GrabCut

Iterative image segmentation through minimum graph cuts

Michael Meyer, Ryan Kuck, Kevin Nadro
CS 445 Computational Photography Final Project

Abstract

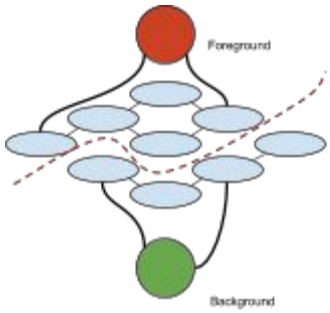
In this paper we describe our final project, GrabCut, which augments graph cut by iteratively minimizing the energy in an image to find the local optimal binary segmentation. The method features a simple user interface which only requires the user to draw a rectangle around the object to be cut. This initialization defines only the background pixels, and then iteratively improves the segmentation by recalculating the probability distributions of the background and foreground colors and then uses these estimates in graph cut.

Introduction

We researched many different potential projects, but decided on implementing GrabCut. We believed this to be something that would be achievable in our time frame. We underestimated the complexity of the algorithm at first, but in the end we were able to successfully implement it.

Algorithm and Technique

In this section we describe how our implemented algorithm works.



This summary of the technique uses these sources
[Graph cut - Boykov and Jolly 2001]
[GrabCut - Rother and Kolmogorvov and Blake 2004]
[Implementing GrabCut - Talbot and Zu]

Image and clustering

The image is defined in an RGB color space. Instead of using histograms of pixels in the original implementation of Graph Cut [Boykov and Jolly 2001] the algorithm of GrabCut uses Gaussian Mixture Models (GMM) to cluster the RGB pixel values from the image. There are two GMMs, one foreground and one for background pixels. Each GMM is defined to have 5 clusters (K = 5, set to be constant based on the original paper). There is a hard segmentation of the pixels between the foreground and background GMMs represented by $\alpha_n = 0$ (background) or 1 (foreground).

Definition of a GMM

A Gaussian Mixture Model (GMM), the probability distribution used, is defined by several parameters.
K - the number of clusters, our GMMs have 5.
Every cluster in a GMM has,

- μ - mean RGB value
- π - weighting coefficient
- Σ - covariance matrix (3x3)

$\theta = \{\pi(\alpha, k),\ \mu(\alpha, k),\ \Sigma(\alpha, k),\ \alpha = 0, 1,\ k = 1 \ldots K\}$

Trimap setup

To help with segmentation, a trimap is used. It is defined by $T = \{T_B,\ T_U,\ T_F\}$. T_b stores all background pixels, T_U stores unknown pixels, and T_F stores foreground pixels. At the beginning of GrabCut, the user draws a box around the foreground object. The background pixels in the trimap (T_B) are initialized with everything outside this rectangle. The unknown pixels are set to be $T_U = \overline{T_B}$ and $T_F = \varnothing$.

Gaussian Mixture Model (GMM) Setup

After initializing the trimap and the alpha values, two GMMs are created. The background GMM is created by T_B which has values $\alpha_n = 0$. The foreground GMM is created by T_U which has values $\alpha_n = 1$

Assigning GMMs

GrabCut performs iterations to refine the labeling of foreground and background pixels. All of the pixels in T_U are assigned a cluster based off the minimum unary weighting function $D(n)$.

$n \in T_U$, z is the RGB row vector of n

$$D(n) = -log \sum_{i=1}^{K} \pi(\alpha_n, i) \frac{1}{\sqrt{det\Sigma(\alpha_n, i)}} e^{(-\frac{1}{2}[z_n - \mu(\alpha_n, i)]^T \Sigma(\alpha_n, i)^{-1} [z_n - \mu(\alpha_n, i)])}$$

Equation 2 from *[Implementing GrabCut - Talbot and Zu]*

for each $n \in T_U$

$k_n := \underset{k_n}{argmin}\, D_n(\alpha_n, k_n, \theta, z_n)$

$k_n \in \{1, \ldots, K\}$

Equation from Figure 3 in [GrabCut *Rother and Kolmogorvov and Blake 2004]*

Learning GMMs

Assigning pixels to the closest cluster in the foreground or background GMM refines the GMM distributions to better define the foreground and background pixels. This learning step takes the assigned k values to recalculate both GMMs. The old GMMs are discarded and the new GMMs will be used to calculate the unary weights in the next step.

Graph structure

The graph structure for GrabCut is defined by each pixel and its edges to its four neighboring pixels (North, East, South, West). Every pixel also connects to two terminal nodes defined as sink (background) and source (foreground). The terminal nodes are important because they are used when performing the min cut to segment foreground and background pixels. The neighborhood weights are set such that min cut will find it expensive to separate pixels with similar intensity, and cheap to cut where they are different (along edges). These weights are static across iterations. Terminal weights must be recalculated based on the new GMMs.

Edge weight function for neighboring pixel is defined as:

m, n are any two pixels from the entire original image
z is the RGB row vector of n or m

$\beta = (2 < (z_m - z_n)^2 >)^{-1}$, $<\cdot>$ denotes expectation over the original image.
Equation 5 in [GrabCut *Rother and Kolmogorvov and Blake 2004]*

$$\delta(\alpha_m, \alpha_n) = \begin{cases} 1 \text{ if } \alpha_m \neq \alpha_n \\ 0 \text{ otherwise} \end{cases}$$

Equation 3 from *[Graph cut - Boykov and Jolly 2001]*

$$N(m, n, \alpha_m, \alpha_n) = \delta(\alpha_m, \alpha_n) \frac{50}{dist(m,n)} e^{-\beta \|z_m - z_n\|^2}$$

Equation 1 from *[Implementing GrabCut - Talbot and Zu]*
Note: In our graph structure we only have 4-way connectivity therefore $dist(m, n)$ will always be 1.

Unary weight function for sink and source is defined as:

$n \in T_U$, z is the RGB row vector of n

$$D(n) = -log \sum_{i=1}^{K} \pi(\alpha_n, i) \frac{1}{\sqrt{det \Sigma(\alpha_n, i)}} e^{(-\frac{1}{2}[z_n - \mu(\alpha_n, i)]^T \Sigma(\alpha_n, i)^{-1} [z_n - \mu(\alpha_n, i)])}$$

Equation 2 from *[Implementing GrabCut - Talbot and Zu]*

Estimating Segmentation

Once the graph structure is initialized with all the weights, min cut is performed on the graph to estimate new α_n values for the pixels in T_u.

The Gibbs energy segmentation is defined as:

α^* is a column vector of all α labels for all $n \in T_U$
z^* is a column vector of all $n \in T_U$, it stores the RGB data
k^* is a column vector of all the k values from Assigning GMM part

Rewriting $N(m,n,\ \alpha_m,\alpha_n)$ as $N(\alpha^*,z^*)$

$E(\alpha,k,\theta,z) = U(\alpha^*,k^*,\theta,z^*) + N(\alpha^*,z^*)$
Equation 7 from [GrabCut *Rother and Kolmogorvov and Blake 2004]*

$U(\alpha^*,k^*,\theta,z^*) = \sum_n D(\alpha_n,\ k_n,\ \theta,\ z_n)$
Equation 8 from [GrabCut *Rother and Kolmogorvov and Blake 2004]*

After each iteration, α_n may switch from 0 to 1, or vice versa. The process is repeated from steps **Assigning GMM** to **Estimating Segmentation** until convergence of the labeling of foreground and background pixels.

GrabCut Summary

Initialization Step

- The user initializes the trimap by only giving the background pixels. The foreground set of pixels is set to the empty set. The unknown pixels are set to the compliment of the background pixels.
- Hard segmentation is performed to create a foreground and background pixel sets. The background pixels go into the background pixel set. The unknown pixels is the foreground set.
- Create foreground and background GMMs based off the sets previously defined.

Iterative Step

1. Assign a foreground and background GMM cluster to every pixel in the unknown set based off the minimum distance to the respective clusters.
2. Learn GMM parameters based off the pixel data with the newly assigned foreground and background clusters. This step will get rid of the old GMM and create a new GMM with the assigned foreground and background clusters from every pixel.
3. A graph is constructed and Min Cut runs to estimate new foreground and background pixels.
4. Repeat steps 1-3 until the the label of foreground background pixels converges.

Results

The following are some final segmentation results with no extra user input given other than the initial bounding box.






References

- Graph Cut
  - http://www.eecs.berkeley.edu/~efros/courses/AP06/Papers/boykov-iccv-01.pdf
- GrabCut -Interactive Foreground Extraction using Iterated Graph Cuts
  - http://research.microsoft.com/apps/pubs/default.aspx?id=67890
- GCMex page
  - http://vision.ucla.edu/~brian/gcmex.html
- Implementing Grabcut
  - http://read.pudn.com/downloads94/doc/374106/Grabcut.pdf
  - http://www.justintalbot.com/research/course-work/