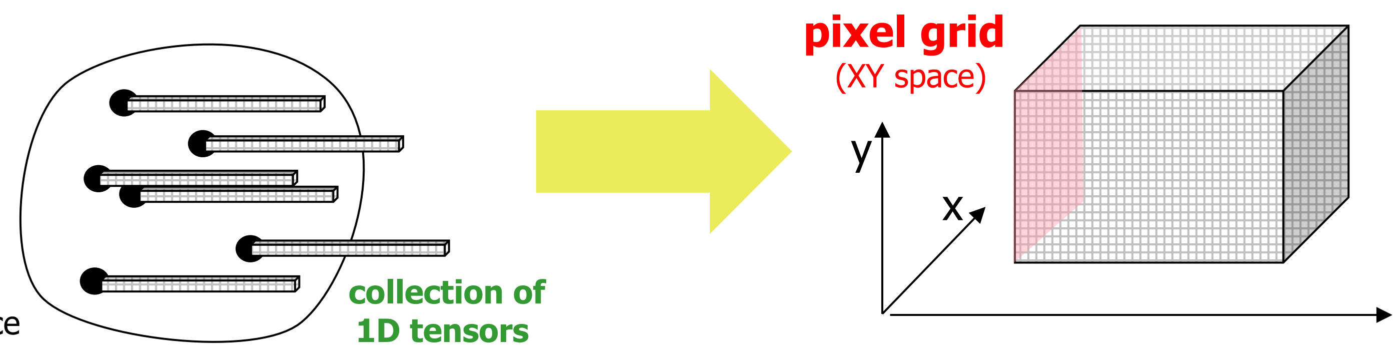
Spatial Regularization for Image Segmentation

alternative views about data representation

collection of feature vectors in R^n

3D tensor

e.g. points in RGB or RGBXY space



e.g. RGB

general features
common in ML

K-means, GMM, general graph clustering, ...

features embedded in a regular 2D grid
common in computer vision

convolution, geometry, shape, spatial regularity, ...

image segmentation objectives

Part II

Spatial Regularization

- Graphical Models on <u>grids</u>
  - boundary regularity (from shortest path to graph cut)
  - weakly-supervised and unsupervised segmentation
  - 3D shape reconstruction
  - losses: smoothness, edge-alignment, color-consistency, seed/label consistency, NLL

- Spatial regularization + feature clustering
  - joint shape regularization and color model fitting
  - variance clustering vs entropy clustering

OPTIONAL MATERIAL

Intelligent Scissors (a.k.a. *live-wire*)

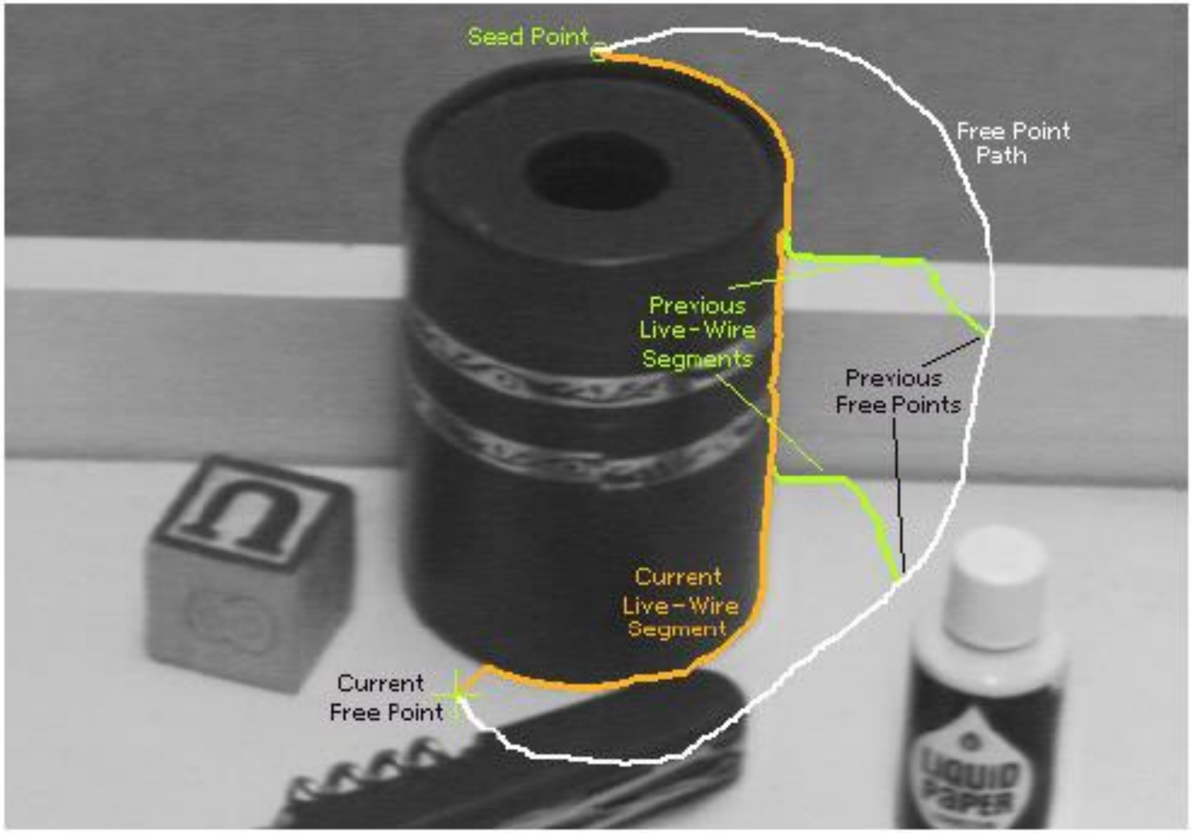


Figure 2: *Image demonstrating how the live-wire segment adapts and snaps to an object boundary as the free point moves (via cursor movement). The path of the free point is shown in white. Live-wire segments from previous free point positions (t_0, t_1, and t_2) are shown in green.*

[Eric Mortensen, William Barrett, 1995]

Intelligent Scissors

- This approach answers a basic question
  - Q: how to find a path from seed to mouse that follows object boundary as closely as possible?
  - A: define a path that stays as close as possible to edges

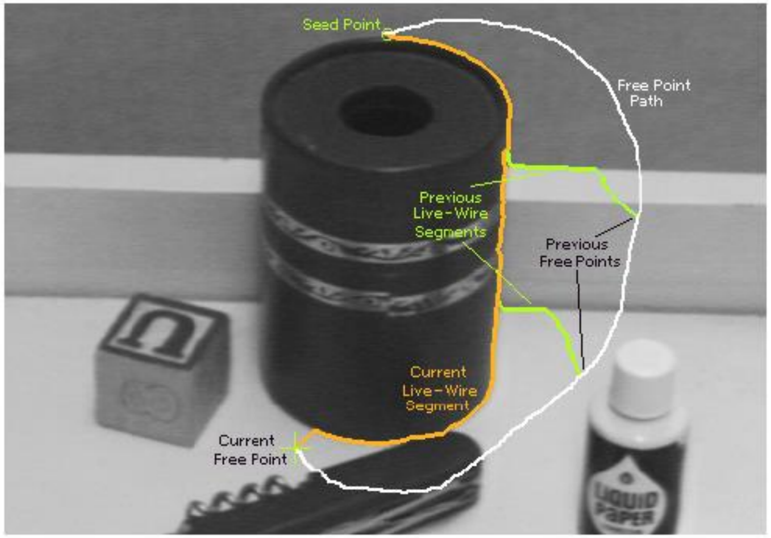


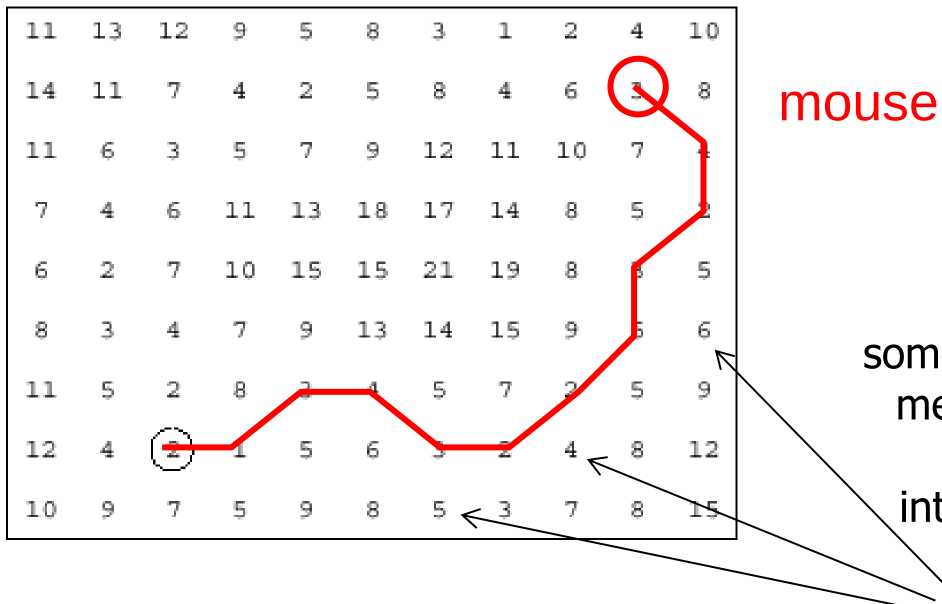
Figure 2: *Image demonstrating how the live-wire segment adapts and snaps to an object boundary as the free point moves (via cursor movement). The path of the free point is shown in white. Live-wire segments from previous free point positions (t_0, t_1, and t_2) are shown in green.*

Intelligent Scissors

- Basic Idea
  - find **lowest cost path** from seed to mouse
    on a graph (*e.g.* N_8 pixel grid) weighted by intensity contrast

simple example:

seed



mouse

some local “contrast” measure based on magnitude of intensity gradient

$$c_p = \frac{25}{1 + |\nabla I_p|}$$

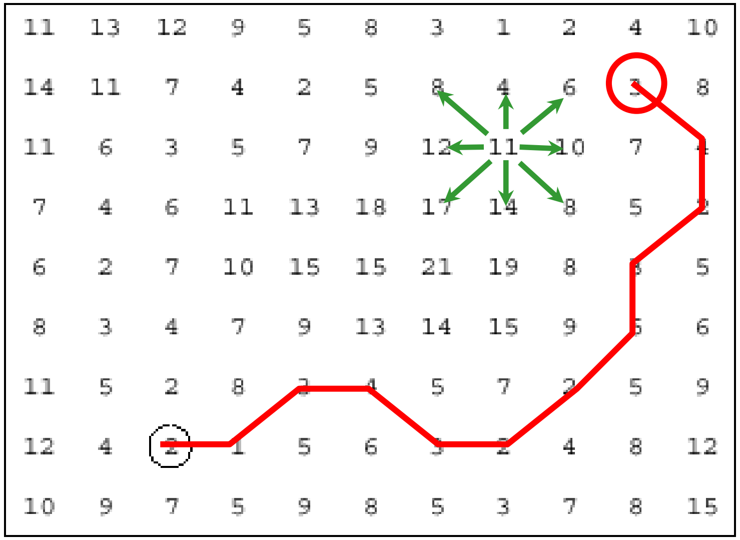
Use node-weighted version of “shortest paths” (Dijkstra)

Shortest Path Search (Dijkstra)

- Computes minimum cost path from the seed to ***all other pixels***

  (once all paths are pre-computed, each path can be instantly shown as mouse moves around)

$$w_{pq} = 11 \quad \forall q$$

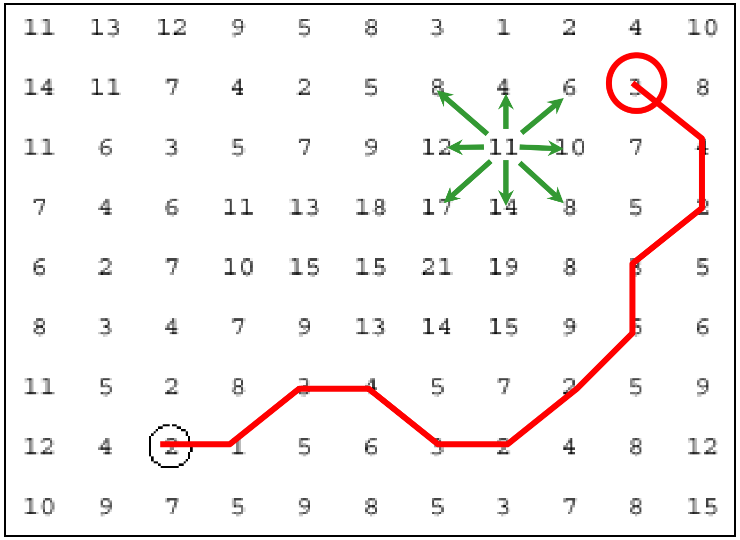


Same as edge-weighted "shortest paths" (Dijkstra) using directed edge weights $w_{pq} = c_p \qquad (w_{qp} = c_q)$

Shortest Path Search (Dijkstra)

- Computes minimum cost path from the seed to ***all other pixels***

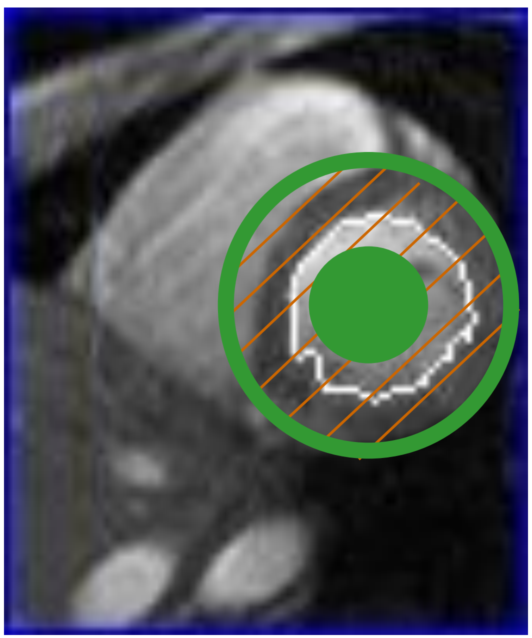
  (once all paths are pre-computed, each path can be instantly shown as mouse moves around)



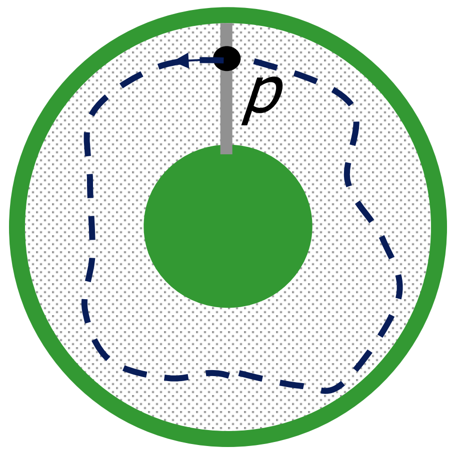
Can also define edge weights w_{pq} directly from intensity contrast across edge pq, e.g. $w_{pq} = \dfrac{25}{1 + \|\nabla I \times \bar{pq}\|}$

Graph cuts vs Shortest paths for 2D segmentation

Example:
find the shortest closed contour on a graph in the shaded domain

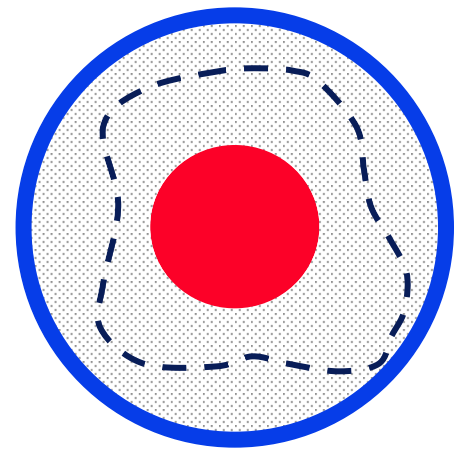


Shortest-path approach



Compute the *shortest path* *p -> p* for a point *p*. Repeat for all points on the gray line. Then choose the optimal contour.

Graph-cut approach

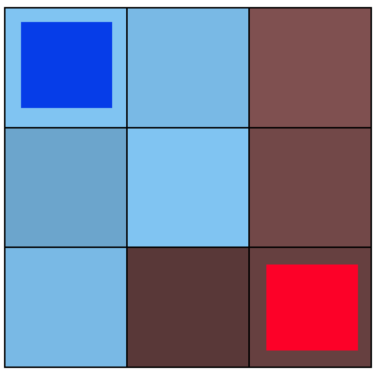


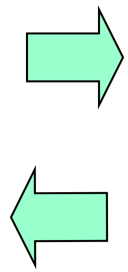
Compute the *minimum cut* that separates red region from blue region

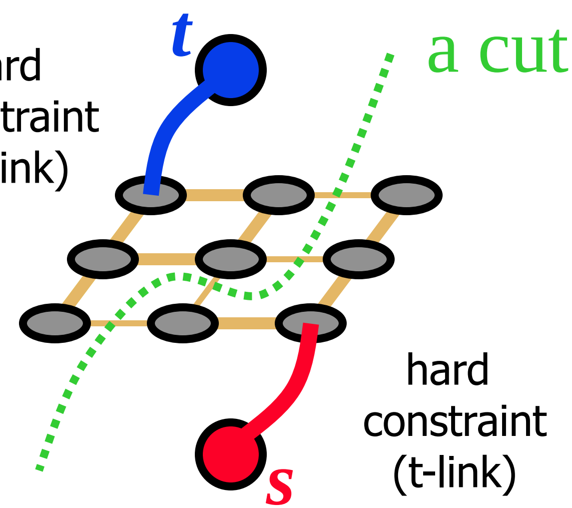
Graph cuts for optimal boundary detection

simple example [*a la* B&J, ICCV'01]

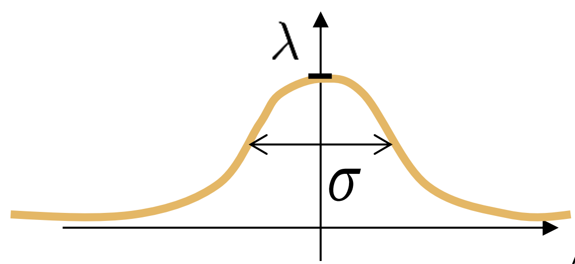
cut's cost = the sum of severed edges weights







$$w_{pq} = \lambda \exp\left\{ - \frac{\|I_p - I_q\|^2}{2\sigma^2} \right\}$$



$$\Delta I_{pq} = I_p - I_q$$

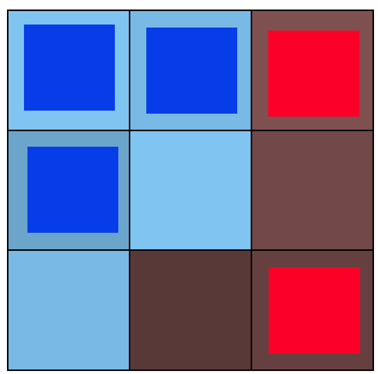
Minimum cost cut can be computed in polynomial time

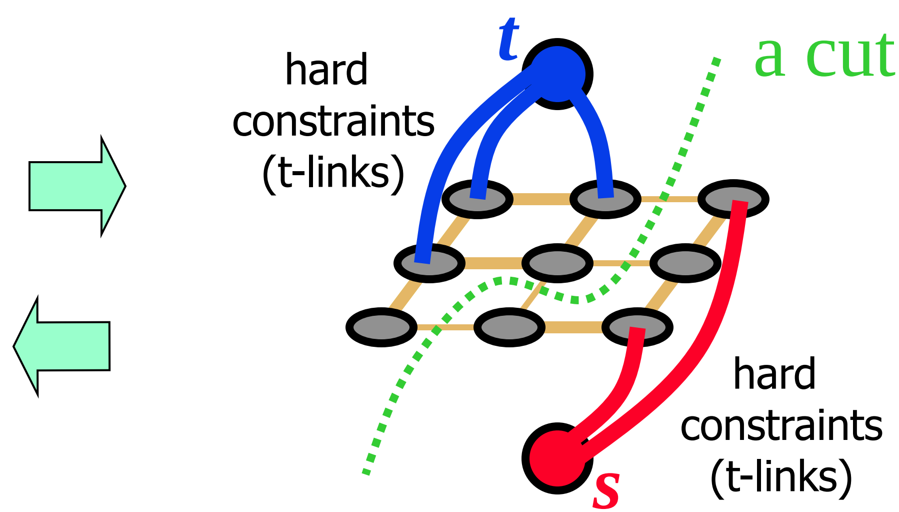
(max-flow/min-cut algorithms)

Graph cuts for optimal boundary detection

simple example [*a la* B&J, ICCV'01]

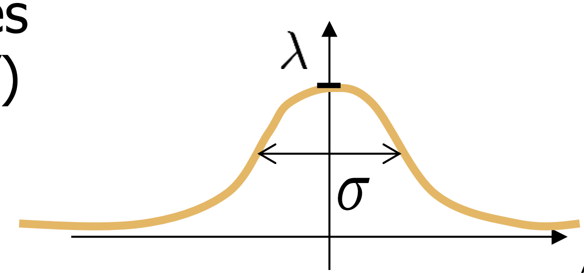
cut's cost = the sum of severed edges weights





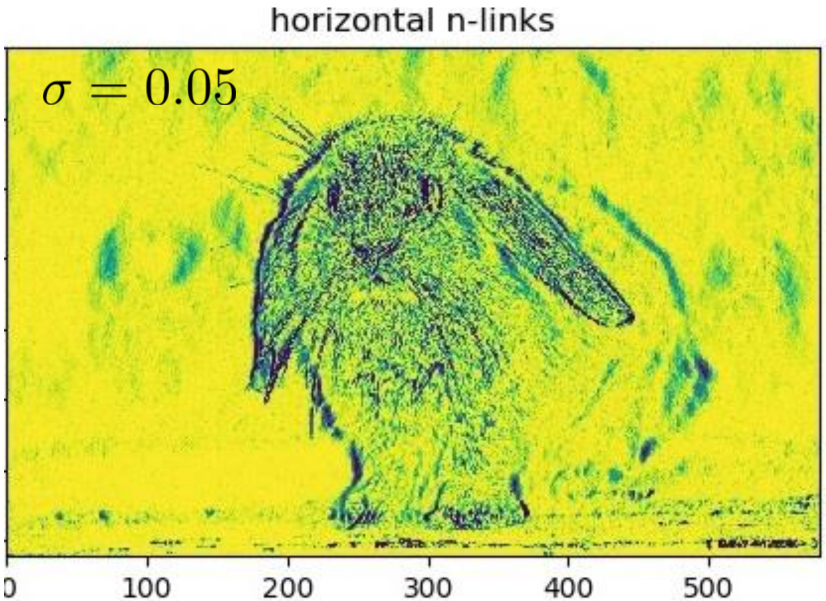
The number of seeds (hard constraints) could be arbitrary - graph cut completes user labeling (*a la* "**Intelligent paint**")

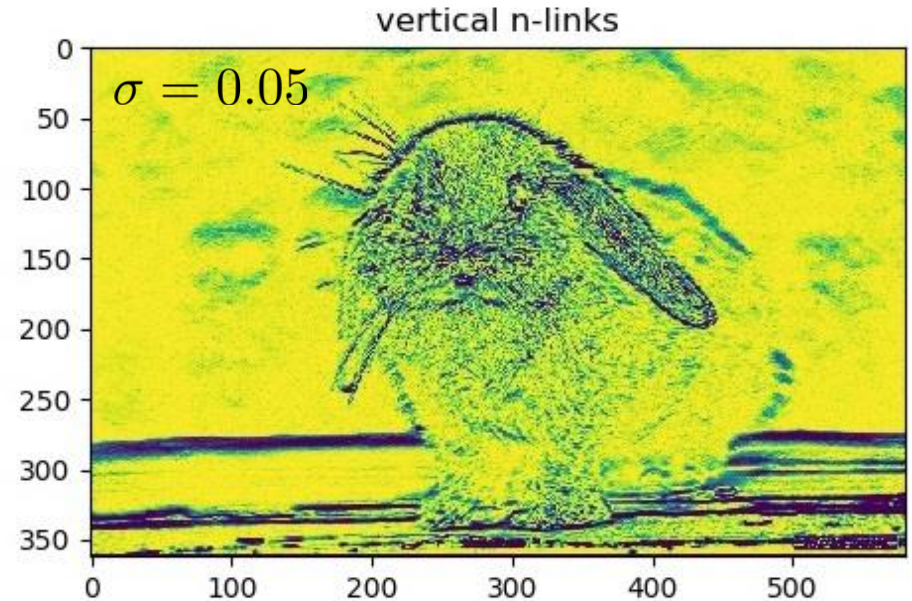
$$w_{pq} = \lambda \exp\left\{ -\frac{\|I_p - I_q\|^2}{2\sigma^2} \right\}$$



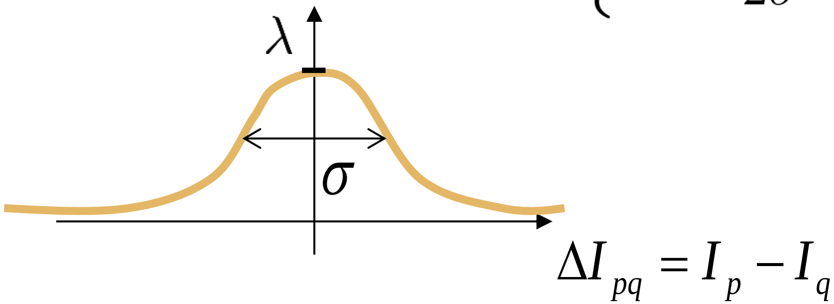
$$\Delta I_{pq} = I_p - I_q$$

Image contrast weighted *n-links*

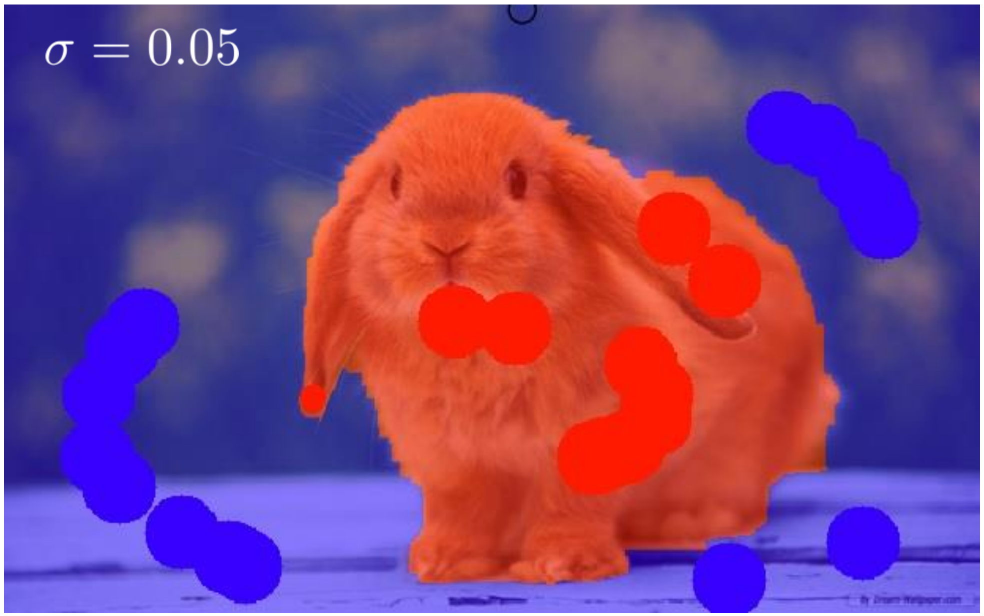




$$w_{pq} = \lambda \exp\left\{-\frac{\|I_p - I_q\|^2}{2\sigma^2}\right\}$$



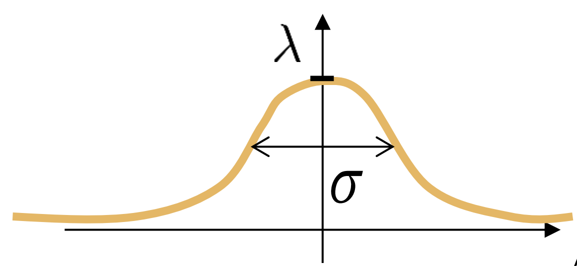
Optimal separation boundary (min cut) in 2D



graph cuts
with hard constrains
and
contrast-weighted n-links

note alignment of
segmentation boundary
with intensity contrast edges
(image edges)

$$w_{pq} = \lambda \exp\left\{-\frac{\|I_p - I_q\|^2}{2\sigma^2}\right\}$$



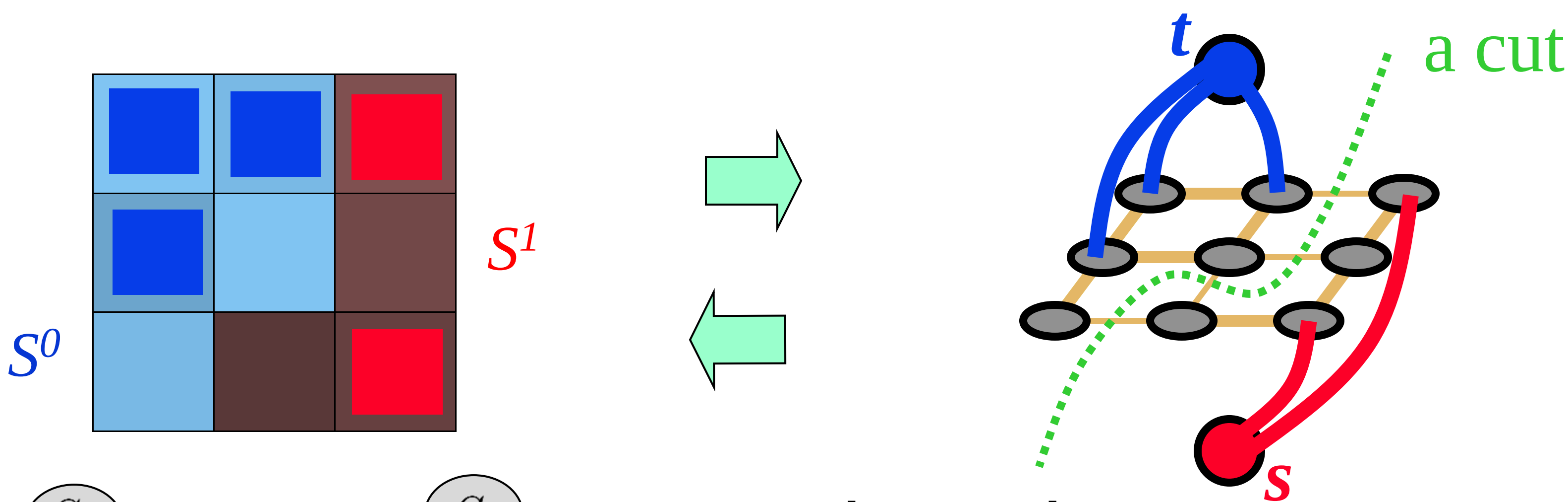
$$\Delta I_{pq} = I_p - I_q$$

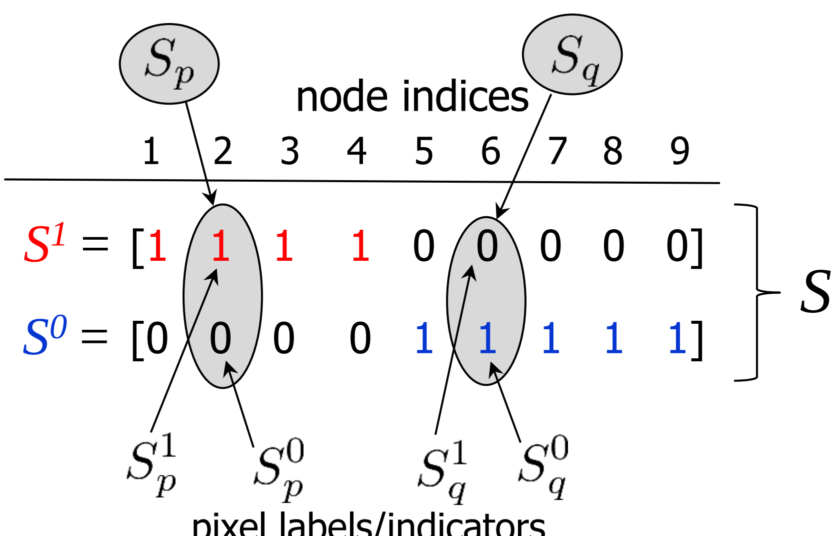
WARNING: "massive" abuse of notation S_p
but most of the time it should be clear from context
if we mean (random) class index or its distribution

s-t graph cut as an example of algorithm for

Loss optimization

cut's cost = the sum of severed edges weights





First question:

How can one represent segmentation as variables? (remember K-means)

$$S_p = \begin{pmatrix} S_p^1 \\ S_p^0 \end{pmatrix} \in \Delta^2$$

categorical distribution (includes *one-hot* case Δ_v^2)

OR

$$S_p \in \{0, 1\}$$

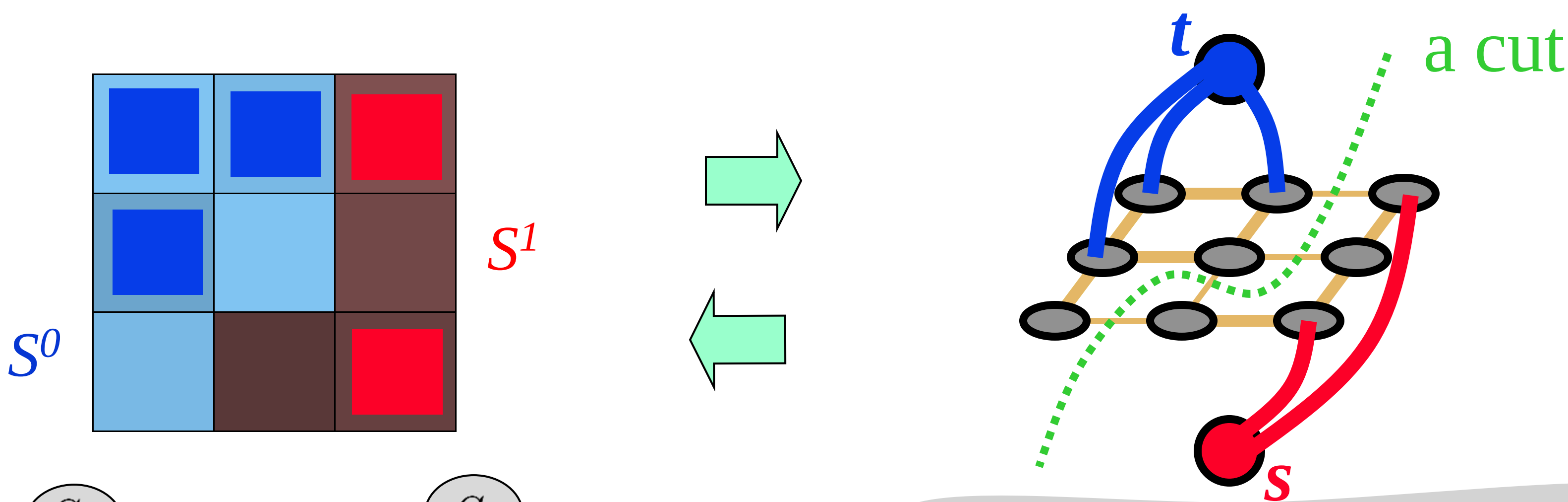
discrete label (*i.e.* corresponding random variable)

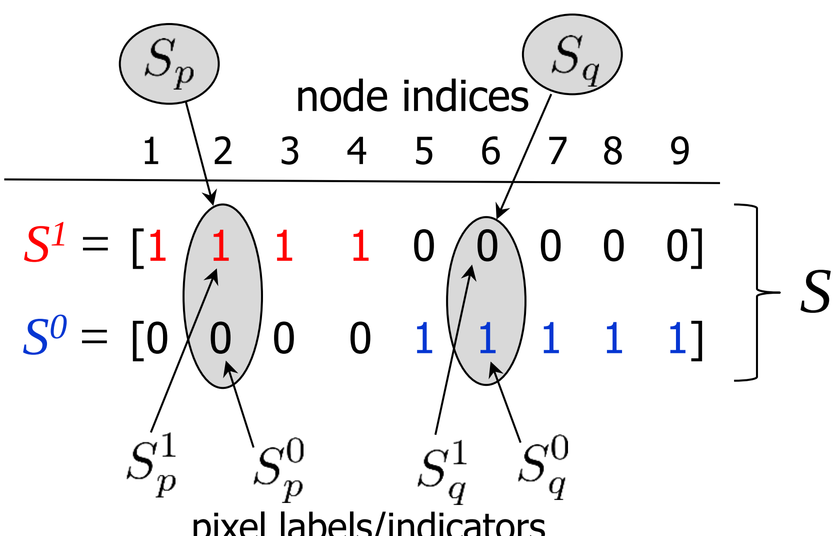
binary case, easily generalizes to $K \geq 2$ categories

s-t graph cut as an example of algorithm for

Loss optimization

cut's cost = the sum of severed edges weights





loss encouraging smooth segmentation boundary aligned with contrast edges

$$\sum_{pq \in N} w_{pq} \, [S_p \neq S_q]$$

cost of severed n-links

$$[x] := \begin{cases} 1 & \text{if } x = True \\ 0 & \text{if } x = False \end{cases}$$

Iverson brackets

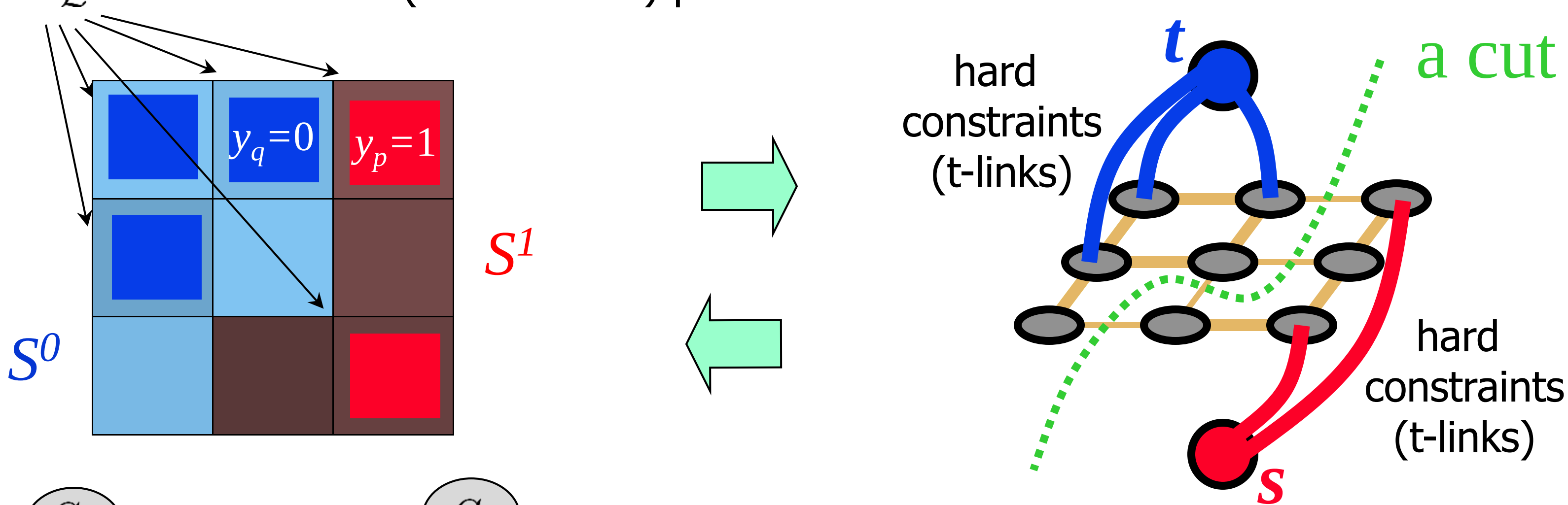
s-t graph cut as an example of algorithm for

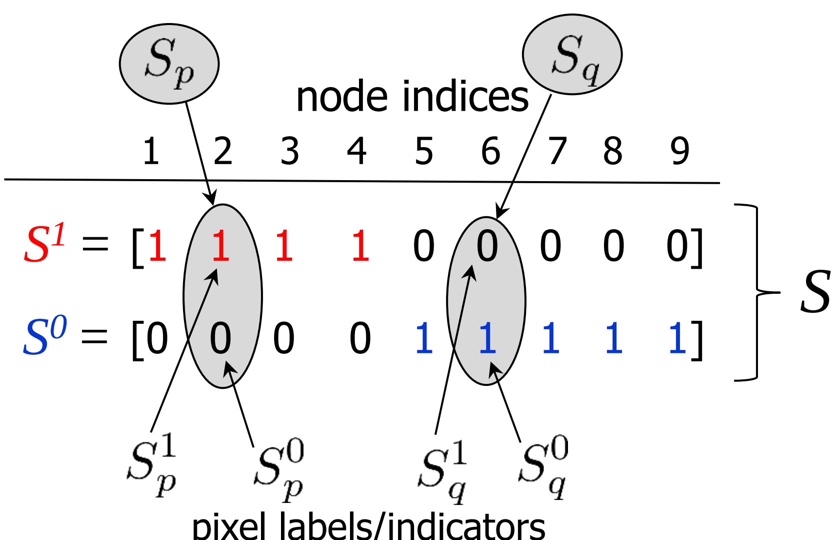
Loss optimization

$\{y_p \,|\, p \in \Omega_{\mathcal{L}}\}$ - seed labels (ground truth)

$\Omega_{\mathcal{L}}$ - set of seeded (user-labeled) pixels

cut's cost = the sum of severed edges weights





Q: What loss function can represent consistency of S with user labels?

$$\sum_{pq \in N} w_{pq}\,[S_p \neq S_q]$$

cost of severed n-links

$$[x] := \begin{cases} 1 & \text{if } x = True \\ 0 & \text{if } x = False \end{cases}$$

Iverson brackets

Supervision loss:
consistency with ground truth labels y_p

Similar "NLL" loss is commonly used for supervised training of neural networks (topics 10,11,12)

UNIVERSITY OF WATERLOO

For generality, assume K classes and ground truth label $y_p \in \{1, \dots, K\}$

$$S_p = \begin{pmatrix} S_p^1 \\ \dots \\ S_p^k \\ \dots \\ S_p^K \end{pmatrix} \quad \xrightarrow{y_p = k} \quad \begin{pmatrix} 0 \\ 0 \\ 1 \\ 0 \\ 0 \end{pmatrix} \quad \Longleftrightarrow \quad \boxed{S_p^{y_p} = 1}$$

S_p segmentation variable

probability that pixel p belongs to category k

categorical distribution over K classes at point p

one-hot distribution consistent with given ground truth label value $y_p = k$

probability that pixel p belongs to category y_p

constraint for S_p that represents known ground truth label y_p

How to enforce supervision constraint $S_p^{y_p} = 1$?

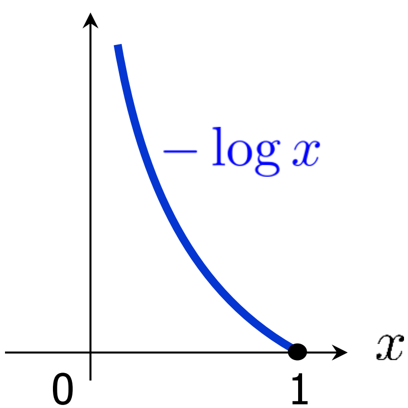
Standard supervision (or seed) **loss:** $\boxed{-\log(S_p^{y_p})}$

that works for discrete or relaxed segmentation

$S_p^k \in \{0, 1\}$ $\quad$ $S_p^k \in [0, 1]$

$\underbrace{S_p^{y_p}}_{\Pr(S_p = y_p)}$

here S_p is interpreted as random variable



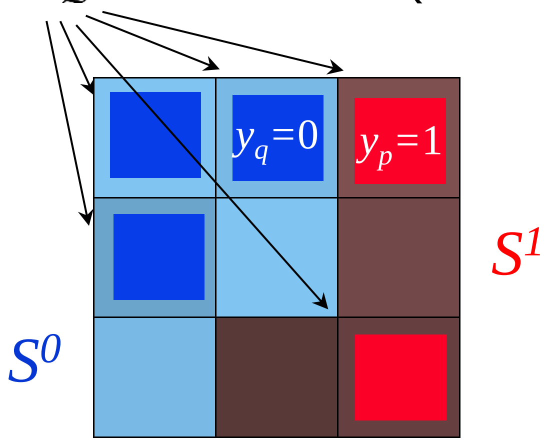
s-t graph cut as an example of algorithm for

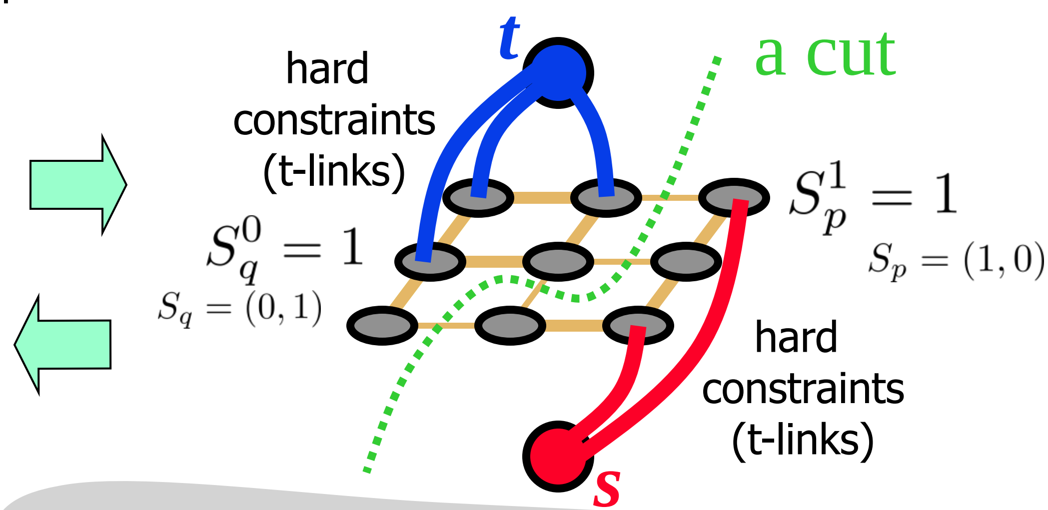
Loss optimization

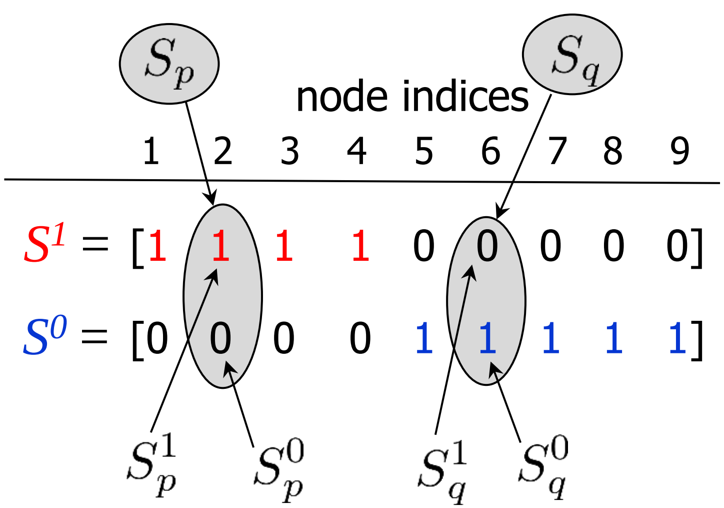
$\{y_p \,|\, p \in \Omega_{\mathcal{L}}\}$ - seed labels (ground truth)

$\Omega_{\mathcal{L}}$ - set of seeded (user-labeled) pixels

cut's cost = the sum of severed edges weights







Seed loss enforcing consistency of S with user labels

$$\sum_{p\in\Omega_{\mathcal{L}}} -\log S_p^{y_p} \quad + \quad \sum_{pq\in N} w_{pq}\,[S_p \neq S_q]$$

cost of severed t-links

$$-\log x := \begin{cases} 0 & \text{if } x = 1 \\ \infty & \text{if } x = 0 \end{cases}$$

cost of severed n-links

$$[x] := \begin{cases} 1 & \text{if } x = True \\ 0 & \text{if } x = False \end{cases}$$

Iverson brackets

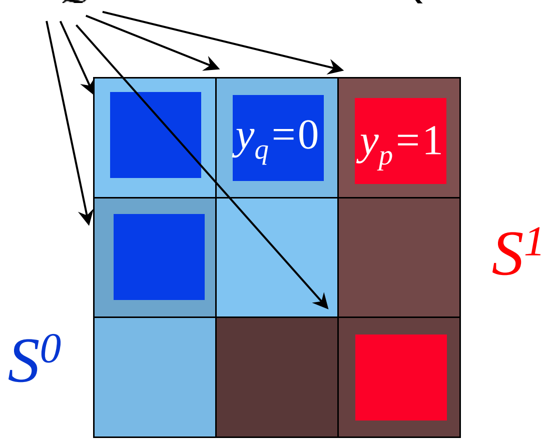
s-t graph cut as an example of algorithm for

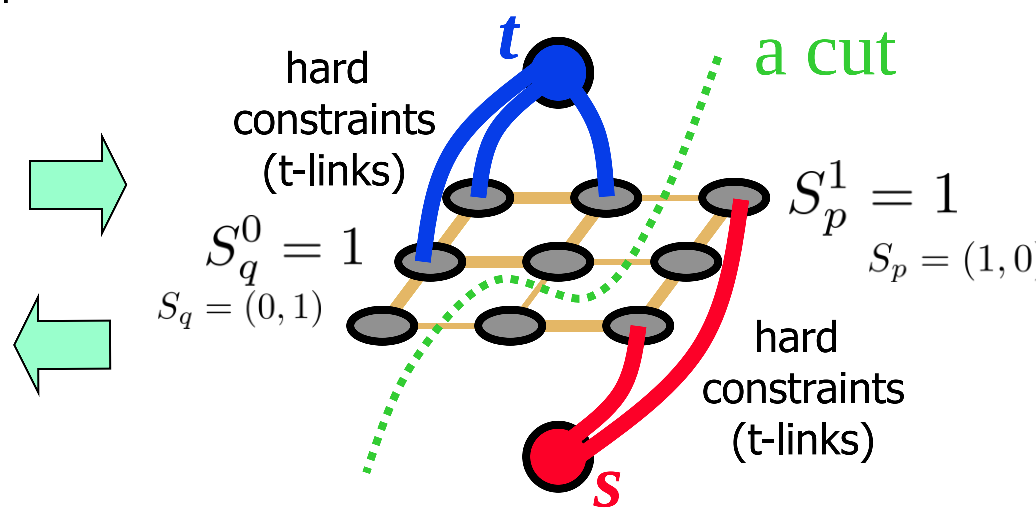
Loss optimization

$\{y_p \,|\, p \in \Omega_{\mathcal{L}}\}$ - seed labels (ground truth)

$\Omega_{\mathcal{L}}$ - set of seeded (user-labeled) pixels

cut's cost = the sum of severed edges weights





node indices

	1	2	3	4	5	6	7	8	9
S^1 =	[1	1	1	1	0	0	0	0	0]
S^0 =	[0	0	0	0	1	1	1	1	1]
		S_p				S_q			

pixel "beliefs" about two classes

cost of any cut $\{S^1, S^0\}$

$$\sum_{p \in \Omega_{\mathcal{L}}} -\log S_p^{y_p} \quad + \quad \sum_{pq \in N} w_{pq}\,[S_p \neq S_q]$$

cost of severed t-links — **cost of severed n-links**

$$-\log x := \begin{cases} 0 & \text{if } x = 1 \\ \infty & \text{if } x = 0 \end{cases}$$

$$[x] := \begin{cases} 1 & \text{if } x = True \\ 0 & \text{if } x = False \end{cases}$$

Iverson brackets

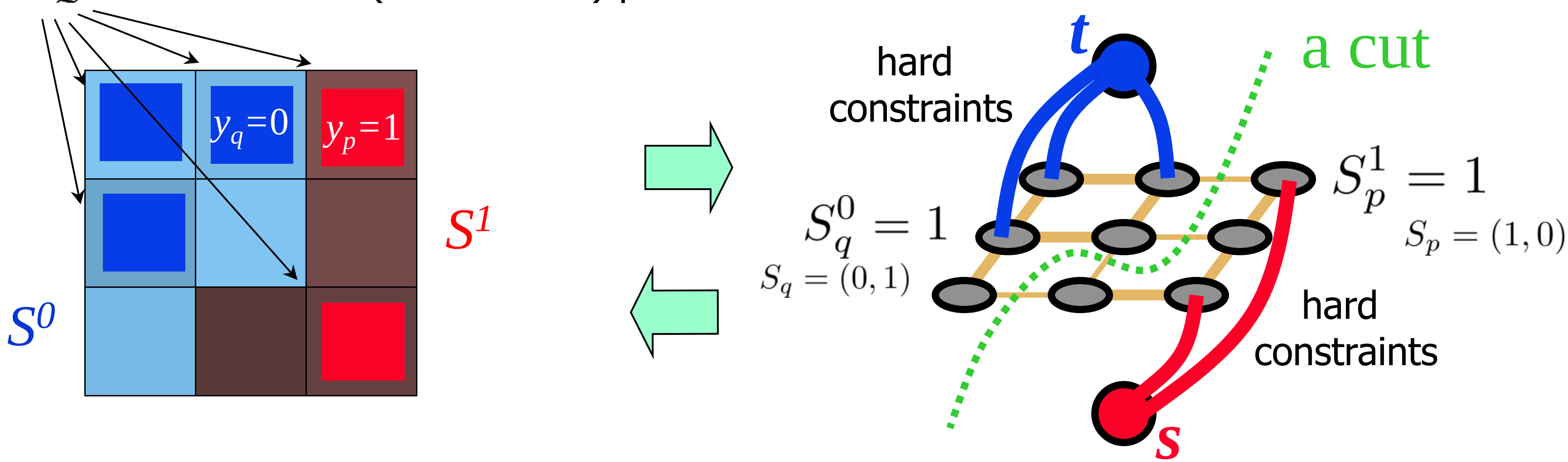
s-t graph cut as an example of algorithm for

Loss optimization

$\{y_p \,|\, p \in \Omega_{\mathcal{L}}\}$ - seed labels (ground truth)

$\Omega_{\mathcal{L}}$ - set of seeded (user-labeled) pixels

cut's cost = the sum of severed edges weights



node indices

	1	2	3	4	5	6	7	8	9
S^1 =	[1	1	1	1	0	0	0	0	0]
S^0 =	[0	0	0	0	1	1	1	1	1]
		S_p				S_q			

pixel "beliefs" about two classes

minimum cut outputs S optimizing total loss $L(S)$

$$\sum_{p\in\Omega_{\mathcal{L}}} -\log S_p^{y_p} \quad + \quad \sum_{pq\in N} w_{pq}\,[S_p \neq S_q]$$

penalty for *inconsistency with seeds*

seed loss - special case of negative log-likelihood loss (**NLL loss**)

cost of segmentation boundary

example of **regularization loss** (shape smoothness, contrast alignment)

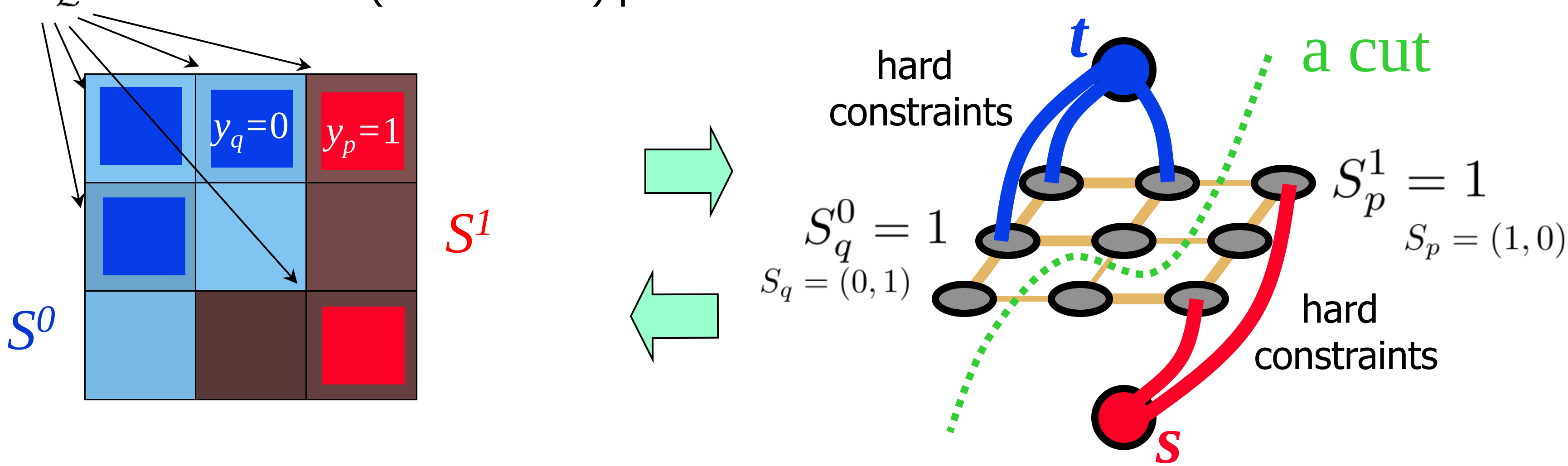
s-t graph cut as an example of algorithm for

Loss optimization

$\{y_p \mid p \in \Omega_{\mathcal{L}}\}$ - seed labels (ground truth)

cut's cost = the sum of severed edges weights

$\Omega_{\mathcal{L}}$ - set of seeded (user-labeled) pixels



node indices

	1	2	3	4	5	6	7	8	9
S^1 =	[1	1	1	1	0	0	0	0	0]
S^0 =	[0	0	0	0	1	1	1	1	1]
		S_p				S_q			

pixel "beliefs" about two classes

total loss $L(S)$

relaxations are also used, e.g. weakly-supervised segmentation networks

$$\sum_{p\in\Omega_{\mathcal{L}}} -\log S_p^{y_p} \;+\; \sum_{pq\in N} w_{pq}\|S_p - S_q\|^2$$

penalty for *inconsistency with seeds*

seed loss - special case of negative log-likelihood loss (**NLL loss**)

cost of segmentation boundary

example of **regularization loss** (shape smoothness, contrast alignment)

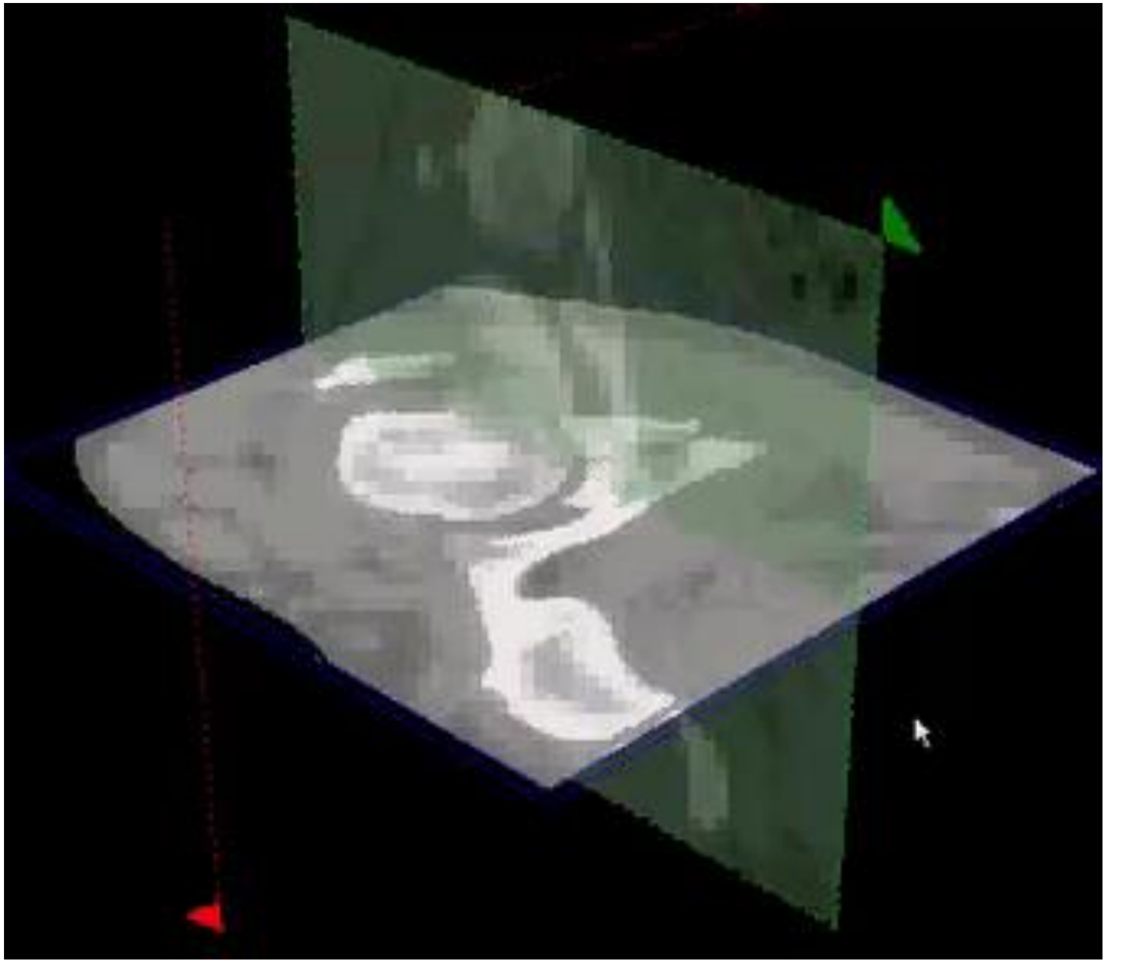
Unlike shortest paths, graph cut works for 3D segmentation:

Optimal separation boundary (min cut) in 3D

Extra correcting seeds can be added **interactively** and new optimal cut will respect them

(new cut respecting extra constraints is faster due to hot start in the algorithm)

Example where
minimum cut representing minimal surface
(surface regularization)

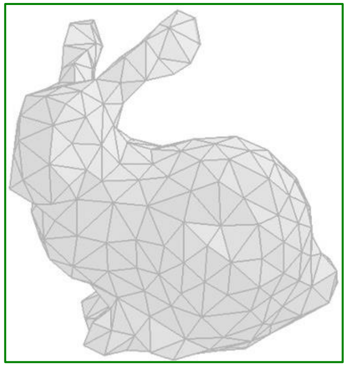


3D bone segmentation
(real time screen capture from early 2000)

Some standards methodologies for

Surface Representation and Regularization

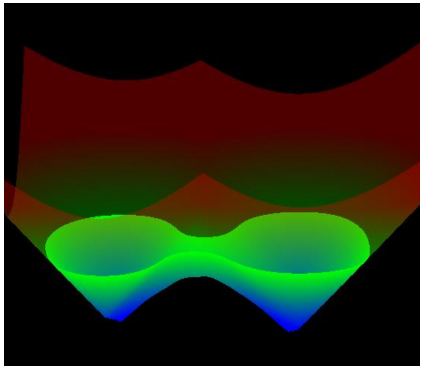
mesh, spline



explicit representation of surface/boundary

continuous variables S_p explicitly represent surface locations

level-set

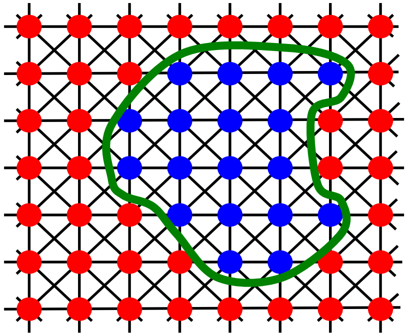


implicit representation of surface/boundary

surface is a zero-level set of continuous function $f(x,y)$

$S = \{ (x,y) : f(x,y)=0 \}$

grid labeling



Graph cut is just one discrete approach to optimizing labels S_p for boundary regularization.

Many alternatives also use relaxed indicator variable S_p

implicit representation of surface/boundary

surface is an implicit interface between subsets or segments represented by set/class/object indicators S_p (**discrete** or **relaxed**)

Surface representation dictates specific optimization methodology, but common **surface regularization objectives** are closely related: typically, they **minimize surface area and/or curvature**

active contours: elasticity and bending (physics)

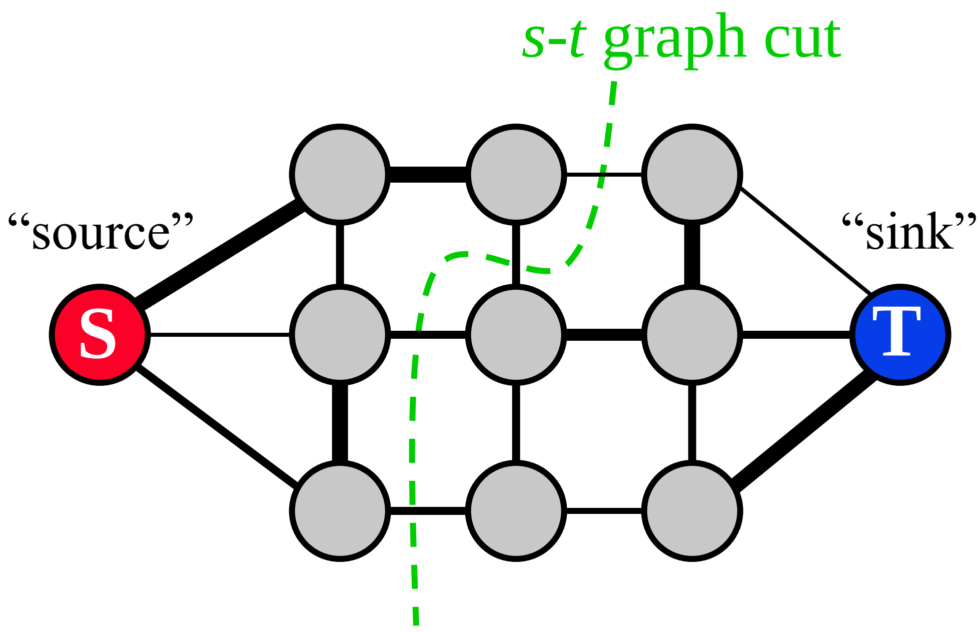
geodesic active contours: surface area and curvature (Riemannian geometry)

graphical models, MRF/CRF: pairwise and higher-order smoothness (relates to minimal surfaces via *integral geometry*)

Graph Cuts Basics (see Cormen's book)

Simple 2D example

Goal: divide the graph into two parts separating red and blue nodes



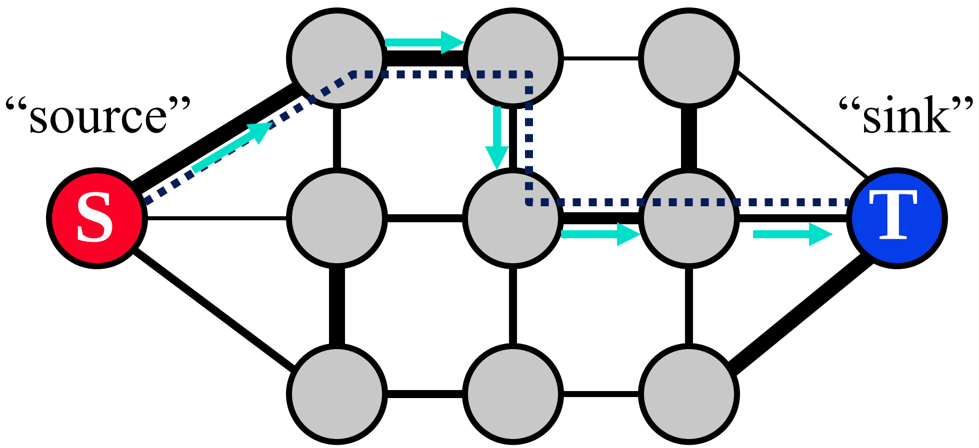
A graph with two terminals *S* and *T*

- Cut cost is a sum of severed edge weights
- **Minimum cost *s-t* cut can be found in polynomial time**

s/t min cut algorithms are widely studied (combinatorial optimization)

- Augmenting paths [Ford & Fulkerson, 1962]
- Push-relabel [Goldberg-Tarjan, 1986]

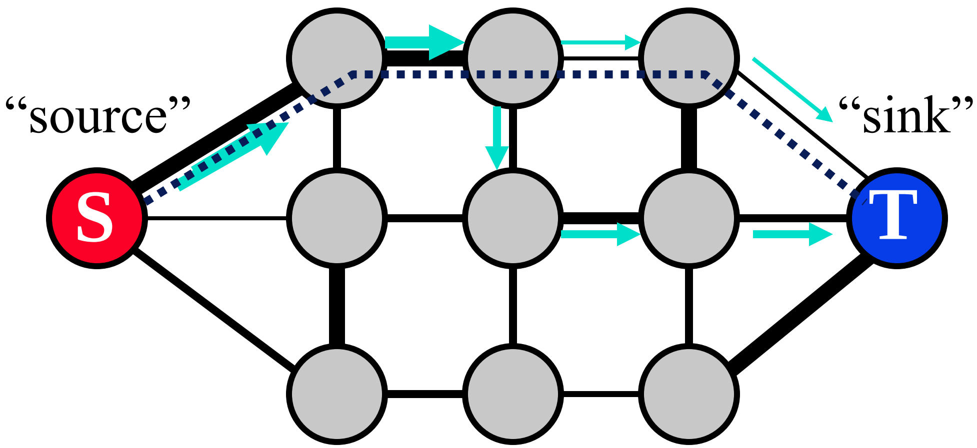
“Augmenting Paths”



A graph with two terminals

- Find a path from S to T along non-saturated edges
- Increase flow along this path until some edge saturates

“Augmenting Paths”

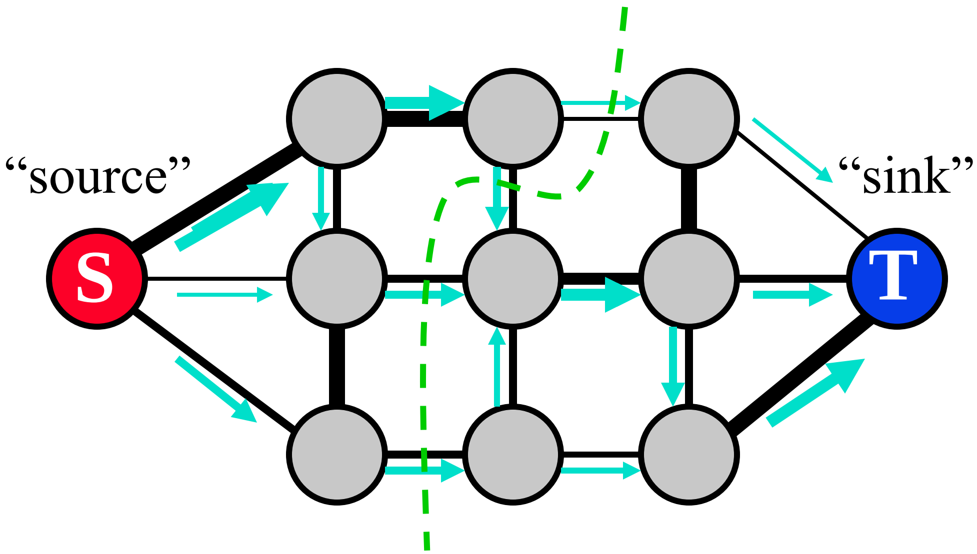


A graph with two terminals

- Find a path from S to T along non-saturated edges
- Increase flow along this path until some edge saturates

- Find next path…
- Increase flow…

"Augmenting Paths"



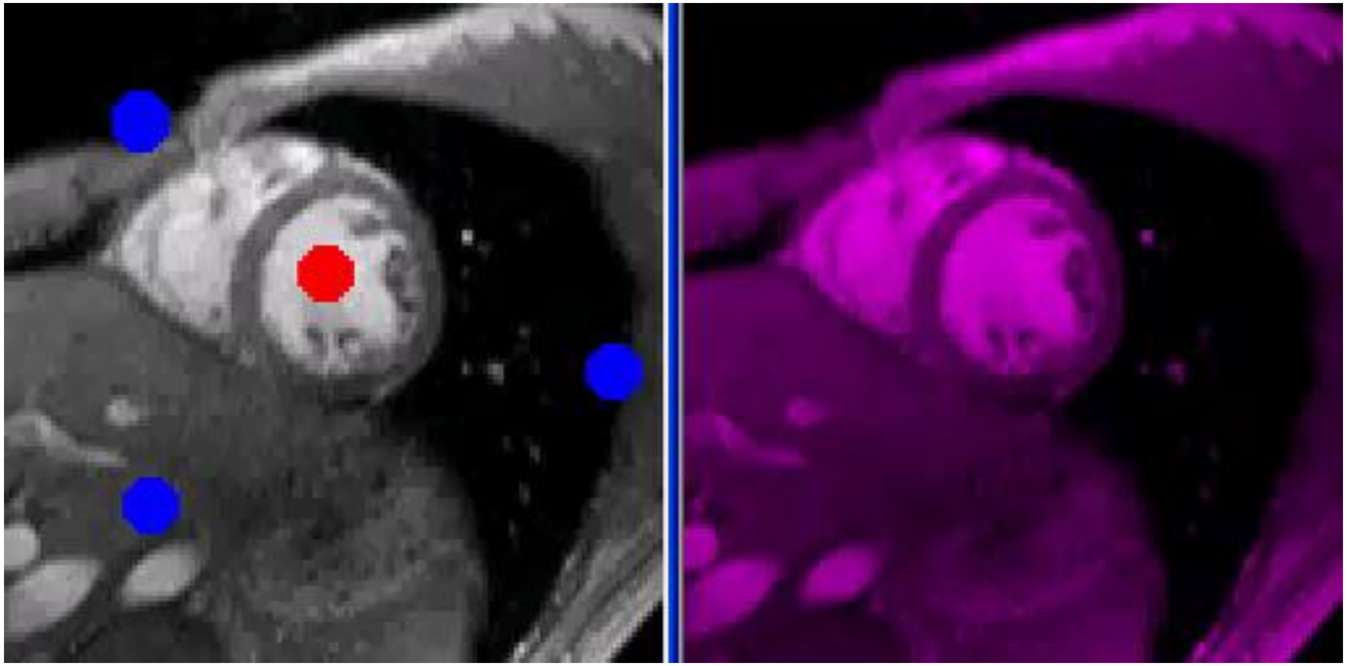
A graph with two terminals

MAX FLOW ⇔ MIN CUT

- Find a path from S to T along non-saturated edges
- Increase flow along this path until some edge saturates

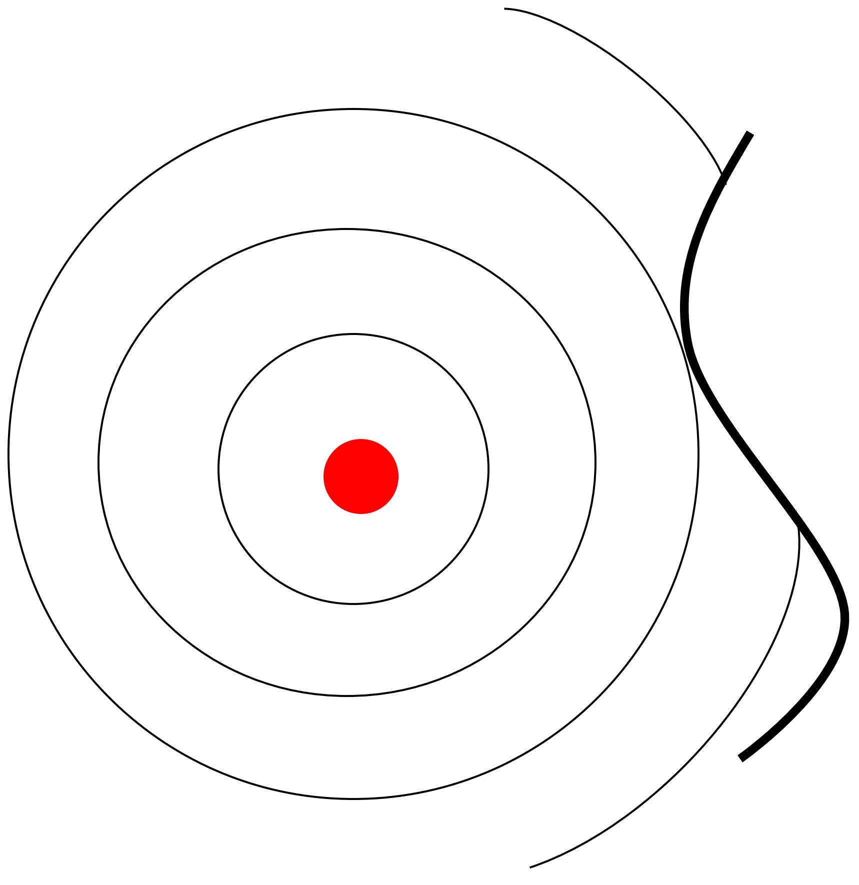
Iterate until … all paths from S to T have at least one saturated edge

Optimal boundary in 2D



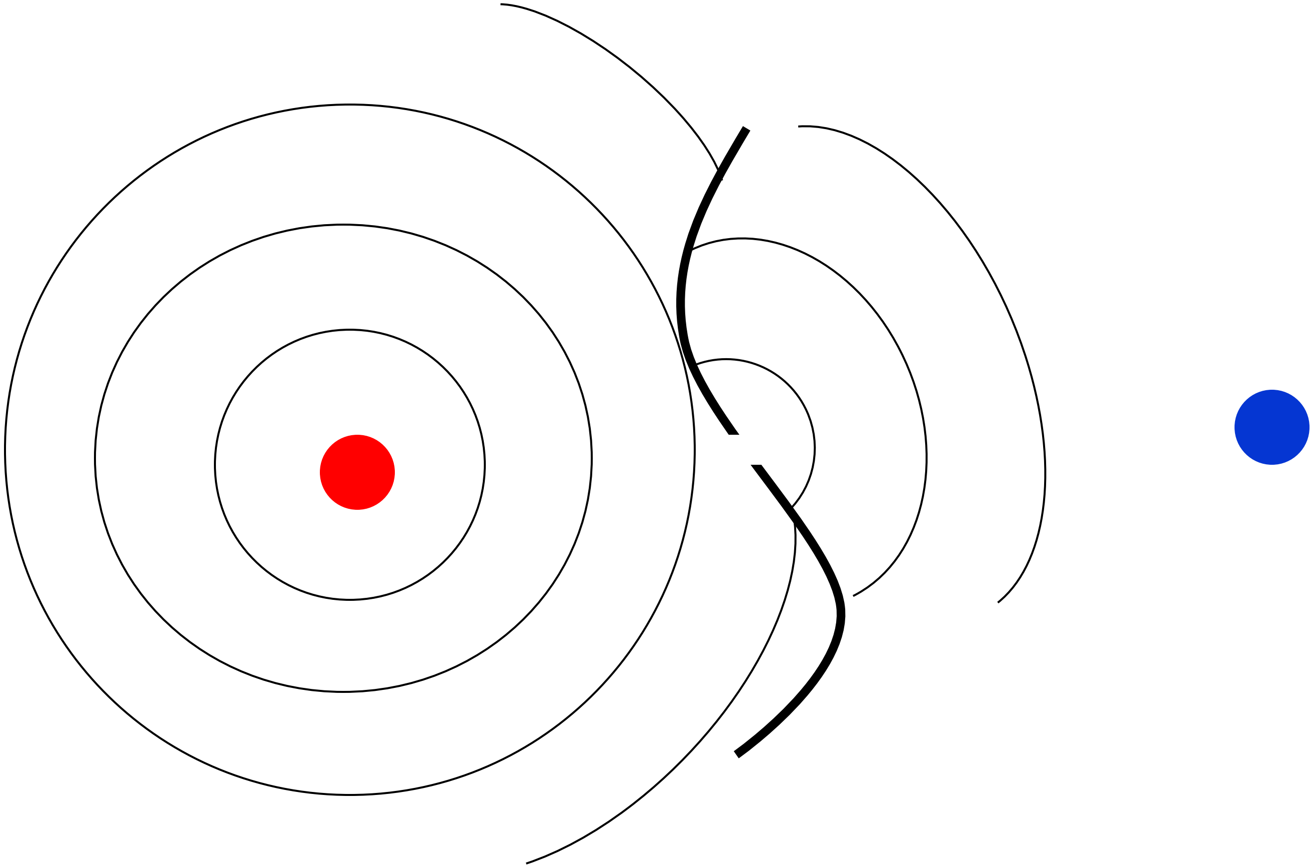
“min-cut = max-flow”

Graph cuts vs Region Growing



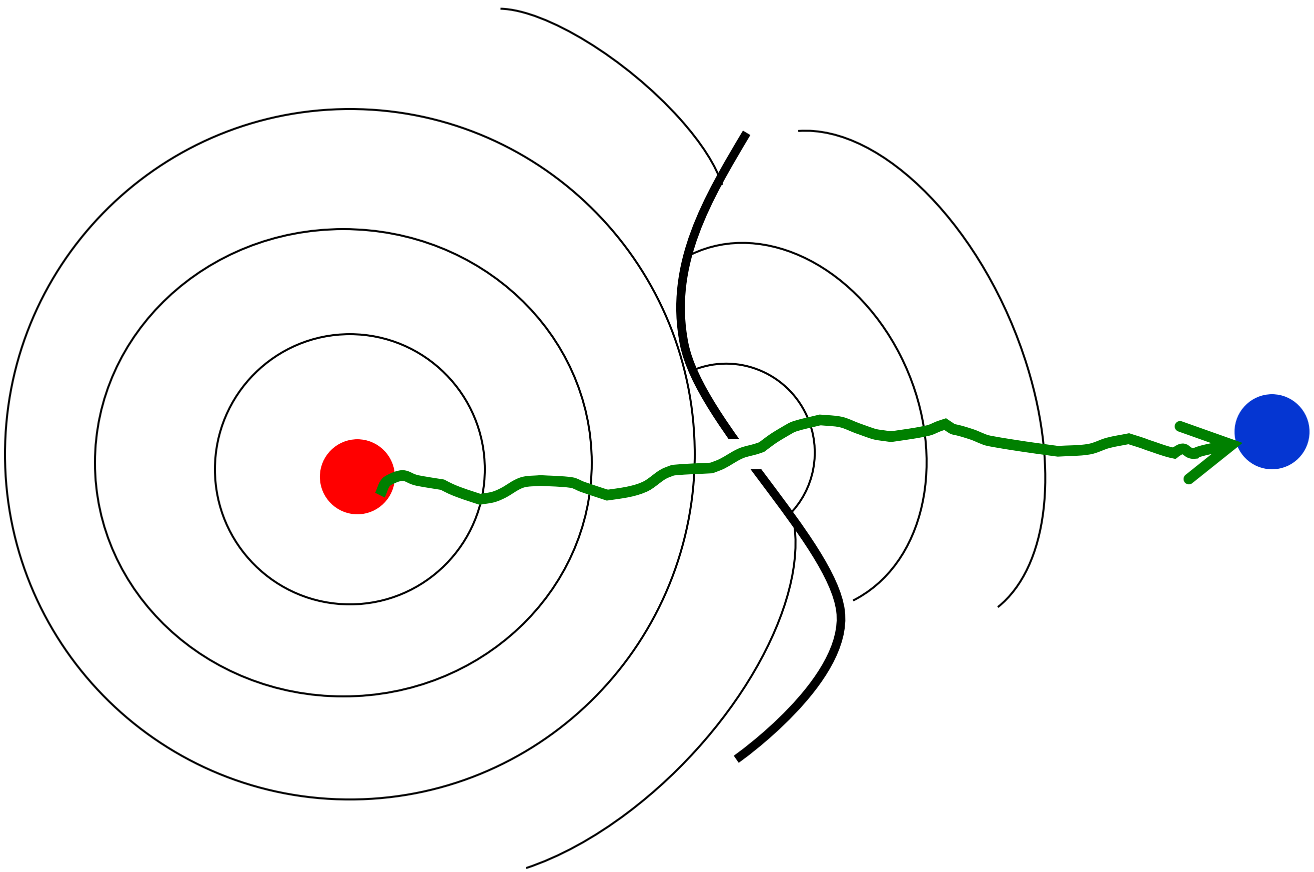
like “region growing”

Graph cuts vs Region Growing



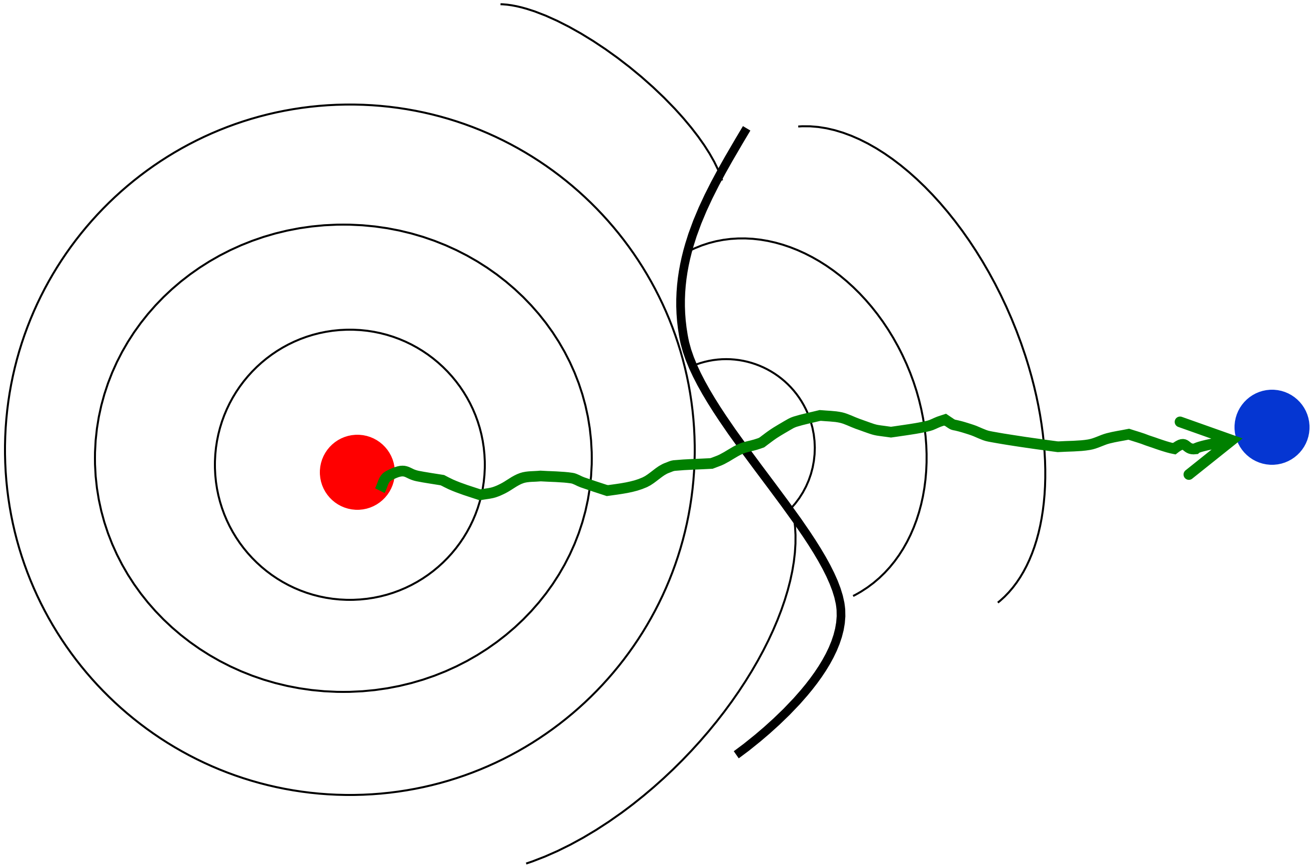
like "region growing"

Graph cuts vs Region Growing



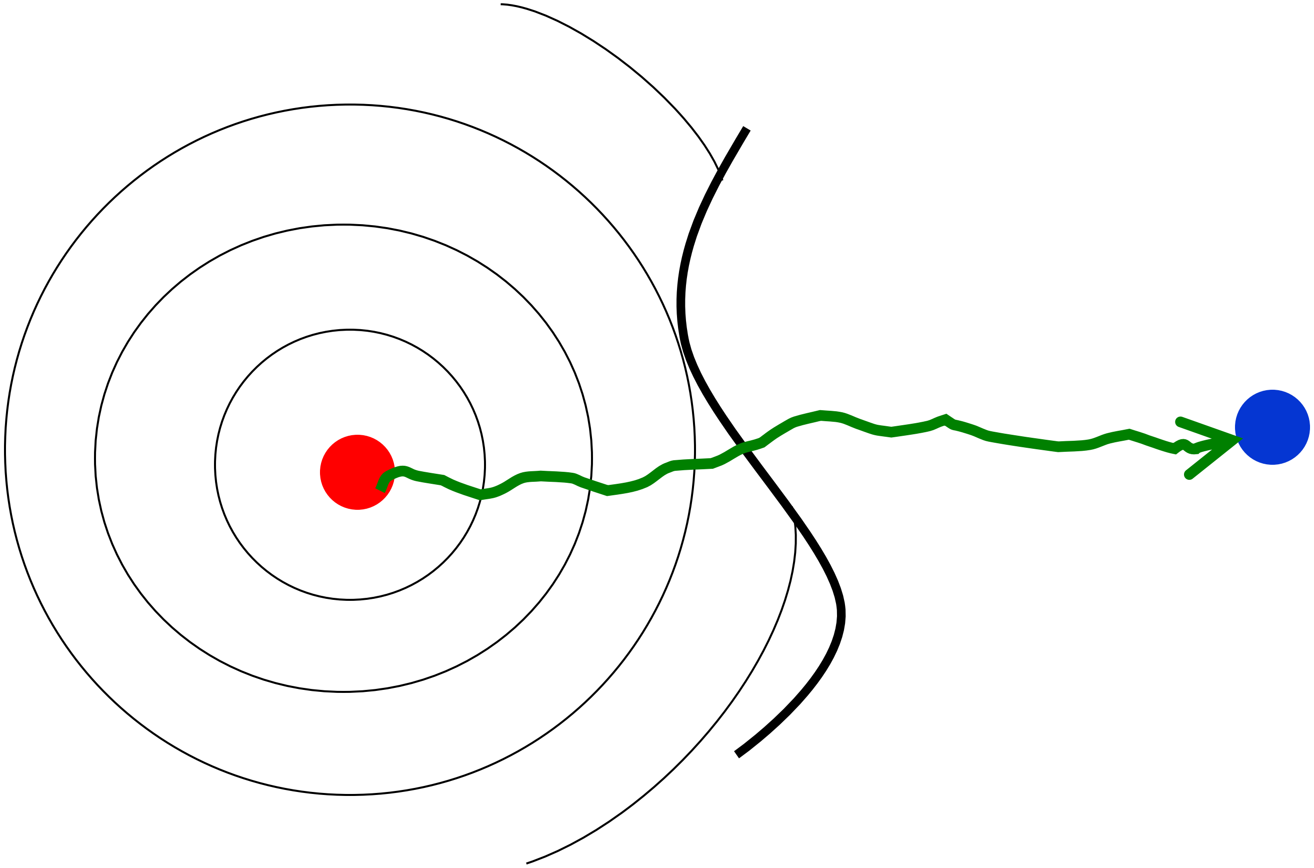
like "region growing"

Graph cuts vs Region Growing



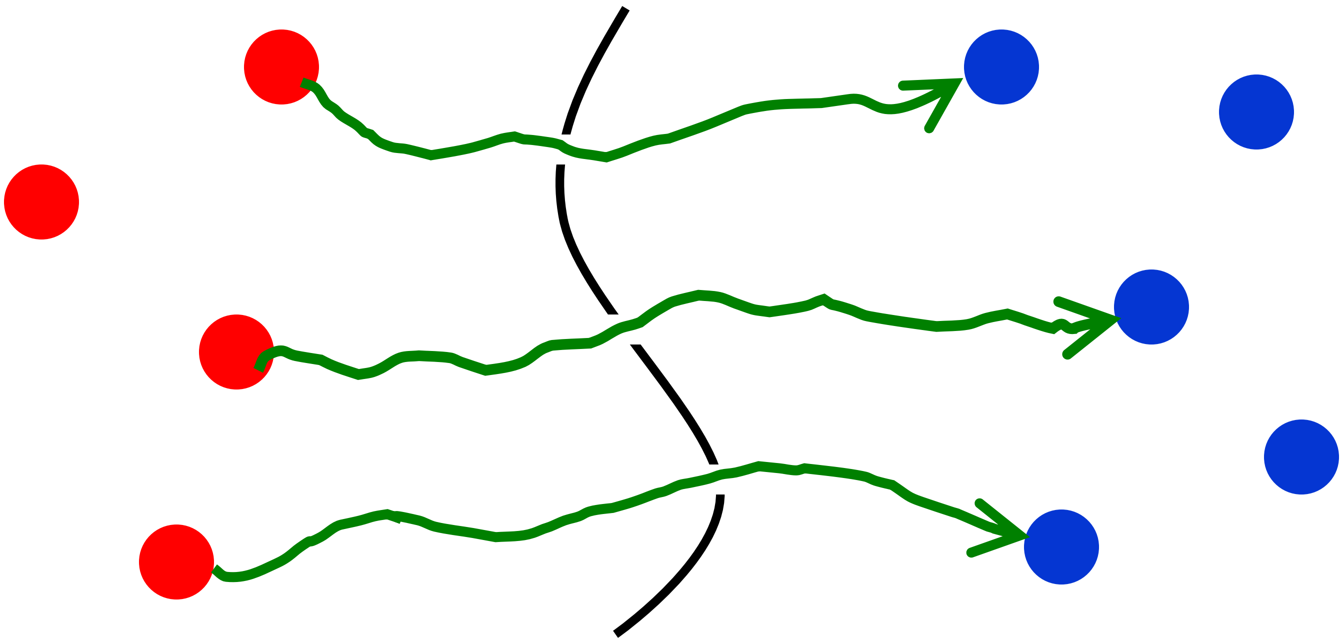
~~like "region growing"~~

Graph cuts

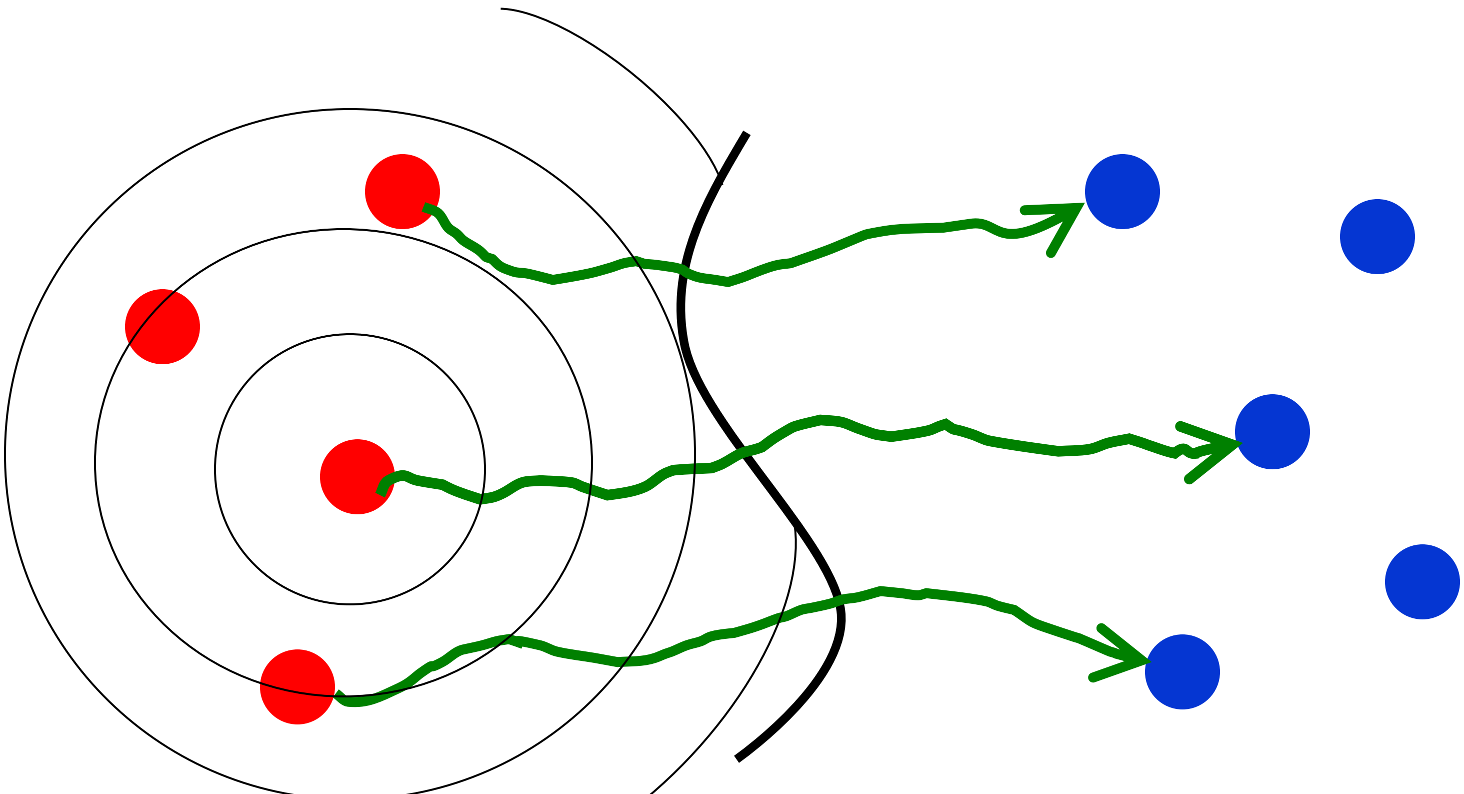


iteration 2

Graph cuts 2

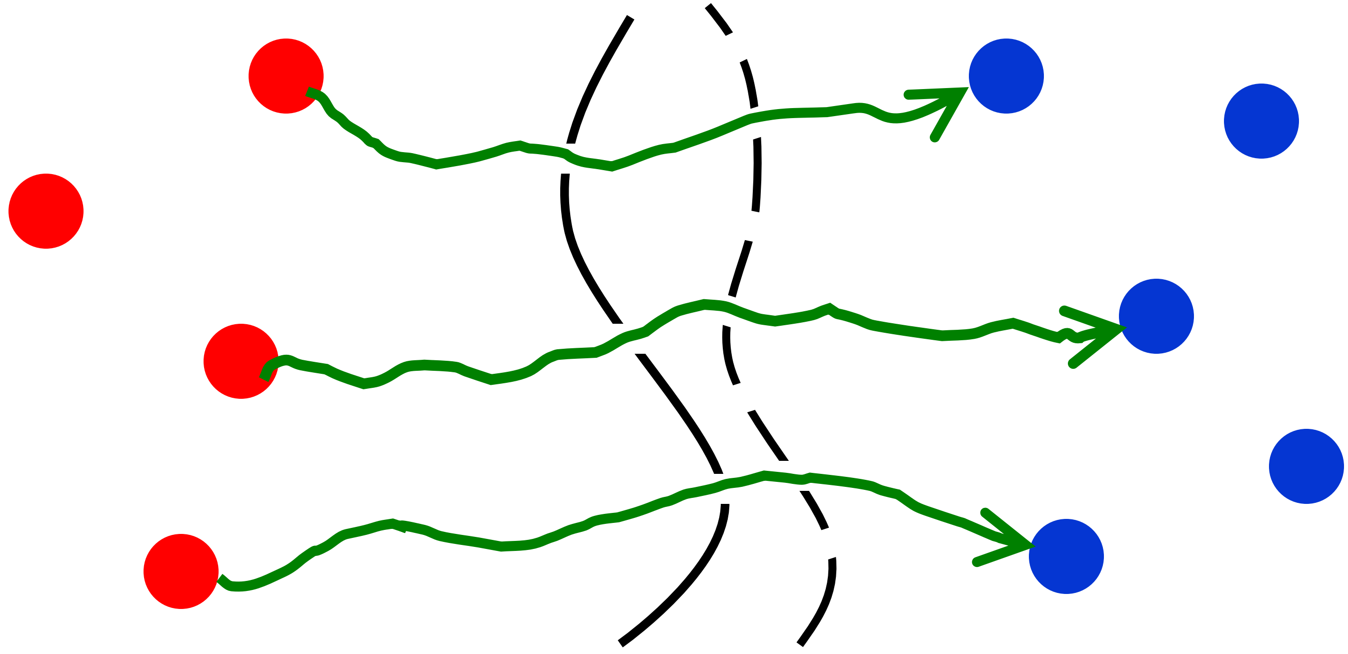


Graph cuts 2



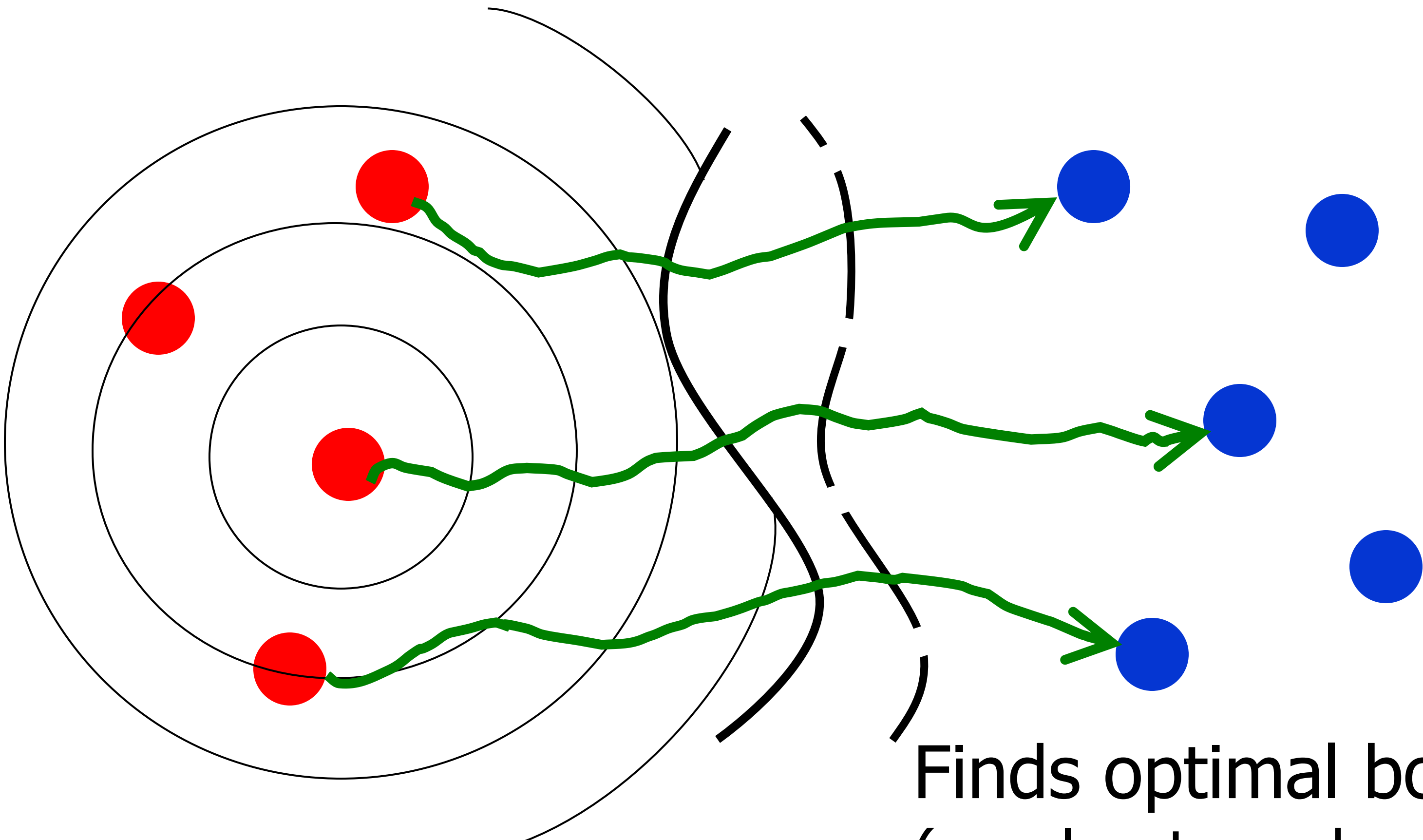
Any paths would work, but **shorter paths** give faster algorithms (in theory and practice)

Graph cuts 3



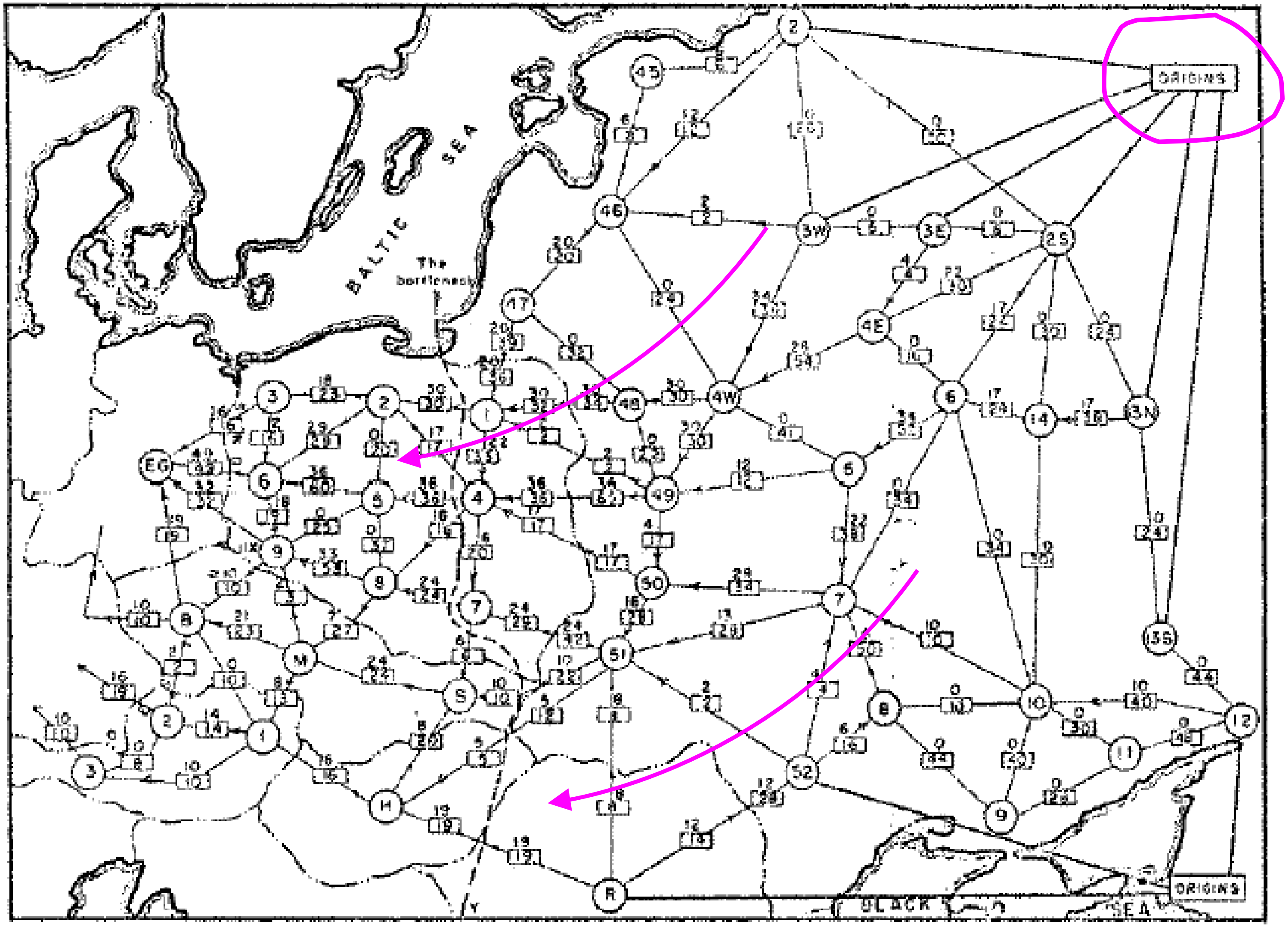
UNIVERSITY OF WATERLOO

Graph cuts 3



Finds optimal boundary
(e.g. least number of *holes*)
best value of a
an objective function

Graph cut is an old standard problem with lots of applications outside vision

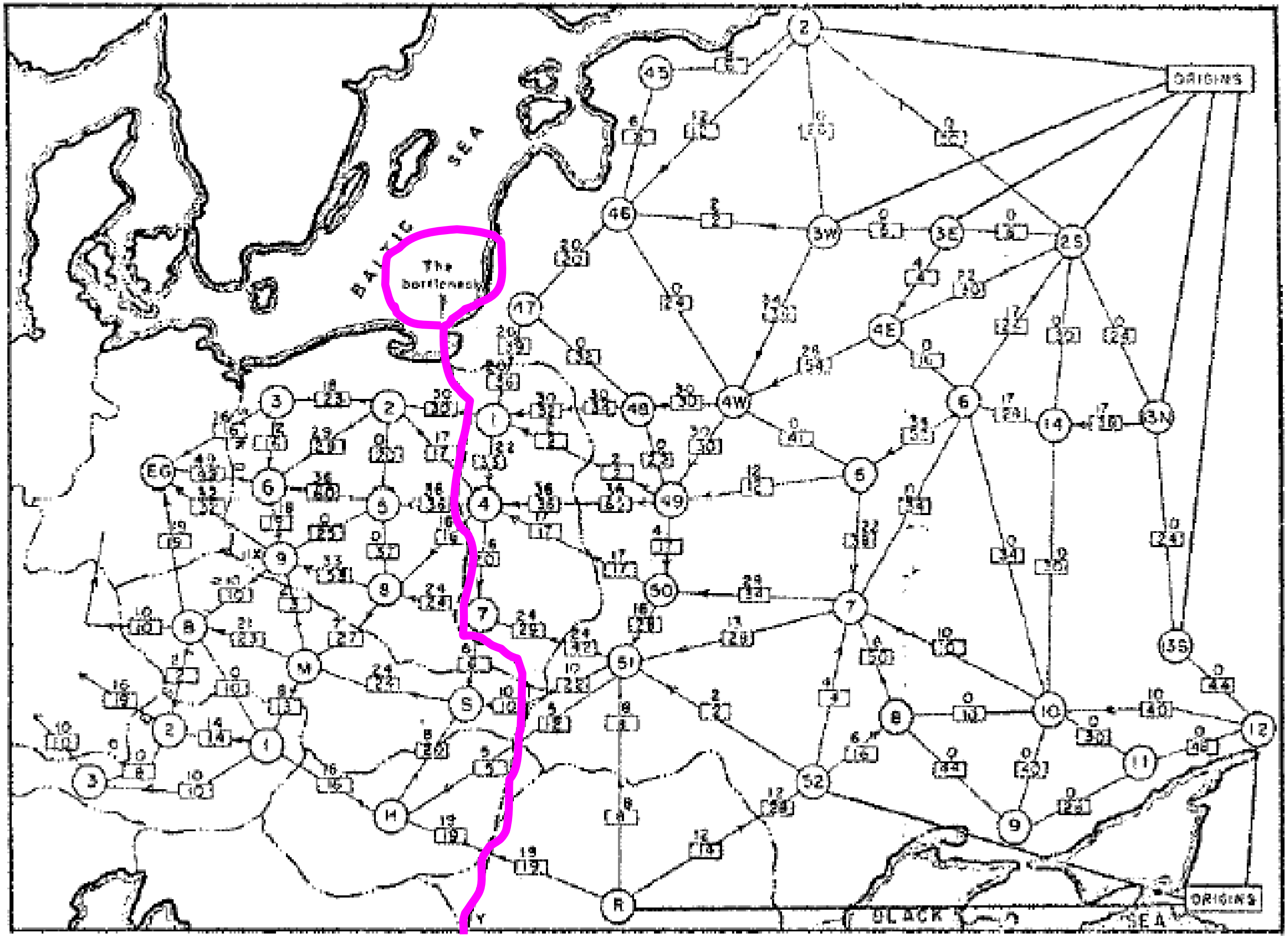


From Harris & Ross [1955]

declassified RAND report that originally inspired Ford and Fulkerson

Problem: find gas/oil pipelines or railways network bottleneck in Eastern Europe

Graph cut is an old standard problem with lots of applications outside vision



From Harris & Ross [1955]

declassified RAND report that originally inspired Ford and Fulkerson

Problem: find gas/oil pipelines or railways network bottleneck in Eastern Europe

Applications of max-flow (min cut) algorithms

- Matrix rounding
- Perfect matching
- Vertex cover
- Routing (airline scheduling)
- Baseball elimination
- Economics (circulation–demand problem)
- Computer vision

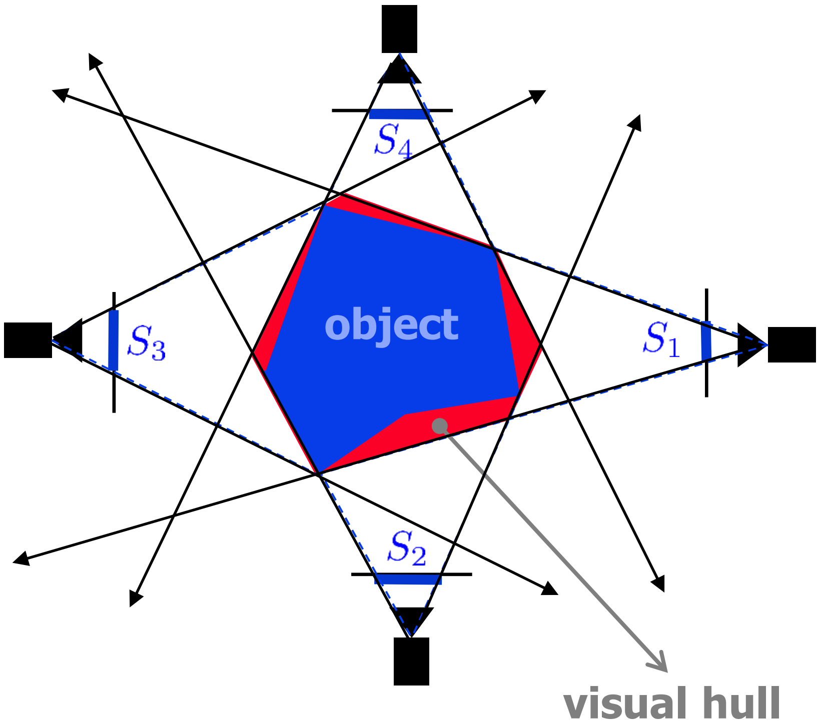
Multi-view volumetric photo-reconstruction

Calibrated images of Lambertian scene

3D model of scene

CVPR'05 slides from Vogiatzis, Torr, Cippola

first pass at dense volumetric multi-view reconstruction: use silhouettes => **visual hull**

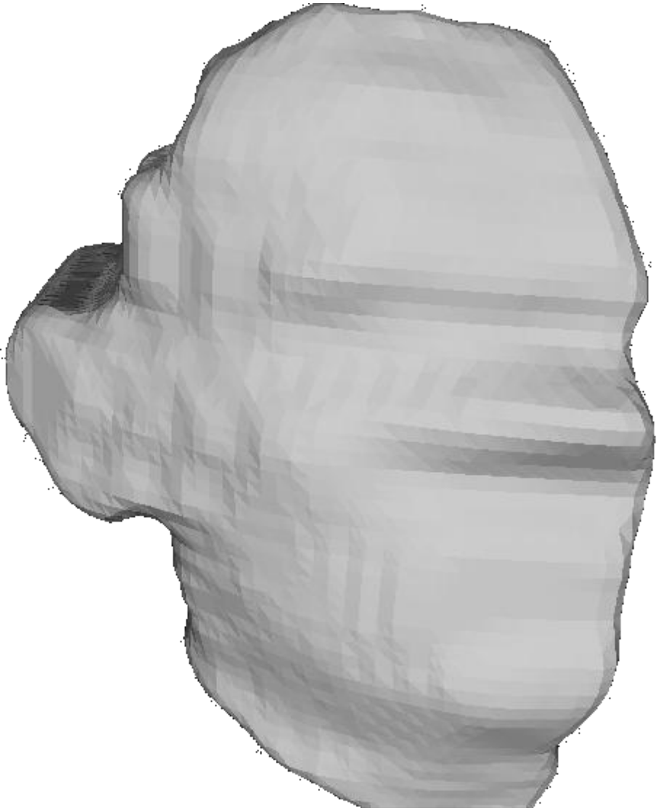


- Assume known cameras $P_i = K_i\,[R_i|T_i]$ (including position/orientation)
- Assume that each camera knows object's 2D silhouette S_i
  - binary image segmentation problem
  - ideas on how to solve it?
- Project each camera's silhouette into space to obtain a 3D *cone*.
- Intersection of the *cones* generated by silhouettes in each image gives the ***visual hull*** of the object

- visual hull is the smallest 3D shape consistent with all silhouettes.

- **object is a subset of its visual hull**

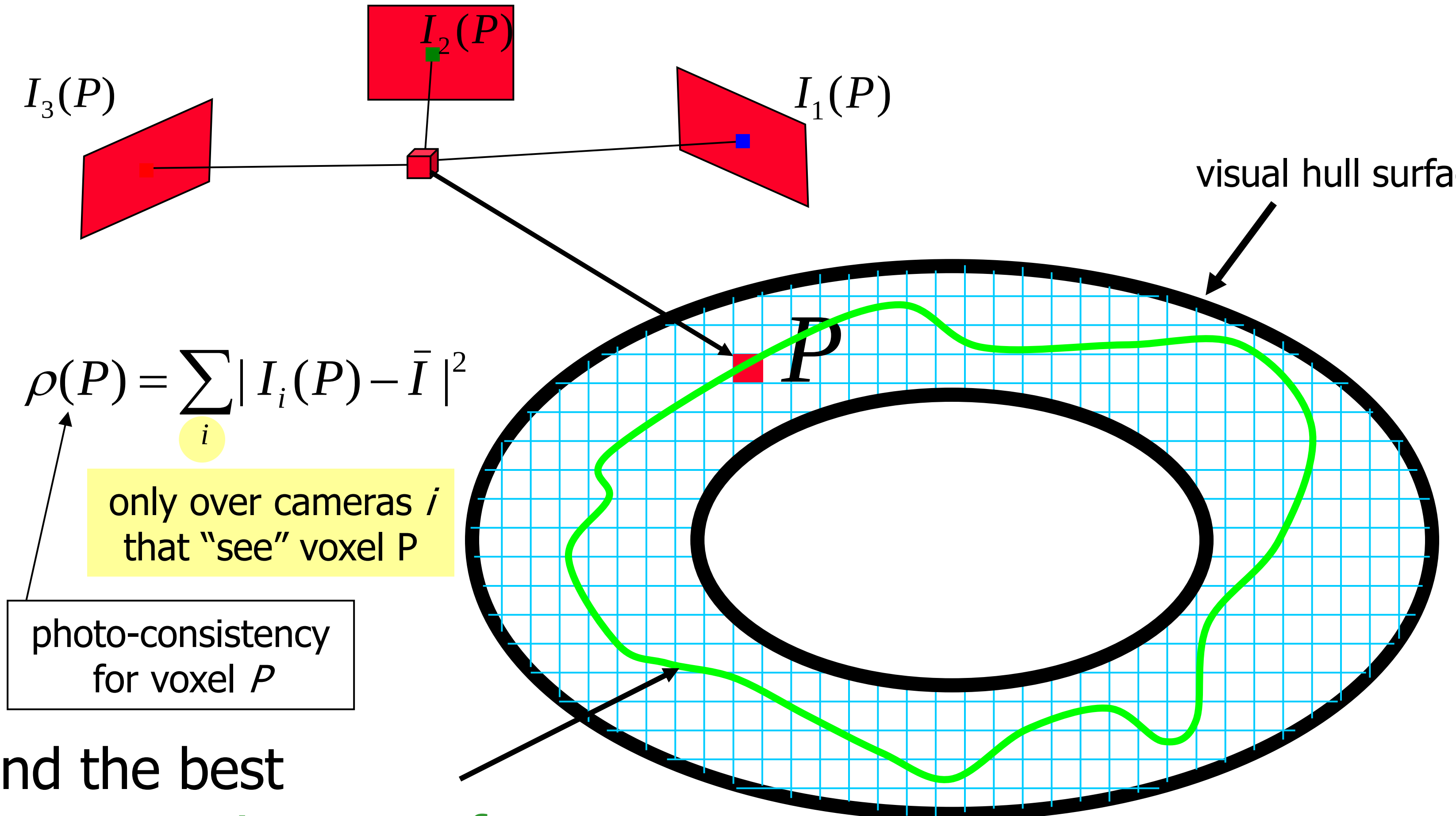
Visual hull of a human face



visual hull
(from silhouettes)

CVPR'05 slides from Vogiatzis, Torr, Cippola

Can refine visual hull using *photoconsistency*



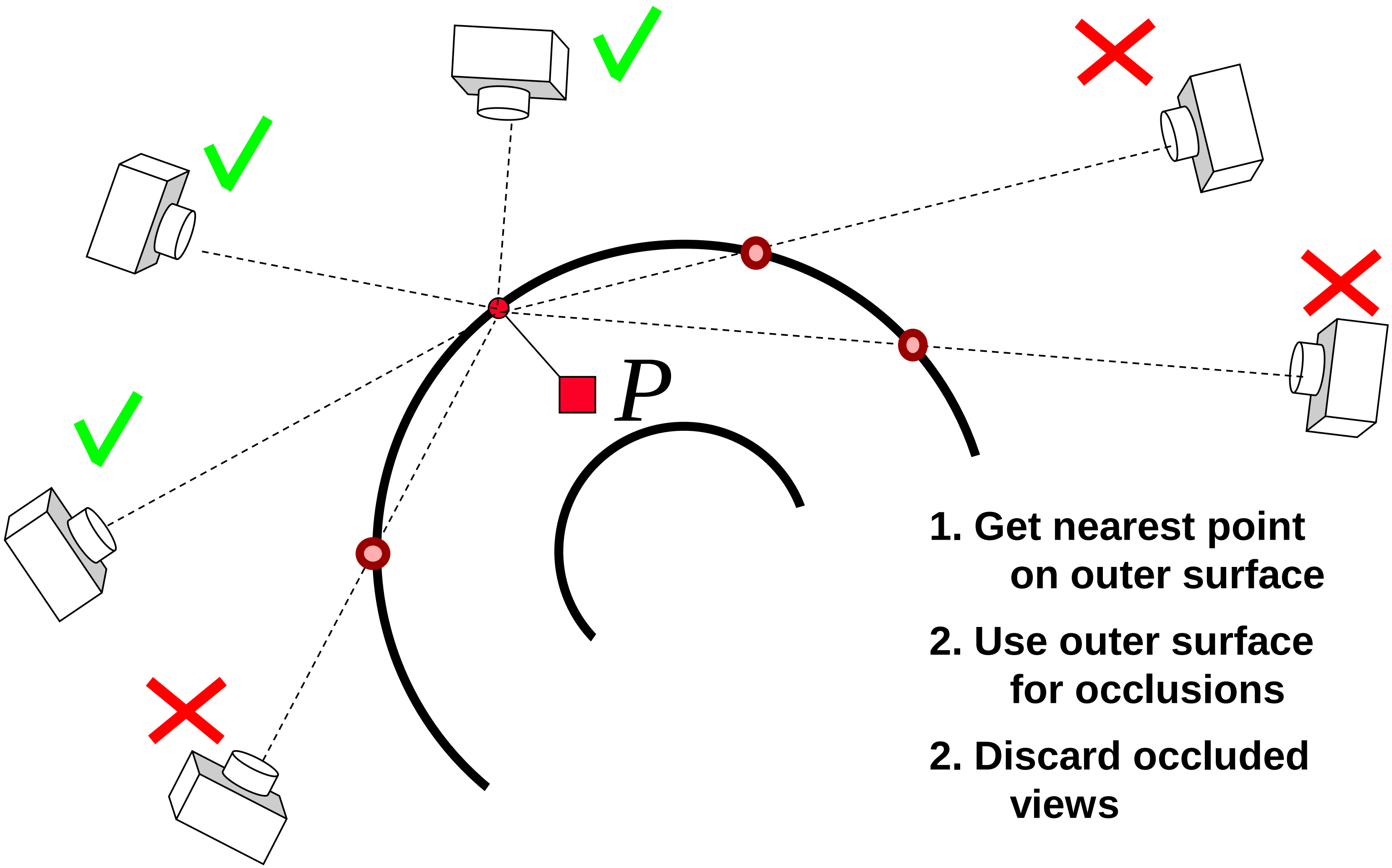
Find the best
photo-consistent surface
minimizing $\sum_{P \in S} \rho(P)$

CVPR'05 slides from Vogiatzis, Torr, Cippola

$$\rho(P) = \sum_{i} | I_i(P) - \bar{I} |^2$$

only over cameras *i* that "see" voxel P

Estimating *visibility*

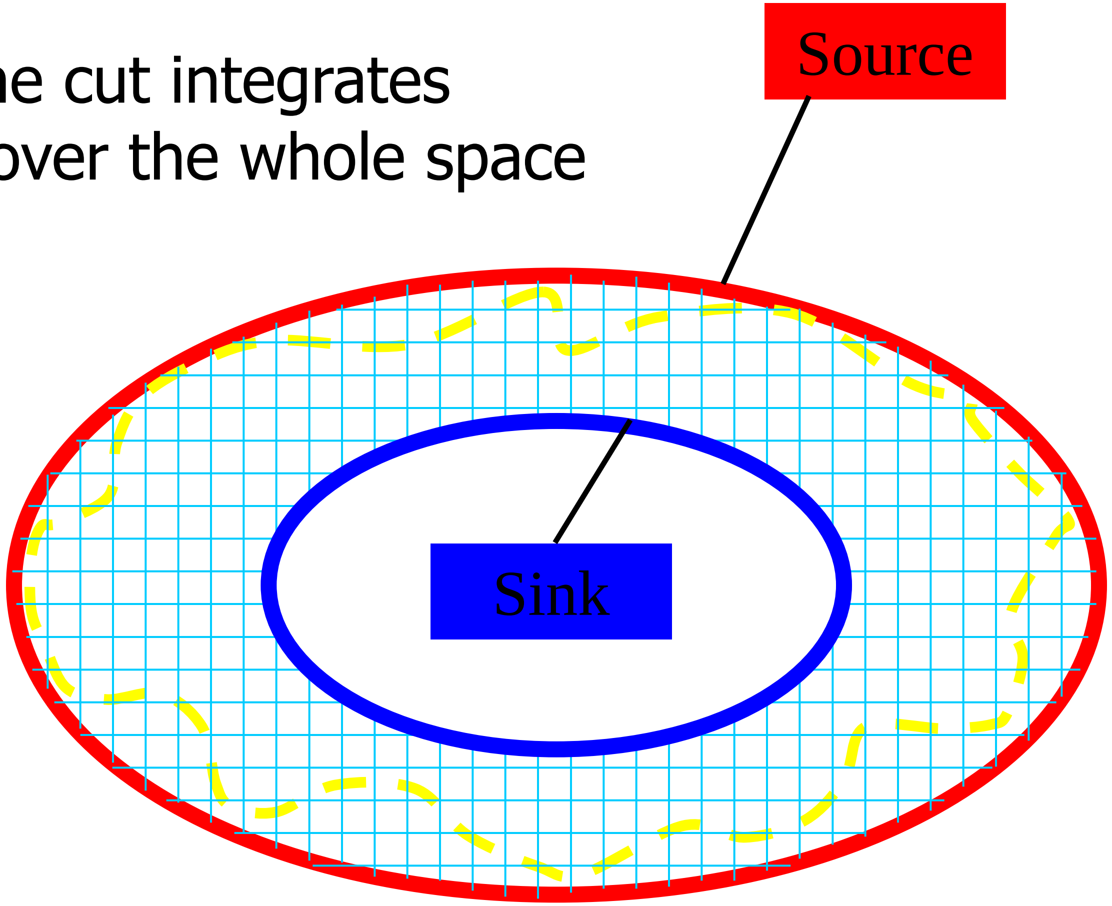


1. **Get nearest point on outer surface**
2. **Use outer surface for occlusions**
2. **Discard occluded views**

CVPR'05 slides from Vogiatzis, Torr, Cippola

Graph cuts applied to multi-view reconstruction

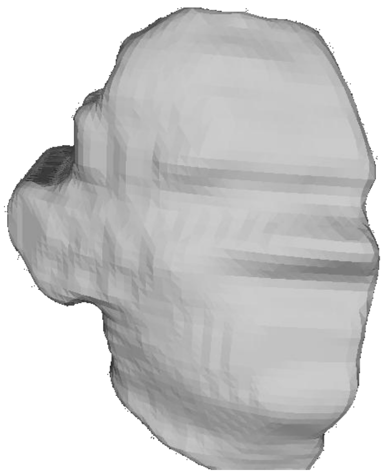
The cost of the cut integrates photoconsistency over the whole space



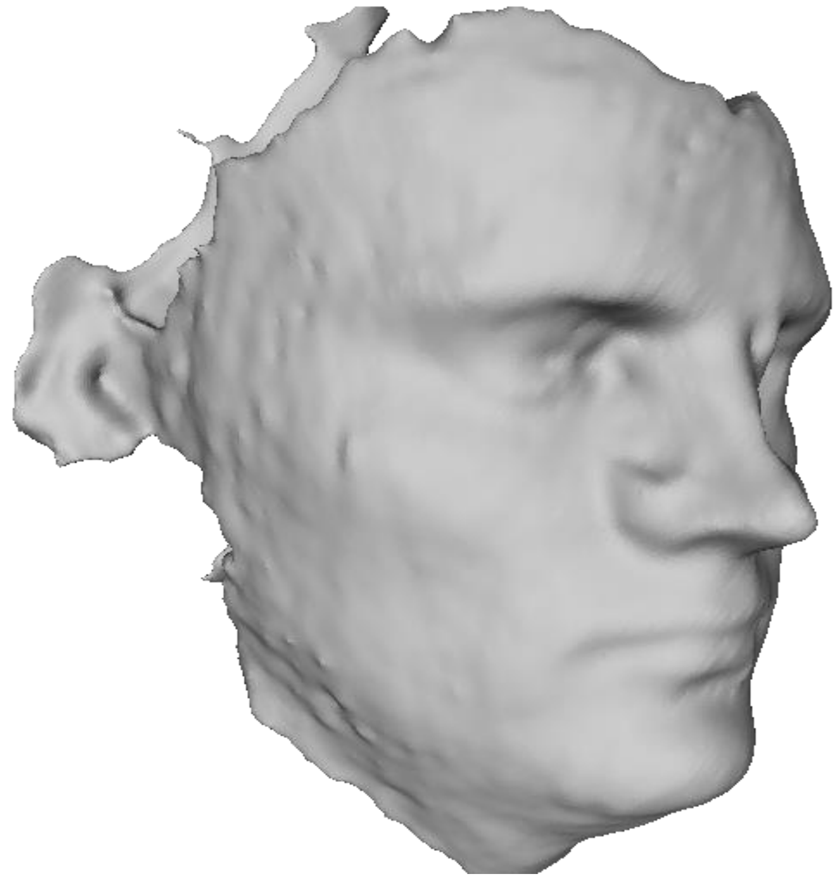
CVPR'05 slides from Vogiatzis, Torr, Cippola

Graph cuts applied to multi-view reconstruction

surface of good photoconsistency



visual hull
(silhouettes)

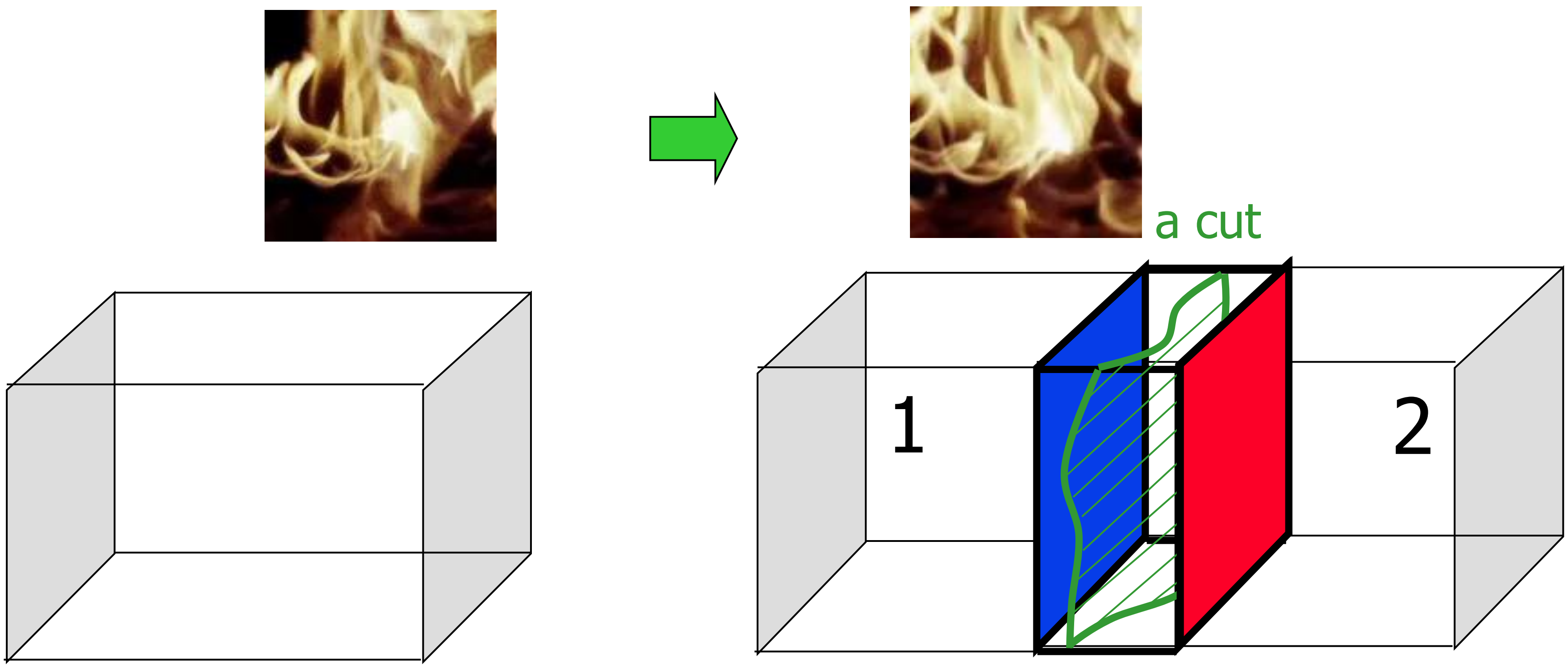


CVPR'05 slides from Vogiatzis, Torr, Cippola

Graph cuts for video textures

Graph-cuts video textures

(Kwatra, Schodl, Essa, Bobick 2003)



Short video clip

Long video clip

What is left to discuss in topic 9:

Combining appearance & boundary in segmentation loss function

A. known color/appearance + boundary regularization

B. color model fitting (K-means) + boundary regularization

C. kernel clustering objectives + boundary regularization

combining color & boundary objectives

"agreement" with given color likelihoods model defines appearance consistency

A: combining **known color model** and boundary regularization

or

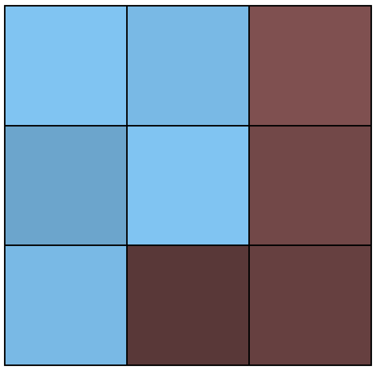
another example of
negative log-likelihood loss (NLL)
for observed (low-level) features, e.g. colors

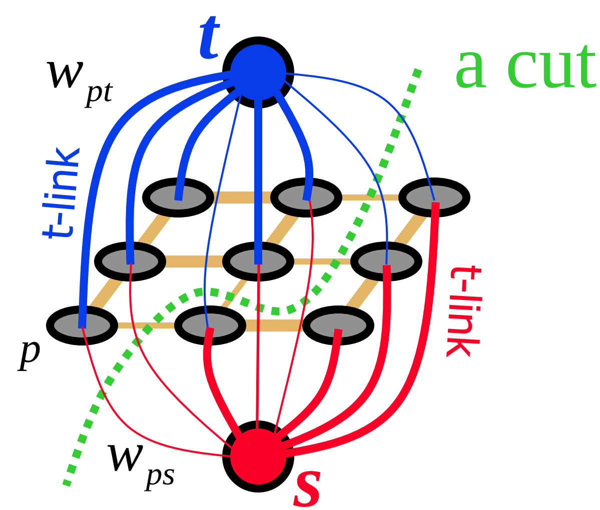
Adding regional properties

another segmentation example [B&J'01]

"regional" hard constrains (seeds) are replaced with "regional" **soft constraints**

optimal segmentation minimizes a **combination of SSE & boundary costs**





regional bias example 1

assume **known** "expected" intensities for object and background

e.g. $\theta_1 = 57$ and $\theta_0 = 213$

$$D_p(s) = ||I_p - \theta_1||^2 = w_{pt}$$

$$D_p(t) = ||I_p - \theta_0||^2 = w_{ps}$$

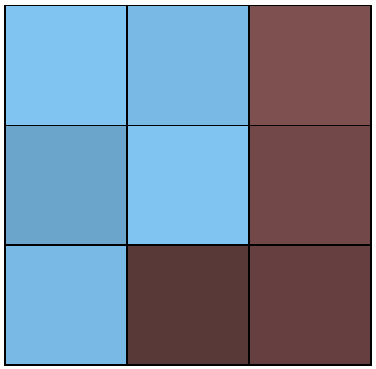
penalties/costs (e.g. ***squared errors***) for assigning labels ***s*** or ***t*** to pixel p

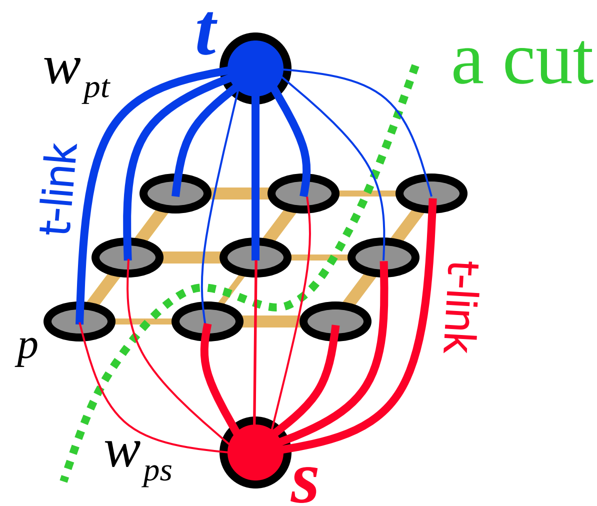
Adding regional properties

another segmentation example [B&J'01]

"regional" hard constrains (seeds) are replaced with "regional" **soft constraints**

optimal segmentation minimizes a **combination of log-likelihoods & boundary costs**





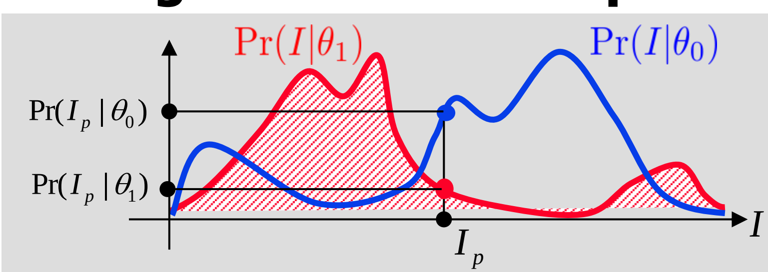
example 1 is a special case for $\|I_p - \theta_k\|^2$ <= **Gaussian pdf**

$$D_p(s) = -\ln \Pr(I_p|\theta_1) = w_{pt}$$

$$D_p(t) = -\ln \Pr(I_p|\theta_0) = w_{ps}$$

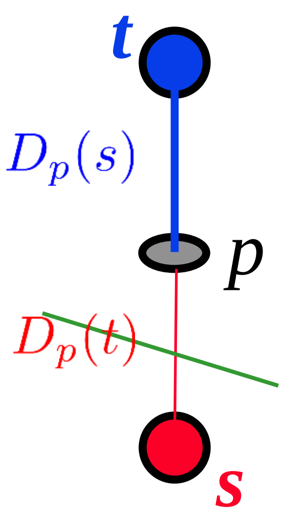
penalties/costs (***neg. log-likelihoods***) for assigning labels ***s*** or ***t*** to pixel *p*

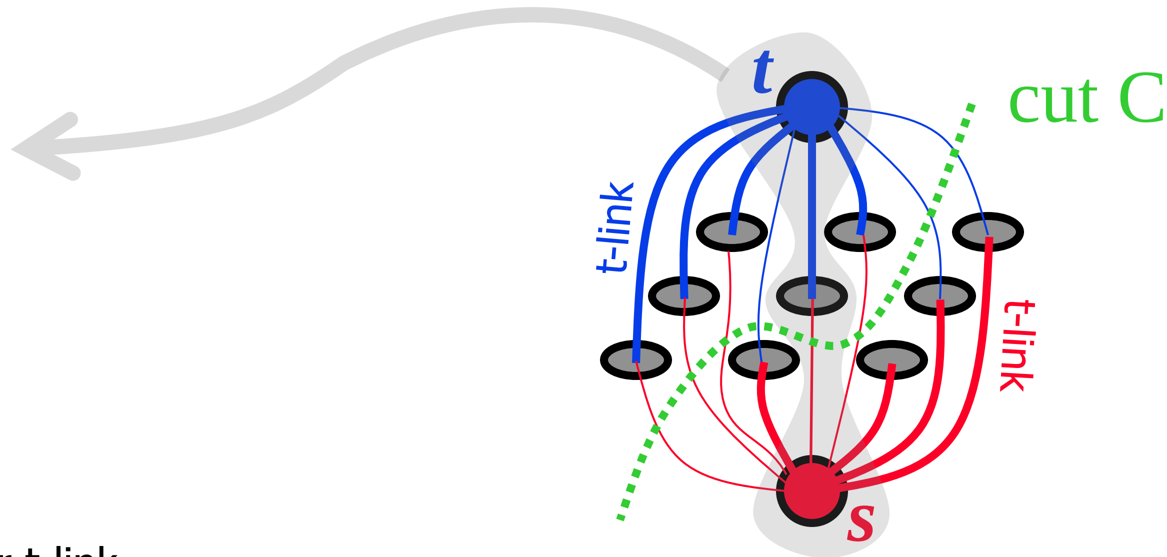
regional bias example 2



known probability distributions for object and background colors/intensities

What are *t-links* about?
(for now, assume no *n-links*)





A: sever the cheaper t-link at every pixel **independently**

trivial problem, no fancy algorithms needed

$$\min_{S_p \in \{s,t\}} D_p(S_p)$$

altogether, we optimize the sum of unary (pixelwise) terms

$$\min_S \sum_p D_p(S_p)$$

Q: What is the minimum cut on a graph if there are only *t-links* (no *n-links*) ?

What are *t-links* about?
(for now, assume no *n-links*)

WARNING: below we use both integer-valued indicators or one-hot distributions, as convenient. The exact interpretation should be clear from context.

with discrete (hard) segmentation (as graph cuts)
we can use either class indicator variables (integers)

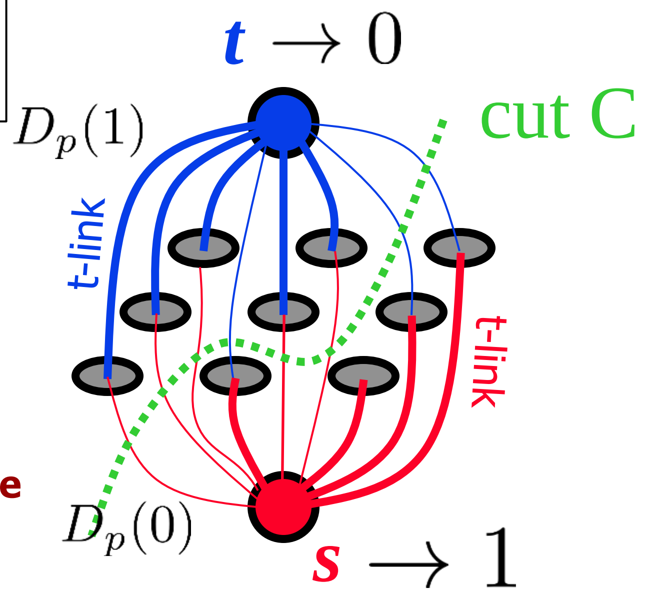
$S_p \in \{0, 1\}$ or $S_p \in \{1, \dots, K\}$

OR (equivalently) one-hot distributions, e.g. (1,0) and (0,1)

$S_p = (S_p^1, S_p^0) \in \Delta_{\mathbf{v}}^2$ or $S_p = (S_p^1, \dots, S_p^K) \in \Delta_{\mathbf{v}}^K$

For continuous/relaxed segmentation, it is common to use (soft) categorical distributions (as was in fuzzy K-means)

$$S_p = (S_p^1, \dots, S_p^K) \in \Delta^K$$



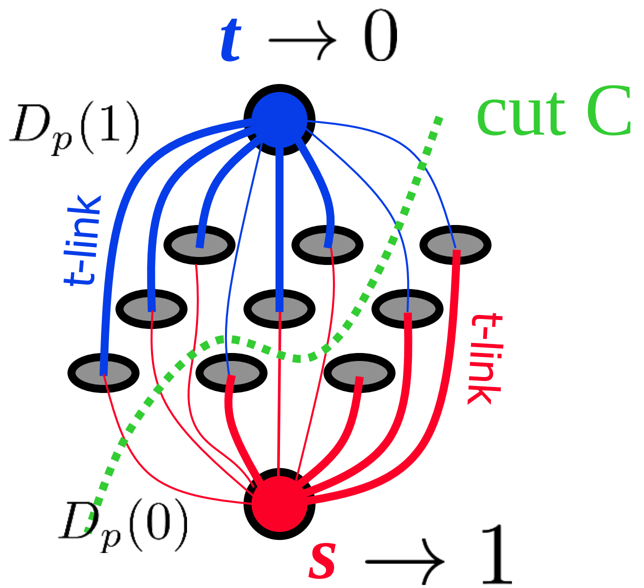
cost of severed t-links

$$||C(S)|| \equiv L(S) = \sum_p D_p(S_p)$$

cost of cut C — loss for segmentation S

What are *t-links* about?
(for now, assume no *n-links*)

t-links describe
individual pixel preferences or
likelihoods of labels (*s* and *t*)



cost of severed t-links

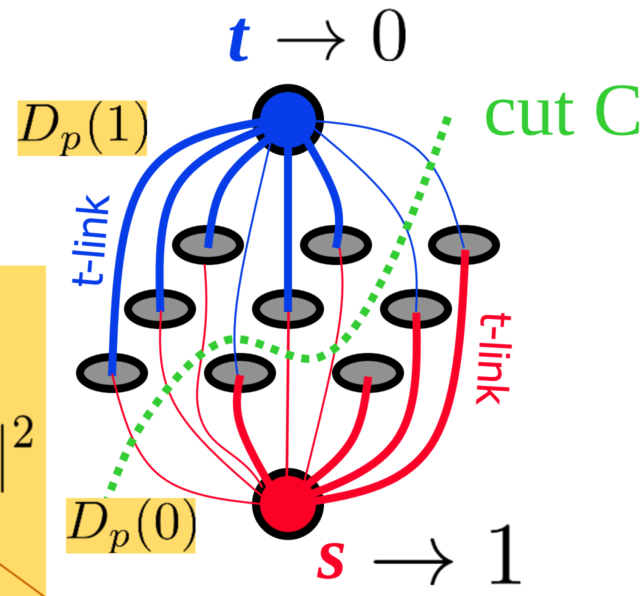
$$\|C(S)\| \equiv L(S) = \sum_p D_p(S_p)$$

cost of cut C

loss for segmentation S

What are *t-links* about?
(for now, assume no *n-links*)

t-links describe <u>individual pixel preferences</u> or likelihoods of labels (*s* and *t*)



remember example 1
with **squared errors**

$$D_p(S_p) = \|I_p - \theta_{S_p}\|^2 \equiv \sum_{k=0}^{1} S_p^k \, \|I_p - \theta_k\|^2$$

lower cost label $S_p \in \{0,1\}$ selects closer "center" θ_k

$S_p \in \Delta^2_{\mathbf{v}}$

NOTE: the second formulation of D allows relaxed segmentation $S_p \in \Delta^2$ ($D(S)$ is **linear** w.r.t. S as in K-means)

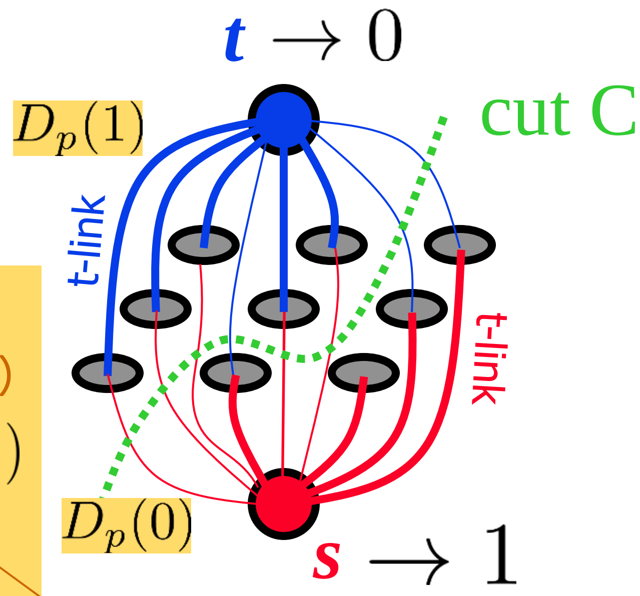
$$\|C(S)\| \equiv L(S) = \sum_p D_p(S_p)$$

cost of cut C — loss for segmentation S — cost of severed t-links

minimizing squared errors

What are *t-links* about?
(for now, assume no *n-links*)

t-links describe
individual pixel preferences or
likelihoods of labels (***s*** and ***t***)



remember example 2
with **neg. log-likelihoods**

"NLL"
example (slide 52)

$$D_p(S_p) = -\ln \Pr(I_p|\theta_{S_p}) \;\equiv\; -\sum_{k=0}^{1} S_p^k \ln \Pr(I_p|\theta_k)$$

lower cost label $S_p \in \{0,1\}$
selects higher likelihood model θ_k

$S_p \in \Delta^2_{\mathrm{v}}$

NOTE: the second formulation of D allows relaxed segmentation $S_p \in \Delta^2$ (**linear** for S as in probabilistic K-means)

cost of severed t-links

$$||C(S)|| \equiv L(S) = \sum_p D_p(S_p)$$

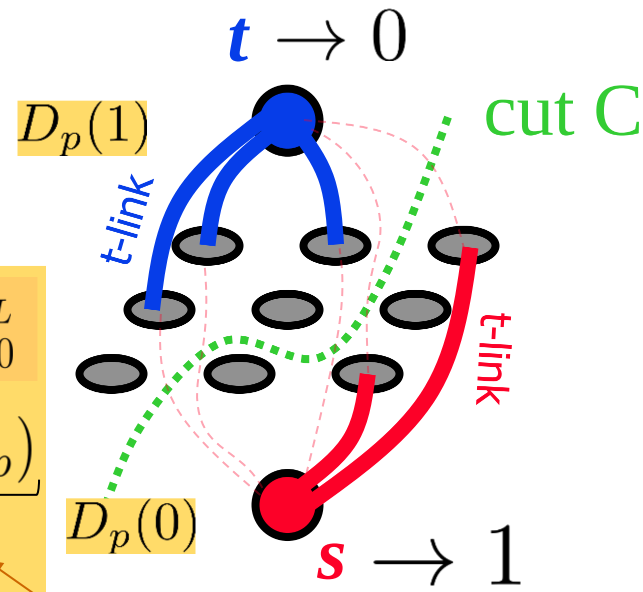
cost of cut C

loss for segmentation S

maximizing log-likelihoods (of features/colors)

What are *t-links* about?
(for now, assume no *n-links*)

t-links describe <u>individual pixel preferences</u> or likelihoods of labels (*s* and *t*)



remember earlier example with **hard constraints** / seed labels y_p

NOTE: for $p \notin \Omega_L$ $D_p(0) = D_p(1) = 0$

$$D_p(S_p) = \begin{cases} 0 & \text{if } S_p = y_p \\ \infty & \text{if } S_p \neq y_p \end{cases} \equiv -\ln \underbrace{\Pr(S_p = y_p)}_{S_p^{y_p}}$$

for $p \in \Omega_L$ (seeds)

lower cost label $S_p \in \{0,1\}$ selects feasible solution

"NLL" example (slide 16)

$S_p \in \Delta^2_{\mathbf{v}}$

NOTE: the second formulation of D allows relaxed segmentation $S_p \in \Delta^2$ (**non-inear** function $-log(S_p^{y_p})$ w.r.t S)

cost of severed t-links

$$\|C(S)\| \equiv L(S) = \sum_p D_p(S_p)$$

cost of cut C

loss for segmentation S

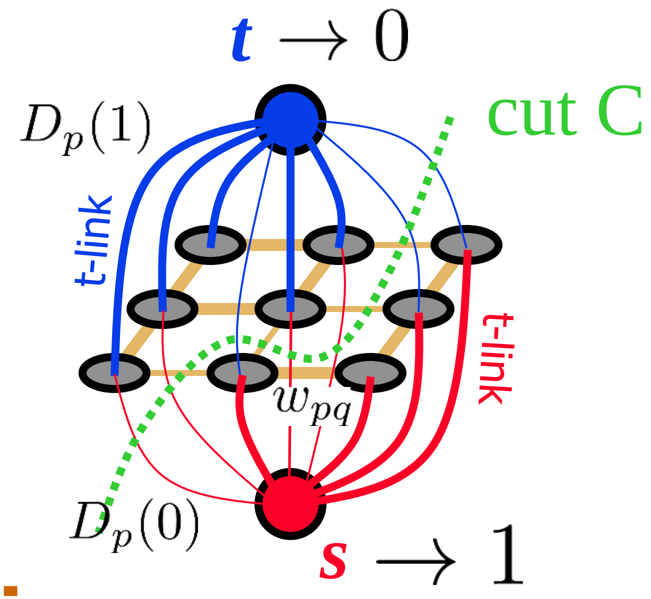
maximizing log-probabilities (of correct labeling)

Summary:

(putting *t-links* and *n-links* back together again)

t-links describe
individual pixel preferences or
likelihoods of labels (*s* and *t*)

n-links describe
pairwise pixel correlations or
structural regularization, which
can be interpreted as (MRF) **prior**



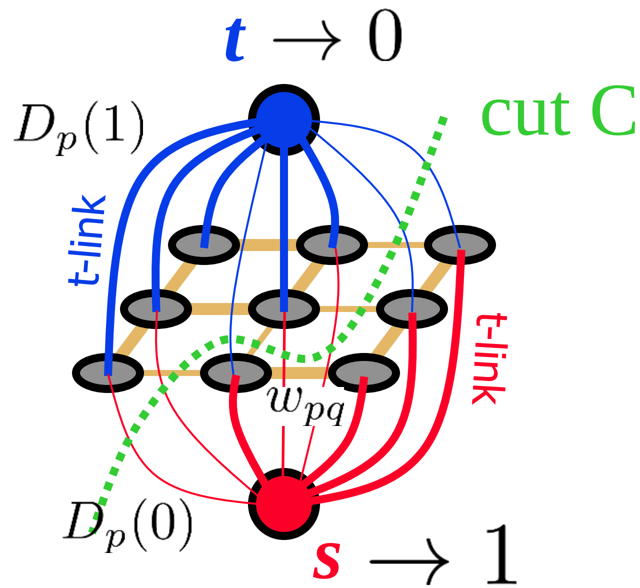
cost of severed t-links — cost of severed n-links

$$\|C(S)\| \equiv L(S) = \sum_p D_p(S_p) + \sum_{pq \in N} w_{pq}\,[S_p \neq S_q]$$

cost of cut C — loss for segmentation S

no longer a trivial optimization problem

Summary:
(putting *t-links* and *n-links* back together again)



Comment on (so-called)
regularization constant

$$w_{pq} = \lambda \exp\left\{-\frac{\|I_p - I_q\|^2}{2\sigma^2}\right\}$$

Important **hyper-parameter** of the (joint) energy since it determines relative weight of the two terms:

regional (unary) vs. **boundary (pairwise)**

cost of severed t-links — cost of severed n-links

$$\|C(S)\| \equiv L(S) = \sum_p D_p(S_p) + \sum_{pq \in N} w_{pq}\,[S_p \neq S_q]$$

cost of cut C — loss for segmentation S

no longer a trivial optimization problem

optional slide

Extensions for segmentation energy/loss optimization:

submodular set functions

(discrete/combinatorial optimization)

$$E(S) = \sum_A E_A(S_A) \quad \text{for} \quad S_A = \{S_p \mid p \in A\}$$

factors (unary, pairwise, high-order)

MRF/CRF

MAP estimation loss

log-likelihoods + log prior

geometric surface functionals

(continuous optimization, PDEs)

minimum surfaces (area, curvature, shape)

e.g. $\int_S d(v)dv + \int_{\partial S} w(s)ds$

multi-label problems

(e.g. multi-way cuts, relaxation)

$S_p \in \{1, 2, \ldots, K\}$ class indices

or

$S_p \in \Delta^K$ categorical distributions

$$\|C(S)\| \equiv L(S) = \sum_p D_p(S_p) + \sum_{pq \in N} w_{pq} \, [S_p \neq S_q]$$

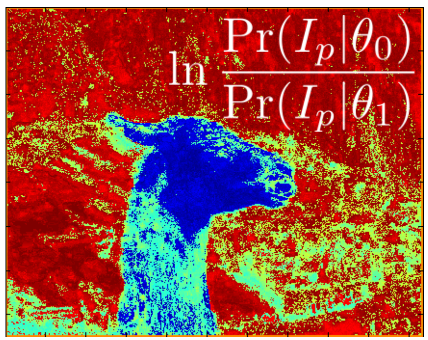
cost of cut C

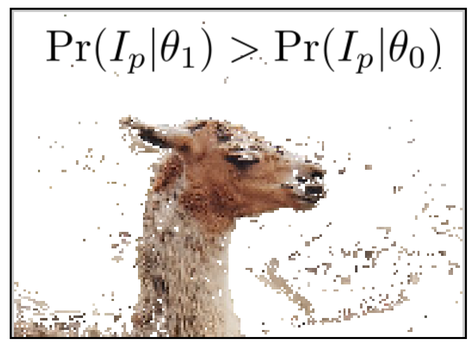
loss for segmentation S

no longer a trivial optimization problem

Graph cut vs Thresholding

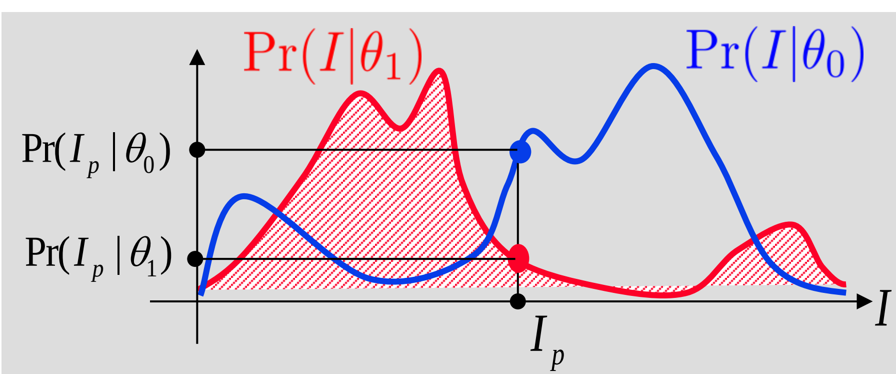
$$E(S) = \underbrace{\sum_p D_p(S_p) + \sum_{pq \in N} w_{pq} [S_p \neq S_q]}$$





thresholding
(naive Bayesian classification, *iid* pixels)

optimal graph cut
(correlated pixels, *MRF*/*CRF* inference)

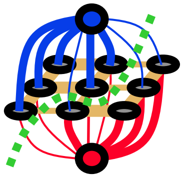


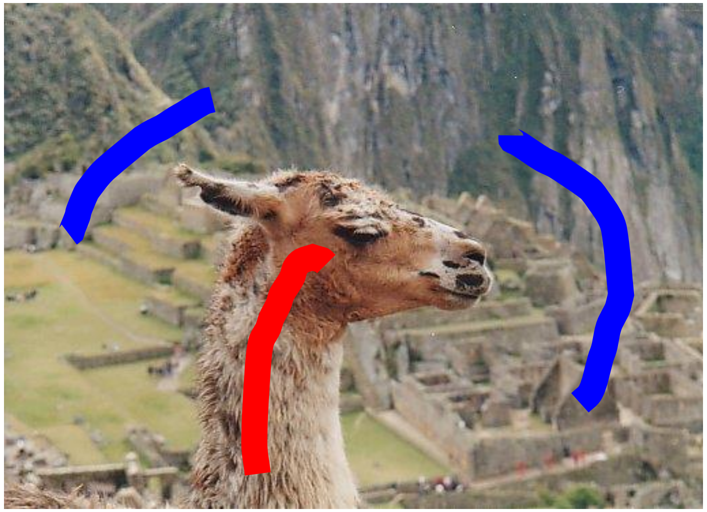
$-\ln \Pr(I_p|\theta_1)$ $\quad$ $-\ln \Pr(I_p|\theta_0)$

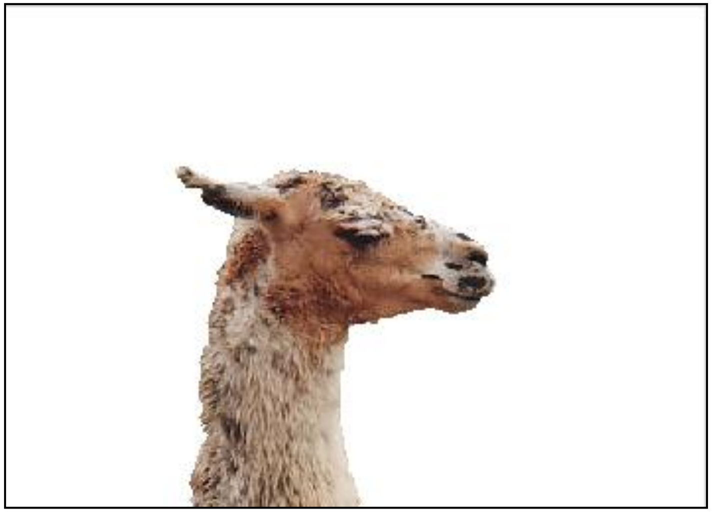
$$S_p = \begin{cases} 1 & \text{if } D_p(1) < D_p(0) \\ 0 & \text{O.W.} \end{cases}$$

result of optimizing unary potentials D_p (only)

Given Color Models



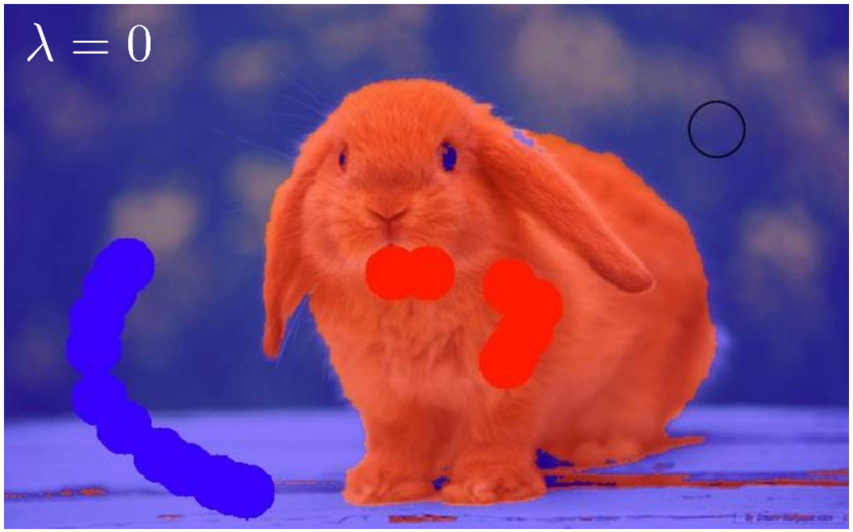




Appearance color distributions θ_0 and θ_1 can be estimated from user seeds

e.g. histograms or GMM distributions (as in HW4)
estimated from RGB colors of pixels in the seeded regions

Comparison:



color likelihoods only (thresholding)

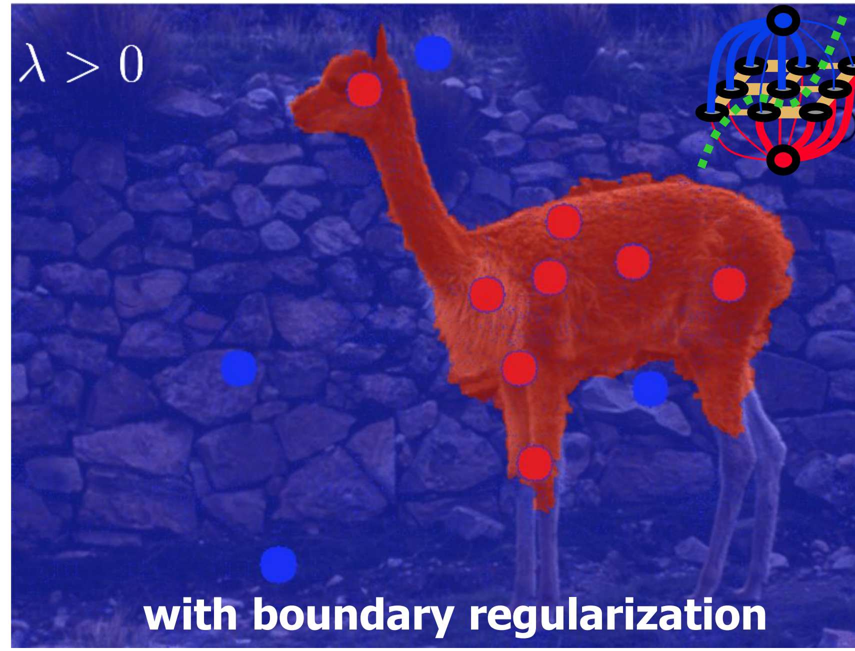
with boundary regularization

Even in examples (as here) where object colors are discriminative, boundary regularization is useful

In this image, adding color models helps a lot. Our earlier result (**slide 12**) with n-links only required more seeds. It also required n-link weighting function w significantly more sensitive to intensity contrast (significantly smaller σ).

Comparison (less trivial example):

Low-level features (like RGB colors) are discriminative only in simple cases

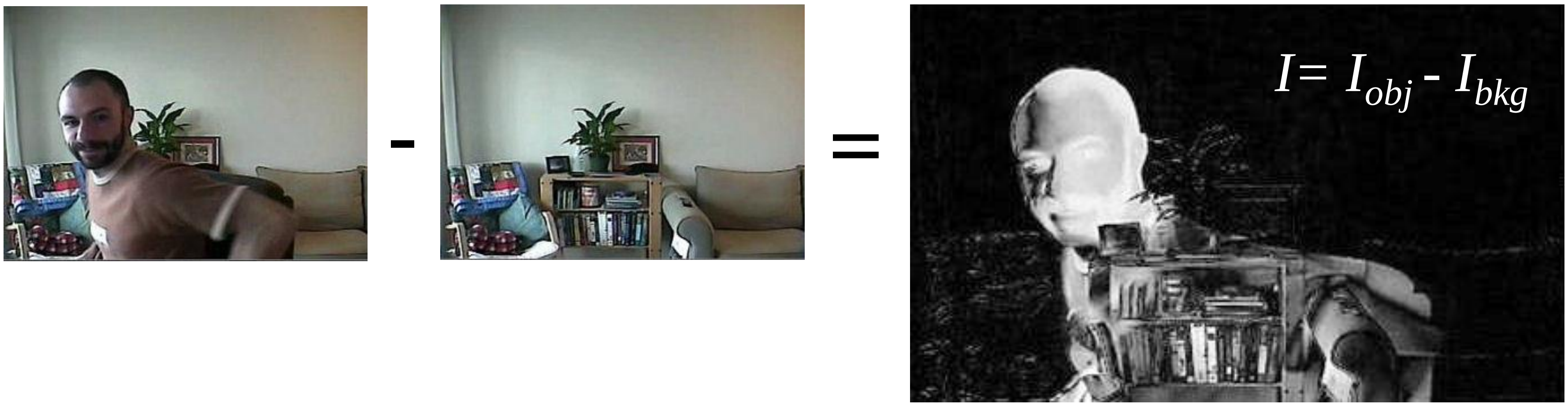


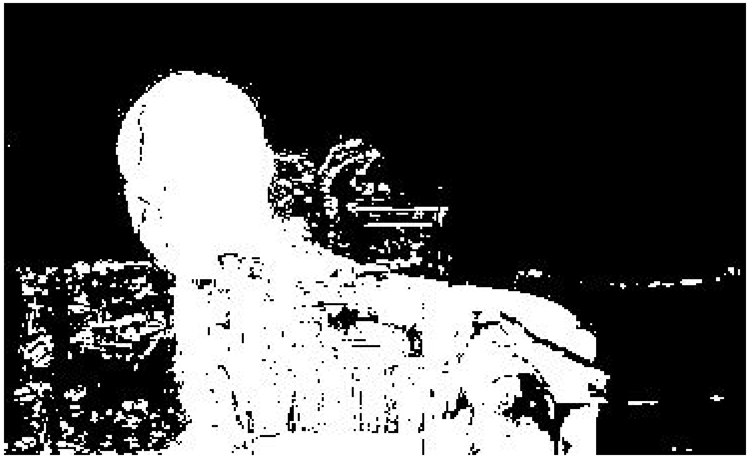
Includes higher-order features (shape boundary, contrast edges)

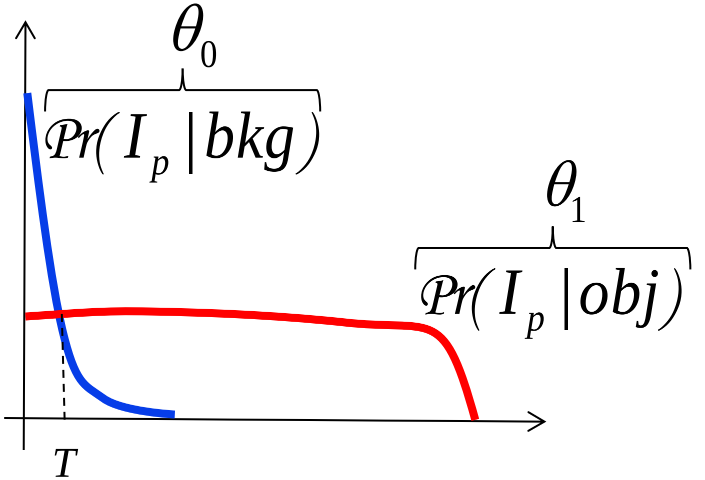
In the context of CNN segmentation (topics 11 & 12) we will discuss methods to automatically learn discriminative high-level (semantic or deep) features from **<u>many</u> fully- or weakly-supervised examples**

Adding regional properties

another segmentation example





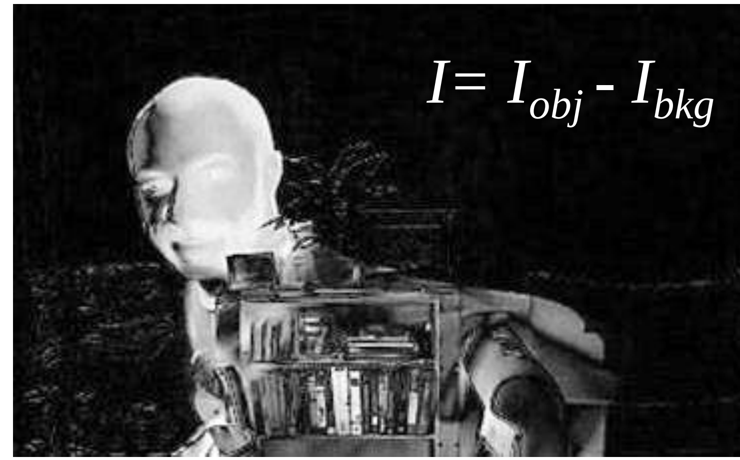


Threshold intensities $S = \{ p : I_p > T \}$

Adding regional properties
(example: regularized background subtraction)

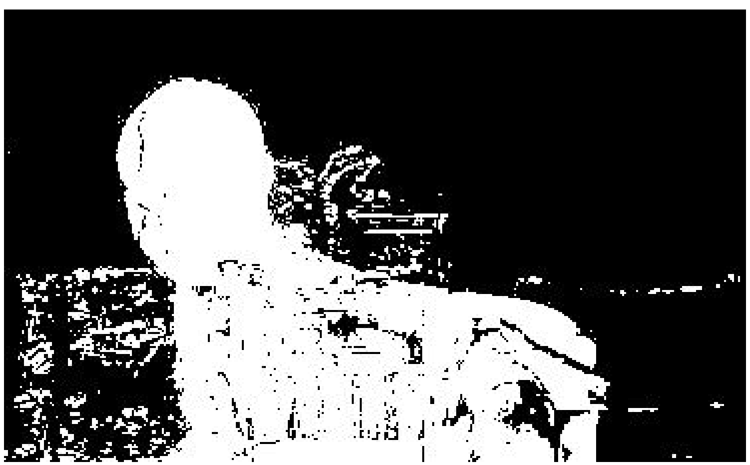
-

=



thresholding

graph cuts



Threshold intensities $S = \{ p : I_p > T \}$

optimal cut

What is left to discuss in topic 9:

Combining appearance & boundary in segmentation loss function

A. known color/appearance + boundary regularization

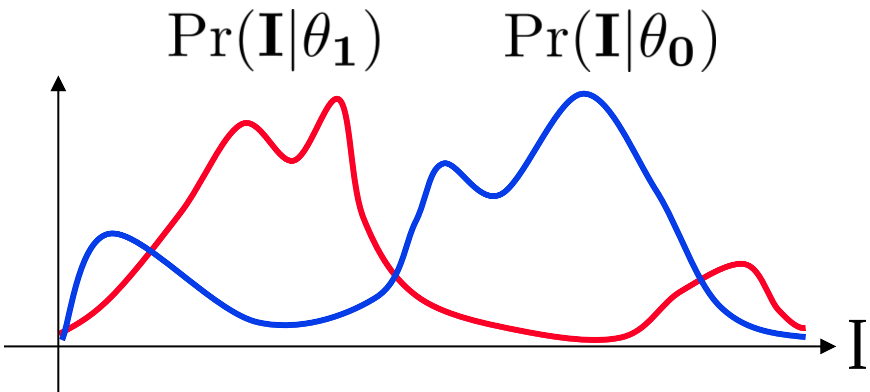
B. color model fitting (K-means) + boundary regularization

C. kernel clustering objectives + boundary regularization

we will do only a quick overview; the detailed slides are left for optional reading

(preview)

What if models $\mathbf{Pr(I \mid \theta_i)}$ are <u>not known</u>?



(preview)

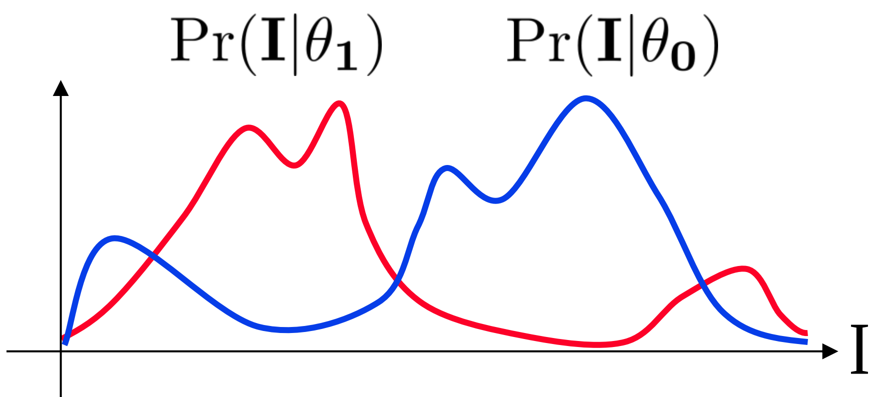
What if models $\mathbf{Pr}(\,\mathbf{I}\,|\,\theta_i)$ are not known?

$$S_p \in \Delta_{\mathbf{v}}^2$$

$$E(S) = \sum_p D_p(S_p) + \sum_{pq\in\mathcal{N}} w_{pq}\cdot[S_p \neq S_q]$$

$$= -\sum_p \sum_k S_p^k \ln \Pr(I_p|\theta_k) + \sum_{pq\in\mathcal{N}} w_{pq}\cdot[S_p \neq S_q]$$

see NLL loss on slides 52,57



(preview)

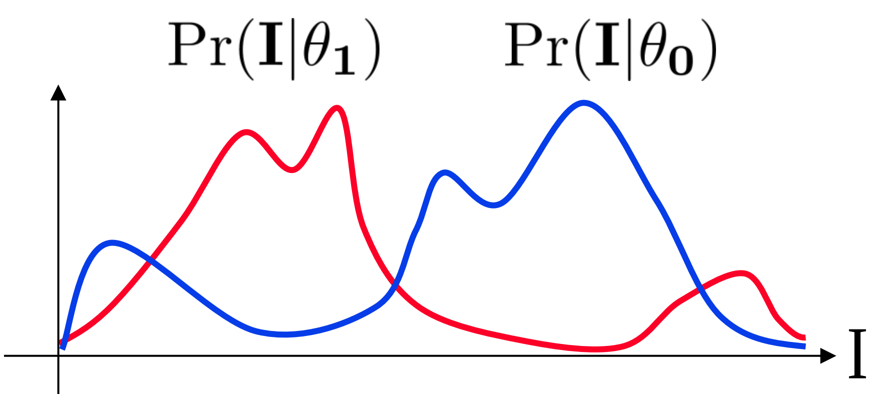
What if models $\mathbf{Pr}(\ \mathbf{I} \mid \boldsymbol{\theta_i})$ are <u>not known</u>?

$$S_p \in \Delta^2_{\mathbf{v}}$$

$$E(S) = \sum_p D_p(S_p) + \sum_{pq \in \mathcal{N}} w_{pq} \cdot [S_p \neq S_q]$$

$$= -\sum_k \sum_p S_p^k \ln \Pr(I_p|\theta_k) \quad + \sum_{pq \in \mathcal{N}} w_{pq} \cdot [S_p \neq S_q]$$

see NLL loss on slides 52,57



(preview)

What if models $\mathbf{Pr}(\mathbf{I} \mid \theta_i)$ are <u>not known</u>?

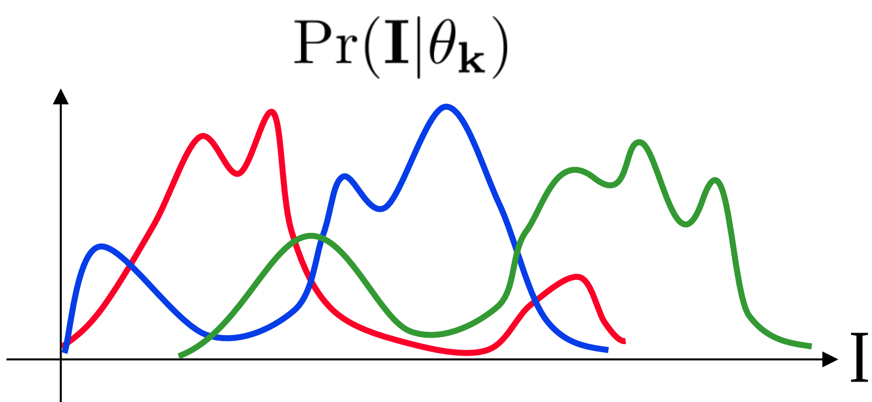
$$S_p \in \Delta_{\mathbf{v}}^K$$

$$E(S) = \sum_p D_p(S_p) + \sum_{pq \in \mathcal{N}} w_{pq} \cdot [S_p \neq S_q]$$

$$= -\sum_k \sum_p S_p^k \ln \Pr(I_p|\theta_k) + \sum_{pq \in \mathcal{N}} w_{pq} \cdot [S_p \neq S_q]$$

see NLL loss on slides 52,57

Let's switch to K-class segmentation, but optimization w.r.t. segmentation S is more difficult, e.g. no polynomial solver for $K > 2$ even for fixed models θ_k



(preview)

What if models $\mathbf{Pr}(\,\mathbf{I}\,|\,\boldsymbol{\theta_i}\,)$ are <u>not known</u>?

$S_p \in \Delta_{\mathbf{v}}^K$

approach

A:

$$E(S,\theta) = \sum_p D_p(S_p) + \sum_{pq\in\mathcal{N}} w_{pq}\cdot[S_p \neq S_q]$$

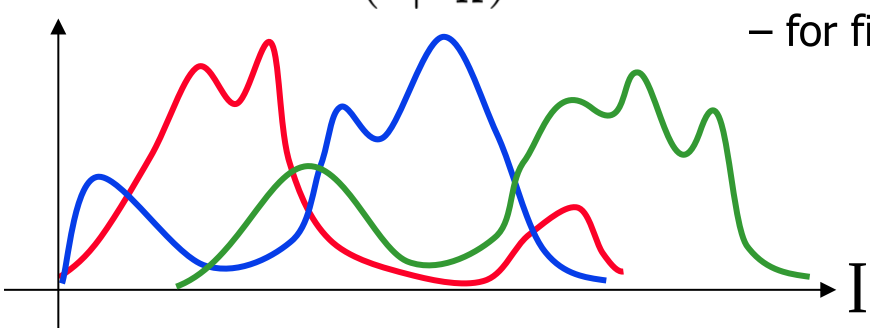
$$= -\sum_k \sum_p S_p^k \ln \Pr(I_p|\theta_k) + \sum_{pq\in\mathcal{N}} w_{pq}\cdot[S_p \neq S_q]$$

see K-means loss, Topic 9A, slide 55

probabilistic K-means over color features I_p
if unknown K distributions Θ_k are treated
as <u>additional optimization variables</u>

segmentation boundary regularization

$\Pr(\mathbf{I}|\theta_\mathbf{k})$



Approximate Optimization Idea (greedy iterations)
– for fixed Θ_i (back to "known" models) optimize over $\{S_p\}$
– for fixed $\{S_p\}$ optimize over model parameters Θ_i

Segmentation combining color model fitting and boundary regularization

(preview)

What if models $\mathbf{Pr}(\mathbf{I} \,|\, \theta_i)$ are not known?

approach

B:

$$S_p \in \Delta^K_{\mathbf{v}}$$

k-th segment indicator vector

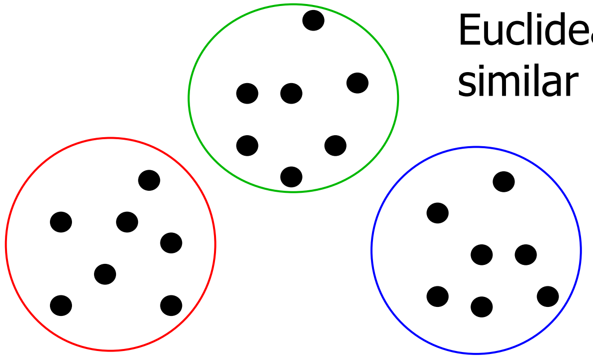
$$S^k = (S^k_p \,|\, p \in \Omega) = (S^k_1, \ldots, S^k_{|\Omega|})$$

$$E(S) \quad = \quad -\sum_k \frac{{S^k}' A S^k}{|S^k|} \quad + \quad \sum_{pq \in \mathcal{N}} w_{pq} \cdot [S_p \neq S_q]$$

non-parametric clustering (e.g. kernel K-means) using any pixel features f_p or affinities $[A_{pq}]$

segmentation boundary regularization

Approximate Optimization Idea: use spectral decomposition of A to convert the first term to basic K-means over low-dimensional Euclidean embedding $\{\tilde{\phi}_p\}$ such that $\langle \tilde{\phi}_p, \tilde{\phi}_q \rangle = \tilde{A}_{pq}$. Then, iterate similar two optimization steps (e.g. graph cuts and mean estimation)



Segmentation combining kernel clustering of image features and boundary regularization

(preview)

Examples: clustering + spatial regularization

- Unsupervised segmentation [Zhu&Yuille, 1996]

$$E(S, \theta_1, \cdots, \theta_K) \;=\; -\sum_{k=1}^{K} \sum_{p} S_p^k \ln \Pr(I_p|\theta_k) \;+\; \sum_{pq \in \mathcal{N}} w_{pq} \cdot [S_p \neq S_q] \;+\; |labels|$$

initialize models θ_0 , θ_1 , θ_2 , ...
from randomly sampled boxes

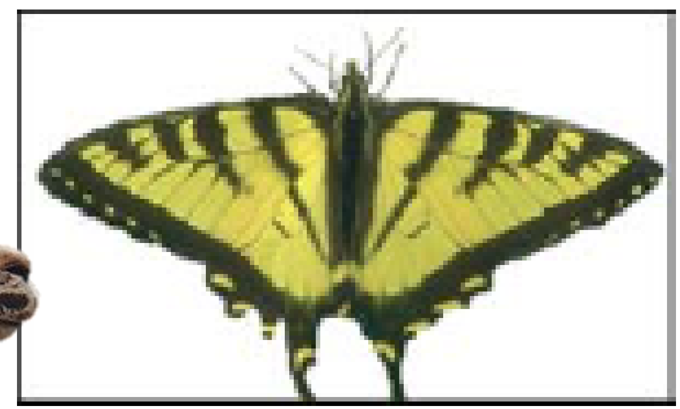
iterate segmentation
and model re-estimation
until convergence

(preview)

Examples: clustering + spatial regularization

- Box-supervised segmentation [*GrabCut*, Rother et al SIGGRAPH'04]

$$E(S,\theta_1,\theta_0) \;=\; -\sum_{k=0}^{1}\sum_{p} S_p^k \ln \Pr(I_p|\theta_k) \;+\; \sum_{pq\in\mathcal{N}} w_{pq}\cdot[S_p \neq S_q]$$



start from models θ_0 , θ_1
based on colors
inside and outside some given

iterate graph cut segmentation
and model re-estimation
until convergence

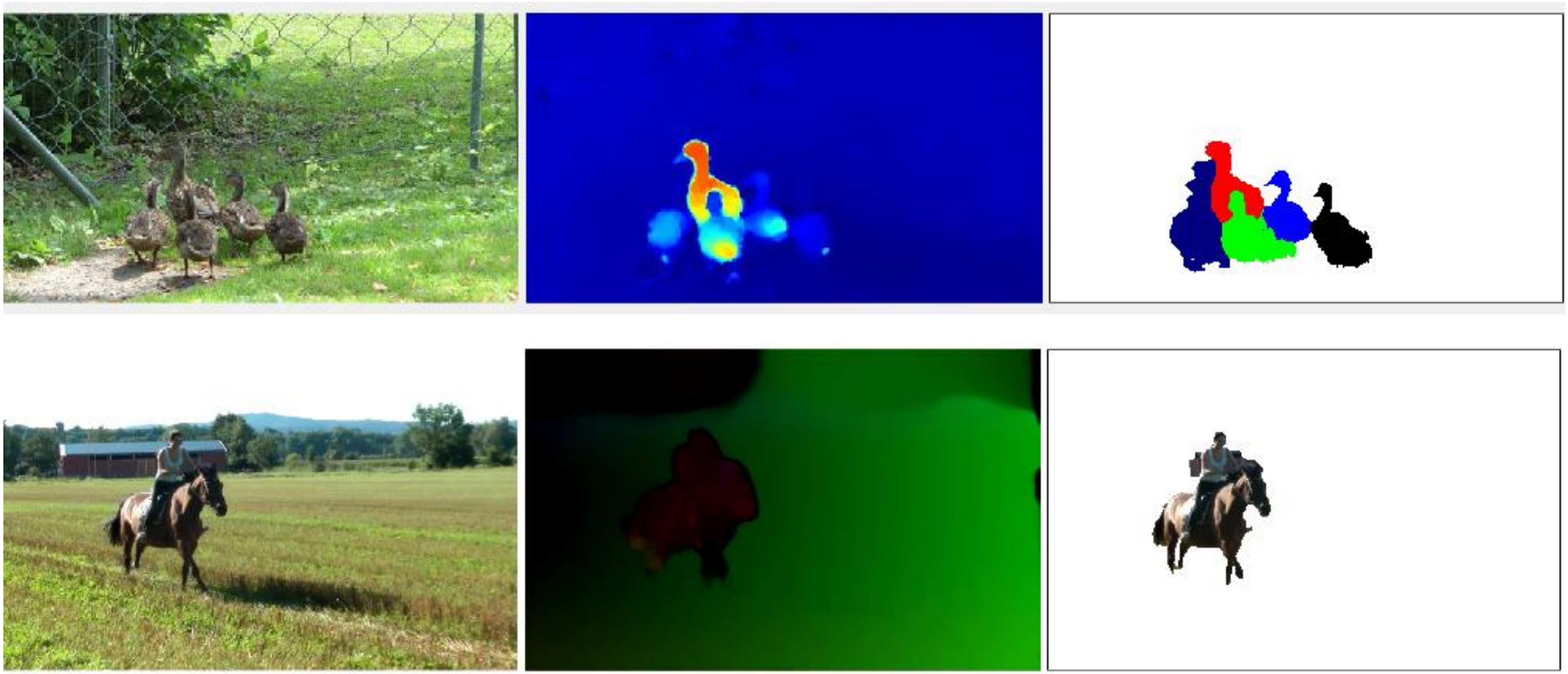
DEMO: "*Remove Background*" tool directly inside "*Picture Format*" tab of MS Power Point software

(preview)

Examples: clustering + spatial regularization

- Self-supervised segmentation [*KernelCut*, Tang et al ECCV'16]

RGBXY | M (motion sensor) | RGBXYM + contrast edges



combining color & boundary objectives

(probabilistic) K-means defines appearance/color consistency

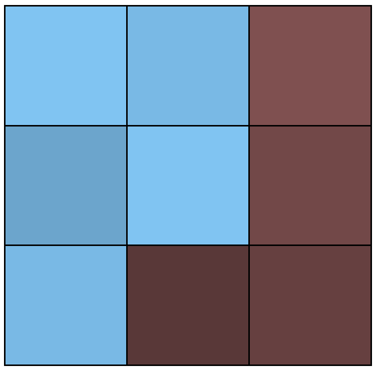
A. Appearance **model fitting**
and boundary regularization

(in the context of image segmentation)

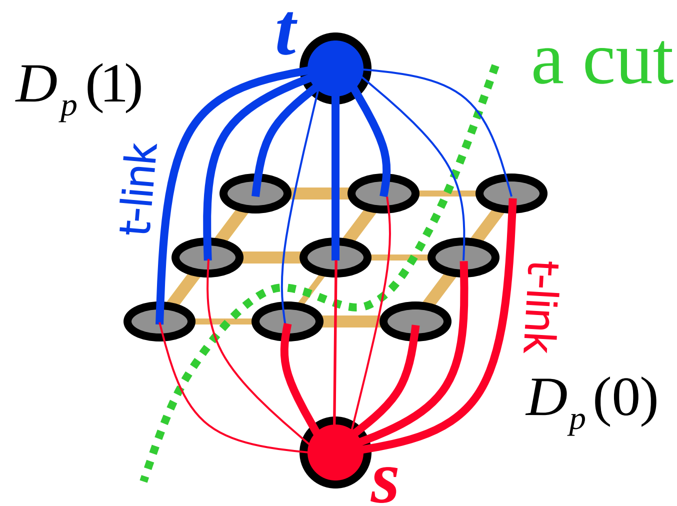
This last portion of
topic 9 is OPTIONAL

Remember simple example
(one color appearance)

optimal segmentation minimizes a **combination of SSE & boundary costs**



remember
regional bias example 1



assume **known** “expected” intensities for object and background

$$D_p(0) = (I^0 - I_p)^2$$

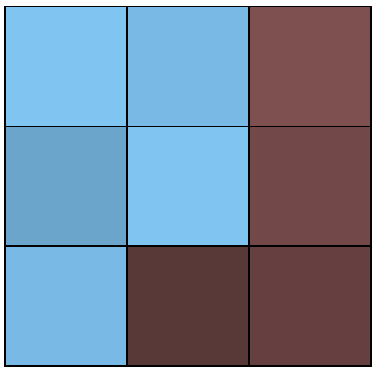
$$D_p(1) = (I^1 - I_p)^2$$

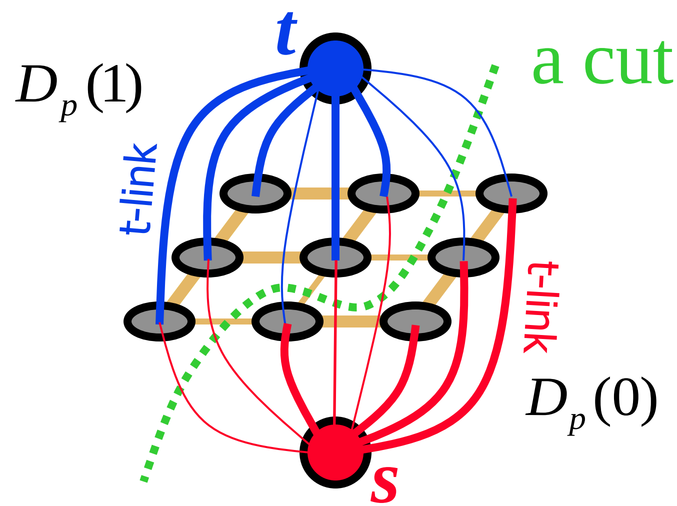
unary potentials — pairwise potentials

$$E(S \mid I^0, I^1) = \sum_{p:S_p=1} (I^1 - I_p)^2 + \sum_{p:S_p=0} (I^0 - I_p)^2 + \sum_{\{pq\}\in N} w_{pq} \cdot [S_p \neq S_q]$$

Remember simple example
(one color appearance)

optimal segmentation minimizes a **combination of SSE & boundary costs**





"expected" intensities of object and background

I^1 and I^0

can be re-estimated

$$D_p(0) = (I^0 - I_p)^2$$

$$D_p(1) = (I^1 - I_p)^2$$

K-means (SSE) loss — with boundary regularization

$$E(S, \underbrace{I^0, I^1}_{\text{extra variables}}) = \sum_{p:S_p=1} (I^1 - I_p)^2 + \sum_{p:S_p=0} (I^0 - I_p)^2 + \sum_{\{pq\}\in N} w_{pq} \cdot [S_p \neq S_q]$$

Block-coordinate descent for $E(S, I^0, I^1)$

- Minimize over labeling S for fixed I^0, I^1

$$E(S, \cancel{I^0, I^1}) = \sum_{p:S_p=0} (I^0 - I_p)^2 + \sum_{p:S_p=1} (I^1 - I_p)^2 + \sum_{\{pq\}\in N} w_{pq} \cdot [S_p \neq S_q]$$

optimal S can be computed using graph cuts

- Minimize over I^0, I^1 for fixed labeling S

fixed for $S = const$

$$E(\cancel{S}, I^0, I^1) = \sum_{p:S_p=0} (I^0 - I_p)^2 + \sum_{p:S_p=1} (I^1 - I_p)^2 + \sum_{\{pq\}\in N} w_{pq} \cdot [S_p \neq S_q]$$

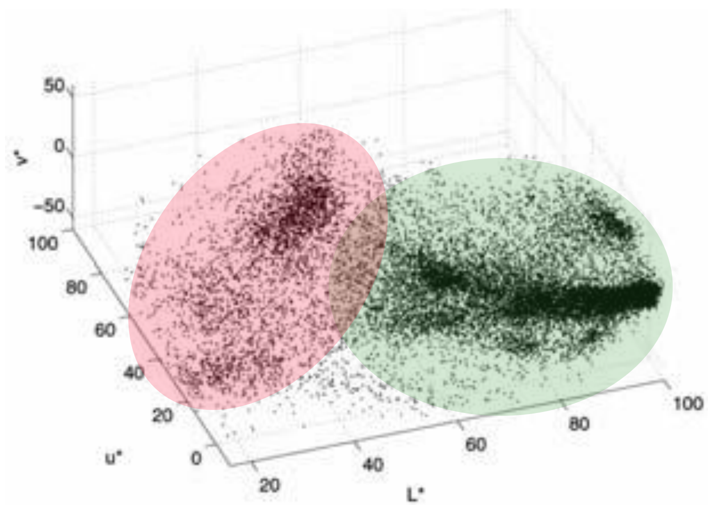
optimal I^1, I^0 can be computed by minimizing squared errors inside object and background segments

$$\hat{I}^0 = \frac{1}{|\bar{S}|} \cdot \sum_{p:S_p=0} I_p \qquad \hat{I}^1 = \frac{1}{|S|} \cdot \sum_{p:S_p=1} I_p$$

mean colors in two segments

Chan-Vese segmentation

(binary case $S_p \in \{0,1\}$)



K-means in RGB space

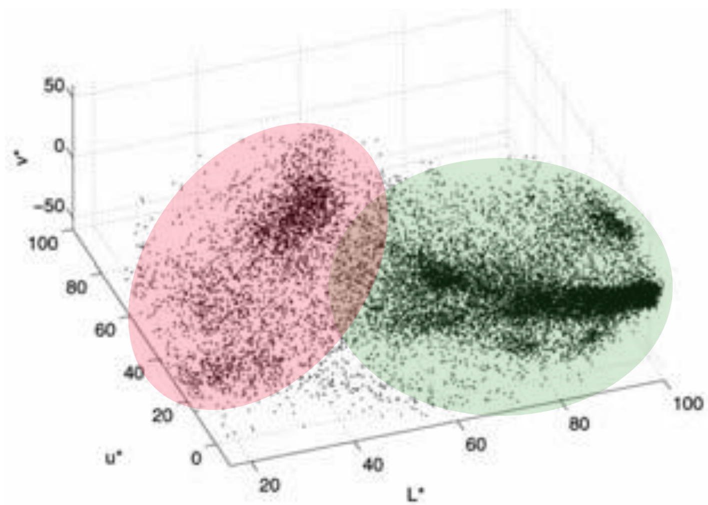
combined with boundary smoothness in XY

$$E(S, I^0, I^1) = \sum_{p:S_p=0} (I_p - I^0)^2 + \sum_{p:S_p=1} (I_p - I^1)^2$$

$$+ \sum_{\{pq\} \in N} w_{pq} \cdot [S_p \neq S_q]$$

Chan-Vese segmentation

(binary case $S_p \in \{0,1\}$)



K-means in RGB space

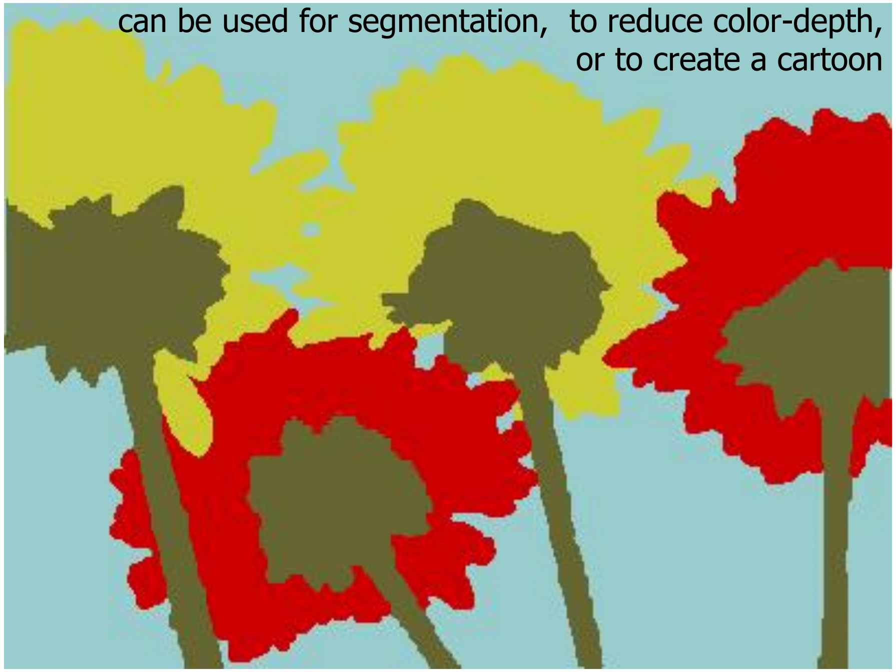
combined with boundary smoothness in XY

$$E(S, I^0, I^1) \quad = \quad \sum_{k=0}^{1} \sum_{p:S_p=k} (I_p - I^k)^2$$

$$+ \sum_{\{pq\}\in N} w_{pq} \cdot [S_p \neq S_q]$$

Chan-Vese segmentation

(could be used for more than 2 labels $S_p \in \{0,1,2,...\}$)



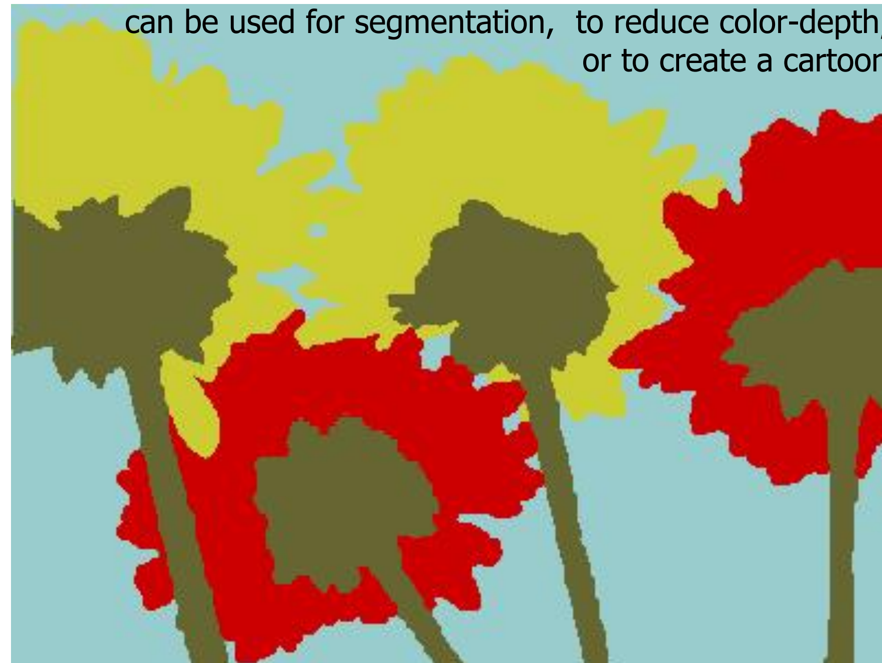
$$E(S, I^0, I^1, \ldots) = \sum_{k=0}^{K} \sum_{p:S_p=k} (I_p - I^k)^2$$

$$+ \sum_{\{pq\}\in N} w_{pq} \cdot [S_p \neq S_q]$$

multi-terminal graph cuts are needed for segmentation step [BVZ, PAMI 2001]

Chan-Vese segmentation

(could be used for more than 2 labels $S_p \in \{0,1,2,\ldots\}$)

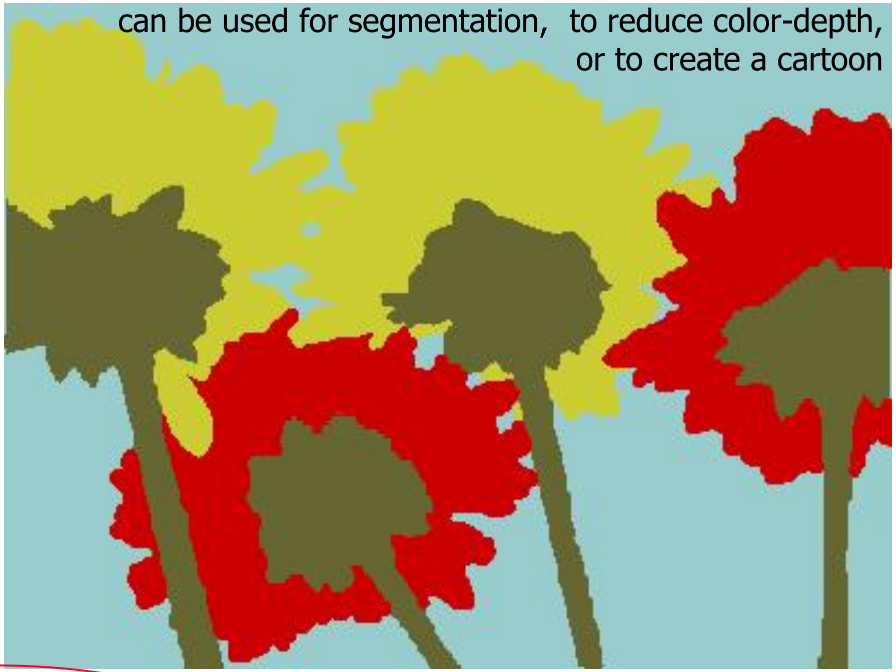


$$E(S, I^0, I^1, \ldots) = \sum_{k=0}^{K} \sum_{p:S_p=k} (I_p - I^k)^2$$

without the smoothing term, this is like "K-means" clustering in the color space

Chan-Vese segmentation

(could be used for more than 2 labels $S_p \in \{0,1,2,...\}$)



$$E(S, I^0, I^1, ...) \quad = \sum_{k=0}^{K} \sum_{p:S_p=k} (I_p - I^k)^2$$

Works well mainly for objects with simple appearance
(approximately one color per segment)

General appearance example
(remember fixed color model example)

$$E(S \mid \theta_0, \theta_1) \;=\; \sum_p -\ln \Pr(I_p \mid \theta_{S_p}) \;+\; \sum_{pq \in N} w_{pq} \cdot [S_p \neq S_q]$$

$S_p \in \{0,1\}$

assuming known

general models (e.g. histograms)

Log-Likelihoods (region)

Spatial smoothness (boundary)

$I_p \in RGB$

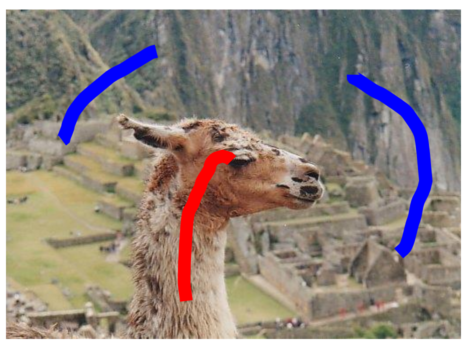
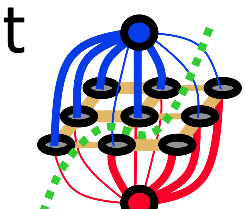


image segmentation, graph cut

[Boykov&Jolly, ICCV2001]



Beyond fixed appearance models

probabilistic K-means loss   with boundary regularization

$$E(S,\theta_0,\theta_1) \;=\; \sum_p -\ln \Pr(I_p \mid \theta_{S_p}) + \sum_{pq\in N} w_{pq}\cdot[S_p \neq S_q]$$

$S_p \in \{0,1\}$

extra variables

general models (e.g. histograms)

Log-Likelihoods (region)

Spatial smoothness (boundary)

$I_p \in RGB$

Models θ_0 , θ_1 can be iteratively re-estimated

iterative image segmentation, Grabcut
(block coordinate descent $S \leftrightarrow \theta_0,\theta_1$)
[Rother, et al. SIGGRAPH'2004]

Block-coordinate descent for $E(S,\theta_0,\theta_1)$

- Minimize over segmentation S for fixed θ_0 , θ_1

$$E(S,\cancel{\theta_0,\theta_1}) = \sum_p -\ln \Pr(I_p \mid \theta_{S_p}) + \sum_{pq \in N} w_{pq} \cdot [S_p \neq S_q]$$

optimal S can be computed using graph cuts

- Minimize over θ_0 , θ_1 for fixed labeling S

fixed for $S=const$

$$E(\cancel{S},\theta_0,\theta_1) = \sum_{p:S_p=0} -\ln \Pr(I_p \mid \theta_0) + \sum_{p:S_p=1} -\ln \Pr(I_p \mid \theta_1) + \sum_{pq \in N} w_{pq} \cdot [S_p \neq S_q]$$

optimal θ_0 , θ_1 can be computed by minimizing the sums of log-likelihoods

$$\hat{\theta}_0 = p^{\bar{S}}$$

distribution of intensities in current bkg. Segment $\bar{S}=\{p:S_p=0\}$

$$\hat{\theta}_1 = p^{S}$$

distribution of intensities in current obj. segment $S=\{p:S_p=1\}$

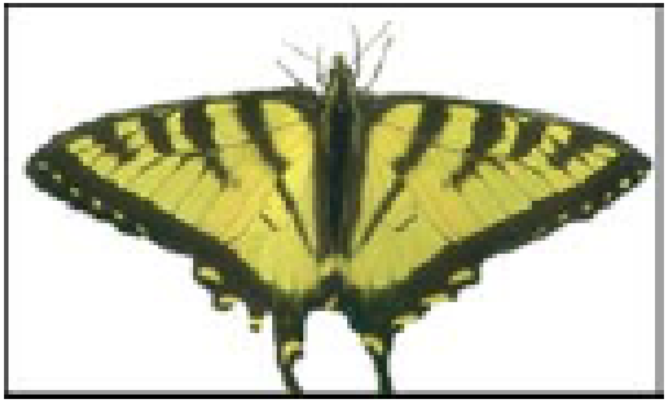
not hard to prove when θ_k are histograms

Iterative learning of color models

(binary case $S_p \in \{0,1\}$)

- GrabCut: iterated graph cuts [Rother et al., SIGGRAPH 04]

$$E(S,\theta_0,\theta_1) = \sum_p -\ln \Pr(I_p \mid \theta_{S_p}) + \sum_{pq\in N} w_{pq} \cdot [S_p \neq S_q]$$



start from models θ_0 , θ_1
based on colors
inside and outside some given box

iterate graph cut segmentation
and model re-estimation
until convergence

Iterative learning of color models

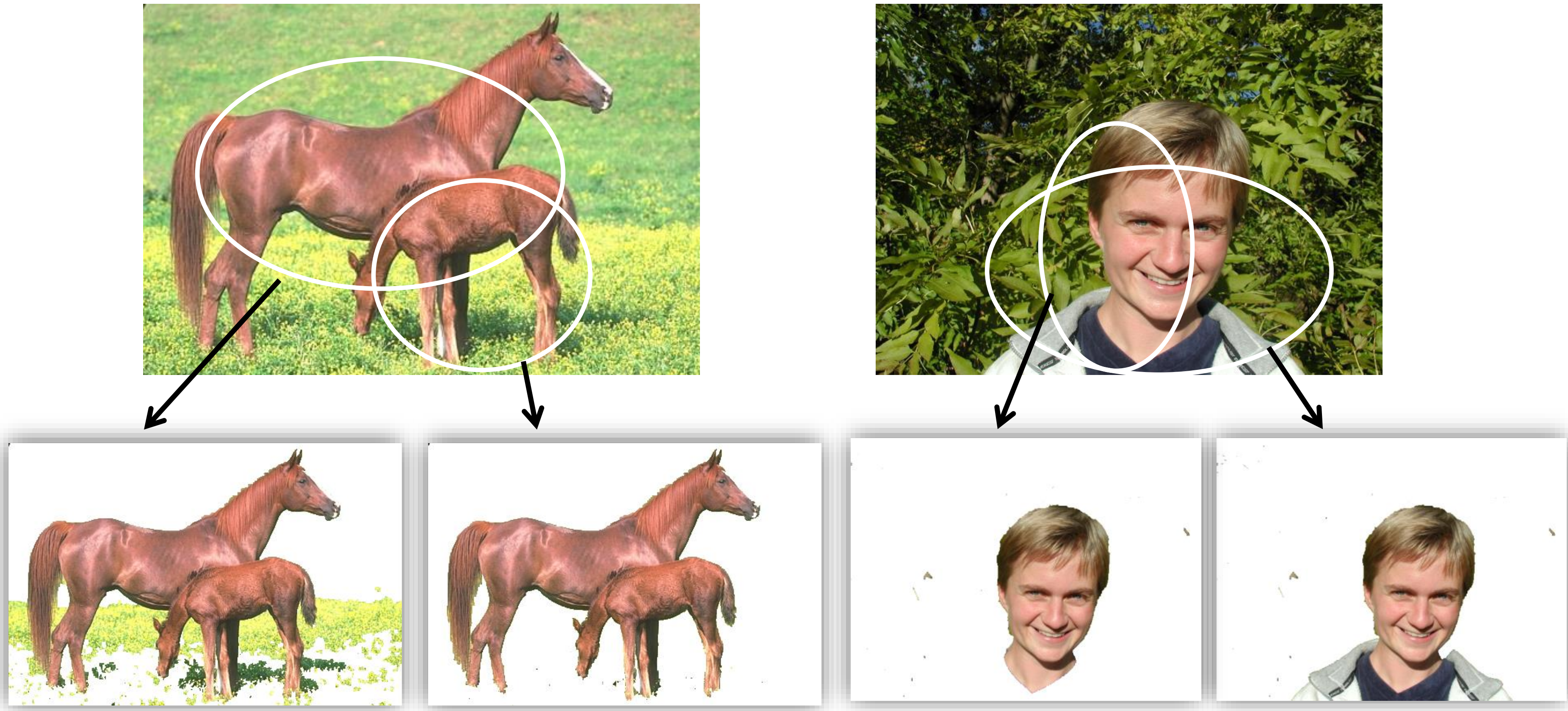
(could be used for more than 2 labels $S_p \in \{0,1,2,...\}$)

- Unsupervised segmentation [Zhu&Yuille, 1996]

$$E(S, \theta_0, \theta_1, \theta_2 ...) = \sum_p -\ln \Pr(I_p \mid \theta_{S_p}) + \sum_{pq \in N} w_{pq} \cdot [S_p \neq S_q] + |labels|$$

initialize models θ_0 , θ_1 , θ_2 , ...
from randomly sampled boxes

iterate segmentation
and model re-estimation
until convergence



E=1.410×10^6 E=1.39×10^6 E=2.41×10^6 E=2.37×10^6

BCD minimization of $E(S,\theta_0,\theta_1)$ **converges to a local minimum**

$$E(S,\theta_0,\theta_1) \;=\; \sum_{p:S_p=0} -\ln \Pr(I_p\,|\,\theta_0) \;+\; \sum_{p:S_p=1} -\ln \Pr(I_p\,|\,\theta_1) \;+ \sum_{pq\in N} w_{pq}\cdot[S_p \neq S_q]$$

Q: Interpretation of this segmentation/clustering energy where θ_i are extra variables?

Statistical answer: it gives *maximum likelihood* (ML) estimation of parameters θ_i

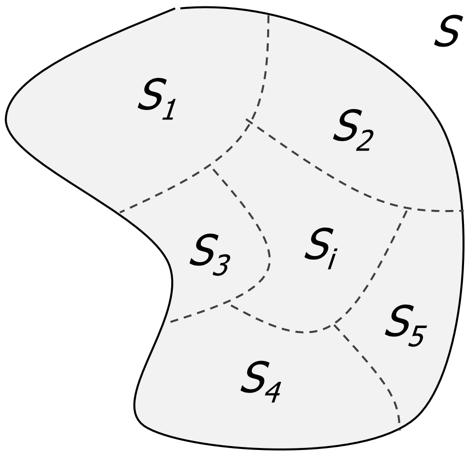
Information theoretic answer: *entropy-based* clustering (…see next slides….)

Interpretation of log-likelihoods: **entropy** of segment intensities

$$-\sum_{p\in S} \ln \Pr(I_p|\theta) \;=\; -|S| \underbrace{\sum_i p_i^S \; \ln p_i^\theta}$$

$$H(S\,|\,\theta)$$

cross entropy of distribution p^S (intensities in S) w.r.t. distribution θ



$S_i = \{p \in S \mid I_p = i\}$

pixiels of color i in S

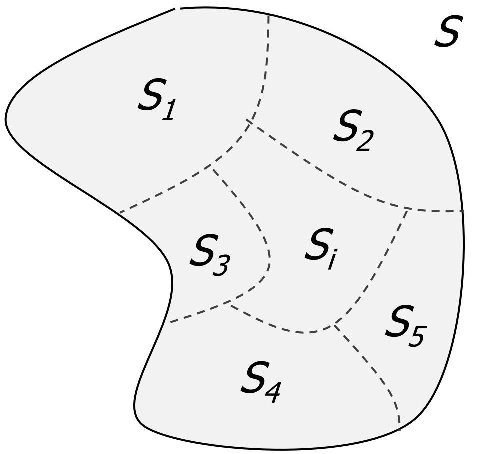
$p_i^s = \frac{|S_i|}{|S|}$ probability of color i in S

Interpretation of log-likelihoods: **entropy** of segment intensities

$$-\sum_{p\in S} \ln \Pr(I_p|\theta) \underset{min\,\theta}{\rightarrow} -|S| \underbrace{\sum_i p_i^S \;\ln p_i^S}$$

$$H(S)$$

entropy of distribution p^S (intensities in S)



$S_i = \{p \in S \mid I_p = i\}$

pixiels of color i in S

$p_i^s = \frac{|S_i|}{|S|}$ probability of color i in S

Interpretation of log-likelihoods: **entropy** of segment intensities

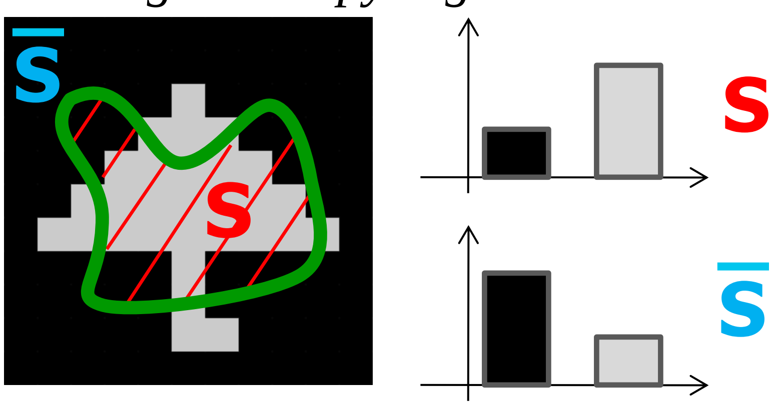
$$\sum_{p:S_p=0} -\ln \Pr(I_p \mid \theta_0) \; + \; \sum_{p:S_p=1} -\ln \Pr(I_p \mid \theta_1) \; + \sum_{pq\in N} w_{pq} \cdot [S_p \neq S_q]$$

$$E(S) = \; |\bar{S}| \cdot H(\bar{S}) \; + \; |S| \cdot H(S) \; + \sum_{pq\in N} w_{pq} [S_p \neq S_q]$$

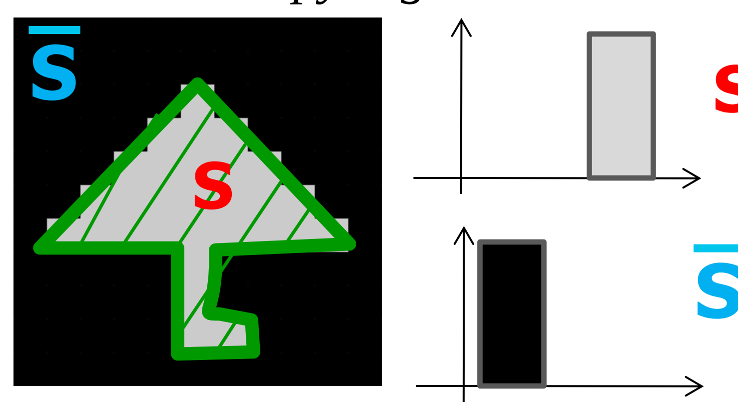
information theoretic energy:
Minimum Description Length
(MDL)
can be interpreted as the number of bits

break image into 2 coherent segments
with low entropy of intensities

high entropy segmentation



low entropy segmentation



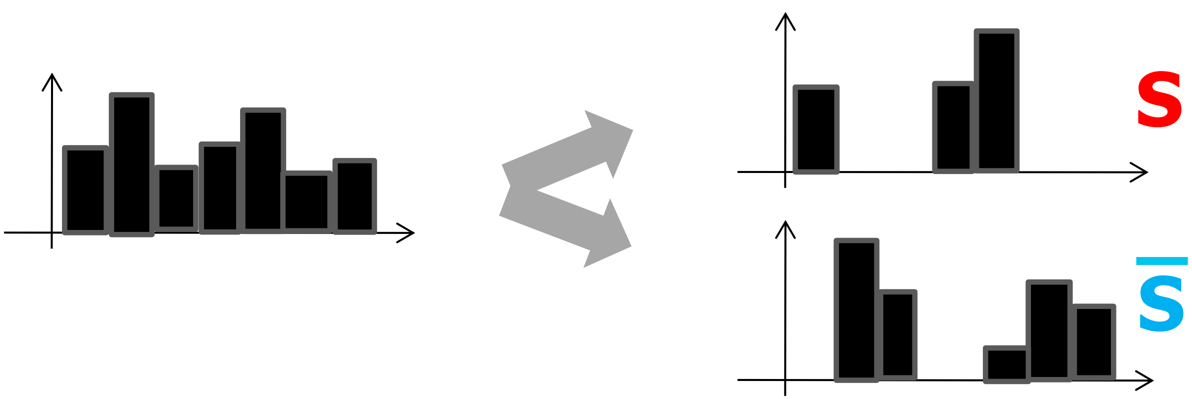
unsupervised image segmentation (like in *Chan-Vese)*

Interpretation of log-likelihoods: **entropy** of segment intensities

$$\sum_{p:S_p=0} -\ln \Pr(I_p \mid \theta_0) \;+\; \sum_{p:S_p=1} -\ln \Pr(I_p \mid \theta_1) \;+ \sum_{pq\in N} w_{pq} \cdot [S_p \neq S_q]$$

$$E(S) = \;|\bar{S}| \cdot H(\bar{S}) \;+\; |S| \cdot H(S) \;+ \sum_{pq\in N} w_{pq}[S_p \neq S_q]$$

break image into 2 coherent segments
with low entropy of intensities



more general than *Chan-Vese* (colors can vary within each segment)

Model Fitting and Color Clustering:

Gaussian models vs Histograms

segments' appearance consistency

edge alignment/regularization

$$\sum_{p:S_p=0} -\ln \Pr(I_p \mid \theta_0) + \sum_{p:S_p=1} -\ln \Pr(I_p \mid \theta_1) + \sum_{pq \in N} w_{pq} \cdot [S_p \neq S_q]$$

normal distributions with means μ_k

$$\Pr(I \mid \theta_k) = N(I \mid \mu_k)$$

$$\sum_{p:S_p=0} (I_p - \mu_0)^2 + \sum_{p:S_p=1} (I_p - \mu_1)^2$$

$$|S| \cdot \mathrm{var}(S) + |\bar{S}| \cdot \mathrm{var}(\bar{S})$$

variance clustering criteria (K-means)

one color segments

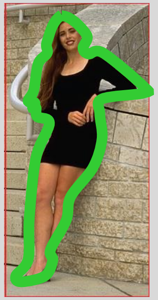
arbitrary histograms $\theta_k = [b^k{}_1, b^k{}_2, \ldots, b^k{}_n]$

$$\Pr(I \mid \theta_k) = b_I^k$$

$$\sum_{p:S_p=0} -\ln \Pr(I_p \mid \theta_0) + \sum_{p:S_p=1} -\ln \Pr(I_p \mid \theta_1)$$

$$|S| \cdot H(S) + |\bar{S}| \cdot H(\bar{S})$$

entropy clustering criteria



"simpler" appearance segments

combining color & boundary objectives

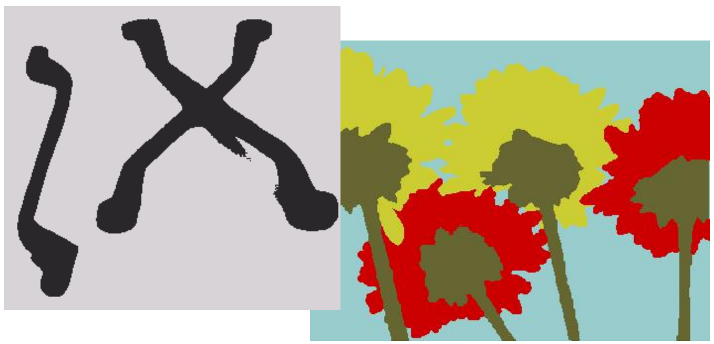
Kernel/affinity clustering objectives define
(color) appearance consistency

B. General **feature consistency**
and boundary regularization

(in the context of image segmentation)

Remember: K-means clustering objective as appearance consistency criterion

- (probabilistic) K-means or model fitting with **simple models** (e.g. Normal/Gaussian) **work fine when data supports such models**.



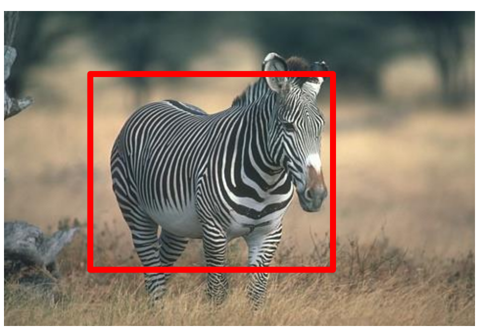
- for more complex objects, fitting **highly descriptive models** (e.g. histograms) is **prone to overfitting**; it barely works even for RGB features:

fitting 2 histograms in RGB
(GrabCut without edges)

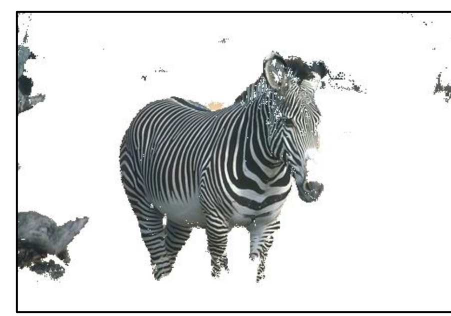
Particularly for higher dimensional features, non-parametric **kernel clustering objectives** are more robust choice for representing "appearance consistency"

Alternative approach:
can use pairwise/kernel clustering

overfitting

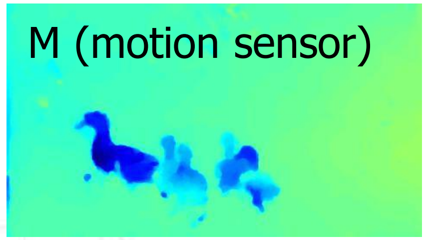


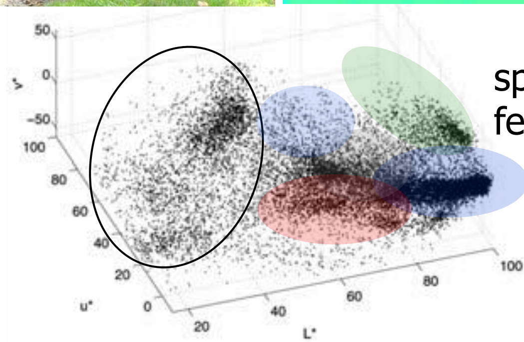
fitting 2 histograms in RGB
(GrabCut without edges)



non-parametric clustering
(Normalized Cut in RGB)

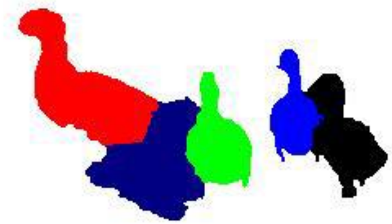
Non-parametric kernel clustering with boundary regularization





Kernel Cut
[M.Tang et al. ECCV 2016]

segmentation S optimizing $E(S)$



<u>Normalized cut</u> in RGBXYM space **combined with <u>boundary regularization</u> in XY**

$$E(S) = -\sum_k \frac{S^{k\,\prime} A_f S^k}{d_f^{\prime} S^k} + \sum_{\{pq\}\in N} w_{pq} \cdot [S_p \neq S_q]$$

$A_f\,[p,q]$ - affinities between all pairs of features f_p in RGBXYM

$d_f\,[p]$ - a "degree" vector (sum of affinities for each p)

Non-parametric kernel clustering with boundary regularization

Kernel Cut
[M.Tang et al. ECCV 2016]

RGBXY | M (motion sensor) | RGBXYM + contrast edges