ON BITSTREAM BASED EDGE DETECTION TECHNIQUES

Ivan Mecimore, William Fahrenkrog and Charles D. Creusere

New Mexico State University
Klipsch School of Electrical and Computer Engineering
Las Cruces, NM
Email: imecimor@nmsu.edu, williamf@nmsu.edu, ccreuser@nmsu.edu

ABSTRACT

We consider here edge detection as a problem of identifying points in an image where the image brightness changes sharply or has discontinuities. Specifically, we investigate several well-known algorithms for edge detection and modify them for performance directly on the bitstream of a JPEG compressed image. By performing the processing in the compressed domain, we dramatically reduce its complexity. The low complexity combined with the fact that it does not require JPEG decoding makes this approach well suited for distributed wireless sensor networks.

To extract information from JPEG compressed images, we can use the number of bits allocated to each of the 8x8 subblocks. Using this preponderance number, we are able to extract information regarding the frequency content of the image. In this paper we will present a comparison of first-order, differential, and phase congruency approaches to edge detection and determine how these algorithms perform when applied directly on the bitstream domain. The performance of the edge detectors will be determined by using the Figure of Merit (FOM) and Closest Distance Metric (CDM).

Index Terms— JPEG-based edge detection, sensor network coding, edge detection, bitstream processing

1. INTRODUCTION

Edge detection is a vital part of image processing, which is used for extracting important features from an image. In many cases, we are only concerned with the shape of an object, or recognizing some part of it. Edge detection is especially important for applications in computer vision. For our purposes, the main idea is to find areas where discontinuities exist. These discontinuities generally include step edges, line edges, and junctions [1].

Because of the importance of edge detection, researchers have already come up with many methods for finding the edges in an image. These methods include Sobel, Canny, Laplacian of Gaussian, and others. Moreover, nearly all methods of edge detection use a smoothing step. This step is performed in order to help minimize false edges due to noise, and other small details. Another step that is commonly seen in edge detection is edge labeling which involves touching up the image, suppressing false edges, and connecting edges [1] . We will be focusing on the actual process of detecting edges.

Our interest lies in applying edge detection methods to the bitstream domain. After JPEG encoding the number of bits used to encode each 8×8 subblock gives us an idea of the attributes of an image. This can have several advantages in image processing, because edge detection can be performed on a reduced image data set without completely decoding the bitstream. This can further be applied to distributed video coding applications very easily.

The proposed algorithm has been developed while exploring alternative methods to steps in [2]. In particular, we propose a method of edge detection to aid in identifying clusters of pixels that have similar frequency structure as shown in [2, 3]. These clusters can be used to identify areas of high correlation between multiple images. For example, these images can be a product of either a video sequence or spatially separated cameras with an overlapping field of view. In this application, emphasis is placed on reducing computational complexity as well as producing accurate edge maps in order to identify patterns in the two different images. Other applications may include extracting information from large collections of images, such as databases or the Internet.

2. BIT-DOMAIN EDGE DETECTION

The details of the JPEG compression algorithm are fully described in [4], but what is important in our application is that it compresses images using 8×8 blocks of pixels in a largely independent manner. This allows the information in each block to be represented by the number of bits necessary to encode it, known as "the preponderance". Consequently, the pattern created by the bit counts of the 8×8 pixel blocks that form regions of an image provide us with information about the spatial composition of those regions which can, in theory, be

Research supported by the ARO, contract # W911NF-06-1-0441

Report Documentation Page

Form Approved
OMB No. 0704-0188

Public reporting burden for the collection of information is estimated to average 1 hour per response, including the time for reviewing instructions, searching existing data sources, gathering and maintaining the data needed, and completing and reviewing the collection of information. Send comments regarding this burden estimate or any other aspect of this collection of information, including suggestions for reducing this burden, to Washington Headquarters Services, Directorate for Information Operations and Reports, 1215 Jefferson Davis Highway, Suite 1204, Arlington VA 22202-4302. Respondents should be aware that notwithstanding any other provision of law, no person shall be subject to a penalty for failing to comply with a collection of information if it does not display a currently valid OMB control number.

1. REPORT DATE **JAN 2009**	2. REPORT TYPE	3. DATES COVERED **00-00-2009 to 00-00-2009**
4. TITLE AND SUBTITLE **On Bitstream Based Edge Detection Techniques**		5a. CONTRACT NUMBER
		5b. GRANT NUMBER
		5c. PROGRAM ELEMENT NUMBER
6. AUTHOR(S)		5d. PROJECT NUMBER
		5e. TASK NUMBER
		5f. WORK UNIT NUMBER
7. PERFORMING ORGANIZATION NAME(S) AND ADDRESS(ES) **New Mexico State University,Klipsch School of Electrical and Computer Engineering,Las Cruces,NM,88003-8001**		8. PERFORMING ORGANIZATION REPORT NUMBER
9. SPONSORING/MONITORING AGENCY NAME(S) AND ADDRESS(ES)		10. SPONSOR/MONITOR'S ACRONYM(S)
		11. SPONSOR/MONITOR'S REPORT NUMBER(S)
12. DISTRIBUTION/AVAILABILITY STATEMENT **Approved for public release; distribution unlimited**		
13. SUPPLEMENTARY NOTES **Live-Virtual Constructive Conference, 12-15 Jan 2009, El Paso, TX**		
14. ABSTRACT **see report**		
15. SUBJECT TERMS		

16. SECURITY CLASSIFICATION OF:			17. LIMITATION OF ABSTRACT	18. NUMBER OF PAGES	19a. NAME OF RESPONSIBLE PERSON
a. REPORT **unclassified**	b. ABSTRACT **unclassified**	c. THIS PAGE **unclassified**	**Same as Report (SAR)**	**5**	

Standard Form 298 (Rev. 8-98)
Prescribed by ANSI Std Z39-18

used for image processing. We propose here to use the patterns formed by these bit allocations instead of the actual pixels to perform edge detection. A method for extracting the preponderance values from the bitstream is detailed in [2]. In addition to the spatial information, since the JPEG encoding step uses an entropy encoder, the preponderance values provide us with course estimates of the entropy of each 8×8 block of pixels. An example of the preponderance values are shown in Figure 1.

Fig. 1. Preponderance image of Lena

The proposed algorithm is a four step process and is illustrated in Figure 2. First, an averaging filter is applied to the Preponderance Block (PB) image, the 2-D array containing the preponderance numbers. This step is helpful in reducing false alarms in edge detection by removing some of the less significant edges. We have found a filter mask of 3×3 tends to give us an acceptable amount of false alarm reduction without removing too much of the useful information.

Following the averaging filter, we determine the gradient of the PB image. To do this, we filter the PB image once horizontally and once vertically using a Sobel edge-emphasizing filter [5]. Each filtering will result in a component gradient in the filtered direction, I_x in the x direction and I_y in the y direction. Equation (1) gives the magnitude of the gradient by calculating the Euclidean distance of the sum of the two component gradients. The resulting gradient is high around edges and tends to be low elsewhere.

$$I = \sqrt{I_x^2 + I_y^2} \tag{1}$$

Lastly, we use k-means clustering to divide the gradient magnitude into a binary map of edges and non-edges.

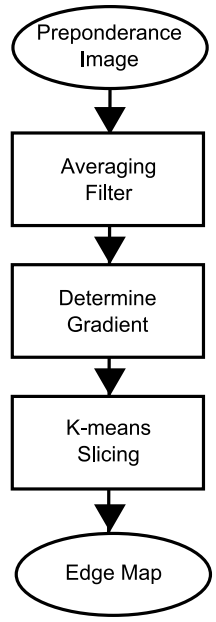


Fig. 2. Block diagram of the proposed algorithm

3. COMPUTATIONAL COMPLEXITY

By performing the segmentation process directly on the bitstream, we are able to reduce processing requirements for our algorithm. By not requiring the decoding of the JPEG image, we eliminate the need to decode the Huffman run-length code and compute the inverse Discrete Cosine Transform (DCT). Determining the complexity for decoding the Huffman code is a challenging problem since the number of table lookups can vary depending on the quantization and image type. Here, we have computed the average number of table lookups required to decode each image in the Berkeley database and determined that each 8×8 block requires about 11 lookups. The 2-D DCT can be implemented using the 2-D FFT and as a result an $N \times N$ image with $D \times D$ sized 2-D DCT blocks, where $N \geq D$ has a complexity of $O((D \log_2 D)^2 * (\frac{N}{D})^2)$ multiplications. For an 512×512 JPEG encoded image, we have an initial savings of about 45,000 lookups and about 1.13×10^6 multiplications that is achieved just by avoiding the decoding process.

Since we are only processing the number of bits used to encode each 8×8 block, we reduce the complexity of each step in our segmentation process by 8 in each dimension. For filtering, this allows us to reduce our complexity by a factor of 64:1.

4. RESULTS

Here, we present quantitative edge detection results for the proposed algorithm using a variety of metrics and techniques. To quantitatively measure performance, we will use the met-

rics discussed next.

4.1. Metrics

In order to compare our results against conventional methods, we will be relying on two objective metrics. The first Method will be Pratts Figure of Merit (FoM) [6]. We will also use the Closest Distance Metric (CDM) described by Prieto and Allen [7].

The first metric we used was the Figure of Merit as described by [6]. By making use of a ground truth image, this metric compares distances of the measured edges to the actual edges in the ground truth images. The formula for FoM is as follows:

$$FOM = \frac{1}{max(P_m, P_a)} \sum_{i=1}^{P_a} \frac{1}{1 + a \times d^2(i)} \tag{2}$$

where P_m is the number of edge pixels in the calculated edge image, P_a is the number of pixels in the actual edge image, a is a constant usually set to $\frac{1}{9}$, and $d(i)$ is the distance between calculated and actual pixels. In this paper we have scaled the FoM to range from [0,100] to more easily compare the FoM to the CDM.

The other metric we used was the Closest Distance Metric as described by [7]. This metric differs from FoM in that it looks for matches between the calculated and actual edge images within a certain distance η. It also takes into account the possibility that no match is found. Below is the formula for CDM:

$$CDM_\eta(m, a) = 100(1 - \frac{\varrho(ClosestMatch)}{|m \cup a|} \tag{3}$$

$$\varrho(CM) = 1 - \varsigma((i,j),(k,l)) \times (1 - |m(i,j) - a(k,l)|) \tag{4}$$

$$\varsigma((i,j),(k,l)) = E(max(|k-i|, |l-j|)) \tag{5}$$

where $\varsigma((i,j),(k,l))$ is the normalized checkerboard distance between the closest matching pixel in the calculated and ground truth images. The values $m(i,j)$ and $a(k,l)$ are the pixel values (0 or 1) of the calculated and actual edge images respectively, and $m \cup a$ is the total number of edge pixels in both images.

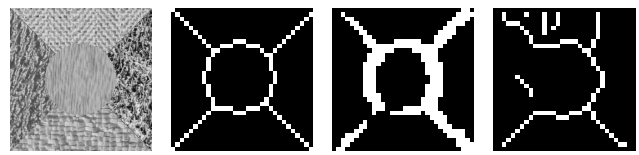
For initial performance evaluation we will compare the performance of the proposed algorithm and the well-known Sobel, Canny, and Laplacian of Gaussian methods. The proposed algorithm will be implemented using a 3×3 averaging filter. In this first experiment we will apply all techniques directly on the preponderance information to develop a baseline for performance. The respective edge detection schemes are applied to a series to 50 Brodatz texture mosaics [8] shown in Figure 3. The purpose of this experiment is to measure how each individual algorithm will perform given the same information, a lossy interpretation (preponderance information) of images with diagonal edges.

Table 1. Average performance on Brodatz textures

	FoM	CDM
Sobel	43.50	79.23
Canny	73.77	82.25
Laplacian of Gaussian	75.77	82.93
Bitstream Based	**83.02**	**85.55**

Metrics range [0, 100], higher is better.

Bold denotes highest value for a given metric.



(a) (b) (c) (d)

Fig. 3. Example results from Experiment 1: a) Original Image b) Ground Truth c) Proposed Method d) Canny Method.

Figure 3 shows a sample of the results from the first experiment. By inspection, the Proposed method outperforms the Canny method. Table 1 shows the quantitative results from the experiment. Using both the Figure of Merit and the Closest Distance Metrics, the proposed bitstream based algorithm outperforms the most commonly used edge detection filters. On the average, our technique can improve upon the computationally complex Canny edge detector by over 12% using FoM and over 4% using CDM. While these metrics were not designed for use in the preponderance domain, this experiment is useful in demonstrating what is possible to do given such a heavily reduced data set.

The second experiment follows the procedure of the first, but instead uses natural images. This experiment uses the four images in Figure 4 and the ground truth provided by [9]. All four images test different aspects of the edge detection problem, and all four edge detection schemes are applied directly on the preponderance domain information to determine how each performs given the same amount of information.

The results in Table 2 demonstrate the performance of the four metrics when applied directly on the preponderance information. In all cases, the proposed bitstream edge detection algorithm performs better than the other three methods. At first glance, it appears that the three commonly used methods, Sobel, Canny, and Laplacian, perform rather poorly. However, it is important to note that those three edge detection schemes were not designed with preponderance numbers in mind, unlike the proposed bitstream processing technique. To better understand how our edge detection system performs

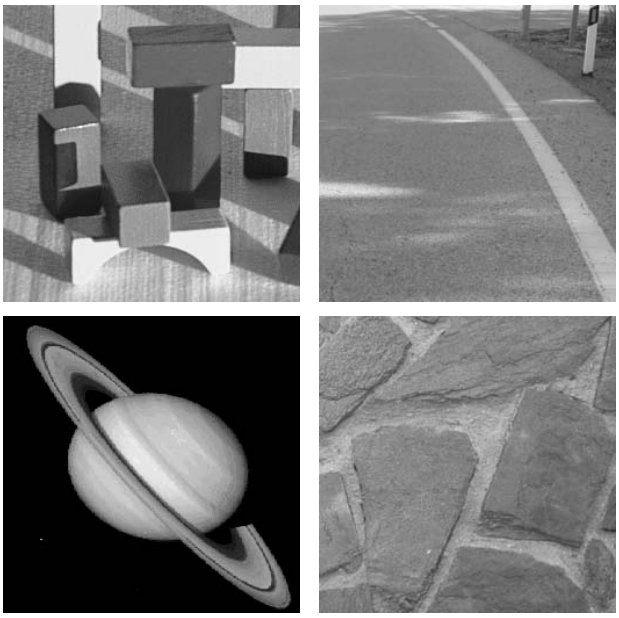


Fig. 4. Test images for Experiments 2 and 3 from [9].

when compared to the other edge detection schemes when applied on the image directly we will have to conduct a third experiment.

This last experiment demonstrates the performance of the proposed algorithm versus well-known edge detectors on natural images. In this case, performance of the Sobel, Canny, and Laplacian of Gaussian methods will be demonstrated on the image domain while the proposed algorithm will still function on the preponderance information. Since the size of the edge map from the bitstream based method is reduced in scale by $1/8$ and thus, has a lower resolution edge map, a scaled down version of the ground truth is used for calculating the results. To scale down the ground truth the image is resized using cubic interpolation and followed by thresholding. This may result in a small advantage for the preponderance domain processing since the preponderance domain tends to remove noise and small details. However, the results from the previous experiments show that the proposed bitstream based method beats the other algorithms when processing directly on the bitstream. It is important to note that bit-domain processing produces lower resolution edge maps; however, for our application, smaller data sets are desired for reduced computational complexity. It is possible to correct the problem of lower resolution edge maps by following the proposed algorithm with an interpolator to scale up the edge maps.

The results shown in Table 3 demonstrate very good performance by the proposed algorithm even when compared to other non-bitstream based approaches operating in the pixel domain. In most cases, the bitstream based method was superior to the pixel-domain methods while having much lower

Table 2. Performance in the preponderance domain.

		FoM	CDM
Blocks	Sobel	7.53	63.78
	Canny	39.81	86.49
	Laplacian of Gaussian	30.54	85.85
	Bitstream Based	**66.52**	**88.31**
Road	Sobel	12.02	68.95
	Canny	42.29	83.17
	Laplacian of Gaussian	36.19	83.69
	Bitstream Based	**71.88**	**84.82**
Saturn	Sobel	24.27	79.88
	Canny	43.18	88.58
	Laplacian of Gaussian	40.87	86.13
	Bitstream Based	**91.30**	**91.98**
Wall	Sobel	9.49	67.61
	Canny	40.86	83.78
	Laplacian of Gaussian	32.55	86.12
	Bitstream Based	**87.65**	**88.61**

Metrics range [0, 100], higher is better.

Bold denotes highest value for a given metric.

Table 3. Performance in the respective domains.

		FoM	CDM
Blocks	Sobel	47.40	80.75
	Canny	60.61	78.86
	Laplacian of Gaussian	**68.37**	79.87
	Bitstream Based	66.52	**88.31**
Road	Sobel	50.23	80.12
	Canny	41.82	70.77
	Laplacian of Gaussian	57.79	77.05
	Bitstream Based	**71.88**	**84.82**
Saturn	Sobel	72.40	91.91
	Canny	57.37	79.49
	Laplacian of Gaussian	70.53	85.94
	Bitstream Based	**91.30**	**91.98**
Wall	Sobel	63.80	77.47
	Canny	30.90	63.87
	Laplacian of Gaussian	37.00	65.86
	Bitstream Based	**87.65**	**88.61**

Metrics range [0, 100], higher is better.

Bold denotes highest value for a given metric.

complexity.

5. CONCLUSION AND FUTURE WORK

In this paper, we have introduced a novel approach to edge detection using bitstream based processing. We have shown that by doing processing directly on the bitstream we are able to reduce computational complexity for use in applications such as distributed sensor networks and JPEG database searching. We have compared the proposed segmentation techniques to other commonly used techniques and the bitstream-based procedure is usually able to achieve performance that is as good as or better than that obtained from other techniques. Its competitive performance, combined with its reduced complexity certainly recommend it as an alternative method for complexity sensitive applications. By operating on a reduced data set and avoiding the decoding process, the complexity savings has been shown to be significant. This type of edge detection technique lends itself to multimedia search applications which seek to extract information from large databases of JPEG-compressed images.

We have only considered here JPEG compressed representations of images. However, the use of other lossy compression algorithms which produce relatively informative compressed representations can also be explored. Since the bit allocations give us a rough approximation of an image, we can also investigate the performance of other previously developed edge detection algorithms on the bitstream.

6. REFERENCES

[1] Djemel Ziou and Salvatore Tabbone, "Edge detection techniques - an overview," Tech. Rep., International Journal of Pattern Recognition and Image Analysis, 1997.

[2] C.D. Creusere and I. Mecimore, "Bitstream-based correlation detector for multi-view distributed video coding applications," in *Proc. IEEE International Conference on Acoustics, Speech and Signal Processing ICASSP 2008*, 2008, pp. 1001–1004.

[3] Charles D. Creusere and Ivan Mecimore, "Bitstream-based overlap analysis for multi-view distributed video coding," in *Proc. IEEE Southwest Symposium on Image Analysis and Interpretation SSIAI 2008*, 2008, pp. 93–96.

[4] G.K. Wallace, "The jpeg still picture compression standard," *Consumer Electronics, IEEE Transactions on*, vol. 38, no. 1, pp. xviii–xxxiv, Feb 1992.

[5] Rafael C. Gonzalez and Richard E. Woods, *Digital Image Processing*, Addison-Wesley Longman Publishing Co., Inc., Boston, MA, USA, 2001.

[6] William K Pratt, *Digital image processing*, Wiley, New York :, 1991.

[7] Miguel Segui Prieto and Alastair R. Allen, "A similarity metric for edge images," *IEEE Transactions on Pattern Analysis and Machine Intelligence*, vol. 25, no. 10, pp. 1265–1273, 2003.

[8] P. Brodatz, *Textures: A Photographic Album for Artists and Designers*, Dover Publications, 1966.

[9] N. L. Fernández-García, A. Carmona-Poyato, R. Medina-Carnicer, and F. J. Madrid-Cuevas, "Automatic generation of consensus ground truth for the comparison of edge detection techniques," *Image Vision Comput.*, vol. 26, no. 4, pp. 496–511, 2008.