SIGGRAPH 2004

Lazy Snapping

Yin Li
Jian Sun
Chi-Keung Tang
Heung-Yeung Shum
Microsoft Research Asia
Hong Kong University

GrabCut

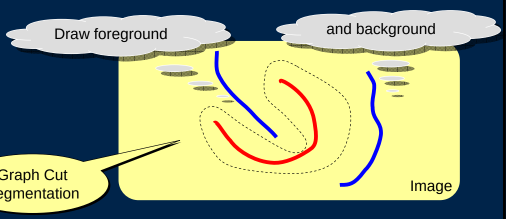
Carsten Rother
Vladimir Kolmogorov
Andrew Blake
Microsoft Research Cambridge-UK

Interactive Image Cutout

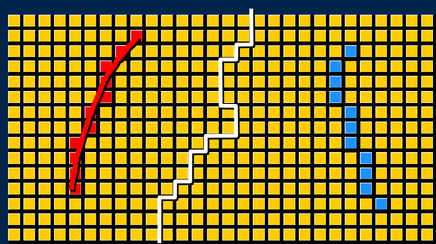
- Separate an object from its background
- Compose the object on another image


Interactive Graph Cut

- (Boykov & Jolly. ICCV'01)
- Optimized by s-t min-cut algorithm

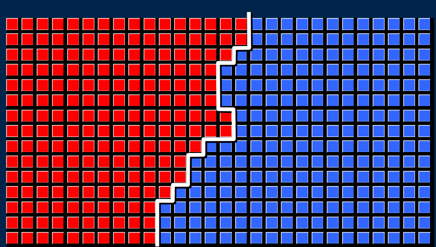


Interactive Graph Cut



- (Boykov & Jolly. ICCV'01)

Interactive Graph Cut



- (Boykov et al. ICCV'01)

Hard Constraints

- X : Segmentation.

$$x_i \in \{"obj", "bkg"\}$$

- Hard Constraint:

$$\forall i \in O \quad x_i = "obj"$$
$$\forall i \in B \quad x_i = "bkg"$$

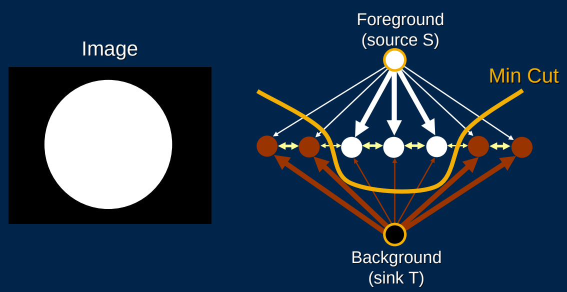
Soft Constraints

- Minimize the Energy:

$$E(X) = \sum_{i \in V} E_1(x_i) + \lambda \sum_{\substack{i,j \in E \\ x_i \neq x_j}} E_2(x_i, x_j)$$

- E_1 : Region: Color difference to user marks
- E_2 : Boundary: Color similarity between pixels

Image as a Weighted Graph



Graph: source & sink, n-links & t-links

Cut=Segmentation: Separate 'source' & 'sink'
Energy of cut: sum wieghts of edges

Min-Cut Max-Flow: Global minimal enegry in polynomial time

Weights

t-links

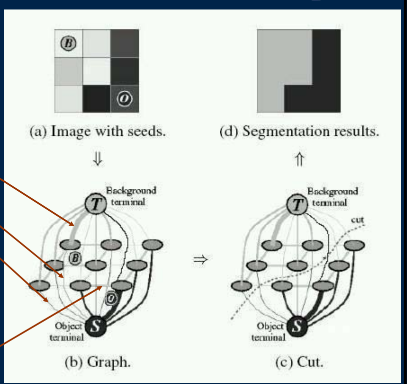
$i \in B \Rightarrow \{i,T\}:\infty$

$\{i,S\}:0$

$i \in U \Rightarrow E_1(x_i) = h_{x_i}(I_i)$

n-links

$E_2(x_i, x_j) \propto exp(-(I_1 - I_2)^2)$



Min Cut = Minimize Soft Constraints keeping Hard Constraints

Lazy Snapping

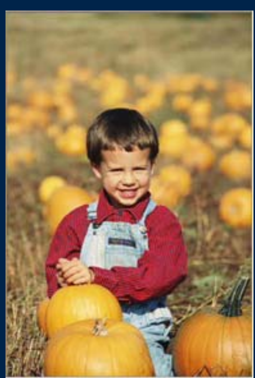
Li et al.

SIGGRAPH'04

Lazy Snapping

- Lazy Snapping for Lazy Users
- 2 Steps UI:
  1. Coarse Step: Obj/Bkg Marking

  => Graph Cut



Lazy Snapping

2. Fine Step:
   a. Border Brush
   b. Pixel Editing

=> Graph-Cut on border

Weights

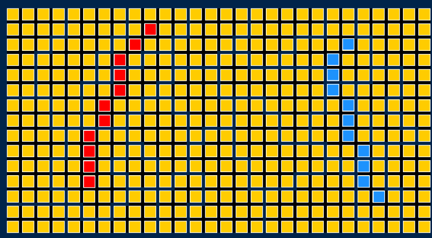
- E_1 : Color difference to user marks
  - Intensities -> Colors
  - Histogram -> “K-means” clustering

$E_1(x_i = \text{"}obj\text{"}) \propto$ RGB_dist to closest cluster centroid

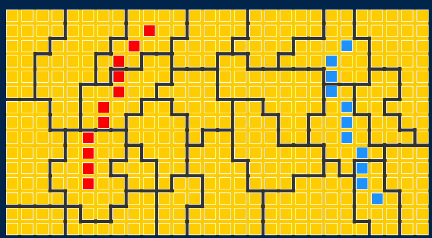
- E_2 : Color similarity between pixels
  - For neighboring pixels of different x_i

$$E_2(x_i, x_j) = \frac{1}{1+\|C_i - C_j\|^2}$$

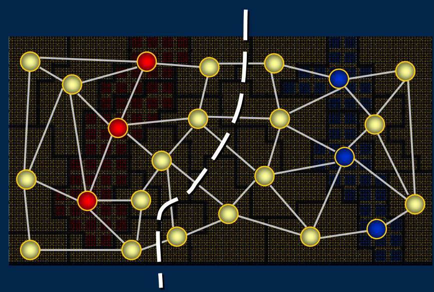
Per-Pix Graph Cut



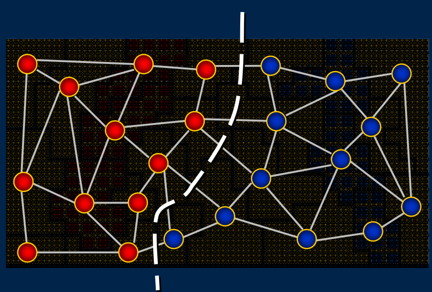
Pre-Segmentation



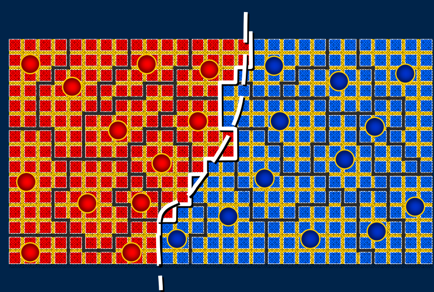
Graph Cut on Regions



Graph Cut on Regions



Graph Cut on Regions



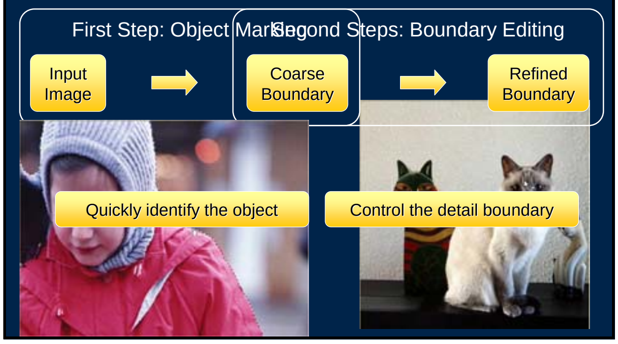
Graph Cut Algorithm

Per-pixel method	Region based method
Pixels	Small regions
Neighbors	Region connection
Pixel color	Region mean color
Color difference	Region color difference

Region-based Graph Cut

- Advantages
  - More than 10 times fewer nodes
  - Instant feedback of cutout result
- Pre-processing overhead
  - 2~3 seconds background processing

Divide and Conquer



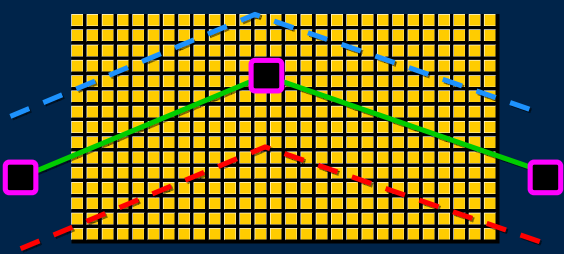
Polygon Fitting

- First vertex – border pixel with highest curvature
- Next vertices: furthest boundary pixel
- Stop when distance < thresh

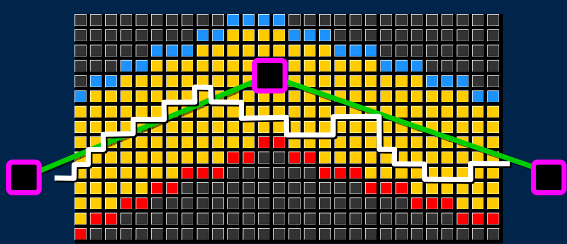
Border Editing

- Brush - Replace polygon segment
- Vertex Editing: Move/Add/Delete
  => Graph Cut on border pixels

Band of Uncertainty

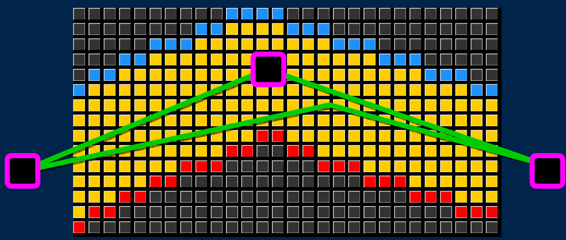


Optimization in the Band

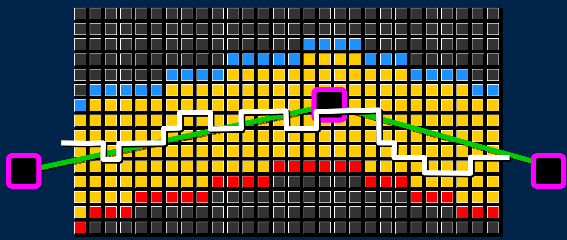


Pixel Based Graph Cut Segmentation

Edit the Polygon Vertices



Edit the Polygon Vertices



Low Contrast Example

Boundary Editing

Boundary Editing

- For Low Contrast case:
  - In E_2 - Add a term to reflect distance from polygon
- Hard Vertex constraint
  - Adjust graph so cut passes through vertex

Video Demo (Left boy)

Video Demo (Right Boy)

Summary: Two Steps

First Step: Object Marking | Second Steps: Boundary Editing

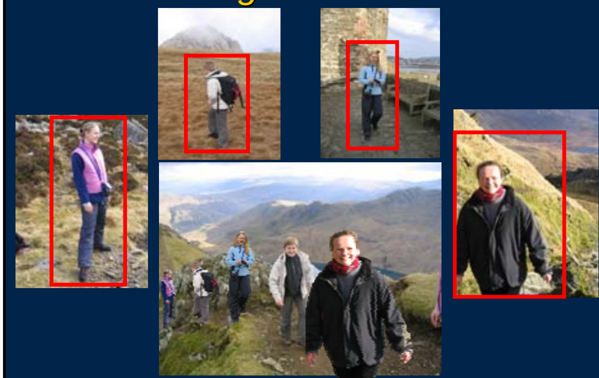
Input Image → Small Regions → Coarse Boundary → Editable Polygon → Refined Boundary

Pre-Segmen | Region Based Graph Cut | Polygon F | Band Pixels Graph Cut

GrabCut

Interactive Foreground Extraction using Iterated Graph Cuts

Photomontage



Iterated Graph Cut

User Initialization

GMM estimation for learning colour distributions

Graph cuts to infer the segmentation

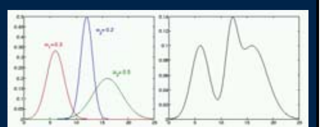
Gaussian Mixture Models (GMMs)

- GMM instead of Histogram (Color model)
- Assume distribution is a mixture of Gaussians

$G_{\mu,\Sigma}(x) - Gaussian$

$GMM(x) = \sum_{k=1}^{K} w_k G_{\mu_k,\Sigma_k}(x)$

$\sum w_k = 1$



- EM algorithm – find best w_k, μ_k, Σ_k for the given set of samples
- GrabCut – Different approach

Iterated Graph Cuts

- E_1 – GMMs (E_2 – No change)
- Algorithm:
  1. Initialize $B,\ U=\overline{B},\ F=\phi$
     Initialize GMMs w_k, μ_k, Σ_k
  2. Repeat (until constant energy)
     a. $\forall p \in U$ assign best G_k => $2K$ clusters
     b. For each cluster calculate w_k, μ_k, Σ_k => 2 GMMs
     c. Find Min Cut => U decreases
  3. Apply border matting
  4. Enable user editing & repeat

Incomplete Labeling

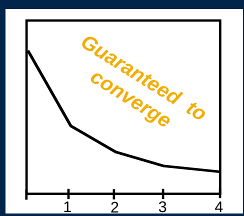
- User specifies border => $B,\ U=\overline{B},\ F=\phi$
- F populates through iterations
- Some F pixels can be retracted. B cannot

Editing (In case of error):

- User adds F, B (brush)
- Re-compute
- Graph Cut can be reused.

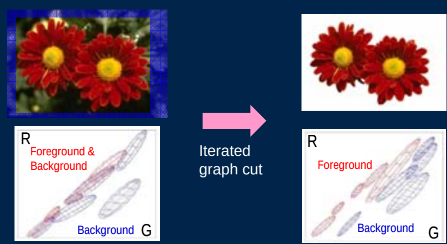
Iterated Graph Cuts

Result



Energy after each Iteration

Gaussian Separation



Gaussian Mixture Model (typically K=5)

Moderately straightforward examples

Difficult Examples

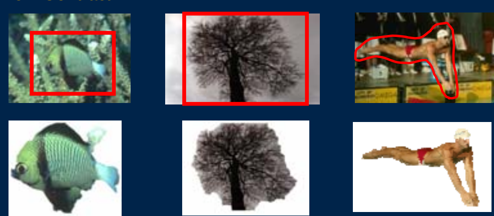
Camouflage & Low Contrast

Fine structure

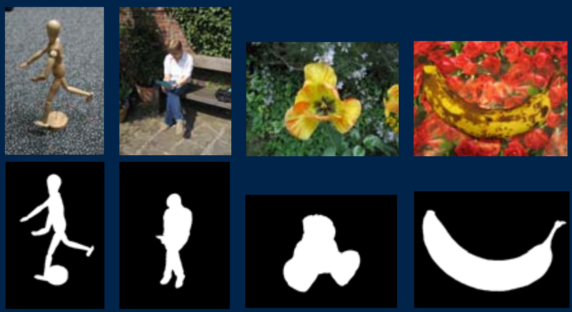
No telepathy

Initial Rectangle

Initial Result

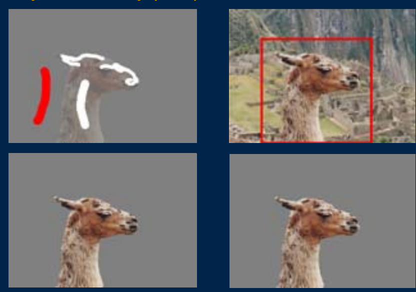


Evaluation – Labelled Database



Comparison

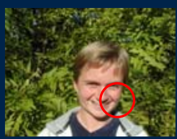
	Boykov and Jolly (2001)	GrabCut
User Input		
Result		

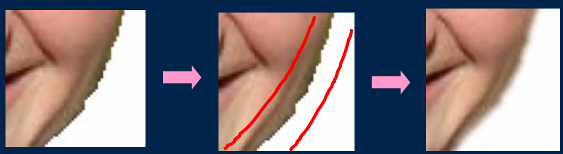


Error Rate: 0.72%   Error Rate: 0.72%

Border Matting

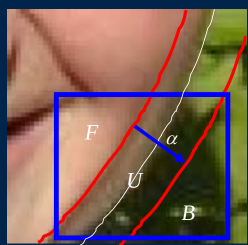
Extract α-values along border





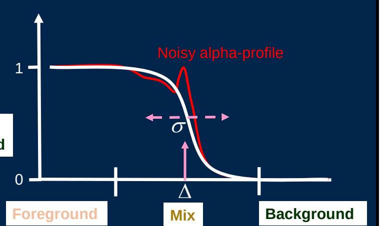
Hard Segmentation → Band of Uncertainty → Soft Segmentation

Bayes Matting - *Chuang et. al. (2001)*



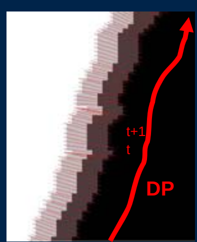
- Create U band $\pm w$
- Local rectangle
- Estimate G_F, G_B
- U: $\mu_\alpha = \alpha\mu_F + (1-\alpha)\mu_B$
- $G_U(\alpha) = G(\mu_\alpha, \Sigma_\alpha)$
- Find α that maximizes G_U with respect to pixels in U

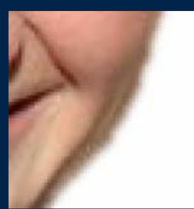
Border Matting - GrabCut



Fit a smooth alpha-profile with parameters Δ, σ

Dynamic Programming



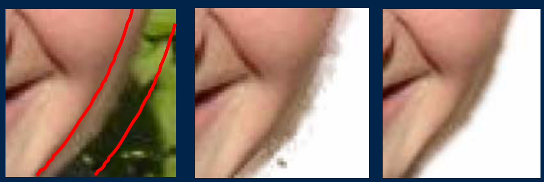


Result using DP Border Matting

$$\underbrace{Max:\ G(\mu_\alpha, \Sigma_\alpha)}_{\text{Noisy alpha-profile}} \qquad \underbrace{Min: \sum_{t=1}^{T} (\Delta_t - \Delta_{t-1})^2 + (\sigma_t - \sigma_{t-1})^2}_{\text{Regularisation}}$$

Summary

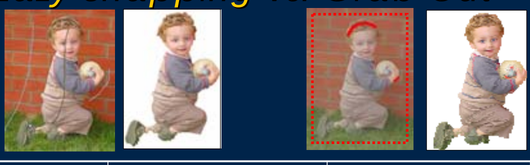
- $G_U(\alpha)$ should match U pixels
- α should change like a soft step function
- Step function should change smoothly along contour



Input | Bayes Matting (no regularization) | GrabCut (with regularization)

Matting Results

Lazy Snapping vs. Grab Cut



	Lazy Snapping	GrabCut
User Interface	Marking brush – FG + BG Overriding brush Vertex editing	Rectangle/lasso – BG only Marking brush - [optional]
Algorithm	Region-based Graph Cut Border pixel Graph Cut	Iterative Graph Cut
Performance	Fully interactive Includes Pre-Processing	Fast
Border	Border Editing	Border Matting

Thank You