2009 Fifth International Conference on Intelligent Information Hiding and Multimedia Signal Processing

Region-Based Segmentation versus Edge Detection

Hassana Grema Kaganami and Zou Beiji
School of Information Science and Engineering
Central South University
Changsha, Hunan Province, P.R. China
hgkaganami@gmail.com
bjzou@vip.163.com

Abstract — **This paper, we will review the main approaches of partitioning an image into regions by using gray values in order to reach a correct interpretation of the image. We mainly compare the region-based segmentation with the boundary estimation using edge detection. Image segmentation is an important step for many image processing and computer vision algorithms while an edge can be described informally as the boundary between adjacent parts of an image. A formal definition is elusive, but edge detection is nonetheless a useful and ubiquitous image processing task. After comparing we have come to a conclusion that the edge detection has advantage of not necessarily needing closed boundaries and also its computation is based on difference. The region-segmentation in spite of improving multi-spectral images has the drawback of being applied only on closed boundaries. To reach the result of edge detection we have used the technique of performance metrics and Canny edge detection. We have applied Canny ground truth to acquire more features via displaying more details.**

Keywords-gray values; region-based segmentation; edge detection

I. INTRODUCTION

Image segmentation is an important step for many image processing and computer vision algorithms. Motivated by its application over a wide spectrum of topics, for example; analyzing different regions of an aerial photo is useful for understanding plant/land distribution. Extracting an object of interest from background of an image is important for building intelligent machines for factory automation systems. Segmenting and counting blood cells from cell images can help hematologists to improve diagnosis of diseases. Scene segmentation is also helpful to retrieve images from large image databases for content-based image retrieval systems [1, 2].

A region is a group of connected pixels with similar properties. It is very important in interpreting an image because it may correspond to objects in a scene. For correct interpretation, image must be partitioned into regions that correspond to objects or parts of an object. The partition into regions is done often by using gray values of the image pixels.

Thus, we will investigate and compare the two general approaches of image segmentation, i.e. Region-based segmentation and Boundary estimation using edge detection.

II. EDGE DETECTION

Edge detection is segmentation by finding pixels on a region boundary. Edges found by looking at neighboring pixels. It is also region boundary formed by measuring gray value differences between neighboring pixels.

Edge detection can be formulated as a binary classification problem at pixel level with a goal of identifying individual pixels as either on-edge or off-edge. To solve this classification problem we use both fixed and adaptive feature selection in conjunction with a support vector machine.

An edge can be described informally as the boundary between adjacent parts of an image. A formal definition is elusive, but edge detection is nonetheless a useful and ubiquitous image processing task. Different applications have different requirements for edges, and these requirements are usually informal as well on one hand, only a few broad strokes are required to capture the essence of an image, on the other hand intricate web of fine lines are required to capture every squiggle and fjord. This lack of formal definition has led to a huge variety of heuristic algorithms for edge detection, and makes it difficult to quantify their performance in a meaningful way.

A. Performance metrics

As long as edges are only informally defined, measuring the performance of an edge detector is necessarily subjective. One approach [3] employed a survey questionnaire in which humans rated the relative performance of various edge detection algorithms; obviously, there are problems with repeatability of this scheme, but it highlights the essentially subjective nature of the problem. Other authors used qualitative properties of the resulting edges like connectivity [4], or some task-oriented metric like object recognition (Shin et al., 1999). But most evaluation is based on agreement with manually determined ground truth, and this agreement is usually measured in terms of Receiver Operating Characteristics (ROC) curves [4, 5, 6, and 7]. When the output of an edge detector is compared to a ground truth value, a count of edge detections and false alarms can be obtained. Comparing edge detection methods based on only these quantities can be ambiguous. It is hard to make a trade-off between high detection rate and low false alarm rate. Thus, by adaptively exploring the parameter space of an algorithm and plotting the resulting detection and false alarm rates against each other, a ROC curve can be used to describe

978-0-7695-3762-7/09 $26.00 © 2009 IEEE
DOI 10.1109/IIH-MSP.2009.13

the performance of an edge detector more completely. For the ROC curves reported here, we sweep over the value of the threshold and compute detection rate and false alarm rate for each value.

B. *The Canny Edge detector*

The Canny edge detector is considered a standard benchmark against which edge detection algorithms are compared [8]. The Canny algorithm consists of four simple steps. First is smoothing by Gaussian convolution to reduce susceptibility to noise. Second, edge strength and edge directions are found by taking a 2-D spatial gradient of the image using the Sobel operator. The third step is non-maximal suppression, this uses edge direction to trace along the edge and suppress any pixel that is not considered an edge. The fourth step, hysteresis, aims at eliminating broken edges. It has two thresholds: high and low. Any pixel above the high threshold is automatically considered an edge. Any pixel between the high and low threshold that is adjacent to an edge pixel is also considered an edge.

Bowyer et al [9] uses Canny gound truth on their machine learning approach. They use Canny on the original South Florida images and apply their algorithm on the original South Florida dataset employing Canny as ground truth as a replacement to the ground truth provided with the original dataset. The implementation of Canny and the same parameter selection procedure has been used in the South Florida dataset. The comparison of adaptive feature selection and fixed feature set modes on the South Florida dataset which also includes the results from Konishi et al [7] on the same data are as shown in figure 1 below.

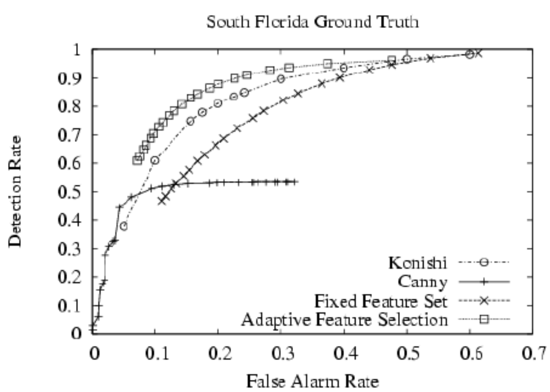


Figure 1. Comparison of adaptive feature selection fixed feature set.

Given that human marked up data is bound to have variability, we presumed that AFREET would do better on the Canny-derived ground truth than on the manual ground truth. However, when Canny is run on the South Florida images and the resulting image was used as ground truth, figure 2 shows that, the results contradict the findings in figure 1. It is also surprising that fixed mode was able to outperform adaptive mode above a certain threshold. Just to be sure that the results were not outliers caused by the particular subset of images, the experiment was performed three times on different sets of images and the results were the same.

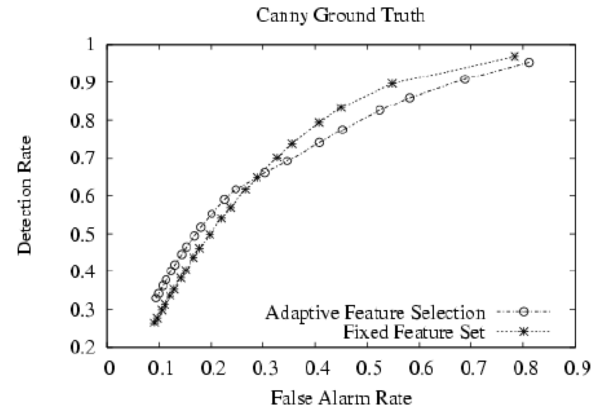


Figure 2. Canny applied on the south Florida image

The summary of the applications of various methods is shown below in figure 3.

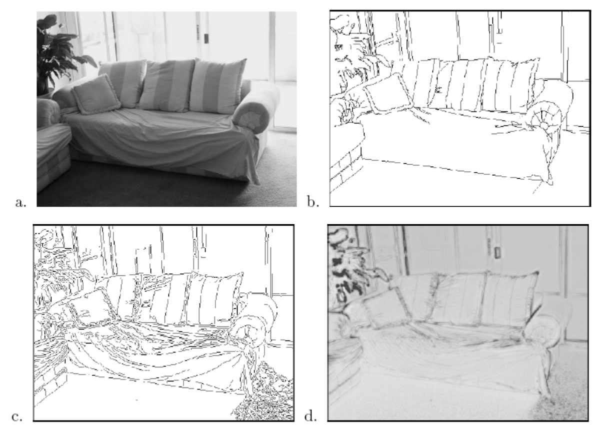


Figure 3. b, c and d are the results of applying various techniques on the original image a.

Applying the Original ground truth with the South Florida data set on the original image of a sofa gives us a brighter image of the sofa (see figure 3.b) while the application of Canny ground truth is allowing us to get more features, the Canny ground is displaying more details and finally the AFREET results trained on Canny ground truth gives us smoother image (see figure 3.d).

III. REGION-BASED SEGMENTATION

In the region-based segmentation, pixels corresponding to an object are grouped together and marked. The important principles are Value similarity (which includes Gray value differences and Gray value variance) and Spatial Proximity (which consists of Euclidean distance and compactness of a region). Region-based segmentation method also requires the use of appropriate thresholding techniques.

A. *Assumption*

Here points on same object will project to spatially close pixels on the image with similar gray values.

B. *Problem statement*

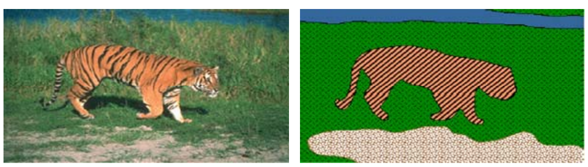
Given a set of image pixels *I* and a homogeneity predicate *P* (.), let us partition the image *I* into a set of n regions *Ri*, if equation (1) holds true, then all pixels of any given region satisfy the homogeneity predicate *P* (2). Also, any two adjacent regions cannot be merged into a single region (3).

$$\bigcup_{i=1}^{n} Ri = True \qquad (1)$$

$$\forall I, P(Ri) = True \qquad (2)$$

$$P(Ri \bigcup Rj) = False \qquad (3)$$

Figure 4 depicts a region segmentation result of an original image.



(a) (b)

Figure 4. a. Original Image b. Region Segmented Image**.**

C. *Thresholding*

We obtain a better segmentation by using correct thresholding. Most algorithms involve establishing a threshold level of certain parameter. Using samples of available image intensities, appropriate threshold are set automatically in a robust algorithm i.e. no hard-wiring of gray values. The automatic thresholding uses one or more of the following: Intensity characteristics of objects, Sizes of objects, Fractions of image occupied by objects, Number of different types of objects i.e., Size and probability of occurrence, most popular and Intensity distributions estimate by histogram computation.

1) *Thresholding method*

Some automatic thresholding schemes are P-tile method, Iterative threshold selection, Adaptive thresholding, Variable thresholding and, Mode method.

a) *P-tile method*

If object occupies P% of image pixels, then set a threshold T such that P% of pixels have intensity below T see figure below.

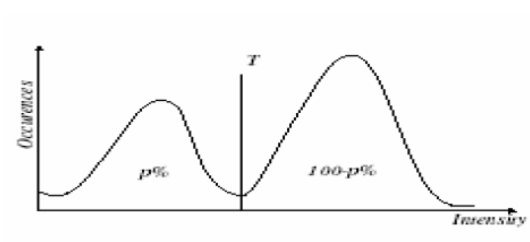


Figure 5. P-tile thresholding method

b) *Iterative method*

In this technique, approximate threshold are successively refined to get a new value so as to partition the image better (4) where $\mu 1, \mu 2$ are means.

$$T = \frac{1}{2}(\mu 1 + \mu 2) \qquad (4)$$

c) *Adaptive thresholding*

It is used in scenes with uneven illumination where same threshold value is not used throughout the image. In such case, look at small regions in the image and obtain thresholds for individual sub-images. Final segmentation is the union of the regions of sub-images.

d) *Variable thresholding*

It approximates the intensity values by a simple function such as a plane or bi-quadratic. It is called background normalization.

e) *Mode method*

Assume that gray values are drawn from two normal distributions with parameters $(\mu 1, \sigma 1), (\mu 2, \sigma 2)$ where $\sigma 1, \sigma 2$ are variances. If the standard deviations are zero, there will be two spikes in the histogram and the threshold can be placed anywhere between them. For non-ideal cases, there will be peaks and valleys and the threshold can be placed corresponding to the valley (see figure 6).

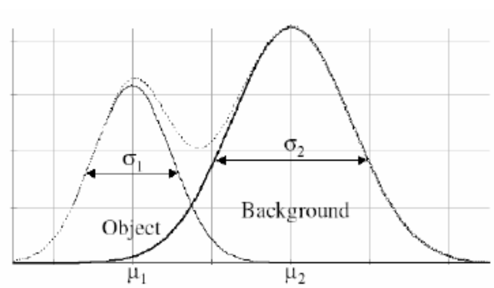


Figure 6. Mode method thresholding

An observation of the global application of histogram principles reveals a great drawback in complex scenes. Consider the two different scenes (figure 7) in which spatial information about intensity values are thrown away.

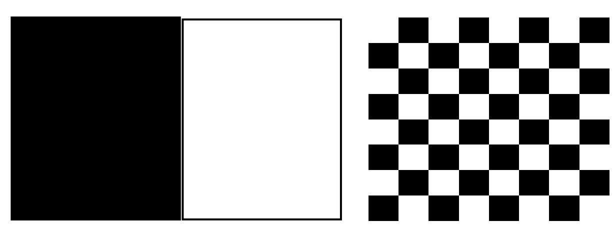


Figure 7. Two different scenes

Figure 7 shows two different scenes for which spatial information about intensity values is thrown away as shown in figure 8, entirely different scenes may give a strikingly similar histogram representation.

Figure 8. Histogram for both images

D. *Region representation*

We have different representations which are suitable to different applications and they can be mainly grouped in three general classes which are Array representation (i.e., same size arrays membership arrays), Hierarchical representation (i.e., pyramid and quad tree) and, Symbolic representation.

- Array representation

It uses array of same size as original image with entries indicating the region to which a pixel belongs. A typical example is the binary image shown in figure 9 and the array representation in figure 10

7	0	0	0	0	0	0	0	0
6	1	0	0	0	0	0	0	0
5	0	0	0	0	0	0	1	0
4	0	0	7	8	8	0	0	0
y3	0	0	7	8	7	0	0	0
2	0	0	0	7	8	0	0	0
1	0	0	0	0	0	0	0	0
0	0	0	0	0	0	0	2	0

Figure 9. Binary image Data

7	0	0	0	0	0	0	0	0
6	0	0	0	0	0	0	0	0
5	0	0	0	0	0	0	0	0
4	0	0	1	1	1	0	0	0
y3	0	0	1	1	1	0	0	0
2	0	0	0	1	1	0	0	0
1	0	0	0	0	0	0	0	0
0	0	0	0	0	0	0	0	0
	0	1	2	3x	4	5	6	7

Figure 10. Array Region Representation

Membership arrays (images) commonly called masks in which each region has a mask that indicates which pixels belong to which region. It has the advantage that a pixel can be allowed to be a member of more than one region. The Array Representation has the following characteristics: It preserves all details of regions required in most applications, it is a very popular method with lots of hardware support available and, its symbolic information is not explicitly represented.

- Hierarchical representation

Images are represented at many resolutions. As resolution, array size and some data is lost. Here it is more difficult to recover information. But, memory and computation requirements are also decreased. It is used to accommodate various segmentation algorithms which operate at different resolutions. They are mainly pyramid and quad tree.

In Pyramids we have, an $n \times n$ image represented by the image and 'k' reduced versions of the image. Pixel at level 'l' has combined information from several pixels at level 'l+1' Top level – level 0 – single pixel. The bottom level of the original image with a simplest method for resolution reduction is averaging. (Figure 11).

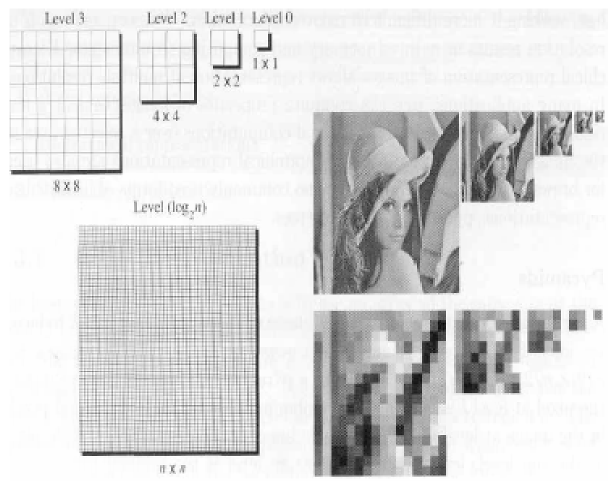


Figure 11. Pyramid structure

On the other hand, Quad tree method is an extension of pyramids for binary images. Three types of nodes white, black and, gray are used. The white or black node will not be split. The Gray node has its node split into 4 sub-regions. Each node is either a leaf or has 4 children (figure 12).

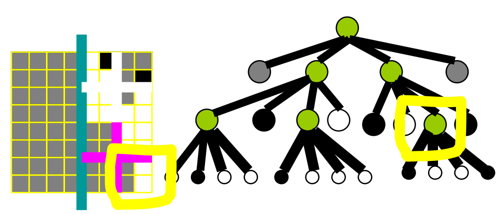


Figure 12. Quad Tree example: Binary image.

Figure 13 below shows the Quad Tree example for Non-binary image.

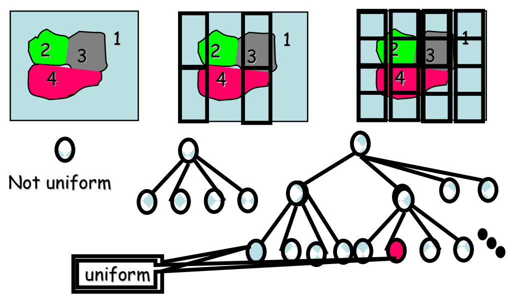


Figure 13. Quad tree for non binary image

- Symbolic representation

Here the most commonly used symbols are Enclosing rectangle, Centroid, Moments and, Euler number.

Region based segmentation is generally done on closed boundaries and it improves multi-spectral images segmentation while the computation is based on similarity.

IV. REGION-BASED SEGMENTATION VERSUS EDGE DETECTION

The following table 1 summarizes the differences between region-based and edge detection segmentations. At this point, we would like to infer that, Edge detection method has the advantage of not necessarily using closed boundaries even though it does not have a significant improvement for multi-spectral images. The computation here is based on difference rather than similarity as in the case of region-based segmentation method.

TABLE I. SUMMARY OF DIFFERENCES BETWEEN REGION-BASED AND EDGE DETECTION.

Region-based segmentation	**Edge detection**
Closed boundaries	Boundaries formed not necessarily closed
Multi-spectral images improve segmentation	No significant improvement for multi-spectral images
Computation based on similarity	Computation based on difference

V. CONCLUSION

Segmentation is an essential preliminary step in most scene analysis and automatic pictorial pattern recognition problems. Choice of suitable segmentation technique depends on peculiar characteristics of individual problems. Canny, seems able to find pixels near edges but is less able to find exact edges. Because edge detection is such a ubiquitous task, an adaptable edge detector has the potential to become a tool of widespread utility in the field of image processing. While the field of edge detection is well studied, we are only aware of one previous study that has utilized support vector machines for this purpose [10]. That study used a single image and no performance metrics beyond a visual comparison of their results with Canny. With dilation metric, the performance of our algorithm is comparable to the suite of state-of-art edge detectors in the study by Bowyer et al. The algorithm in Bowyer et al [9] works well works well using both pixel-to-pixels, neutral pixel, and dilation metrics emphasizes its adaptability, at least within the domain of edge width. In contrast, a fixed algorithm, such as Canny, seems able to find pixels near edges but is less able to find exact edges. Because edge detection is such a ubiquitous task, an adaptable edge detector has the potential to become a tool of widespread utility in the field of image processing. It is also possible that other low level image processing tasks, such as pixel interpolation or image re-sampling, can be made more adaptable by reformulating them as problems in machine learning.

ACKNOWLEDGMENT

This research is funded by: National Natural Science Foundation of China (Project No. 60673093), Hunan Provincial Natural Science Foundation of China (Project No. 07JJ3125), Major Program of National Natural Science Foundation of China (Project No. 90715043), Changing Scholars and Innovative Research Team in University (Project No. IRT0661) and National Natural Science Foundation of China (Project No. 60803024).

REFERENCES

[1] Y. Rui, T.S. Huang, S.F. Chang, Image retrieval: current techniques, promising directions and open issues, J. Visual Commun. Image Representation 10 (1999) 39–62.

[2] W. M. Smeulders, M. Worring, S. Santini, A. Gupta, R. Jain, Contentbased image retrieval at the end of early years, IEEE Trans. Pattern Anal. Mach. Intell. 22 (2000) 1349–1379.

[3] Heath, M., Sarkar, S., Sanocki, T., & Bowyer, K. (1997). A robust method for assessing the relative performance of edge-detection algorithms. IEEE Transactions on Pattern Analysis and Machine Intelligence, 19, 1338–1359.

[4] Zhu (1996). Efficient evaluations of edge connectivityand width uniformity. Image and Vision Computing, 14, 21–34.

[5] Shin, M. C., Goldgof, D. B., & Bowyer, K. W. (1999). Comparison of edge detectors using and object recognition task. Computer Vision and Pattern Recognition (pp. 360–365).

[6] Forbes, L. A., & Draper, B. A. (2000). Inconsistencies in edge detector evaluation. Computer Vision and Pattern Recognition (pp. 398–404). Hilton Head, SC.

[7] Konishi, S., Yuille, A., & Coughlan, J. (2003). Statistical edge detection: Learning and evaluating edge cues. IEEE Transactions on Pattern Analysis and Machine Intelligence, 25.

[8] Canny, J. F. (1986). A computation approach to edge detectors. IEEE Transactions on Pattern Analysis and Machine Intelligence, 8, 34–43.

[9] Bowyer, K., Kranenburg, C., & Dougherty, S. (2001). Edge detector evaluation using empirical roc curves. Computer Vision and Image Understanding, 84, 77– 103.

[10] Gomez-Moreno, H., Maldonado-Bascon, S., & Lopez-Ferreras, F. (2001). Edge detection in noisy images using the support vector machines. IWANN (1) (pp. 685–692).

[11] Jain R., Rangachar K., Schunk B. “Machine Vision” McGraw-Hill 1995

[12] Gonzalez R., Woods R. “Digital Image Processing” 2nd Ed. Pearson Education