Ray-based Color Image Segmentation

Changhai Xu, Yong Jae Lee, and Benjamin Kuipers
The University of Texas at Austin
{changhai, yjlee, kuipers}@cs.utexas.edu

Abstract

We propose a ray-based segmentation method for color images. A segment is represented by a centroid and evenly-distributed rays shooting out from it. First, a bottom-up low-level boundary detection process coarsely constructs candidate segments. Then, two top-down learning processes, mid-level intra-segment learning and high-level inter-segment learning, create the best segments. Segments are created sequentially until all pixels are classified. The number of segments is determined automatically. We test our method on the Berkeley Segmentation Dataset. Evaluation results show that our algorithm produces better results than those of the Normalized Cuts segmentation method.

1. Introduction

Image segmentation is a fundamental step for a robot towards achieving high level goals, such as navigation and learning object concepts. Picture a mobile robot equipped with a camera as its vision sensor. If the robot wants to learn the models of some objects appearing in its visual field, the first thing the robot should do is to individuate the objects it is interested in. But the data streaming in from the camera is huge and at pixel level. Without segmentation of the data, the robot will never learn the models of its surrounding objects. The segmentation here can be based on each pixel's location, intensity, color, motion, etc. After the robot separates the visual world into a small number of segments, it can track each of them and learn their underlying properties. The segmentation process not only makes tractable representations but also decreases ambiguities and uncertainties in pixel-level data.

In this paper, we propose a new segmentation method. We represent a segment by a centroid and evenly-distributed rays shooting out from it. Each ray has a boundary point, which is determined by a set of cues such as position, color, texture, and edge. A segment is formed by combining all the boundary points. Our segmentation algorithm includes a bottom-up boundary detection process and two top-down learning processes. In the boundary detection process, we determine the boundary point for every ray using local edge/color information. This is a low-level process. A confidence score is assigned for each ray based on how "good" its boundary point is. The learning processes include intra-segment learning and inter-segment learning. In intra-segment learning, a ray uses mid-level cues, i.e., information from its neighboring rays, to evaluate its boundary point and re-detect it, if necessary. In inter-segment learning, a number of candidate segments are generated, and based on some measurement function the best one is chosen to be the new segment.

Unlike many other segmentation algorithms such as k-means and Normalized Cuts, our algorithm determines the number of segments automatically. This is particularly important for an autonomous robot, since there would be no interaction between it and a human during operation.

The paper is organized as follows. In section 2, we review related work. Section 3 describes the representation of segments. Section 4 presents our segmentation algorithm in detail. Section 5 evaluates our method and compares it with the Normalized Cuts segmentation method. Conclusions and future work are described in Section 6.

2. Related Work

k-means is a commonly used clustering method for image segmentation. One difficulty in using k-means is that the resulting segments may be disconnected and scattered, when the feature vectors are composed of only the pixel values. The problem can be alleviated by adding the spatial position of each pixel as feature attributes. However, the resulting segmentation is one that has large regions broken up, due to the influence of the location as well as the pixel values. Luo et al. [5] showed that better segmentation results could be obtained by performing hierarchical k-means on non-spatial features combined with spatial constraints. First, the data is clustered based on the non-spatial features by k-means clustering. The clustered data is further split using spatial constraints with connected components. Then, k-means clustering is performed on the new segmentations,

(a) (b)

Figure 1. The left image shows the sky, trees, houses and grass. The right image is created from the left one, in which the blue sky is represented by a centroid and 64 rays emanating from it.

and the process is repeated.

Denzler et al. [3] used active rays to represent object boundaries for object tracking. Boundary points were detected by minimizing an energy function. Objects with simple shapes were segmented and tracked over a sequence of images. We use a similar model to represent the clusters. However, we deal with much more complex data where simple boundary detection techniques fail.

Comaniciu and Meer [2] employed the $mean\ shift\ algorithm$, a non-parametric technique to analyze and find arbitrarily shaped clusters in feature space. Shi and Malik [8] proposed the $Normalized\ Cuts\ algorithm$, which treats clustering as a graph partitioning problem. The nodes are represented by the points in feature space and edges are established between each pair of nodes. The algorithm tries to find the minimum cut in the graph that preserves high similarity for intra-cluster points, and low similarity for inter-cluster points. However, the Normalized Cuts algorithm requires the number of segments to be specified, and can be sensitive to the chosen number.

There has also been work done in literature on cluster evaluation techniques. Entropy, purity, precision, recall, and the F-measure are classification evaluation measures that have been used in the cluster evaluation domain. Distance measures such as the Rand Index [7] and the Jaccard Index [1] have been standard techniques for evaluating cluster similarities. Unnikrishnan et al. [9] used a measure called the Normalized Probabilistic Rand (NPR) index, an evaluation specific for the image segmentation domain. The NPR index measures quantitative comparison between an image segmentation algorithm and a hand-labeled set of ground-truth segmentations. Jiang et al. [4] also considered performance evaluation techniques for region segmentation algorithms in the ground-truth based paradigm. They focused on measures for comparing partitions made by different clustering algorithms on the same set of objects, including those that were hand-segmented.

3. Segment Representation

A segment S is represented by a centroid point, and boundary points determined by evenly distributed k rays shooting out from the centroid. The distance from the boundary point to the centroid is defined as the radius of the corresponding ray. The segment's space is defined by the region enclosed by the ray boundaries. We use the terms "boundary of a segment" and "shape of a segment", interchangeably.

Let's take an image for example, as shown in Figure 1. The left image is the original image. In the right image the blue sky segment has been separated out. The green circle represents the centroid of the segment. The red lines are the rays emanating from the centroid in 64 directions $0, \pi/32, 2\pi/32, ...,$ and $63\pi/32$. The boundary/shape of the segment is the yellow contour. Note that the segment centroid is not necessarily the gravitational center of that segment. We call it a centroid because the rays shoot out from this point. We have found that in practice, our learning algorithm, which will be described later, produces centroids that tend to be close to the gravitational center.

This representation is similar to [3], but we handle more complex data and use a different boundary detection method.

4. Algorithm

We would like to group pixels $\{p_1, p_2, ..., p_n\}$ into segments $\{S_1, S_2, ..., S_N\}$. The pixels within a segment should have similar properties, while pixels in adjacent segments should not.

We define three auxiliary maps in our segmentation algorithm: indicator map, edge map and color map.

• Indicator map, $\{b_1, b_2, ..., b_n\}$, where $b_i \in \{0, 1\}$ represents whether p_i has already been classified into some segment.

• Edge map, $\{e_1, e_2, ..., e_n\}$, where $e_i \in [0, 1]$ is a real number and represents the confidence that p_i is a point on an edge. We use the Canny detector to detect edges in the image several times, each time with different thresholds. Confidence values are assigned to the edges based on the threshold conditions.

• Color map, $\{c_1, c_2, ..., c_n\}$, where c_i is the color of p_i, in RGB space.

In our algorithm, segments are sequentially created. A number of candidate segments are generated from unclassified pixels and an inter-segment learning process selects the best candidate as the new segment. Pixels within this segment are then marked as classified in the indicator map. This procedure iterates until all pixels are marked as classified.

Each candidate segment is generated from an unclassified seed point. The seed point can be selected either randomly or in a pre-determined manner, e.g., uniform sampling. Taking this seed point as the centroid, we shoot out a number of evenly-distributed rays. For each ray we find a boundary point by the ray boundary detection process. The pixels enclosed by these boundary points belong to this candidate segment. After a candidate segment is generated, an intra-segment learning process identifies those rays that are significantly different from their neighbors and re-builds these rays.

The ray boundary detection process is a low-level bottom-up process that constructs coarse segments from pixels. The intra-segment learning is a mid-level top-down process. It re-builds a "bad" ray by learning from its neighboring rays within the same segment. The inter-segment learning is a high-level top-down process, which uses global cues among all candidate segments to select the best one.

A few simple rules are used in ray boundary detection process, which may result in incorrect boundaries because only local information around the point of interest is considered. But the top-down intra-segment learning and inter-segment learning use mid-level cues and high-level cues, and make the segmentation results much more robust and accurate.

4.1. Ray Boundary detection

The task of ray boundary detection is to find the boundary point on a ray. We first group consecutive pixels in similar color into a 1D-segment. If a pixel's edge confidence in the edge map exceeds some threshold, or the color difference between this pixel and the previous pixel exceeds some threshold, a new 1D-segment is created. Let $\{q_1, q_2, ..., q_m\}$ represent the 1D-segments on that ray. Note that 1D-segments that are not adjacent could have similar colors. Our goal is to find a boundary point, that delineates similar 1D-segments to non-similar 1D-segments.

In our method, we use a few simple rules to determine a boundary point. Starting with q_1, which is the 1D-segment that begins at the centroid, we group subsequent 1D-segments q_i, where $1 \leq i \leq m$, that have similar color and similar length to q_1. We also impose the constraint that for every neighboring q_i and q_j in this group, where $i < j$, the distance between q_i and q_j should be less than some threshold.

Some attributes are defined for a ray.

• Radius, R, is the distance from the centroid to the boundary point.

• Color, $color$, is the average color in RGB space of all the pixels from the centroid to the boundary point.

• 1D-segment number, m, is the number of 1D-segments from the centroid to the boundary point.

• Texture confidence, $cf_{texture}$, denotes how uniform/textured the ray is. It is defined as

$$cf_{texture} = \min_{1 \leq i \leq m} \frac{1 - len_{seg_i}}{R} \tag{1}$$

where len_{seg_i} is the length of the i-th 1D-segment.

• Edge confidence, cf_{edge}, is the edge confidence of the boundary point in the edge map.

• Dominant element ratio, $ratio$, is the ratio between the number of pixels that have similar color to the centroid pixel and the number of pixels from the centroid to the boundary point. This is equivalent to the ratio between the total length of 1D-segments that have similar color to the first 1D-segment q_1 and R.

With the above attributes, we assign a confidence score to a ray, indicating how "good" the boundary point is on this ray. The confidence score is

$$cf = \alpha_1 R/m + \alpha_2(1 - cf_{texture}) + \alpha_3 cf_{edge} + \alpha_4 ratio \tag{2}$$

where $\alpha_1, \alpha_2, \alpha_3, \alpha_4 \in [0, 1]$ and $\alpha_1 + \alpha_2 + \alpha_3 + \alpha_4 = 1$. We set $\alpha_1 = 0.4, \alpha_2 = 0.3, \alpha_3 = 0.25, \alpha_4 = 0.05$.

The meaning of this confidence score is intuitive. In general, a higher score is assigned for boundary points that are more easily detected.

4.2. Intra-segment learning

With the ray boundary detection process, we can generate a coarse segment from an arbitrary centroid point. But some of the boundary points of the segment may be incorrect, due to noise. Since we have a coarse segment, we may re-build a "bad" ray by making use of information from its neighboring rays.

We define some similarity measures between two rays.

• Similarity of radius

$$sim_{radius}(ray_1, ray_2) = e^{-\frac{|R_1 - R_2|}{\eta_1 \min(R_1, R_2)}} \tag{3}$$

• Similarity of dominant element ratio

$$sim_{ratio}(ray_1, ray_2) = e^{-\frac{|ratio_1 - ratio_2|}{\eta_2 \min(ratio_1, ratio_2)}} \quad (4)$$

• Similarity of color

$$sim_{color}(ray_1, ray_2) = e^{-\frac{\|color_1 - color_2\|_2}{\eta_3 \min(\|color_1\|_2, \|color_2\|_2)}} \quad (5)$$

where $color_1$ and $color_2$ are the colors of the two rays, in RGB color space.

In the above measures, $\eta_1, \eta_2, \eta_3 \in (0, 1)$. We set $\eta_1 = \eta_2 = \eta_3 = 0.3$. The total similarity between two rays is a combination of the above measures.

$$sim(ray_1, ray_2) = \beta_1 sim_{radius} + \beta_2 sim_{ratio} + \beta_3 sim_{color} \quad (6)$$

where $\beta_1, \beta_2, \beta_3 \in [0, 1]$ and $\beta_1 + \beta_2 + \beta_3 = 1$. We set $\beta_1 = 0.6, \beta_2 = 0.2, \beta_3 = 0.2$.

We make the assumption that a segment should have a smooth boundary. This is a reasonable assumption and is generally true in real-world images.

For each ray in a segment, if the measure sim_{radius} with its left or right neighboring rays is greater than some threshold, it is identified as a potential "bad" ray. We examine a number of rays in its neighborhood, including itself. We calculate the average radius R_{avg}, and the average dominant element ratio $ratio_{avg}$ among these rays. If the "bad" ray's radius or dominant element ratio is significantly different from the average values, the ray is rebuilt as follows.

First, pixels in $[R_{avg} - \delta, R_{avg} + \delta]$ on that ray are examined where δ is a real positive number. Then the boundary point is assigned to be the point that best matches the boundary points among the neighboring rays, where the matching is based on local color information in the color map. We compute the new dominant element ratio for the ray, and if it is closer to $ratio_{avg}$ than the previous value, then the learning succeeds and the new boundary point is set. Otherwise, the learning fails and no change is made to the boundary point.

4.3. Inter-segment learning

In our algorithm, segments are created sequentially. When creating a new segment, we choose the best possible segment among a number of candidates. The candidate segments are generated from seed points. Seed points can be selected either randomly or in some pre-determined manner, e.g., uniform sampling. From each seed point, we generate a candidate segment based on the ray boundary detection process and the intra-segment learning process.

For every new candidate segment that is generated, the confidence of the boundary points for each of its rays is obtained from previously created segments if the boundary points happen to fall on those segments' boundaries.

For each candidate segment that has k rays within it, we define the following attributes.

• Confidence of shape smoothness, which measures the average radius difference of neighboring rays,

$$conf_{smooth} = \frac{1}{k} \sum_{i=1}^{k} sim_{radius}(ray_i, ray_{i+1}) \quad (7)$$

in which ray_{k+1} is defined as ray_1.

• Confidence of shape regularity, which measures the local maximum/minimum radii,

$$conf_{reg} = \frac{1}{2k}\Big(\sum_{\substack{1 \le i \le k \\ R_i > R_{i-1} \\ R_i > R_{i+1}}} reg_i + \sum_{\substack{1 \le i \le k \\ R_i < R_{i-1} \\ R_i < R_{i+1}}} reg_i \Big) \quad (8)$$

where

$$reg_i = sim_{radius}(ray_i, ray_{i-1}) + sim_{radius}(ray_i, ray_{i+1}) \quad (9)$$

in which $1 \le i \le k$, ray_{k+1} is defined as ray_1, and ray_0 is defined as ray_k.

• Confidence of shape symmetry, which measures how symmetric the shape is in terms of the centroid,

$$conf_{sym} = \frac{1}{k} \sum_{i=1}^{k/2} sim(ray_i, ray_{i+k/2}) \quad (10)$$

Since k is a user-determined parameter, it should always be chosen to be an even number.

Based on the above similarity functions, we define the total shape confidence of a segment as

$$conf = \lambda \sum_{i=1}^{k} cf_i \quad (11)$$

where cf_i is the boundary confidence of ray_i, and λ is a coefficient defined as

$$\lambda = (\gamma_1 conf_{smooth} + \gamma_2 conf_{reg}) * conf_{sym} * area \quad (12)$$

where $\gamma_1, \gamma_2 \in [0, 1]$, $\gamma_1 + \gamma_2 = 1$, and $area$ is the area of this candidate segment. We set $\gamma_1 = 0.6, \gamma_2 = 0.4$.

$$area = \frac{1}{2} sin\frac{2\pi}{k} \sum_{i=1}^{k} R_i R_{i+1} \quad (13)$$

where $R_{k+1} = R_1$.

Among all the candidate segments, we choose the one with the best shape confidence as the new segment. In essence, we give preference to segments with large areas

and "good" shapes, that is, shapes with good smoothness, regularity and symmetry. This way, the segments created last are usually very small, which do not affect the segmentation too much. Hence, our algorithm is not sensitive to the number of segments, unlike many other segmentation algorithms such as k-means and Normalized Cuts.

After a new segment is created, we mark all pixels within the segment as classified. When the number of classified pixels equals the total pixel number in the image, or is greater than some user-defined threshold, the algorithm terminates.

4.4. Segment merging

Even though our segment representation is able to represent many shapes, it may break down for very complex shapes. This can result in an over-segmented image. We perform segment merging to compensate for this. For each created segment, we define a few more attributes.

• Color:

$$color_{seg} = \frac{1}{\sum_{i=1}^{k} R_i} \sum_{i=1}^{k} R_i color_i \tag{14}$$

in which $color_i$ is the color of ray_i.

• Dominant element ratio:

$$ratio_{seg} = \frac{1}{k} \sum_{i=1}^{k} ratio_i \tag{15}$$

in which $ratio_i$ is the dominant element ratio of ray_i.

• Texture:

$$texture_{seg} = \frac{1}{k} \sum_{i=1}^{k} cf_{texturei} \tag{16}$$

in which $cf_{texturei}$ is the texture confidence of ray_i.

If the differences between two adjacent segments' colors, dominant element ratios and textures are within some thresholds, we merge the two segments. This process iterates until no adjacent segments can be merged.

5. Experimental Evaluation

5.1. Source Dataset

The Berkeley Segmentation Dataset [6] is used to evaluate our algorithm. This dataset provides 300 color images divided into 200 train and 100 test sets with hand-labeled ground truth segmentation images from 30 human subjects. We select the test set images for our experiments. We compare the Normalized Cuts algorithm and our algorithm on the provided ground-truth images.

5.2. Methodology

Performance evaluation of image segmentation techniques is very difficult, since what defines a "good" segmentation of the underlying image is subjective. Unlike classification algorithms that measure how many instances are classified correctly, segmentation algorithms need evaluation of results that are much more subtle.

If we treat the image segmentation problem as one of data clustering, then we can use cluster evaluation techniques developed in the machine learning and statistics community to evaluate the segmentation results. This is the approach taken in [4]. Some of the measures used are the Rand Index [7], Jaccard index [1], and a distance index proposed by van Dongen [10]. We use the same measures for our evaluation.

Given a set of n objects $S = (o_1, ..., o_n)$, a partition is defined as a set of clusters $C = (c_1, ..., c_k)$, where $c_i \subseteq S$, $c_i \cap c_j = \emptyset$ if $i \neq j$, $\bigcup_{i=1}^{k} c_i = S$. Given two partitions, $X = (x_1, ..., x_r)$ and $Y = (y_1, ..., y_s)$ of the same set of objects S, the following four quantities can be measured for all pairs of objects (o_i, o_j), $i \neq j$, from X and Y, respectively:

(i) f_{00} = number of such pairs that fall in different clusters under X and Y.

(ii) f_{01} = number of such pairs that fall in the same cluster under Y but not under X.

(iii) f_{10} = number of such pairs that fall in the same cluster under X but not under Y.

(iv) f_{11} = number of such pairs that fall in the same cluster under X and Y.

where $f_{00} + f_{01} + f_{10} + f_{11} = \binom{n}{2}$, and n is the total number of objects in S. Intuitively, one can think of $f_{00} + f_{11}$ as the number of agreements between X and Y and $f_{01} + f_{10}$ as the number of disagreements between X and Y.

The Rand index is defined as:

$$R(X,Y) = 1 - \frac{f_{11} + f_{00}}{f_{00} + f_{01} + f_{10} + f_{11}} \tag{17}$$

The Jaccard index is defined as:

$$J(X,Y) = 1 - \frac{f_{11}}{f_{11} + f_{10} + f_{01}} \tag{18}$$

The distance measures are in the domain [0,1]; a value of 0 means maximum similarity, while a value of 1 means maximum dissimilarity. The Jaccard index does not give any weight to pairs of objects that belong to different clusters under the two partitions. Hence, its distance measure is generally higher than that of the Rand index.

Table 1. Average index measures with +/- 1 standard deviation for Normalized Cuts and our algorithm on all test set images of the Berkeley Segmentation dataset. The top values correspond to the $average$ index values obtained from comparison to all ground truth segmented images, while the bottom values correspond to the $minimum$ index values. Lower values indicate higher similarity to the ground truth segmentations.

METHOD	RAND	JACCARD	D
NCUTS (AVG)	0.26± 0.13	**0.77± 0.06**	**0.38± 0.05**
OUR ALG. (AVG)	0.24± 0.12	**0.62± 0.15**	**0.28± 0.08**
NCUTS (MIN)	0.18± 0.10	**0.68± 0.10**	**0.33± 0.07**
OUR ALG. (MIN)	0.18± 0.10	**0.55± 0.17**	**0.25± 0.08**

Another evaluation method is one based on set matching between clusters of two partitions. The following term measures the matching degree between clusters of X and Y:

$$A(X,Y) = \sum_{x_i \in X} \max_{y_j \in Y} |x_i \cap y_j| \tag{19}$$

where the maximum value is n if $X = Y$. This leads to the following index:

$$D(X,Y) = \frac{2n - A(X,Y) - A(Y,X)}{2n} \tag{20}$$

The Rand index and Jaccard index are distance measures of two partitions based on counting pairs of objects, while the D index is a distance measure of two partitions based on set matching.

5.3. Discussion

For our performance evaluation, we choose the two partitions, X and Y, to be the Algorithm Segmentation (AS) and the Ground Truth (GT) segmentation images, respectively. Since there is no one universal measure for cluster evaluation, we use all three index measures for our experiments. The image is represented as S and each pixel is represented as an object o_i, where $i \in n$, and n = number of pixels in the image.

The Berkeley Segmentation Dataset provides 300 color images with hand-labeled ground truth segmentation images from 30 human subjects. Many of the images have various foreground and background objects in the image, making the segmentation task quite difficult. Each color image has 5 to 7 hand-segmented ground truth images, where the number of partitions and the manner in which the image has been segmented differs for each human subject. To make our evaluation "fair", we evaluate our algorithm by comparing it to each of the ground truth segmentations and obtain the corresponding distance index values. For each AS image, we present the index values in two ways: the average value obtained from comparison to all ground truth segmented images, and the minimum (corresponding to the "best") value.

We compare our algorithm with the Normalized Cuts algorithm [8](we used the authors' implementation). Note that we use the same values for all the parameters described in the Section 4, for our algorithm. The features used for the Normalized Cuts are the gray-scale pixel values. Each of the test images are segmented with the Normalized Cuts algorithm and compared to the ground truth segmentations to obtain the index values. Table 1 shows the different distance index values for the images. The results shown are averaged values for all images, with +/- 1 standard deviation. Specific distance index values for each image can be seen in Figure 2. Again, two sets of results are presented: the average index values obtained from comparison to all ground truth segmentations, and the "best" index value. Overall, the proposed algorithm's segmentation results are better or comparable to those of Normalized Cuts. Normalized Cuts is sensitive to the number of segments k, which is a parameter to the algorithm. To choose the "ideal" k for the Normalized Cuts algorithm, for every ground-truth segmentation image that is used for computing the distance measures, we choose k to be equal to the number of segments specified in the ground-truth segmentation image. Unlike the Normalized Cuts algorithm, our algorithm determines the number of segments automatically.

For the Rand Index measure, our algorithm and the Normalized Cuts algorithm is comparable, with our algorithm showing slightly better results. This is because for two segmentations of the same image, the Rand Index considers pairs of pixels that belong to different segments.

Results on some examples can be seen in Figure 3.

6. Conclusions and Future Work

We propose a ray-based segmentation method for color images. Our algorithm uses top-down learning processes to strengthen the bottom-up naive boundary detection process and improve segmentation results, by making use of mid-level and high-level cues. Experiments on Berkeley Segmentation Dataset for our algorithm show better results than Normalized Cuts.

Our evaluation results indicate that, with learning processes, even a naive boundary detection algorithm works well. For future work, we would like to integrate advanced boundary detection algorithms into the learning framework.

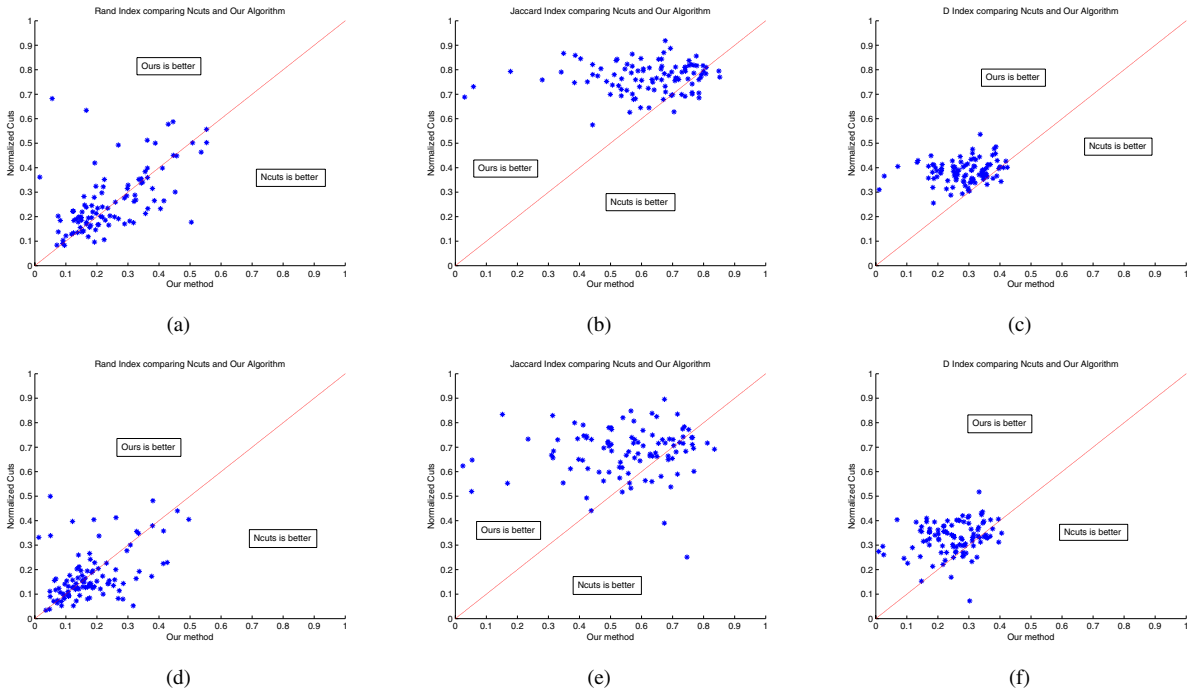

Figure 2. Each point represents the distance measure of an image. Our algorithm's measure is the x-value and Normalized Cuts' measure is the y-value. Points above the red diagonal are images where our algorithm produced better segmentations (lower index values) than Normalized Cuts. (a) $\sim$ (c): average Rand, Jaccard, and D index values obtained from comparison to all ground truth segmented images per test image. (d) $\sim$ (f): minimum Rand, Jaccard, and D index values obtained from comparison to all ground truth segmented images per test image.

Acknowledgements. This work has taken place in the Intelligent Robotics Lab at the Artificial Intelligence Laboratory, The University of Texas at Austin. Research of the Intelligent Robotics lab is supported in part by grants from the National Science Foundation (IIS-0413257, IIS-0713150 , and IIS-0750011) and from the National Institutes of Health (EY016089). The authors would like to thank Aniket Murarka, and the reviewers for their helpful comments.

References

[1] A. Ben-Hur, A. Elisseeff, and I. Guyon. A stability based method for discovering structure in clustered data. In *Pacific Symposium on Biocomputing*, pages 6–17, 2002.

[2] D. Comaniciu and P. Meer. Mean shift: a robust approach toward feature space analysis. *Pattern Analysis and Machine Intelligence, IEEE Transactions on*, 24(5):603–619, 2002.

[3] J. Denzler, B. Heigl, and H. Niemann. An Efficient Combination of 2D and 3D Shape Descriptions for Contour Based Tracking of Moving Objects. *Proc. 5th European Conf. Computer Vision*, pages 843–857, 1998.

[4] X. Jiang, C. Marti, C. Irniger, and H. Bunke. Distance measures for image segmentation evaluation. *EURASIP J. Appl. Signal Process.*, 2006(1):209–209, 2006.

[5] M. Luo, Y. Ma, and H. Zhang. A spatial constrained k-means approach to image segmentation. *Fourth IEEE Pacific-Rim Conference on Multimedia*, 2:738–742, 2003.

[6] D. Martin, C. Fowlkes, D. Tal, and J. Malik. A database of human segmented natural images and its application to evaluating segmentation algorithms and measuring ecological statistics. In *Proc. 8th Int'l Conf. Computer Vision*, volume 2, pages 416–423, July 2001.

[7] W. M. Rand. Objective criteria for the evaluation of clustering methods. *Journal of the American Statistical Association*, 66:622–626, 1971.

[8] J. Shi and J. Malik. Normalized cuts and image segmentation. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 22(8):888–905, 2000.

[9] R. Unnikrishnan, C. Pantofaru, and M. Hebert. Toward objective evaluation of image segmentation algorithms. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 29(6):929–944, 2007.

[10] S. Van Dongen. Performance criteria for graph clustering and Markov cluster experiments. Report No. INS-R0012. *Center for Mathematics and Computer Science (CWI), Amsterdam*, 2000.

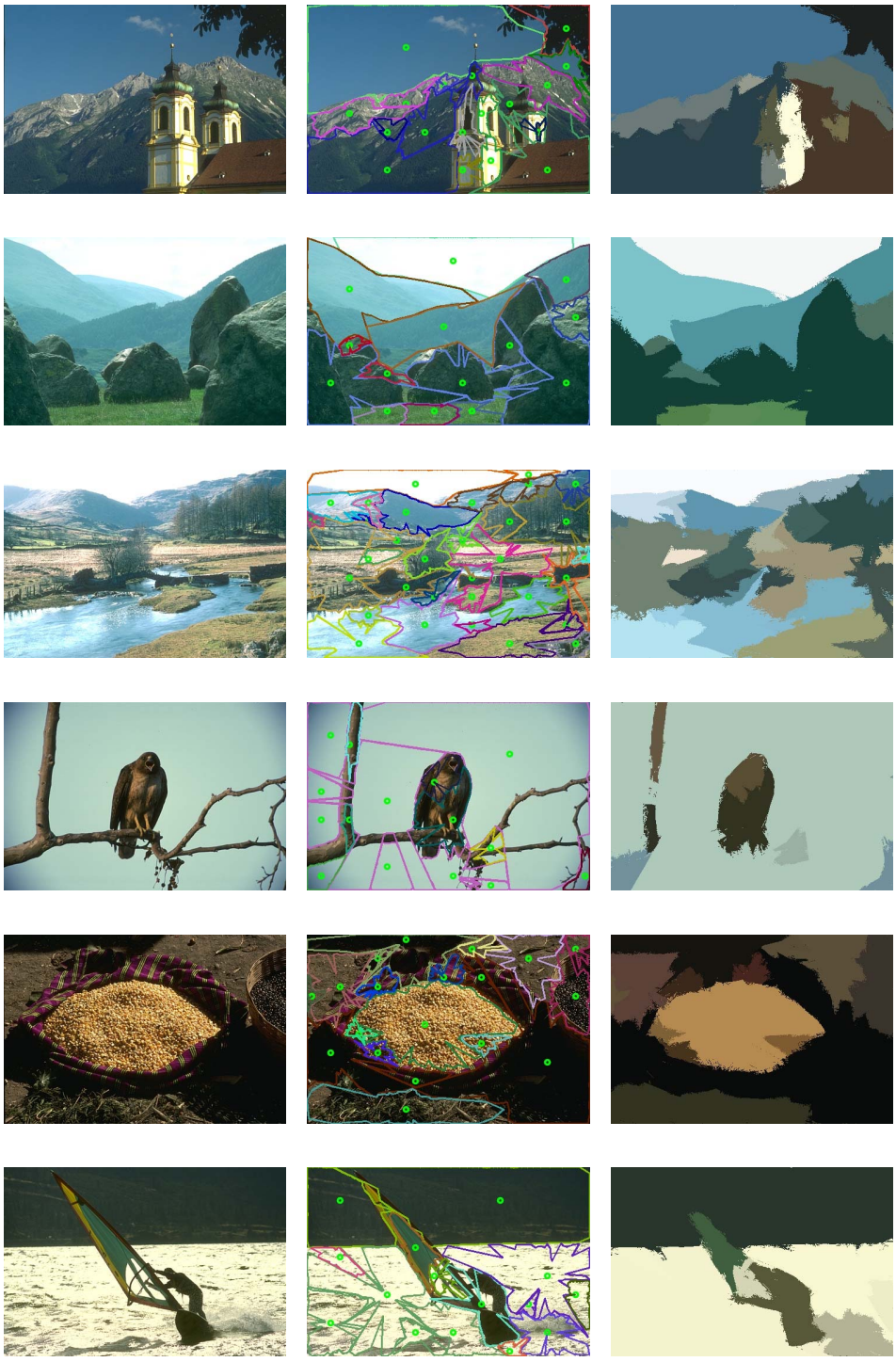

Figure 3. The first column shows the original image. The second column shows the centroids and boundaries of each segment detected by our algorithm. The third column shows the final segmentation results.