Graph-based Segmentation

基于图论的图像分割

戴嘉伦 王如晨 赵海伟

CVBIOUC

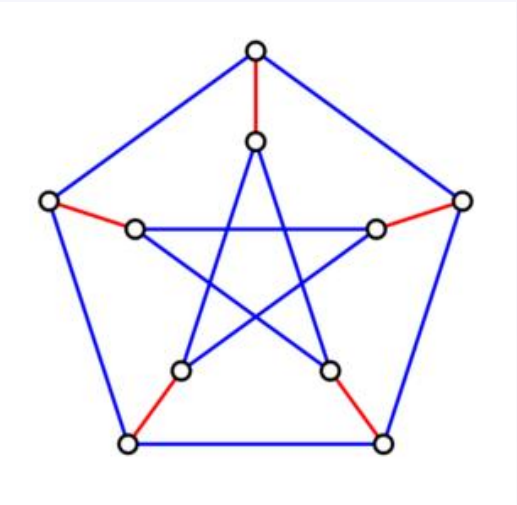
May 19, 2015

目录

Graph

Graph:

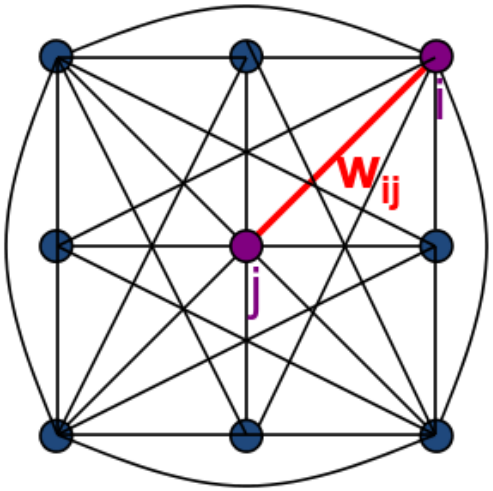
- node
- link
- weight
- direction



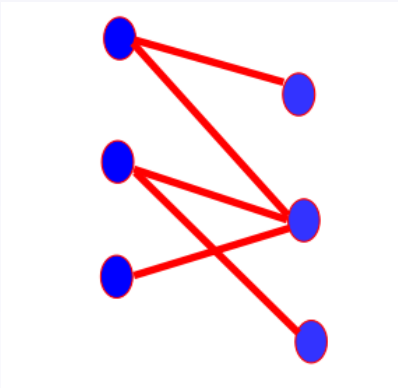
Graph

Image as graph:

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



Graph

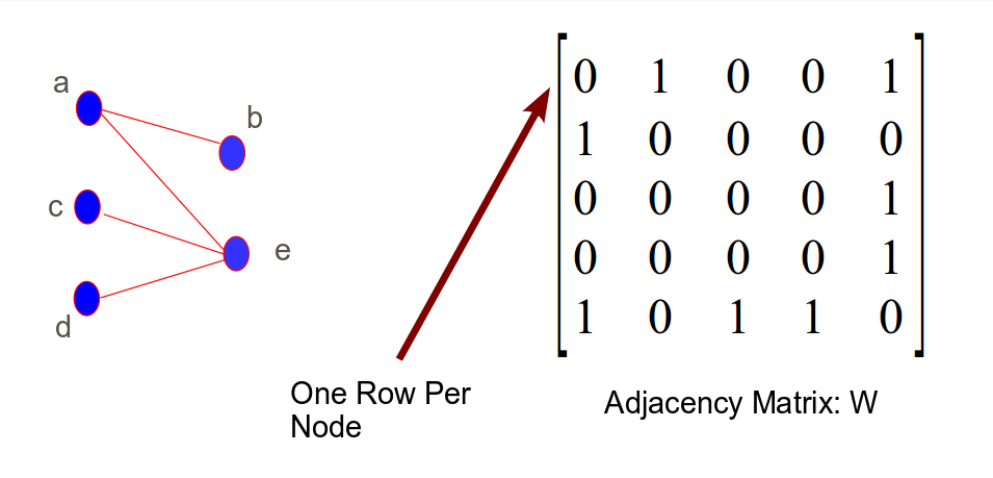


$$G = (V, E)$$

$V\ (Vertex)$: Nodes
$E\ (Edge)$: Links

Image $= \{$ pixels$\}$
Pixel similarity

Graph

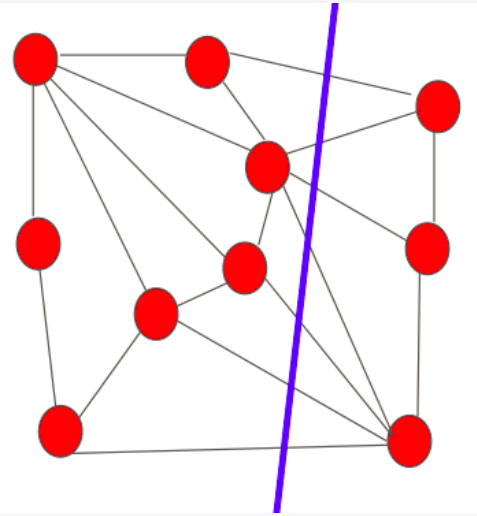


A graph can be represented with a matirx.

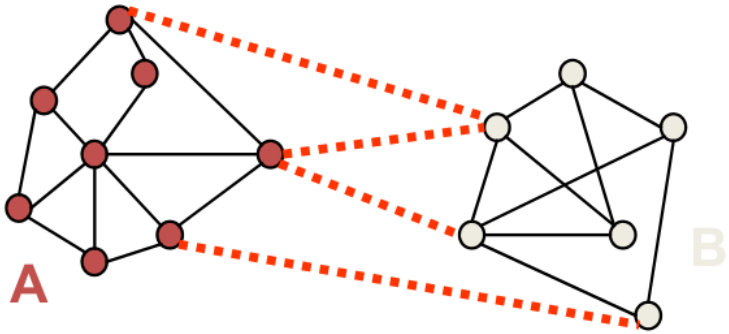
Graph

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

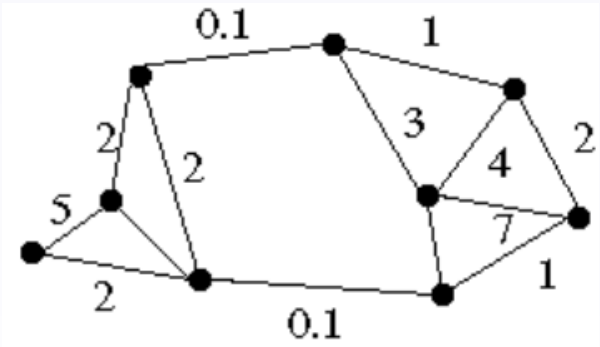
$$w_{ij} = \exp(\frac{\parallel F_i - F_j \parallel_2^2}{\sigma^2}) \times \begin{cases} \exp(\frac{\|X_i - X_j\|_2^2}{\sigma^2}) & \frac{\|X_i - X_j\|_2^2}{\sigma^2} < R \\ 0 & \text{otherwise} \end{cases}$$

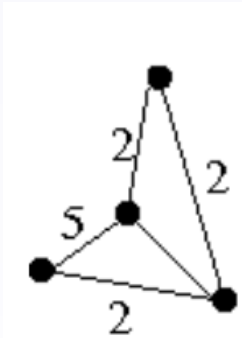
-
  - value of gray scale F_i
  - position X_i
  - distance R

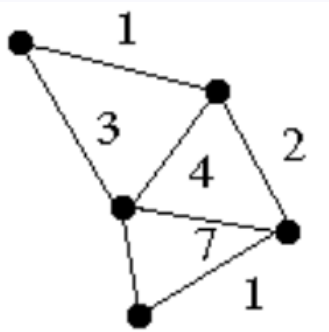
Graph

Min Cut:

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



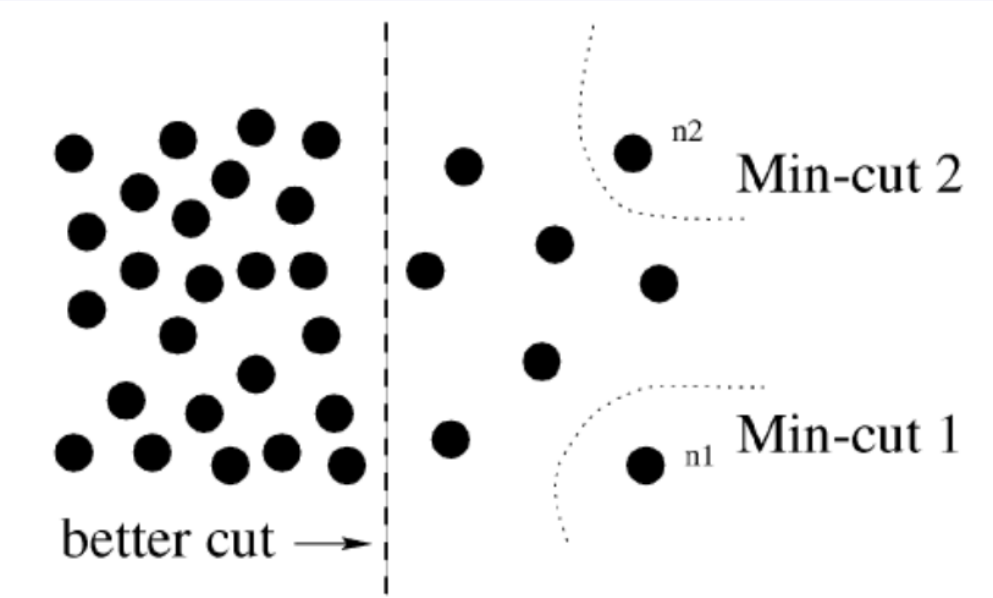




Graph

Min Cut Problem:

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



Normalized Cuts

Normalized Cuts[1]:

- cut is minimum
- cluster size is similar

[1] "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.

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

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

Normalized Cuts

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

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

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

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

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

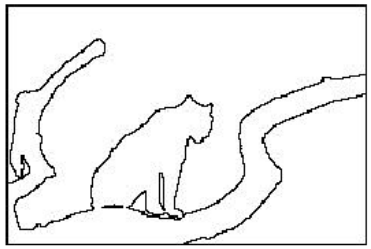
Disadavantage:

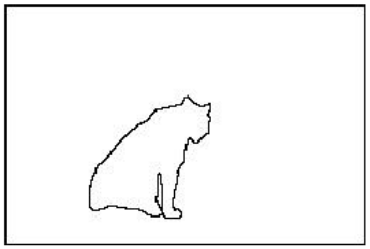
- Need to chose number of segments
- High sotrage requirement and time complexity
- Bias towards partitioning into equal segments

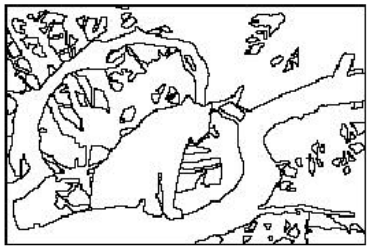
Normalized Cuts

[Demo]

Graph Cut

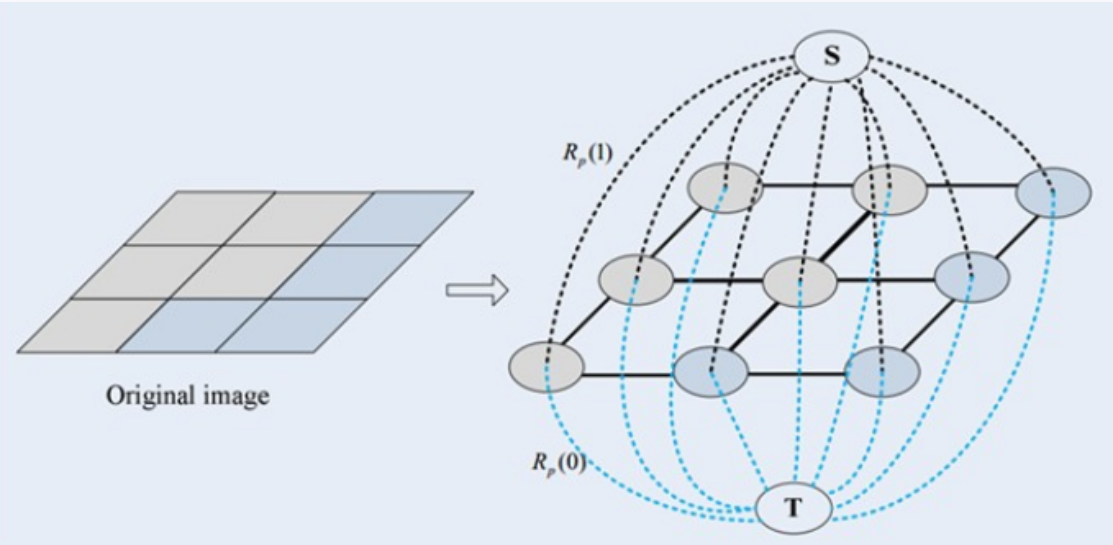






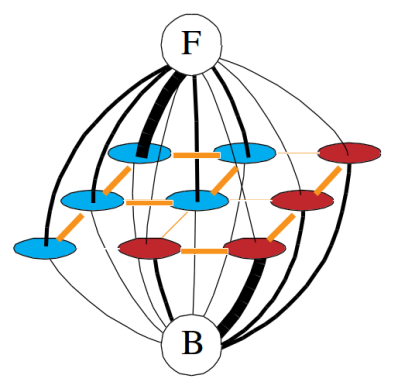
Graph Cuts[2]

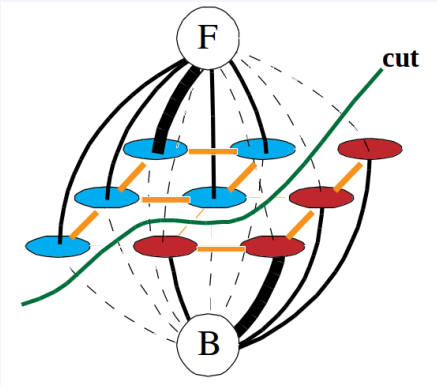
- Two kinds of vertices
  - normal vertices : nodes
  - terminal vertices: S (source point), T (sink point)
- Two kinds of edges
  - n-link: node to node
  - t-link: s-node or t-node to node



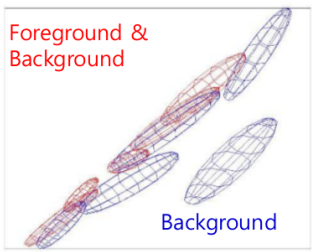
[2] "Interactive Graph Cuts for Optimal Boundary & Region Segmentation of Objects in N-D Images", Boykov. ICCV, 2001

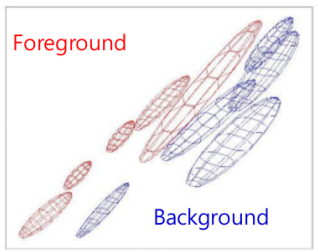
Graph Cuts





- 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**

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

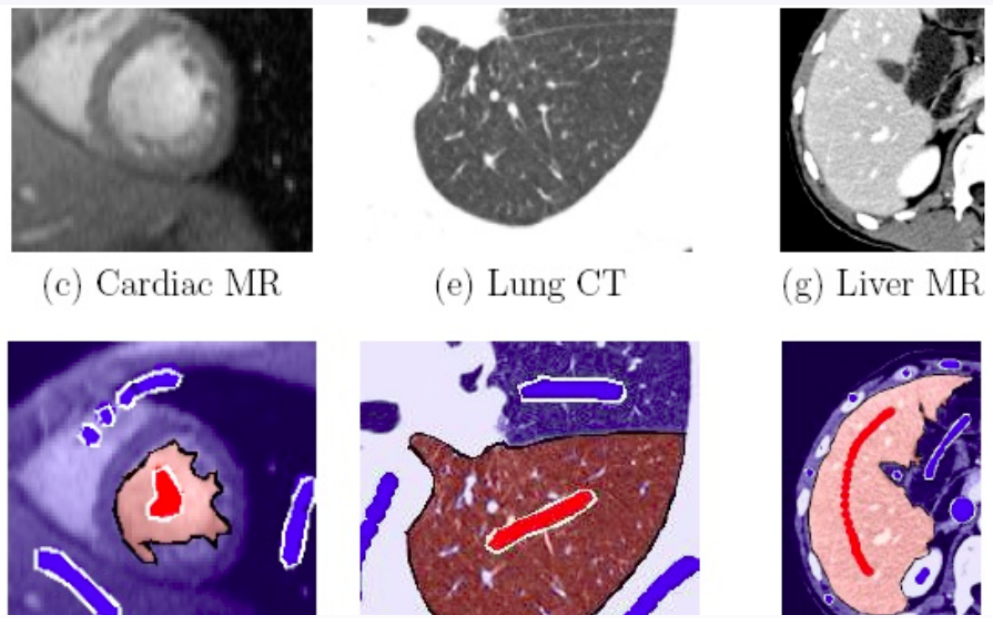
Graph Cuts

Energy Function:

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

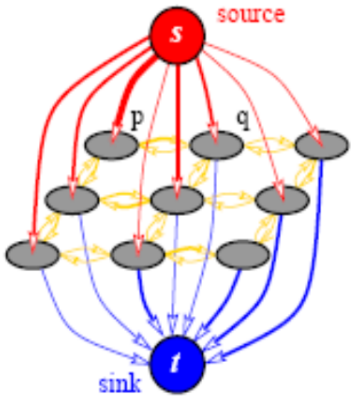
- $R(A)$ defines penalties for assigning each pixel to object or background
  - **t-link**
  - R_p("obj") is lower, p belongs to object
  - R_p("bkg") is lower, p belongs to background
- $B(A)$ sets penalties of discontinuities between pixels
  - **n-link**
  - $B(A)$ describes the boundary properties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

Graph Cuts



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

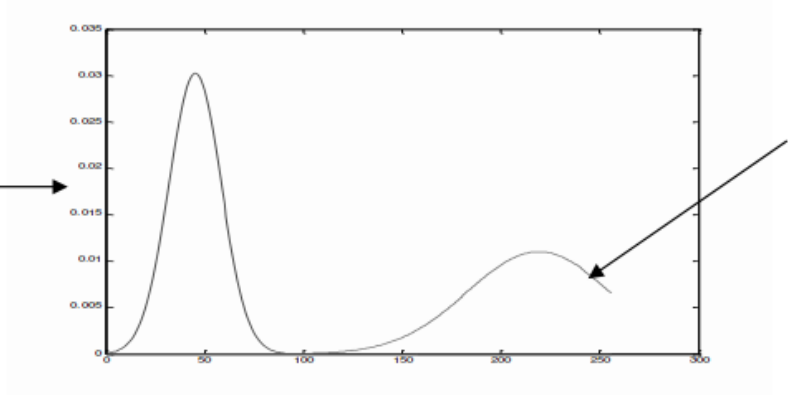
t-links:

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

- Estimate object and background intensity distribution based on object or background seeds
- $R_p(obj) = -\ln P_r(A_p \mid obj)$
- $R_p(bkg) = -\ln P_r(A_p \mid bkg)$
- - *if* $P_r(A_p \mid obj) > P_r(A_p \mid bkg)$
    - $R_p(obj) < R_p(bkg)$
    - 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:

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

$$B_{p,q} \propto \exp(-\frac{(A_p - A_q)^2}{2\sigma^2}) \qquad \delta(A_p, A_q) = \begin{cases} 1 & A_p \neq A_q \\ 0 & \text{otherwise} \end{cases}$$

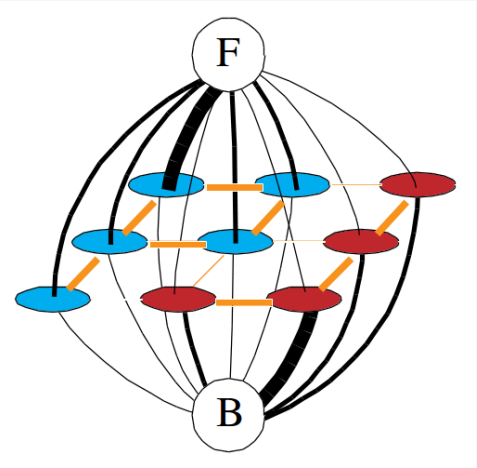
- $B(A)$ sets penalty of discontinuities between pixels of similar intensities
- A_p and A_q are the intensities of pixel p and q
  - $| A_p - A_q | < r$, the penalty is large
  - $| A_p - A_q | > r$, the penalty is small

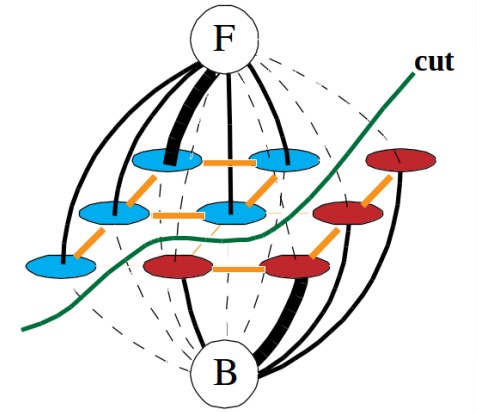
Graph Cuts

Conclusion:

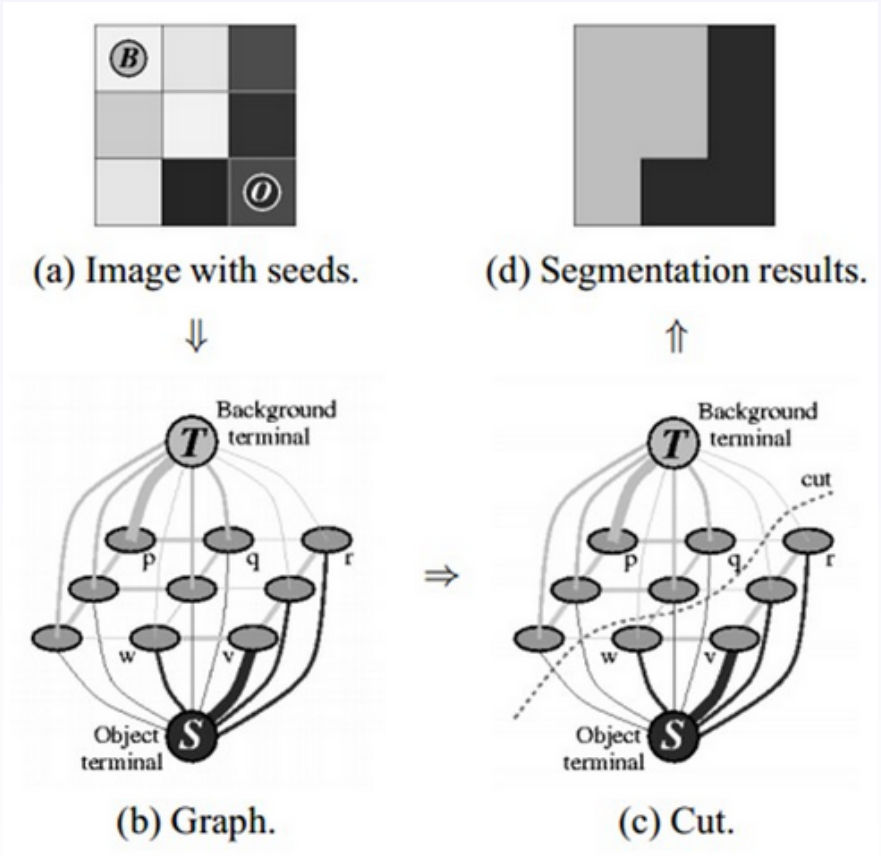
Partition into object and background

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

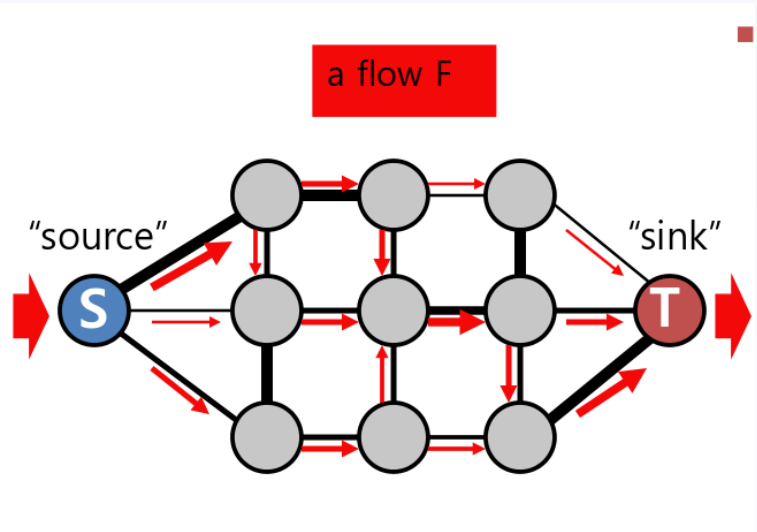




Graph Cuts


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

Graph Cuts



Max Flow:

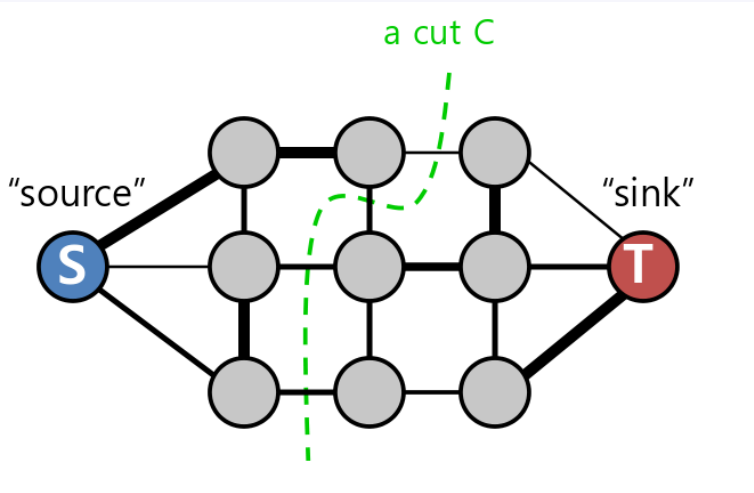
- Each edge is a "pipe"
- Edge weights give the pipe's capacity
- S=start T=end
- Find the largest flow F of "water" that can be sent from "source" to "sink" along the pipes

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: $f(u, v) = -f(u, v)$
- Flow Conservation: $\sum f(u, v) = 0$

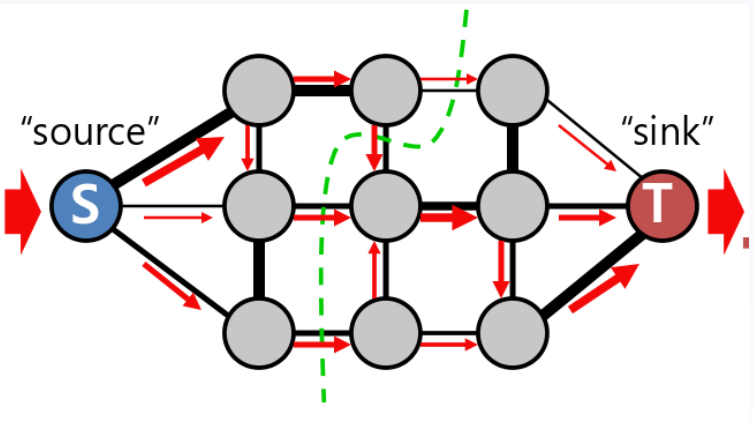
Graph Cuts



Min Cut:

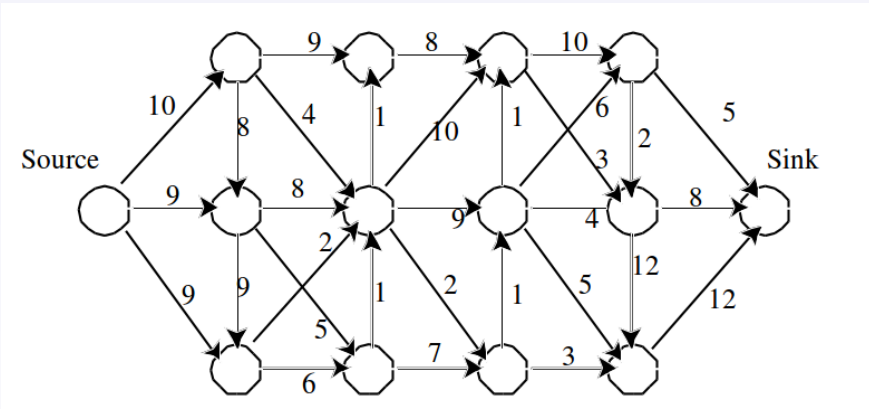
- Find the way to cut the edges so that "source" is seperated from "sink"
- Cut edges going from source side to sink side

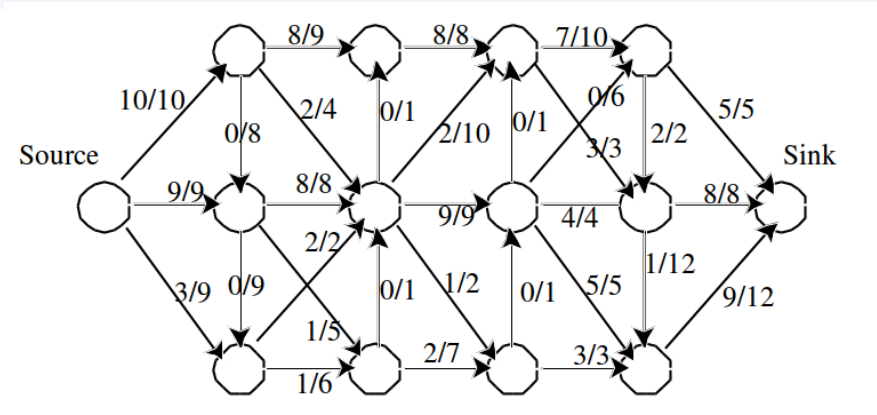
Graph Cuts

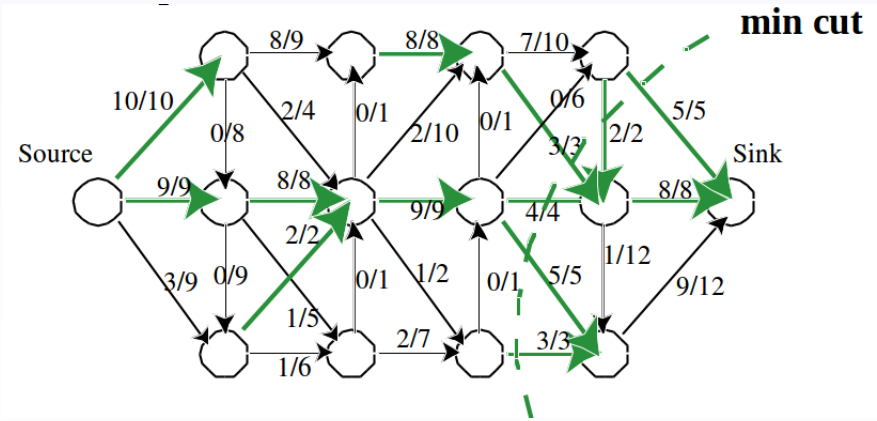


Max Flow = Min Cut:
In graph G, the maximum source-to-sink flow possible is equal to the capacity of the minimum cut in G.

Graph Cuts


Source
Sink
9
8
10
10
8
4
1
10
1
6
5
3
2
9
8
8
9
4
2
12
9
9
1
2
1
5
12
5
7
3
6

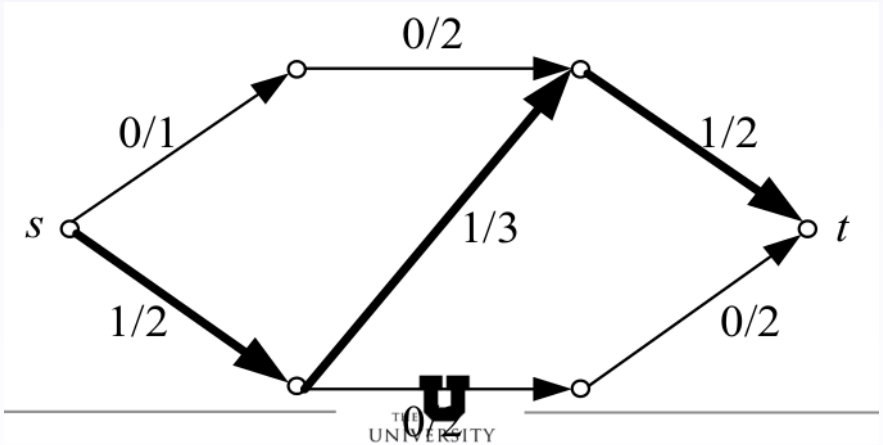

Source
Sink
8/9
8/8
7/10
10/10
2/4
0/1
0/6
5/5
0/8
2/10
0/1
3/3
2/2
9/9
8/8
8/8
9/9
4/4
2/2
1/12
3/9
0/9
0/1
1/2
0/1
5/5
9/12
1/5
2/7
3/3
1/6


min cut
Source
Sink
8/9
8/8
7/10
10/10
2/4
0/1
0/6
5/5
0/8
2/10
0/1
3/3
2/2
9/9
8/8
8/8
9/9
4/4
2/2
1/12
3/9
0/9
0/1
1/2
0/1
5/5
9/12
1/5
2/7
3/3
1/6

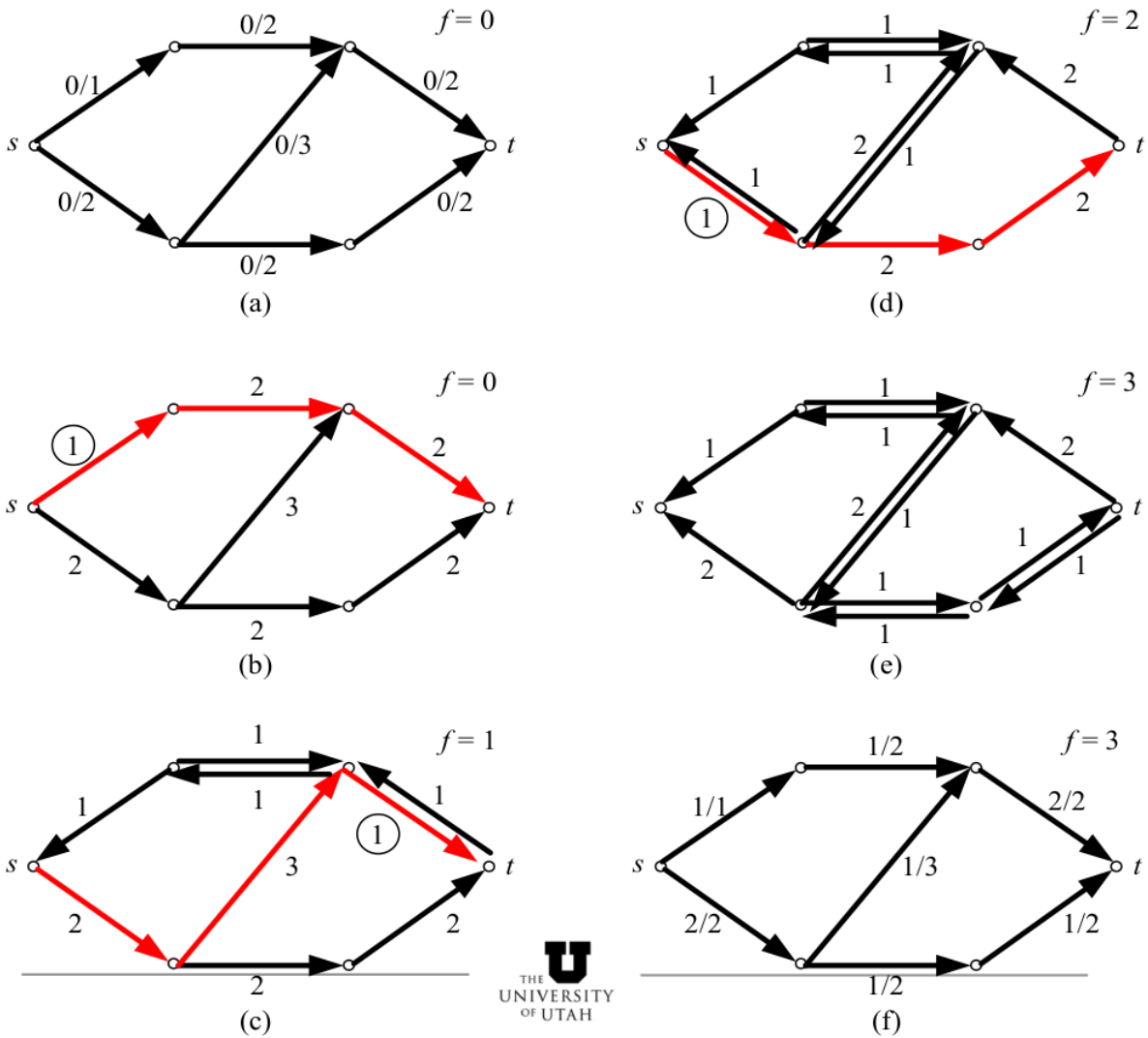
Graph Cuts

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


f = 0
0/2
0/1
0/2
0/3
0/2
0/2
0/2
s
t
(a)
f = 2
1
1
1
2
2
1
1
2
1
2
s
t
(d)
f = 0
2
1
2
3
2
2
2
s
t
(b)
f = 3
1
1
1
2
2
1
1
2
1
1
1
s
t
(e)
f = 1
1
1
1
1
1
3
2
2
2
s
t
(c)
f = 3
1/2
1/1
2/2
1/3
2/2
1/2
1/2
s
t
(f)
THE UNIVERSITY OF UTAH

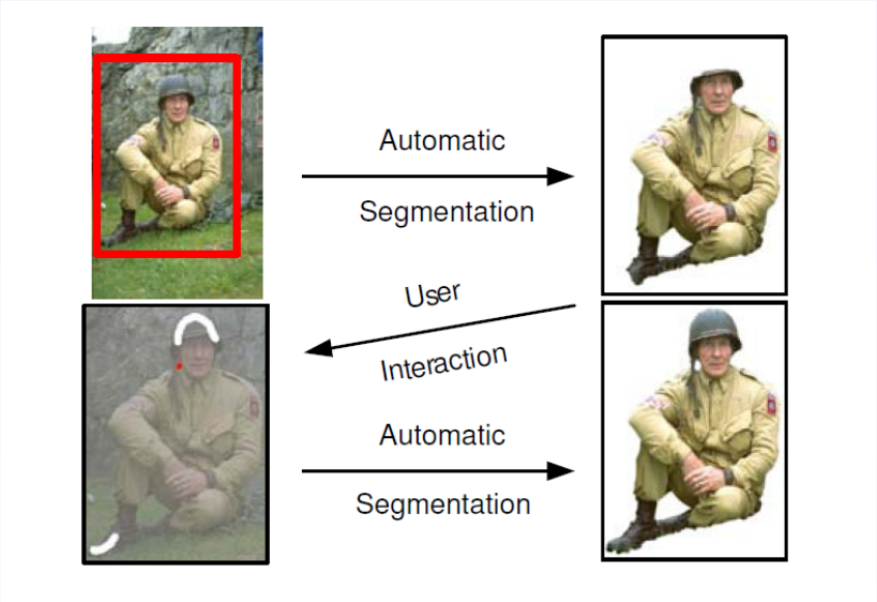
Grab Cut

Grab Cut[3]:

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

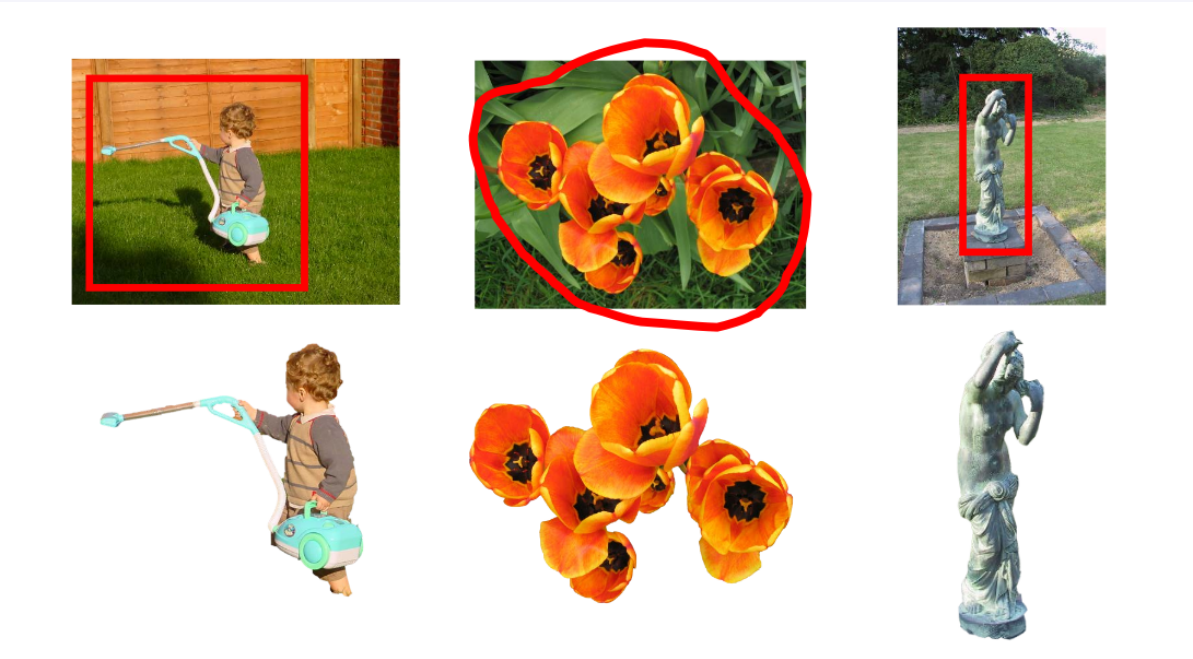
[3]"GrabCut Interactive Foreground Extraction using Iterated Graph Cuts", Carsten Rother. Microsoft Research Cambridge, UK, 2004

Grab Cut

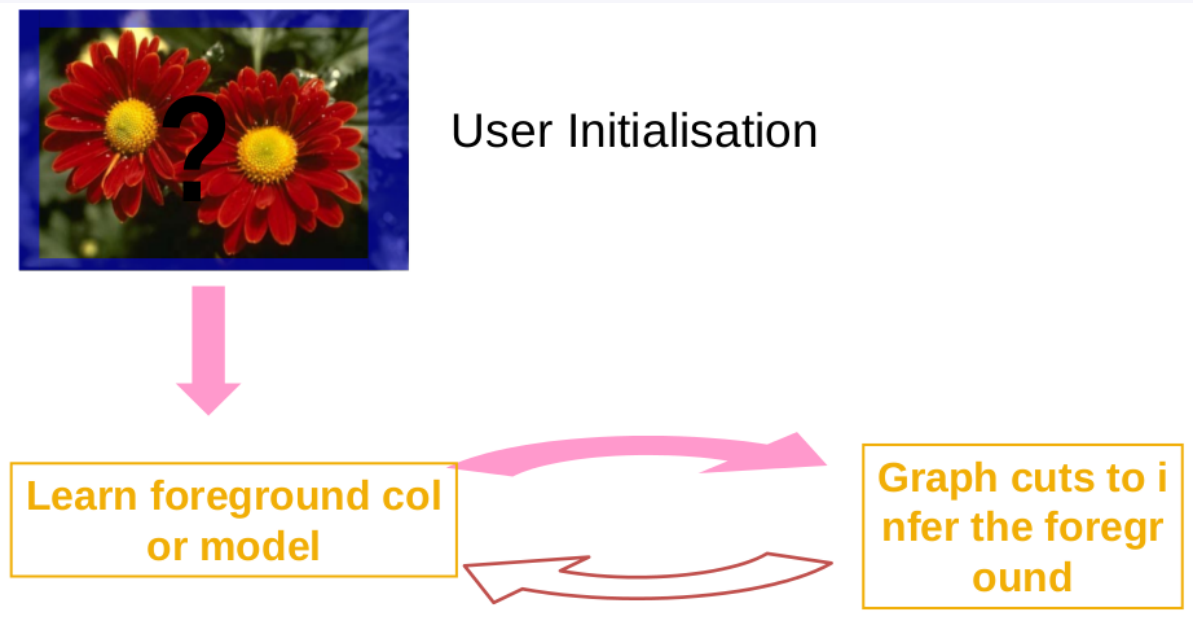

Automatic
Segmentation
User
Interaction
Automatic
Segmentation


Knockout 2
Bayes Matte
GrabCut

Grab Cut

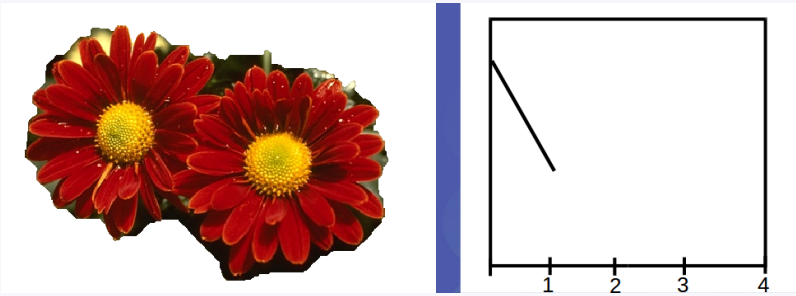


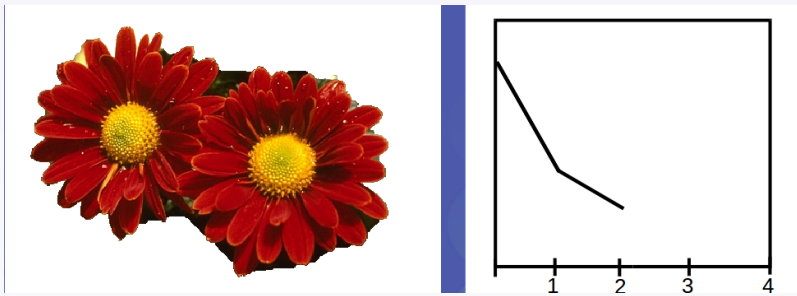
Graph Cut

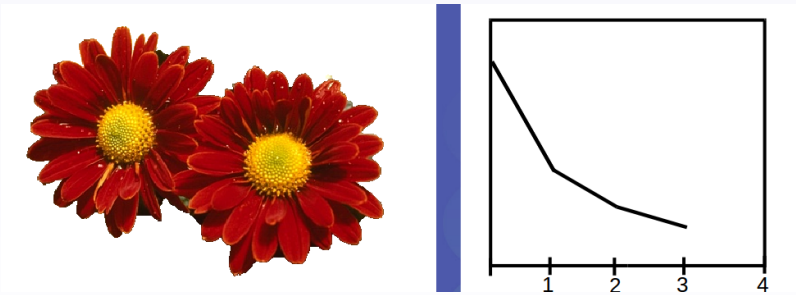

User Initialisation
Learn foreground col or model
Graph cuts to i nfer the foregr ound

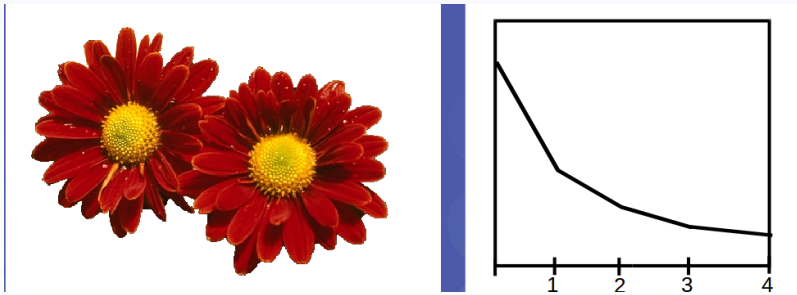
Grab Cut

Grab Cut







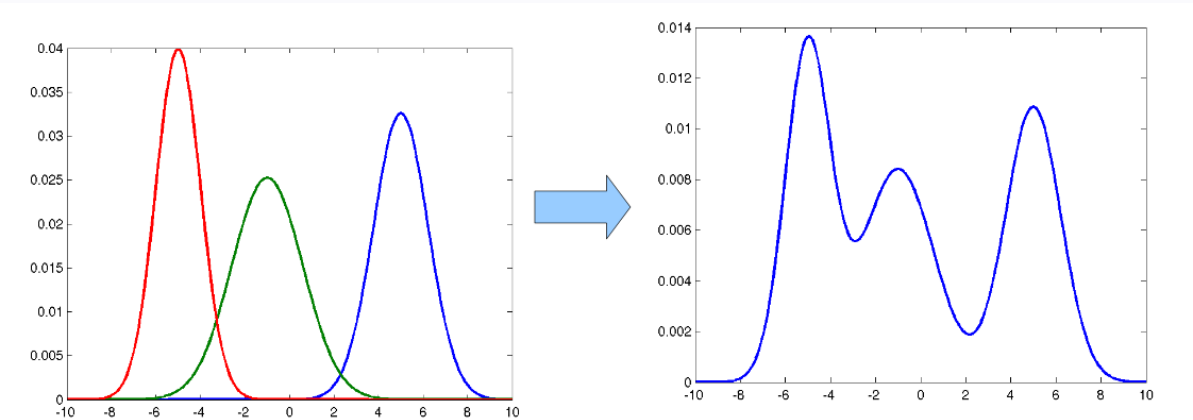


Grab Cut

GMM: Gaussion Mixture Model

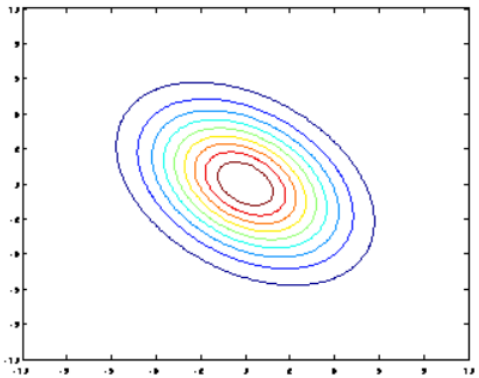
$$p(x) = \sum_{k=1}^{K} p(k)p(x|k)$$

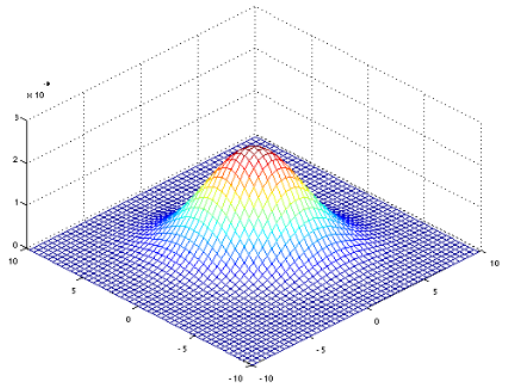
$$= \sum_{k=1}^{K} \pi_k N(x \mid \mu_k, \sigma_k)$$



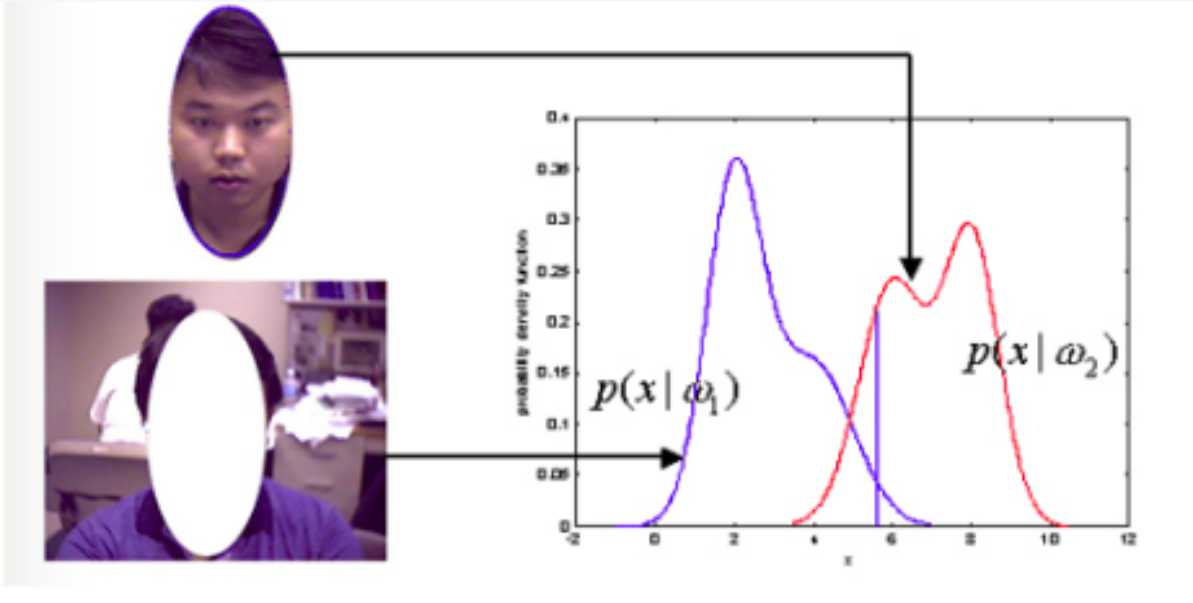
Grab Cut

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





Grab Cut



Grab Cut

Gibbs Energy:

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

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

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

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

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

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

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

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

- μ, Σ, π_i parameters need to learn to define.

Grab Cut

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

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

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 .