Markov Random Fields for Binary Image Segmentation

Chris Whiten

University of Ottawa

ABSTRACT

Throughout the scope of this project, the application of using Markov Random Field representations of images for binary image segmentation is explored. Specifically, graph cuts and loopy belief propagation are explained, and a comparison of several energy minimization techniques is given. Additionally, a C++ implementation of graph cuts segmentation was developed for experimentation purposes.

1. INTRODUCTION

One of the most fundamental problems in image processing is that of image segmentation. Image segmentation is viewed as the act of separating an image into its constituent parts. More generally, it can be viewed as a transformation from an entire image region $f[x]$ to a set of sub-regions, $G = \bigcup_i g_i[x]$, representing the different semantic pieces of an image. What each of the members of G represents is usually dependent on the intended application of the segmentation. Alternatively, it can be viewed as the act of assigning a label to every pixel, where the label is a representative of each segmentation region.

One common[1–4] way of segmenting an image is through energy minimization over a Markov random field. By exploiting the smoothness tendencies of natural images, the transformation from a matrix of pixels to a Markov random field with intuitive energy functions becomes very simple, as is shown in this report.

2. IMAGE SEGMENTATION

A perfect solution to the image segmentation problem is often viewed as one of the "holy grails" of image analysis. A perfect segmentation would permit for more efficient image processing algorithms, and could be used as input to higher level computer vision algorithms. A good segmentation of an image would help the object recognition problem by reducing it to that of silhouette matching, or it could be used for perfect background subtraction in tracking applications.

Nevertheless, a perfect, general-purpose, segmentation algorithm has yet to be achieved. However, many applications still make use of segmentation algorithms to this day. This is achieved by limiting assumptions or user interaction. For example, foreground/background segmentation is achieved in surveillance applications by analyzing multiple successive image frames to "learn" what pixels tend to be relatively stationary (background) versus what pixels belong to a free-moving object (foreground). In still-images, the most common method of making segmentation tractable is through user-interaction, often dubbed "interactive segmentation".

In contrast to automatic segmentation, interactive segmentation achieves performance gains through the incorporation of knowledge through user input. In interactive segmentation, a user selects pixels from each segmentation class, which provides the algorithm with posterior knowledge from which to perform a more accurate segmentation. The most common case of interactive segmentation, and the case explored in this report, is that where a user selects "strokes" of background and foreground pixels to perform binary segmentation. A stroke is a contiguous set of pixels, usually passed as input through a mouse click-and-drag. An example of such input is shown in Figure 1. By providing these strokes, a segmentation algorithm can build a histogram of pixel intensities and normalize to retrieve a posterior probability distribution of the segmentation class in question. This approach is used in the graph cuts algorithm, discused in Section 4.

cwhit025@uottawa.ca

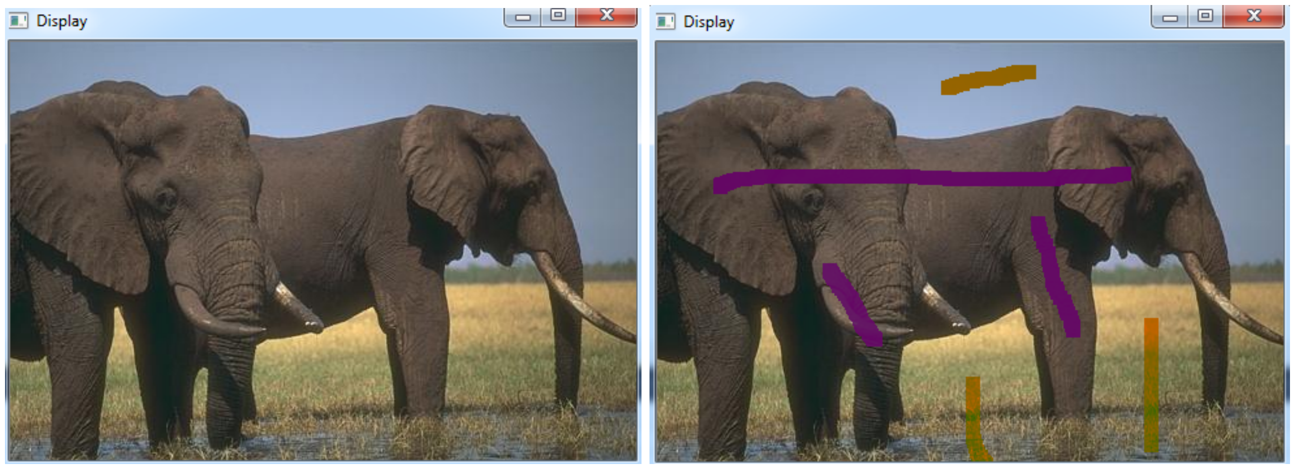

Figure 1. On the left is an input image. On the right is that same image, marked up by a user to provide knowledge of the foreground and background of the image.

3. MARKOV RANDOM FIELDS FOR SEGMENTATION

In the problem of binary image segmentation, it is logical to look at a formulation that permits assignments to one of two classes, and conveys a prior over the image that accounts for spatial correlation between pixels. One formulation that encodes such relationships is a Markov Random Field with the Ising model. For an assignment $\mathcal{A}$ over an image, the Ising prior states that $Pr(\mathcal{A}) \propto \exp(-V)$, where $V = \sum_{m,n \in \mathcal{N}, \mathcal{A}_n \neq \mathcal{A}_m} \gamma(\mathcal{A}_m, \mathcal{A}_n)$. In other words, the probability of any given assignment is proportional to some potential function, γ, over all pairs of neighbouring pixels that do not share the same assignment. γ encodes the strength of the spatial interaction between the two neighbouring pixels.

For the interactive binary segmentation problem, we aim to find the most likely assignment for each pixel in our Markov random field, given some evidence (user input). It has been shown[4] that this is equivalent to minimizing the energy over such a structure. According to Szeliski et al,[4] the majority of the top-performing energy minimization tasks in image processing rely on graph cuts or loopy belief propagation techniques. As such, these two techniques are the central focus of this report. A comparison of these two techniques has been previously considered by Tappen and Freeman,[5] but the compared application domain is solely stereo matching.

3.1 Formulation as an Energy Minimization Problem

It is useful to note that the image segmentation problem can be viewed as a sub-class of the broader "pixel labeling" problem. That is, for a set of pixels $p \in \mathcal{P}$, we aim to assign the most likely labeling $l \in \mathcal{L}$. In the binary image segmentation problem, this label set can be denoted $\mathcal{L} = \{l_{fg}, l_{bg}\}$. It is reasonable to assume that natural images should have labelings that are consistently smooth along an image, with discontinuities only at the segmentation boundaries. We encode this smoothness assumption, as well as our prior knowledge about a segmentation, as a function of the energy $\mathcal{E}$ over an image, where $\mathcal{E}$ is a linear combination of a "boundary smoothness" $\mathcal{B}_{p,q}$ over each pair of neighbouring pixels p, q, and a "region energy" $\mathcal{R}_p$ that encodes the energy assumed to be present for a single pixel p when assigned to a specific label $l \in \mathcal{L}$. For some weighting λ, we denote $\mathcal{E} = \lambda \sum_{(p,q) \in \mathcal{N}} \mathcal{B}_{p,q} + \sum_{p \in \mathcal{P}} \mathcal{R}_p$. The solution to the segmentation problem is assumed to be the joint assignment of pixels to labels that minimizes the energy $\mathcal{E}$. We explore two approaches for minimizing this energy, graph cuts and loopy belief propagation.

4. GRAPH CUTS

Graph cuts are a modern technique for energy minimization that are prevalent throughout the image processing literature. As such, I chose to implement this technique for the implementation portion of this report. Graph

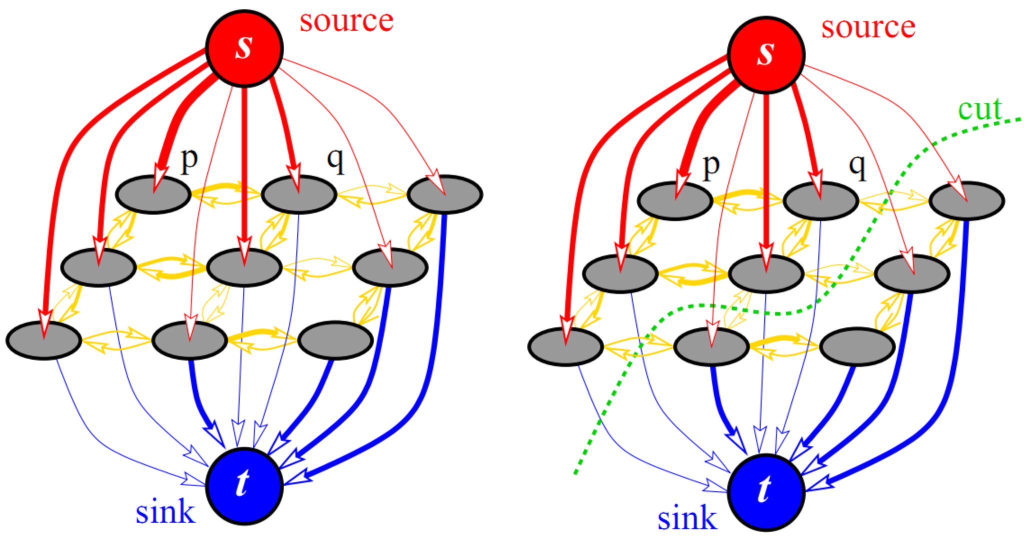

Figure 2. *Left:* An image represented as a graph for performing graph cuts. The source and sink nodes correspond to the background and foreground, respectively. Each pixel is connected to each labeling, as well as its neighbours. *Right:* A possible cut over this graph.[3]

cuts segmentation is appealing because it has both good complexity and good accuracy guarantees. Beyond this, it is also a simple algorithm that naturally decomposes a Markov random field representation of an image into a well connected graph with terminal nodes representing each of the image regions, where a partitioning of the graph corresponds to a segmentation of the image.

The transformed representation of an image for graph cuts is a graph $G = (V, E)$, where the vertices V are a set of nodes corresponding to the pixels in an image, V_p, and nodes that are representatives of each possible labeling of a pixel, called *terminal nodes*, V_t. $V = \{V_p \cup V_t\}$. In the case of binary segmentation, these terminal nodes may represent the foreground and the background. The edges E are comprised of a set of edges connecting neighbouring (adjacent) pixels, E_n, and an edge from every pixel to every terminal node, E_t, representing the corresponding pixel's affinity to each possible labeling. $E = \{E_n \cup E_t\}$. The graphical structure is illustrated in Figure 2. The weights on these edges correspond to the energies defined by a Markov random field formulation, which is algorithm dependent. We explore one such formulation in section 4.2.

Given the constructed graph G, we can define a cut C over G as a subset of edges that, when removed, separates each of the terminal nodes from each other, such that there no longer exists a path between any two terminal nodes. This cut graph is denoted $G' = (V, E \backslash C)$ and is visualized in Figure 2. We define the cost of a cut to be $|C| = \sum\limits_{c \in C} w_c$, where w_c is the weight of the edge c. In graph cuts, our interest is in finding the minimum cost cut. In the context of image segmentation, a minimum cost cut over G corresponds to a MAP (maximum a posteriori) assignment of pixels to labels.

4.1 Computational Complexity

As outlined earlier, one appeal of graph cuts is computational tractability. The decomposition from an image into the graphical representation can be done in linear time. The transformation of each pixel on a finite-sized rectangular lattice into a node in the graph is linear in the number of pixels, $O(mn)$. Each of these nodes computes pairwise clique energies within its neighbourhood of 8 pixels, resulting in $O(8mn) = O(mn)$ computations. Finally, the computation of singleton-clique energies is a simple histogram lookup, which is a constant time operation. The overall complexity of building this structure is the sum of these listed complexities, which is still linear.

Comptuation of the minimum cut across this graphical representation is still a polynomial time operation. Many possible algorithms for computing this cut have been investigated[6] with specific application to images.

Pixel type	$Pr(I_p = l_{bg})$	$Pr(I_p = l_{fg})$
Background seed pixel	1	0
Foreground seed pixel	0	1
Unmarked pixel	$Pr(I_p = l_{bg}\|B)$	$Pr(I_p = l_{fg}\|F)$

Figure 3. The probability of a given pixel belonging to the foreground or background class.

This investigation has found that the most experimentally efficient algorithm on images is not the algorithm with the best theoretical guarantees. The algorithm of choice has a computational complexity of $O(mn^2|C|)$, where $|C|$ is the cost of the minimum cut. Although not theoretically optimal, this is still tractable (polynomial), which is a desirable trait for segmentation.

4.2 Interactive Graph Cuts for Binary Segmetation

4.2.1 Implementation Usage

The implementation accompanying this report is based on the approach presented by Boykov and Jolly[3] for performing foreground/background segmentation with user-guided graph cuts. The use case with this implementation is that the user "seeds" the graph cuts algorithm with some foreground and background areas, with mouse clicks, and then the "enter" key is pressed on the keyboard to perform the segmentation. Figure 7 showcases an image with pixel strokes for foreground and background areas (selected by an active user), and the resultant segmentation.

4.2.2 Algorithm Details

As this algorithm falls directly into the family of graph cuts algorithms, the majority of the algorithm is very straight forward. We formulate a graph G based on the image, and calculate the minimum cost cut on this graph. The relevant details to fill in are the clique energies, that persist as the weights of the edges in the Markov random field formulation, the neighbourhood system, and other minor implementation details.

We first consider pairwise clique energies. Using an 8-neighbourhood Markov random field, a pairwise clique consists of two adjacent pixels. Since we are ultimately cutting our graph across minimum costs, we want to encourage adjacent pixels with similar intensities to belong to the same side of the cut (implying that they share the same labeling, foreground or background). We enforce this by allocating high energy between adjacent pixels with similar intensity, and lower energy as adjacent pixels become less similar. We denote the pairwise clique energy between pixels p and q as the "boundary energy", $B_{p,q} = \exp(\frac{-(I_p - I_q)^2}{2\sigma^2})$. I_p and I_q are the intensity values for respective pixels p and q, while σ can be thought of as the estimated camera noise. This fomulation penalizes for discontinuities between pixels of similar intensities, when $|I_p - I_q| < \sigma$, encouraging the intuition that adjacent pixels that are visually similar are likely to belong to the same object.

Next, we consider the singleton clique energies. These are proportional to the prior probability that a given pixel belongs to the foreground or background. In our interactive segmentation example, these prior probabilities are augmented to become the posterior probability of a pixel being background or foreground, given the user-provided foreground and background seed pixels. Each pixel in the MRF holds one of three potential probability distributions over the probability of being foreground or background. These probability distributions are outlined in Figure 3. B and F correspond to the empirical statistics gathered of background and foreground pixels, respectively, from the user-provided seed pixels. Each of the pixels marked as background (respectively, foreground) are collected and binned into a histogram. At the end, these are normalized to give a valid probability distribution, $Pr(I_p = l_{bg}|B)$ (respectively, $Pr(I_p = l_{fg}|F)$).

We model these singleton energies by adding a terminal node to the graphical model for each label $l \in \mathcal{L}$, and connecting all pixel nodes to all of the terminal nodes. The connection weights on these new edges will be the singleton clique energies, which are proportional to (but not precisely) the previously described probability distribtutions. This energy, which we denote as $R_p(I_p)$, the "region" term of pixel I_p, is described in Figure 4. This energy function contains a term K that is meant to put an expensive cost on separating a marked pixel from its terminal node. That is because we have prior knowledge that the pixel in question almost certainly

Pixel type	Background energy	Foreground energy
Background seed pixel	K	0
Foreground seed pixel	0	K
Unmarked pixel	$-\ln(Pr(I_p = bg\|B))$	$-\ln(Pr(I_p = fg\|F))$

Figure 4. The energy cost allocated to the weight on the edge between node I_p and the foreground and background terminal nodes.

belongs to the segmentation labeling represented by that terminal node (with the sole exception being a mistake made in the user-input phase). This value of K is equal to 1 + the sum of the largest neighbourhood in the entire image, $K = 1 + \arg\max_{p\in\mathcal{P}} \sum_{q:(p,q)\in\mathcal{N}} B_{p,q}$.

Once the energy costs are set, our graphical model is entirely described and we can run the minimum cost cut algorithm of our choice[6] to compute a binary segmentation of the input image.

5. BELIEF PROPAGATION

Loopy belief propagation algorithms generally come in two flavours, the sum-product and the max-sum algorithm. Both of these algorithms operate by passing messages around a graph constructed from an image realized as a Markov random field.[7] The key difference between the two is how the passed messages are computed, which results in a different piece of information being output. Sum-product belief propagation results in the marginal probabilities, which are unimportant for the application of segmentation. Max-sum belief propagation outputs the maximum a posteriori estimate, which is analogous to minimizing the energy over our problem formulation. Thus, we narrow our focus to max-sum LBP.

In max-sum LBP, we pass messages around the MRF representing our image, where the messages are a probability distribution over the possible labelings $l \in \mathcal{L}$ of the current pixel. With a labelling denoted $\mathcal{P}_{\mathcal{L}}$, a message from pixel p to q at iteration i is denoted $m^i_{pq} = \arg\min_{\mathcal{P}_{\mathcal{L}}}(\mathcal{B}_{p,q} + \mathcal{R}_p + \sum_{s\in\mathcal{N}(p)\backslash q} m^{i-1}_{sp})$. Put simply, the message is value over the labeling that minimizes the sum of: the boundary cost between the two interacting pixels, the region cost of the source pixel, and the sum of all other messages coming into the source pixel, except for the message from the destination pixel.

These messages are passed continuously, as it may take many iterations of the message passing algorithm to converge. In fact, it is often the case that convergence is never achieved, in which case it is wise to stop the process after some set number of iterations. At the end of the process, we compute the "beliefs" of each pixel, which correspond to the probability distribution over all possible labelings. The beliefs are computed as $belief(p) = \mathcal{R}_p + \sum_{q\in\mathcal{N}(p)} m^i_{qp}$. We then select the label $l \in \mathcal{L}$ from $belief(p)$ that has the maximal probability, and use that as the MAP solution to the segmentation problem.

Belief propagation forms the basis for several segmentation algorithms. Beyond binary segmentation, it has also been applied to segmentations with mattes,[8] where a pixel is not necessarily exclusive to foreground or background, but has some alpha opacity mixture between the two classes. This is often referred to as soft segmentation. Alternatively, belief propagation has been used[9] for hard segmentation in both the binary and n-label scenario.

6. EXPERIMENTS AND COMPARISONS

Due to the nature of this report, the experiments are separated into two sections. First, we evaluate experiments performed on the implementation accompanying this report, discussing potential shortcomings. Then, we discuss an evaluation performed over the state of the art, as outlined in the studied[4] comparison paper by Szeliski et al.

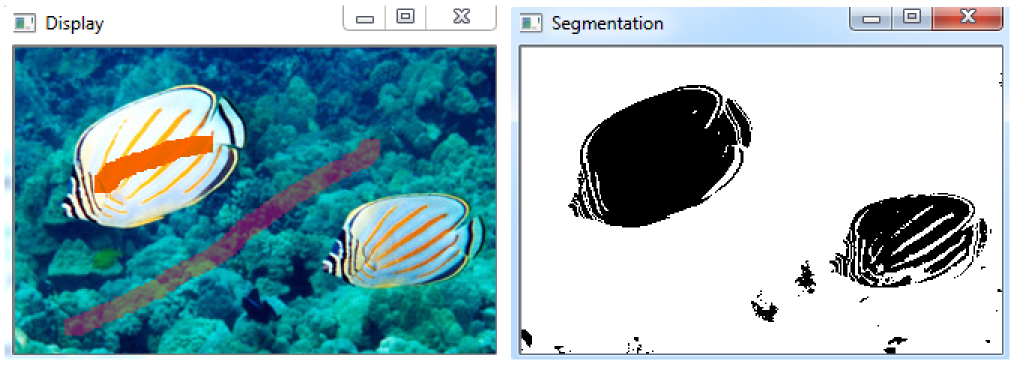

Figure 5. The segmentation of two fish, as performed by the implementation accompanying this report.

6.1 Implementation Details and Experiments

The implementation accompanying this report makes use of available public domain code from two main sources. The code was written in C++ with OpenCV for image processing functionality, though the only image processing function in the final product is a function that converts an RGB image to its grayscale equivalent. Besides that, its main use was loading an image file as a matrix, and utilizing the HighGUI user interface functions for quick mockup UIs. The second piece of public domain code was a min-cut/max-flow implementation provided by Yuri Boykov and Vladimir Kolmogorov.[10] This allowed for less time debugging edge cases in the min-cut algorithm for graph cuts. Previous iterations of the report used C++'s Boost framework, but this dependency has been removed.

The implementation is based on Boykov and Jolly's[3] paper on interactive region segmentation with graph cuts. As can be seen in Figure 7, the segmentation with minimal user input performs reasonably well, but not perfectly. Tuning different parameters can greatly affect the results of a segmentation, but the aim is to showcase the strengths and weaknesses of the algorithm within the same settings. The segmentation in Figure 7 performs well due to the clear correlation between all of the foreground pixels, and all of the background pixels. There is very little noise in the image in the way of highly varying pixel intensities belonging to the same segmentation labeling. In contrast, Figure 6 displays a very poor segmentation.

The poor segmentation is relatively easy to explain. When collecting foreground and background pixels, we are placing them into a finite-sized histogram (in this case, with 8 bins). By binning the pixels, we are effectively quantizing our pixel space into 8 possibilities, and building a probability distribution over these possibilities. When building the probability distribution over the foreground region, the black shirt heavily dominates the distribution, skewing the data to believe that all foreground pixels are very likely to be black. Meanwhile, at the edge contours of a large majority of the image (including the background), we have very dark colours leading to a high probability of that pixel belonging to the foreground. This, in turn, affects the pixels around each of the edge contours, due to the smoothness constraint. This creates very conflicting distributions and leads to a poor segmentation. It is possible to partially alleviate this by testing different values for the relative weighting between the region and smoothness energies, but the problem does not entirely disappear. A possible solution to this problem would be to smoothen the histogram representing each of the distributions, so they are not dominated by something like a solid black shirt. Alternatively, more histogram bins may also help.

6.2 Comparing Energy Minimization Methods for Markov Random Fields

The investigated paper[4] compares the run-times and performance of several methods for performing energy minimization on MRFs for image processing tasks. These include tasks such as image segmentation, stereo matching, and image stitching. Narrowing the focus to image segmentation (as per the scope of this report), it is clear that there are methods that out-perform others. For example, it has been shown[11] that the global minimum energy can be computed in the binary segmentation case with minimum cost cut methods, like graph cuts.

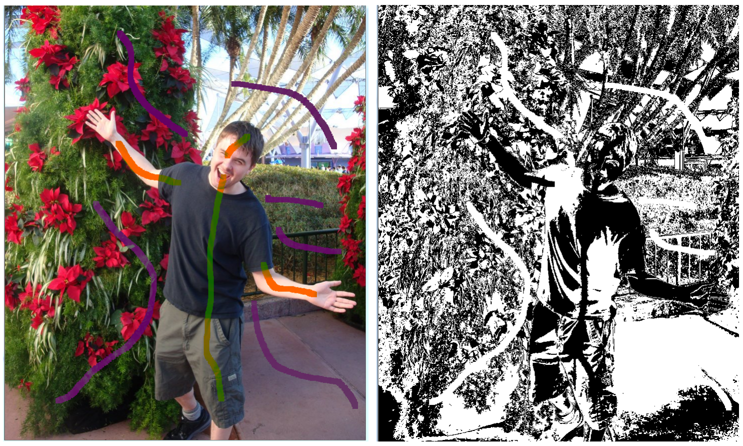

Figure 6. An example of an image that is poorly segmented by the accompanying implementation.

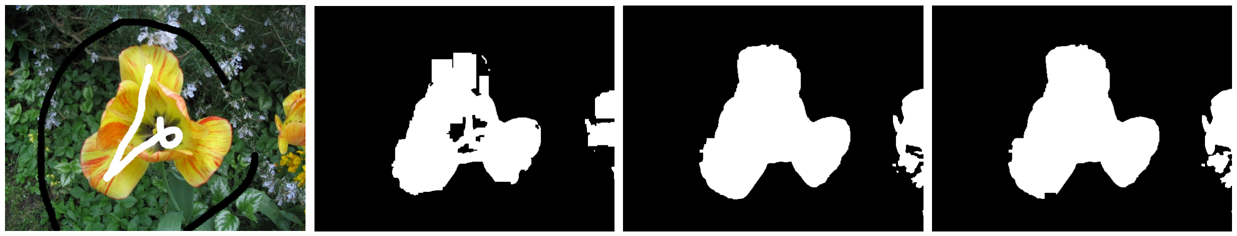

Figure 7. Segmentation input and output for a sample experiment[4] evaluating different methods of energy minimization. The first image is the input image with user-added strokes for foreground and background pixels. The remaining images are the output segmentation from: iterated conditional modes, loopy belief propagation, and graph cuts.

Although the implementation accompanying this report is a simple graph cuts implementation, the graph cuts algorithms explored in this paper are slightly more sophisticated variants that have shown greater performance. These variants are called the $\alpha - \beta$ swap-move algorithm and the expansion-move algorithm. In contrast to the standard graph-cuts approach, these algorithms work over several iterations, making local modifications over image regions to produce lower energy labelings. Other approaches tested include iterated conditional modes, max-product loopy belief propagation, tree-reweighted message passing, and loopy belief propagation derived from tree-reweighted message passing.

The first comparison was performed over an image of a flower, as seen in Figure 7, along with a select few sample segmentations. In this example, iterated conditional modes appears to be far-and-away the poorest performing algorithm. As we can see in Figure 8, all methods besides ICM converge to minimum energies of approximately 100% of the actual energy. We also see note that graph cuts converge almost immediately to the global minimum, showcasing both its computational speed and performance. As we zoom in on the graphs, we note that the belief propagation algorithms both take a long time to converge, relative to other techniques, and do not achieve the global minimum in energy (although it is quite close, at approximately 100.04%). As an aside, ICM is a considerably older algorithm than the other techniques evaluated, showcasing the progress made in recent years on the segmentation problem. Similar results are seen with other images, as seen in Figure 9 and Figure 10.

In class, another older approach, simulated annealing, was presented as an energy minimization method over MRFs. We, once again, view it outperformed by more modern methods in Figure 11, where the running time and energy achieved by graph cuts are plotted against simulated annealing.

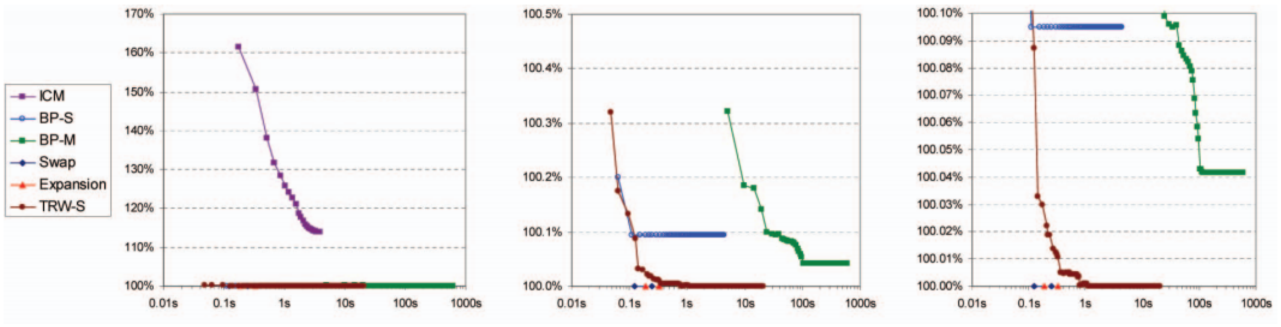

Figure 8. A comparison of different energy minimization techniques on binary segmentation over the sample flower image in Figure 7. From left to right, the graphs zoom in on the results to show performance differences at higher granularities. On the x-axis is the running time of the algorithm (on a logarithmic scale), and on the y-axis is the achieved energy, relative to the global minimum over the image.

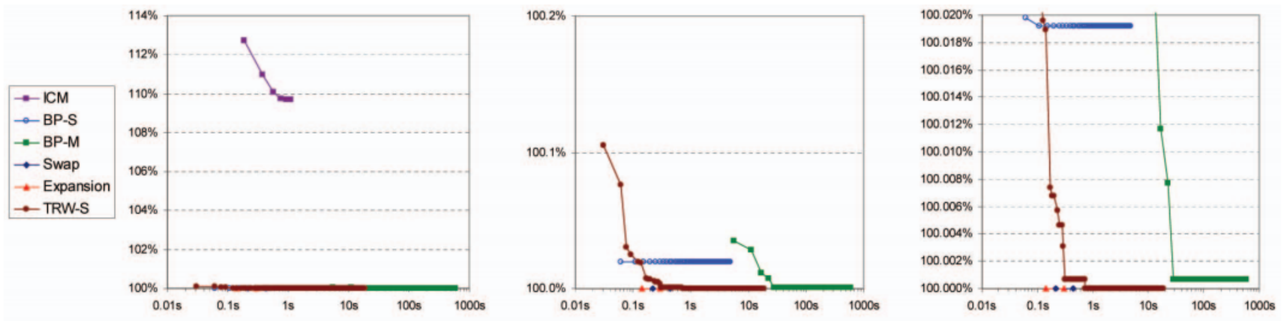

Figure 9. A comparison of different energy minimization techniques on binary segmentation over an image of a sponge. The format of these graphs is identical to Figure 8.

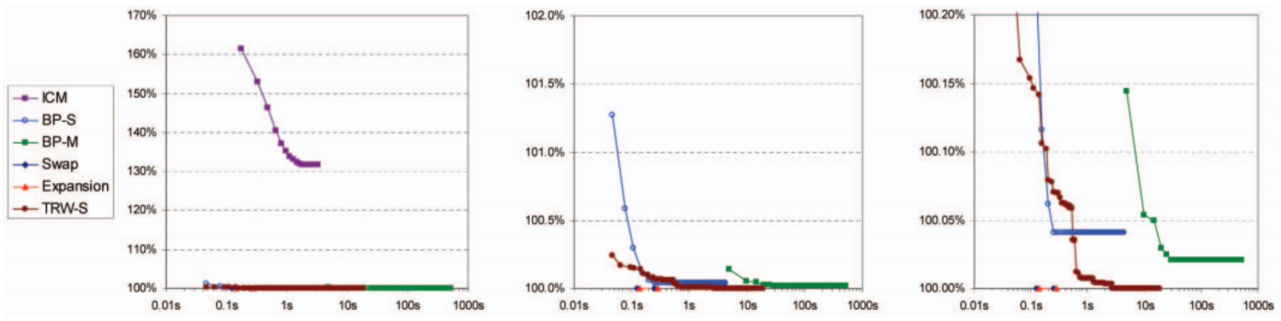

Figure 10. A comparison of different energy minimization techniques on binary segmentation over an image of a person. The format of these graphs is identical to Figure 8.

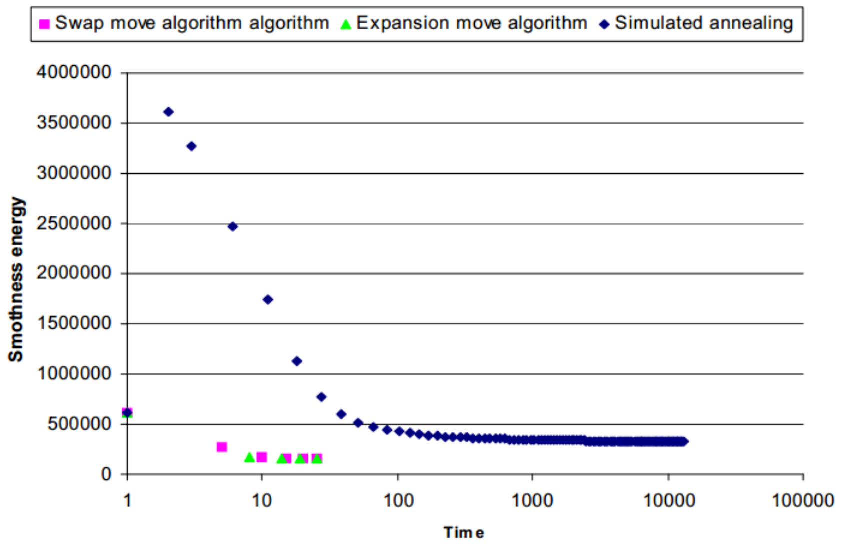

Figure 11. A comparison of simulated annealing to graph cuts for energy minimization. From this figure, it is clear that graph cuts both achieve a lower energy and converge in a shorter period of time.[12]

7. CONCLUSION

Throughout this project, I had the opportunity to review the current literature on using Markov random fields for image segmentation, and to implement one of such approaches and test it against natural images. An understanding of image segmentation is useful for both image processing and higher-level computer vision tasks, as the information gained from the segmentation of an image can be useful in improving the results of other algorithms. I have briefly introduced graph cuts and loopy belief propagation in relation to image segmentation, and presented comparison results of these algorithms against other popular energy minimization methods.

REFERENCES

1. Z. Wu and R. Leahy, "An optimal graph theoretic approach to data clustering: Theory and its application to image segmentation," *IEEE Transactions on Pattern Analysis and Machine Intelligence* **15**, pp. 1101–1113, 1993.
2. O. Veksler, "Image segmentation by nested cuts," in *In IEEE Conference on Computer Vision and Pattern Recognition*, pp. 339–344, 2000.
3. Y. Boykov and M.-P. Jolly, "Interactive graph cuts for optimal boundary and region segmentation of objects in n-d images," in *ICCV*, pp. 105–112, 2001.
4. R. Szeliski, R. Zabih, D. Scharstein, O. Veksler, V. Kolmogorov, A. Agarwala, M. F. Tappen, and C. Rother, "A comparative study of energy minimization methods for markov random fields with smoothness-based priors," *IEEE Trans. Pattern Anal. Mach. Intell.* **30**(6), pp. 1068–1080, 2008.
5. M. F. Tappen and W. T. Freeman, "Comparison of graph cuts with belief propagation for stereo, using identical mrf parameters," in *ICCV*, pp. 900–907, IEEE Computer Society, 2003.
6. Y. Boykov and V. Kolmogorov, "An experimental comparison of min-cut/max-flow algorithms for energy minimization in vision," *IEEE Trans. Pattern Anal. Mach. Intell.* **26**, pp. 1124–1137, Sept. 2004.
7. P. F. Felzenszwalb and D. P. Huttenlocher, "Efficient belief propagation for early vision," *International Journal of Computer Vision* **70**(1), pp. 41–54, 2006.
8. J. Wang and M. F. Cohen, "An iterative optimization approach for unified image segmentation and matting," in *Proceedings of the Tenth IEEE International Conference on Computer Vision - Volume 2, ICCV '05*, pp. 936–943, IEEE Computer Society, (Washington, DC, USA), 2005.

9. N. Shental, A. Zomet, T. Hertz, and Y. Weiss, "Learning and inferring image segmentations using the gbp typical cut algorithm," in *Proceedings of the Ninth IEEE International Conference on Computer Vision - Volume 2, ICCV '03*, pp. 1243–, IEEE Computer Society, (Washington, DC, USA), 2003.
10. Y. Boykov and V. Kolmogorov, "An experimental comparison of min-cut/max-flow algorithms for energy minimization in vision," *IEEE Trans. Pattern Anal. Mach. Intell.* **26**(9), pp. 1124–1137, 2004.
11. D. M. Greig, B. T. Porteous, and A. H. Seheult, "Exact maximum a posteriori estimation for binary images," *Journal of the Royal Statistical Society Series B Methodological* **51**(2), pp. 271–279, 1989.
12. Y. Boykov, O. Veksler, and R. Zabih, "Fast approximate energy minimization via graph cuts," *IEEE Trans. Pattern Anal. Mach. Intell.* **23**(11), pp. 1222–1239, 2001.