Geometric Constraints in Shape Representation

Thomas C. Henderson, Scott Morris, Charlotte Sanders
and Mohamed Dekhil
Department of Computer Science
University of Utah

June 14, 1997

Abstract

We propose to represent major shape elements in digital images in terms of the topographical features (e.g., ridges and ravines) of the intensity image. We will describe a new ridge/ravine extraction technique motivated by vector field considerations. The shapes are then defined by the geometric constraints that exist between the features. These can be represented as relation matrixes. These require manipulation and are quite large (perhaps 1000x1000), and we are exploring the use of compression techniques applied directly to the relation matrixes in order to reduce the computational complexity of graph matching.

1 Introduction

Image databases have been the focus of much recent work in the computer vision and multimedia fields. Methods have been proposed for image compression, storage and query. Most query methods are based on the use of color, or intensity statistics (usually some form of histogram), as well as various texture parameters, and some limited kinds of shape analysis. In this paper we propose to base shape queries on the topographic features of the image, and in particular on the ridge and ravine features described here. For other approaches, see [3, 4].

We first describe our method for recovering ridges and ravines in a grayscale image, and then explain how we segment the image and compute geometric relations between the segments. These relations are used as the keys by which image querying is achieved. We describe several techniques for compressing the relations matrixes, including the use of image compression transforms and the SVD technique; finally, we describe an approach to graph matching by comparing a subsample of the transform coefficients of the node adjacency graphs.

2 Ridge and Ravine Detection

Ridges and ravines are important features in some image analysis tasks and represent a basic topographic type in digital terrain data. Several methods have been proposed to recover these features, but they have major shortcomings including (1) their sensitivity, and (2) their computational cost (usually as a result of fitting a polynomial). We describe here an approach based on the Laplacian operator that has a firm theoretical foundation and which is relatively inexpensive to compute.

Haralick[2] describes how the facet model approach can be used to recover ridges and ravines. A bicubic polynominal is fit to a patch in the image; ridges are then characterized by a negative second derivative across the ridge line and a zero first derivative in the same direction. The only difference for a ravine is that the second derivative across the ravine is positive. (Haralick's book reviews several earlier techniques for ridge and ravine detection; note that Rosenfeld and Kak maintain that the Laplacian can be used to detect lines.) The computational cost is high due to the ten coefficients that are computed at each pixel.

A more recent technique related to our approach is that proposed by Gauch and Pizer[1]. In their approach, they find places where the "intensity falls off sharply in two opposite directions." They determine curvature extrema of the level curves of the image in order to achieve this. Unfortunately, their calculation requires the evaluation of a large polynominal in the first-, second- and thurd-order partial derivatives of the image, where cubic splines are used to calculate the partial derivatives.

2.1 Curl Method

Our method is based on the following sequence of observations concerning the behavior of the gradient in the neighborhood of a ridge or ravine. Figure 1 shows an image with 2 parallel horizontal ridges. Figure 1 shows an image as a surface in 3D (this is a subimage of a medical image with intensity bands). The gradient produces vectors on the side of a ridge which point toward the ridge and which point away from a ravine. (see Figure 2). Although the gradient can be analyzed directly to determine the location of ridges and ravines, it is computationally more convenient to do the following:

- Rotate (locally) each gradient vector -90 degrees about the out of image axis.
- Calculate the curl at each point to determine the opposed flow that exists at ridge lines.
- Calculate the extremum of this function across the ridge.

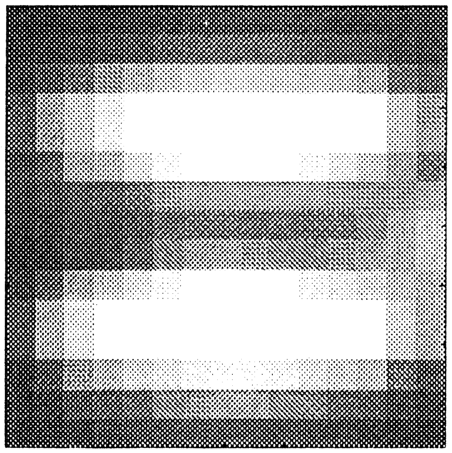

Figure 1: Intensity Image of Two Ridges

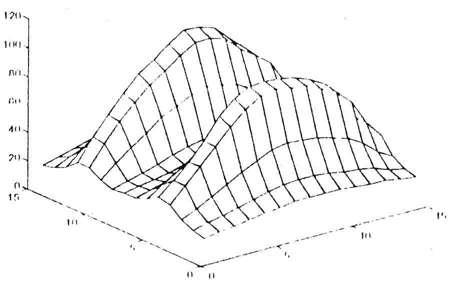

Figure 2: Ridges Viewed as Topography

Figure 3 shows the rotated gradient for the image of Figure 1, while Figure 4 shows the extracted ridge pixels.

Now that the ideas should be clear, we give a formal development of this technique. Let the image function be $f(x,y)$. Then the gradient is:

$$\bigtriangledown f = f_x(x,y) \cdot i + f_y(x,y) \cdot j + 0 \cdot k$$

The rotation is:

$$rot(\bigtriangledown f) = f_y(x,y) \cdot \bar{i} - f_x(x,y) \cdot \bar{j} + 0 \cdot \bar{k}$$

The curl of this is:

$$curl(rot(\bigtriangledown f)) = \begin{vmatrix} \bar{i} & \bar{j} & \bar{k} \\ \frac{\partial}{\partial x} & \frac{\partial}{\partial y} & \frac{\partial}{\partial z} \\ f_y & -f_x & 0 \end{vmatrix}$$

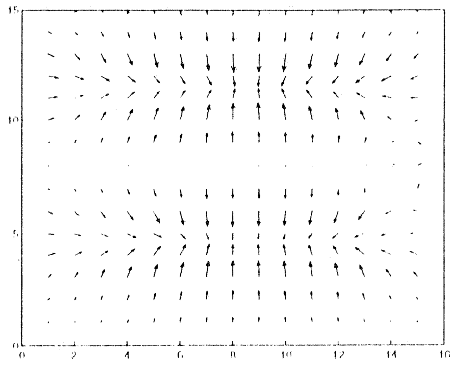

Figure 3: Gradient Vectors of Ridge Image

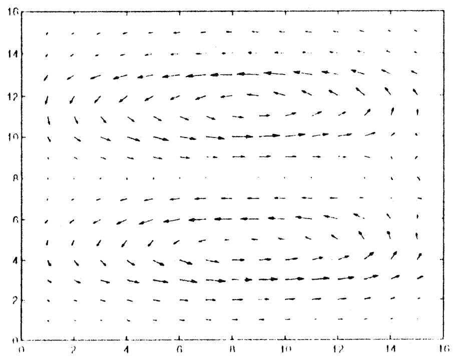

Figure 4: Rotated Gradient

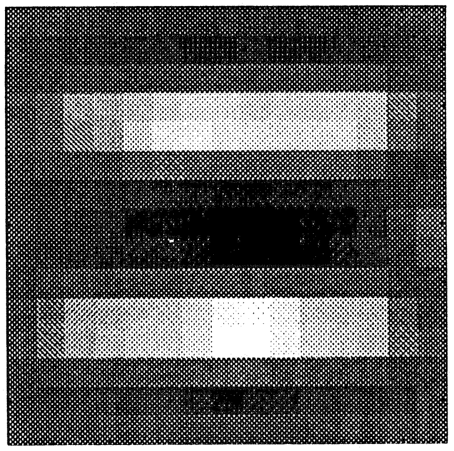

Figure 5: Curl of Rotated Gradient

$$= 0 \cdot i + 0 \cdot j + (-f_{xx} - f_{yy}) \cdot k$$

which is just the negative of the Laplacian.

Finally, a principal direction of curvature for a ridge pixel is:

$$\alpha = \frac{atan2(f_{xy}, f_{yy} - f_{xx})}{2}$$

as well as $\alpha + \frac{\pi}{2}$. We search in these directions to determine that the pixel is a local maximum across the ridge.

3 Encoding and Matching Geometric Relations

3.1 Geometric Relation Graphs

Image segmentation is achieved by thresholding the ridge image. The resulting image is then labeled (connected components are grouped together into segments). Properties are then computed on the resulting segments. For example, consider the 100 by 100 image in Figure 6. The segments are shown in the following figure. Next, the original angle matrix is shown (this has the angle between the principal axes of the segments pairwise). From this matrix various geometric relations matrixes can be formed; for example, Parallel(i,j) by setting the (i,j) entry to 1 wherever the angle between two segments is close enough in value to 90 degrees.

3.2 Graph Matching

If a set of geometric relations graphs are used to represent an image in the image database, then when a query is posed, it is necessary to compare the graphs. We have studied the effectiveness of compressing the geometric relations graphs by means of the SVD, or standard compression transforms. Figure 9 shows our generic approach; instead of matching matrixes (graphs), we transform the graph matrix, and compare the transform coefficients.

If the singular values are computed, then the SVD difference (calculated as the infinity norm on an n-vector for an n by n matrix), correlates very well with the matrix difference (see Figure 10). This graph shows the results for a relations matrix with 80% edge density. If the singular value difference is large, then the graphs are unrelated; however, if it is small, the graphs may still differ significantly. That is, low singular value difference is a necessary but not sufficient condition for graph similarity. If the singular values from several geometric relations are used, then the likelihood of significantly different graphs matching is unlikely.

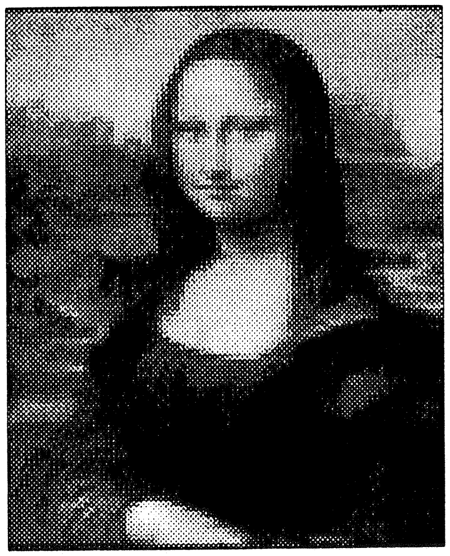

Figure 6: Test Image

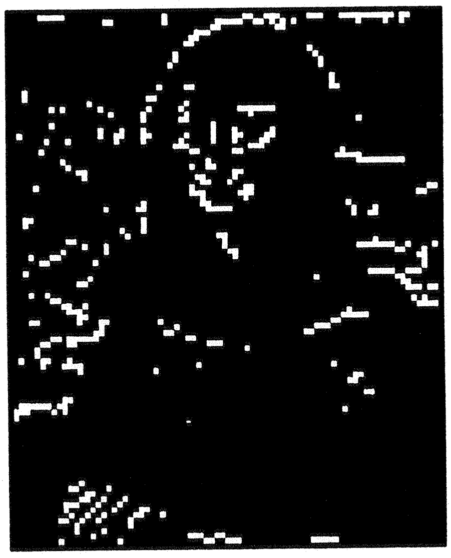

Figure 7: Test Image Ridge Segments

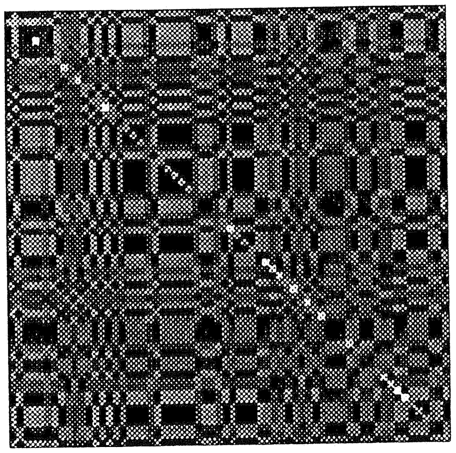

Figure 8: Test Image Segment Angle Matrix

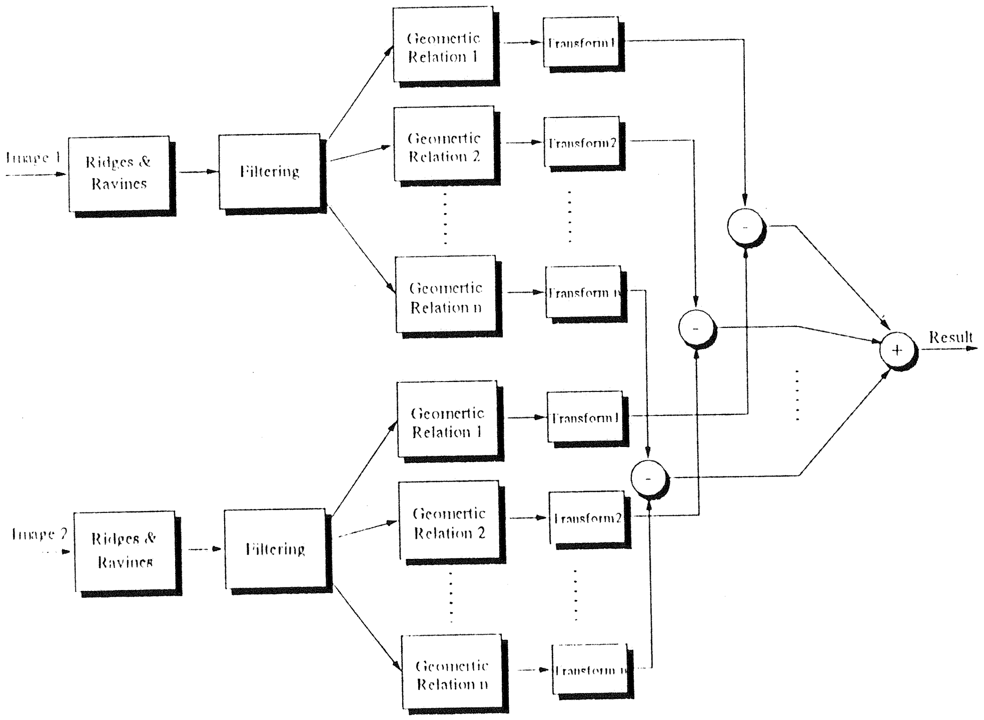

Figure 9: Transform-Based Graph Matching

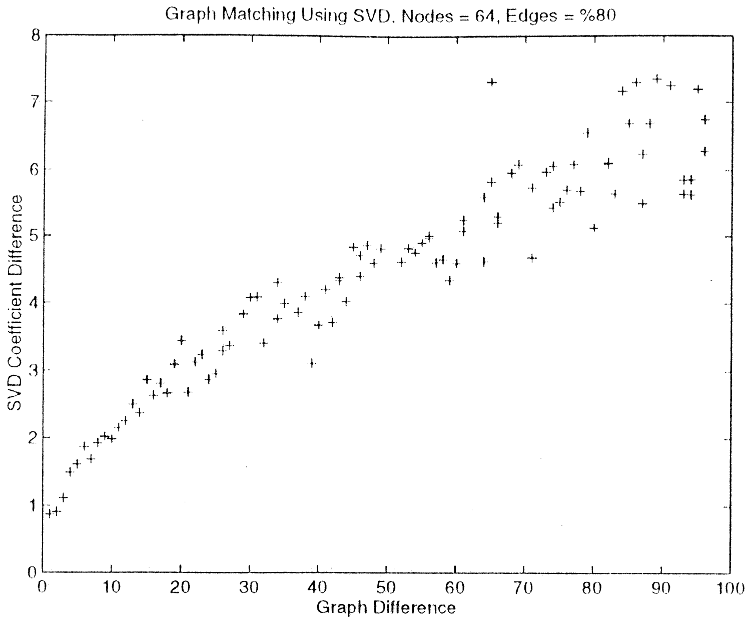

Figure 10: Graph Matching Using SVD

We are also exploring the use of standard image compression techniques (e.g., the Hadamard transform). First, we examined whether the Hadamard transform could efficiently encode a binary relations matrix. Figures 11 and 12 show the graph difference (the sum of the i,j entries which differ) versus the number of Hadamard coefficients used for graphs with 64 nodes at 30% edge density and 256 nodes at 20%, respectively. Our experiments indicate that using about 15% of the coefficients leads to about a 10% error in graph reconstruction. As for the relationship between coefficient difference versus graph difference, Figures 13 and 14 show results for 64 and 128 node graphs, respectively.

4 Summary

We have shown that geometric relations between topographic features of an intensity image can be exploited as a shape representation and query method. Furthermore, the curl of the rotated gradient provides an excellent basis upon which to construct a robust ridge and ravine detector. It is comparable to existing operators, but much less costly. We have tried this technique on a number of types of images and found the results to be very good.

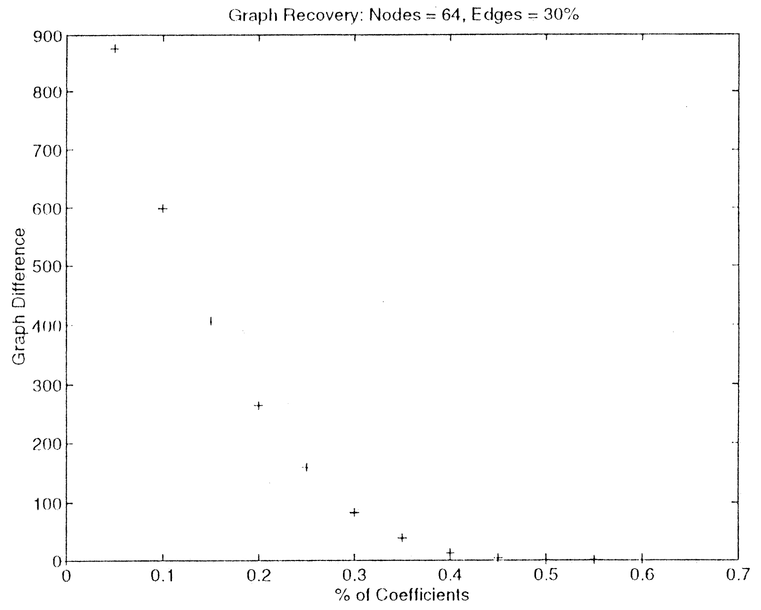

Figure 11: Quality of Inverse Transform Matrix versus Coefficients

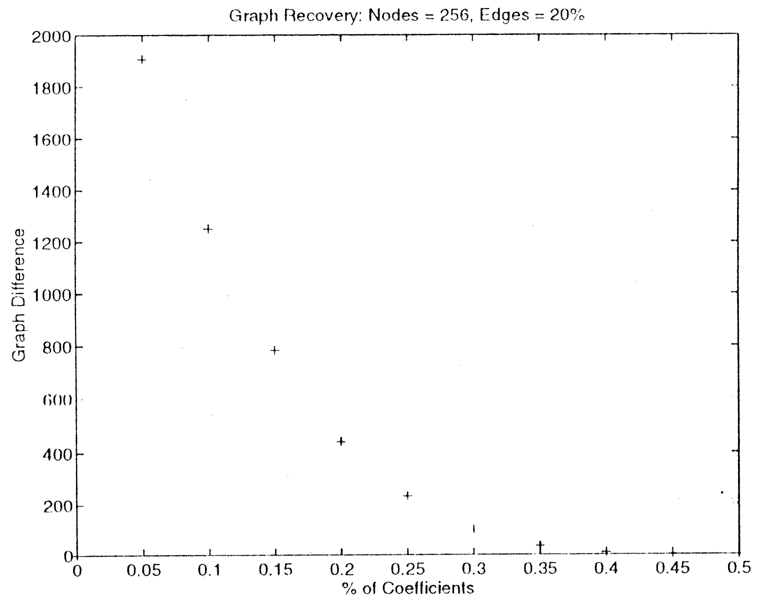

Figure 12: Quality of Inverse Transform Matrix versus Coefficients

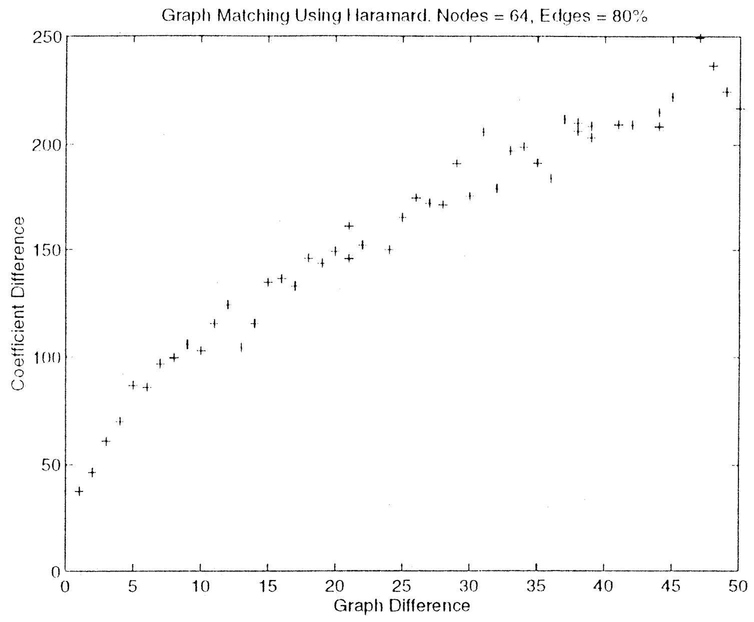

Figure 13: Graph Difference versus Coefficient Difference (64 nodes)

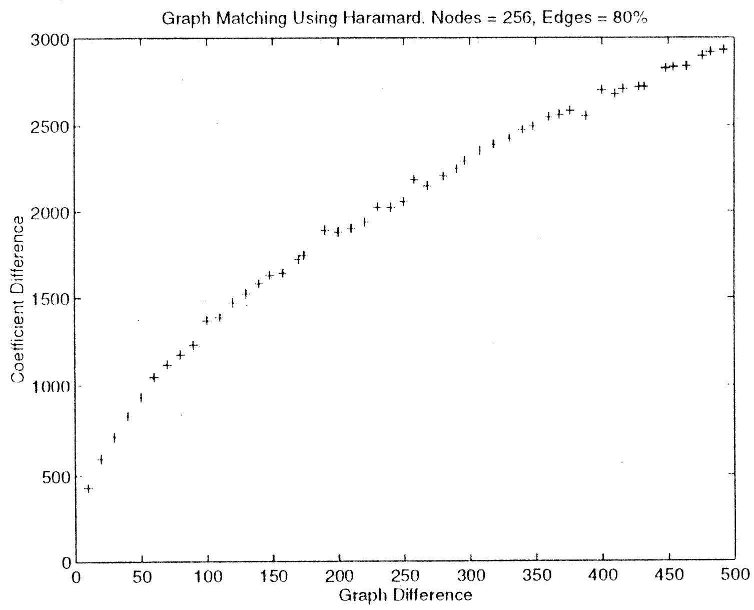

Figure 14: Graph Difference versus Coefficient Difference (256 nodes)

References

[1] John Gauch and Stephen Pizer. Multi-resolution analysis of ridges and valleys in grey-scale images. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 15(6):635–646, June 1993.

[2] Robert Haralick and Linda Shapiro. *Computer and Robot Vision*. Addison-Wesley Pub Co, Reading, MA, 1992.

[3] C. Schmid and Roger Mohr. Local grayvalue invariants for image retrieval. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 19(5):530–535, May 1997.

[4] S. Sclaroff. Deformable prototypes for encoding shape categories in image databases. *Pattern Recognition*, 30(4):627–641, 1997.