Deformable Object Shape Refinement and Tracking Using Graph Cuts and Support Vector Machines

Mehmet Kemal Kocamaz, Yan Lu, and Christopher Rasmussen

Department of Computer and Information Sciences
University of Delaware, Newark, DE, U.S.A.
{kocamaz,yanlu,cer}@udel.edu

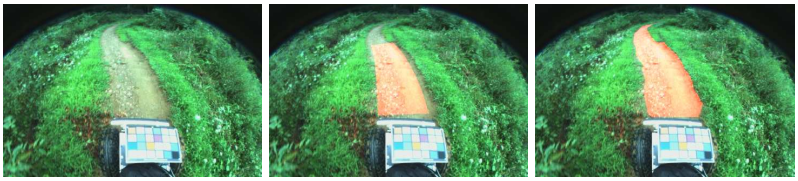
Abstract. This paper describes several approaches to the problem of obtaining a refined segmentation of an object given a coarse initial segmentation of it. One line of investigation modifies the standard graph cut method by incorporating color and shape distance terms, adaptively weighted at run time to try to favor the most informative cue given visual conditions. We also discuss a machine learning approach based on support vector machines which uses color and spatial features as well. Furthermore, we extend these single-frame refinement methods to serve as the basis of trackers which work for a variety of object types with complex, deformable shapes. Comparative results are presented for several diverse datasets including objects such as trail regions used for robot navigation, hands, and faces.

1 INTRODUCTION

In this paper, we describe a form of shape estimation in which an initial, rough shape is *refined* automatically, without user interaction. The contribution of this work is a comparison of several algorithms for this purpose based on graph cuts [1] and graph cuts with distance maps [2], as well as support vector machines (SVMs) [3], and a demonstration of their efficacy on several diverse datasets. These techniques work on individual images for which rough object segmentations are available, but we have also extended them so that they may run in a self-sustaining fashion over a video sequence given an initial indication of the object location.

An early motivation for this work came from a robotic application, *trail following*. Trails are man-made or natural visual features such as roads, hiking tracks, above-ground pipelines, rivers, and powerlines which can be navigationally exploited by unmanned vehicles. While the trail following problem may be considered as a form of road following [4–6], several factors make this task often harder, including indistinct borders, illumination changes, abrupt elevation changes, texture variety, and dead ends and forks.

In [7, 8], a trail tracking algorithm was presented which takes into account primarily the color contrast difference between the trail and neighboring image



(a) (b) (c)

Fig. 1. (a) Sample trail image; (b) Coarse ground truth overlaid; (c) Detailed ground truth

regions. While the method works well in many instances, one shortcoming is that it maintains a low-dimensional representation of the trail shape. In [7] the trail was represented strictly in the image domain as a triangle (to account for perspective), and in [8] the trail was represented as a circular arc in vehicle coordinates, projected to an omnidirectional image. These representations are necessarily approximate, and thus can miss important border details and possible in-trail obstacles (see Fig. 1 for a sample of the difference between a coarse and detailed trail segmentation).

Although the focus in this paper is on trail image data most relevant to mobile robot applications, we believe that the problem of automatically refining segmentations is a general one, and we will present results for images and videos of other kinds of objects with coarse segmentations. Graph cuts [1] are a popular and powerful computer vision technique which offer a potential solution to the problem of taking an approximate or coarse object segmentation and refining it to obtain more detailed and accurate object boundaries. User interactions are often required for best results [9], but automatically obtained coarse regions have been successfully used to seed the foreground/background regions needed for graph cut [10]. Work in this area has included using motion detection [11] and superpixel-based saliency measures [12] to obtain seed regions. Graph cut approaches for tracking objects have been described in [2, 13, 14].

In this paper, we describe a novel graph cut-based algorithm to refine an object's estimated borders if an initial coarse estimate is given. The standard graph cut method [1] uses only intensity information in its formulation. Intensity alone is often not enough to segment or track objects with diverse appearance and shape characteristics in a large range of images. Therefore, we made several changes in the standard graph cut method to increase the accuracy of the refinement task. First, an adaptive color space selection mechanism is employed in the algorithm. Second, color information in the image is clustered by k-means. The knowledge gathered from the clusters are included in the formulation of the region and boundary terms of the graph cut. Third, the spatial distance information is incorporated into the algorithm to constrain the final segmentation to stay around the region of interest. The influence of the distance penalty information is adaptively set by the algorithm.

For comparison, we also looked at applying SVMs to the same problem to learn a model of the object using both appearance and spatial features, and then to classify an image into object and non-object pixels based on the learned model. SVM is an effective supervised learning method that has been widely used for data analysis, pattern recognition, and classification.

Finally, we extend these single-image refinement methods to work as trackers over video sequences of the object moving and deforming. The graph cut tracking algorithm which only has color information is called as *Color GC-Tracker*, the one which also includes distance penalties is called *ColorD GC-Tracker*, and the SVM-based tracker is *SVM-Tracker*.

The rest of the paper is organized as follows. Section 2 describes the modifications we made in the standard graph cut method and the training and the classification stages of an SVM to segment object borders in a single image if a coarse estimate trail is given. Section 3 describes how these approaches can easily be extended to build tracking algorithms. Section 4 explains the experiments, shows some results of the algorithms, and compares their accuracies with ground-truth data. Finally, in Section 5, we summarize the methods and their results.

2 Single Frame Image Segmentation

This section describes how an object region is segmented by our graph cut and a support vector machine based algorithms given in a single image. Both techniques require to take a background and foreground region information in the image, M_B and M_F, respectively. M_F is obtained by scaling down initial estimated shape of the object, M_t. M_B is obtained by scaling up M_t and taking all the pixels outside of that region. M_t can come from manually segmented ground truths of the object, previous frames in the tracking procedures, or can be given by any other algorithm. Figure 2(b) shows M_B and M_F .

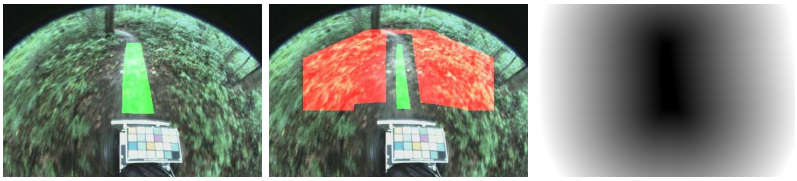
2.1 Graph Cut Segmentation

In this subsection, we describe the changes made in the standard graph cut segmentation algorithm proposed in [1, 15] to make it more suitable for the object segmentation and tracking purpose.

An image segmentation task is converted to a foreground/background labeling problem in the graph cut method. The following energy function is minimized by graph cut to assign a foreground or background label to each pixel in the image:

$$E(L) = \sum_{i=0}^{n} R_i(L_i) + \lambda \sum_{\{i,j\} \in N} B(L_i, L_j) \tag{1}$$

where L is the segmentation labeling set, $R_i(L_i)$ is called as *regional term* and $B(L_i, L_j)$ as *boundary term*. $\lambda \geq 0$ is a weight term to set the relative influence of *boundary term* versus *regional term* in the function. In this paper, 'object' refers to the foreground.



(a) (b) (c)

Fig. 2. (a) shows the image overlaid with its ground-truth polygon; (b) its obtained background model, red colored pixels, and foreground model, green colored pixels, for the graph cut and SVM algorithms; (c) and its distance penalty map for the graph cut algorithm.

Automatic Color Space Selection Standard graph cut algorithm requires some feature information from the background and foreground regions in the image. Choosing these models as much as informative and distinctive helps graph cut to produce better segmentation results. Our image sets contain high illumination changes. Working and sticking to only one color space such as *RGB* does not always produce good results. We investigate that in some cases using CIE-LAB color space gives better results. Therefore, we developed a mechanism to switch among the color spaces during the run time of the algorithm to improve the accuracy. Four possible color feature spaces are considered for this purpose: RGB, LAB which uses three channels of CIE-LAB color space, AB which uses only the chromaticity information, and L which uses only brightness of CIE-LAB.

To achieve self-adaptiveness among the feature spaces, our method collects some feature statistics from the inside and outside of the object regions in run time. To do that, we cluster all the pixels in the image into k different labels by applying k-means algorithm. The number of the clusters is chosen as 12, $k = 12$, for k-means. K-means is separately performed four times in RGB, LAB, AB and L feature spaces and four different cluster labels are obtained from the image. An object region color distribution, M_F, is modeled by a histogram $h = (f_1, \ldots, f_k)$ of the label frequencies inside it. The background region model, M_B, is formed in the same way. This allows to capture multi-model color distribution from the inside and outside of the object. We measure the appearance dissimilarity, d, between M_B and M_F by histogram distance function which is chi-squared metric $\chi^2(h_i, h_j)$. This measurement is done for each feature space and separate dissimilarity values are retrieved, d_{RGB} is for RGB, d_{LAB} is for LAB, d_{AB} is for AB and d_L is for L feature spaces. The feature space which provides the highest dissimilarity taken as the working color space of the graph cut. In this way, more informative and distinctive foreground and background models are provided to the graph cut. *Regional term* and *Boundary term* of the graph cut are changed as follows:

$$R_i(\text{"}Object\text{"}) = -ln(\frac{Pr(l_i|M_F)}{Pr(l_i|M_F) + Pr(l_i|M_B)}) \tag{2}$$

$$R_i(\text{"}Background\text{"}) = -ln(\frac{Pr(l_i|M_B)}{Pr(l_i|M_F) + Pr(l_i|M_B)}) \quad (3)$$

$$B_{(i,j)} = |R_i(\text{"}Object\text{"}) - R_i(\text{"}Background\text{"})| \quad (4)$$

where i and j are two neighbor pixels, l_i is k-means label of pixel i, M_F and M_B are the histogram label frequencies of the object and the background regions in the image, respectively.

Distance Penalty Map Construction and Weighting Graph-cut method produces global segmentation and tends to catch some unintended regions which are similar to the desired object. However, in the tracking procedure of the objects, the pixels labeled as the object in the current frame are most likely will be labeled as the object again in the next frame. To incorporate this biasing information, we need to penalize the pixels which are far away from the object region in the last frame. This penalty information is added to the standard graph-cut by constructing a distance penalty map as in [2]. However, we weight the distance penalty in a different way. Instead of guessing the location of the object in next frame as described in [2], the dissimilarity, d, between the object and background regions is chosen as a criteria to weight the distance penalty information adaptively. In our weighting technique, if the dissimilarity, d, is high between the regions, graph-cut is forced more to stay around the object location which is segmented in the last frame. Specifically, if M_F and M_B are similar to each other, we do not want graph cut to go far away from the last position of the object and look for more pixels to add to the object region, The distance map of the image is constructed as: $Map_{dist}(i) = \|i - closestTo_i\|$, where $closestTo_i$ is the closest pixel of the object to pixel i in the image space. $\|i - closestTo_i\|$ is the distance between $closestTo_i$ and pixel i. Figure 2(c) shows the distance penalty map of a given image. The distance penalty function is added to *regional term* of the standard graph cut algorithm in the following way:

$$R_i(\text{"}Object\text{"}) = Pr(P_i|O) + \beta(\alpha(d_{cs})Map_{dist}(i)) \quad (5)$$

where $Pr(P_i|O)$ is the penalty of adding pixel i to the object region. β is a constant term to set the relative influence. $\alpha(.)$ is the negative log-likelihood function to weight the distance penalty adaptively. d_{cs} is the smallest dissimilarity value returned by chi-squared metric among all feature color spaces considered in the algorithm.

2.2 SVM Segmentation

In this paper, we also apply SVM for the object segmentation as an alternative to the graph cut method. Our goal is to classify whether a pixel in the image belongs to the object region or not using a object model built by SVM. In this two-class classification problem, we define object pixels as positive examples and background pixels as negative examples. An object model is learned from selected color and geometric features using a radial basis SVM kernel. The features of a

pixel are subset of (L, a, b, x), where (L, a, b) are the Lab color space components and x is simply the horizontal position of the pixel in the image coordinates.

As a supervised learning algorithm, SVM needs training images with labeled positive and negative examples in order to build an object model. For the single image segmentation, such training images can be obtained from the ground truth. For the tracking case, training images for the current frame can be the tracking results of the previous frames, as explained in Section 3.2. We do not include pixels in the training images that are close to the boundaries of the negative and positive classes, because the classification of those pixels are marginal. At the classification stage, we use n-fold cross validation to enhance the accuracy. The largest connected component of all positive examples are then treated as the object region.

3 The Trackers

This section explains the details of our two proposed algorithms to track the objects. Both trackers are initialized with the ground truth positions of the object in the first frame.

3.1 Graph Cut Tracker

ColorD GC-Tracker is a single frame tracker which passes the color and position information of the object from the last frame to the next frame. It applies the graph cut segmentation algorithm, with the changes explained in Section 2, in each frame of the tracking process. The desired object region is retrieved as a mask from the graph cut. However, the raw foreground mask generated by graph cut technique might contain some noisy, small and weakly-connected foreground regions, because our images contain a non-homogeneous color distribution inside the foreground regions. In order to clean up those regions, we first do morphological opening and closing and find the connected components. The largest region is taken as the final object region.

3.2 Support Vector Machine Tracker

Similar to how we extend graph cut segmentation algorithm for tracking, the SVM object segmentation can also be extended to *SVM-Tracker*. After an initialization, the object model is updated based on the classification results from the previous n frames by adding the newly predicted positive and negative examples and throwing away the oldest examples, like a sliding window scheme. However, the training data for SVM is no longer ground truth but the classified results from the previous n frames. We exclude boundaries pixels of two classes when training as mentioned in Section 2.2.

4 RESULTS

We experimented with 3 different data-sets. The *Trail* dataset from [8] consists of 17,358 frames of video taken along a hiking/mountain biking trail in a mixture of field and forested terrain. The *Head* dataset is a short 383-frame clip from a standard video compression benchmark in which a person's head bobs around in the back of a car. Finally, *Hand* is a 5-minute (5,616-frame) video recorded outside our lab of a hand waving and gesturing in front of a complex background.

We manually generated ground-truth object segmentations for about $5-10\%$ of each dataset's frames at regular intervals. We use the following area overlap formula suggested by [16] to measure the accuracies between the ground-truth segmentations and our results: $Overlap(\mathcal{R}_1, \mathcal{R}_2) = A(\mathcal{R}_1 \cap \mathcal{R}_2)^2/(A(\mathcal{R}_1)A(\mathcal{R}_2))$, where $\mathcal{R}_1$ and $\mathcal{R}_2$ are given two regions to calculate the overlap between them. The SVM classifier for this work is implemented using LIBSVM [17] with a 10-fold cross validation.

4.1 Single Frame Image Segmentation Experiments

In this experiment, we aimed to see the accuracies of the algorithms when only one single image is given and expected to segment the object borders in it. We performed this experiment for the images which we have ground-truths for. In order to see the performance of adding distance penalty to the graph cut, we removed the distance information from the graph cut method and applied the graph cut method which only includes the changes described in Section 2.1. All experimented methods are initialized with the ground-truth segmentations, so M_t is set to be the ground-truth. Table 1 summarizes the median overlap scores between the methods and the ground-truth segmentations for each data sets. Some results of this experiment are shown in Figure 3. As expected, adding the color information to the standard graph cut helps to increase the accuracy of the segmentation. Also, incorporating the distance penalty and weighting it according to the dissimilarity between M_F and M_B improves the performance. For SVM segmentation, we subsampled positive and negative examples from training images for efficiency purposes at a rate of 1 out of every 8 pixels in a regular basis.

Method Name	*Trail*	*Hand*	*Head*
Standard GC [1]	0.55	0.79	0.52
Color GC	0.69	0.95	0.84
ColorD GC	0.79	0.97	0.89
SVM	0.76	0.95	0.86

Table 1. Median overlap scores of the single frame image segmentation results for the data sets.

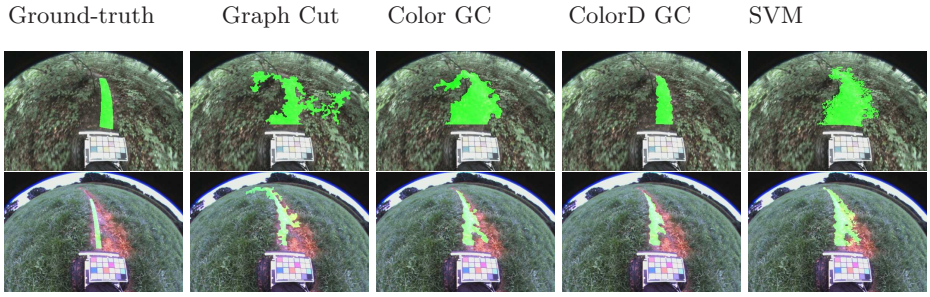


Fig. 3. Sample results for single frame refinement on the trail dataset

4.2 Measuring the Accuracies of the Trackers

In this experiment, the accuracies of the trackers described in Section 3 are measured and analyzed. The trackers are started with the ground-truth image in the first frame and run until the frame which has a ground-truth. The generated object mask in the last frame is saved. Total 502 results, the number of the images in the ground-truth sets, are produced in this way for *Trail*, *Hand*, and *Head* data sets. *Color GC-Tracker* does not use distance penalty, it is the method described in Section 2.1 without distance penalty. Table 2 shows the median overlap scores of this experiment. Some of final segmented trail, hand and head regions at the end of the sequences can be shown in Figure 3 and 4. Overall accuracy of *ColorD GC-Tracker* is comparable with *SVM-Tracker* as can be seen in Table 2. *ColorD GC-Tracker* performs better than *SVM-Tracker* in this test. The big overlap score difference between *SVM-Tracker* and *ColorD GC-Tracker* for *Head* set test is caused by similar colored background and foreground regions in the images. Distance penalty information incorporated in *ColorD GC-Tracker* helped it for better tracking.

For SVM tracker, we use the same feature vector as that for single image segmentation. We assign different weights for positive and negative examples during training stage based on the ratio between the number of positive and negative examples. Such an adaptive weight assign scheme helps to stabilize the SVM tracker by balancing the number of both examples for training. SVM tracker might lose tracking when all pixels in the images are classified as positive or negative given the model trained from the detection results of the previous frames. This happens due to an abrupt illumination changes, such as entering a shadow region or sudden camera exposure change. Therefore, we do not update the object model when all pixels are classified into one class, and the SVM tracker can pick up the object region back after several frames when the illumination condition is stable. The SVM results in Table 2 are obtained when $n = 1$ as mentioned in Section 3.2.

Method Name	*Trail*	*Hand*	*Head*
CC-Tracker [8]	0.72	—	—
Color GC-Tracker	0.46	0.92	0.27
ColorD GC-Tracker	0.77	0.96	0.82
SVM-Tracker	0.74	0.95	0.68

Table 2. Median overlap scores of the trackers for the data sets. Since *CC-Tracker* is specific to tracking trails, it does not have overlap scores for the *Hand* and *Head* data sets.

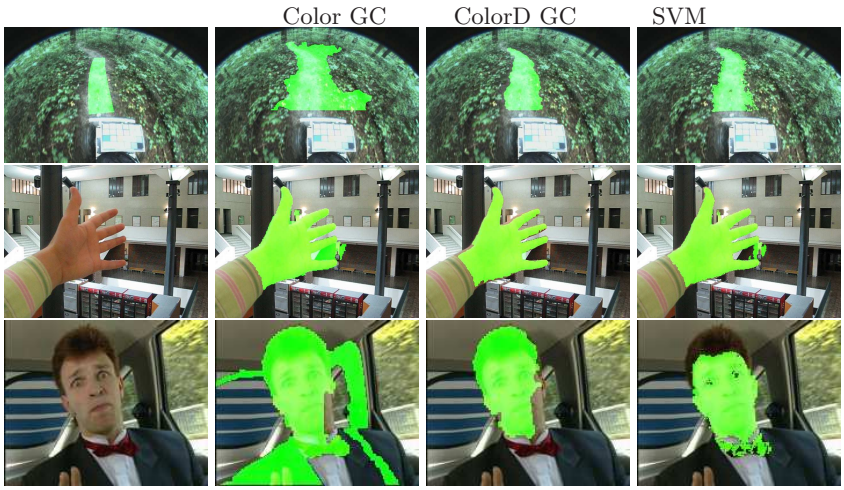


Fig. 4. Sample tracking results on all three data sets (the trail first column has ground truth overlaid).

5 CONCLUSION

We described two algorithms to segment the object borders in a single image if an estimated position of it is provided and their extended tracking algorithms. They are separately build on two well known methods, graph cut segmentation technique and support vector machines. In order to improve the accuracy of the standard graph cut method, adaptive color space selection mechanism, distance penalty and its adaptive weighting method are employed in it. Also, the image is clustered by k-means and its cluster labels are used in the formulation of graph cut terms. Our SVM based tracker is trained and tested by incorporating various features selected from color and 3D space of the object. We compared and analyzed these methods with a previously published color contrast trail tracker and standard graph cut method.

As a future direction of this research, SVM and graph cut can be combined in one single tracker to utilize from the advantages of two methods. SVM pro-

vides the opportunity to maintain the color and structure information over the image sequences. Graph cut has the advantage of encompassing the connectivity information of neighbor pixels in the image. Maintaining the background and foreground models by SVM and using these models in the graph cut segmentation might improve the accuracy and robustness of the tracker. Also, object location prediction mechanism can be added to the trackers. In this way, a better object and background models can be obtained to feed the algorithms.

References

1. Boykov, Y., Funk-Lea, G.: Graph cuts and efficient n-d image segmentation. Int Journal of Computer Vision **70** (2006) 109–131
2. Malcolm, J., Rathi, Y., Tannenbaum, A.: Tracking through clutter using graph cuts. In British Machine Vision Conf. (BMVC) (2007)
3. Burges, C.: A tutorial on support vector machines for pattern recognition. Data Mining and Knowledge Discovery (1998) 121–167
4. Taylor, C., Malik, J., Weber, J.: A real-time approach to stereopsis and lane-finding. In: Proc. IEEE Intelligent Vehicles Symposium. (1996)
5. Southall, B., Taylor, C.: Stochastic road shape estimation. In: Proc. Int. Conf. Computer Vision. (2001) 205–212
6. Huang, A., Moore, D., Antone, M., Olson, E., Teller, S.: Multi-sensor lane finding in urban road networks. In: Robotics: Science and Systems. (2008)
7. Rasmussen, C., Lu, Y., Kocamaz, M.: Appearance contrast for fast, robust trail-following. In: Proc. Int. Conf. Intelligent Robots and Systems. (2009)
8. Rasmussen, C., Lu, Y., Kocamaz, M.: Trail following with omnidirectional vision. In: Proc. Int. Conf. Intelligent Robots and Systems. (2010)
9. Rother, C., Kolmogorov, V., Blake, A.: Grabcut - interactive foreground extraction using iterated graph cuts. In: SIGGRAPH. (2004)
10. Kocamaz, M., Rasmussen, C.: Automatic refinement of foreground regions for robot trail following. In: Proc. Int. Conf. Pattern Recognition. (2010)
11. Dinh, T., Medioni, G.: Two-frames accurate motion segmentation using tensor voting and graph-cuts. In: IEEE Workshop on Motion and Video Computing. (2008)
12. Mehrani, P., Veksler, O.: Saliency segmentation based on learning and graph cut refinement. In: Proc. British Machine Vision Conference. (2010)
13. Nelson, A., Neubert, J.: Object tracking via graph cuts. In: SPIE Applications of Digital Image Processing. (2009)
14. Papadakis, N., Bugeau, A.: Tracking with occlusions via graph cuts. IEEE Trans. Pattern Analysis and Machine Intelligence **33** (2011) 144–157
15. Boykov, Y., Jolly, M.: Interactive graph cuts for optimal boundary and region segmentation of objects in N-D images. In: Proc. Int. Conf. Computer Vision. (2001)
16. Sclaroff, S., Liu, L.: Deformable shape detection and description via model-based region grouping. IEEE Trans. Pattern Analysis and Machine Intelligence **23** (2001)
17. Chang, C.C., Lin, C.J.: LIBSVM: A library for support vector machines. ACM Transactions on Intelligent Systems and Technology **2** (2011) 27:1–27:27 Software available at http://www.csie.ntu.edu.tw/ cjlin/libsvm.