Integrative Foreground and Background Segmentation using Graph Cuts

Aashish Sheshadri
University of Texas at Austin
aashishs@cs.utexas.edu

Shruti Bhosale
University of Texas at Austin
shruti@cs.utexas.edu

***Abstract*—Accurate Image segmentation enables higher precision in computer vision tasks of recognition, tracking and is often the most desired property in image editing applications. Fully automatic image segmentation is an extremely challenging task, mainly because of the non-trivial association of local segments to an object, which generally has high variability in color and appearance. Often times the association is more semantic in nature, and hence easier for a human to comprehend. An approach widely explored in this context is interactive image segmentation, where in human input steers the segmentation in the desired direction. We implement an interactive segmentation method as a st-Mincut problem as formulated by Boykov and Jolly[6]. We extend the algorithm they proposed by using local structure encoding image descriptors, mixture models, and normalized color histograms to model the prior likelihoods of pixels belonging to the foreground or background segments and investigate the results of the same.**

I. Introduction

Image segmentation is an important preprocessing step for problems in computer vision, object recognition, image editing etc. The focus of our project is on interactive foreground and background segmentation using graph cuts. This is a binary segmentation problem i.e. each pixel belongs to either the foreground or the background. Fully automatic image segmentation is a hard problem. Interactive image segmentation asks the user to select some seed points belonging to all segments. The approach uses hard constraints defined by the user to alleviate the problems associated with automatic segmentation.

In particular, we implement the seminal work of Boykov and Jolly [6]. They were the first to use graph cuts to obtain a globally optimum segmentation for the interactive segmentation problem. The approach they developed is fast and generic. It is generic because it can segment any N-dimensional image. This is useful for applications like video editing and medical image segmentation. They model the problem in terms of a Markov Random Field. They calculate edge costs to introduce soft constraints related to regional and boundary coherence. Modeled this way, the problem of binary segmentation reduces to finding the minimum cost cut through the graph.

We extend the work done in [6] by using local structure encoding image descriptors, mixture models, and normalized color histograms to model st-Mincut graph and investigate the results for the same.

II. Related Work

We discuss related work in the field go image segmentation and as three main sections. We first discuss approaches that do not use graph cuts to compute segmentation of an image. We then move to automated image segmentation algorithms that use graph cuts. Finally we look at algorithms that are based on Boykov and Jolly's seminal work[6].

A. Interactive Image Segmentation Without Graph Cuts

A tool called Intelligent Scissors was proposed by [7], which requires the user input as points on the boundary of the desired segment. The image is then modeled as a weighted graph, on which the Dijkstra's shortest path algorithm is used to compute the shortest path between the user-defined points, which is used to segment the image. However, this method is not very practical, as user input has to be placed accurately along or very close to the boundary.

Magic Wand is a basic segmentation tool available in Adobe Photoshop 7[8]. Given a single user-specified seed point, a segmentation is computed which consists of pixels in the neighborhood of the seed point whose color/texture values lie within a given threshold from the color or the texture of the seed point. However, this is a very naive approach to the complex problem of segmentation and works only for simple images. Geodesic Active Contours proposed in [9] involves computing dynamic contours within an image, which merge and split according to the geometry of the image. The drawback of this method is that it can get stuck in local optima.

To summarize, classical interactive segmentation algorithms optimize either color (texture) information or edge (contrast) information, not both simultaneously. Boykov-Jolly's graph cut framework[6] was one of the first to balance both regional and boundary properties.

B. Automated Segmentation using Graph Cuts

In [10], the image is segmented into parts such that the corresponding maximum cut between the segments is

minimized. But, since this approach takes only regional consistency into account, it tends to produce small segments. This is remedied by [11] in which the cost of a graph cut is normalized by a value, which signifies the associativity of the generated segments to the rest of the segments in the graph. This approach takes the global properties of the image into account while segmenting an image. However, finding the optimum cut is NP-hard and is hence obtained by approximation, which may not always give the best result.

In [12], image segmentation based on a metric which measures the dissimilarity between pixels along the boundary of the two proposed segments relative to a measure of the dissimilarity among neighboring pixels within each of the two segments. The resulting predicate compares the inter-component differences to the intra-segment differences and is thereby adaptive with respect to the local as well as global characteristics of the image.

C. *Interactive Segmentation Based on the work of [6]*

Boykov et al. postulate that a contour can be modeled as a graph cut if the edges are weighted such that the cost of the cut is a near approximation to the length of the corresponding contour for an anisotropic Riemannian metric[13]. This formulation is done in order to compute a globally optimal solution for the problem of segmentation using geodesic active contours. Freedman et al. incorporate the knowledge of the shape of the segment in [14] by appropriately modifying the edge weights in the graph cut formulation proposed by Boykov et al. [6].

Carsten et al., propose an algorithm called "Grabcut"[3] which uses an iterative version of the graph cut algorithm proposed by [6]. They use Gaussian Mixture Models (GMMs) to model the foreground and background prior likelihoods. The algorithm iteratively estimates the optimal values of parameters involved in the energy formulation of the segmentation problem. As a result, user effort required for initialization and optimization of segmentation is less as compared to the one-shot algorithm proposed by [6]. In addition, they developed a border matting algorithm to get soft boundaries for the segments.

Andrew Blake et al.[15], propose the use of a pseudo-likelihood algorithm to jointly learn the color mixture and coherence parameters for foreground and background respectively unlike in [6] which estimates the background and foreground like hood functions based on only the seed points chosen by the user. In other words, they formulate the energy minimization model proposed by Boykov et al[6], as a probabilistic Gaussian Mixture Markov Random Field (GMMRF) and develop a pseudo-likelihood algorithm for parameter learning.

III. APPROACH

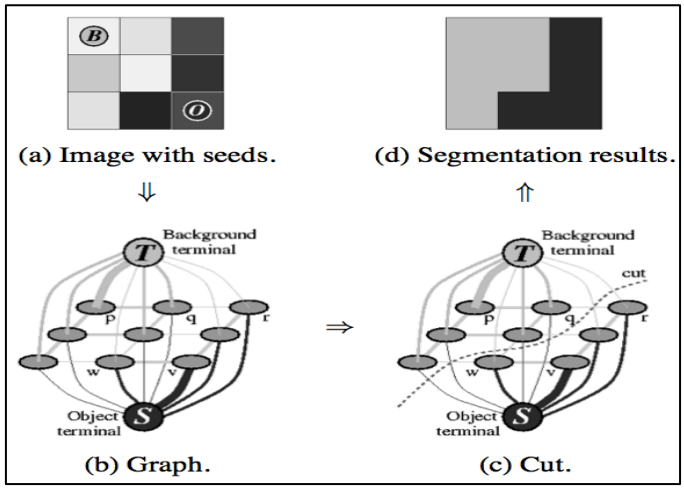

Figure 1 Formulation of foreground-background segmentation as an ST-mincut problem. Thick terminal edges indicate hard constraints from user seeds. Image credit [6].

Consider a directed weighted graph $\mathcal{G}(\mathrm{V},\mathrm{E},\mathrm{W})$ with non-negative edge weights where V is the set of nodes, E is the set of edges and the W is the edge cost function W : E → R which maps each edge (i, j) of the graph to a real number c_{ij}. Graphs used in the st-Mincut problem have certain special nodes called the terminal nodes, namely the source *s,* and the sink *t.* The edges in the graph can be divided into two disjoint categories: t-edges which connect a node to a terminal node, and n-edges which connect nodes other than the terminal nodes with each other.

A cut of the graph is defined as the set of edges whose removal causes the nodes of the graph to be divided into two disjoint sets. For two nodes in the same set, there exists a path in the modified graph, whereas for two nodes from different sets, there exists no such path. An st-cut satisfies the property that $s \epsilon S$ and $t \epsilon \bar{S}$. Given a directed weighted graph *G,* the st-mincut problem is that of finding an st-cut with the smallest cost. Figure 1 shows an example graph formulation.

A. *Modelling Segmentation as a st-Mincut Problem*

We build a graph $G(V,E)$ with nodes corresponding to pixels $p \epsilon P$ of the image. In addition, there are two terminal nodes – source node S (corresponding to the foreground segment) and sink node T (corresponding to the background segment). The set of edges E consists of terminal-edges and non-terminal or neighbor edges. Each pixel has two terminal edges $\{p,S\}$ and $\{p,T\}$, connecting it to each terminal. Each pair of neighboring pixels is connected by a neighbor edge.

1) *Modelling Edge Weights*

a) *Non-terminal Edge Weights:*

The neighbor edges have to be assigned weights such that if the two corresponding pixels have similar intensities, the penalty of assigning them to different segments should be large, and vice versa. Similarly, this penalty decreases as the

distance between the neighboring pixels increases (this is of significance in a 26-neighbor system). These constraints are representing as the cost associated with non-terminal edges as follows:

$$B_{\{p,q\}} \;\alpha\; exp\left(-\frac{(I_p - I_q)^2}{2\sigma^2}\right) \cdot \frac{1}{dist(p,q)}$$

$$where\;\; \sigma = \sqrt{\mathbb{E}[(I_p - I_q)^2]}$$

$\mathbb{E}[x]$: $Expectation\ of\ x, given\ the\ image\ pixel\ set\ P$

$dist(p,q) = \ Distance\ between\ pixels\ p\ and\ q$

b) Terminal Edge Weights:

The terminal edge weights represent regional penalties and penalties for hard constraints. The regional constraint penalizes the segmentation if a pixel has low likelihood of belonging to the assigned segment. Similarly, the segmentation is heavily penalized if it segments the user defined seeds are labeled in a manner opposite to that intended by the user. The terminal edge weights, denoted by $R_{\{p,S\}}$ and $R_{\{p,T\}}$ are given by:

$$R_{\{p,S\}} = \begin{cases} -\lambda \ln \Pr\,(\mathbb{I}_p \mid \text{Bkg}) & p \in P, p \notin\ F \cup B \\ K & p \in F \\ 0 & p \in B \end{cases}$$

$$R_{\{p,T\}} = \begin{cases} -\lambda \ln \Pr\,(\mathbb{I}_p \mid \text{For}) & p \in P, p \notin\ F \cup B \\ K & p \in B \\ 0 & p \in F \end{cases}$$

where,

$K = 1 + \max_{p \in P} \sum_{q:\{p,q\} \in N} B_{\{p,q\}}$

$(-\ln \Pr\,(\mathbb{I}_p \mid \text{Bkg}))$ - Represents the negative log likelihoods of the pixel p belonging to the background.

$(-\ln \Pr\,(\mathbb{I}_p \mid \text{For}))$ - Represents the negative log likelihoods of the pixel p belonging to the foreground.

λ – Relative weight assigned to regional constraints with respect to boundary constraints.

B. *Assigning Edge Weights and Boundary Penalties*

We assign edge weights by learning the distribution of pixel intensities and RGB values of foreground and background seed points. We assign boundary penalties as a similarity measure with respect to eight neighbors. Assignments are made as discussed in Section III.A.

1) Modelling Pixel Intensity Distribution

Given user seed points, we learn a probability distribution each for background and foreground by building an intensity histogram for each set of seed points and normalizing it. We learn the color distribution in a similar fashion by using pixel RGB values to build channel specific normalized histograms. In this formulation we approximate the color distribution by treating each channel to be independent, and hence the joint color probability as the product of each individual channel. Since this might not model the color distribution accurately, we also use a Gaussian mixture model to learn the distribution by expectation maximization, on the pixel RGB values. See Section III.A.1)b) for the corresponding equations.

2) Assigning Boundary Penalties

We assign boundary penalty as a pixel similarity measure with respect to its neighbors. We compute the similarity measure in the space of pixel intensity and RGB values. Using pixel intensity and RGB values measures similarity in the appearance space. To model structural similarity we also extract a Histogram of Oriented Gradients descriptor[1] at each pixel. See Section III.A.1)a) for the corresponding equations.

Finally with the graphical model fully defined, we compute the st-Mincut by the Boykov-Kolmogorov max-flow algorithm[6], see Figure 2.

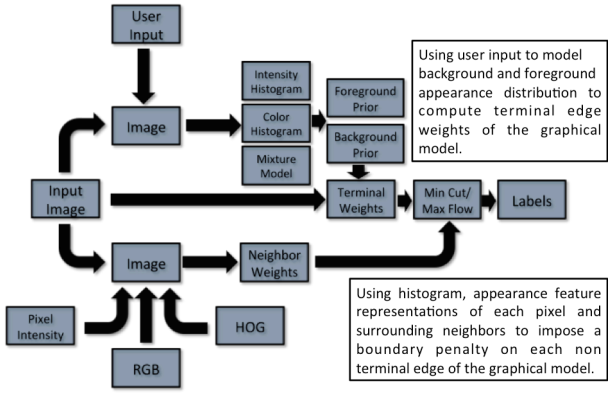

Figure 2 Method Flow Diagram

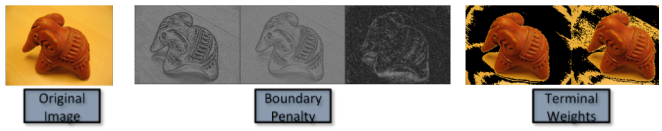

Figure 3 Visualizing Terminal Weights and Boundary Penalties. Note the structure based boundary penalty encoded on the rightmost image, generated using HoG descriptors.

IV. IMPLEMENTATION

We implemented most of our software using built-in matlab methods. However due to the complexity involved in building a graph cut implementation efficiently, we choose to use the implementation made available by [4]. We use the Netlab matlab toolbox[2] for implementing Gaussian mixture models, and training routines.

V. EXPERIMENTS

We setup each experiment with the user having to pick seed points for background and foreground. We then analyze the segmentation performance by varying each component in the prior and boundary penalty computation, see Figure 2

Table 1 shows all the configurations used and the assigned configuration number. We also analyze performance of the segmentation algorithm as the free parameter λ is varied, see Section III.A.1)b). Finally we make a qualitative comparison with results using the GrabCut segmentation method[3].

All our experiments are performed on a subset of the Berkley segmentation dataset with ground truth segmentation[5], see Figure 4 for a select few examples from the dataset.

Figure 4 Examples from the Berkley segmentation dataset[5], bottom row shows the ground truth segmentations.

Table 1 Terminal and Boundary Penalty Configuration

Configuration Number	*Terminal*	*Boundary Penalty*
1	*Intensity Histogram*	*Pixel Intensity*
2	*Color Histogram*	*Pixel Intensity*
3	*Mixture Model*	*Pixel Intensity*
4	*Intensity Histogram*	*RGB*
5	*Color Histogram*	*RGB*
6	*Mixture Model*	*RGB*
7	*Intensity Histogram*	*HOG*
8	*Color Histogram*	*HOG*
9	*Mixture Model*	*HOG*

VI. Results

In Section VI.A we discuss evaluation of each segmentation configuration using average precision, recall and segmentation error rate[6] as metrics. In Section VI.B we discuss results as a qualitative comparison with GrabCut[3].

A. Evaluation of Edge and Boundry Penalty Configurations

We evaluate each configuration on 50 images, with λ varying from being equal to K to K/9, which included some images without a unique foreground and background label assignment. We get the highest precision of 0.5963 with recall of 0.8231 and an error rate of 0.01881, using the configuration 5, with λ set at K/9. The configuration 4 with λ equal to K reported the lowest precision value of 0.3456, with a 0.7019 recall and an error rate of 0.03593. We note the dependence of method performance with different values of λ suggesting high dependence to the prior assignments, and a nontrivial choice in deciding the best configuration. Though results indicate a low precision rate when using the distribution modeled by a Gaussian mixture model we note that for some images this model computed the best segmentation, however since the segmentation result is not consistent, the average precision drops. See Figure 6 for the rest of the results.

Since many of the 50 images do not have a prominent foreground and background distinction, we crop the image set to exclude such instances, resulting in a remainder set of 20 images. We see overall improvement, but the configurations now, which uses a mixture model, now reports an impressive average precision of 0.8795 with recall of 0.8297 and reduction in the error rate to 0.0022, for the configuration 6, with λ at K/9. This result is interesting in that using a Gaussian mixture model has the potential to improve precision and recall drastically. See Figure 7 for the rest of the results. See Section III.B.2) for definitions of K and λ.

B. Qualitative comparision with GrabCut[3]

We make a qualitative comparison with the segmentation results of GrabCut[3], Figure 5 shows the best and the worst segmentation results of both approaches. We observe that in the presence of high variability in appearance both methods perform equally well, however with reduced variability we see a drop in precision of our method, while the GrabCut[3] method performs consistently well in most cases.

VII. Conclusion

We extended the seminal work of Boykov and Jolly[6] by modeling the RGB space and using the HOG descriptor to model neighbor similarity. We compare performance of these methods with GrabCut[3]. Though we do not beat their method in average precision rates, the performance is comparable and sometimes better for certain images.

Performance with foreground and background priors computed from RGB data enable more accurate segmentation, hence we can expect an improvement in performance with better modeling on the prior distribution. Use of HOG features to model boundary penalty does not improve segmentation, because HOG measures only structural similarity, hence we expect improvement with the use of features which better model neighborhood appearance.

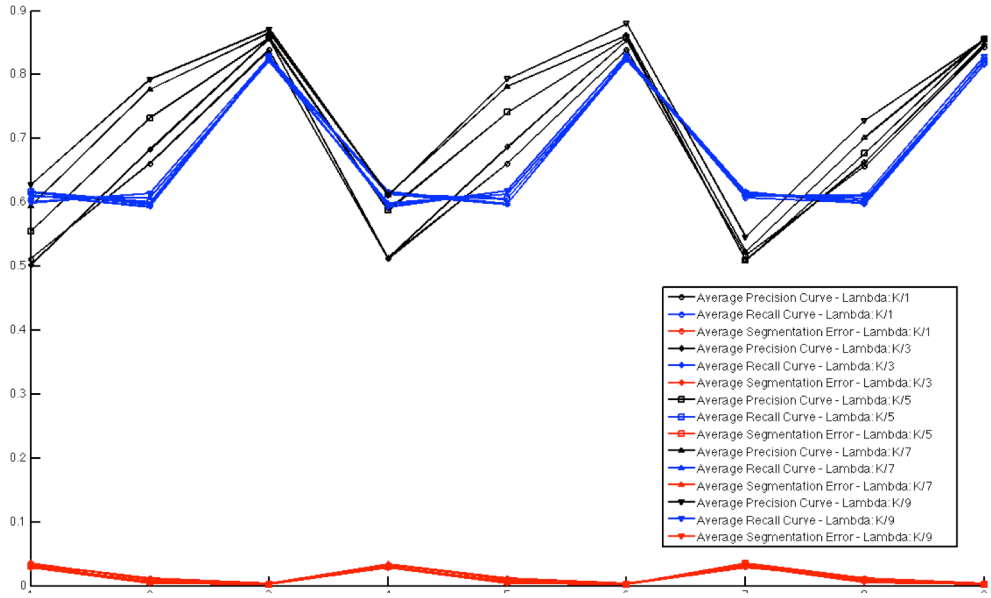

Figure 7 Precision Recall and Error Rate on select 20 Images.

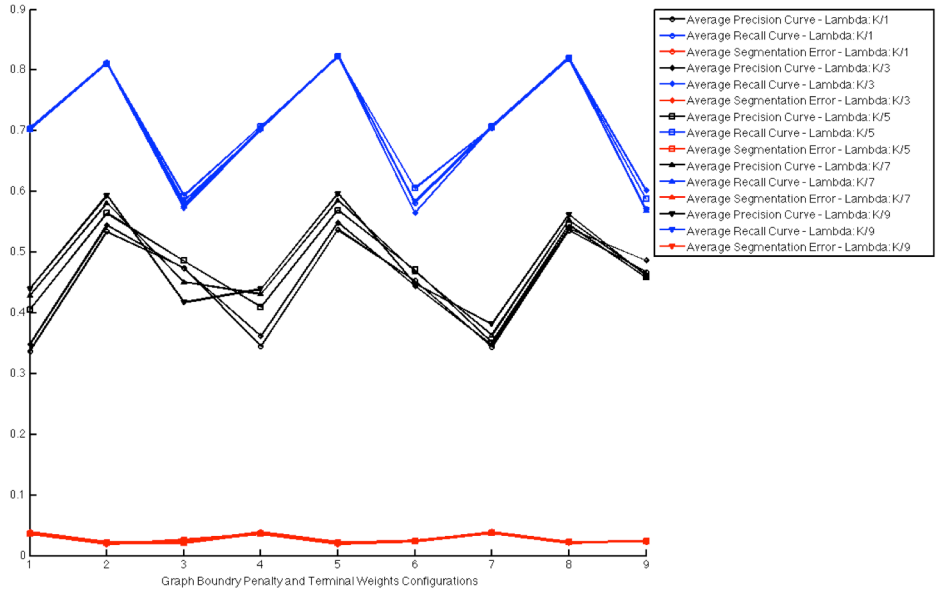

Figure 6 Precision Recall and Error Rate on 50 Images.

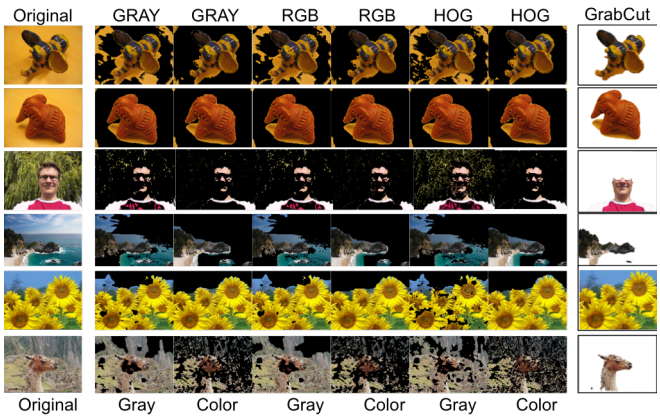

Figure 5 A sample set of segmentation results for a qualitative comparision.

We observe high model sensitivity to the changes in the prior, and parameters of the model. Hence the method is not robust and requires manual tuning in the present form.

VIII. Future Work

A definite extension is to investigate methods to accurately model color distributions of the user points. Evaluation of the method with features that capture neighborhood appearance more robustly with tunable overflow is another step forward.

In the present form, the method is not iterative, and we intend to improve results by utilizing corrective user inputs. Evaluation and comparison with other graph cut methods such as normalized cuts will be incorporated in future work. The method in the present form requires manual tuning of many of

the model parameters, we intend to automatically choose optimal parameters as future work. Finally we would like to extend the method to be less dependent on the prior and incorporate learning methods to proceed towards automatic image segmentation.

REFERENCES

[1] Dalal, Navneet, and Bill Triggs. "Histograms of oriented gradients for human detection." Computer Vision and Pattern Recognition, 2005. CVPR 2005. IEEE Computer Society Conference on. Vol. 1. IEEE, 2005.

[2] Nabney, Ian T. NETLAB: algorithms for pattern recognition. Springer, 2004.

[3] Rother, Carsten, Vladimir Kolmogorov, and Andrew Blake. "Grabcut: Interactive foreground extraction using iterated graph cuts." ACM Transactions on Graphics (TOG). Vol. 23. No. 3. ACM, 2004.

[4] Delong, Andrew, and Yuri Boykov. "A scalable graph-cut algorithm for ND grids." Computer Vision and Pattern Recognition, 2008. CVPR 2008. IEEE Conference on. IEEE, 2008.

[5] Arbelaez, Pablo, Charless Fowlkes, and David Martin. "The berkeley segmentation dataset and benchmark." see http://www. eecs. berkeley. edu/Research/Projects/CS/vision/bsds (2007).

[6] Boykov, Yuri Y., and M-P. Jolly. "Interactive graph cuts for optimal boundary & region segmentation of objects in ND images." Computer Vision, 2001. ICCV 2001. Proceedings. Eighth IEEE International Conference on. Vol. 1. IEEE, 2001.

[7] Mortensen, Eric N., and William A. Barrett. "Intelligent scissors for image composition." *Proceedings of the 22nd annual conference on Computer graphics and interactive techniques*. ACM, 1995.

[8] Adobe Systems Incorp. Adobe Photoshop User Guide. 2002.

[9] Caselles, Vicent, Ron Kimmel, and Guillermo Sapiro. "Geodesic active contours." *International journal of computer vision* 22.1 (1997): 61-79.

[10] Z. Wu and R. Leahy. An optimal graph theoretic approach to data clustering: Theory and its application to image segmentation

[11] Shi, Jianbo, and Jitendra Malik. "Normalized cuts and image segmentation."Pattern Analysis and Machine Intelligence, IEEE Transactions on 22.8 (2000): 888-905.

[12] Felzenszwalb, Pedro F., and Daniel P. Huttenlocher. "Efficient graph-based image segmentation." International Journal of Computer Vision 59.2 (2004): 167-181.

[13] Boykov, Yuri, and Vladimir Kolmogorov. "Computing geodesics and minimal surfaces via graph cuts." *Computer Vision, 2003. Proceedings. Ninth IEEE International Conference on*. IEEE, 2003

[14] Freedman, Daniel, and Tao Zhang. "Interactive graph cut based segmentation with shape priors." *Computer Vision and Pattern Recognition, 2005. CVPR 2005. IEEE Computer Society Conference on*. Vol. 1. IEEE, 2005

[15] Blake, Andrew, et al. "Interactive image segmentation using an adaptive GMMRF model." *Computer Vision-ECCV 2004* (2004): 428-441.