Image Segmentation with Markov Random Fields (Part 2)

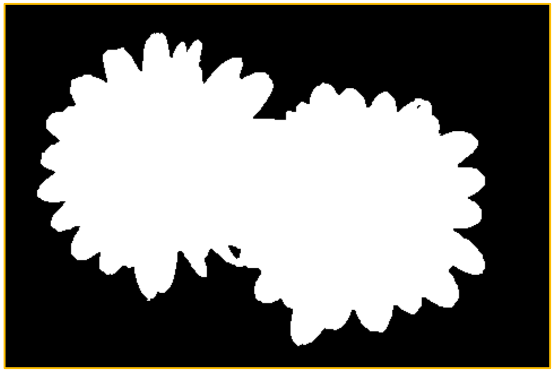
Carsten Rother

Roadmap this lecture

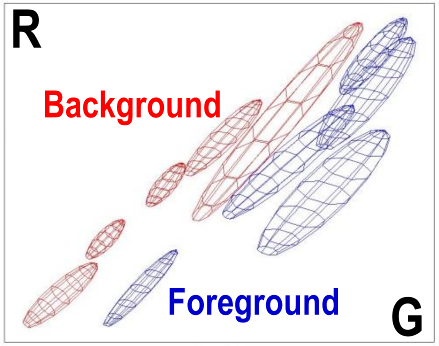
- GrabCut: Interactive Image Segmentation from a Bounding Box
- Joint optimization of segmentation and appearance models
  [Vicente, Kolmogorov, Rother, ICCV 2009]
- A state-of-the-art approach
  [GrabCut in OneCut, Tang, Gorelick, Veksler, Boykov; ICCV 2013]
- Gaussian Markov Random Fields

So far: Brush-Interface Image Segmentation

Image I



Output $\boldsymbol{x}$



$\theta^{F/B}$ Gaussian Mixture models

Energy:

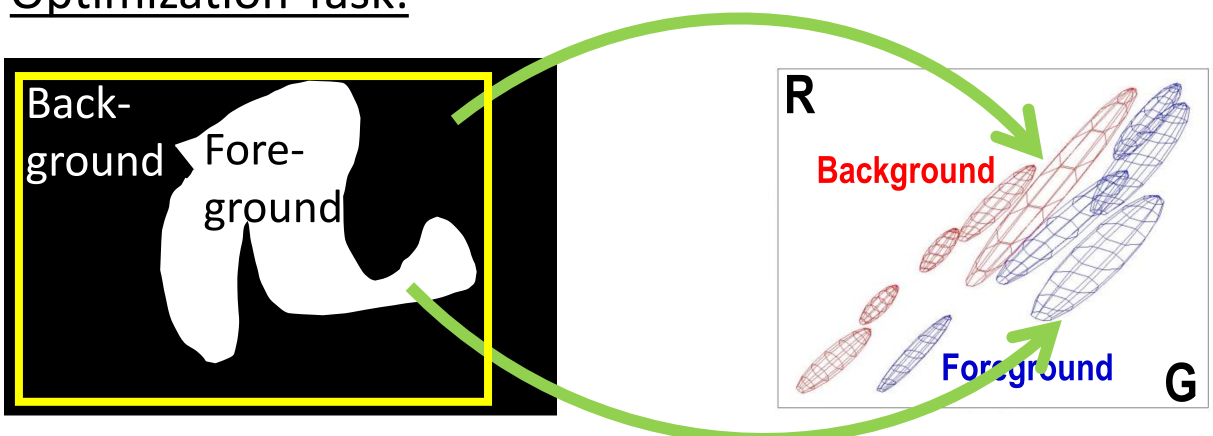
$$E(\boldsymbol{x}) = \sum_i \theta_i(x_i) + \sum_{i,j \in N_4} |x_i - x_j|$$

$$\theta_i(x_i = 1) = -\log P^{blue}(z_i | x_i = 1)$$

$$\theta_i(x_i = 0) = -\log P^{red}(z_i | x_i = 0)$$

Image I

Optimization Task:



Search for a Segmentation

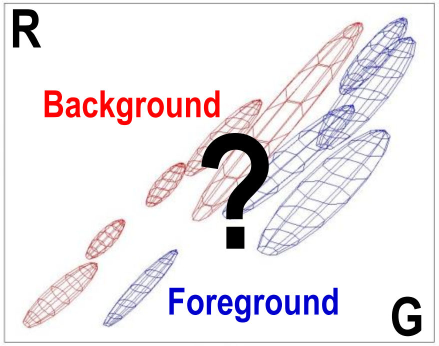
Search for $\theta^{F/B}$ Gaussian Mixture models that explains the fore- and background segment

Note, all pixels in the image now define the foreground and background color models

[Rother et al. Siggraph '04]

Bounding Box Image Segmentation: GrabCut

Image I



$\theta^{F/B}$ Gaussian Mixture models

Energy:

$$E(\boldsymbol{x}, \Theta^F, \Theta^B) = \sum_i \theta_i(x_i, \Theta^F, \Theta^B) + \sum_{i,j \in N_4} |x_i - x_j|$$

$$\theta_i(x_i = 0, \Theta^B) = -\log P^{red}(z_i | x_i = 0, \Theta^B)$$

$$\theta_i(x_i = 1, \Theta^F) = -\log P^{blue}(z_i | x_i = 1, \Theta^F)$$

Note all pixels in the image now define the foreground and background color models

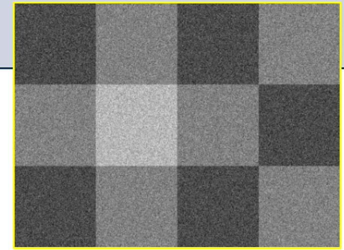
Optimization Problem: $\boldsymbol{x}^* = \underset{\boldsymbol{x}, \Theta^B, \Theta^F}{argmin}\, E(\boldsymbol{x}, \Theta^B, \Theta^F)$

[Rother et al. Siggraph '04]

Joint Model: Motivation

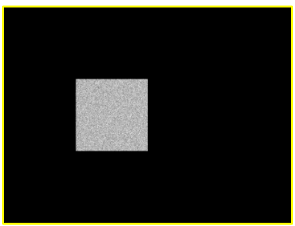
- The joint optimization over appearance (Θ’s) and segmentation has a long history, e.g. Chan-Vese functional, TextonBoost for Semantic segmentation, etc
- The implicit assumption is that the foreground segment has similar colors and also the background segment has similar colors (see next slides)

Question

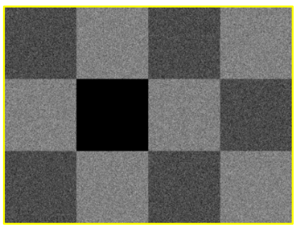


Input Image

Option 1

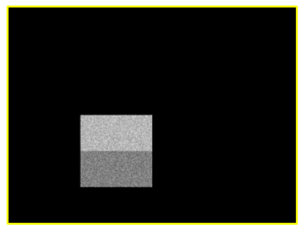


Foreground segmentation

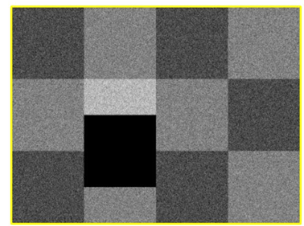


Background segmentation

Option 2

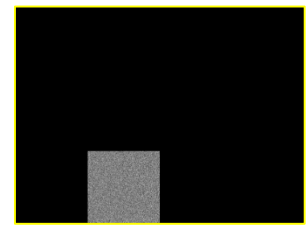


Foreground segmentation

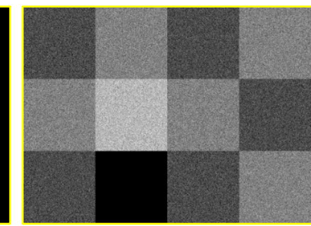


Background segmentation

Option 3



Foreground segmentation

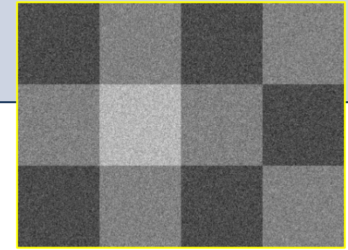


Background segmentation

Which Segmentation do you prefer:

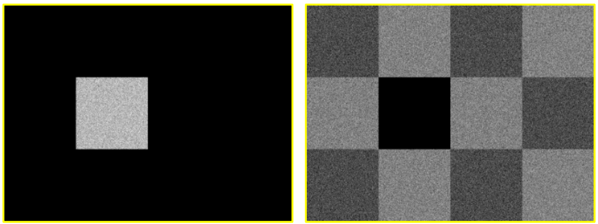
1) Option 1
2) Option 2
3) Option 3

Joint Model: Motivation

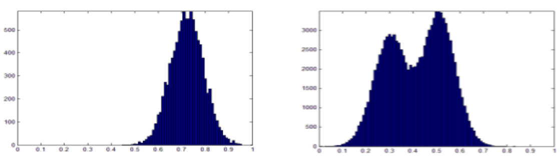


Input Image

Option 1



foreground background

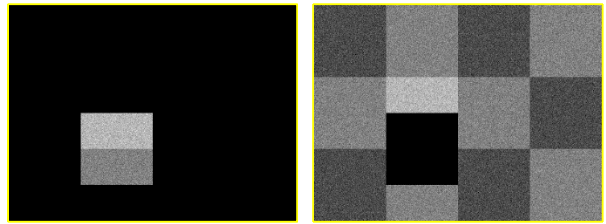


Θ^F Θ^B

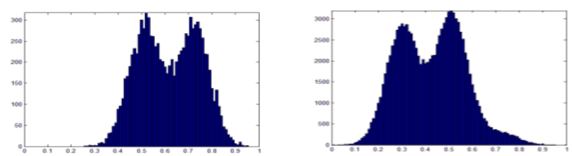
$E_{unary} = 460000$

(optimal color model, proof later)

Option 2



foreground background

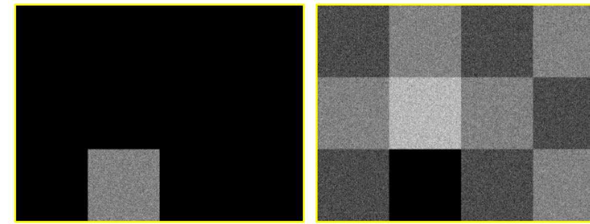


Θ^F Θ^B

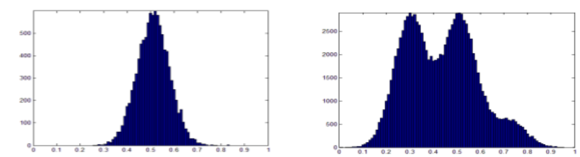
$E_{unary} = 482000$

(optimal color model, proof later)

Option 3



foreground background



Θ^F Θ^B

$E_{unary} = 483000$

(optimal color model, proof later)

$$E(\boldsymbol{x}, \Theta^F, \Theta^B) = \underbrace{\sum_i \theta_i(x_i, \Theta^F, \Theta^B)}_{E_{unary}} + \sum_{i,j \in N_4} |x_i - x_j|$$

$$\theta_i(x_i = 0, \Theta^B) = -\log P^{red}(z_i | x_i = 0, \Theta^B)$$

$$\theta_i(x_i = 1, \Theta^F) = -\log P^{blue}(z_i | x_i = 1, \Theta^F)$$

Joint Model: Optimization

- Block Coordinate Descent (iterative optimization) (see next slides)

- Globally Optimal segmentation (see next section)

Block-Coordinate Descent

Image I

Initial Segmentation

$$\min_{\Theta^B, \Theta^F} E(x, \Theta^F, \Theta^B)$$

Estimating the colour distributions using pixels from fore- and background segment respectively. Keep segmentation fixed.

(done as before using GMMs)

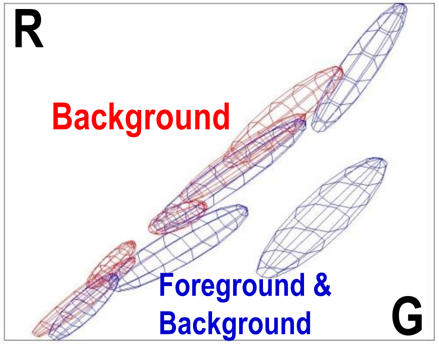
$$\min_{\boldsymbol{x}} E(x, \Theta^F, \Theta^B)$$

Graph cut to infer segmentation using fixed colour models

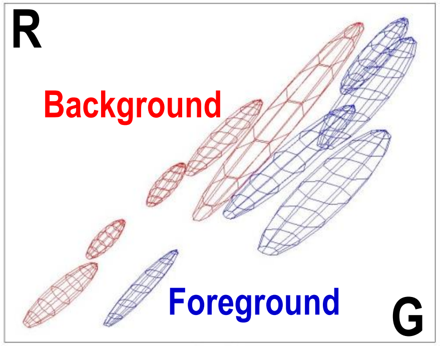
[Rother et al. Siggraph '04]

Block-Coordinate Descent

Result



Initial Color Models



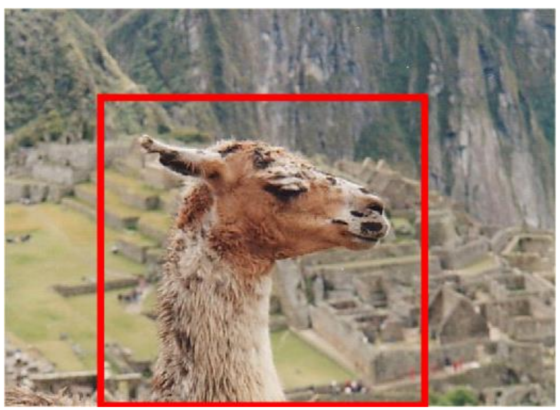
Final Color Models

Optimization over $\Theta's$ helps in other cases too

Input

Color models
from brushes only
[Boykov&Jolly ‘01]

Color models
from all image pixels
[GrabCut ‘04]



Input

Color models
from all image pixels
[GrabCut ‘04]

How to initialize?

Idea 1:

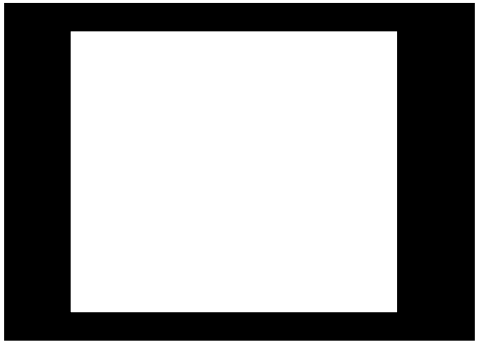
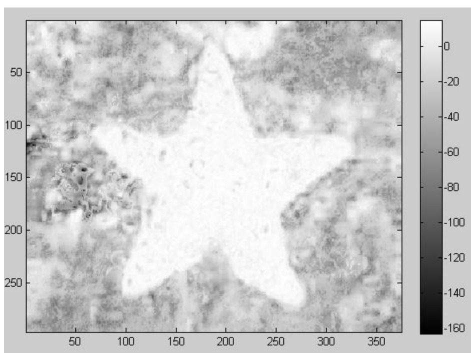


Image I

Idea 2:



$-\log P^{red}(z_i|x_i = 0)$

Take as foreground 33% of pixels where $-\log P^{red}(z_i|x_i = 0)$ is high (result sketched)

How to initialize?

Idea 3: Use Parametric Maxflow [Kolmogorov, Boykov, Rother ICCV 07]

Input image

parametric maxflow gives you a sequence of solutions (faked here)
[Kolmogorov et al. ICCV 2007]

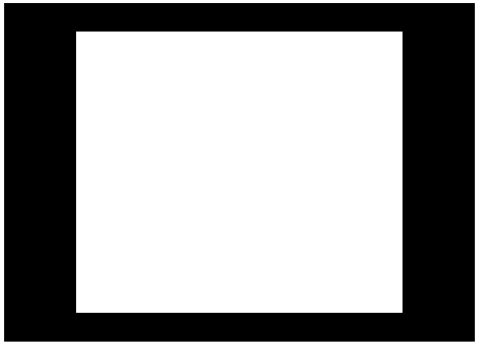
Initialize with a result that is close to the bounding box

Paramteric Maxflow

<u>Energy:</u> $E(\boldsymbol{x},\lambda) = \sum_i \theta_i(x_i) + \sum_{i,j \in N_4} |x_i - x_j| + \sum_i \lambda x_i$

<u>Optimization:</u> find all possible segmentations (for any λ)

$Extreme\ case\!: \lambda = -\infty$



$Extreme\ case\!: \lambda = \infty$

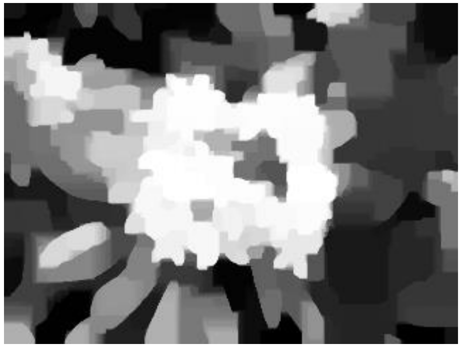
Paramteric Maxflow - Example

$$E(\boldsymbol{x},\lambda) = \sum_i \theta_i(x_i) + \omega \sum_{i,j\in N_4} |x_i - x_j| + \sum_i \lambda x_i \qquad \theta_i(0) = z_i \quad \theta_i(1) = 255 - z_i$$

9389 solutions (superimposed) (ω = 0.01)

6058 solutions (superimposed) (ω = 0.1)

Input Image



2027 solutions (superimposed) (ω = 1)

214 solutions (superimposed) (ω = 10)

Roadmap this lecture

- GrabCut: Interactive Image Segmentation from a Bounding Box
- Joint optimization of segmentation and appearance models
  [Vicente, Kolmogorov, Rother, ICCV 2009]
- A state-of-the-art approach
  [GrabCut in OneCut, Tang, Gorelick, Veksler, Boykov; ICCV 2013]
- Gaussian Markov Random Fields

Recap: Block-Coordinate Descent

Image I

Initial Segmentation

$$\min_{\Theta^B,\Theta^F} E(x,\Theta^F,\Theta^B)$$

Estimating the colour distributions using pixels from fore- and background segment respectively

(done as before using GMMs)

$$\min_{\boldsymbol{x}} E(x,\Theta^F,\Theta^B)$$

Graph cut to infer segmentation using fixed colour models

<u>Optimization Problem:</u> $\boldsymbol{x}^* = \underset{\boldsymbol{x},\,\Theta^B,\,\Theta^F}{argmin}\, E(\boldsymbol{x},\Theta^B,\Theta^F)$

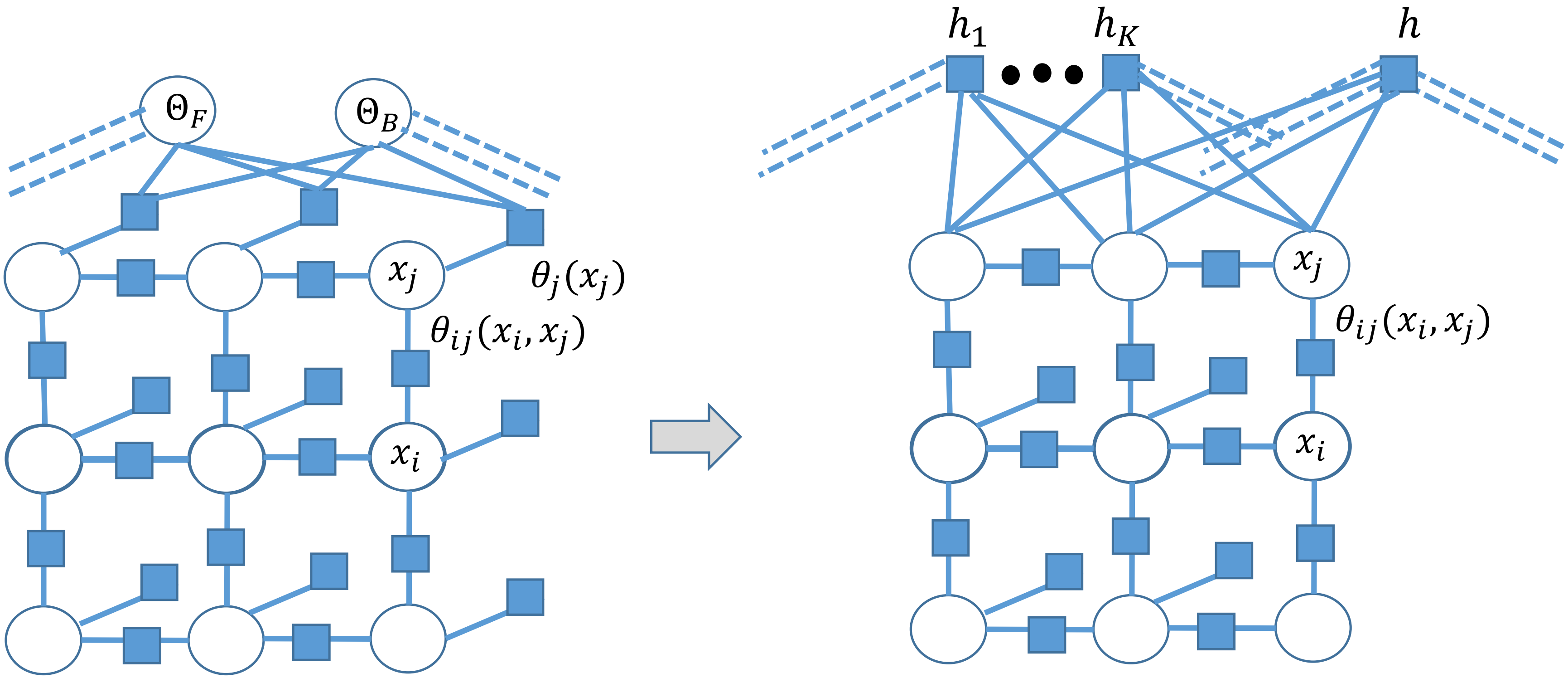
[Rother et al. Siggraph '04]

Our goal

$$\boldsymbol{x}^* = \underset{\boldsymbol{x},\Theta^F,\Theta^B}{argmin}\, E(\boldsymbol{x},\Theta^F,\Theta^B) = \underset{\boldsymbol{x},\Theta^F,\Theta^B}{argmin}\, [\, \sum_i \theta_i(x_i,\Theta^F,\Theta^B) \;+\; \sum_{i,j\in N_4} |x_i - x_j|\,]$$

<u>Reformulate to:</u>

$$\boldsymbol{x}^* = \underset{\boldsymbol{x}}{argmin}\, E'(\boldsymbol{x}) = \underset{\boldsymbol{x}}{argmin}\, [\, h\big(n(\boldsymbol{x})\big) + \sum_k h_k(n_k(\boldsymbol{x})) \;+\; \sum_{i,j\in N_4} |x_i - x_j|\,]$$



Our goal: Visualization of the final energy

$$x^* = \underset{x}{argmin}\, E'(x) = \underset{x}{argmin}\, [\, h(n^F) + \sum_k h_k(n_k^F) + \sum_{i,j \in N_4} |x_i - x_j| \,]$$

$h(n^F) = \; g(n) + g(n - n^F)$

$h_k(n_k^F) = \; -g(n_k^F) - g(n_k - n_k^F)$

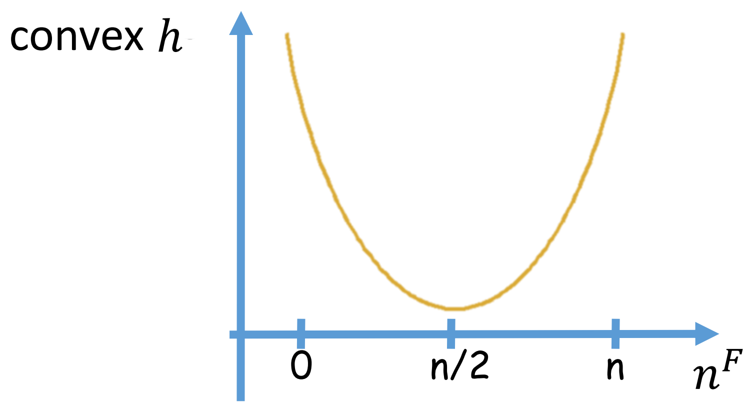
with $g(z) = z \log z$

$n^F = \sum_i x_i$ number of foreground pixel

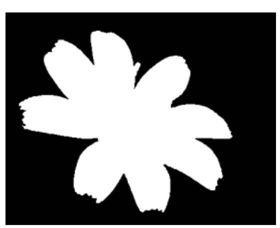
$n_k^F = \sum_i x_i$ number of foreground pixel in bin k

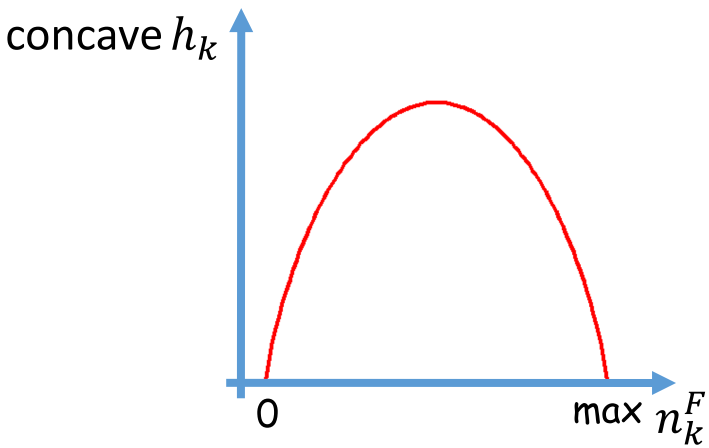
$n_k =$ number of pixels in bin k (independent of $\boldsymbol{x}$)

$n =$ number of pixels in image (independent of $\boldsymbol{x}$)



Prefers “equal area” segmentation





Each color either fore- or background

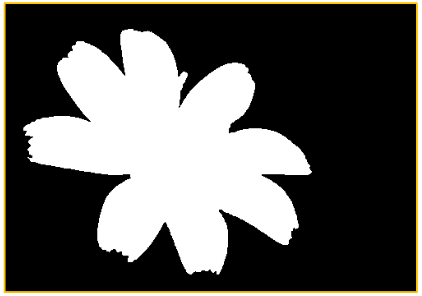
Reminder: Gaussian Mixture Model

$$\boldsymbol{x}^* = \underset{\boldsymbol{x},\Theta^F,\Theta^B}{argmin}\, E(\boldsymbol{x},\Theta^F,\Theta^B) = \underset{\boldsymbol{x},\Theta^F,\Theta^B}{argmin} \sum_i \theta_i(x_i,\Theta^F,\Theta^B) \; + \sum_{i,j\in N_4} |x_i - x_j|$$

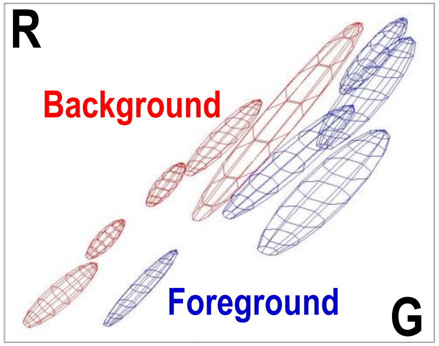
$$\theta_i(x_i = 0, \Theta^B) = -\log P^{red}(z_i|x_i = 0, \Theta^B)$$

$$\theta_i(x_i = 1, \Theta^F) = -\log P^{blue}(z_i|x_i = 1, \Theta^F)$$

Image $\boldsymbol{z}$



given $\boldsymbol{x}$



$\Theta^{F/B}$ Gaussian Mixture models

Histogram Model

$$\boldsymbol{x}^* = \underset{\boldsymbol{x},\Theta^F,\Theta^B}{argmin}\, E(\boldsymbol{x},\Theta^F,\Theta^B) = \underset{\boldsymbol{x},\Theta^F,\Theta^B}{argmin} \sum_i \theta_i(x_i,\Theta^F,\Theta^B) + \sum_{i,j\in N_4} |x_i - x_j|$$

Image

discretized in bins

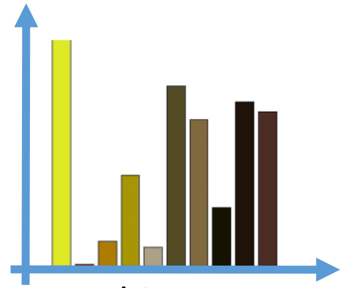


Image histogram

$K = 16^3$

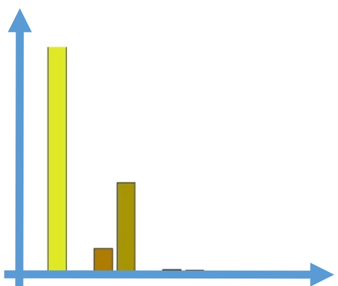
$\mathrm{b}(\mathrm{z_i}) \in \{1,..K\}$ is the bin in which $\mathrm{z_i}$ falls in

$$\theta_i(x_i = 1,\Theta^F) = -\log \Theta^F_{\mathrm{b(z_i)}}$$
$$\theta_i(x_i = 0,\Theta^B) = -\log \Theta^B_{\mathrm{b(z_i)}}$$

<u>Our Goal is to find optimal:</u>

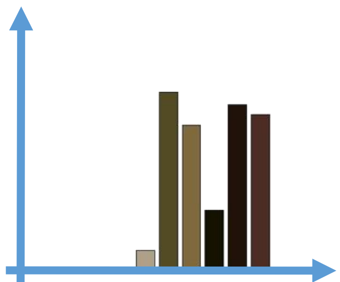
- Segmentation $\boldsymbol{x}$,
- K-value vector Θ^F_k
- K-value vector Θ^B_k

Θ^F



foreground distribution

Θ^B



background distribution

$$\Theta_k^{F/B} \in [0,1]$$

$$\sum_k \Theta_k^{F/B} = 1$$

Derive optimal appearance model

$$\boldsymbol{x}^* = \underset{\boldsymbol{x},\Theta^F,\Theta^B}{argmin} \sum_i \theta_i(x_i,\Theta^F,\Theta^B) \; + \sum_{i,j\in N_4} |x_i - x_j|$$

$$= \underset{\boldsymbol{x}}{argmin} \;\; \underset{\Theta^F,\Theta^B}{\min} \sum_i \theta_i(x_i,\Theta^F,\Theta^B) \; + \sum_{i,j\in N_4} |x_i - x_j|$$

$$\theta_i(x_i = 1, \Theta^F) = -\log \Theta^F_{\mathrm{b(z_i)}}$$

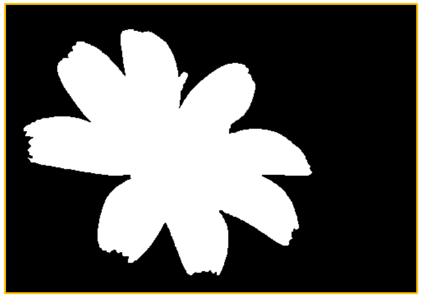
$$\theta_i(x_i = 0, \Theta^B) = -\log \Theta^B_{\mathrm{b(z_i)}}$$

It is: $\Theta_k^{F*} = n_k^F \,/\, n^F$ (i.e. is the empirical histogram, Θ_k^{F*} means optimal solution (Θ_k^{B*} done in the same fashion)

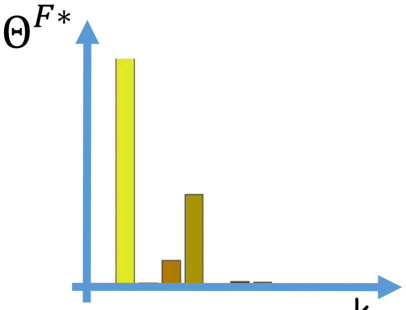
$n^F = \sum_i x_i$ number of foreground pixel (n^B background in same fashion)

$n_k^F = \sum_i x_i$ number of foreground pixel in bin k (n_k^B background in same fashion)

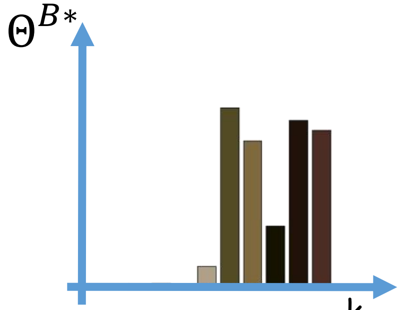
discretized image



given $\boldsymbol{x}$



foreground distribution



background distribution

Proof: $\Theta_{k,F}^{*} = n_k^F \,/\, n^F$

Example with two bins

$n = 100, \quad n_1 = 30, \quad n_2 = 70$

what is the optimal θ_1 and θ_2 ?

objective

$$\min_{\theta_1, \theta_2} -n_1 \log \theta_1 - n_2 \log \theta_2 \; ; \quad \text{sbj. to } \theta_1 + \theta_2 = 1$$

$$\rightsquigarrow \min_{\theta_1, \theta_2} \underbrace{-n_1 \log(1-\theta_2) - n_2 \log \theta_2}_{f(\theta_2)}$$

(substitute Θ_1)

$$\frac{\partial f(\theta_2)}{\partial \theta_2} = \frac{+n_1}{1-\theta_2} - \frac{n_2}{\theta_2} \stackrel{!}{=} 0$$

$$\rightsquigarrow \frac{+n_1 \theta_2 - n_2 (1-\theta_2)}{\theta_2 (1-\theta_2)} \stackrel{!}{=} 0$$

$$\rightsquigarrow +n_1 \theta_2 - n_2 + n_2 \theta_2 \stackrel{!}{=} 0$$

$$\rightsquigarrow \theta_2 = \frac{n_2}{n_1 + n_2}$$

$$\rightsquigarrow \theta_1 = \frac{n_1}{n_1 + n_2}$$

Re-write the energy

$$\boldsymbol{x}^* = \underset{\boldsymbol{x}}{argmin} \min_{\Theta^F, \Theta^B} \sum_i \theta_i(x_i, \Theta^F, \Theta^B) + \sum_{i,j \in N_4} |x_i - x_j|$$

$$\theta_i(x_i = 1, \Theta^F) = -\log \Theta^F_{\mathrm{b(z_i)}}$$

$$\theta_i(x_i = 0, \Theta^B) = -\log \Theta^B_{\mathrm{b(z_i)}}$$

Given: $\Theta_k^{F*} = n_k^F \,/\, n^F$

$$\boldsymbol{x}^* = \underset{\boldsymbol{x}}{argmin} \sum_i \theta_i(x_i) + \sum_{i,j \in N_4} |x_i - x_j|$$

$$\theta_i(x_i = 1\,) = -\log\, [n_k^F \,/\, n^F]$$

$$\theta_i(x_i = 0) = -\log\, [n_k^B \,/\, n^B]$$

Remember:
Depends on the full labeling $\boldsymbol{x}$!

$n^F = \sum_i x_i$ number of foreground pixel

$n_k^F = \sum_i x_i$ number of foreground pixel in bin k

Re-write the energy

$$\boldsymbol{x}^* = \underset{\boldsymbol{x}}{argmin} \sum_k [-\mathrm{n}_\mathrm{k}^\mathrm{F} \log\,[n_k^F \,/\, n^F] - \mathrm{n}_\mathrm{k}^\mathrm{B} \log\,[n_k^B \,/\, n^B]\,] + \sum_{i,j \in N_4} |x_i - x_j|$$

Sum over all ***K*** bins in the histogram

$n^F = \sum_i x_i$ number of foreground pixel

$n_k^F = \sum_i x_i$ number of foreground pixel in bin k

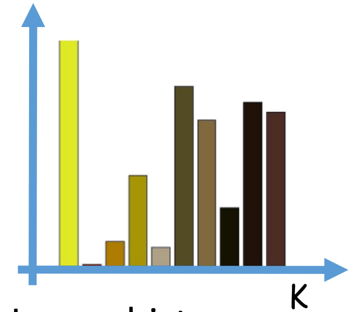


Image histogram

$$\boldsymbol{x}^* = \underset{\boldsymbol{x}}{argmin}\, [\, h(n^F) + \sum_k h_k\left(n_k^F\right) + \sum_{i,j \in N_4} \left|x_i - x_j\right| \,]$$

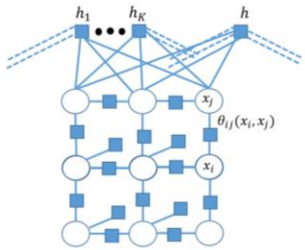
$$h(n^F) = \; g(n) + g(n - n^F)$$

$$h_k\left(n_k^F\right) = \; -g\left(n_k^F\right) - g\left(n_k - n_k^F\right)$$

with $g(z) = z \log z$

$n_k =$ number of pixels in bin k (independent of $\boldsymbol{x}$)

$n =$ number of pixels in image (independent of $\boldsymbol{x}$)



Proof: Previous re-formulation

$$\sum_k -n_k^F \log \frac{n_k^F}{n^F} - n_k^B \log \frac{n_k^B}{n^B} \qquad \boxed{\begin{array}{l} n = n^F + n^B \\ n_k = n_k^B + n_k^F \end{array}}$$

$$\rightsquigarrow \sum_k -n_k^F \log \frac{n_k^F}{n^F} - (n_k - n_k^F) \log \frac{n_k - n_k^F}{n - n^F}$$

$$\rightsquigarrow \sum_k -n_k^F \log n_k^F + n_k^F \log n_F - n_k \log \frac{n_k - n_k^F}{n - n^F} + n_k^F \log \frac{n_k - n_k^F}{n - n^F}$$

$$\rightsquigarrow \sum_k -n_k^F \log n_k^F + n_k^F \log n^F - n_k \log (n_k - n_k^F) + n_k \log (n - n^F)$$
$$+ n_k^F \log (n_k - n_k^F) - n_k^F \log (n - n^F)$$

$$\rightsquigarrow n^F \log n^F + n \log (n - n^F) - n^F \log (n - n^F) \qquad \boxed{\sum_k n_k = n}$$
$$- \sum_k n_k^F \log n_k^F + n_k \log (n_k - n_k^F) - n_k^F \log (n_k - n_k^F)$$

$$\rightsquigarrow n^F \log n^F + (n - n^F) \log (n - n^F)$$
$$- \sum_k n_k^F \log n_k^F + (n_k - n_k^F) \log (n_k - n_k^F)$$

$$\rightsquigarrow h(n^F) + \sum_k h_k(n_k^F)$$

Visualization of the energy

$$\boldsymbol{x}^* = \underset{\boldsymbol{x}}{argmin}\, E'(\boldsymbol{x}) = \underset{\boldsymbol{x}}{argmin}\, [\, h(n^F) + \sum_k h_k(n_k^F) + \sum_{i,j \in N_4} |x_i - x_j|\,]$$

$$h(n^F) = g(n^F) + g(n - n^F)$$

$$h_k(n_k^F) = -g(n_k^F) - g(n_k - n_k^F)$$

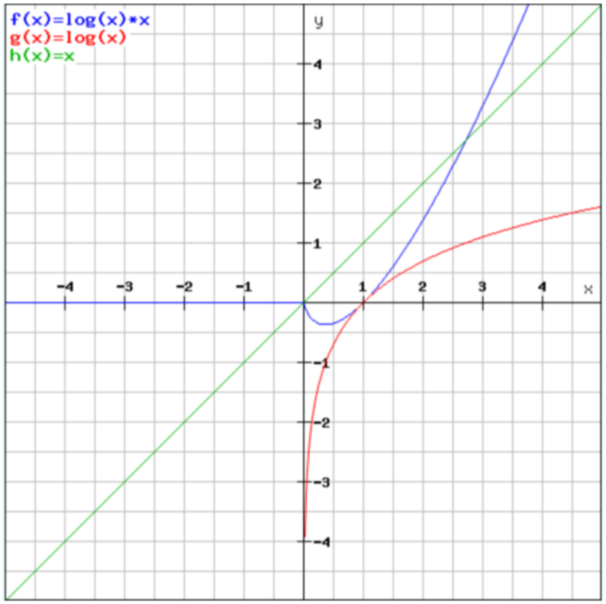
with $g(z) = z \log z$

$n^F = \sum_i x_i$ number of foreground pixel

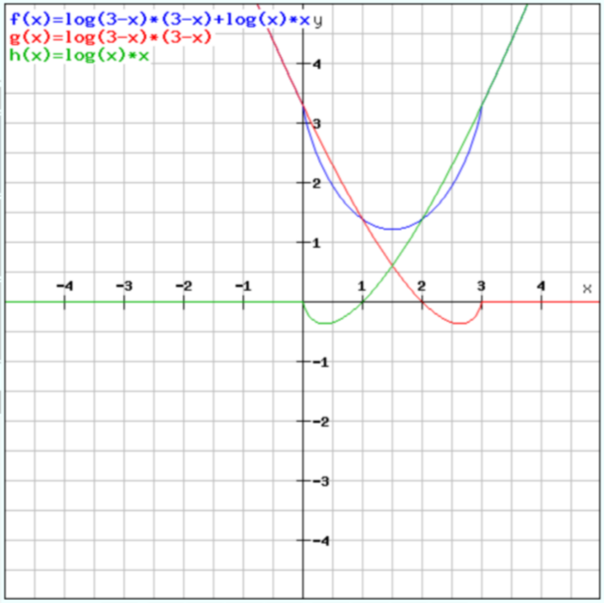
$n_k^F = \sum_i x_i$ number of foreground pixel in bin k

$n_k =$ number of pixels in bin k (independent of $\boldsymbol{x}$)

$n =$ number of pixels in image (independent of $\boldsymbol{x}$)



Visualization: $g(z) = z \log z$



Visualization: $\mathrm{h}(z)$

Proof: minimum of function h

$$h(z) = z \log z + (n-z)\log(n-z)$$

$$\frac{\partial h(z)}{\partial z} = \frac{z}{z} + \log z + \frac{(n-z)}{(n-z)\cdot(-1)} - \log(n-z)$$

$$= \log z - \log(n-z) \overset{!}{=} 0$$

$$\leadsto \quad z = n - z$$

$$\leadsto \quad z = \frac{n}{2}$$

Visualization of the energy

$$\boldsymbol{x}^* = \underset{\boldsymbol{x}}{argmin}\, E'(\boldsymbol{x}) = \underset{\boldsymbol{x}}{argmin}\, [\, h(n^F) + \sum_k h_k(n_k^F) + \sum_{i,j\in N_4} |x_i - x_j|\,]$$

$$h(n^F) = \; g(n^F) + g(n - n^F)$$

$$h_k(n_k^F) = \; -g(n_k^F) - g(n_k - n_k^F)$$

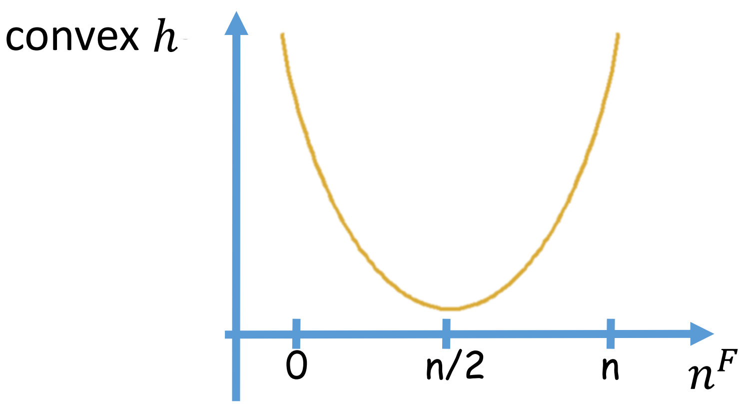
with $g(z) = z \log z$

$n^F = \sum_i x_i$ number of foreground pixel

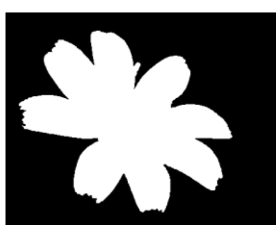
$n_k^F = \sum_i x_i$ number of foreground pixel in bin k

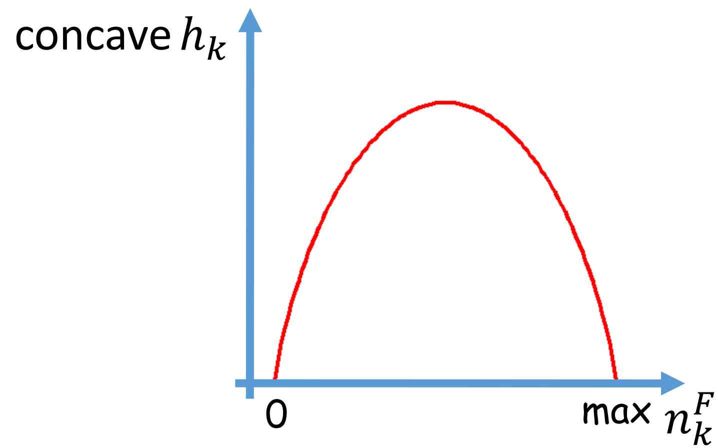
$n_k =$ number of pixels in bin k (independent of $\boldsymbol{x}$)

$n =$ number of pixels in image (independent of $\boldsymbol{x}$)



Prefers “equal area” segmentation





Each color either fore- or background

Some Results

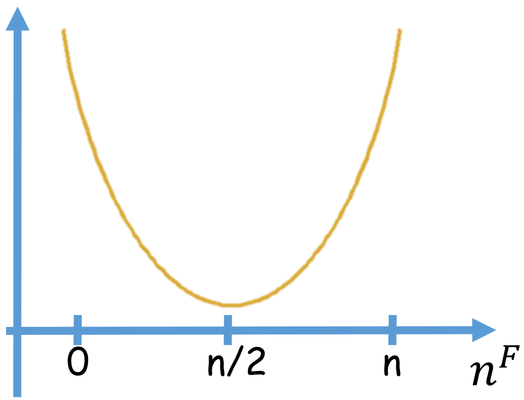
Input GrabCut Global Optimum

Globally Optimal in 61% of cases … but runtime is ~90sec for a 250 x 160 image

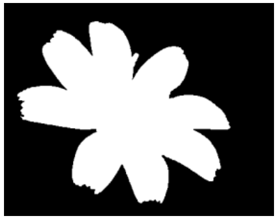
Adapting the weighting

0.3 h 0.4 h h 1.5 h

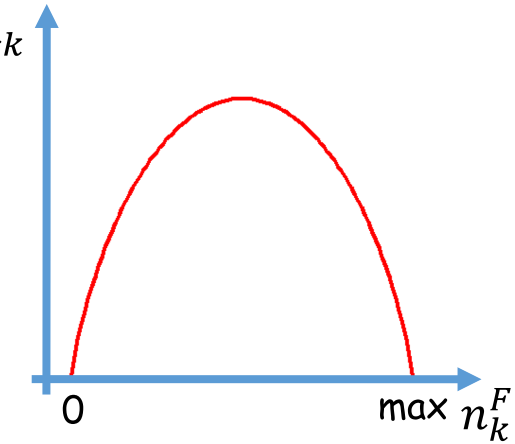
convex h



Prefers “equal area” segmentation

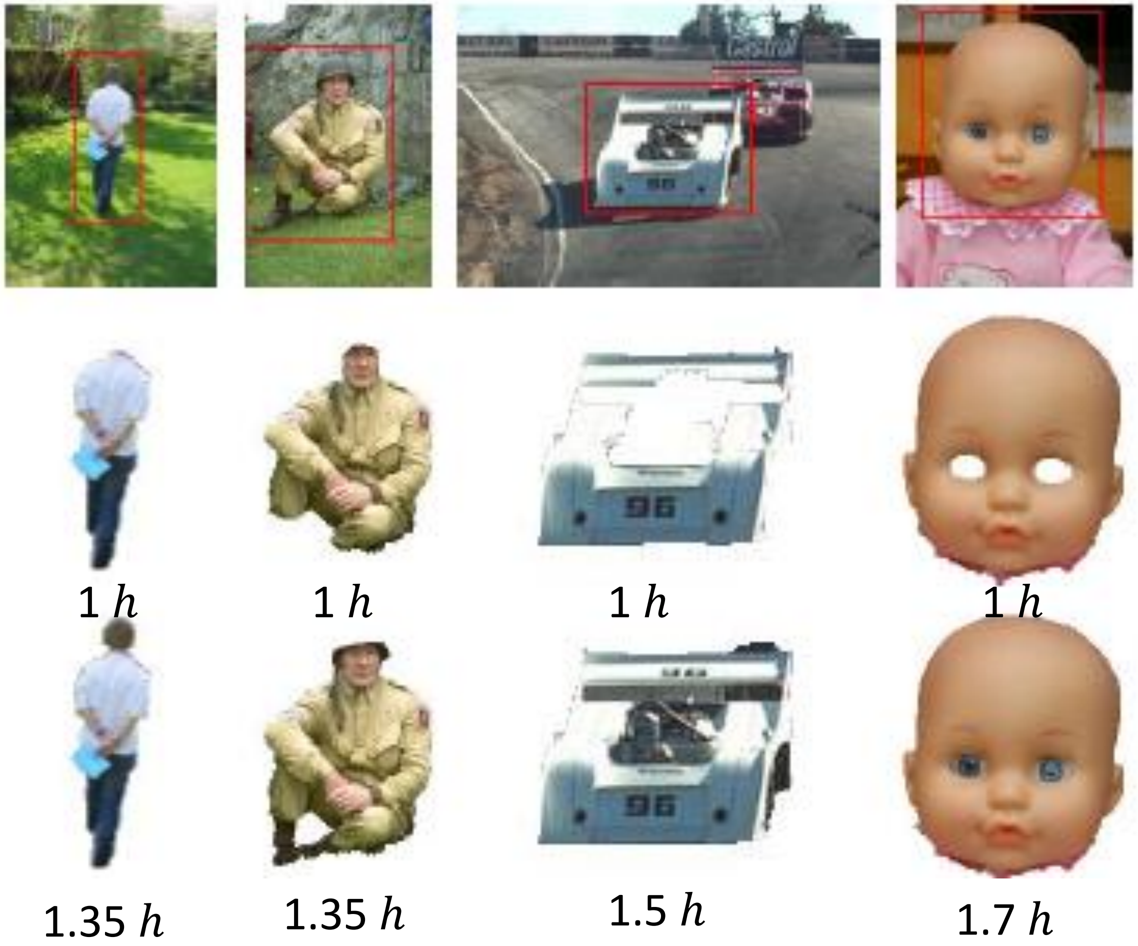


concave h_k



Each color either fore- or background

Adapting the weighting



How to optimize it?

$$\boldsymbol{x}^* = \underset{\boldsymbol{x}}{argmin}\, E'(\boldsymbol{x}) = \underset{\boldsymbol{x}}{argmin}\, [\, h(n^F) + \sum_k h_k(n_k^F) + \sum_{i,j \in N_4} |x_i - x_j| \,]$$

$$h(n^F) = \; g(n) + g(n - n^F)$$

$$h_k(n_k^F) = \; -g(n_k^F) - g(n_k - n_k^F)$$

with $g(z) = z \log z$

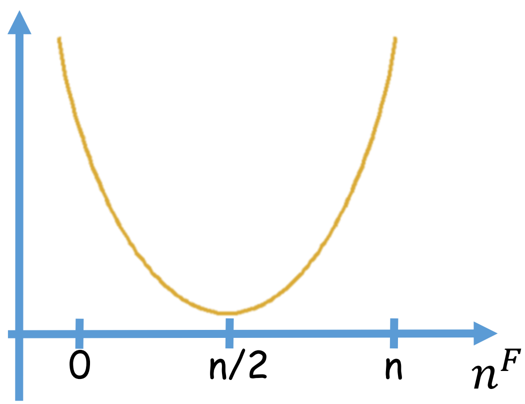
$n^F = \sum_i x_i$ number of foreground pixel

$n_k^F = \sum_i x_i$ number of foreground pixel in bin k

$n_k =$ number of pixels in bin k (independent of $\boldsymbol{x}$)

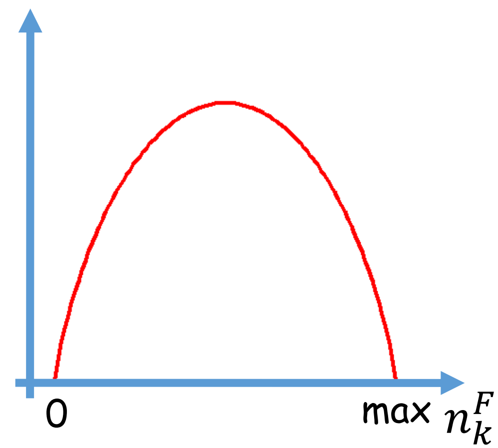
$n =$ number of pixels in image (independent of $\boldsymbol{x}$)

convex h



Prefers “equal area” segmentation

concave h_k



Each color either fore- or background

Question

$$\min_x E(x) = \min_x [\ E_1(x) + \theta^T x \ + E_2(x) - \theta^T x\ ]$$

$$? \ \min_x [E_1(x) + \theta^T x] + \min_x [E_2(x) - \theta^T x]$$

<u>Question:</u>

1) Is ? a $\leq$
2) Is ? a $<$
3) Is ? a $\geq$
4) Is ? a $>$
5) Is ? a $=$

Dual Decomposition

Hard to optimize

Possible to optimize

Possible to optimize

$$\min_x E(x) = \min_x [\ E_1(x) + \theta^T x \ + E_2(x) - \theta^T x\ ]$$

$$\geq \min_{x_1} [E_1(x_1) + \theta^T x_1] + \min_{x_2} [E_2(x_2) - \theta^T x_2] =: L(\theta)$$

"Lower bound"

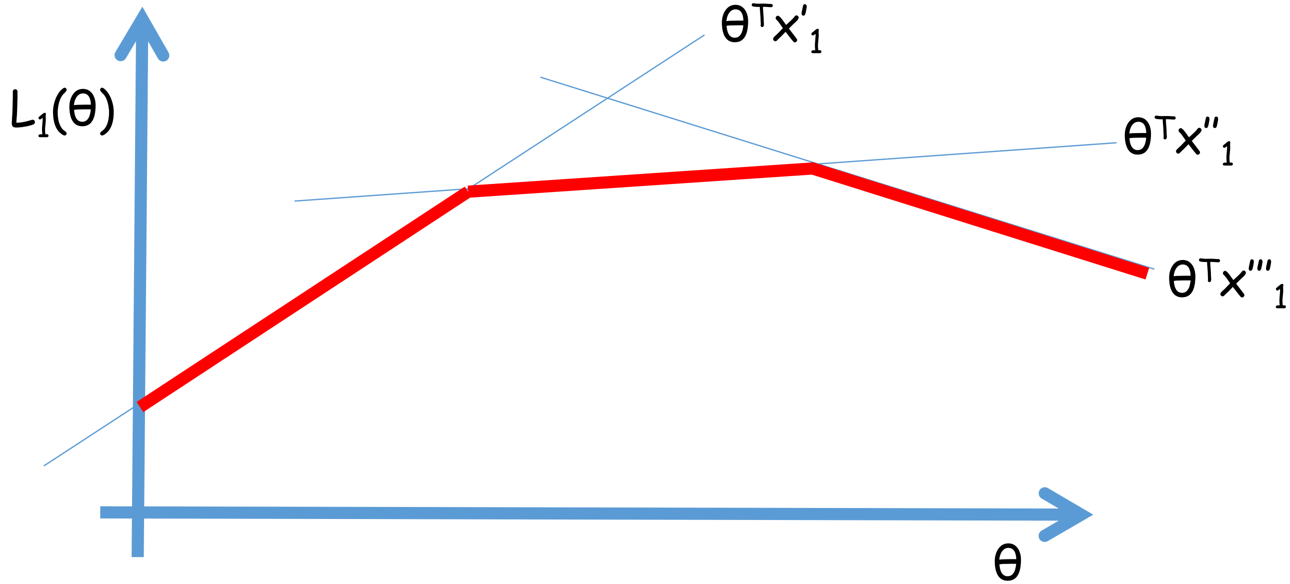
- θ is called the dual vector (same size as x)
- **Goal:** $\max_\theta L(\theta) \leq \min_x E(x)$
- Properties:
  - $L(\theta)$ is concave (optimal bound can be found)
  - If $x_1 = x_2$ then problem solved (no guarantee that this will happen)
- Dual Decomposition is sometimes also called Problem Decomposition

Why is the lower bound a concave function?

$$L(\theta) = \underbrace{\min_{x_1} [E_1(x_1) + \theta^T x_1]}_{L_1(\theta)} + \underbrace{\min_{x_2} [E_2(x_2) - \theta^T x_2]}_{L_2(\theta)} \qquad L(\theta) : R^n \text{ -> } R$$

<u>1D illustration:</u>

(note θ is high-dimensional)



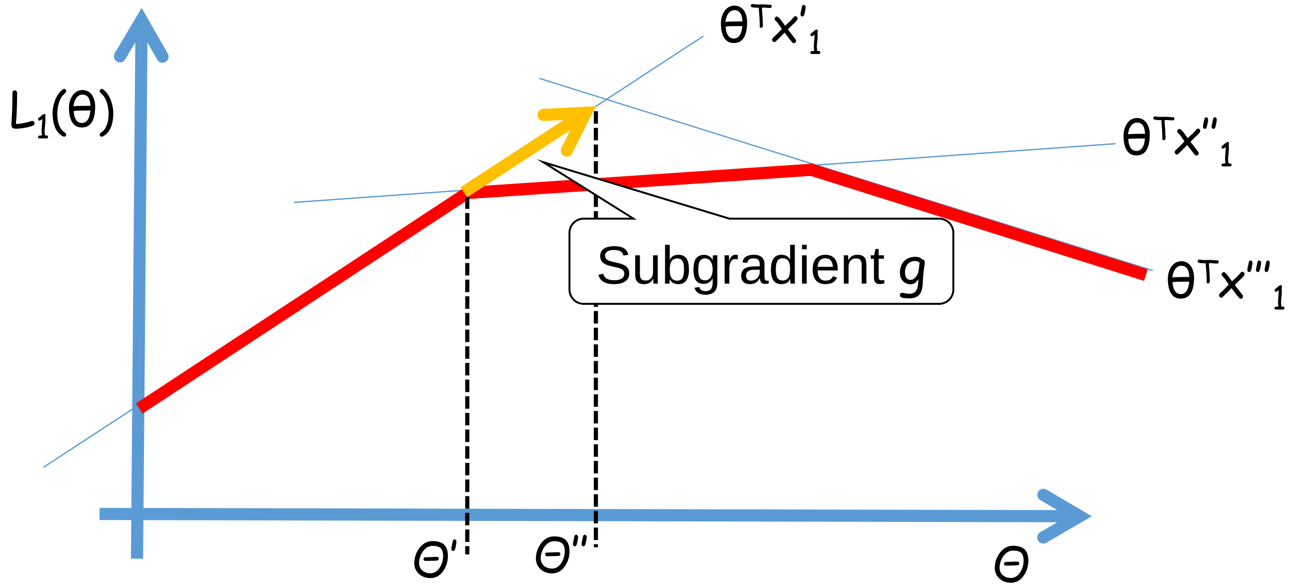
$L(\theta)$ concave since a sum of two concave functions: $L_1(\theta)$, $L_2(\theta)$

How to maximize the lower bound?

$$L(\theta) = \underbrace{\min_{x_1} [E_1(x_1) + \theta^T x_1]}_{L_1(\theta)} + \underbrace{\min_{x_2} [E_2(x_2) - \theta^T x_2]}_{L_2(\theta)} \qquad L(\theta) : R^n \to R$$

1D illustration:

(since θ is high dimensional)



$$\Theta'' = \Theta' + \lambda\, g = \Theta' + \lambda\, x'_1$$

(for $L_1(\theta)$)

$$\boxed{\Theta'' = \Theta' + \lambda\,(x_1 - x_2) \quad \text{(for } L(\theta)\text{)}}$$

How to maximize the lower bound?

$$L(\theta) = \min_{x_1} [E_1(x_1) + \theta^T x_1] + \min_{x_2} [E_2(x_2) - \theta^T x_2]$$

Iterative Optimization:

"Master"

$$\theta = \theta + \lambda(x_1^* - x_2^*)$$

subgradient

x_1^* θ θ x_2^*

Subproblem 1

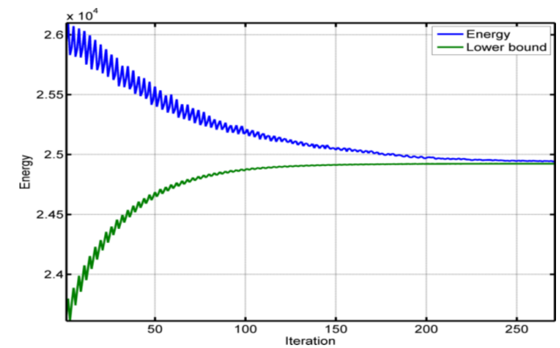
$$x_1^* = \min_{x_1} [E_1(x_1) + \theta^T x_1]$$

Subproblem 2

$$x_2^* = \min_{x_2} [E_2(x_2) - \theta^T x_2]$$

"Slaves"

- Guaranteed to converge to optimal bound $L(\theta)$
- Choose step-width λ correctly ([Bertsekas '95])
- Pick solution $\mathbf{x}$ as the best of $\mathbf{x_1}$ or $\mathbf{x_2}$
- E and L can in- and decrease during optimization



Illustrating the optimization

Dual Decomposition - Analysis

push x_{1p} towards 1

push x_{2p} towards 0

$$L(\theta) = \min [E_1(x_1) + \theta^T x_1] + \min [E_2(x_2) - \theta^T x_2]$$

Update step: $\Theta'' = \Theta' + \Lambda\, (x_1^* - x_2^*)$

Consider pixel p:

Case1: $x_{1p}^* = x_{2p}^*$ then $\Theta'' = \Theta'$

Case2: $x_{1p}^* = 1$ $x_{2p}^* = 0$ then $\Theta'' = \Theta' + \Lambda$

Case3: $x_{1p}^* = 0$ $x_{2p}^* = 1$ then $\Theta'' = \Theta' - \Lambda$

Dual Decomposition – Our Model

$$\boldsymbol{x}^* = \underset{\boldsymbol{x}}{argmin} \left[\underbrace{h(n^F) + \sum_i \Theta_i x_i}_{E_1(x)} + \underbrace{\sum_k h_k(n_k^F) + \sum_{i,j \in N_4} |x_i - x_j| - \sum_i \Theta_i x_i}_{E_2(x)} \right]$$

$h(n^F) = \; g(n) + g(n - n^F)$

$h_k(n_k^F) = \; -g(n_k^F) - g(n_k - n_k^F)$

with $g(z) = z \log z$

$n^F = \sum_i x_i$ number of foreground pixel

$n_k^F = \sum_i x_i$ number of foreground pixel in bin k

$n_k =$ number of pixels in bin k (independent of $\boldsymbol{x}$)

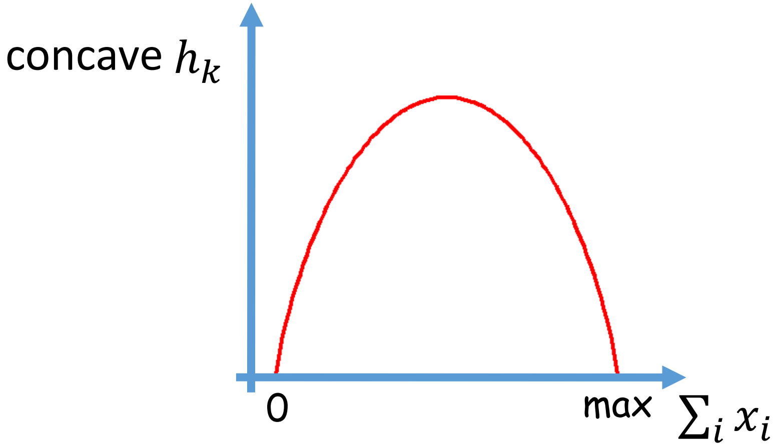
Solving the Sub-problems:

- $E_1(x)$ can be computed globally optimal (see article)
- $E_2(x)$ can be computed globally optimal (see next)

Special high-order functions

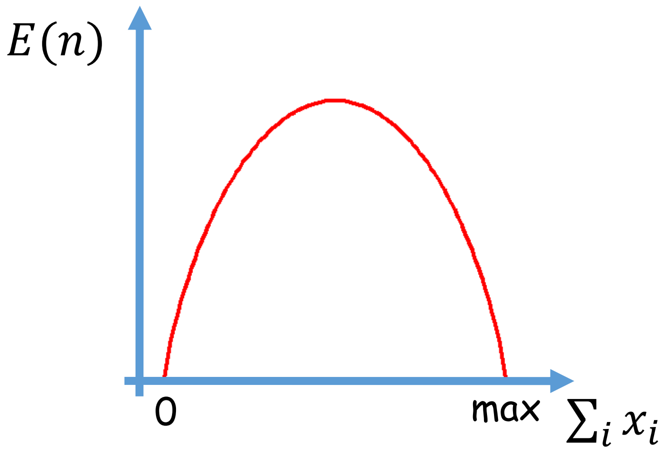
$n_k^F = \sum_i x_i$ number of foreground pixel in bin k

$$E_2(\boldsymbol{x}) = \sum_k h_k(n_k^F) + \sum_{i,j \in N_4} |x_i - x_j| - \sum_i \Theta_i x_i$$

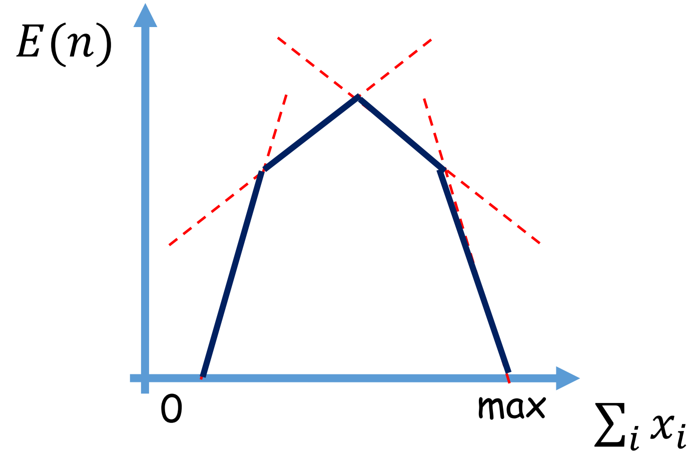


Goal: convert this higher-order function into a pairwise function using P^n Potts Model

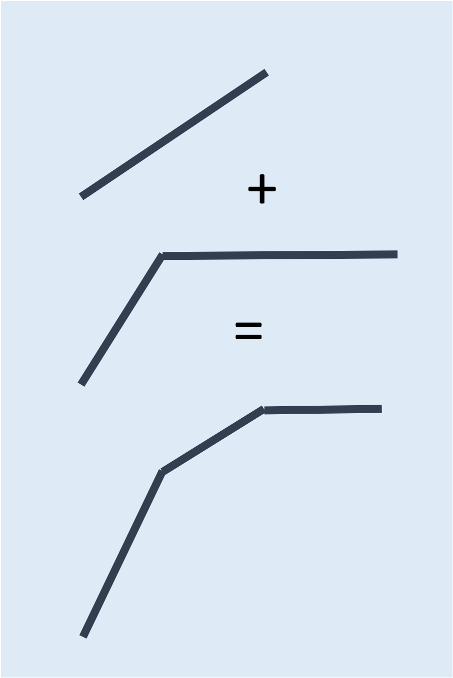
Reminder: P^n Potts Model



Arbitrary concave function



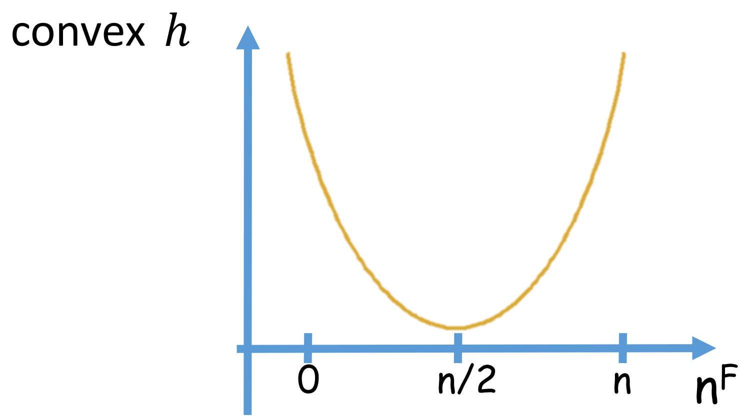
Approximate with lower envelop of linear functions



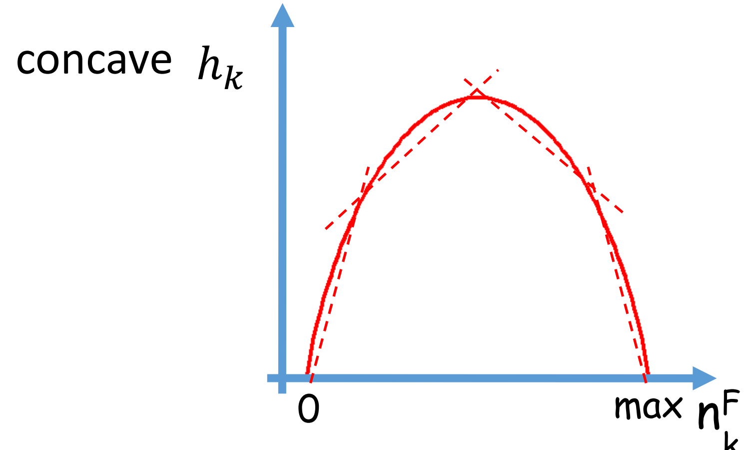
Roadmap this lecture

- GrabCut: Interactive Image Segmentation from a Bounding Box
- Joint optimization of segmentation and appearance models
  [Vicente, Kolmogorov, Rother, ICCV 2009]
- A state-of-the-art approach
  [GrabCut in OneCut, Tang, Gorelick, Veksler, Boykov; ICCV 2013]
- Gaussian Markov Random Fields

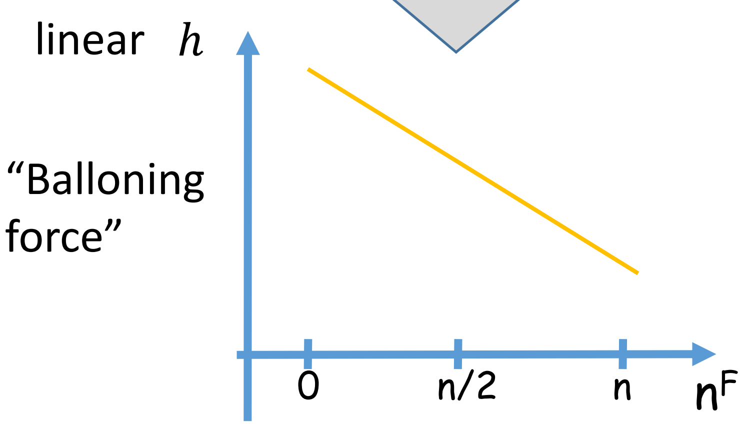
Remove the parts which are hard to optimize



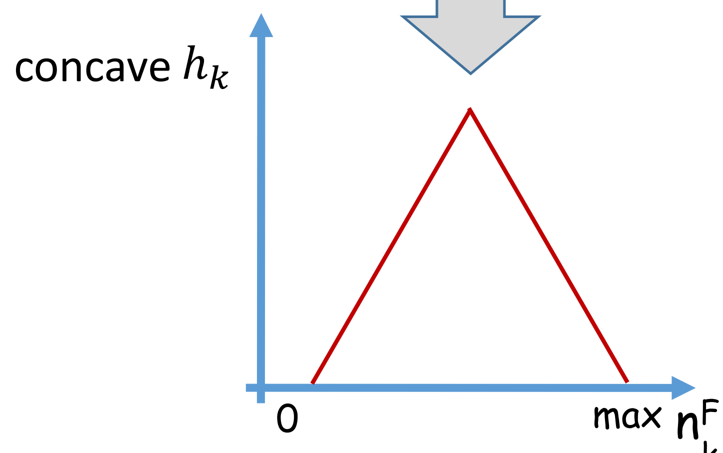
Prefers “equal area” segmentation



Each color either fore- or background



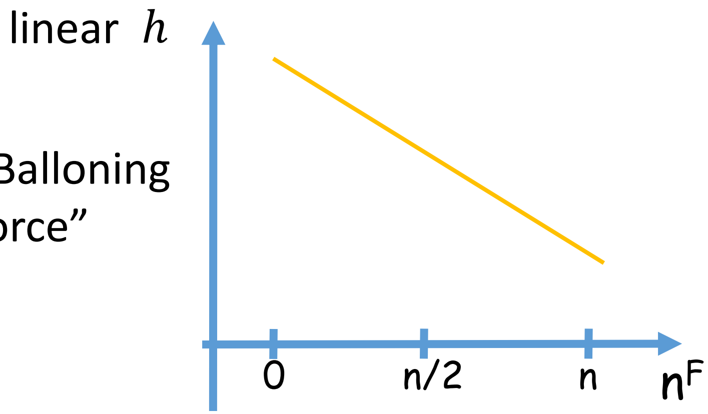
Prefers segmentation with all pixels assigned to foreground



Each color either fore- or background

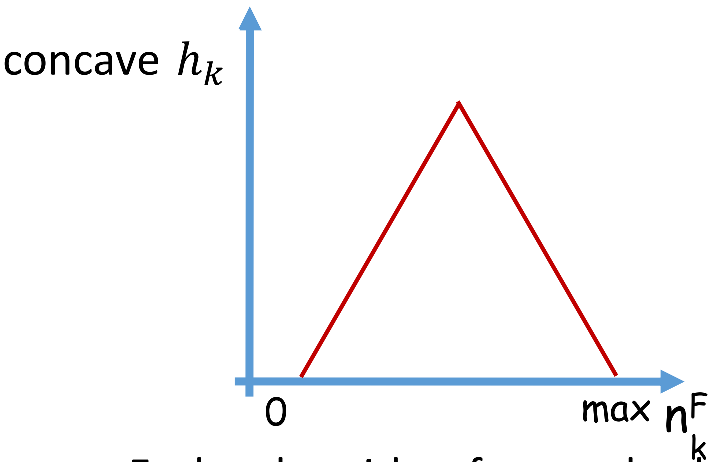
[GrabCut in One Cut, Tang, Gorelick, Veksler, Boykov, ICCV ’13]

OneCut Energy



"Balloning force"

Prefers segmentation with all pixels assigned to foreground

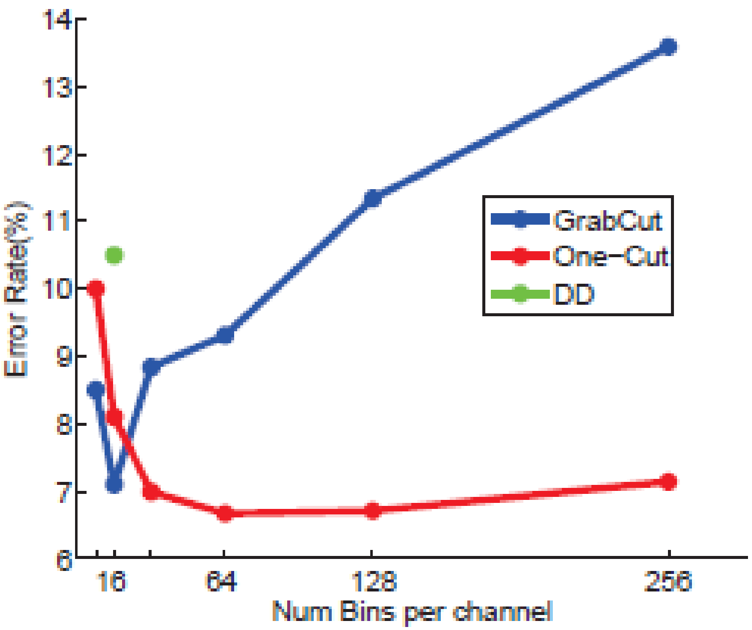


Each color either fore- or background

$$\boldsymbol{x}^* = \underset{\boldsymbol{x}}{argmin}\, E'(\boldsymbol{x}) = \underset{\boldsymbol{x}}{argmin}\, [\textstyle\sum_i \lambda\,(1 - x_i) + \sum_k h_k\left(n_k^F\right) + \sum_{i,j\in N_4} \left|x_i - x_j\right|]$$

The global optimum can be computed efficiently (as seen above)

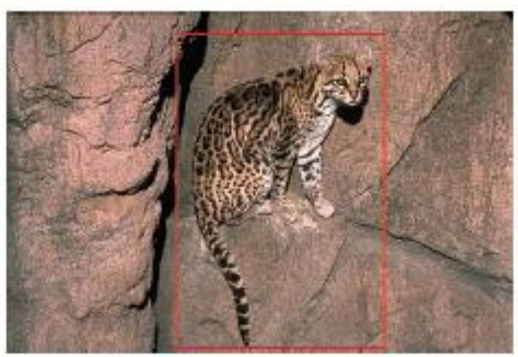
Comparison to previous approaches



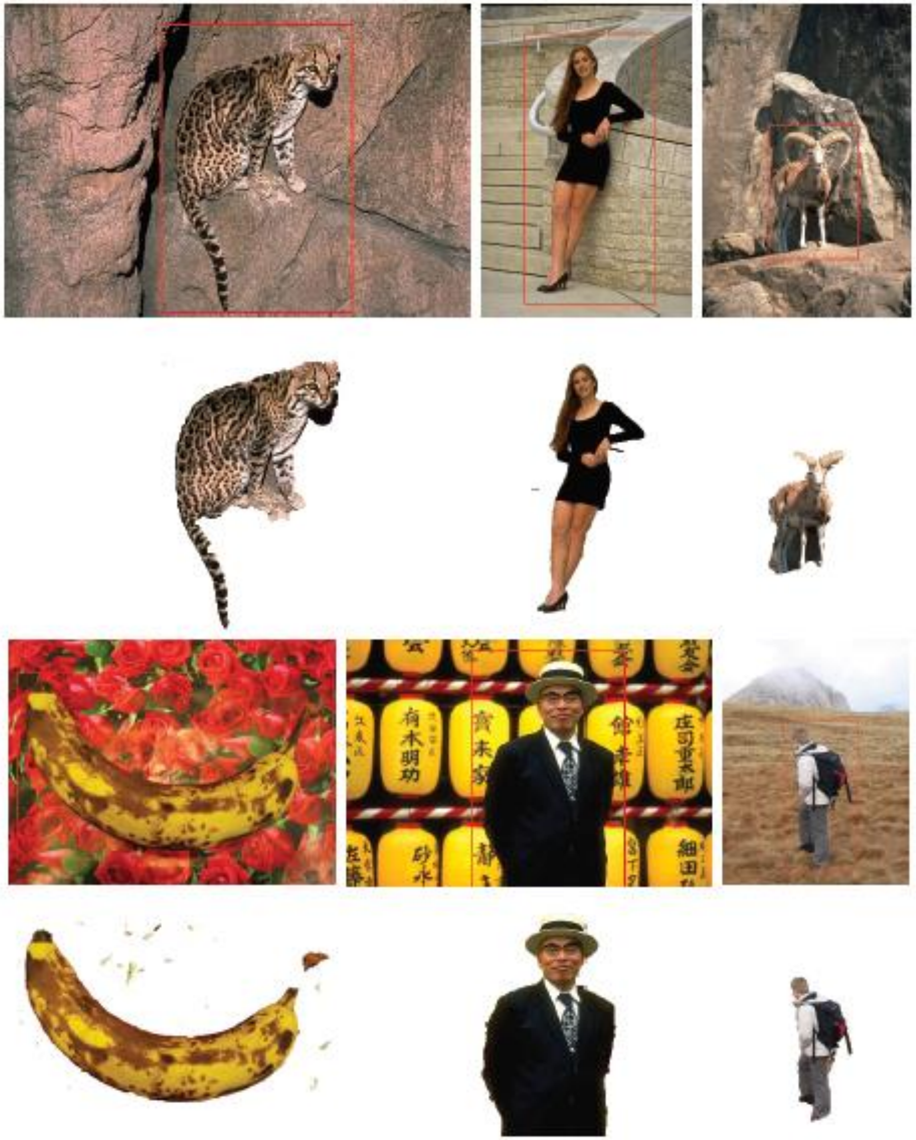
	Error rate	Mean runtime
GrabCut (8^3 bins)	8.54%	2.48 s
GrabCut (16^3 bins)	7.1%[2]	1.78 s
GrabCut (32^3 bins)	8.78%	1.63s
GrabCut (64^3 bins)	9.31%	1.64s
GrabCut (128^3 bins)	11.34%	1.45s
GrabCut (256^3 bins)	13.59%	1.46s
DD (16^3 bins)	10.5%	576 s
One-Cut (8^3 bins)	9.98%	18 s
One-Cut (16^3 bins)	8.1%	5.8 s
One-Cut (32^3 bins)	6.99%	2.4 s
One-Cut (64^3 bins)	6.67%	1.3 s
One-Cut (128^3 bins)	6.71%	0.8 s
One-Cut (256^3 bins)	7.14%	0.8 s

GrabCut = Block Coordinate Descent
DD = Dual decomposition
(as discussed above)

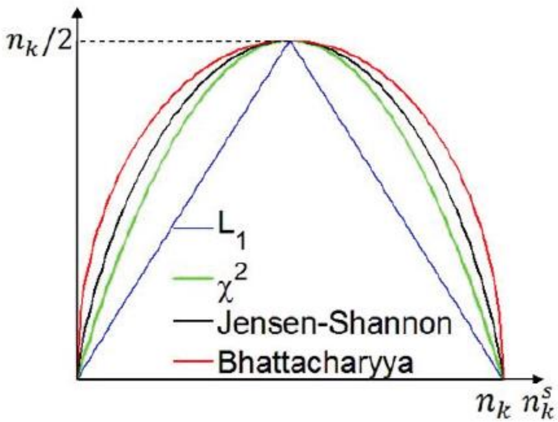
More bins gives better separation in camouflage cases:

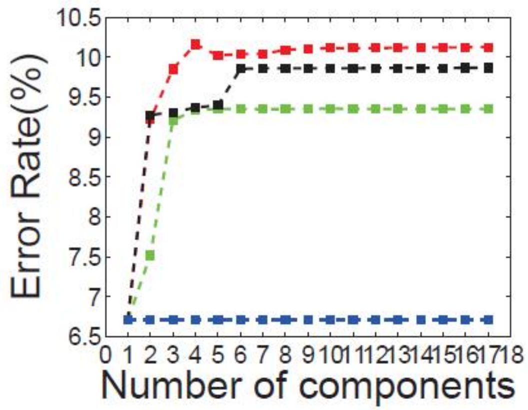


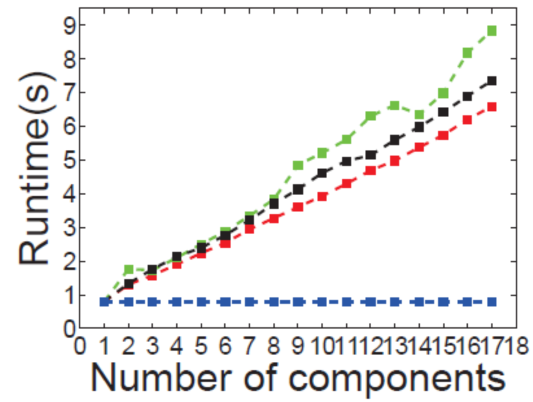
Results

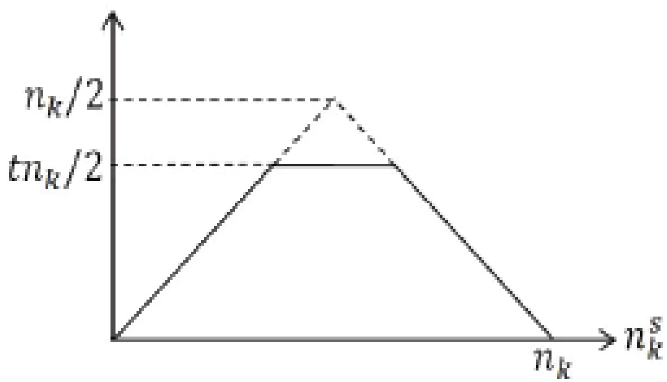


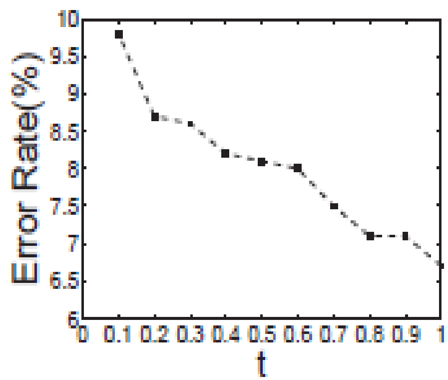
Different Terms h_k











Conclusion: the simple L_1 term is best in terms of accuracy and speed

Roadmap this lecture

- GrabCut: Interactive Image Segmentation from a Bounding Box
- Joint optimization of segmentation and appearance models
  [Vicente, Kolmogorov, Rother, ICCV 2009]
- A state-of-the-art approach
  [GrabCut in OneCut, Tang, Gorelick, Veksler, Boykov; ICCV 2013]
- Gaussian Markov Random Fields

The following part is not relevant for the exam

Gaussian MRF (GMRF)

$P(x) = 1/\sqrt{\det(2\pi\Sigma)} \exp\{-1/2\, (x-\mu)^T \Sigma^{-1} (x-\mu)\}$

$x \in \mathbb{R}^n$, $\Sigma^{-1} \in \mathbb{R}^{nxn}$ i.e. one multi-dimensional Gaussian

where Σ^{-1} is positive-definite

Re-write as Energy (Gibbs distribution):

$P(x) = 1/f \exp\{-E(x)\}$ with $E(x) = \frac{1}{2} x^T A x - x^T b + \text{constant}$

It is $x^* = \underset{x}{\operatorname{argmin}} E(x) = \underset{x}{\operatorname{argmax}} P(x)$

$$(x-b)^T A (x-b) = (x-b)^T (Ax - Ab)$$
$$= (x^T - b^T)(Ax - Ab)$$
$$= x^T A x - x^T A b - b^T A x + b^T A b$$
$$= \frac{1}{2} x^T A' x - x^T b' + \text{constant}$$

Reminder

In linear algebra, a symmetric *n* × *n* real matrix M is said to be **positive definite** if the scalar $z^{\mathrm{T}} M z$ is positive for every non-zero column vector z of n real numbers. Here z^{T} denotes the transpose of z.[1]

- The real symmetric matrix

$$M = \begin{bmatrix} 2 & -1 & 0 \\ -1 & 2 & -1 \\ 0 & -1 & 2 \end{bmatrix}$$

is positive definite since for any non-zero column vector *z* with entries *a*, *b* and *c*, we have

$$\begin{aligned} z^{\mathrm{T}} M z = (z^{\mathrm{T}} M) z &= \begin{bmatrix} (2a-b) & (-a+2b-c) & (-b+2c) \end{bmatrix} \begin{bmatrix} a \\ b \\ c \end{bmatrix} \\ &= 2a^2 - 2ab + 2b^2 - 2bc + 2c^2 \\ &= a^2 + (a-b)^2 + (b-c)^2 + c^2 \end{aligned}$$

This result is a sum of squares, and therefore non-negative; and is zero only if *a* = *b* = *c* = 0, that is, when *z* is zero.

Gaussian MRF

Reminder

$$\frac{\partial y}{\partial x} = \left(\frac{\partial y}{\partial x_1}, \dots, \frac{\partial y}{\partial x_n}\right) \in \mathbb{R}^n, \quad y \in \mathbb{R}, \quad x \in \mathbb{R}^n$$

vector

$$\frac{\partial y}{\partial x} = \left(\frac{\partial y}{\partial x_1} \Big| \dots \Big| \frac{\partial y}{\partial x_n}\right) \in \mathbb{R}^{n \times n}, \quad y \in \mathbb{R}^n, \quad x \in \mathbb{R}^n$$

$$\frac{\partial x^T b}{\partial x} = \left(\frac{\partial b_1 x_1 + \dots b_n x_n}{\partial x_1}, \dots, \frac{\partial b_1 x_1 + \dots + b_n x_n}{\partial x_n}\right) = (b_1, \dots, b_n) = b$$

$x, b \in \mathbb{R}^n$

$$\frac{\partial Bx}{\partial x} = \left(\frac{\partial Bx}{\partial x_1} \Big| \dots \Big| \frac{\partial Bx}{\partial x_n}\right) = (b_1 | \dots | b_n) = B$$

b_i is the ith column of B

$x \in \mathbb{R}^n, \; B \in \mathbb{R}^{n \times n}$

Let us show $\frac{\partial x^T A x}{\partial x} = \left(\frac{\partial x^T A x}{\partial x_1}, \dots, \frac{\partial x^T A x}{\partial x_n}\right) = 2Ax$

we have $A = A^T$ (symmetric), $x \in \mathbb{R}^n$, $A \in \mathbb{R}^{n \times n}$

we use $A = B^T B$ (valid for any symmetric matrix)

Proof: $\frac{\partial x^T A x}{\partial x} = \frac{\partial x^T B^T B x}{\partial x}$ (use product rule)

$$= \frac{\partial x^T B^T}{\partial x} B x + \left(x^T B^T \frac{\partial B x}{\partial x}\right)^T \leftarrow \text{to match vectors}$$

$$= B^T B x + (x^T B^T B)^T = Ax + (x^T A)^T = Ax + A^T x$$

$$\overset{A^T = A}{=} Ax + Ax = 2Ax$$

Gaussian MRF

Let's use:

$$\frac{\partial x^T b}{\partial x} = \left(\frac{\partial b_1 x_1 + \dots b_n x_n}{\partial x_1}, \dots, \frac{\partial b_1 x_1 + \dots + b_n x_n}{\partial x_n}\right) = (b_1 \dots, b_n) = b$$

$x, b \in \mathbb{R}^n$

$$\frac{\partial x^T A x}{\partial x} = \left(\frac{\partial x^T A x}{\partial x_1}, \dots, \frac{\partial x^T A x}{\partial x_n}\right) = 2Ax$$

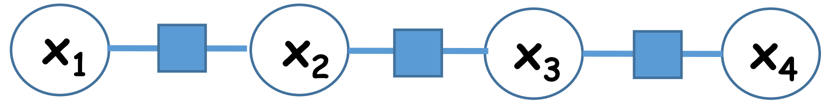
Then it is: $E(x) = \frac{1}{2}\boldsymbol{x}^{\boldsymbol{T}}\boldsymbol{A}\boldsymbol{x} - \boldsymbol{x}^{\boldsymbol{T}}\boldsymbol{b} + constant$

$$\frac{\partial E(x)}{\partial x} = \left(\frac{\partial E(x)}{\partial x_1}, \dots, \frac{\partial E(x)}{\partial x_n}\right) = \boldsymbol{A}\boldsymbol{x} - \boldsymbol{b} \overset{!}{=} 0$$

$\boldsymbol{x}^* = \boldsymbol{A}^{-1}\boldsymbol{b}$ (closed-from solution, or other solvers)

Example: GMRF

We would like to model: $E(x) = \sum_{i,j \in N} (x_i - x_j)^2$



$x_i \in \mathbb{R}$

$$\begin{bmatrix} \mathbf{1} & \mathbf{-1} & & \\ -1 & 2 & -1 & \\ & -1 & 2 & -1 \\ & & -1 & 1 \end{bmatrix} =: \tfrac{1}{2} A \quad \text{for } E(x) = \tfrac{1}{2} x^T A x - x^T b$$

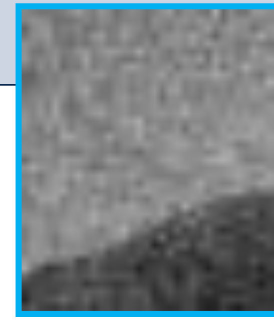
with $b = 0$

$$E(x) = (x_1, x_2, x_3, x_4) \begin{bmatrix} \mathbf{1} & & \\ -1 & 1 & \\ & -1 & 1 \\ & & -1 \end{bmatrix} \begin{bmatrix} \mathbf{1} & \mathbf{-1} & & \\ & 1 & -1 & \\ & & 1 & -1 \end{bmatrix} \begin{bmatrix} x_1 \\ x_2 \\ x_3 \\ x_4 \end{bmatrix}$$

$$(x_1 - x_2, x_2 - x_3, x_3 - x_4) \qquad (x_1 - x_2, x_2 - x_3, x_3 - x_4)^T$$

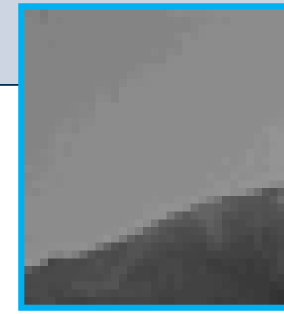
$$E(x) = \sum_{i,j} (x_i - x_j)^2$$

A simple de-noising model



Data $\mathbf{z}$

(pixel independent Gaussian noise)



Label $\mathbf{x}$

<u>Generative model for a noisy image:</u>

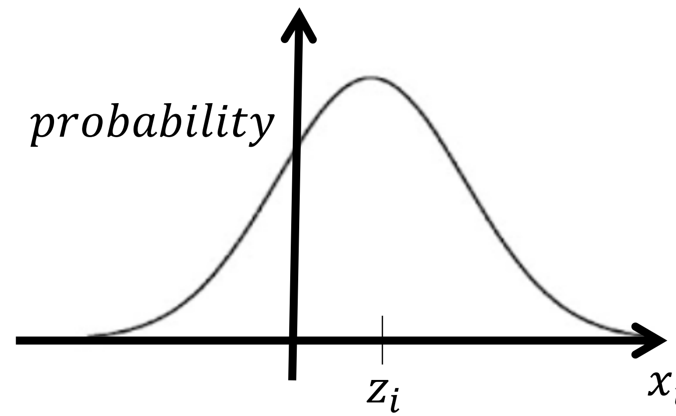
$$P(x,z) = P(z|x)\,P(x)$$

$$x^* = \operatorname*{argmax}_x P(z|x)\,P(x) \quad \text{(for a given } z\text{)}$$

$$x^* = \operatorname*{argmin}_x E_u(x,z) + E_p(x)$$

<u>Likelihood:</u>

$$P(z|x) = 1/\sigma\sqrt{2\pi}\ \prod_i \exp\{-1/(2\sigma^2)\,(x_i-z_i)^2\}$$

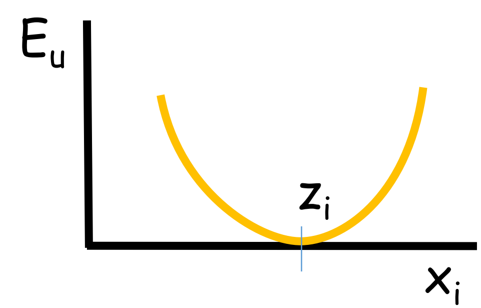


(f is a factor, which can be ignored)

$$-\log P(z|x) = E_u(x) = f \sum_i (x_i-z_i)^2$$

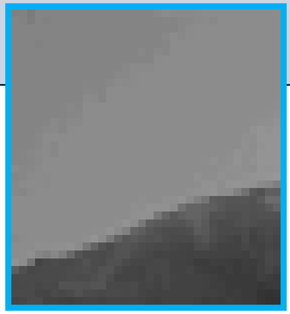
$$= f\,(x-z)^T I\,(x-z) = f\,(x^T I\,x - 2x^T z + z^T z)$$

(this is a quadratic form)



A simple de-noising model

Data **z**
(pixel independent Gaussian noise)



Label **x**

Generative model for a noisy image:

$P(x,z) = P(z|x)\,P(x)$

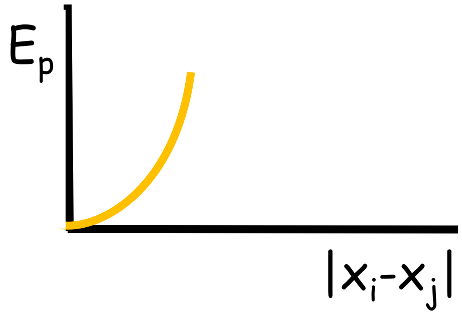
$x^* = \underset{x}{\operatorname{argmax}}\ P(z|x)\,P(x)$ (for a given z)

$x^* = \underset{x}{\operatorname{argmin}}\ E_u(x,z) + E_p(x)$

Prior:

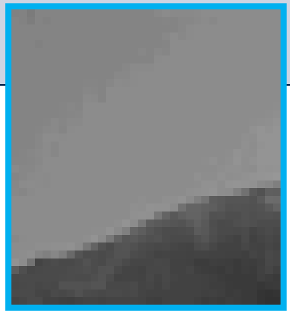
$-\log P(x) = f\,E_p(x) = f \sum_{i,j \in \mathcal{N}} (x_i - x_j)^2$ (f is a factor, which can be ignored)

$= f\,\frac{1}{2}\,x^T A x$ (as done above)



A simple de-noising system

Data $\mathbf{z}$
(pixel independent Gaussian noise)



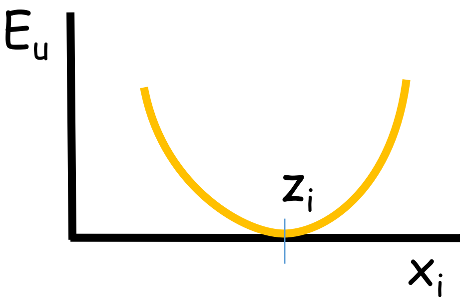
Label $\mathbf{x}$

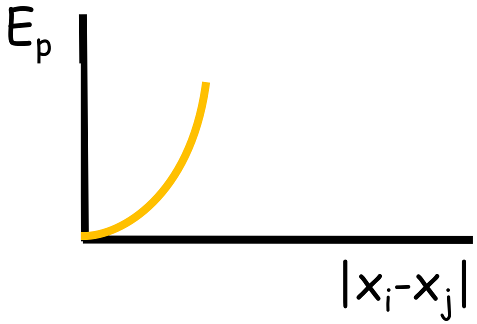
<u>Generative model for a noisy image:</u>

$$P(x,z) = P(z|x)\,P(x)$$

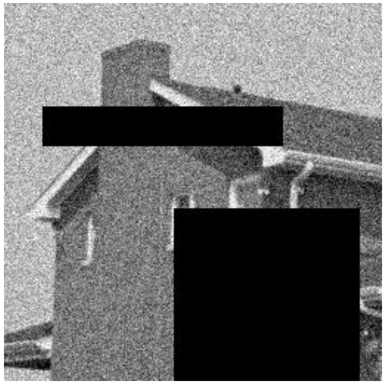
$$x^* = \underset{x}{\operatorname{argmax}}\; P(z|x)\,P(x) \quad \text{(for a given } z\text{)}$$

$$x^* = \underset{x}{\operatorname{argmin}}\; E_u(x) + E_p(x)$$

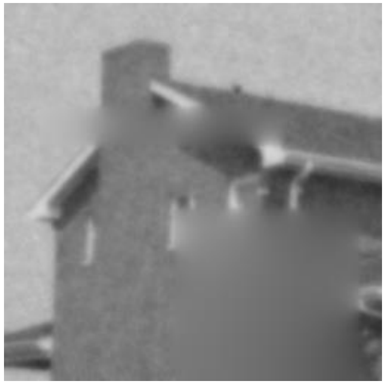




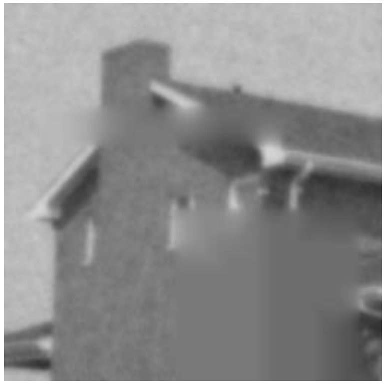
original



Input $\mathbf{z}$



continuous label $\mathbf{x}$
HBF [Szeliski '06]~ 15times faster

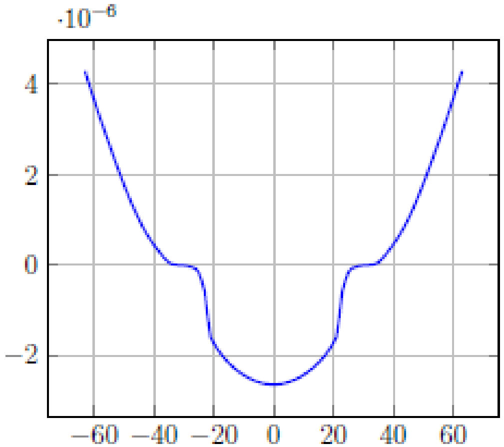


256 discrete labels $\mathbf{x}$
(TRW-S)

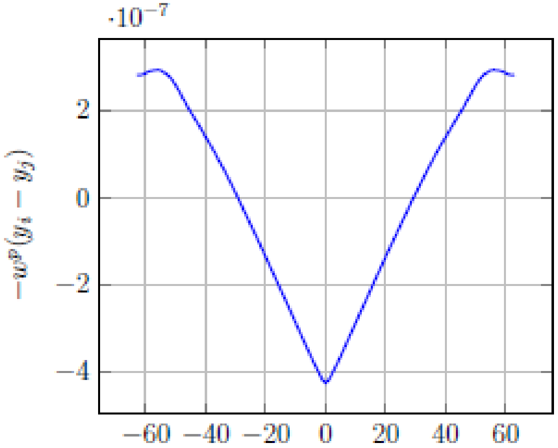
Practical view: Potentials are often not Gaussian

Ideal output Label **x**

Noisy input image **z**



Unary potential: $\mathbf{z_i - x_i}$



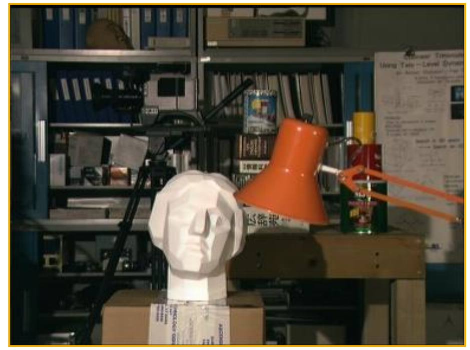
Pairwise potential: $\mathbf{x_i - x_j}$

Learning the potentials (Loss-minimizing parameter learning) gives non-Gaussian potentials

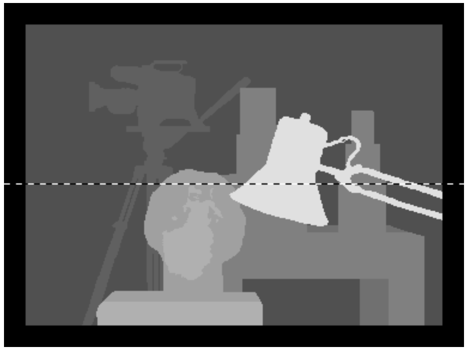
[Putting MAP back on the map, Pletscher et al. DAGM 2010]

Stereo Matching - Energy

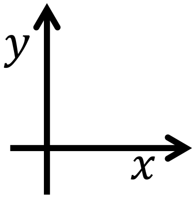
Left image



Right image



Ground truth depth

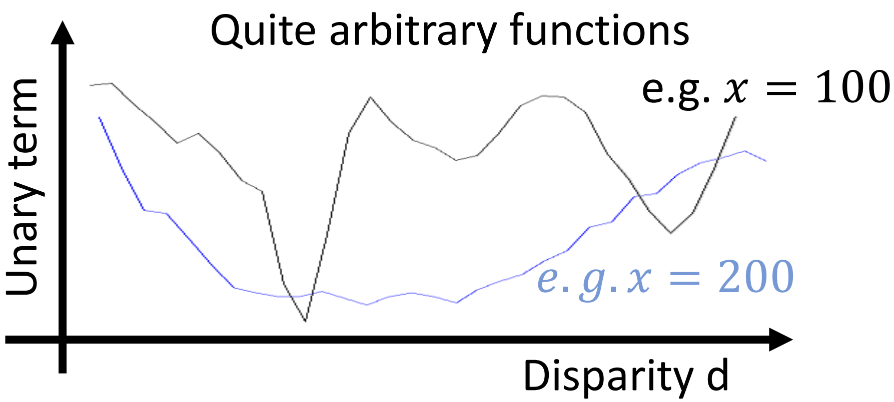


x-direction (black low cost) (e.g. $y = 100$)

<u>Unary terms</u> (many options)

Patch-Cost for a pixel i with disparity d_i:

$$\theta_i(d_i) = \sum_{j \in N_i} \left(I_j^l - I_{j-d_i}^r\right)^2$$

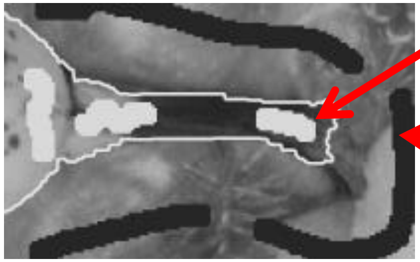


[Scharstein et al. IJCV '02]

Continuous-valued MRF for Segmentation

Energy: $E(x) = \sum_{i,j \in N_4} (w_{ij} \, |x_i - x_j|)^p \quad x_i \in \mathbb{R}$

where $w_{ij} = \exp\{-\beta ||z_i - z_j||\})$
(as before)



Optimize: $x^* = \operatorname{argmin}_x E(x)$

sb.t. $x_i=1$ for foreground; $x_i=0$ for background

Output $x^o = \text{round}(x^*)$ i.e. $x^o = 0$ if $x^* < 0.5$, 1 otherwise

For any $p \geq 1$ this can be solved globally optimal (but not guaranteed to have a unique solution). Here optimized with IRLS (iterated reweighted least squares)

[Singaraju, Grady et al. MRF book, ch. 8]

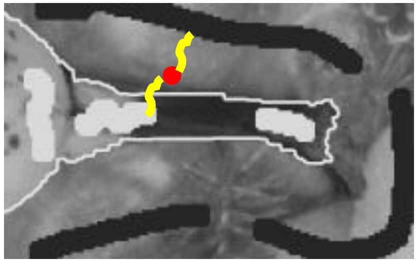
Continuous-valued MRF for Segmentation

Varying the p-norm: **Energy:** $E(x) = \sum_{i,j \in N_4} (w_{ij} \, |x_i - x_j|)^p$

- p=1, the rounded solution is the same as the solution to discrete problem: $x_i \in \{0,1\}$
  (i.e GraphCut [Boykov, Jolly, ICCV '01])

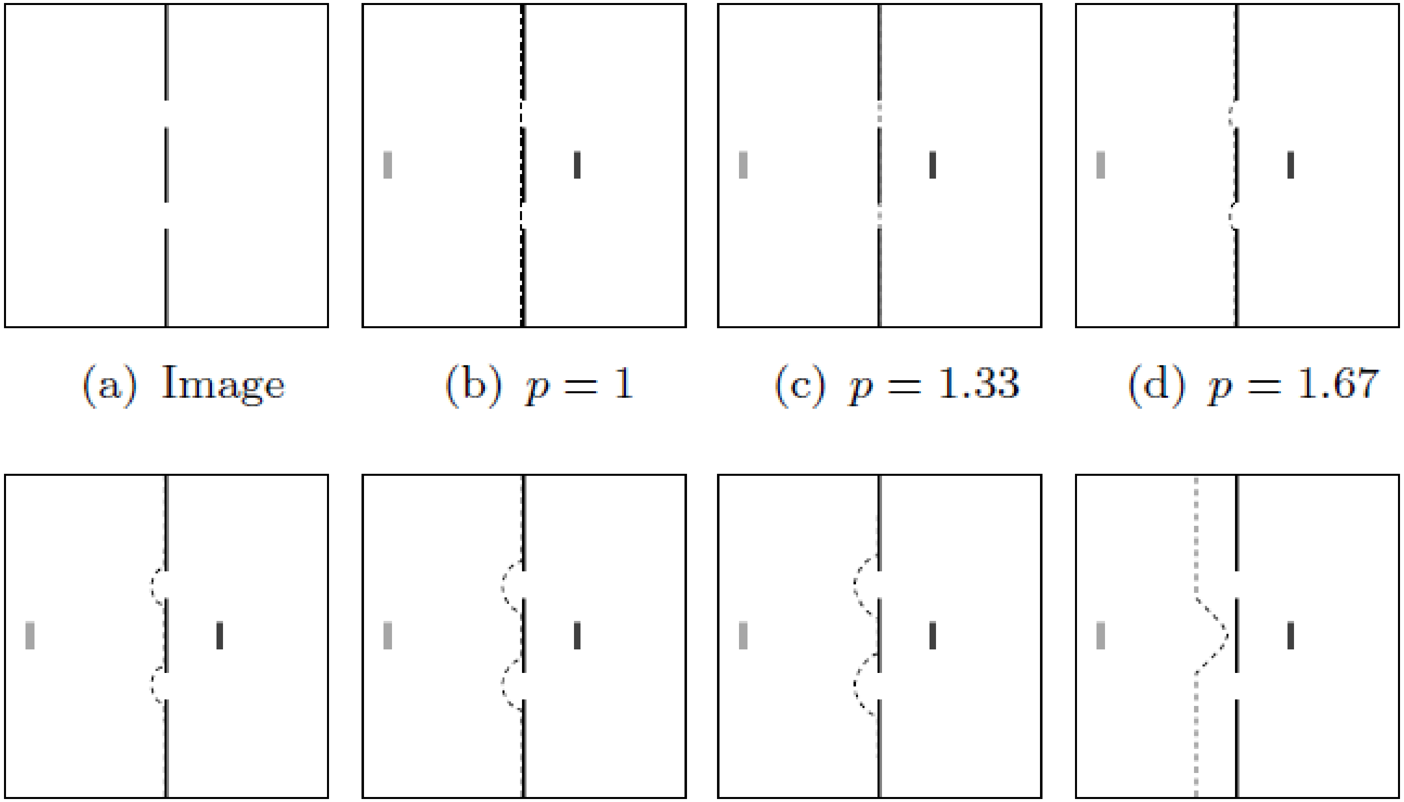
- p=2, Gaussian MRF
  (known as random walker solution [Grady PAMI '06])

- p -> ∞, solution becomes ambiguous: one solution is shortest geodesic path to a foreground or background brush [Bai et al. ICCV '07]



Proximity bias

Sensitivity of the segmentation to the placement of the brush strokes

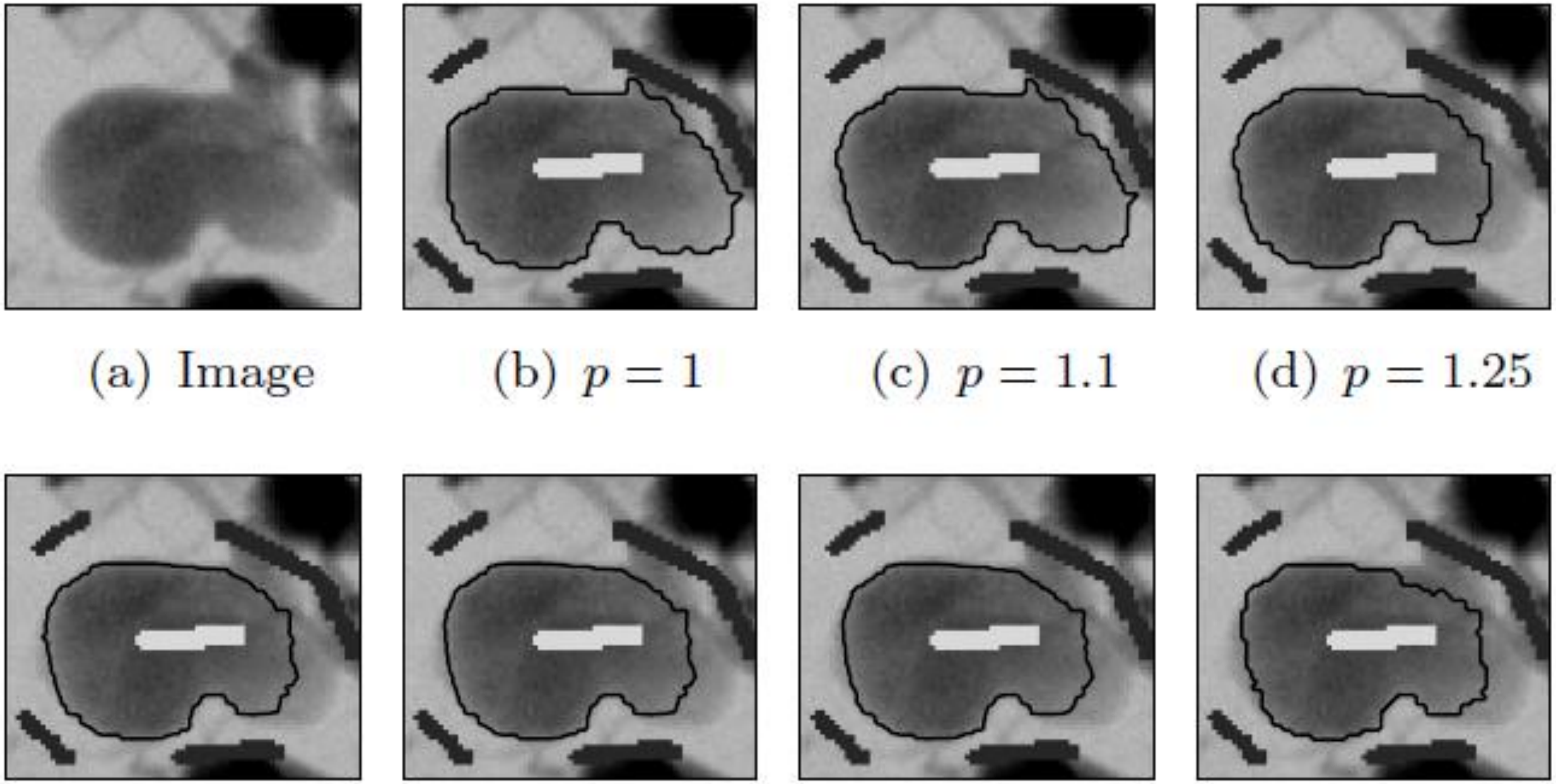


(a) Image (b) $p = 1$ (c) $p = 1.33$ (d) $p = 1.67$

(e) $p = 2$ (f) $p = 2.33$ (g) $p = 2.67$ (h) $p = \infty$

Proximity bias

Sensitivity of the segmentation to the placement of the brush strokes



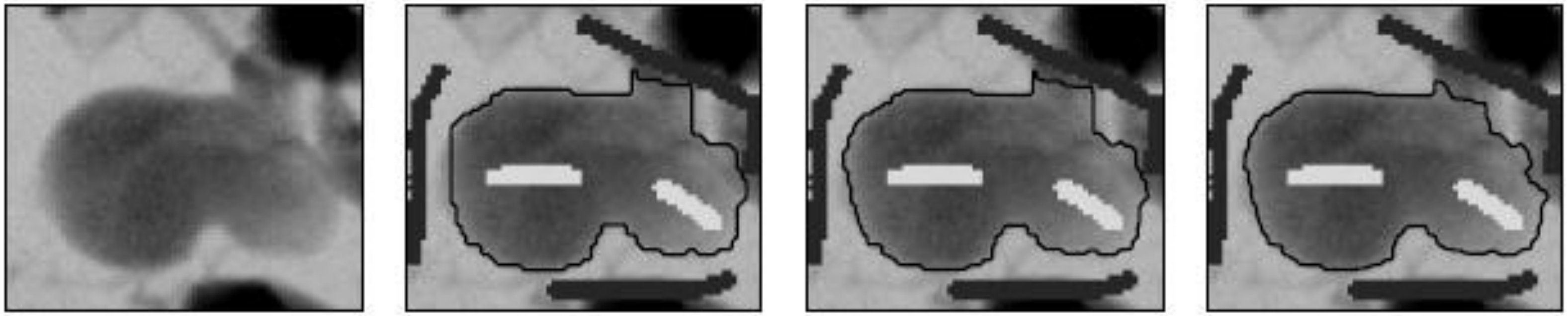
(a) Image (b) $p = 1$ (c) $p = 1.1$ (d) $p = 1.25$

(e) $p = 1.75$ (f) $p = 2$ (g) $p = 2.75$ (h) $p = \infty$

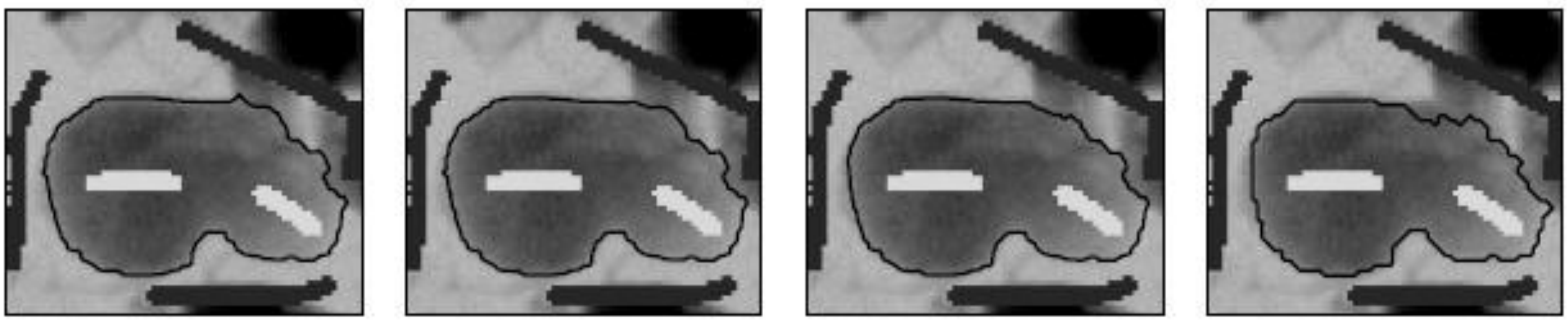
Small p may be better

Metrication (discretization) artefacts

"Blockiness" of the segmentation due to the underlying pixel grid



(a) image (b) $p = 1$ (c) $p = 1.25$ (d) $p = 1.75$

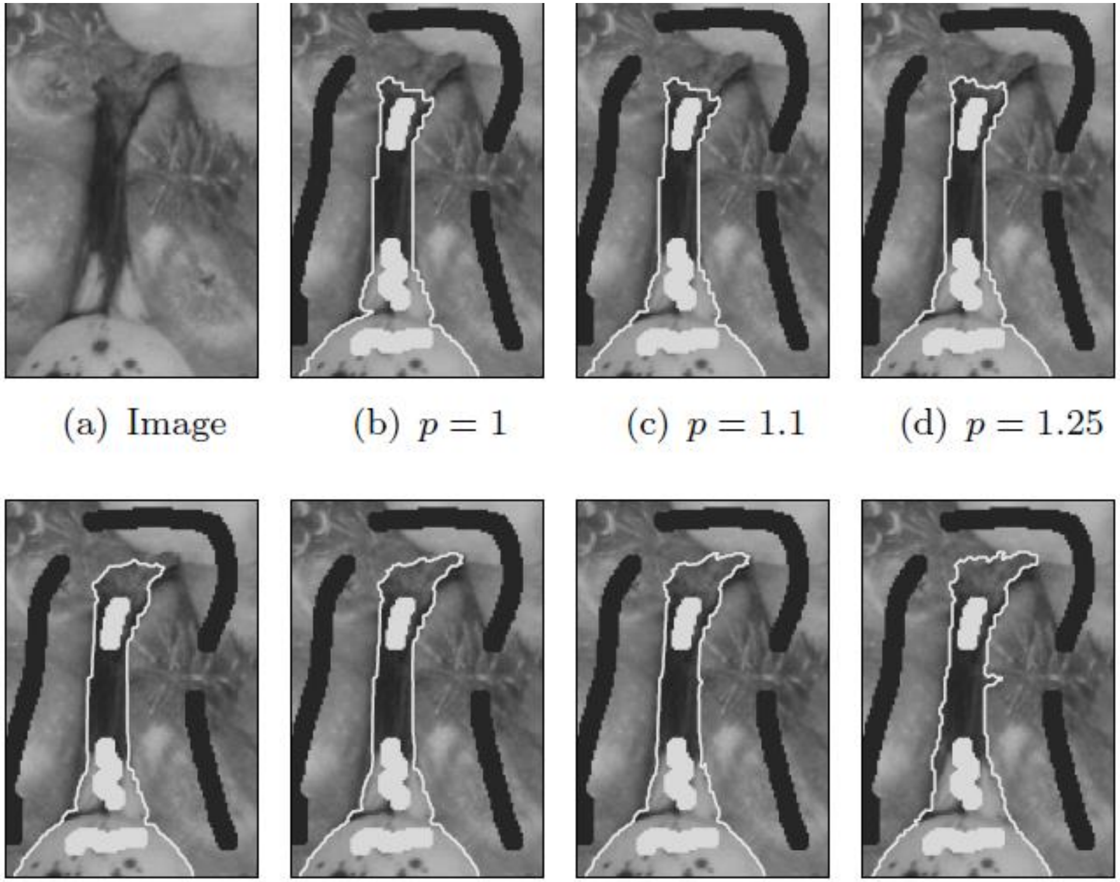


(e) $p = 2$ (f) $p = 2.25$ (g) $p = 2.75$ (h) $p = \infty$

Larger p is better

Shrinking Bias

Shortening of the segmentation due to regularization



(a) Image (b) $p = 1$ (c) $p = 1.1$ (d) $p = 1.25$

(e) $p = 1.75$ (f) $p = 2$ (g) $p = 2.75$ (h) $p = \infty$

a practical study showed that p=1.4 is empirically a good value

Small p may be better

Gaussian MRF for Matting

Input z | User constraints | Output α | Output F (true foreground color; background color is not shown) | Composition on new background

Input: z
Output: α , F,B
Constraint: $z_i = \alpha_i F_i + (1 - \alpha_i) B_i$
Each pixel gives 3 constraints and has 7 unknowns

$E(\alpha) = \alpha^T L \alpha \qquad \alpha^* = \underset{\alpha}{\operatorname{argmin}} E(\alpha)$ sb.t brush strokes

L called Matting Laplacian and has the form: $L = \begin{pmatrix} & & 0 \\ & \neq 0 & \\ 0 & & \end{pmatrix}$

[Levin et al. CVPR '06]

Gaussian MRF for Matting

www.alphamatting.com

Image Matting | Evaluation • Datasets • Code • Submit

Hide Help

How to use this page?

1. Please be patient until all images have loaded completely.
2. Move the mouse over the numbers in the table to see the corresponding images.
3. Drag the red rectangle in the leftmost image to change the location of the zoom.
4. Press and hold any key to temporarily deactivate the links.

Image matting evaluation results **Competition:** Low resolution High resolution
Error type: SAD MSE Gradient Connectivity

Sum of Absolute Differences	overall rank	avg. small rank	avg. large rank	avg. user rank	Troll (Strongly Transparent) Input small	large	user	Doll (Strongly Transparent) Input small	large	user	Donkey (Medium Transparent) Input small	large	user
Shared Matting	**2.7**	2.8	3.3	2	**10.8** 1	20.5 2	**15** 1	7.8 6	11.6 6	8.1 3	**4.2** 1	**5.3** 1	**4.2** 1
Segmentation-based matting	**3.7**	3.6	3.9	3.5	12.8 4	23.5 7	16.6 3	6.6 2	8.3 2	**7.3** 1	4.8 7	6.1 5	4.3 3
Improved color matting	**4**	3.9	4.1	4.1	14.9 5	24.5 8	20 7	6.7 4	9.5 4	8.5 5	4.6 3	6.1 6	4.3 4
Shared Matting (Real Time)	**4.8**	4.8	5	4.5	12.4 2	21.6 3	16.3 2	9.5 8	13.5 7	9.9 7	4.4 2	5.6 2	4.4 6
Learning Based Matting	**5.1**	5.1	4.6	5.6	16 6	22 6	18.7 5	6.6 3	**7.4** 1	7.4 2	4.8 6	6.1 4	4.3 5
Closed-Form Matting	**5.2**	5	4.4	6.1	12.7 3	21.9 5	17.2 4	**5.9** 1	8.5 3	8.6 6	4.7 4	6 3	4.3 2
Large Kernel Matting	**6.3**	7	5.9	6.1	17.2 7	21.8 4	20.7 8	7.2 5	9.6 5	8.4 4	5.3 9	6.6 9	4.6 7
Robust Matting	**7.1**	6	7.8	7.5	17.3 8	28.4 10	21.1 9	10.1 9	16.9 11	11.4 9	4.8 5	6.5 8	5 9
High-res matting	**8.4**	8	9.3	8	18.6 10	25.8 9	24.6 10	8.6 7	14.1 8	11.1 8	5 8	6.2 7	4.8 8
Random Walk Matting	**10.5**	11.3	9.4	10.8	17.9 9	**20.3** 1	19.4 6	11.3 10	15.6 9	11.8 10	5.8 10	7 10	6.3 12
Geodesic Matting	**11**	11.6	10.8	10.8	26.9 13	38.5 13	32.5 13	14.2 11	16.5 10	17.4 11	11.7 14	14 15	9.4 14
Iterative BP Matting	**11.5**	10.9	11.8	12	23.6 11	29.9 11	27.2 11	16.7 12	24.3 13	20.7 14	6.7 12	9 12	6.3 11
Easy Matting	**12**	12.1	11.9	12	23.9 12	32.6 12	30 12	17.1 13	21.8 12	19.4 13	6.3 11	7.5 11	5.8 10
Bayesian Matting	**12.9**	13	13.5	12.3	30.3 14	42.4 14	33.4 14	19.2 14	25.8 14	18.4 12	10.8 13	12.4 13	10.8 15
Poisson Matting	**14.8**	15	14.6	14.8	51.8 15	56.2 15	52 15	28.3 15	43.5 15	30.7 15	12.1 15	13.7 14	9.2 13

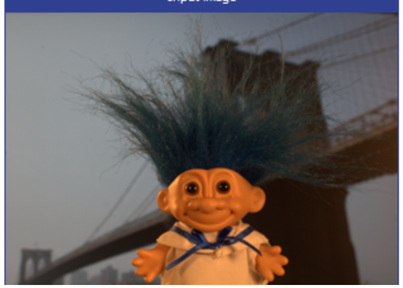
Sum of Absolute Differences	Elephant (Medium Transparent) Input small	large	user	Plant (Little Transparent) Input small	large	user	Pineapple (Little Transparent) Input small	large	user	Plastic bag (Highly Transparent) Input small	large	user	Net (Highly Transparent) Input small	large	user
Shared Matting	2.1 2	5.8 5	2.9 2	**5.9** 1	9.2 2	**11.4** 1	**5** 1	**8.8** 1	**6.8** 1	34.9 8	34.9 6	34.3 6	23.9 2	28.4 3	**25.7** 1
Segmentation-based matting	2.1 3	3.9 2	3.1 3	6.7 2	**8** 1	13.4 4	6 4	8.8 2	8.2 3	31.6 4	35.6 7	38.8 8	24.5 3	32 5	26.7 3
Improved color matting	2.6 7	5.4 4	3.4 7	7.5 5	9.9 3	12.5 2	6 3	10.1 4	8.4 4	26.1 3	26.7 3	23.6 2	**23.8** 1	**25.6** 1	26.7 2
Shared Matting (Real Time)	2.5 6	6.8 7	3.2 5	7.1 3	10.8 4	12.6 3	5.4 2	9.7 3	7.4 2	35.5 10	35.8 8	35.5 7	27.6 5	33.4 6	29.8 4
Learning Based Matting	**2.1** 1	**3.7** 1	**2.8** 1	7.5 6	14.5 9	19.5 9	8.6 10	14.1 7	14.6 10	**22.5** 1	**24.8** 1	**19.9** 1	34.6 8	38.5 8	51.2 12
Closed-Form Matting	2.2 4	4.6 3	3.3 6	9.3 9	12.1 5	19.3 8	8.3 8	14.9 9	13.4 9	34.2 7	32.4 5	27.4 4	26.5 4	25.7 2	48.3 10
Large Kernel Matting	2.9 9	8.2 9	4.2 8	8.6 8	12.1 6	14.7 5	8 7	13.4 6	11.2 7	33 5	31.8 4	26.1 3	32.1 6	32 4	38.4 7
Robust Matting	2.8 8	7.3 8	4.4 9	7.3 4	14 8	18.1 7	6.8 5	14.6 8	10.6 6	22.7 2	26.1 2	32.1 5	34.4 7	37 7	38 6
High-res matting	2.5 5	8.3 10	3.2 4	7.8 7	14 7	21.4 10	8.5 9	18.1 12	12.2 8	35.3 9	38.1 10	42.6 11	38.7 9	54.6 11	36.8 5
Random Walk Matting	3.4 10	6.7 6	4.6 10	13.1 12	22.1 12	27.4 12	12.3 13	18 11	15.7 13	44.1 13	43.5 13	41 10	75.1 13	81.8 13	80.6 13
Geodesic Matting	7.6 14	15.1 13	8.7 14	12.8 11	16.7 11	15.1 6	7.3 6	12.1 5	9.8 5	37.3 12	37.4 9	42.8 12	48.6 12	50 10	48.6 11
Iterative BP Matting	3.8 11	11.3 12	6.8 13	14.1 13	22.8 13	27.9 13	11.4 12	19 13	14.7 11	33.4 6	39.3 11	47.5 14	40.6 10	48.1 9	45.1 9
Easy Matting	4.7 12	10.5 11	5.6 11	12.1 10	15.7 10	22.9 11	11.2 11	17 10	14.8 12	49.5 14	49.6 14	46.2 13	77.8 14	108.6 15	109.2 14
Bayesian Matting	6.6 13	18.5 15	6.2 12	14.2 14	29.8 14	33.2 14	15.4 14	30.6 14	19.7 14	35.8 11	40.6 12	39.6 9	45.3 11	76.8 12	43.6 8
Poisson Matting	11.7 15	18.4 14	11.2 15	22.4 15	36.8 15	55.5 15	21.4 15	32.2 15	22.7 15	53.6 15	72.9 15	58.4 15	125.5 15	84.8 14	139.7 15

Move the mouse over the numbers in the table to see the corresponding images.Click to compare with the ground truth. Press a key to deactivate the links (to better use the zoom).

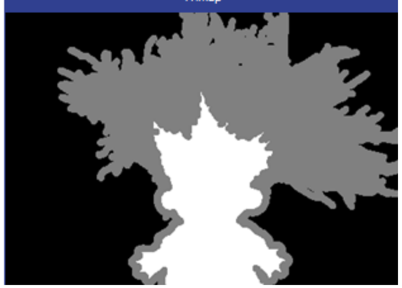
Troll - small - Segmentation-based matting alpha

Zoom in

Input image



Trimap



http://www.alphamatting.com/

Continuous Labels - Summary

- Used for problems with continuous label space:
  e.g. image intensity, depth, motion, transparency (i.e. matting)

- Regularization term (pairwise, higher-order) must be written in some parametric form (Gaussian, Filters, etc)

- Gaussian Random Field are often not used since data-term is typically non-Gaussian and often expansive to evaluate for continuous labels.

- Gaussian Random Field can be useful successfully when the terms are set in an image-adaptive way (see e.g. Jancsary, Nowozin, Sharp, and Rother, Regression Tree Fields, CVPR 2012)