Proceedings of the Institute of Acoustics

BOUNDARY DETECTION AND SUPERPIXEL FORMATION IN SYNTHETIC APERTURE SONAR IMAGERY

J. T. Cobb Naval Surface Warfare Center Panama City Division, Panama City, FL, USA
A. Zare University of Missouri, Columbia, MO, USA

1 INTRODUCTION

Synthetic aperture sonar (SAS) strip map systems produce finely-detailed seafloor images with constant across-track resolution. The fine seafloor resolution of SAS images enables advanced image segmentation techniques previously applicable only to high-quality photographic images. In this paper we develop techniques to automatically detect and localize SAS image region boundaries using texture and intensity gradient operators combined with a superpixel merging algorithm.

In solving the boundary detection problem we seek to automatically separate different seabed environments into distinct homogeneous regions. In remotely-sensed data such as SAS strip maps, this problem is difficult due to several reasons. First, the transition between seabed environments is rarely distinct. For example, in regions that separate flat hard pack sand from sand ripple fields, there is usually a few meters of transition between the two regions. As the ripple amplitude gradually increases in the transition region, local texture and brightness (or intensity) measures will be blended between the two adjoining distinct regions, confusing parameter estimates used for separation. Second, a horizontal raster line in the SAS strip map is formed over a range of grazing angles to the seafloor. So, although pixel resolution is constant as a function of range due to SAS processing, the effects of shadowing are range-dependent and can change region image statistics appreciably. Finally, high-frequency SAS images are grayscale and contain no color information to aid in boundary separation.

Previous work on sonar image boundary detection includes approaches based on level-sets[1], statistical snakes[2], and Markov random field modeling[3], among other approaches[4,5]. Here we apply linear combinations of oriented brightness and texture gradients as our primary method for distinguishing distinct regions across boundaries. This approach has been applied successfully to boundary detection problems in color and grayscale photographs[6]. Unlike the referenced approach[6] which relies on training a regression model on a set of candidate images, we develop an unsupervised merging algorithm that determines final region boundaries by combining over segmented miniature image regions called superpixels[7].

Here we exploit a commonly used superpixel formation scheme[8,9] to define region boundaries in SAS images with distinct seabed environments such as hard pack sand, mud, sea grass, rock, and sand ripple. Superpixel boundaries are determined by oriented intensity and texture gradients fed into a spectral clustering technique known as the K-way normalized cuts algorithm. Superpixels are then merged based on their intensity and textural similarity using a pdf distance metric known as the Jensen-Shannon divergence. The merged superpixels form regions on a coarser scale that separate homogeneous regions of different seabed environments. These larger over segmented regions are then suitable for large-scale automated SAS survey labeling. We demonstrate the effectiveness of the boundary detection algorithm on a labeled set of 130 SAS images containing a variety of commonly encountered seabed environments.

2 BOUNDARY DETECTION ALGORITHM

The boundary detection algorithm presented here will draw upon several recent advances in image processing research. First, the features (or discriminatory information) that separates the different

Fig. 1. Block diagram of the boundary detection algorithm.

image regions are a combination of both texture and brightness information[6,10]. This information is combined via histogram binning and these histograms provide the elemental level of similarity between like regions (or difference between contrasting regions). A kernel distance metric encodes the similarity into an entry in an affinity matrix. A spectral graph-cut method quantizes the pixel level information into superpixels. The superpixels are then merged based on their brightness and texture information content to yield the final region boundaries for the image. The steps for histogram formation, similarity encoding, and spectral graph-cuts were adapted from freely available MATLAB® and C-code[11]. Fig. 1 is a block diagram of the processing steps detailed in this section.

2.1 Image Preprocessing and Normalization

We take some initial preprocessing steps to reduce dimensionality, condition the images for filtering, and ensure that pixel dynamic ranges are similar between images in the survey. First the SAS amplitude images are down sampled in row and column direction, then normalized to a common background mean by subtracting the minimum image pixel value and dividing each image pixel by a mean pixel level,

$$, \qquad (1)$$

where the mean pixel value is calculated from a region in the image with low contrast, typically a homogeneous sandy region. Images are normalized in this manner to obtain an ad hoc calibration of background reverberation between images so that similar seabed environments have the same mean backscatter energy.

Following normalization, images are despeckled with a simple median filter and transformed by the logarithm function. Taking the logarithm of each despeckled image is a way to decouple the mean from the variance in the single-point intensity or amplitude pixel statistics[12]. The log image is scaled so pixel values are within the range (0,1) prior to computing brightness and texture histograms.

2.2 Brightness and Texture Histogram Generation

Here we choose brightness and texture content within a spatial neighborhood as discriminatory information in the segmentation algorithm. The brightness and texture content is binned into a histogram and differences between histograms, *i.e.* gradients, form the elemental distance metric between neighboring regions. The normalized image pixel brightness values in a half-disc centered on each pixel is binned into a histogram composed of 32 bins with centers evenly distributed between the minimum and maximum pixel values of the image. Texture information is encoded for each pixel via *texton* generation.

Textons are used to encode a superposition of Gabor filter responses centered on a pixel of interest[10,13]. After filtering each normalized N x M image with a D-sized filterbank of various orientations, the N x M x D filter responses are quantized using a Dirichlet process clustering algorithm. This algorithm is designed to function like the K-means algorithm except that the number of clusters or the K value is determined automatically. A description of the algorithm is included in

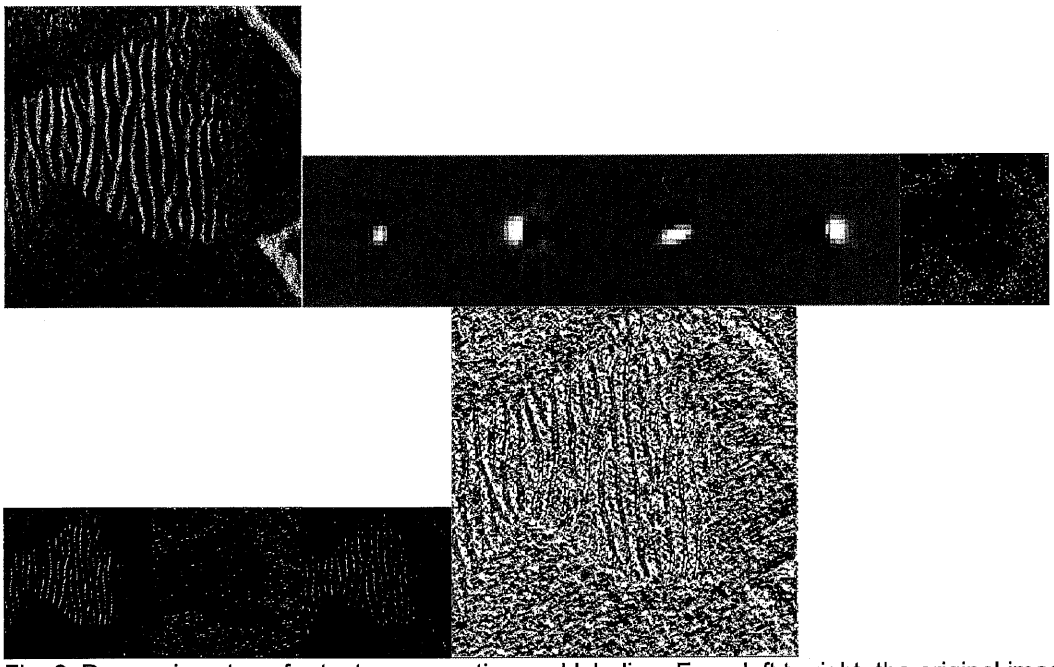


Fig. 2. Processing steps for texton generation and labeling. From left to right, the original image N x M is convolved with a D-dimensional filterbank. After filtering, the N x M x D filter coefficients are clustered using a DP clustering algorithm. The centroids from the clustering algorithm form texture primitives called textons. The texton map pixel values are encoded with the label of the texton that is closest to the D filter coefficient vector at that pixel location. The final texton map on bottom right of the processing stream depicts the spatial regularity of the texton labels for the different textures in the original image.

the appendix in Section 5. The centroids of these clusters are the linear superposition of filter responses and can be visualized as a texture primitive. Following texton generation, each pixel is assigned the label of the closest texton by minimum Euclidean distance between the filter response at the pixel value and the cluster centroid. The texton pixel labels are binned over a half-disc centered at each pixel in the image. Disc size choice in texture histogram binning (as well as brightness histogram binning) entails a tradeoff between boundary accuracy and accurate description of large textural features. The number of bins in the texture histogram is the texton label cardinality following DP clustering. Fig. 2 shows processing steps for texton generation and labeling.

Previous work[5] attempted to optimize the filters in the filterbank to represent typical SAS image textures. In this work we choose a simpler filterbank[6], using two sets of six rotated even- and odd-symmetric 3:1 length-ratio filters rotated at 30° intervals and a single center surround or Laplacian of Gaussian filters, for a total of 13. All filters have a very small support region, i.e. . Through trial and error we found that the smaller filters more accurately maintained sharp boundaries with only a small tradeoff in lower texton fidelity for large texture features such as large sand ripple fields.

Smoothed brightness and texture histogram differences are computed separately for half-disc regions on either side of each pixel location. These histogram differences are computed for 8 20° increments between 0° to 160°. The histogram difference for two half-disc regions (left and right) and centered on pixel location at rotation angle is computed using the distance operator

(2)

These difference arrays are smoothed using a 2-D Savitsky-Golay (or elliptical data fit) filter long the direction of . Other 2D smoothing techniques should also be effective at reducing noise imposed by the difference operation in (2). Each array is then normalized to lie in the range (0,1). We now assume that the arrays represent a brightness boundary probability and texture boundary probability. Assuming texture and brightness are independent phenomena, the total boundary

probability is simply . Prior to computing superpixels the boundary probabilities in the array are thinned by nonmax suppression in the direction of the maximum gradient.

2.3 Superpixel Formation

A superpixel is simply a local spatial neighborhood of connected image pixels that have a similar set of defining characteristics[7]. Using superpixels rather than the individual pixels in the image to define region characteristics has several advantages. A properly defined superpixel retains all of the important properties of the underlying object or distinct region, yet has a smaller dimension. This quantization is important in SAS strip map imagery where data dimensions are particularly large, typically on the order of 4 x 10^6 pixels per image. In addition to dimension reduction, the spatial grouping constraints produce clean, natural-looking boundaries between superpixels of different textures or objects.

Here we will use the histogram differences generated in the previous section to compute a distance metric between pixels in the original SAS image. The distance metric is then recorded in an affinity matrix and the K-way spectral cuts algorithm is used to compute superpixel boundaries[8].

Spectral clustering algorithms are a popular tool for computer vision segmentation tasks[14]. These algorithms model image pixels as a weighted graph where the pixels are nodes and the edge weights are entries in the matrix . is called the affinity matrix and the larger the value of a particular entry, the closer two nodes are in terms of their respective distance metric. is usually a symmetric positive definite (psd) matrix, a special matrix form that plays a key role in the node partitioning

algorithm. The spectral clustering task is to separate into a disjoint set of K partitions or and . This

multiclass partitioning problem is formulated in graph notation as follows[15].

We define the links between node sets and using affinity matrix as

(3)

We define the degree of a node partition (total connections to all nodes) as

(4)

Using these two terms we define the linkratio, or node links between and normalized by the total connectedness of partition

(5)

Based on this problem definition an optimal K-way partitioning scheme should choose the set of K partitions that would maximize the link ratios between the node sets. We can write this objective function in matrix notation by choosing a binary partitioning matrix with values . The objective function in matrix notation is now

, (6)

where the notation represents the i^{th} column of .

Assuming is psd, various spectral clustering schemes solve for by recognizing that (6) is a Rayleigh quotient optimization problem and the Eigen decomposition of provides a set of eigenvectors that can be manipulated into a set of partitioning vectors. Yu[8] is the first to recognize

that the optimal K-way partitioning solution to (6) can be found by first scaling the partition matrix X,

then finding an optimal continuous solution, and finally iterating between discrete and continuous solutions to find an optimal binary partition matrix. We implement this version of the K-way normalized cuts here for superpixel formation.

To compute superpixels, we first must assign values to the entries of . First we determine a maximum neighbor distance between pixels that we will allow within the boundaries of a single superpixel. Recall that a desirable superpixel property is that they be connected within an isolated spatial region. The maximum neighbor distance provides the entry candidates for . In each entry of , we simply assign the maximum value of that lies along a line between neighboring pixels and or

. (7)

Thus the larger the probability of a boundary, the smaller the weight between the two pixels. After forming , the K-way partitioning scheme computes the image superpixels. As in many clustering algorithms, K must be specified here. Since we are oversegmenting the image at this stage we choose a K much larger than the usual number of regions we expect to find in a SAS image, *e.g.* K=40 or larger.

2.4 Superpixel Merging

Having found the superpixels, we now compute a larger scale brightness and texture merging metric to determine whether two superpixels should be merged into a larger segment. First, we concatenate pdfs of brightness and texton values computed by a nonparametric density estimator for each superpixel. A Parzen window estimator[16] is used to calculate the superpixel pdf as follows

, (8)

where is a suitable normalizing factor such that is a valid pdf and is a Parzen window of width .

A superpixel's neighbors are considered a candidate for merging if the relative entropy between the neighboring pdfs is small. Relative entropy (also known as the Kullback-Leibler divergence) measures the similarity between two superpixel pdfs by the operation

. (9)

Since the KL divergence is not a symmetric operator, *i.e.* in many cases , and we will interchange the order of operations between neighbors, we compute a symmetric divergence metric known as the Jensen-Shannon (JS) divergence[17]

. (10)

Using the superpixel pdfs and the JS divergence we employ a greedy algorithm to merge superpixels as follows:

1. Calculate a set of M superpixels for a SAS image, .
2. Choose a superpixel at random and compute the JS divergence between and its neighbors.
3. If the JS divergence between and a neighbor is less than some threshold, merge the superpixels.
4. Recompute the merged superpixel pdf.
5. Randomly select a different superpixel and repeat Steps 2-4.
6. After all superpixels have been visited, compute the average entropy of all merged superpixels

   , (11)

   where is the number of merged superpixels after Step 7, and is the length of .

This greedy algorithm is repeated many times with random starting points. The configuration with the smallest value for is chosen as the "best" boundary partition for the SAS image at that particular JS divergence threshold. We seek a small entropy change after merging because large entropy values are indicative of merges between unlike regions and a flattening or smoothing of the superpixel pdf. Correct merges should preserve the overall superpixel pdf.

3 EXPERIMENT AND RESULTS

The algorithm was tested on a set of 130 500 x 500 SAS images containing a variety of sand ripple, rock, coral, sand, mud, and seagrass seabed textures. The steps outlined above were used to produce region boundaries for each image using JS divergence thresholds . The maximum neighbor distance was set to 35 and the initial number of superpixels per image was set to 40, *i.e.* the K in the K-way normalized cuts algorithm. This K value is a tradeoff between boundary accuracy, more superpixels imply a finer over segmentation, and depending upon textural feature size, less superpixels may over segment sand ripple regions and prevent the region merging algorithm from combining similar regions. A half-disc radius of 16 pixels was chosen for histogram binning. The hyper parameters for the DP texton clustering algorithm were set to , , and for all images as well.

Fig. 3 depicts some selected results for the boundary detection algorithm on images with well-defined regions. The JS divergence threshold is noted below each figure to show the effect of conservative (small) and aggressive (large) superpixel merging. Fig. 3 (a-f), depict two SAS images with well-defined ripple fields against a background of sand that transitions to sea grass fields. The segmentation works fairly well for these two images for small values of , though some dark and bright sand regions are merged for large values. The boundary detection algorithm performs exceptionally well in Fig. 3 (g-i), where a sharply defined rock field (upper right) is accurately segmented against a sandy region (lower left).

Fig. 4 depicts some selected results on images with diffuse boundaries and mixed seabed region labels. Fig. 4 (a-c), show a sand ripple field that gradually morphs, from right to left, into a smooth sandy region. Despite this diffuse transition area the boundary detection algorithm produces a reasonable boundary. The images in Fig. 4 (d-f) contain a mixture of rock outcroppings, featureless sand, and eroded sand ripples. The slightly over segmented images for and contain plausible boundaries of the different image regions. In Fig. 4 (g-i) a flat sand region is bisected by a sea grass field. Despite the similar textures between the two regions, a clear boundary is drawn between the sand and sea grass on the left-hand side of the image. At the sea grass and sand textures begin to blend on the right-hand side of the image.

Fig. 5 illustrates the results in terms of a receiver-operator characteristic (ROC) curve for increasing values of . To ground-truth the boundaries in the sample images, an analyst hand-labeled the boundaries with image processing software. These boundaries were then "grown" to a width of 10 pixels using a morphological dilation operation. A true positive was recorded if a boundary pixel produced by the algorithm intersected any part of the ground truth boundary images. False positives were recorded for all boundary pixels that lay outside the ground truth boundaries. False positive results are presented in units of pixels per image. It is clear in the ROC there is a traditional tradeoff

(a) (b) (c)

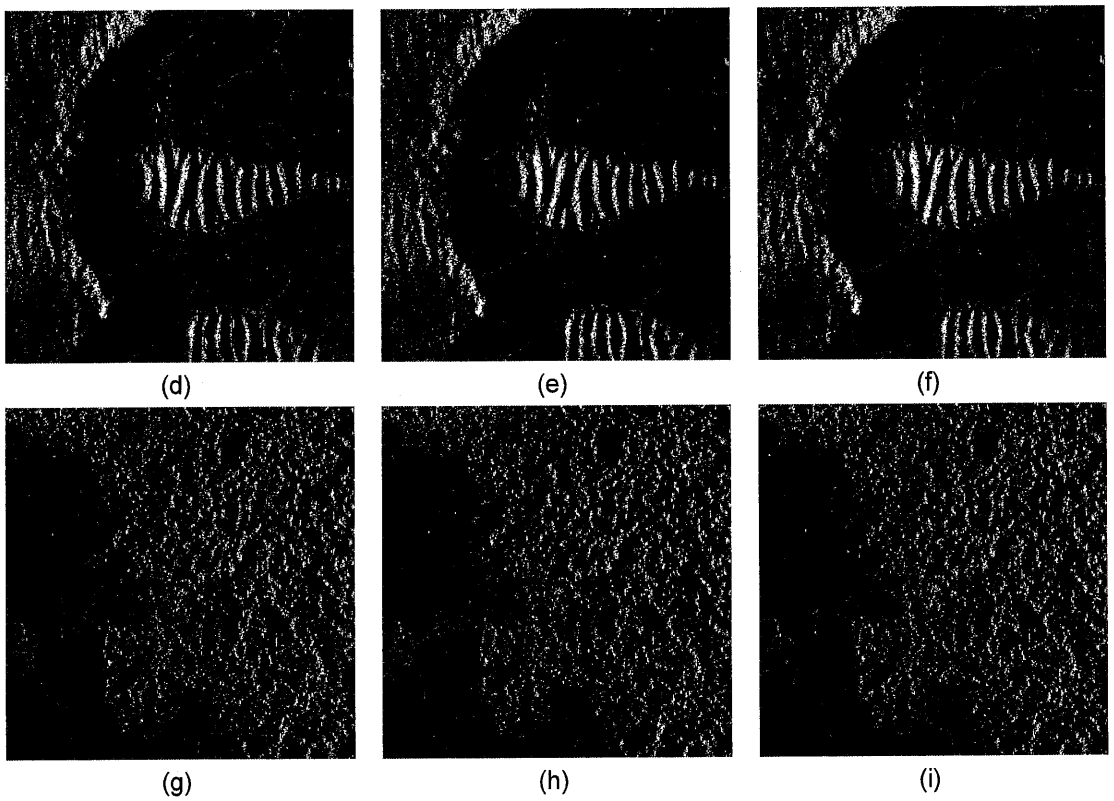
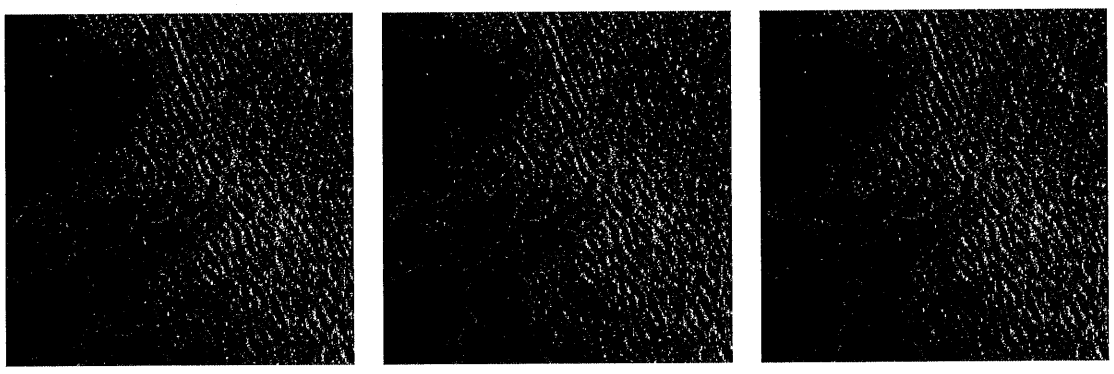
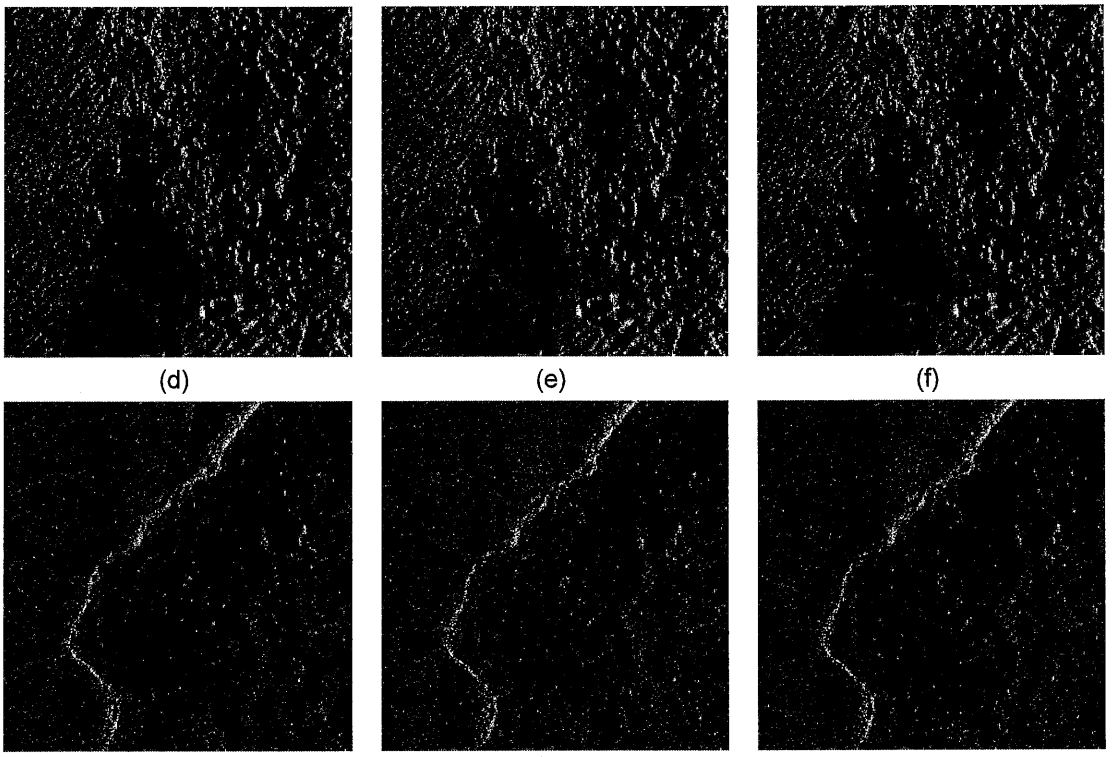


Fig. 3. Selected results for the boundary detection algorithm on images with well-defined regions. The JS divergence threshold is noted below each figure to show the effect of conservative (small) and aggressive (large) superpixel merging. Fig. 3 (a-f), depict two SAS images with well-defined ripple fields against a background of sand that transition to seagrass fields. The segmentation works fairly well for these two images for small values of , though some dark and bright sand regions are merged for large values. The boundary detection algorithm performs exceptionally well in Fig. 3 (g-i), where a sharply defined rock field (upper right) is accurately segmented against a sandy region (lower left).

between false positive and correct detection rate. The cost of a false positive in this case is simply an over segmentation. So for follow-on image segmentation where additional merging algorithms may be employed, the over segmentation from the superpixel merging may be an acceptable tradeoff if the detected boundary is very accurate.





(d) (e) (f)

(g) (h) (i)

Fig. 4. Selected results on images with diffuse boundaries and mixed seabed region labels. Fig. 4 (a-c), show a sand ripple field that gradually morphs, from right to left, into a smooth sandy region. Despite this diffuse boundary the boundary detection algorithm produces an acceptable boundary. The images in Fig. 4 (d-f) contain a mixture of rock outcroppings, featureless sand, and eroded sand ripples. The slightly oversegmented images for and show plausible boundaries of the different image regions. In Fig. 4 (g-i) a flat sand region is bisected by a seagrass field. Despite the similar textures between the two regions, a clear boundary is drawn between the sand and seagrass on the left-hand side of the image. At the seagrass and sand textures begin to blend on the right-hand side of the image.

4 CONCLUSIONS AND FUTURE WORK

In this paper we presented an algorithm for detecting seabed region boundaries in SAS images. The approach first computes intensity features from the image pixel values and textural features using a novel texton clustering algorithm employing DP clustering. The intensity and texture features are used to compute superpixels, over segmenting the original image into many small

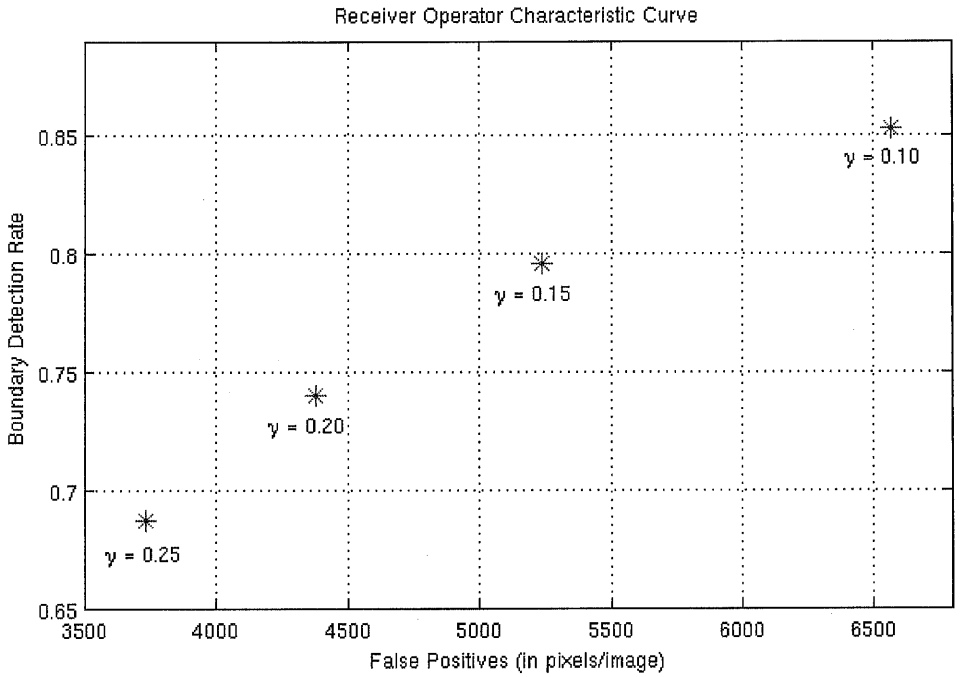


Fig. 5. ROC curve for varying values of . A true positive occurs when the detected boundary matches within 10 pixels of the ground truth boundary in the original SAS image. False positives are counted when detected boundary pixels do not match dilated ground truth boundary. Each SAS image is 500 x 500 pixels.

homogeneous regions. The superpixel feature pdfs are then merged based on the Jensen-Shannon divergence metric. The merged superpixels are the detected region boundaries and are suitable for follow on image processing tasks such as image segmentation (or co-segmentation) and environmental characterization.

We attempted to calculate the performance of the boundary detection code using hand-drawn boundaries around plausibly separate regions of the image as ground truth. This is inspired by the segmentation benchmark dataset[18]. Unfortunately we do not have a large number of human subjects with access to the SAS data that could provide consensus ground truth boundary data. Future work will include more rigorous ground-truthing and comparative results with other image segmentation schemes.

5 APPENDIX – DP TEXTON CLUSTERING

Clustering is a common technique for quantizing information. The K-means algorithm is an extraordinarily popular method for clustering due to its intuitive, simple steps and predictable results[19]. The K-means algorithm is initialized with K cluster centers or centroids and iterates between two steps:

1. Associate data samples with cluster label denoted when minimizes , (12)
   where is the squared Euclidean distance. Simply put, in this step each data sample is labeled as being associated with the closest centroid.
2. Recompute 's using labels from Step 1.

The algorithm iterates between Steps 1 and 2 until convergence. Like many greedy algorithms, K-means can become stuck in local minima and can depend strongly on initial centroid starting

locations. Here we address a different problem, choosing a good value for K. In many applications the correct number of centroids is unknown *a priori*. We address the problem of deducing K by recasting the K-means algorithm within the framework of Dirichlet process (DP) clustering.

Given a probability model for the data samples, DP clustering algorithms can produce sample labels without prior knowledge of the number of clusters. Equation (12) implies that the underlying probability model for each cluster is multivariate Gaussian with an identity covariance matrix. DP mixture models assume data samples are drawn from mixture component distributions with mixing parameter drawn from a Dirichlet process with concentration parameter and base component distribution . Using this model we construct a DP mixture model with each data sample drawn from multivariate Gaussian component distributions with an identity covariance matrix. Additionally we choose a multivariate normal distribution for the base distribution because it is the conjugate prior of the the multivariate normal distribution of and simplifies some of the sampling calculations. We choose the gamma distribution for . The hierarchical DP probability model is summarized below:

, (13)

, (14)

, (15)

, (16)

. (17)

where The terms , , and are hyperparameters chosen to ensure uniformative priors for the centroids and the concentration parameter .

Data labels c_i are inferred through a series of Markov chain (MC) sampling steps using the model

described above. Here we implement a version of DP inference known as stickbreaking for a truncated DP[20] *(i.e.* $K \neq \infty$) that also samples for the concentration parameter in each iteration. This type of data modeling relies only indirectly on the prior specification of parameters (through the hyperparameters , , and and is commonly called non-parametric Bayesian modeling. The clustering algorithm is defined by the following steps:

1. Assign values to hyperparameters , , and . Initialize stick breaks and mixing parameters . Assign each a random label based on the mixing parameter distribution. Choose a maximum number of possible clusters , where
2. Draw a value for the concentration parameter based on the gamma distribution
   . (18)
3. Update centroids based on labels
   (19)
   where is drawn from with precision .
4. Assign label to based on the likelihood
   . (20)
5. Draw stick breaks from beta random distribution
   , (21)
   where denotes a beta random variable with parameters and , is the number of data samples in cluster , and means the number of data points with a cluster assignment label greater than .
6. Update mixing parameter values by
   . (22)
7. Draw the concentration parameter from the parameterized gamma distribution
   . (23)
8. Repeat Steps 3-7.

Since the MC sampling algorithm in essence quantifies the parameters of a probability model with random variables, the resulting cluster assignments will usually not converge to a fixed point. However, we find that after several hundred iterations the cluster assignments typically are somewhat stable and similar in results to K-means for the same cluster cardinality.

6 ACKNOWLEDGEMENTS

The authors graciously thank the Office of Naval Research, Code 321, for funding this research.

7 REFERENCES

M. Lianantonakis and Y. Petillot, "Sidescan Sonar Segmentation Using Texture Descriptors and Active Contours," *IEEE J. Ocean. Eng.*, vol. 32, pp. 744–752, Jul. 2007.

S. Reed, Y. Petillot, and J. Bell, "An Automatic Approach to the Detection and Extraction of Mine Features in Sidescan Sonar," *IEEE J. Ocean. Eng.*, vol. 28, no. 1, pp. 90–105, Jan. 2003.

M. Mignotte, C. Collet, P. Perez, and P. Boutheny, "Sonar Image Segmentation Using an Unsupervised Hierarchical MRF Model," *IEEE Trans. Image Process.*, vol. 9, no. 7, pp. 1216–1231, Jul. 2000.

T. Celik and T. Tjahjadi, "A Novel Method for Sidescan Sonar Image Segmentation," *IEEE J. Ocean. Eng.*, vol. 36, no. 2, pp. 186–194, Apr. 2011.

J. Cobb and A. Zare, "Multi-Image Texton Selection for Sonar Image Seabed Segmentation," *Proc. SPIE Defense and Security Symposium*, vol. 8709, Apr. 2013.

D. Martin, C. Fowlkes, and J. Malik, "Learning to Detect Natural Image Boundaries Using Local Brightness, Color, and Texture Cues," *IEEE Trans. Pattern Anal. Mach. Intell.*, vol. 26, no. 5, pp. 530–548, May 2004.

X. Ren and J. Malik, "Learning a Classification Model for Segmentation," in *9th IEEE Int'l Conf. on Comp. Vision (ICCV '03)*, vol. 1, Oct. 2003, pp. 10–17.

S. Yu and J. Shi, "Multiclass Spectral Clustering," in *Proc. Int. Conf. Computer Vision (ICCV '03)*, vol. 1, 2003, pp. 313–319.

G. Mori, X. Ren, A. Efros, and J. Malik, "Recovering Human Body Configurations: Combining Segmentation and Recognition," in *IEEE Computer Vision and Pattern Recognition (CVPR) 2004*, vol. 2, 2004, pp. 326–333.

J. Malik, S. Belongie, T. Leung, and J. Shi, "Contour and Texture Analysis for Image Segmentation," *Int'l J. Computer Vision*, vol. 43, no. 1, pp. 7–27, Jun. 2001.

G. Mori, "Superpixel Computation Code (in MATLAB and C)," 2006, http://www.cs.sfu.ca/mori/research/superpixels.

C. Oliver and S. Quegan, *Understanding Synthetic Aperture Radar Images*. Raleigh, NC: SciTech, 2004.

J. Malik and P. Perona, "Preattentive Texture Discrimination With Early Vision Mechanisms," *J. Opt. Soc. Am. A*, vol. 7, no. 5, pp. 923–932, May 1990.

Y. Weiss, "Segmentation Using Eigenvectors: A Unifying View," in *Proc. Intl. Conf. Comp. Vision (ICCV 1999)*, vol. 2, 1999, pp. 975–982.

S. Yu, "Computational Models of Perceptual Organization," *Ph. D. Thesis*, Carnegie Mellon University, Pittsburgh, Pennsylvania, May 2003.

C. Bishop, *Pattern Recognition and Machine Learning*. New York, NY: Springer, 2006.

D. Endres and J. Schindelin, "A New Metric for Probability Distributions," *IEEE Trans. Inf. Theory*, vol. 49, no. 7, pp. 1858–1860, 2003.

P. Arbelaez, C. Fowlkes, and D. Martin, "Berkley Image Segmentation Dataset," 2007, http://www.eecs.berkely.edu/Research/Projects/CS/vision/grouping/segbench.

R. Duda, P. Hart, and D. Stork, *Pattern Classification*, 2nd ed. New York, NY: John Wiley Sons, 2001.

H. Ishwaran and L. James, "Approximate Dirichlet Process Computing in Finite Normal Mixtures: Smoothing and Prior Information," *J. Comp. Graph. Stat.*, vol. 11, no. 3, pp. 508–532, 2002.