FAST FACET EDGE DETECTION IN IMAGE SEQUENCES USING VECTOR QUANTIZATION

M.Y. Jaisimha, Eve A. Riskin, and Robert M. Haralick
Department of Electrical Engineering, FT-10
University of Washington
Seattle, WA 98195. *

ABSTRACT

This paper presents an application that uses vector quantization (VQ) to speed up the process of gradient magnitude edge detection for image sequences. Because image VQ and this type of edge detection operate on block-based neighborhoods, it is possible to use VQ to perform the edge detection. The image is encoded with a VQ for which the edge/no edge decision has already been made for each block. The process of edge detection becomes a simple lookup of this information. The algorithm behaves as a "trainable edge detector" which has the advantage of having lower computational complexity than the facet edge detector. For a VQ with an average rate of 6 bits per vector, our method requires 55% of the multiplications and and 62% of the additions of the conventional facet edge detector. It also enhances the quality of the output by rejecting low contrast, high frequency texture edges.

1. INTRODUCTION

Vector quantization algorithms have been used with much success for image compression [3]. VQ is the extension of scalar quantization to a higher dimensional space and is able to exploit memory or correlation between neighboring samples of an image. More recently, VQ has been used to achieve computational speed-up in signal and image processing [4, 5]. The VQ codebook can be designed specifically for image processing operations such as inverse transforming, histogram equalization, or halftoning. This is done by performing the processing operation on each codeword at VQ design time and storing the results with the codeword. The input image is encoded with this codebook and the processed image is output. It is possible to achieve computational speed-up if the complexity of the encoding process is less than that of the image processing operation by itself. In this paper, we apply this concept to the process of gradient magnitude edge detection.

*This work was supported in part by NSF Grant No. MIP-9110508

The following section describes the facet model and its use for edge detection. The use of VQ to perform edge detection is discussed in Section 3. A comparison of the computational complexity of the algorithm described in this paper and conventional facet model edge detection is given in Section 4. We describe the results obtained when our approach is applied to a sample sequence of images in Section 5and conclude in Section 6.

2. FACET MODEL EDGE DETECTION

The gradient magnitude edge detector that we study is the Second Directional Derivative edge detector described in [6, 7]. The edge detector uses the facet model to estimate the gradient magnitude at each pixel location in the image. The facet model models a digital image as being derived by sampling a continuous underlying graytone intensity surface. This surface can be represented by a low-degree (usually quadratic or cubic) bivariate (in row and column coordinates r and c respectively) polynomial. In the cubic case, the polynomial or "facet" is:

$$f(r,c) = k_1 + k_2 r + k_3 c + k_4 r^2 + k_5 rc + k_6 c^2 + k_7 r^3$$

$$+ k_8 r^2 c + k_9 r c^2 + k_{10} c^3.$$

The first step in the edge detection sequence is the computation of the polynomial coefficients $\{k_i\}$ by performing a local surface fit over a neighborhood centered around each pixel in the image. We choose $P_n(r,c)$, the set of Chebyshev polynomials, as basis polynomials and then compute the coefficients $\{a_i\}$ to minimize:

$$e^2 = \sum_{r \in H} \sum_{c \in W} [I(r,c) - \sum_{n=0}^{K} a_n P_n(r,c)]$$

where $I(r,c)$ is the pixel intensity at coordinate (r,c), and H and W are the set of row and column coordinates (respectively) that belong to the neighborhood.

The gradient direction is given in terms of the polynomial coefficients as $\arctan(\frac{k3}{k2})$ and the gradient magnitude is given by $\sqrt{k_2^2 + k_3^2}$. The second directional derivative in the direction of the gradient and the contrast can also be obtained in terms of the facet polynomial coefficients [6]. The center pixel of the block is labeled as an edge pixel if the second directional derivative in the direction of the gradient has a negatively sloped zero crossing within a threshold radius of the center of the pixel and if the edge contrast exceeds a threshold value. That is, an edge is detected at (r, c) if (a) $f'((r, c)) \neq 0$, (b) $f''((r, c)) = 0$, and (c) $f'''((r, c)) < 0$, where f', f'' and f''' denote the first, second, and third directional derivatives of the facet polynomial.

3. USING VQ FOR EDGE DETECTION

Because both VQ and gradient magnitude edge detectors operate on blocks of an image, VQ can be used for edge detection. For each $M \times M$ vector in the codebook, we determine if the center pixel is an edge pixel using the process described in Section 2 and store the edge/no edge decision with the vector. Images are encoded using the VQ and the process of edge detection becomes a simple lookup operation. If the complexity of the encoding process is less than that of the gradient edge detector, then the process of edge detection can be sped up. We use an unbalanced tree-structured vector quantizer (TSVQ) to lower the complexity of the encoding process [1, 2].

The overall process consists of the following:

- VQ Codebook Design: The training set is formed from all overlapping $M \times M$ blocks of the first image of a motion sequence. It is then used to design an unbalanced TSVQ.

- Facet Edge Detection on the VQ Codebook: For each vector in the codebook, the facet parameters are computed. Second directional derivative edge detection is then performed on each of the codewords and the edge/no edge decision for the center pixel is stored with the codeword.

- Fast Facet Edge Detection: Later images in the sequence are encoded using the VQ and the edge/no edge decision is output for each center pixel.

4. COMPUTATIONAL COMPLEXITY

In this section we present an analysis of the computational complexity of the two approaches. We show under what conditions the approach proposed in this paper has lower complexity than conventional facet edge detection.

Let us assume that the codewords have dimension $M \times M$. At each node of the tree, the VQ performs the following hyperplane test:

$$(\mathbf{x} - \mathbf{y}_L)^T(\mathbf{x} - \mathbf{y}_L) \stackrel{?}{\leq} (\mathbf{x} - \mathbf{y}_R)^T(\mathbf{x} - \mathbf{y}_R).$$

The vector $\mathbf{x}$ is an $M \times M$ dimensional row ordered vector of the input block and $\mathbf{y}_L$ and $\mathbf{y}_R$ are $M \times M$ dimensional row vectors of the centroids of the left and right child of the node under consideration. The operation $(.)^T$ denotes the transpose of the vector. This computation can be simplified to yield the expression:

$$(\mathbf{y}_L^T\mathbf{y}_L - \mathbf{y}_R^T\mathbf{y}_R) \stackrel{?}{\leq} 2\mathbf{x}^T(\mathbf{y}_L - \mathbf{y}_R).$$

This requires M^2 multiplications and $M^2 - 1$ additions. A simple algorithm for the hyperplane test requires 1 difference computation followed by a check of the most significant bit of the result. A conservative estimate for this is 2 additions. Let R be the average rate (height of the tree) of the unbalanced TSVQ. Then the computational complexity is RM^2 multiplications and $R(M^2 + 1)$ additions. In the case of $M = 5$ (which we use in the illustrative example of Section 5.), this is $25R$ multiplications and $26R$ additions.

We now analyze the computational complexity of the facet edge detector for the case of a cubic facet (with 10 coefficients) computed over a $M \times M$ neighborhood centered at each pixel. Each of the 10 facet polynomial coefficients can be computed as a convolution with a precomputed $M \times M$ mask. This requires $10M^2$ multiplications and $10(M^2 - 1)$ additions. The gradient magnitude is computed in terms of the facet polynomial coefficients as shown in Section 2. The sine and cosine of the gradient direction are given by $k_2/\sqrt{k_2^2 + k_3^2}$ (2 multiplications, 1 addition, and 1 division) and $k_3/\sqrt{k_2^2 + k_3^2}$ respectively (2 multiplications, 1 addition, and 1 division). The second directional derivative in the direction of the gradient f'', is given by $(k_4 \sin\theta + k_5 \cos\theta)\sin\theta + k_6 \cos^2\theta$ (5 multiplications and 2 additions) where θ is the gradient direction. The third directional derivative in the direction of the gradient f''', is given by $(k_7 \sin\theta + k_8 \cos\theta)\sin^2\theta + (k_9 \sin\theta + k_{10} \cos\theta)\cos^2\theta$ (8 multiplications and 3 additions). The distance to the negatively sloped zero crossing of the second directional derivative in the direction of the gradient (if it exists) is given by $|-f'/3f'''|$ (1 multiplication and 1 division). In addition we also have three tests to determine if (a) the gradient magnitude exceeds the threshold, (b) $f''' < 0$, and (c) the distance to the negatively sloped zero crossing of the second directional derivative in the direction of the gradient is less than the radius threshold (2 additions and 3 most significant bit tests which can be conservatively

estimated by 2 additions). If we assume that square roots and divisions are computationally equivalent to a multiplication (a very conservative assumption!), then the total computational load of the conventional facet edge detector is $10M^2 + 21$ multiplications and $10M^2$ additions. Thus, the approach proposed in this paper would have a lower computational complexity than conventional facet edge detection if the encoding rate R is such that RM^2 multiplications and $R(M^2 + 1)$ additions are less than $10M^2 + 21$ multiplications and $10M^2$ additions. In our case of $M = 5$, this would be $R < 9.6$.

5. RESULTS

Frame 1, the first image of a motion sequence obtained from a camera mounted on a mobile robot in an outdoor environment (Figure 1), was used as the training image. Figure 2 shows Frame 2, the first test image, and Figure 3 shows the results of applying the second directional derivative edge detector directly to the original test image (no encoding is done). The edge image obtained with our technique when the designed encoder has an average rate of 6.0 bits per vector (hereafter referred to as R = 6.0) is shown in Figure 4. Comparing to Figure 3, we can see that all dominant edges in the image are detected at this rate and the algorithm enhances the edge image by rejecting texture edges. Figure 5 shows the edge image for R=10.0. This figure shows that increasing the rate results in the detection of high frequency texture edges; this is because the higher rate coder can reproduce the image more faithfully, even if this is not the desired effect. We obtained the best edge detection results at R=6.0 which only requires an average of 6 hyperplane tests for the encoding (150 multiplications and 156 additions).

Since Frame 1 is taken at a time that is very close to the training frame, we expect that the performance of the algorithm will be better than if the frame were acquired at a much later time (the later image has less correlation with the training image). Figures 6– 9 show the results obtained when the algorithm is applied to the 22nd frame in the sequence. The anticipated deterioration in performance can be seen quite readily, but the quality of the output has far fewer texture edges compared to applying the facet edge detector directly to the uncoded image.

6. CONCLUSIONS

We have used unbalanced TSVQ for gradient magnitude edge detection At R=6.0, the complexity of the TSVQ for each 5 x 5 vector is 156 additions and 150 multiplications versus 271 multiplications and 250 additions for the conventional edge detector. Furthermore, the VQ-based edge detector outperformed the standard edge detector by rejecting high frequency edges.

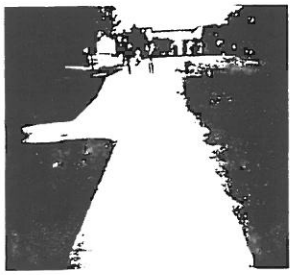

Figure 1: Frame 1 – the Training Image

REFERENCES

[1] A. Buzo, A. H. Gray Jr., R. M. Gray, and J. D. Markel, "Speech Coding Based upon Vector Quantization", *IEEE Transactions on Acoustics Speech and Signal Processing*, 28:562–574, October 1980.

[2] E. A. Riskin and R. M. Gray, "A Greedy Tree Growing Algorithm for the Design of Variable Rate Vector Quantizers", *IEEE Transactions on Signal Processing*, 39(11):2500–2507, November 1991.

[3] R. M. Gray, "Vector quantization", *IEEE ASSP Magazine*, 1:4–29, April 1984.

[4] C. J. Read, D. M. Chabries, R. W. Christiansen, and J. K. Flanagan, "A Method for Computing the DFT of Vector Quantized Data", In *Proceedings of the ICASSP*, pages 1015–1018, IEEE Acoustics Speech and Signal Processing Society, 1989

[5] P. Cosman, E. A. Riskin, and R. M. Gray, "Combining vector quantization and histogram equalization", In *Proceedings Data Compression Conference*, pages 113-118, April 1991.

[6] R. M. Haralick, "Digital step edges from zero crossings of second directional derivatives", *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 6:58–68, January 1984.

[7] R. M. Haralick and L. G. Shapiro, *Computer and Robot Vision*, Addison-Wesley, 1991.

Figure 2: Original Image of Frame 2

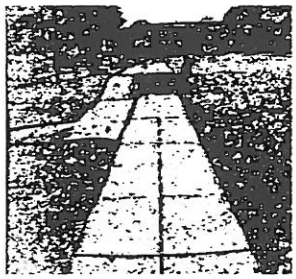

Figure 3: Direct Application of Edge Detector to Frame 2 (No Encoding is Done)

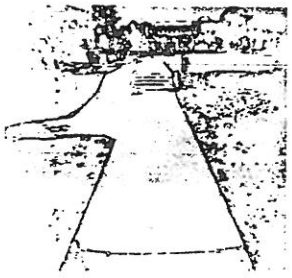

Figure 4: VQ-based Edge Detector at Rate 6 bits per vector for Frame 2

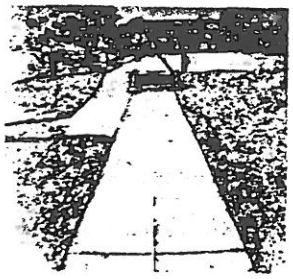

Figure 5: VQ-based Edge Detector at Rate 10 bits per vector for Frame 2

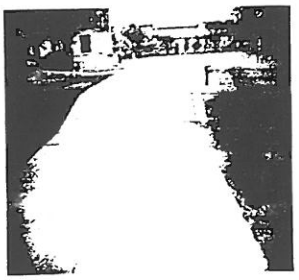

Figure 6: Original Image of Frame 22

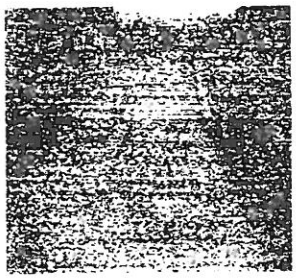

Figure 7: Direct Application of Edge Detector to Frame 22

Figure 8: VQ-based Edge Detector at Rate 6 bits per vector for Frame 22

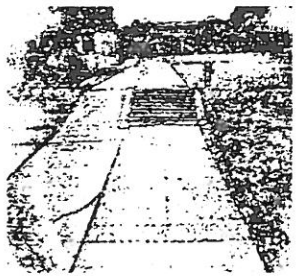

Figure 9: VQ-based Edge Detector at Rate 10 bits per vector for Frame 22