

# Graph-based Segmentation

基于图论的图像分割

戴嘉伦 王如晨 赵海伟

CVBIOUC

May 19, 2015

# 目录

① Graph

② Normalized Cuts

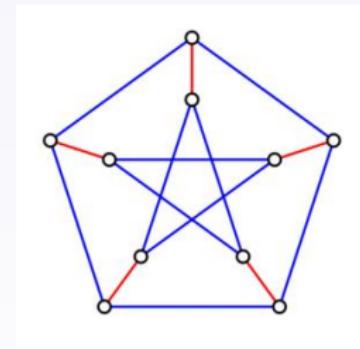
③ Graph Cuts

④ Grab Cut

# Graph

## Graph:

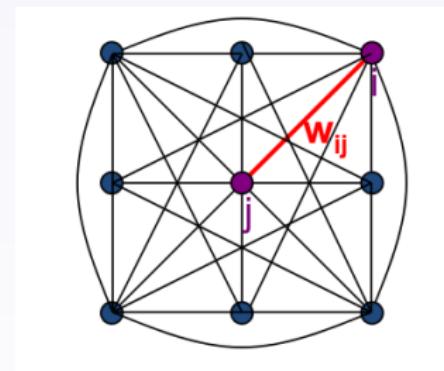
- node
- link
- weight
- direction



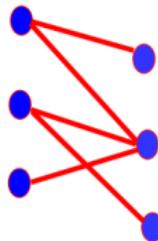
# Graph

## Image as graph:

- node for every pixel
- link between every pair of pixels
- weight of link for similarity



# Graph



$$G = (V, E)$$

$V$  (*Vertex*): Nodes

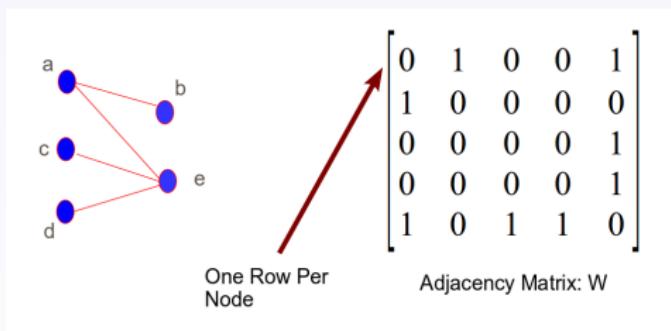
$E$  (*Edge*): Links



$\text{Image} = \{ \text{pixels} \}$

Pixel similarity

# Graph

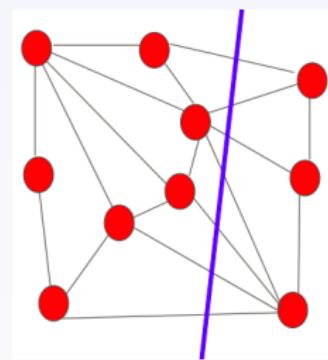


A graph can be represented with a matrix.

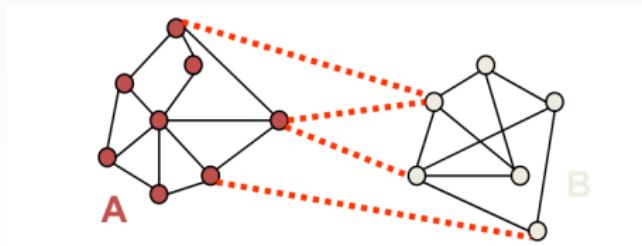
# Graph

## Cut:

- sub-set of edges  $E$
- set of links whose removal makes a graph disconnected
- cost of partition



$$Cut(A, B) = \sum_{i \in A, j \in B} w(i, j)$$



# Graph

## Weight:

- relation of pixels,  $w(i, j)$ 
  - similarity
  - closeness

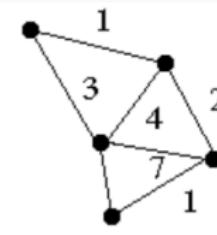
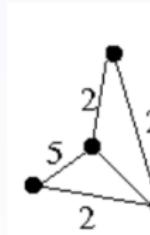
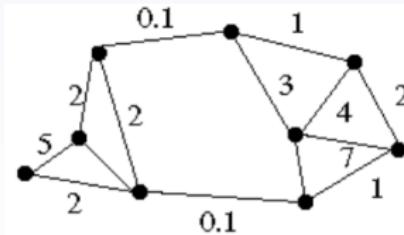
$$w_{ij} = \exp\left(\frac{\|F_i - F_j\|_2^2}{\sigma^2}\right) \times \begin{cases} \exp\left(\frac{\|X_i - X_j\|_2^2}{\sigma^2}\right) & \frac{\|X_i - X_j\|_2^2}{\sigma^2} < R \\ 0 & \text{otherwise} \end{cases}$$

- - value of gray scale  $F_i$
  - position  $X_i$
  - distance  $R$

# Graph

## Min Cut:

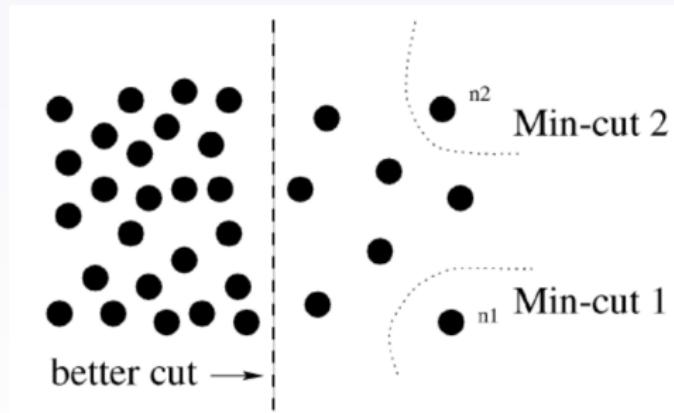
- Minimum cut is the cut of minimum weight, where the value of  $Cut(A, B)$  is minimum.



# Graph

## Min Cut Problem:

- Min Cut will choose a cut with one small cluster.



# Normalized Cuts

## Normalized Cuts<sup>1</sup>:

- cut is minimum
- cluster size is similar

---

<sup>1</sup>“Normalized Cuts and Image Segmentation”, JianBo Shi and Jitendra Malik. PAMI, 2000

# Normalized Cuts

## NCut:

$$NCut(A, B) = \frac{Cut(A, B)}{assoc(A, V)} + \frac{Cut(A, B)}{assoc(B, V)}$$

-

- $Cut(A, B)$ : the **cost** of partition as  $A$  and  $B$

$$Cut(A, B) = \sum_{i \in A, j \in B} w(i, j)$$

- $assoc(A, V)$ : **volume** of set  $A$ .

$$assoc(A, V) = \sum_{i \in A} d_i$$

$$d_i = \sum_{j \in V} w(i, j)$$

# Normalized Cuts

$$\begin{aligned} NCut(A, B) &= \frac{Cut(A, B)}{assoc(A, V)} + \frac{Cut(A, B)}{assoc(B, V)} \\ &= \frac{\sum_{(x_i>0, x_j<0)} -w_{ij} \cdot x_i \cdot x_j}{\sum_{x_i>0} d_i} + \frac{\sum_{(x_i>0, x_j<0)} -w_{ij} \cdot x_i \cdot x_j}{\sum_{x_j<0} d_i} \end{aligned}$$

$$y = (1 + x) - b(1 - x)$$

$$\min_x NCut(x) = \min_y \frac{y^T(D - W)y}{y^T D y}$$

$$\implies (D - W)y = \lambda D y$$

# Normalized Cuts

## Recursive normalized cuts:

- Given an image or image sequence, set up a **weighted graph**:  
 $G = (V, E)$ 
  - Vertex for each pixel
  - Edge weight for nearby pairs of pixels
- Solve for **eigenvecotrs** with the smallest eigenvalues:  $(D - W)y = \lambda Dy$ 
  - Use the eigenvector with the second smallest eigenvalue to bipartition the graph
  - The result is an approximation
- Recursively repartition the segmented parts if necessary

# Normalized Cuts

## Adavantage:

- Generic framework that can be used with different features and affinity formulations
- Provides regular segments

## Disadavantage:

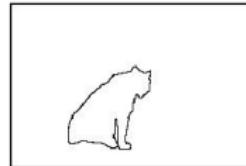
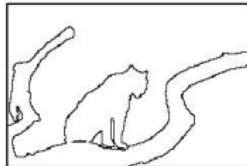
- Need to chose number of segments
- High storage requirement and time complexity
- Bias towards partitioning into equal segments

# Normalized Cuts

[Demo]

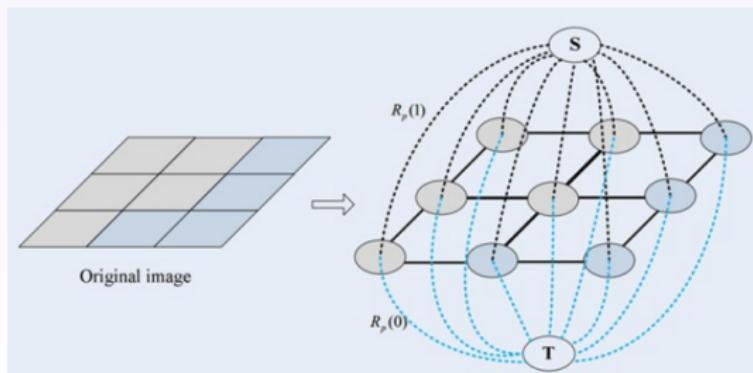


# Graph Cut



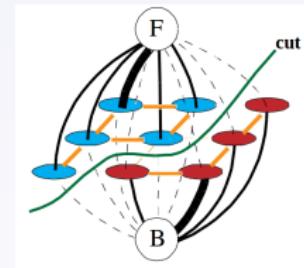
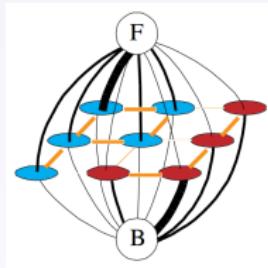
# Graph Cuts<sup>2</sup>

- Two kinds of **vertices**
  - **normal** vertices : nodes
  - **terminal** vertices: S (source point), T (sink point)
- Two kinds of **edges**
  - **n-link**: node to node
  - **t-link**: s-node or t-node to node

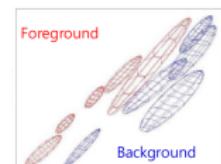
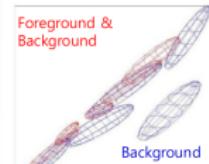


<sup>2</sup>"Interactive Graph Cuts for Optimal Boundary & Region Segmentation of Objects in N-D Images", Boykov. ICCV, 2001

# Graph Cuts



- $S \Rightarrow$  foreground
- $T \Rightarrow$  background
- $S \cup T = V$



# Graph Cuts

**Label image with 0, 1**

- “object” pixel is 1
- “background” pixel is 0

**Data element set  $P$  representing the image pixels**

- $A = (A_1, A_2, \dots, A_p, \dots, A_P), \quad A_p = \{0, 1\}$ 
  - $P$  the number of pixels
  - $A_p$  the label of pixel  $p$

# Graph Cuts

## Energy Function:

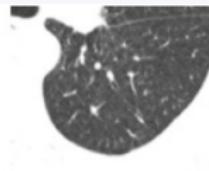
$$\begin{aligned} E(A) &= \lambda \cdot R(A) + B(A) \\ &= \lambda \cdot \sum_{p \in P} R_p(A_p) + \sum_{p \in P} \sum_{\{p,q\} \in N} B_{\{p,q\}} \cdot \delta(A_p, A_q) \end{aligned}$$

- $R(A)$  defines **penalties** for assigning each pixel to object or background
  - **t-link**
  - $R_p(\text{"obj"})$  is lower,  $p$  belongs to object
  - $R_p(\text{"bkg"})$  is lower,  $p$  belongs to background
- $B(A)$  sets **penalties** of discontinuities between pixels
  - **n-link**
  - $B(A)$  describes the boundary properties of the segmentation
  - $B_{\{p,q\}}$  is large,  $p$  and  $q$  are similar
  - $B_{\{p,q\}} \rightarrow 0$ ,  $p$  and  $q$  are different

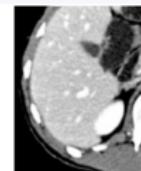
# Graph Cuts



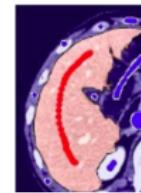
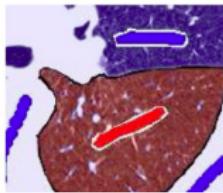
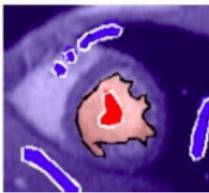
(c) Cardiac MR



(e) Lung CT

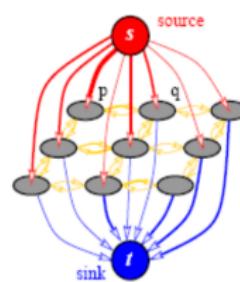


(g) Liver MR



# Graph Cuts

Edge	Weight	Condition
n-link $\{p, q\}$	$B_{\{p, q\}}$	$\{p, q\} \in N$
t-link $\{p, s\}$	$\lambda \cdot R_p(\text{"bkg"})$	$p \in P, p \in O \cup B$
	$K$	$p \in O$
	0	$p \in B$
t-link $\{p, t\}$	$\lambda \cdot R_p(\text{"obj"})$	$p \in P, p \in O \cup B$
	0	$p \in O$
	$K$	$p \in B$



# Graph Cuts

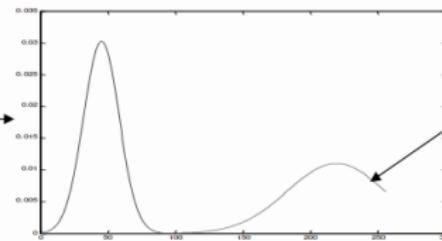
## t-links:

$$R(A) = \sum_{p \in P} R_p(A_p)$$

- Estimate **object and background** intensity distribution based on object or background seeds
- $R_p(obj) = -\ln P_r(A_p | obj)$
- $R_p(bkg) = -\ln P_r(A_p | bkg)$
- - if  $P_r(A_p | obj) > P_r(A_p | bkg)$ 
    - $R_p(obj) < R_p(bkg)$
    - the pixel belongs to object
  - else
    - $R_p(obj) > R_p(bkg)$
    - the pixel belongs to background

# Graph Cuts

Estimated  
Background  
Intensity  
Distribution  
Based on  
Background  
Seeds



Estimated  
Object Intensity  
Distribution  
Based on  
Background  
Seeds

# Graph Cuts

## n-links:

$$B(A) = \sum_{p \in P} \sum_{\{p,q\} \in N} B_{\{p,q\}} \cdot \delta(A_p, A_q)$$

$$B_{p,q} \propto \exp\left(-\frac{(A_p - A_q)^2}{2\sigma^2}\right)$$

$$\delta(A_p, A_q) = \begin{cases} 1 & A_p \neq A_q \\ 0 & \text{otherwise} \end{cases}$$

- $B(A)$  sets penalty of **discontinuities** between pixels of similar intensities
- $A_p$  and  $A_q$  are the intensities of pixel  $p$  and  $q$ 
  - $|A_p - A_q| < r$ , the penalty is large
  - $|A_p - A_q| > r$ , the penalty is small

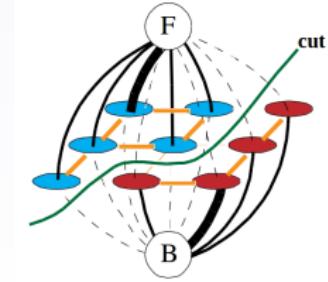
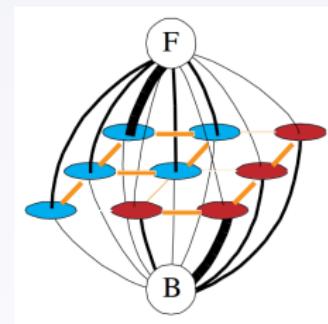
# Graph Cuts

## Conclusion:

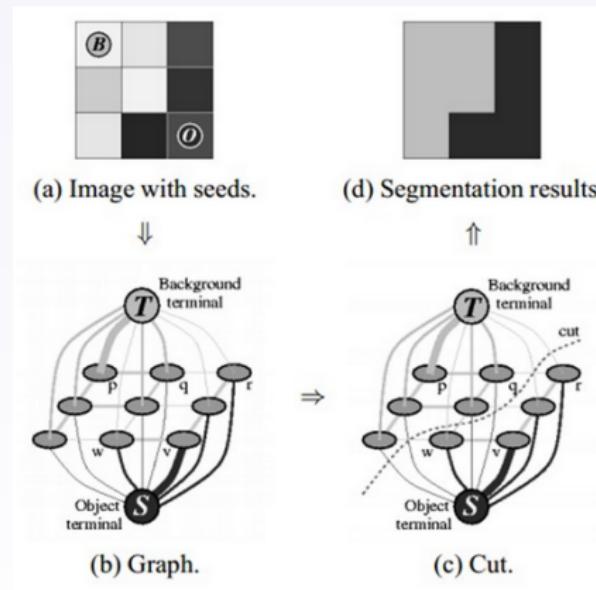
Partition into object and background

- image  $\rightarrow$  graph
  - two kinds of nodes
  - two kinds of edges
- n-links : nodes  $\rightarrow B(A)$
- t-links : s-node and t-node to nodes  $\rightarrow R(A)$
- minimum energy:  

$$E(A) = \lambda \cdot R(A) + R(B)$$

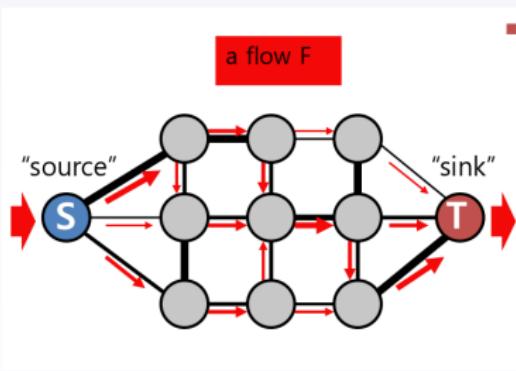


# Graph Cuts



# Graph Cuts

## Max Flow:



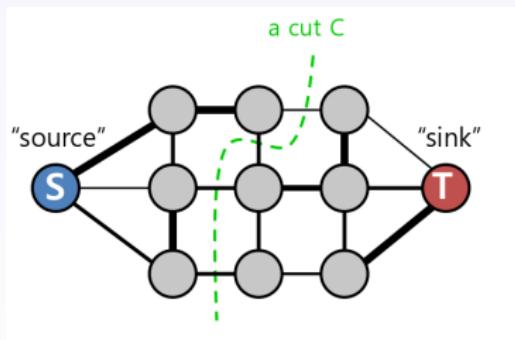
- Each edge is a “pipe”
- Edge **weights** give the pipe’s capacity
- $S=\text{start}$     $T=\text{end}$
- Find the **largest flow**  $F$  of “water” that can be sent from “source” to “sink” along the pipes

# Graph Cuts

A flow in  $G$  satisfies the following three properties:  $u, v \in V$

- Capacity Constraint:  $f(u, v) \leq c(u, v)$
- Skew Symmetry:  $f(u, v) = -f(v, u)$
- Flow Conservation:  $\sum f(u, v) = 0$

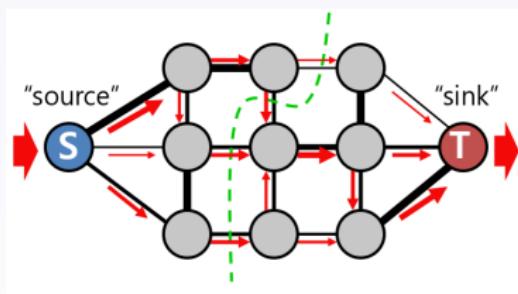
# Graph Cuts



## Min Cut:

- Find the way to cut the edges so that “source” is separated from “sink”
- Cut edges going from source side to sink side

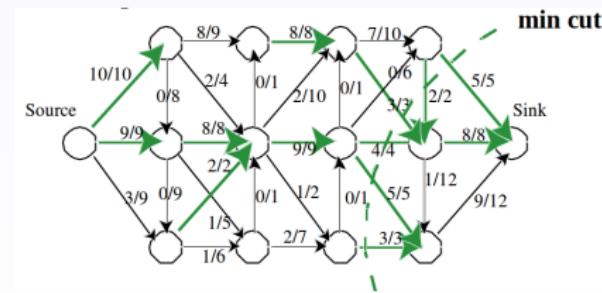
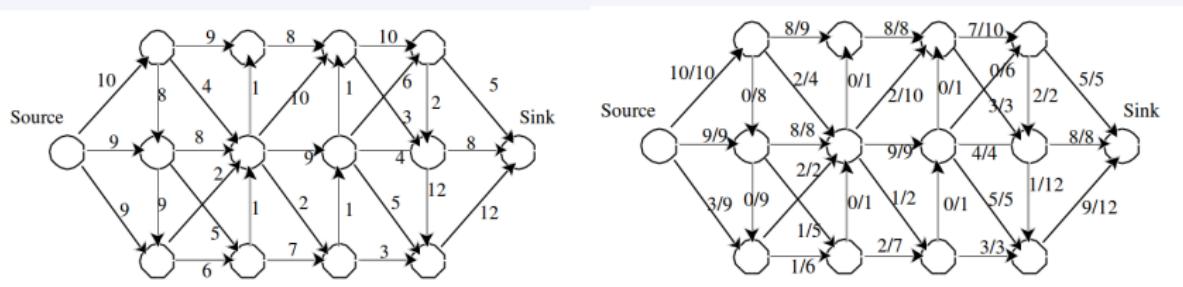
# Graph Cuts



**Max Flow = Min Cut:**

In graph G, the **maximum source-to-sink flow** possible is equal to the capacity of the **minimum cut** in G.

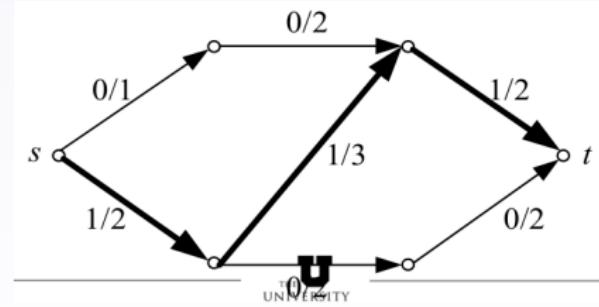
# Graph Cuts



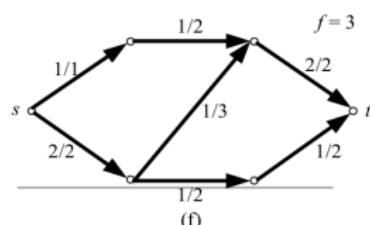
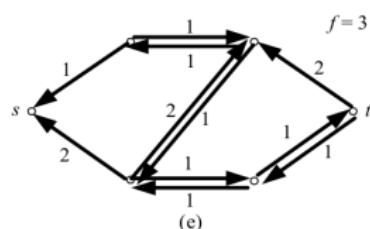
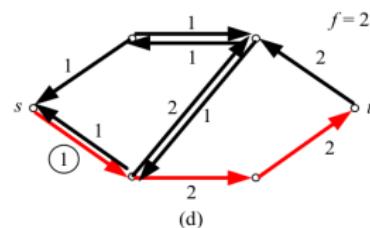
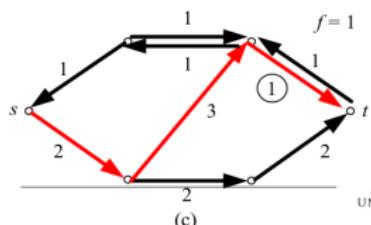
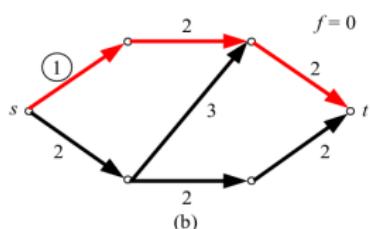
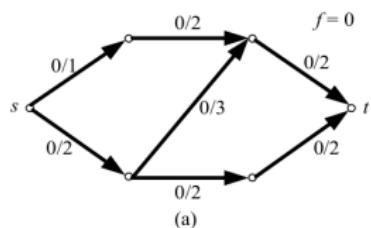
# Graph Cuts

## Augmenting Path:

- Start from zero flow
- Increase the flow gradually by finding a path from  $S$  to  $T$ , along which more flow can be sent, until a max-flow is achieved



# Graph Cuts



# Grab Cut

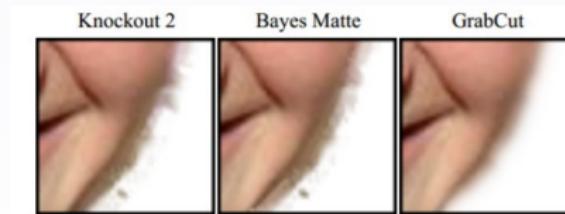
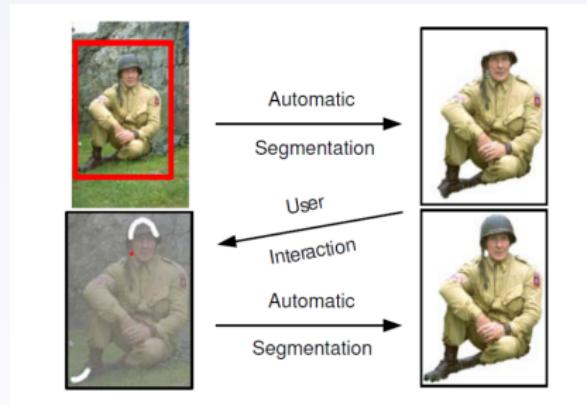
## Grab Cut<sup>3</sup>:

- Uses GMM to work with color images
- Allows an iterative approach to segmentation
- Draws frame to label possible object
- Adds Border Matting

---

<sup>3</sup>"GrabCut Interactive Foreground Extraction using Iterated Graph Cuts", Carsten Rother. Microsoft Research Cambridge, UK, 2004

# Grab Cut



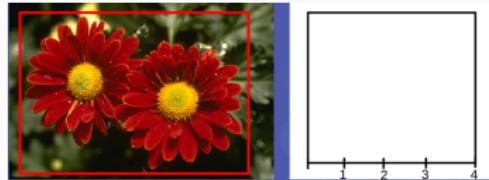
# Grab Cut



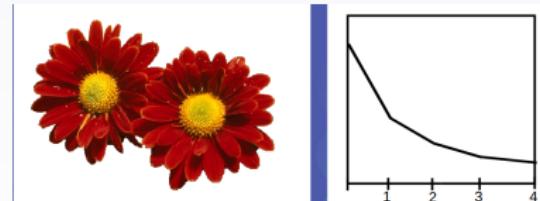
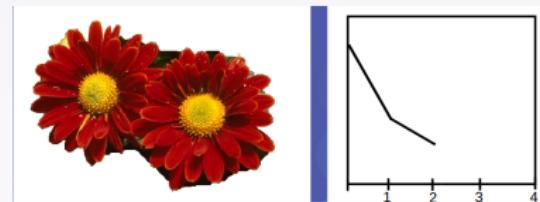
# Graph Cut



# Grab Cut



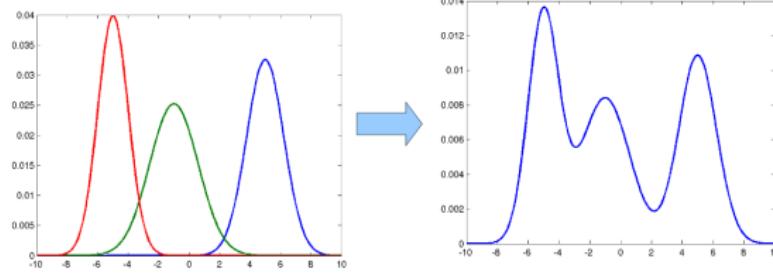
# Grab Cut



# Grab Cut

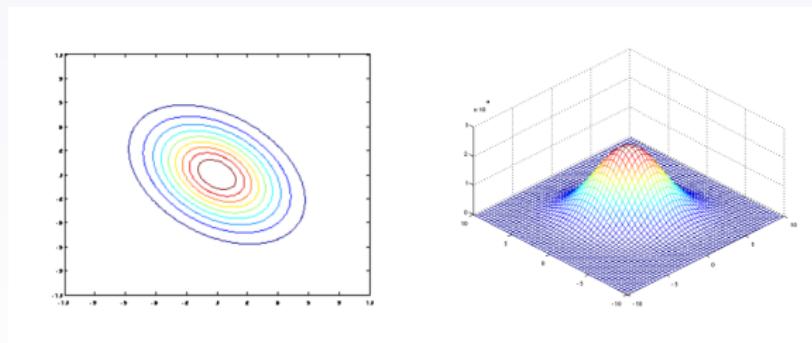
## GMM: Gaussian Mixture Model

$$\begin{aligned} p(x) &= \sum_{k=1}^K p(k)p(x|k) \\ &= \sum_{k=1}^K \pi_k N(x | \mu_k, \sigma_k) \end{aligned}$$

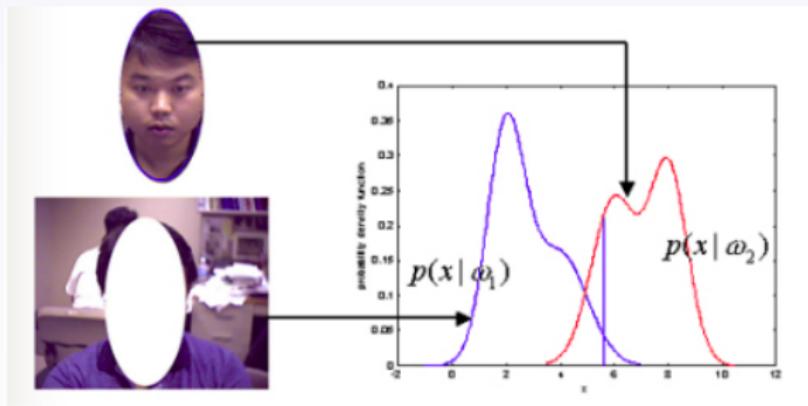


# Grab Cut

$$N(x \mid \mu, \Sigma) = \frac{1}{(2\pi)^{D/2}} \frac{1}{|\Sigma|^{\frac{1}{2}}} \exp\left(-\frac{1}{2}(x - \mu)^T \Sigma^{-1} (x - \mu)\right)$$



# Grab Cut



# Grab Cut

## Gibbs Energy:

$$E(\underline{\alpha}, k, \underline{\theta}, z) = U(\underline{\alpha}, k, \underline{\theta}, z) + V(\underline{\alpha}, z)$$

$$U(\underline{\alpha}, k, \underline{\theta}, z) = \sum_n D(\alpha_n, k_n, \underline{\theta}, z_n)$$

$$\begin{aligned} D(\alpha_n, k_n, \underline{\theta}, z_n) &= -\log \pi(\alpha_n, k_n) + \frac{1}{2} \log \det \Sigma(\alpha_n, k_n) \\ &+ \frac{1}{2} [z_n - \mu(\alpha_n, k_n)]^T \Sigma(\alpha_n, k_n)^{-1} [z_n - \mu(\alpha_n, k_n)] \end{aligned}$$

$$\underline{\theta} = \{\pi(\alpha, k), \mu(\alpha, k), \Sigma(\alpha, k), \quad \alpha = 0, 1 \quad k = 1 \dots K\}$$

$$V(\underline{\alpha}, z) = \gamma \sum_{(n,n) \in C} [\alpha_n \neq \alpha_m] \exp -\beta \| z_m - z_n \|^2$$

# Grab Cut

$$U(\underline{\alpha}, k, \underline{\theta}, z) = \sum_n D(\alpha_n, k_n, \underline{\theta}, z_n)$$

- $U(\underline{\alpha}, k, \underline{\theta}, z)$  defines **penalty** for **assigning** pixel to object or background

$$D(x) = \sum_{i=1}^K \pi_i g_i(x | \mu_i, \Sigma_i), \quad \sum_{i=1}^K \pi_i = 1$$

$$g(x | \mu, \Sigma) = \frac{1}{[(2\pi)^d | \Sigma |]^{\frac{1}{2}}} \exp[-\frac{1}{2}(x - \mu)^T \Sigma^{-1} (x - \mu)]$$

- $\mu, \Sigma, \pi_i$  parameters need to learn to define.

# Grab Cut

$$V(\underline{\alpha}, z) = \gamma \sum_{(n,n) \in C} [\alpha_n \neq \alpha_m] \exp -\beta \| z_m - z_n \|^2$$

- $V(\underline{\alpha}, z)$  sets **penalty** of **discontinuities** between pixels of similar intensities
- $\beta$  is determined by the contrast to modify the value of  $\| z_m - z_n \|$

# Grab Cut

## Initialisation:

- step1 User initialises trimap  $T$ . The foreground is set to  $T_F=1$ , the background is set to  $T_B=0$ .
- step2 Initialise  $\alpha_n = 0$  for  $n \in T_B$  and  $\alpha_n = 1$  for  $n \in T_F$ .
- step3 Background and foreground GMMs are initialised from sets  $\alpha_n = 0$  and  $\alpha_n = 1$  respectively.

# Grab Cut

## Iterative minimisation:

step1 Assign  $GMM$  components to pixels: for each  $n$  in  $T_F$ :

$$k_n = \arg \min_{k_n} D_n(\alpha_n, k_n, \theta, z_n)$$

step2 Learn  $GMM$  parameters from data  $z$ :

$$\underline{\theta} = \arg \min_{\underline{\theta}} U(\underline{\alpha}, k, \underline{\theta}, z)$$

step3 Estimate segmentation: use min cut to solve:

$$\min_{\{\alpha_n: n \in T_U\}} = \min_k E(\underline{\alpha}, k, \underline{\theta}, z)$$

step4 Repeat from Step1, until convergence.

step5 Apply border matting.

# Grab Cut

[Demo]

**User editing:**

- *Edit*: fix some pixels either to  $\alpha_n = 0$  (background brush) or  $\alpha_n = 1$  (foreground brush); update trimap  $T$  accordingly. Perform *Step3* above, just once.
- *Refine operation*: [optional] perform entire iterative minimisation algorithm .