GrabCut
Interactive Foreground Extraction using Iterated Graph Cuts

Carsten Rother
Vladimir Kolmogorov
Andrew Blake

Microsoft Research Cambridge-UK
Photomontage
Problem

Fast & Accurate ?

GrabCut – Interactive Foreground Extraction
Graph Cuts

Boykov and Jolly (2001)

**Cut:** separating source and sink; **Energy:** collection of edges

**Min Cut:** Global minimal energy in polynomial time
Iterated Graph Cut

User Initialisation

K-means for learning colour distributions

Graph cuts to infer the segmentation
Iterated Graph Cuts

Result

Energy after each Iteration

Guaranteed to converge
Colour Model

Gaussian Mixture Model (typically 5-8 components)

Iterated graph cut

GrabCut – Interactive Foreground Extraction
Coherence Model

An object is a coherent set of pixels:

Blake et al. (2004): Learn jointly
Border Matting

Hard Segmentation → Automatic Trimap → Soft Segmentation

GrabCut – Interactive Foreground Extraction
Border Matting

Fit a smooth alpha-profile with parameters

Noisy alpha-profile
Comparison

Boykov and Jolly (2001) vs GrabCut

User Input

Result

Error Rate: 0.72%  Error Rate: 0.72%

GrabCut – Interactive Foreground Extraction
Moderately straightforward examples

... GrabCut completes automatically
Difficult Examples

- Initial Rectangle
- Initial Result

GrabCut – Interactive Foreground Extraction
Difficult Examples
Lazy Snapping

†Yin Li*
‡Jian Sun
†Chi-Keung Tang
‡Heung-Yeung Shum
†Hong Kong University of Science and Technology
‡Microsoft Research Asia

(a) Input image
(b) Object Marking
(c) Boundary editing
(d) Output composition
Lazy Snapping Overview

- Main Steps:
  1. Mark strokes as foreground & background (object marking)
  2. Perform Boykov & Jolly style graphcut
  3. Boundary Editing

- Familiar Formulation
  - Minimize Gibbs Energy:
    \[ E(X) = \sum_{i \in \mathcal{V}} E_1(x_i) + \lambda \sum_{(i,j) \in \mathcal{E}} E_2(x_i, x_j) \]

- \( E_1(X) \) – Cluster (K-means), encodes color similarity of node (likelihood energy)
- \( E_2(X) \) – encodes energy due to gradient along object boundary (color gradient between two nodes)
Lazy Snapping—Pre-processing

Watershed Algorithm

- Picture is presegmented with the Watershed Algorithm
- Graphcut works on segment instead of pixel level
- Segments are connected if one pixel within them is connected
- Results are available almost instantly
Lazy Snaping – Speedup

<table>
<thead>
<tr>
<th>Image</th>
<th>Dimension</th>
<th>Nodes Ratio</th>
<th>Edges Ratio</th>
<th>Lag with Pre-segmentation</th>
<th>Lag without Pre-segmentation</th>
</tr>
</thead>
<tbody>
<tr>
<td>Boy</td>
<td>(408, 600)</td>
<td>10.7</td>
<td>16.8</td>
<td>0.12 s</td>
<td>0.57 s</td>
</tr>
<tr>
<td>Ballet</td>
<td>(440, 800)</td>
<td>11.4</td>
<td>18.3</td>
<td>0.21 s</td>
<td>1.39 s</td>
</tr>
<tr>
<td>Twins</td>
<td>(1024, 768)</td>
<td>20.7</td>
<td>32.5</td>
<td>0.25 s</td>
<td>1.82 s</td>
</tr>
<tr>
<td>Girl</td>
<td>(768, 1147)</td>
<td>23.8</td>
<td>37.6</td>
<td>0.22 s</td>
<td>2.49 s</td>
</tr>
<tr>
<td>Grandpa</td>
<td>(1147, 768)</td>
<td>19.3</td>
<td>30.5</td>
<td>0.22 s</td>
<td>3.56 s</td>
</tr>
</tbody>
</table>
Boundary Editing

- Graph Cut is only step 1
- Allow user to edit boundary directly
- Formulate as Graph Cut
- Allow user to guide boundary regions w/ brush
User Study

- **Results**
  - Easy of Use – Lazy Snapping 20% less mistakes
  - Speed – Lazy Snapping 60% less time
  - Accuracy – Lazy Snapping 60% less pixels wrong

- **Feedback**
  - “Almost Magic”
  - “Much Easier”

- **Suggestions**
  - Get rid of 2 step process
  - Make it easy to switch between graph cut strokes and boundary editing
More Results

(a) Girl (4/2/12)  
(b) Ballet (4/7/14)  
(c) Boy (6/2/13)  
(c) Grandpa (4/2/11)  
(d) Twins (4/4/12)