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.
  \( x_i \in \{\text{"obj", \"bkg"}\} \)
- Hard Constraint:
  \[ \forall i \in O \quad x_i = \text{"obj"} \]
  \[ \forall i \in B \quad x_i = \text{"bkg"} \]
**Soft Constraints**

- Minimize the Energy:

\[ E(X) = \sum_{i \in B} E_1(x_i) + \lambda \sum_{(i,j) \in E} E_2(x_i, x_j) \]

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

**Image as a Weighted Graph**

Image as a Weighted Graph

- **Graph**:
  - Source & sink, n-links & t-links
- **Cut-Segmentation**:
  - Separate ‘source’ & ‘sink’
  - Energy of cut: sum weights of edges
- **Min-Cut Max-Flow**:
  - Global minimal energy in polynomial time

**Weights**

- n-links:
  - \( i \in B \Rightarrow \{i, T\} : \infty \)
  - \( \{i, S\} : 0 \)

- t-links:
  - \( i \in U \Rightarrow E_j(x_i) = h_i(I_j) \)

- Energy of cut:
  - \( E(x_i, x_j) = \exp(-\theta - I_{ji}^2) \)

**Lazy Snapping**

- Lazy Snapping for Lazy Users
- 2 Steps UI:
  1. Coarse Step:
     - Obj/Bkg Marking
     - Graph Cut
  2. Fine Step:
     - Border Brush
     - Pixel Editing
     => Graph-Cut on border

**Li et al.**

SIGGRAPH’04
Weights

- $E_1$: Color difference to user marks
  - Intensities -> Colors
  - Histogram -> “K-means” clustering
  - $E_1(x_i = \text{obj}) \propto \text{RGB dist to closest cluster centroid}$

- $E_2$: Color similarity between pixels
  - For neighboring pixels of different $x_i$
  - $E_2(x_i, x_j) = \frac{1}{1 + \|C_i - C_j\|}$

Per-Pix Graph Cut

Pre-Segmentation

Graph Cut on Regions

Graph Cut on Regions

Graph Cut on Regions

Graph Cut on Regions

Graph Cut on Regions
**Graph Cut Algorithm**

<table>
<thead>
<tr>
<th>Per-pixel method</th>
<th>Region based method</th>
</tr>
</thead>
<tbody>
<tr>
<td>Pixels</td>
<td>Small regions</td>
</tr>
<tr>
<td>Neighbors</td>
<td>Region connection</td>
</tr>
<tr>
<td>Pixel color</td>
<td>Region mean color</td>
</tr>
<tr>
<td>Color difference</td>
<td>Region color difference</td>
</tr>
</tbody>
</table>

**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**

- **First Step:** Object Marking
- **Second Steps:** Boundary Editing

  - Quickly identify the object
  - Control the detail boundary

**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
  
  \[ \Rightarrow \text{Graph Cut on border pixels} \]

**Band of Uncertainty**

**Optimization in the Band**

**Pixel Based Graph Cut Segmentation**
Low Contrast Example

Boundary Editing

- For Low Contrast case:
  - 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

**GrabCut**

Interactive Foreground Extraction using Iterated Graph Cuts

**Photomontage**

**Iterated Graph Cut**

**Gaussian Mixture Models (GMMs)**

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

\[ G_{\mu, \Sigma}(x) = \sum_{k=1}^{K} w_k G_{\mu_k, \Sigma_k}(x) \]

- EM algorithm - find best \( w_k, \mu_k, \Sigma_k \) for the given set of samples
- GrabCut - Different approach
Iterated Graph Cuts

- $E_i$ - GMMs ($E_j$ - No change)
- **Algorithm:**
  1. Initialize $B_i$, $U = \overline{B_i}$, $F = \phi$
  2. Repeat (until constant energy)
     a. Assign best $G_k$ to $2k$ clusters
     b. For each cluster calculate $w_i, \mu_i, \Sigma_i$ to $2$ GMMs
     c. Find Min Cut $\Rightarrow U$ decreases
  3. Apply border matting
  4. Enable user editing & repeat

Incomplete Labeling

- User specifies border $\Rightarrow B_i, U = \overline{B_i}$, $F = \phi$
- $F$ 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.

Moderately straightforward examples

Gaussian Separation

- Gaussian Mixture Model (typically $K=5$)

Difficult Examples

- Camouflage & Low Contrast
- Fine structure
- No telepathy
Evaluation - Labelled Database

Comparison

Boykov and Jolly (2001) vs GrabCut

Error Rate: 0.72%

Error Rate: 0.72%

Border Matting

Extract \( \alpha \)-values along border

Hard Segmentation → Band of Uncertainty → Soft Segmentation

Bayes Matting - Chuang et. al. (2001)

- Create \( U \) band ± \( w \)
- Local rectangle
- Estimate \( G_F, G_B \)
- \( U: \mu_x = a \mu_x \times (1-\alpha) \mu_x \)
- \( G(x) = G(\mu_x, \Sigma_x) \)
- Find \( \alpha \) that maximizes \( G_U \) with respect to pixels in \( U \)

Border Matting - GrabCut

Fit a smooth alpha-profile with parameters \( \Delta, \sigma \)

Dynamic Programming

Result using DP Border Matting

Max: \( G(\mu_x, \Sigma_x) \)

Min: \( \sum (\Delta - \Delta_i)^2 + (\sigma - \sigma_i)^2 \)
Summary

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

Matting Results

Lazy Snapping vs. Grab Cut

<table>
<thead>
<tr>
<th>Feature</th>
<th>Lazy Snapping</th>
<th>Grab Cut</th>
</tr>
</thead>
<tbody>
<tr>
<td>User Interface</td>
<td>Marking brush - FG + BG</td>
<td>Rectangular lasso - BG only</td>
</tr>
<tr>
<td></td>
<td>Overiding brush</td>
<td>Masking brush - (optional)</td>
</tr>
<tr>
<td></td>
<td>Vertex editing</td>
<td></td>
</tr>
<tr>
<td>Algorithm</td>
<td>Region-based Graph Cut</td>
<td>Iterative Graph Cut</td>
</tr>
<tr>
<td></td>
<td>Border pixel</td>
<td>Graph Cut</td>
</tr>
<tr>
<td>Performance</td>
<td>Fully Interactive</td>
<td>Fast</td>
</tr>
<tr>
<td></td>
<td>Includes Pre-Processing</td>
<td></td>
</tr>
<tr>
<td>Border Edge</td>
<td>Border Editing</td>
<td>Border Matting</td>
</tr>
</tbody>
</table>

Thank You