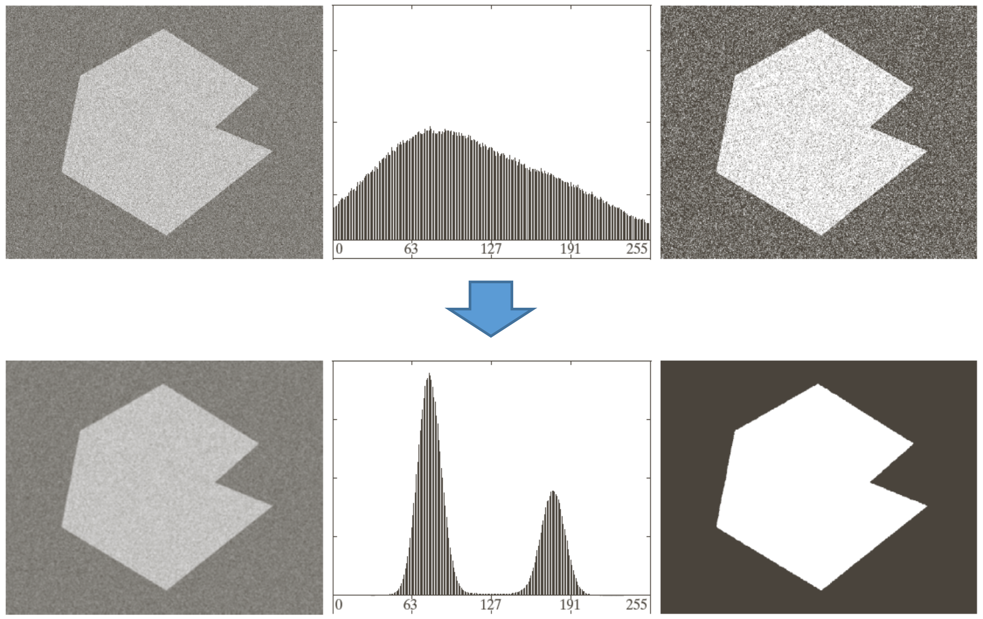
Thresholding

- Challenges
  - Illumination and reflectance



Thresholding

- Thresholding after applying smoothing



- By applying smoothing before thresholding, we may obtain the better result

Thresholding

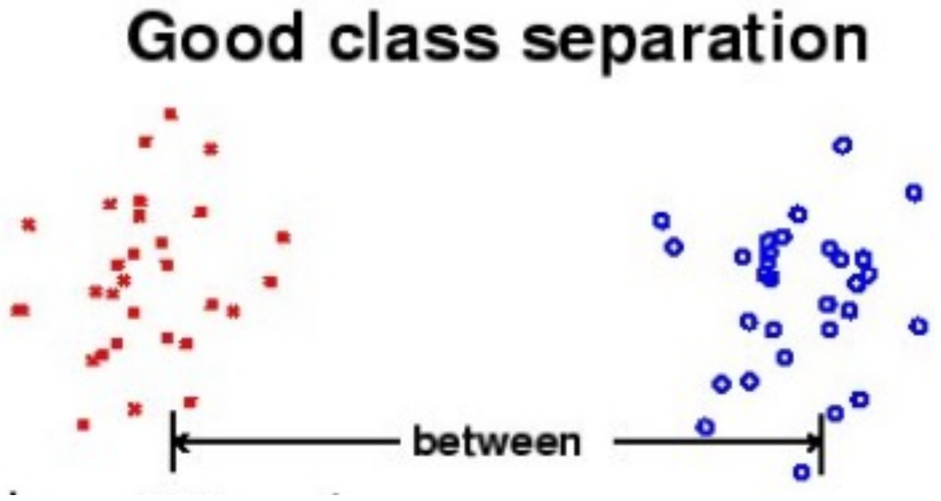
- Global thresholding
  - Use same threshold for every pixel
- Local (adaptive) thresholding
  - Use different threshold for each pixel

Global Thresholding

- Basic method
  1. Select an initial estimate for the global threshold T
  2. Segment the image using T into two groups
  3. compute the mean(m1,m2) for each group
  4. compute new threshold as T=0.5X(m1+m2)
  5. repeat step 2 through 4 until the difference between values of T in successive iterations is small

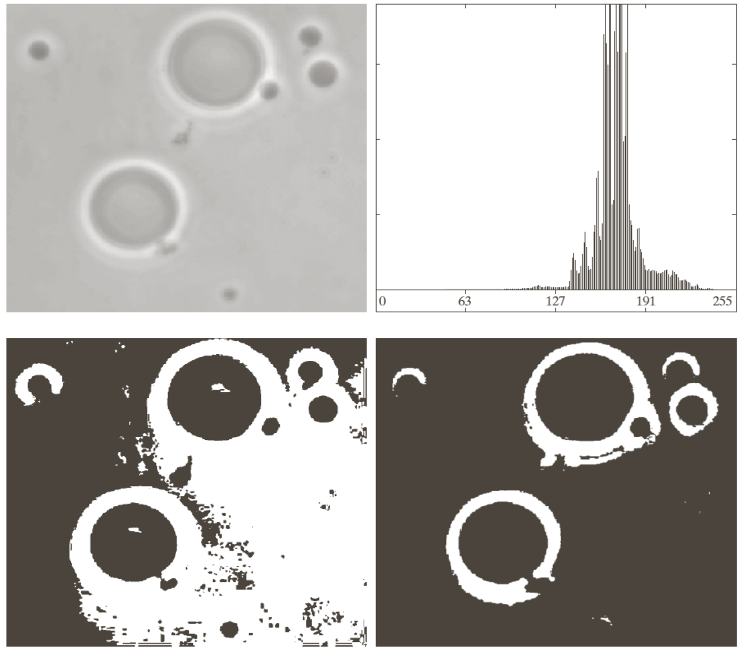
Global Thresholding

- Otsu’s method
  - Concept
    - Well-thresholded classes should be distinct with respect to the intensity values of their pixels
    - Conversely, a threshold giving the best separation between classes would be the best threshold
    - It is based on computations performed on the histogram of an image



Global Thresholding

- Otsu's method
  1. Compute the normalized histogram
  2. For each threshold k, compute between-class variance $\sigma^2{}_B$
  3. Obtain the Otsu threshold k for which $\sigma^2{}_B$ is maximized



Local(Adaptive) Thresholding

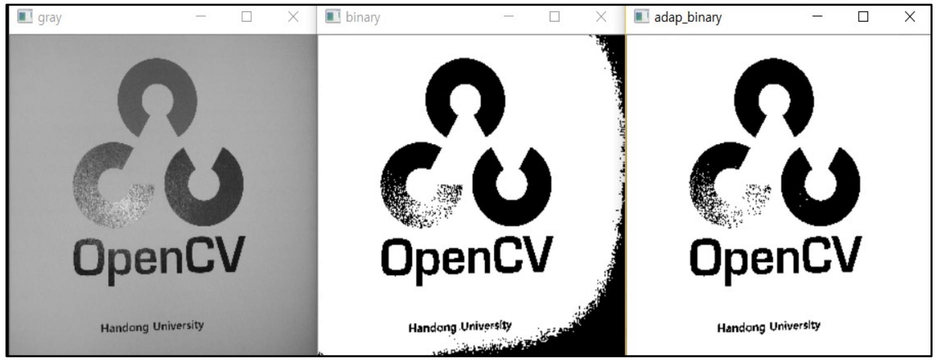
- Set a threshold for each point depending on the intensity distributions of adjacent pixels

ADAPTIVE_THRESH_MEAN_C :

$$T(x,y) = mean\ of\ the\ blocksize \times blocksize\ neighborhood\ of\ (x,y) - C$$

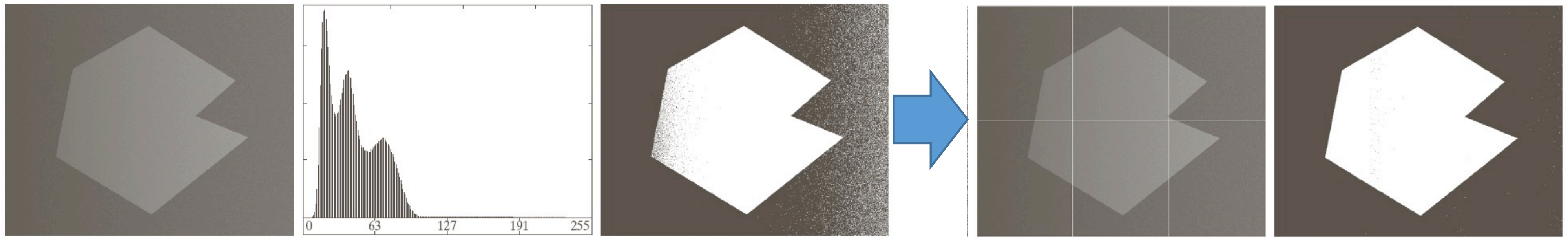
ADAPTIVE_THRESH_GAUSSIAN_C :

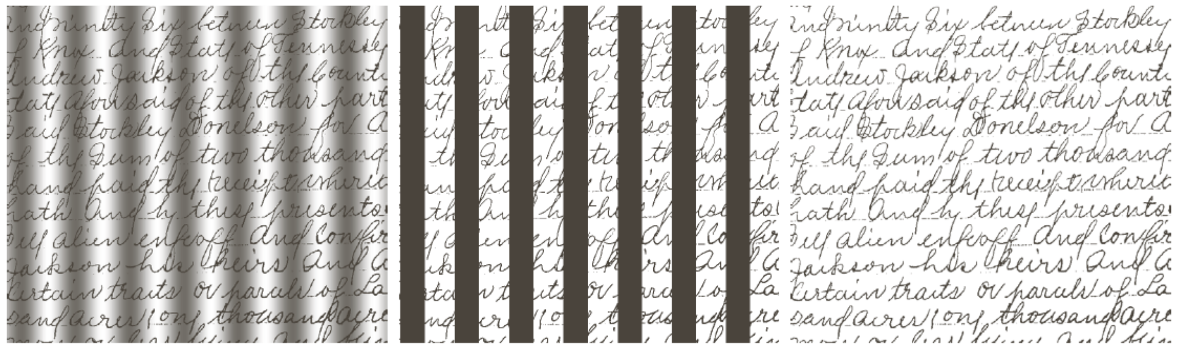
$$T(x,y) = a\ weighted\ sum(cross - correlation\ with\ a\ Gaussian\ windonw)\ of\ the\ blocksize \times blocksize\ neighborhood\ of\ (x,y) - C$$



Local(Adaptive) Thresholding

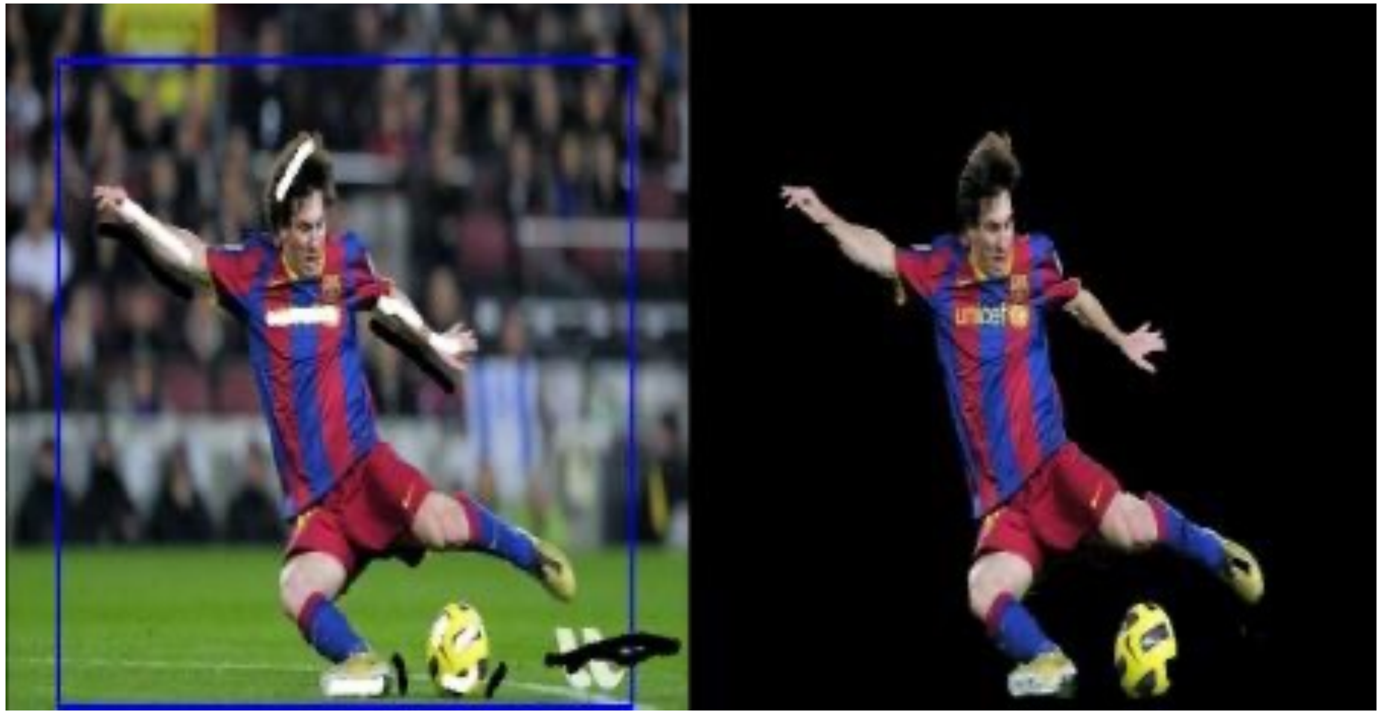
- Set a threshold for each point depending on the intensity distributions of adjacent pixels
  - Image partitioning





GrabCut

- GrabCut operation



GrabCut

- GrabCut operation

    1. User inputs the rectangle. Everything outside this rectangle will be taken as sure background. Everything inside rectangle is unknown.

    2. Computer does an initial labelling depending on the data we gave. It labels the foreground and background pixels (or it hard-labels)

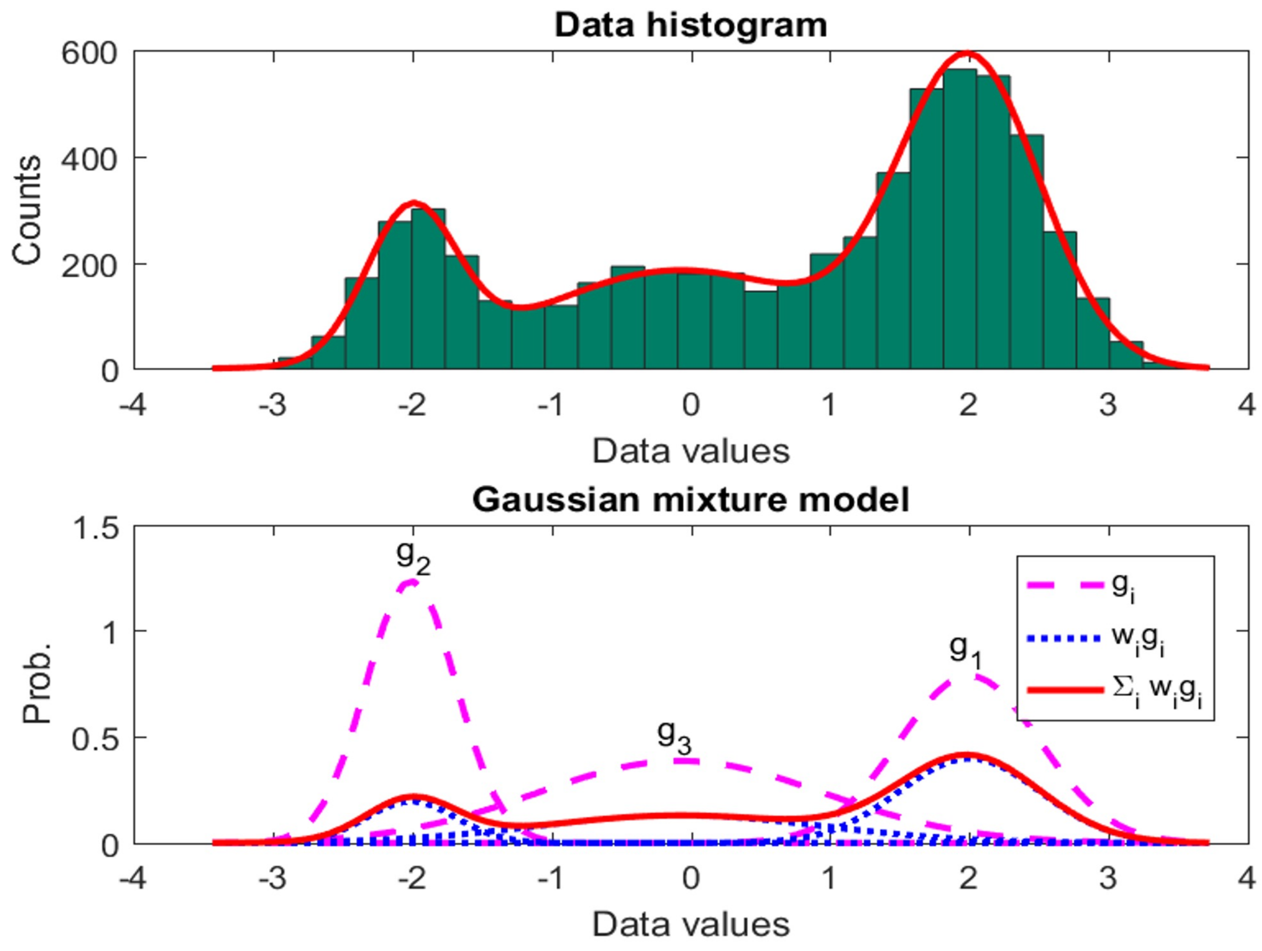
    3. Now a Gaussian Mixture Model(GMM) is used to model the foreground and background.

GrabCut

- GMM
  - Conditional probabilities
    - $p(A \cap B) = p(A|B)p(B) = p(B|A)p(A)$
  - Bayes rule
    - $p(B|A) = \frac{p(A \cap B)}{p(A)} = \frac{p(A|B)p(B)}{p(A)}$
  - Assume A is pixel value, and B is background
    - If we can figure out(or estimate) $p(A|B)$, then we can find out the probability of a certain pixel value being background → $p(B|A)$

GrabCut

- GMM



Data histogram
Counts
600
400
200
0
-4
-3
-2
-1
0
1
2
3
4
Data values
Gaussian mixture model
Prob.
1.5
1
0.5
0
g_2
g_3
g_1
g_i
$w_i g_i$
$\Sigma_i w_i g_i$
Data values

GrabCut

- Background estimation using GMM
  - Determine the number of mode of GMM
  - At the training stage, estimate mean and variance of each Gaussian model with the training data

    →estimate $p(A|B)$

    Background image is totally white

    P(255|Background) = 1

    P(0|Background) = 0
  - Each pixel is classified into background/foreground by calculating$p(B|A)$
    - P(background|255) = high
    - P(background|0) = low
    - P(background|128) = half

GrabCut

- GrabCut operation

  4. Depending on the data we gave, GMM learns and create new pixel distribution. That is, the unknown pixels are labelled either probable foreground or probable background.

  5. A graph is built from this pixel distribution. Nodes in the graphs are pixels. Additional two nodes are added, Source node and Sink node. Every foreground pixel is connected to Source node and every background pixel is connected to Sink node.

GrabCut

- GrabCut operation

  6. The weights of edges connecting pixels to source node/end node are defined by the probability of a pixel being foreground/background.

  7. The weights between the pixels are defined by the edge information or pixel similarity. If there is a large difference in pixel color, the edge between them will get a low weight.

  8. A mincut algorithm is used to segment the graph. It cuts the graph into two separating source node and sink node with minimum cost function. The cost function is the sum of all weights of the edges that are cut.

GrabCut

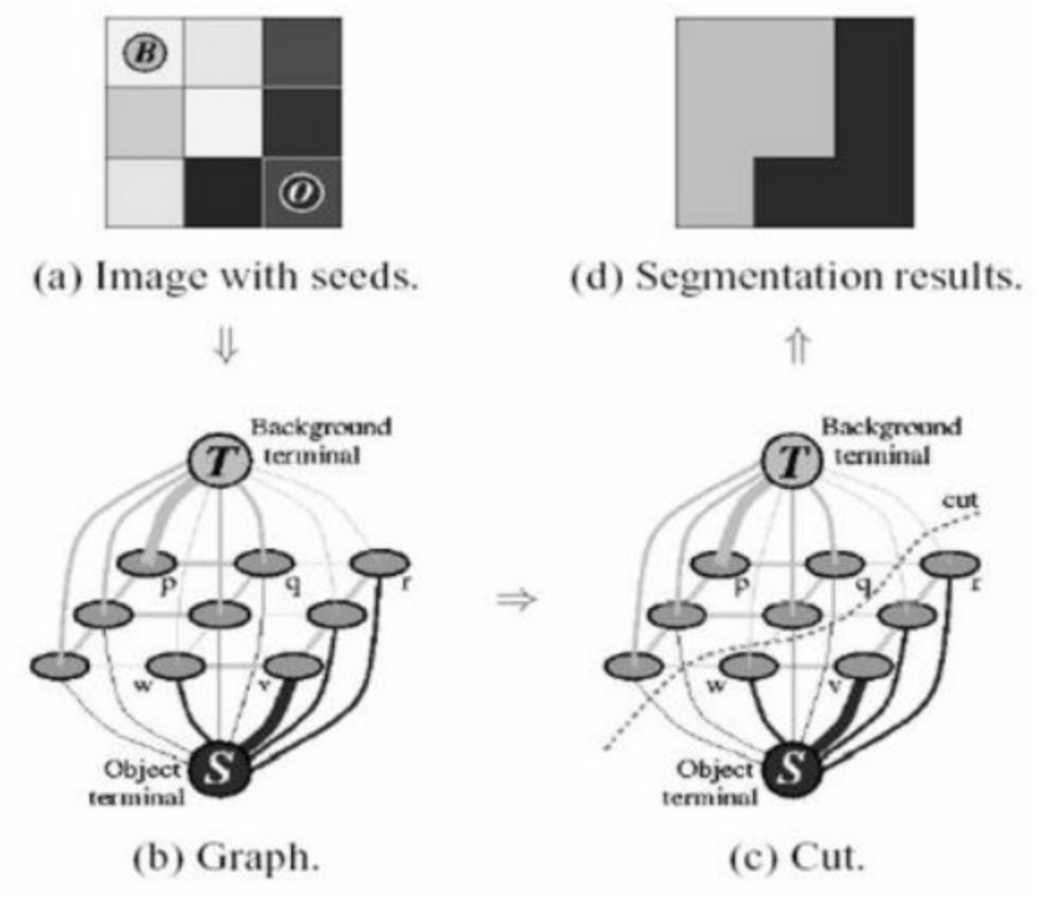
- GrabCut operation

  9. After the cut, all the pixels connected to Source node become foreground and those connected to Sink node become background.

  10. The process is continued until the classification converges.

GrabCut

- GrabCut operation


B
O
(a) Image with seeds.
(d) Segmentation results.
Background terminal
T
p
q
r
w
v
Object terminal
S
(b) Graph.
Background terminal
cut
(c) Cut.