Fast Segmentation via Randomized Hashing

Camillo J. Taylor
http://www.cis.upenn.edu/~cjtaylor/
Anthony Cowley
acowley@seas.upenn.edu

GRASP Laboratory
University of Pennsylvania
Philadelphia PA, USA

Abstract

This paper describes a feature based approach to segmenting images into coherent regions. The method draws inspiration from earlier work on randomized projection schemes for approximate nearest neighbor computation. The method proceeds by first computing a descriptor vector for each of the pixels in the image. These vectors are then randomly hashed to yield binary vectors. Salient clusters in the hash space are automatically identified by considering the populations associated with various hash codes. Since the method avoids the explicit vector distance computations associated with other methods, it is very amenable to fast implementation. Experimental results are presented on standard data sets.

1 Introduction and Related Work

Segmentation, the problem of breaking an image into coherent regions is, of course, a fundamental problem in Computer Vision. This paper proposes a new approach to the segmentation problem that leverages ideas developed in the Theoretical Computer Science literature to derive a new feature space based clustering algorithm that is amenable to real time implementation.

To date most of the approaches that have been developed to tackle the segmentation problem can be broadly divided into two groups. The first group consists of algorithms that view the image as a graph and use various metrics to measure the difference in appearance between neighboring pixels or regions. Once the problem has been formulated in this way, the algorithms center on the problem of dividing this graph into pieces so as to maximize coherence. The Normalized Cut algorithm developed by Shi and Malik [14] proceeds by recasting the graph segmentation problem in terms of a spectral analysis. This approach involves computing the distance between the pixels in the image and then solving a series of large but sparse eigenvector problems.

Felzenszswalb and Huttenlocher [6] proposed an efficient approach to grouping pixels in an image by making use of a spanning tree and showed that locally greedy grouping decisions can yield plausible results. This approach also revolves around the computation of multiple pairwise distance values. The method proposed in this paper avoids the computational costs associated with distance computation in favor of a randomized hashing approach which relies upon the locality preserving properties of the hashing function.

The second broad class of segmentation schemes are termed feature based methods since they proceed by associating a feature vector with each pixel in the image [2, 9, 17]. The entries in this feature vector characterize salient properties of the region surrounding that pixel such as color, texture or frequency content. Once all of the pixels have been mapped to the feature space, the segmentation process is treated as a clustering problem where the goal is to identify salient clusters in the population of feature vectors. The approach proposed in this paper falls into this second category.

One of the most commonly employed clustering methods is the venerable k-means algorithm which seeks to divide the population into k-clusters using an Expectation Maximization approach [1]. A key issue that one needs to address in applying this algorithm to segmentation problems is the question of choosing an appropriate value for k which is typically not known beforehand. A second issue is the fact that the k-means scheme involves repeated rounds of distance computations. This means that the computational complexity grows with the number of pixels, the dimension of the feature space and the number of clusters. Various approaches have been proposed to mitigate this problem including the method developed by Elkan [5] which seeks to accelerate the process by invoking the triangle inequality. Locality Sensitive Hashing schemes[8] have also been suggested to accelerate the search for near neighbors in the feature space. This accelerates but does not eliminate the distance computations required.

The Mean Shift segmentation algorithm proposed by Comaniciu and Meer [2] has been used very successfully to subdivide color images into regions. This feature based approach proceeds by searching for modes of the distribution in the feature space using a Parzen Window based approach. The method involves tracing the paths of various feature vectors as they evolve under the mean shift rule. This non-parameteric estimation scheme can be very time consuming which makes it less useful in situations where real time response is desired. The Parzen Window density estimation scheme employed in this approach also limits the dimension of the feature spaces to which it can be applied effectively. In contrast the method proposed in this paper can be applied to arbitrary feature spaces and has been implemented in real time on modest hardware.

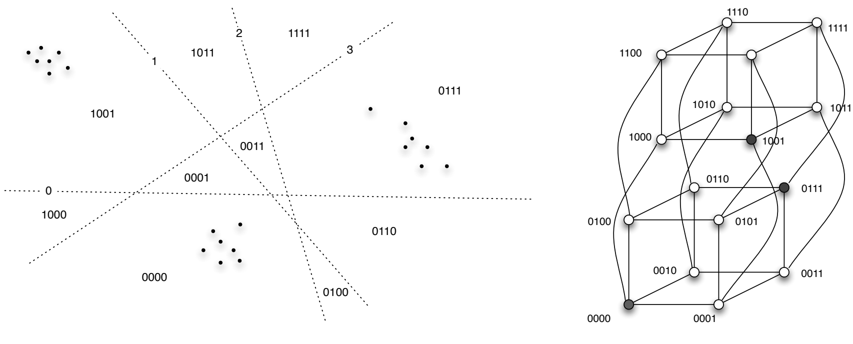
When labeled image data is available, effective algorithms that learn how to classify pixels and segment images have been proposed by a number of researchers including Shotton et al. [15] and Maire et al. [10]. These approaches can leverage the training data to associate semantic labels with pixels and segments. The proposed method works at a lower level and does not make use of any training data.

The remainder of this paper is organized as follows Section 2 discusses the technical approach taken to the segmentation problem in this work. Section 3 describes results obtained by running the proposed method on the imagery in the Berkeley Segmentation Database. Finally Section 4 discusses some of the conclusions drawn from the work and describes further avenues of research.

2 Technical Approach

The segmentation scheme described in this paper employs a feature based approach. Each pixel in the image is described by a feature vector which encodes a set of properties used to describe that pixel. In all of the experiments described in this paper we employ a simple color descriptor vector but one could equally easily use more sophisticated feature vectors such as a histogram of color values or a vector of texture coefficients.

Given this set of feature vectors, the goal of the segmentation procedure is to divide them into a set of clusters which capture the most salient groupings in the distribution. To do



(a) (b)

Figure 1: This figure depicts a simplified 2D version of the randomized hashing scheme. Figure (a) depicts a 2D feature space fractured into regions by a set of randomly chosen splitting planes. Each region is associated with a hash code indicating where it falls with respect to the splitting planes. The set of all hash codes can be associated with the vertices of a hypercube as shown in Figure (b) here the shading of the nodes indicates how many feature vectors are hashed to that code. The segmentation scheme proceeds by identifying local maxima in this hash code space.

this, the scheme employs a series of randomly chosen splitting planes. Figure 1 shows a simplified view of this procedure in two dimensions. Here the random splitting planes are used to hash the feature vectors into a set of disjoint cells based on their location.

The notion of randomized hashing has been employed before most notably by Indyk and Motwani in the context of Locality Sensitive Hashing [8]. These authors used a similar approach to hash a set of vectors into a set of discrete bins in order to accelerate the search for nearest neighbors. Their approach leveraged the fact that this randomized hashing procedure tends to preserves locality so points that are near to each other in the feature space are hashed to the same bin with high probability.

The proposed segmentation scheme leverages the same phenomenon for a different purpose - namely to cluster the feature vectors into groups. Returning to Figure 1 we note that the n spitting planes fracture the feature space into a set of 2^n disjoint convex cells each of which corresponds to an n-bit hash code. More specifically, each vector in the feature space $\mathbf{v_j}$ is assigned an n-bit hash code where the ith bit in the code, b_{ij}, is derived from the ith splitting plane as follows $b_{ij} = (\mathbf{v_j} \cdot \mathbf{u_i}) > \mathbf{s_i}$ where $\mathbf{u_i}$ denotes the normal associated with the ith splitting plane and s_i denotes the corresponding splitting value. Neighboring cells in the feature space differ by a single bit so the Hamming distance between the codes provides some indication of the distance between vectors in the feature space. More generally we can construct a correspondence between the set of all possible hash codes and the vertices of an n-dimensional hypercube. The topology of the hypercube reflects the structure of the feature space since neighboring cells in feature space will correspond to neighboring vertices in the hypercube.

For each of the hash codes the clustering procedure records how many feature vectors are mapped to that code. We expect that clusters in feature space will induce population maxima in the code space. That is, if we consider the hypercube as a graph we would expect to observe that some of the hash codes have a greater population than their neighbors. This motivates us to replace the original problem of clustering vectors in the feature space in favor of the simpler problem of looking for population maxima in the code space graph.

For every populated code in the hypercube the algorithm interrogates all of the codes that differ from the current code by k bits or fewer. This parameter, k, is referred to as the Hamming Distance Threshold. If the code under consideration has a population greater than all of its neighbors it is declared a local maxima and a cluster center. In this way the number of clusters recovered by the procedure is determined automatically based on the data as opposed to being imposed a'priori as in k-means. Note that this scheme can be used to distinguish up to $2^{(n-k)}$ local maxima.

The normals associated with the splitting planes, $\mathbf{u_i}$, are chosen randomly. The splitting values, s_i, are chosen by considering the distribution of the projected values, $(\mathbf{v_j} \cdot \mathbf{u_i})$. Some reasonable choices include the mean of the distribution, which corresponds to casting all of the splitting planes through the centroid of the distribution, the median value, and the value midway between the maximum and minimum projected values. All of these schemes tend to produce similar results in practice.

After the local maxima have been identified, each of the feature vectors is labeled with the hash code of the closest local maxima based on the Hamming Distance. In the case where a feature vector is equidistant from two or more local maxima based on Hamming Distance the Euclidean distance between the feature vector and the mean cluster vector is used to break the tie and decide the label. Once each of the pixels has been labeled with the index of its local maxima, a connected components procedure is run to divide the image into coherent connected regions. The entire scheme is outlined below in pseudo-code.

Algorithm 1 Segmentation via Randomized Hashing

```
1: Hash each feature vector to an $n$-bit code using the $n$ randomly chosen splitting planes
2: Maintain a count of the number of feature vectors mapped to each hash code
3: Identify local maxima in the code space - these are the cluster centers
4: Assign each feature vector to the closest local maxima
5: Run connected components on the labeled pixels to identify coherent connected components.
```

The proposed scheme is similar in spirit to the Mean Shift segmentation algorithm which also seeks to identify modes in the distribution of feature vectors. Where the mean shift scheme uses a Parzen Window based scheme to estimate density in feature space, the proposed scheme uses randomized hashing to identify salient groupings of feature vectors. A distinct but related approach to using randomized hashing for clustering has been explored in the context of noun clustering by Ravichandran et al [13]. They also exploit the Hamming distance between randomized hash codes but search for clusters in their data in a different manner.

Like Locality Sensitive Hashing, the segmentation scheme makes implicit use of the Johnsson-Lindenstrauss theorem which justifies the use of random projection by bounding the distortion of the relative distances between the feature vectors induced by the projection process.

From a computational perspective the principal effort revolves around computing the hash codes which involves $O(nmN)$ operations where n denotes the number of projection directions, m denotes the dimension of the feature space and N denotes the total number of pixels or feature vectors. Note that the scheme avoids the explicit distance computations between the feature vectors that one uses in most agglomerative and k-means segmentation schemes in favor of randomized hashing.

In searching for the local maxima in the code space one can simply store the hash code populations in an array with 2^n entries. For each populated hash code the procedure involves interrogating on the order of $\binom{n}{k}$ neighboring codes. For example to run the local maxima detection algorithm on $n = 12$ dimensions with a Hamming distance threshold, $k = 2$, one would construct a table with $2^{12} = 4096$ entries and each hash code would have $\binom{12}{1} + \binom{12}{2} = 12 + 66 = 78$ neighbors. Typically many of the hash bins are empty which further simplifies processing. For larger value of n one could employ a binary tree data structure to store and query the contents of the hash table efficiently.

3 Experimental Results

In order to characterize the performance of the proposed segmentation scheme experiments were carried out using the Berkeley Segmentation Database [11] which contains 1633 manual segmentations of 300 color images. The manual segmentations provided by the users were compared with the segmentations produced by the algorithm using two different measures, the Global Consistency Error and the Rand Index. The Global Consistency Error (GCE) developed by Martin, Fowlkes, Tal and Malik [11]captures the difference between two segmentations in a single number between 0 and 1 where lower numbers indicate lower error. The measure was specifically designed such that if one segmentation is a refinement of the other the score will be zero. This is a useful feature since it accounts for the fact that human subjects often choose to segment scenes to various levels, however, it also implies that machine segmentations that are strongly over or under segmented can also yield very low GCE scores which can be misleading.

To provide a different but related perspective on the algorithm we also chose to record and report the Rand Index for each segmentation. This measure is commonly used in statistics to measure the quality of clustering algorithms [16]. In order to compute the Rand Index we consider every pair of pixels in the image and determine whether they are labeled consistently in the human and machine segmentations. That is, if the two pixels have the same label in the human segmentation they should have the same label in the machine segmentation and vice versa. The Rand Index represents the fraction of the pixel pairs that are labeled consistently in the two segmentations, values that are closer to 1 indicate better segmentations. Unlike the GCE the Rand Index will suffer if the machine segmentation is over or under segmented with respect to the human segmentation.

A series of segmentation experiments was carried out using feature spaces based on color information. In all of these experiments the color values were averaged over a square window of width w centered around each pixel. Increasing the size of this window increases the level of smoothing and leads to a coarser segmentation.

The first set of experiments that was carried out was designed to determine how the performance of the segmentation scheme varied as we varied the color space. Experiments were carried out using the standard RGB values, the HSV color space, the LAB color space and a color vector that concatenated the RGV and HSV values into a six dimensional color vector. These experiments were carried out using a randomly chosen subset of 150 of the

Color Space	GCE	Rand Index
RGB	0.2805	0.7327
HSV	0.2421	0.7527
LAB	0.2578	0.7351
RGBHSV	0.2014	0.7614

Table 1: Results of running the segmentation procedure using various color spaces

n	k	GCE	Rand Index
8	1	0.1952	0.7632
8	2	0.2511	0.7443
8	3	0.2450	0.7104
12	1	0.1652	0.7535
12	2	0.2250	0.7640
12	3	0.2482	0.7417
16	1	0.1005	0.7438
16	2	0.1670	0.7583
16	3	0.2236	0.7527

Table 2: Results of running the segmentation procedure using various values for the n and k parameters

segmentations in the database. The average GCE and Rand index values are reported. In all of these experiments the value of n was fixed at 12 the value of k was fixed at 1 and the value of w was fixed at 3. Table 1 summarizes the results of these experiments and indicates that the RGBHSV color space offers the best performance with respect to the two metrics.

The second set of experiments explored how the performance of the scheme varied as we varied the number of splitting planes, n, and the Hamming Distance Threshold used to find local minima, k. The experiments were carried out using the HSV color space on the same subset of 150 segmentations from the database. The mean GCE and Rand Index values were recorded for every combination of parameters and the results are summarized in Table 2. In practice increasing values of the n parameter provide more ways to distinguish between feature vectors and leads to over segmentation while increasing the k parameter decreases the number of local maxima detected in the code space and leads to under segmentation. It is important therefore to strike a balance between these two parameters to achieve the desired effect.

A third set of experiments was carried out to investigate how the performance of the scheme varied as the window size parameter, w, was varied. The experiments were carried out using the HSV color space with the n and k parameters fixed at 12 and 1 respectively. Table 3 contains the results of these trials. Increasing the value of w leads to increases the level of smoothing which typically leads to under segmentation.

A fourth experiment was run to compare the results of the automated segmentation procedure to each of the 1633 human segmentations in the database. For this experiment the HSV color space was employed, the number of splitting planes, n was 12, the Hamming Distance threshold, k was 2 and the window size, w was 3. These parameter values were chosen to produce a visually pleasing over segmentation of the images rather than to optimize the GCE or Rand Index values. Over the entire database the mean GCE value was 0.2235 and

w	GCE	Rand Index
3	0.1219	0.7479
5	0.1350	0.7494
7	0.1477	0.7513
11	0.1660	0.7520
21	0.1992	0.7537

Table 3: Results of running the segmentation procedure using various values for the size of the smoothing window, *w* in pixels.

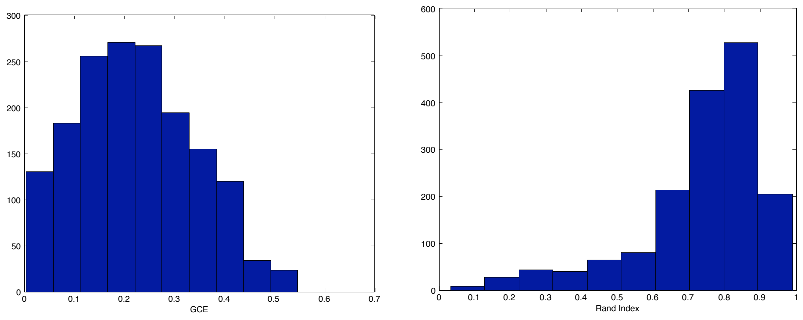


Figure 2: The first graph on the left indicates the distribution of the GCE values over all of the segmentations in the database while the graph on the right denotes the distribution of the Rand Index values.

the median GCE value was 0.2157 the mean Rand Index value was 0.7370 and the median Rand Index was 0.7833.

Figure 2 shows histograms of the distributions of the GCE and Rand Index metrics over the entire data set. Figure 3 shows a few of the images from the data set along with the human segmentation and the machine segmentation.

As a point of comparison we reproduce the table provided by Vazquez, van de Weijer and Baldrich who report on the performance of a few well regarded segmentation algorithms including the Mean Shift Algorithm, the Ridge Based Distribution Analysis Method and Normalized Cuts. Note that the segmentation method proposed in this paper is comparable to all of these methods with respect to the reported GCE values.

Figure 4 provides a direct comparison of segmentations produced by the proposed methods with those produced by the Mean Shift procedure [2] for a few randomly chosen images in the data set.

3.1 Real Time Implementation

A significant advantage of the proposed segmentation scheme is that the computational effort required scales linearly in the number of pixels and the operations required are simple and regular. In order to demonstrate this a real time version of the scheme was implemented on a Macbook Pro laptop computer. This implementation was used to segment 640 by 480 video frames at a rate of 10 frames per second using a single core of an Intel Core 2 Duo processor running at 2.33 GHz. This rate includes the time taken for all phases of the algorithm,

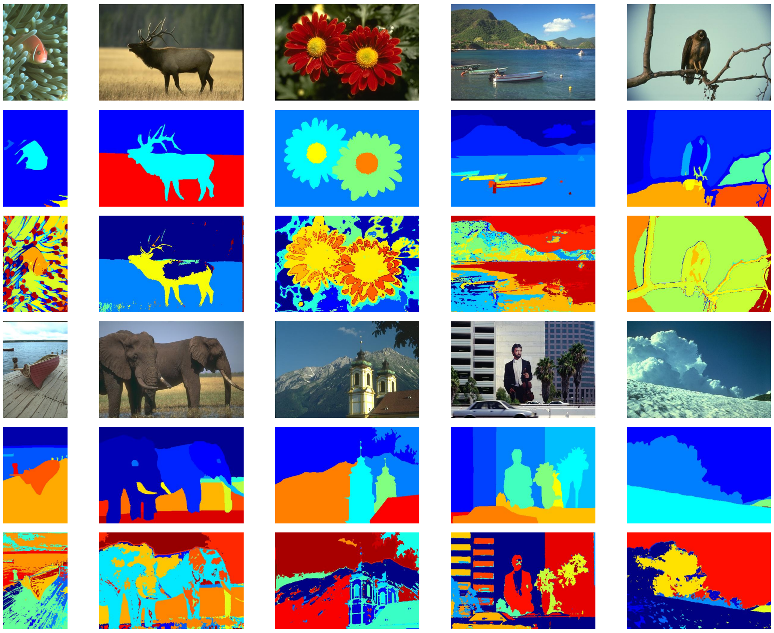


Figure 3: This figure compares the output of the automated segmentation procedure to human labeled segmentations. The first and fourth rows contain the input imagery, the second and fifth rows contain human segmentations while the third and sixth rows contain machine segmentations.

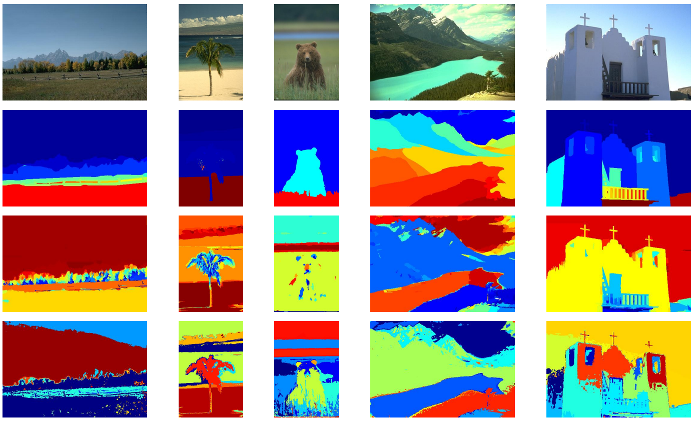


Figure 4: This figure compares the output of the proposed segmentation scheme with the results obtained using the Edison segmentation tool. The first row corresponds to the input image, the second to a human segmentation, the third to the mean shift result and the fourth to the randomized hash result. The parameters used for the Edison tool were $(h_s, h_r, M) = (7, 6.5, 20)$ and the parameters used for the randomized method were $(n, k, w) = (12, 2, 3)$.

Method	GCE
human segmentation	0.080
RAD [17]	0.205
seed [12]	0.209
Learned Affinity [7]	0.214
Mean Shift [2]	0.260
Normalized Cuts [14]	0.336

Table 4: Global Consistency Error scores for several state of the art methods as reported in [17] Including: seed [12], Learned Affinity [7], Mean Shift MS, [2] Normalized Cuts [14] and Ridge Based Distribution Analysis, RAD [17].

image acquisition, randomized hashing, local maxima detection and connected components processing. Since almost all of the steps in the procedure are embarrassingly parallel, the algorithm is a well suited to implementation on modern multi-core processors and GPUs and should be amenable to further acceleration.

4 Conclusion

This paper describes a new approach to segmenting natural images which leverages the idea of randomized hashing. The procedure aims to replace the problem of finding clusters in the feature space with the problem of finding local maxima in a graph whose topology approximates the geometry of the underlying feature space. In so doing the method bypasses the computational effort associated with computing distances between feature vectors which comprises a significant fraction of the effort in other techniques such as k-means clustering and mean shift segmentation.

The method is controlled by a few important parameters namely, the number of random splitting planes, n, The Hamming Distance threshold, k, and the window size that is used to average the color vectors, w. By adjusting these parameters the algorithm can be made to produce over segmentations or under segmentations of the input imagery. Importantly the number of segments that are produced is implicitly controlled by these parameters rather than explicitly provided as an input to the algorithm. The procedure produces segmentation results which are comparable to other state of the art methods and experiments have been presented which indicate how the performance changes as the principal parameters are varied.

The proposed algorithm is highly parallelizable and can be implemented in real time on modest hardware. This is an important advantage since it means that the method could be used as a cheap preprocessing step in a variety of image interpretation applications much as edge detection is used today. We could imagine using the method on a mobile robot to produce a fast, rough segmentation of the scene into sky ground, road and tree regions. Similarly, the scheme could be used as part of the loop in real time tracking applications where it would allow the system to automatically delineate targets. Ultimately we would envision such a real time segmentation scheme being used as a pre-processing step which would suggest possible groupings in the image to higher level interperetation algorithms. In this way the system would be able to focus its attention on regions based on their, size, shape, texture or position in the image.

While the segmentation algorithm has been discussed and implemented in the context of color descriptors, it could equally easily be applied to feature spaces with higher dimension. Future work will investigate how effectively the scheme will perform when applied to more sophisticated feature descriptors which will involve both color and texture values. In particular it would be interesting to determine whether increasing the dimension of the feature space helps or hinders the segmentation process. Many data interpretation problems suffer under the curse of dimensionality but there is some research [3, 4] to suggest that randomized projections of the data set can serve to separate and condition clusters which may lead to better performance.

References

[1] Morten Rufus Blas, Motilal Agrawal, Aravind Sundaresan, and Kurt Konolige. Fast color/texture segmentation for outdoor robots. In *IROS*, pages 4078–4085, 2008.

[2] Dorin Comaniciu, Peter Meer, and Senior Member. Mean shift: A robust approach toward feature space analysis. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 24:603–619, 2002.

[3] Sanjoy Dasgupta. Experiments with random projection. In *UAI '00: Proceedings of the 16th Conference on Uncertainty in Artificial Intelligence*, pages 143–151, San Francisco, CA, USA, 2000. Morgan Kaufmann Publishers Inc. ISBN 1-55860-709-9.

[4] Sanjoy Dasgupta. Learning mixtures of gaussians. In *FOCS '99: Proceedings of the 40th Annual Symposium on Foundations of Computer Science*, page 634, Washington, DC, USA, 1999. IEEE Computer Society. ISBN 0-7695-0409-4.

[5] Charles Elkan. Using the triangle inequality to accelerate k-means. In *International Conference on Machine Learning*, 2003.

[6] Pedro F. Felzenszwalb and Daniel P. Huttenlocher. Efficient graph-based image segmentation. *Int. J. Comput. Vision*, 59(2):167–181, 2004. ISSN 0920-5691. doi: http://dx.doi.org/10.1023/B:VISI.0000022288.19776.77.

[7] C. Fowlkes, D. Martin, and J. Malik. Learning affinity functions for image segmentation: combining patch-based and gradient-based approaches. In *Computer Vision and Pattern Recognition*, volume 2, pages II–54–61 vol.2, 2003.

[8] Piotr Indyk and Rajeev Motwani. Approximate nearest neighbors: towards removing the curse of dimensionality. In *STOC '98: Proceedings of the thirtieth annual ACM symposium on Theory of computing*, pages 604–613, New York, NY, USA, 1998. ACM. ISBN 0-89791-962-9. doi: http://doi.acm.org/10.1145/276698.276876.

[9] W. Y. Ma and B. S. Manjunath. Texture features and learning similarity. *Computer Vision and Pattern Recognition, IEEE Computer Society Conference on*, 0:425, 1996. ISSN 1063-6919. doi: http://doi.ieeecomputersociety.org/10.1109/CVPR.1996.517107.

[10] Michael Maire, Pablo Arbelaez, Charles Fowlkes, and Jitendra Malik. Using contours to detect and localize junctions in natural images. In *Computer Vision and Pattern Recognition, 2008. CVPR 2008. IEEE Conference on*, pages 1–8, June 2008.

[11] David Martin, Charless Fowlkes, Doron Tal, and Jitendra Malik. A database of human segmented natural images and its application to evaluating segmentation algorithms and measuring ecological statistics. In *in Proc. 8th Int'l Conf. Computer Vision*, pages 416–423, 2001.

[12] B. Micusik and A. Hanbury. Automatic image segmentation by positioning a seed. In *European Conference on Compuer Vision*, pages II: 468–480, 2006.

[13] Deepak Ravichandran, Patrick Pantel, and Eduard Hovy. Randomized algorithms and nlp: using locality sensitive hash function for high speed noun clustering. In *ACL '05: Proceedings of the 43rd Annual Meeting on Association for Computational Linguistics*, pages 622–629, Morristown, NJ, USA, 2005. Association for Computational Linguistics. doi: http://dx.doi.org/10.3115/1219840.1219917.

[14] Jianbo Shi and Jitendra Malik. Normalized cuts and image segmentation. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 22:888–905, 1997.

[15] J. Shotton, M. Johnson, and R. Cipolla. Semantic texton forests for image categorization and segmentation. In *Computer Vision and Pattern Recognition, 2008. CVPR 2008. IEEE Conference on*, pages 1–8, 2008.

[16] Ranjith Unnikrishnan, Caroline Pantofaru, and Martial Hebert. Toward objective evaluation of image segmentation algorithms. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 29(6):929–944, 2007. ISSN 0162-8828. doi: http://doi.ieeecomputersociety.org/10.1109/TPAMI.2007.1046.

[17] Eduard Vazquez, Joost Weijer, and Ramon Baldrich. Image segmentation in the presence of shadows and highlights. In *ECCV '08: Proceedings of the 10th European Conference on Computer Vision*, pages 1–14, Berlin, Heidelberg, 2008. Springer-Verlag. ISBN 978-3-540-88692-1. doi: http://dx.doi.org/10.1007/978-3-540-88693-8_1.