SEMANTIC FEATURE ANALYSIS IN RASTER MAPS

by

Trevor Forbes Linton

A thesis submitted to the faculty of
The University of Utah
in partial fulfillment of the requirements for the degree of

Master of Science
in
Computer Science

School of Computing
The University of Utah
August 2009
SUPERVISORY COMMITTEE APPROVAL

of a thesis submitted by

Trevor Forbes Linton

This thesis has been read by each member of the following supervisory committee and by majority vote has been found to be satisfactory.

Chair:  Thomas C. Henderson

Tolga Tasdizen

Ross Whitaker
To the Graduate Council of the University of Utah:

I have read the thesis of Trevor Forbes Linton in its final form and have found that (1) its format, citations, and bibliographic style are consistent and acceptable; (2) its illustrative materials including figures, tables, and charts are in place; and (3) the final manuscript is satisfactory to the Supervisory Committee and is ready for submission to The Graduate School.

Date

Thomas C. Henderson
Chair: Supervisory Committee

Approved for the Major Department

Martin Berzins
Chair/Director

Approved for the Graduate Council

David S. Chapman
Dean of The Graduate School
ABSTRACT

The extraction of semantic features from images of geographic maps is a difficult and interesting problem. Such features may be robustly segmented through the use of Gestalt principles such as similarity and continuity as realized through the use of tensor voting methods and color histogram analysis, respectively.

A framework is developed implementing these Gestalt principles through various algorithms. Linear feature segmentation and intersection detection methods are given, and their performance is demonstrated on a set of real and synthetic map images.
## CONTENTS

<table>
<thead>
<tr>
<th>Section</th>
<th>Page</th>
</tr>
</thead>
<tbody>
<tr>
<td>ABSTRACT</td>
<td>iv</td>
</tr>
<tr>
<td>LIST OF FIGURES</td>
<td>vii</td>
</tr>
<tr>
<td>LIST OF TABLES</td>
<td>ix</td>
</tr>
<tr>
<td>ACKNOWLEDGEMENTS</td>
<td>x</td>
</tr>
<tr>
<td>CHAPTERS</td>
<td></td>
</tr>
<tr>
<td>1. INTRODUCTION</td>
<td>1</td>
</tr>
<tr>
<td>1.1 Research Goals</td>
<td>4</td>
</tr>
<tr>
<td>1.2 Thesis Statement</td>
<td>5</td>
</tr>
<tr>
<td>2. RELATED WORK</td>
<td>6</td>
</tr>
<tr>
<td>3. APPROACH</td>
<td>11</td>
</tr>
<tr>
<td>3.1 USGS Maps</td>
<td>11</td>
</tr>
<tr>
<td>3.2 Gestalt Principles</td>
<td>14</td>
</tr>
<tr>
<td>3.3 Low Level Features</td>
<td>16</td>
</tr>
<tr>
<td>4. PREPROCESSING TECHNIQUES</td>
<td>17</td>
</tr>
<tr>
<td>4.1 Histogram Models</td>
<td>17</td>
</tr>
<tr>
<td>4.2 Class Conditional Density Classifier</td>
<td>21</td>
</tr>
<tr>
<td>4.3 Knowledge Based Classifier</td>
<td>21</td>
</tr>
<tr>
<td>4.4 $k$-Nearest Neighbors</td>
<td>22</td>
</tr>
<tr>
<td>5. TENSOR VOTING</td>
<td>24</td>
</tr>
<tr>
<td>5.1 Dynamic $c$ Values</td>
<td>28</td>
</tr>
<tr>
<td>5.2 Dynamic $\sigma$ Values</td>
<td>28</td>
</tr>
<tr>
<td>6. POSTPROCESSING TECHNIQUES</td>
<td>30</td>
</tr>
<tr>
<td>6.1 Global Thresholding and Thinning</td>
<td>30</td>
</tr>
<tr>
<td>6.2 Local Thresholding and Thinning</td>
<td>32</td>
</tr>
<tr>
<td>6.3 Local Normal Maximum</td>
<td>32</td>
</tr>
<tr>
<td>6.4 Knowledge Based Approach</td>
<td>33</td>
</tr>
<tr>
<td>7. EXPERIMENTS</td>
<td>35</td>
</tr>
<tr>
<td>7.1 Data Selection</td>
<td>36</td>
</tr>
<tr>
<td>7.2 Ground Truth</td>
<td>36</td>
</tr>
<tr>
<td>7.3 Preprocessing</td>
<td>37</td>
</tr>
<tr>
<td>Section</td>
<td>Page</td>
</tr>
<tr>
<td>-------------------------------</td>
<td>------</td>
</tr>
<tr>
<td>7.4 Tensor Voting</td>
<td>39</td>
</tr>
<tr>
<td>7.5 Postprocessing</td>
<td>41</td>
</tr>
<tr>
<td>7.6 Best Combination</td>
<td>41</td>
</tr>
<tr>
<td>7.7 Perfect Data</td>
<td>43</td>
</tr>
<tr>
<td>7.8 No Text</td>
<td>43</td>
</tr>
<tr>
<td>7.9 Comments</td>
<td>45</td>
</tr>
<tr>
<td>8. CONCLUSIONS AND FUTURE WORK</td>
<td>46</td>
</tr>
<tr>
<td>8.1 Future Work</td>
<td>47</td>
</tr>
<tr>
<td>APPENDIX</td>
<td>48</td>
</tr>
<tr>
<td>REFERENCES</td>
<td>49</td>
</tr>
</tbody>
</table>
LIST OF FIGURES

1.1 USGS subimage sample map. ................................................. 2
1.2 Overview of the road framework. ..................................................... 3
3.1 USGS subimage of a city. .......................................................... 12
3.2 USGS subimage of topographical contours. ...................................... 12
3.3 USGS example of a symbol in a map. ........................................... 13
3.4 USGS subimage of a rivers and lakes. ........................................... 13
3.5 USGS subimage of roads and freeways. ....................................... 14
3.6 Demonstration of the law of similarity. .......................................... 15
3.7 Demonstration of the law of proximity. ......................................... 15
3.8 Demonstration of the laws of closure and continuity. ......................... 16
4.1 Initial image (left), initial estimate of features (middle), cleaned up initial estimate (right). ......................................................... 17
4.2 Example of USGS symbols legend. ............................................. 18
4.3 Original image used in comparing the various preprocessing methods below. Original (top left), Forest (top middle), Contours (top right), Geopolitical Symbols (bottom left), Water (bottom middle), Roads (bottom right). .......... 20
4.4 Roads classified by class conditional density classifier with Mahalanobis distance. ........................................................................ 21
4.5 Roads classified by knowledge based system. ................................ 22
4.6 Roads classified by k-nearest neighbors. ......................................... 23
5.1 Synthetic image (left), calculated curve map (middle), and junction map (right). Darker means higher likelihood of feature. ......................... 25
5.2 Visualization of a tensor in two dimensions. The left is a representation of a junction (ball) tensor and the right a representation of a curve (stick) tensor. 25
5.3 2D and 3D view of a voting field’s ideal attenuation as defined by Eq. (5.5). 27
5.4 2D view of a voting field’s attenuation, left $c = 10^7$ vs. right $c = 0.1$. .... 28
6.1 Initial image (left), curve map (middle), junction map (right). ............ 31
6.2 Threshold applied image (left), thinned and cleaned image (right). ........ 31
6.3 Curve map processed by local thresholding and cleaning. Threshold applied image (left), thinned and cleaned image (right). ..................... 32
6.4 Results of looking for the local maxima over the normal. 33
6.5 Results from thresholding, thinning and knowledge based approach using connected component analysis. 34
7.1 Sample ground truth (left) and original image (right) from F34086A1 map. 37
7.2 Class Conditional Density Example, Original Image (left) with roads identified (right). 38
7.3 Effect of varying the sigma parameter of the Tensor Voting system on recall and precision. 39
7.4 Effect of varying the window size of the Local Thresholding method on recall and precision. 42
8.1 Sample of intersecting roads that are too close. 47
8.2 Sample of the output from tensor voting and postprocessing method for intersecting roads that are too close. 47
LIST OF TABLES

4.1 Primary Highways .......................................................... 19
4.2 Secondary Highways ......................................................... 19
4.3 Light Duty Road .............................................................. 19
4.4 Other Street ................................................................. 19

7.1 Comparison of preprocessing methods without tensor voting or postprocessing method. (RR = Road Recall / RP = Road Precision / IR = Intersection Recall / IP = Intersection Precision) ........................................ 38
7.2 Comparison of preprocessing (column) and postprocessing (row) methods with tensor voting. Each cell contains the percentage for (Road Recall / Road Precision / Intersection Recall / Intersection Precision) .......... 40
7.3 Comparison of preprocessing (column) and postprocessing (row) methods without tensor voting. Each cell contains the percentage for (Road Recall / Road Precision / Intersection Recall / Intersection Precision) .......... 40
7.4 Comparison of postprocessing methods with tensor voting and a naive (selecting all black) preprocessing method. (RR = Road Recall / RP = Road Precision / IR = Intersection Recall / IP = Intersection Precision) ....... 42
7.5 Comparison of preprocessing (column) and postprocessing (row) methods with tensor voting. Each cell contains the percentage for (Road Recall / Road Precision / Intersection Recall / Intersection Precision) .......... 43
7.6 Comparison of postprocessing techniques with tensor voting on perfect data. (RR = Road Recall / RP = Road Precision / IR = Intersection Recall / IP = Intersection Precision) ........................................ 44
7.7 Comparison of preprocessing and postprocessing methods run on ground truth images without text. (Road Recall / Road Precision / Int. Recall / Int. Precision) ................................................................. 44
ACKNOWLEDGEMENTS

This thesis was supported by IAVO Research, Inc. through contract FA9550-08-C-0005.
CHAPTER 1

INTRODUCTION

Beginning in 1879 the United States Geological Survey (USGS) began surveying land in the United States. Since then they have developed over 55,000 1:24,000-scale topographic maps covering the 48 conterminous states in a standard, detailed manner. The result is a wealth of data contained in physical documents. Unfortunately many of these documents over the years have begun to deteriorate.

Advances in digital information technology of the 21st century have brought about the conversion of these older physical documents into digitized representations. The knowledge available in these maps is important and therefore it is critical to be able to recover the semantic content (roads, iso-contours, road intersections) of these into meaningful, accurate representations. New methods are developed here in order to extract higher order semantic content.

Work on extracting these features has already started. However parsing the document to extract semantic content has proven difficult. The documents themselves have been scanned and color corrected, with the result that the digitization of older maps has introduced noise and errors in the digital versions. Overlapping features and gaps in the data, as seen in Figure 1.1, cause existing GIS extraction tools to fail. Along with these issues, the amount of descriptive information and variety of symbols have compounded the problem.

To overcome this a framework is developed which implements a robust set of organizing rules used in graphic design and psychology derived from fundamental principles of Gestalt perception. Gestalt principles are derived from the Law of Prägnanz which defines a philosophical method for segmenting objects and an explanation of human perception. The Gestalt principles include the law of closure, similarity, proximity, symmetry, continuity and common fate. Of these laws we attempt to implement similarity, proximity and continuity using various image processing techniques.
The principles of similarity, proximity and continuity are important as they allow the consideration of different perspectives in reconstructing features from noisy data. For a curve segment with gaps, an algorithm implementing the principle of continuity helps correct these errors. In addition, an algorithm which adheres to the principle of similarity such as a histogram model analysis can be very effective at creating rough estimates of features.

Such a method requires the extraction of lower level features such as curves, lines, regions, boundaries, outliers, symbols and junctions as the basis for analysis. The
extraction of complete and accurate representations of lower order features helps to derive higher order semantic features such as interstates, state highways, rivers, etc. This thesis motivates and describes methods for extracting low level features from rasterized images containing geographical map data, and for obtaining robust segmentations of semantic features in the map.

The road framework is broken up into a preprocessing stage, tensor voting and a postprocessing stage shown in Figure 1.2. Using methods such as histogram model analysis, dilation, erosion and thinning has proved adequate at creating an initial estimate of lower level features in the maps. Using tensor voting afterwards we can then fill in and clean the image by reinforcing Gestalt principles.

The resulting output of the tensor voting system is further processed in order to create objects that can be classified in some manner and used in order to extract higher order features. Connected components, thresholding, thinning, and local maxima are used in this postprocessing phase and are compared and contrasted for their effectiveness. The combined efforts of this process allow us to achieve noise reduction, fitting, junction analysis, gap filling and region boundary extraction.

Figure 1.2. Overview of the road framework.
Tools and models developed in the framework are also computationally effective with respect to their memory and CPU time footprint and performance. This is a consideration in order for this research to be usefully exploited in actual data analysis settings.

1.1 Research Goals

The research goals of this thesis arise from the requirements of real life use of extracted content. The primary use of the results from this research is to register the semantic content with other GIS (Geographic Information Systems) software. In order to successfully register our results with other GIS software we must decide on the primary needs of GIS software in order to exploit the results. The primary features considered are roads and road intersections. However this method can be expanded to other linear features with different histogram models.

The detection and localization of intersections provide a clean quantitative framework for the analysis of performance and is highly useful to register maps with aerial images. Using more complicated objects such as curve segments requires more extensive analysis with ground truth data.

Goals are set and performance evaluation for road intersection detection is performed quantitatively through recall and precision as defined in Eq. (1.1) (relevant is the set of relevant pixels in the image, while retrieved is the set of pixels returned by the algorithm). Any two corresponding intersection points are matched if they are within a specified distance (in pixels) of each other in the raster map. Quantitative analysis of linear curve segments is done by comparing the linear curve segments to ground truth images. In situations of gap filling or fitting an optimal curve a manually generated ground truth is used as a comparison for the missing data.

Production use of the framework must also consider runtime performance and memory or CPU-time footprints and constrain ourselves to algorithms that can be run in a realistic

\[
\begin{align*}
\text{Recall} &= \frac{|\text{relevant} \cap \text{retrieved}|}{|\text{relevant}|} \\
\text{Precision} &= \frac{|\text{relevant} \cap \text{retrieved}|}{|\text{retrieved}|}
\end{align*}
\]
amount of time. The tensor voting method specifically can run in a completely parallel fashion and thus constant time (if enough resources are available).

Below are the core goals of this thesis.

- To develop road segmentation algorithms for raster maps that perform at the level of 95% recall and 95% precision for roads in a quantitative analysis.

- To develop road intersection detection algorithms for raster maps that perform at the level of 99% recall and 90% precision for intersections in a quantitative analysis.

Extraction of roads and intersections can be expanded to other features such as iso-contours, rivers, and geopolitical boundaries. While this thesis does not specifically state these as goals, it is well assumed that extraction of them would be similar in its implementation and performance.

1.2 Thesis Statement

Gestalt Principles of continuity, similarity, and proximity can be used to robustly extract low level and semantic features in raster images of USGS maps.
CHAPTER 2

RELATED WORK

Work on extracting semantic content from raster maps has faced many challenges. Maps are scanned under an extremely high resolution TIFF format. A lot of work has been done on color correction and accurate conversion of the scanned data into an indexed color palette; removing color distortions due to scanning was once a topic of research but now has been adequately addressed [10, 16].

With high resolution images and reasonably accurate color in map images, the subject of research is now the extraction of semantic data. Most recent algorithms in the subject of topographical feature extraction typically use a system of thinning to produce vector representations of linear features, followed by A* to connect components along with some sort of preference and simple pattern matching, geometric analysis or histogram analysis to determine features in the raster map image.

The framework described utilizes a method called tensor voting proposed by Medioni [14, 13]. The method is useful for denoising and reconstructing linear features from a sparse data set. The implementation of tensor voting was based on the description in [13].

The method used for implementing tensor voting in the USGS maps is similar to that described by Shao, Ye, Cai and Wang [22]. The method proposed for implementing tensor voting and systems for preprocessing and postprocessing involves thresholding, thinning and denoising the output of the curve map from the tensor voting system.

Chiang, Knoblock and Chen [2] describe a method for extracting roads, filling gaps in roads and identifying intersections. The method extracts pixels thought to be roads using parallel line tracing techniques as well as simply extracting layers of color which correspond to road. The road data are then dilated, eroded and thinned to fill in gaps. The result of this is a 1 pixel wide (unit width) road that was examined for intersections by looking for pixels with three or more adjacent connected pixels. This method for filling gaps may not work for larger gaps as well as situations where gaps curve. The accuracy
due to thinning may not keep the original road boundaries intact. The results from USGS topographic maps for intersections was a precision of 84% with a recall of 75% and a positional accuracy of 80%. A similar method of thinning is used in the preprocessing steps but further processing is needed in order to fill gaps and obtain better performance.

Khotanzad and Zink [11] described a method for contour extraction which first removes any colors not part of the feature set to be examined. Linear features are extracted using valley seeking algorithms. Then the A* algorithm is used to connect valleys together and form linear features or close gaps. While this method has been studied in depth, A* is not as effective at following a curve due to its tendency to prefer proximity over continuity which leads to connections that are not correct. Another concern is the time and space complexities of A*, although these might be overcome with optimization, thinning techniques and the right heuristic function built into the A* algorithm. The results of this method on the sample images in the paper were qualitatively excellent, but quantitatively had incorrectly classified 1.5% of the noncontour lines as contour lines, and 2.4% of contour lines were misclassified as other features. While this method is effective its results on roads rather than contours is still unknown and it is not used to perform intersection detection.

Ahn, Kim and Rhee [1] proposed a method of color separation, noise elimination using erosion and dilation, and thinning and vectorization. This method was used on Korean topographical maps and seemed effective; however no quantitative results were given. The method also seems to be unable to overcome severe noise and gaps in linear features.

The preprocessing techniques described here utilizing histogram models to find feature estimates are based on work published by Henderson, Linton, Potupchik and Ostanin [5]. The models formed are capable of finding good estimates of features that can be later used for tensor voting and postprocessing methods.

Pouderoux and Spinello [19] proposed a method that would reinforce and fill gaps in contour lines using the Gestalt principle of good continuation. The method is based on a gradient orientation field generated by the contour lines and uses the tangent at the ends of the gap to approximate the best curve between the two points without crossing another linear feature. The framework described uses principles similar to our method although we use tensor voting to enforce good continuation rather then a gradient orientation field. An example was given in the paper. However no quantitative results were published.
Pezeshk and Tutwiler [16] proposed a system of histogram equalization (HE, AHE and CLAHE) to color correct USGS images and enhance features in the image. They did not publish quantitative results for this method. However enhancing a feature through blurring, dilation and subsequently a form of histogram equalization may prove useful for enhancing a feature prior to extraction.

Miyoshi, Li, Kaneda, Yamashita and Nakamae [15] published a review of methods for extracting buildings utilizing geometric features. Extraction was achieved by thinning line segments and using a vector-chaining procedure to produce various types of connected, branching and nonconnection vectors. Features of the vectors were used to identify buildings and other features in the image. For example, a connected loop of a certain size would be classified as a building while any nonconnecting loop or loops which were too large or did not contain sides with a straight line were not considered buildings. The results of this method correctly identified 87.3% of village buildings, 83.3% of urban buildings and 89.4% of residential buildings. False positive identifications were 5.2%, 3.3% and 9.7%, respectively.

Using methods of this type may be effective; however, the data input used in these examples were well formed scanned drawings. The USGS data set used in this thesis contains various amounts of noise and gaps in the data, making preprocessing steps useful to extract lower order features.

The analysis of technical and engineering drawings using distributed agent based systems has been shown beneficial. Swaminathan [23] and Henderson [6, 7, 8] examined this technique using various image processing techniques combined with an agent based system and a hierarchy of grammars to examine if communication of networks of agents could better correct for errors and identify both syntactic and semantic symbols in drawings. The use of agents showed a significant ability to account for unknown semantics in engineering drawings.

A similar method of thinning and vectoring were described by San, Yatim, Sheriff and Ismail [21]. Connecting components of contour lines was done using radial symmetry in the contour lines and closest extremity. The initial problem with this approach is the lack of smooth curves between disconnected components.

Zheng, Liu, Shi and Zhu described a method [24] for extracting roads out of satellite images. The method used colors to extract specific road types and make an initial estimate of the feature. The tensor voting framework was then used on the binary image to
produce a curve map. The tensor voting framework curve map was thresholded and then thinned in order to make a unit width vector of the road. The results were qualitatively good. However no quantitative results were published. The method used in this thesis is similar. However the preprocessing and postprocessing techniques are different. The method described by Zheng, Liu, Shi and Zhu was applied to satellite images while the method described here is applied to USGS maps.

Poullis, You and Neumann [20] used a similar method of tensor voting on satellite image data, but combined it with other images such as aerial photographs to more accurately segment the image. No quantitative results were reported.

Hinz and Baumgartner described a method [9] of identifying roads in aerial images by first segmenting the image into urban and rural areas, removing shadows, occlusion and building outlines. The result is then segmented using histogram model analysis, thresholding and then modeling techniques. The result of this method is 75% recall and 95% precision for roads.

Li, Nagy, Samal, Seth and Xu [12] published a method for extracting labels, roads and graphics from USGS maps. The method traces roads until text is found and wraps it in a bound box extracting the text then filling in the gap for the next road. This method was effective at finding well defined roads and text, but failed on curved roads and noisy data.

Podlasov, Ageenko and Franti [18] published a method of restoration of binary semantic layers of map images. In the paper they proposed a method for restoring raster image maps from artifacts caused by color separation. However their method was typically used on maps which are in a RGB or CMYK color space we work on images with color indexed palettes that are well defined.

Ageenko and Podlasov [17] proposed a novel method for removing noise and reducing error in features. The method to extract a feature begins by taking a union of overlapping features and then dilating the feature of interest and using the unioned features as a mask to reduce the affects of over-dilation. The results of the method were taken over NLS Map Series for the water and field features. The results reduced the error rate in the features by 12.72% to 14.14% for water and fields, respectively.

Chaing, Knoblock, Shahabi and Chen [3] published methods for extracting roads and intersections using a wide variety of image processing techniques from thinning, erosion, dilation and histogram analysis. The use of double-line format detection or
parallel-pattern tracing was used to identify roads via a geometric analysis of the structure in the map. A knowledge based logical process was followed to extract the roads and intersections once the preprocessing steps were finished. The method described has a precision of 82% and a recall of 60% for USGS Topographic maps.

Previous work with tensor voting and postprocessing methods were described by Henderson and Linton [4]. The methods introduced in this paper are the basis for the thesis and work described here. The preliminary results published in this paper are 93% recall and 66% precision for intersections.
CHAPTER 3

APPROACH

To extract features from USGS maps, a system is built utilizing Gestalt principles of similarity, continuity and proximity. The system is separated into three parts: a preprocessing phase, tensor voting and a postprocessing phase. The preprocessing phase cleans up initial noise, finds rough estimates of features utilizing histogram models and reduces features to smaller unit width through thinning. The tensor voting phase reduces noise, bridges gaps and reinforces curves in a manner which preserves continuation over proximity. The postprocessing phase takes the resulting output from the tensor voting system, thresholds the data and then uses a variety of techniques discussed here to accurately extract unit width vector features.

The result of this process is a binary unit width segmentation of the specific linear feature of interest. Processing on these segments then allows us to extract further details such as intersections (junctions) within the linear features.

3.1 USGS Maps

United States Geographical Survey maps are large (average of 9495 by 5552 pixels and 15 megabytes) TIFF 8-bit color indexed raster images. Care is taken to make sure the maps are read in and not converted to any other color-space or format. Casting the raster map into a different color-space could result in inaccurate colors due to interpolation. Converting the map into a lossy image format may result in compression artifacts and inaccurate colors and geometrical features.

The maps contain a range of features including (but not limited to) contour lines, geopolitical boundaries, symbols, roads, train tracks, labels, lakes, rivers, highways and freeways. While most maps only contain a small number of colors, features are identified through mixed color combinations giving a distinct texture to a feature. Human defined symbols like text and road markers can be found by exploiting their distinct color and shape.
Cities are generally identified with smaller clusters of roads and a text label identifying the name of the region such as in Figure 3.1. Contour lines are generally identified with a brown color and are condensed curves (Figure 3.2) describing topographical information and height of the area. Symbols in maps such as in Figure 3.3 identify contextual information for roads, cities and other geopolitical information. Large areas with a dotted blue and white pattern identify water such as in Figure 3.4. We limit ourselves to the

![Figure 3.1. USGS subimage of a city.](image1)

![Figure 3.2. USGS subimage of topographical contours.](image2)
extraction of roads, usually red and black depending on the type of roads, which are
generally straight intersecting lines shown in Figures 3.1 and 3.5.

Roads can be defined by a variety of textured colors. Some maps include a pink or
black double lined roads with specific markers for direction. Others include thicker black
and red lines to describe freeways, thinner to describe highways and solid thin black
lines to describe regular roads, urban or rural. Train tracks and other railways may be
described by a black line with stylized cross ties.

Features may overlap in the map making gap filling a necessity in order to accurately
extract and represent roads, contour lines, etc. Gaps may be exceedingly large which
represents a challenge in accurately estimating a curve or line correctly. A larger problem

\begin{figure}
\centering
\includegraphics[width=0.4\textwidth]{1325.png}
\caption{USGS example of a symbol in a map.}
\end{figure}

\begin{figure}
\centering
\includegraphics[width=0.4\textwidth]{Graveyard Point.png}
\caption{USGS subimage of a rivers and lakes.}
\end{figure}
arises when two distinct features having similar textures overlap. This can make correctly segmenting features increasingly difficult.

Text is largely ignored in this analysis as document analysis has defined methods for extracting text within maps. The process of extracting text is made simpler as the need to accurately represent the original shape of the text (as is necessary with other features such as roads) is not required.

3.2 Gestalt Principles

Gestalt psychology or gestaltism is a theory of the brains self-organizing capability with respect to visual recognition of figures and shapes. Within Gestalt psychology the Law of Prägnanz states that we experience visual phenomena as regular, orderly, symmetric, and simple. In an attempt to refine the Law of Prägnanz, Gestalt psychologists have created the Gestalt principles which help organize visual information. These principles include the law of closure, similarity, proximity, symmetry, continuity and common fate. In an attempt to achieve image segmentation, some of these laws or principles are used in our framework in order to draw out features.

- Similarity: The grouping of similar elements into a whole element or grouping based on shape and texture to complete a regular figure or region. As can be seen in Figure 3.6 the dark circles and outlined circles produce separate segmentations in the image.
• **Proximity**: The amount of space between objects and pixels is a tool for grouping items together and determining boundaries, regions and features. As can be seen in Figure 3.7 the spacing between the lines of circles on the left cause a segmentation.

• **Continuity**: Patterns which repeat are followed by the mind. Even in the event the pattern is missing in places, the mind will still fill in gaps and conjure up the complete pattern in the mind. The principle of continuity is demonstrated in Figure 3.8.

These principles are represented in the methods used here. The law of similarity is exploited through the use of histogram models. The histogram models match various textures and group them together to get rough estimates of features. The laws of proximity and continuity are represented through the tensor voting system. The tensor voting system enforces good continuity and proximity when filling gaps and optimally
3.3 Low Level Features

In order to extract semantic features low level features must first be analyzed. Low level features include lines, curves, color histograms, junctions, regions and end points. The four main low level features to be exploited are:

- Linear nature of the pixel (lines, curves).
- Color histogram of local window at each point.
- Pixel regions.
- End points.

There are a few constraints on the output of lower order features.

1. Each feature must be independent from others. For example two crossing roads should be separated into the two road segments and an intersection as low level features.
2. Segments or objects must be unit width and connected.
3. Gaps must be filled with the closest approximated curve and in a manner which preserves continuity, not proximity.
4. Features must be semantically smooth and preserve the original feature in the raster image.
CHAPTER 4

PREPROCESSING TECHNIQUES

To properly prepare the raster maps for analysis by the tensor voting framework, a series of preprocessing steps is necessary to remove noise and approximate feature classes. The process begins by assigning initial feature labels using a variety of techniques. This segmentation produces a binary image. The results are then cleaned by removing outliers and isolated elements. The data are run through a skeletonization process as seen in Figure 4.1. Once properly cleaned the results are then used in the tensor voting framework.

4.1 Histogram Models

USGS maps have a well-defined structure which we can exploit. Using the Gestalt principles of similarity, initial assignments can be made on the various types of roads in the map by examining histogram models representing a class. In order to make initial assignments of pixels, the specified features histogram model is created as a set of sample histograms representative of the class. The samples are taken from areas in the maps where each feature is shown in Figure 4.2. Each class has the same number of histogram samples in order to prevent bias to any one class. Tables 4.1 - 4.4 are common road types and their corresponding histogram models.

Figure 4.1. Initial image (left), initial estimate of features (middle), cleaned up initial estimate (right).
Figure 4.2. Example of USGS symbols legend.
### Table 4.1. Primary Highways

<table>
<thead>
<tr>
<th>Indexed Pixel Value</th>
<th>Color</th>
<th>Number in Subimage</th>
</tr>
</thead>
<tbody>
<tr>
<td>0</td>
<td>Black</td>
<td>1030</td>
</tr>
<tr>
<td>1</td>
<td>White</td>
<td>2925</td>
</tr>
<tr>
<td>2</td>
<td>Blue</td>
<td>7</td>
</tr>
<tr>
<td>3</td>
<td>Red</td>
<td>456</td>
</tr>
<tr>
<td>4</td>
<td>Brown</td>
<td>352</td>
</tr>
<tr>
<td>5-12</td>
<td>Green</td>
<td>0</td>
</tr>
<tr>
<td>13+</td>
<td>Unknown</td>
<td>0</td>
</tr>
</tbody>
</table>

### Table 4.2. Secondary Highways

<table>
<thead>
<tr>
<th>Indexed Pixel Value</th>
<th>Color</th>
<th>Number in Subimage</th>
</tr>
</thead>
<tbody>
<tr>
<td>0</td>
<td>Black</td>
<td>533</td>
</tr>
<tr>
<td>1</td>
<td>White</td>
<td>3300</td>
</tr>
<tr>
<td>2</td>
<td>Blue</td>
<td>1</td>
</tr>
<tr>
<td>3</td>
<td>Red</td>
<td>366</td>
</tr>
<tr>
<td>4</td>
<td>Brown</td>
<td>305</td>
</tr>
<tr>
<td>5-12</td>
<td>Green</td>
<td>0</td>
</tr>
<tr>
<td>13+</td>
<td>Unknown</td>
<td>0</td>
</tr>
</tbody>
</table>

### Table 4.3. Light Duty Road

<table>
<thead>
<tr>
<th>Indexed Pixel Value</th>
<th>Color</th>
<th>Number in Subimage</th>
</tr>
</thead>
<tbody>
<tr>
<td>0</td>
<td>Black</td>
<td>807</td>
</tr>
<tr>
<td>1</td>
<td>White</td>
<td>2404</td>
</tr>
<tr>
<td>2</td>
<td>Blue</td>
<td>111</td>
</tr>
<tr>
<td>3</td>
<td>Red</td>
<td>0</td>
</tr>
<tr>
<td>4</td>
<td>Brown</td>
<td>12</td>
</tr>
<tr>
<td>5-12</td>
<td>Green</td>
<td>20</td>
</tr>
<tr>
<td>13+</td>
<td>Unknown</td>
<td>0</td>
</tr>
</tbody>
</table>

### Table 4.4. Other Street

<table>
<thead>
<tr>
<th>Indexed Pixel Value</th>
<th>Color</th>
<th>Number in Subimage</th>
</tr>
</thead>
<tbody>
<tr>
<td>0</td>
<td>Black</td>
<td>163</td>
</tr>
<tr>
<td>1</td>
<td>White</td>
<td>1060</td>
</tr>
<tr>
<td>2</td>
<td>Blue</td>
<td>45</td>
</tr>
<tr>
<td>3</td>
<td>Red</td>
<td>0</td>
</tr>
<tr>
<td>4</td>
<td>Brown</td>
<td>8</td>
</tr>
<tr>
<td>5-12</td>
<td>Green</td>
<td>0</td>
</tr>
<tr>
<td>13+</td>
<td>Unknown</td>
<td>0</td>
</tr>
</tbody>
</table>
Most features in the system are linear and can be analyzed with a histogram model as their geometric structure is less important than color texture; this makes histogram models effective. However some features such as route markers have a geometric shape which makes it necessary to employ some other detection system for them.

Several different methods are used to create initial assignments of features. A process of using mean, standard deviation and other statistical methods called the class conditional density with Mahalanobis distance is shown using the histogram models described earlier. A knowledge based classifier uses information known about the various features to identify and classify them. This method seems to work well at producing sharp and clean assignments with minimal dilation or bleeding. The last process considered here is $k$-nearest neighbors method using the histogram example as the model described earlier. The methods described here are compared qualitatively with respect to the map shown in Figure 4.3.

Figure 4.3. Original image used in comparing the various preprocessing methods below. Original (top left), Forest (top middle), Contours (top right), Geopolitical Symbols (bottom left), Water (bottom middle), Roads (bottom right).
4.2 Class Conditional Density Classifier

The class conditional density with Mahalanobis distance classifier is a statistical method to classify each pixel in the image. The class model is the mean of the histogram examples defined by Eq. 4.1 and the variance of each class of histograms defined by Eq. 4.2. The Mahalanobis distance is defined by Eq. 4.3. Eq. 4.3 is calculated for each $k$ and the $k$ with the minimal value is the assigned class. The result of this method on a 500 by 500 pixel region is shown in Figure 4.4.

$$
\mu_k = \frac{1}{N_k} \sum_{i=1}^{N_k} x_{k,i} \tag{4.1}
$$

$$
\Sigma_k = \frac{1}{N_k - 1} \sum_{i=1}^{N_k} (x_{k,i} - \mu_k)(x_{k,i} - \mu_k)^T \tag{4.2}
$$

$$
\delta(x; \mu_k, \Sigma_k) = \frac{1}{2} \sqrt{\left(x - \mu_k\right)^T \Sigma_k^{-1} \left(x - \mu_k\right)} \tag{4.3}
$$

4.3 Knowledge Based Classifier

To classify features in the image a knowledge based approach can be used in which the user specifies a predicate which characterizes class membership. The approach is based on a normalized histogram of an area around the pixel that will be classified. The process uses the normalized histogram bins to determine the class. The results of a road classification are shown in Figure 4.5.

![Figure 4.4](image)

**Figure 4.4.** Roads classified by class conditional density classifier with Mahalanobis distance.
To classify roads the bins of the histogram are analyzed and roads are further subclassified into light duty and primary roads. The light duty roads are identified if the color black is the largest color in the histogram, there is blue in the histogram, there are less or equal amounts of red than blue, and there is less brown than blue. Primary roads are classified if black is the most numerous color in the histogram, the color blue is less than or equal to red, blue is less than or equal to brown, there is some red in the histogram, and there is some brown.

4.4 $k$-Nearest Neighbors

The $k$-Nearest Neighbors process begins by taking a training set of examples (histograms at selected pixels and their respective classes) and treating these as vectors, producing an $n$-dimension vector where $n$ is the number of bins in the histogram. The histogram from the window surrounding the point of interest is then represented in the vector space and the $k$ closest neighbors to it are found. An example of roads classified by $k$-nearest neighbors is shown in Figure 4.6. The distance is calculated by the Euclidean distance defined by Eq. 4.4 where $p$ is the point of interest and $q$ is the point of the histogram model and the index $i$ represents the bin. The assignment is then determined by finding the class out of the $k$ neighbors that is most frequent.

$$D = \sqrt[n]{\sum_{i=1}^{n} (p_i - q_i)^2}$$  \hspace{1cm} (4.4)
Figure 4.6. Roads classified by $k$-nearest neighbors.
CHAPTER 5

TENSOR VOTING

Each pixel classified as road during the preprocessing step gives rise to a curve estimate through that pixel; this curve is then used to cast votes at other pixels where the curve should pass. A nonroad pixel will thus accumulate votes for the various curves that could pass through it, with straight continuations favored over those sharply changing direction. Thus, a closer curve may have less influence than a more distant one for which the pixel provides straight continuation. Linear features that are in close proximity but have little continuation may still be connected if no other ideal curve can be found.

Tensor voting is a powerful image processing tool that helps clean images, fill gaps and reinforce curves based on Gestalt principles of closure and continuity. The tensor voting system takes in a binary input image and outputs a junction and curve map (see example shown in Figure 5.1). The curve map gives the likelihood of the presence of a curve passing through each pixel. Likewise the junction map expresses the likelihood of a junction at a pixel. The curve and junction maps are valued from 0 to 1. The tensor voting system takes two parameters, \( \sigma \) and \( c \). While \( c \) can be derived from \( \sigma \) for optimal balance, it can also be varied to give preference to either continuation or proximity as desired.

Eq. (5.1) defines a tensor and is in the form of a second order, symmetric, non-negative definite tensor. The tensor here is a 2x2 matrix and a mathematical representation of the structure type, direction and saliency. The structure type is either a junction or curve and is measured by the difference between \( \lambda_1 \) and \( \lambda_2 \) as shown in Figure 5.2. If \( \lambda_1 \) is significantly larger than \( \lambda_2 \), the point where the tensor is located is thought to be a part of a curve. If the differences between the two \( \lambda \)'s is small, the tensor is thought of as a junction. (Note Medioni referred to the structure types as stick tensors and ball tensors, respectively.) The direction is the alignment of the tensor in the direction of the preferred tangent (or in other words the preferred direction of the curve at that point, where \( \hat{e}_1 \)
Figure 5.1. Synthetic image (left), calculated curve map (middle), and junction map (right). Darker means higher likelihood of feature.

Figure 5.2. Visualization of a tensor in two dimensions. The left is a representation of a junction (ball) tensor and the right a representation of a curve (stick) tensor.

is the preferred tangent and \( \hat{e}_2 \) is the normal.) The saliency is the confidence that this feature exists and is measured by the magnitude of \( \lambda_1 \).

\[
T = \lambda_1 \hat{e}_1 \hat{e}_1^T + \lambda_2 \hat{e}_2 \hat{e}_2^T \tag{5.1}
\]

The tensor voting process begins by building a sparse tensor field. The sparse tensor field is an \( P \times N \) array (the same size as the input) with an initial estimate of each tensor at each point defined in the input. The estimates are built using principal component analysis using the direction or estimated tangent as \( \hat{e}_1 \) and its normal as \( \hat{e}_2 \). The values of \( \lambda_1 \) and \( \lambda_2 \) are set to one for the initial estimate. Estimates can also be built using a variety of other techniques such as ball voting defined by Medioni or a Canny edge detection algorithm.

Once a sparse tensor field is generated, a voting field is created for each point defined in the sparse tensor field. The voting field is an \( M \times M \) field where the size (M) is defined by solving for \( W_{\text{size}} \) in Eq. (5.2). The voting field contains a tensor at each point in the field. The direction of the tensors in the voting field are defined by Eq. (5.3) where \( \theta \) is the
angle from the location of the tensor in the field to the origin (center) of the field defined
along the x-axis and \( l \) is the distance from the origin to the tensor. The attenuation of
the tensors in the voting field are defined by Eq. (5.4) where \( \theta \) is as defined before, \( s \) is
the arc length, and \( k \) is the curvature. The voting field is aligned with the tangent of the
tensor in the sparse tensor field and positioned to be centered above it. Once the voting
field is generated and aligned with the tensor in the sparse field, the tensors in the voting
field and sparse field are added together to produce a dense tensor field.

\[
W_{\text{size}} = \sqrt{-\sigma^2 \ln(0.01)} \tag{5.2}
\]

\[
s = \frac{\theta l}{\sin(\theta)}
\]

\[
k = \frac{2\sin(\theta)}{l}
\]

\[
S_{\text{SO}}(l, \theta) = DF(s, k, \theta) \begin{bmatrix} -\sin(2\theta) \\ \cos(2\theta) \end{bmatrix} \begin{bmatrix} -\sin(2\theta) & \cos(2\theta) \end{bmatrix} \tag{5.3}
\]

\[
DF(s, k, \theta) = \left( e^{-\frac{s^2 + sk^2}{\sigma^2}} \right) \tag{5.4}
\]

The dense tensor field can then be decomposed in order to determine where curves
and junctions exist. The field also describes the direction of any curves through the \( \hat{e}_1 \)
value. For each tensor in the dense tensor field the curve map and junction map can be
found by considering:

- If \( \lambda_1 - \lambda_2 > 0 \), demonstrates the likelihood of the point being on a curve. The
  saliency of a stick component being larger than that of the ball component indicates
  a certainty of the orientation. Therefore the tensor most likely belongs to a curve.

- If \( \lambda_1 \approx \lambda_2 > 0 \), demonstrates the likelihood of the point being a junction or
  irrelevant. If the \( \lambda_1 \) is approximately the same size as \( \lambda_2 \) no orientation is likely and
  therefore the point can either be considered irrelevant or a junction. The intensity
  of \( \lambda_1 \) and \( \lambda_2 \) shows the certainty that it is junction; higher values indicate a junction
  while low values indicate an irrelevant point.
As noted the tensor voting field depends on two parameters $\sigma$ and $c$. The $\sigma$ parameter defines the size of the attenuation field. The parameter $c$ defines the shape of the attenuation field, smaller values of $c$ result in an attenuation field that is more circular or in other words considers proximity as much as it does continuation when calculating curves. When the value of $c$ is large, the attenuation field is more stick shaped, preferring continuation heavily over proximity when determining curves. The ideal value of $c$ (one which prefers an optimal balance of continuation vs. proximity as defined by Medioni) can also depend directly on $\sigma$ and is determined by Eq. (5.5) and shown in Figure 5.3. Figure 5.4 shows the difference between small and large $c$ values and how it affects the voting field’s attenuation.

\[
c = \frac{-16 \ln (0.1)(\sigma - 1)}{\pi^2}\] (5.5)

Choosing appropriate parameters for the tensor voting framework is of considerable importance for producing good results. The size of the voting field affects how well the tensor voting framework performs. Although relatively insensitive to size parameters, error can be introduced by too large or too small a voting field. Too large a field can ruin subtle details and features. Too small a field does not adequately remove noise or fill gaps in the raster image. To ensure we have a proper field size and continuation vs. proximity, methods must be developed to approximate an ideal $\sigma$ and $c$ for each point in the sparse tensor field prior to voting.

Figure 5.3. 2D and 3D view of a voting field’s ideal attenuation as defined by Eq. (5.5).
5.1 Dynamic $c$ Values

As shown in Figure 5.4 we can vary the $c$ values in order to prefer continuation vs. proximity. This can be valuable if we have an insight into what type of feature we are trying to reinforce. By adjusting the $c$ values to make the value higher on long straight roads we can better fill in gaps (even very large gaps). By making the $c$ values small for roads in towns we can better account for the influence of proximal road segments and preserve smaller features.

Dynamically adjusting the $c$ values is done by taking a histogram of the tangent direction in the tensors surrounding a point. The window size of the histogram is based on the value of $\sigma$ and found by solving for $W_{size}$ in Eq. (5.2). If the mass of the histogram is contained in a small number of bins then $c$ is made higher. If the mass of the histogram is spread out over all the bins and fairly equal, then the $c$ approximates the ideal $c$ value as defined in Eq. (5.5).

5.2 Dynamic $\sigma$ Values

The parameter $\sigma$ adjusts the size of the attenuation field. Larger attenuation fields reinforce features with more extent while smaller attenuation fields are necessary to preserve features with smaller details. To adjust the $\sigma$ dynamically, a count of the features within a window is made in order to determine an estimate of the ideal size of the attenuation field that will preserve smaller features and will reinforce extended ones.
If there is a large number of curves or intersections within the window, then the voting field size may be adjusted smaller in order to reinforce fine details and adequately remove noise. However if the ratio of features to the window size is small it can be assumed that there is a limited number of larger features in the window and thus a larger attenuation field may be used. The value of $\sigma$ is adjusted on a range of acceptable values, between 3 and 20.
CHAPTER 6

POSTPROCESSING TECHNIQUES

The output of the tensor voting system is a pair of real valued curve and junction maps. From this information the system must derive a binary raster map of the curves (roads) found, then combine this information with the junction map to determine the junctions (intersections). This is complicated by the fact that within the curve and junction maps, the likelihood of a feature is relative to its surrounding likeliness. For instance, a curve being filled in by the tensor voting system will have a much lower value within the curve map than a curve which exists in the preprocessed image. Therefore, we must take a smart approach to interpreting the data from these systems.

As seen in Figure 6.1 the junction map and curve map require robust methods to extract information from these raster images. Several different methods are used to extract the features. A process of thresholding and thinning has proved useful. However it misses weak curves in the curve map and is not very effective on junction maps. The process of using a local maximum algorithm that looks at a window, uses the tensors to determine the normal direction to the curve and then finds the local maximum along the normal has proved useful, but fails at curve pixels closer to junction points. The process of a local maximum is mostly useful for finding junctions. Above all, the most effective algorithm is the process of using connected components and knowledge of the features to find curves and junctions.

6.1 Global Thresholding and Thinning

The process of thresholding and thinning as a postprocessing technique begins by calculating an optimal thresholding value from the histogram of the curve and junction maps. To do this an iterative selection method is used. The method begins by finding the maximum value. The histogram is then split into parts based on the half maximum value. The mean of the intensities in the two halves are then calculated, and the average
of the two means produces a new threshold. The process is then repeated with the new value until the threshold converges.

The result of the thresholding can be seen in Figure 6.2 (left). The binary image can then be cleaned and thinned in order to produce a unit width vector feature space (as seen in Figure 6.2 (right)). Thresholding and thinning still have particular problems making it unsuitable for our needs. Thresholding can remove lines and curves in the feature space which are (relatively) pronounced but in context of the entire image very weak. Thinning (along with skeletonization) can also introduce artifacts as seen in Figure 6.2 (right), e.g., the thinned image typically has looped nodules and light noise still apparent in the final image.

Even though thresholding and thinning does not usually provide a perfect solution, it does produce reasonable features and is very efficient in terms of memory requirements.

Figure 6.1. Initial image (left), curve map (middle), junction map (right).

Figure 6.2. Threshold applied image (left), thinned and cleaned image (right).
6.2 Local Thresholding and Thinning

The process of local thresholding works similar to global thresholding. However the method uses a window of 40 by 40 pixel area rather than the entire image. This process helps preserve features where they may be removed by a global thresholding method. The results of the local thresholding can be seen in Figure 6.3.

Local thresholding has similar problems to global. Artifacts are still apparent in the thinned image and some features are removed creating gaps in the data. Local thresholding also introduces noise to an image that must be resolved.

6.3 Local Normal Maximum

Finding the local maximum in the curve map may allow us to figure out the best curve while using the surrounding context of each pixel to determine whether or not the curve passes there. This is important since the likelihood of many reinforced features may not be very high relative to the global curve map, but significantly higher then their neighbors. The process of taking the maximum in the normal direction first identifies the normal axis for each tensor point in the image. Once this is done, the likelihoods are checked along the sides of the current pixel to see if it is in fact a maximum.

The results of this process can be seen in Figure 6.4. While the process does a fairly good job at identifying curves, it has two major problems. First for pixels close to junctions, good curve pixels may not be the local maximum due to interference from other nearby curves. This causes disconnected curves in the output.

![Figure 6.3](image_url)

**Figure 6.3.** Curve map processed by local thresholding and cleaning. Threshold applied image (left), thinned and cleaned image (right).
The other problem with this approach is that small differences in the likelihood that are caused by noise get profoundly exaggerated by this process since its goal is to look for relative maxima in likelihood. While this could be solved by thresholding the image (granted less than what would be needed in thresholding the image for thinning) it eliminates smaller roads and less defined features we want to preserve.

Due to the drawbacks described, this algorithm it is not effective for our needs. It is effective, however, in finding junctions by looking for local maxima in a local area.

### 6.4 Knowledge Based Approach

An effective method to find the best paths in the curve map is a connected component approach. The method begins by producing a binary version of the curve map using local thresholding. The resulting raster image is then thinned. While this produces artifacts, noise and raises issues similar to thresholding and thinning described above, the method then uses a connected component and knowledge based algorithm to remove noise and clean the image. This eliminates most of the artifacts in the image.

The thinned image is broken up by its intersections and then processed by running a connected components analysis. Once the image has been broken up into connected components, end points are found as follows. To find end points a 3x3 image window is taken around each defined pixel and run against a look up table of end points. Each segment from the connected component analysis is examined to see if it lies on an
endpoint. If it does and the line does not cross a border of the image and it is smaller than 15 pixels (a number derived from the maximum size found in noise produced by the process) then it is removed. The results of this process can be seen in Figure 6.5. Any segments found in the connected component analysis not connected to the border of the image are also removed. A combination of logical decisions based on the connected components can help decide whether certain components should be removed and considered noise.

Figure 6.5. Results from thresholding, thinning and knowledge based approach using connected component analysis.
CHAPTER 7

EXPERIMENTS

To determine the performance of the tensor voting, preprocessing and postprocessing methods a framework for evaluating recall and precision is necessary. To evaluate each method a set of one hundred 200 by 200 pixel image samples are taken from various USGS maps. The maps are then processed through the framework and examined against a ground truth for both the roads and intersections.

Experiments are selected to quantitatively answer questions related to performance. Considerations beyond which method performed best are evaluated, and the goals for these experiments are as follows:

- Determine adequate parameters for preprocessing, postprocessing methods and tensor voting.
- Identify weaknesses and strengths of each method.
- Determine the best performing preprocessing method.
- Determine the best performing postprocessing method.
- Quantify the contributions of tensor voting.
- Characterize classification distortion of the preprocessing, tensor voting and post-processing methods on perfect inputs.
- Determine the impact of misclassification of text as roads.

The results for each experiment are published with a recall and precision metric for both roads and intersections. Recall and precision are defined by Eq. 7.1. relevant is the set of pixels which belong to the class and retrieved is the set of pixels which are classified as being in the class. Recall measures the system’s capability to find features while precision characterizes whether it was able to find only those features; both are...
measured on a scale from 0 to 1 where 1 (for both recall and precision) is best. A pixel classified by an algorithm is said to correspond to a feature in the ground truth set if the two are within a specified Euclidean distance (usually 5 pixels).

\[
\text{Recall} = \frac{|\text{relevant} \cap \text{retrieved}|}{|\text{relevant}|}
\]

\[
\text{Precision} = \frac{|\text{relevant} \cap \text{retrieved}|}{|\text{retrieved}|}
\]  

(7.1)

7.1 Data Selection

Data selection is a key aspect of creating accurate and replicable results. To properly select a data set to use in the evaluation, the following constraints are considered:

- The data set must be a large enough to adequately represent features.
- One subimage of the data must not be biased by the selector.
- One subimage may not overlap another.
- A subimage may not be a portion of the map which contains map borders, margins or the legend.

To meet the constraints of data selection a system was built in order to select adequate data sets from two USGS maps (F34086A1.TIF and F34086E1.TIF). The system began by randomly selecting 100 - 200 by 200 pixel area subimages from the maps. The system was constrained to only select regions which were inside the boundaries of the map so that items such as the legend and meta data would not be included in the test samples. The system then went through each map in the sample set and checked for duplicates. The same number number of images were selected from both the F34086A1.TIF and F34086E1.TIF map.

7.2 Ground Truth

Ground truth defines a baseline of what would qualify as a correct answer for both roads and intersections. Once the data selection system finishes, the ground truth is generated from it manually by users. Each image generated by the data selection process is examined and a binary raster map (e.g., see Figure 7.1) is made from it by identifying where the user thinks roads exist. Similarly the intersections are manually identified and
stored in a structured data set. To identify roads the user refers to the legend to make distinctions between geopolitical lines, water, contour lines and roads of all types.

To ensure the ground truth is accurate, the masks are generated twice and the difference between the two is used to identify problems. The ground truth pairs which contained significant differences (above 5%) are re-examined. Neither the original raster map nor the ground truth images were modified so as not to affect the results or bias them for the system. All features defined as roads in the ground truth are marked by a 3x3 white pixel line.

### 7.3 Preprocessing

The three preprocessing methods examined are $k$-Nearest Neighbors, Class Conditional Density Classifier and the Knowledge Based Classifier. The only parameter investigated and varied out of these methods is $k$ in $k$-Nearest Neighbors. The window size of both $k$-Nearest Neighbors and Class Conditional Density Classifier was examined in a qualitative manner to determine the best window size (3x3). To determine the best preprocessing method, an appropriate value must be selected for the parameter $k$. To find the best value for $k$, $k$-Nearest Neighbors is run on the ground truth images over a range of acceptable values to determine which value produces the highest recall and precision. The recall and precision measurements are only calculated on the output of $k$-Nearest Neighbors and no postprocessing methods or tensor voting system is used. Tests run on $k$ from 1 to 20 show little (less than 1%) difference in the recall and precision of the system. The highest value was shown to be at 10 which produced an average road recall of 100%, average road precision of 87%, average intersection recall of 49% and average intersection precision of 11%.
Table 7.1 shows quantitatively that the Knowledge Classifier performed the best (adding together road recall, road precision, intersection recall and intersection precision) and produced similar recall to the other preprocessing techniques but much better precision. The Class Conditional Density Classifier seemed to over-classify items which were more prevalent in the histogram models yet ignore smaller features. The effect of this is shown in Figure 7.2. While this issue could possibly be resolved by using different histogram models, the results for the CCD on various models yielded only small differences in its recall and precision. $k$-Nearest Neighbors did fairly well but still had more noise than the Knowledge Based Classifier.

The knowledge based classifier did better than any other method. However it had difficulties finding smaller dirt and utility roads in the image in addition to introducing enough noise to lower its precision. The naive approach of selecting all black did considerably well performing slightly better than $k$-Nearest Neighbors and the Class Conditional Density Classifier. This was mainly because most road textures end up being primarily black. In addition to this all preprocessing method results go through an open and close

<table>
<thead>
<tr>
<th>Method</th>
<th>RR</th>
<th>RP</th>
<th>IR</th>
<th>IP</th>
</tr>
</thead>
<tbody>
<tr>
<td>$k$-Nearest Neighbors</td>
<td>100%</td>
<td>87%</td>
<td>49%</td>
<td>11%</td>
</tr>
<tr>
<td>Class Conditional Classifier</td>
<td>100%</td>
<td>86%</td>
<td>46%</td>
<td>9%</td>
</tr>
<tr>
<td>Knowledge Classifier</td>
<td>99%</td>
<td>92%</td>
<td>51%</td>
<td>17%</td>
</tr>
<tr>
<td>Naive (Selecting Black)</td>
<td>100%</td>
<td>86%</td>
<td>53%</td>
<td>13%</td>
</tr>
</tbody>
</table>

Figure 7.2. Class Conditional Density Example, Original Image (left) with roads identified (right).
operation which tends to fill the majority of the minor gaps and filter out noise in the naive method.

### 7.4 Tensor Voting

The Tensor Voting system is examined to find its contributions to the overall recall and precision of the framework. To do this an appropriate value for the parameter $\sigma$ must be found. To find the best value the Tensor Voting system is run on the ground truth images with a range of $\sigma$’s. Because the Tensor Voting system requires a preprocessing method and postprocessing method in order to function the naive preprocessing method (selecting all black) and the Knowledge Based Approach postprocessing method are used. These were selected because they do not require parameters and do not need to be tuned for best performance. Therefore we can accurately determine the affect of various $\sigma$’s on the Tensor Voting system. Figure 7.3 shows the effect of varying the $\sigma$ parameter on recall and precision.

The results from this test show that the best value for $\sigma$ is between 10 and 16 with little difference in the performance between these ranges. The value 13 is used as the parameter for $\sigma$ for all further tests.

To determine the Tensor Voting system’s contribution to the overall performance, the framework is run on the ground truth tests with tensor voting and without. The

![Figure 7.3](image_url)

**Figure 7.3.** Effect of varying the sigma parameter of the Tensor Voting system on recall and precision.
framework is run for all combinations of preprocessing and postprocessing methods to
determine the contribution of Tensor Voting for each combination. Table 7.2 shows the
results of recall and precision for all combinations of preprocessing and postprocessing
methods with Tensor Voting. Table 7.3 shows the results of recall and precision for all
combinations of preprocessing and postprocessing methods without Tensor Voting.

From the data in Tables 7.2 and 7.3 one can determine that certain preprocessing and
postprocessing methods actually perform better without tensor voting. Specifically the
global and local thresholds actually perform better simply because they receive a binary

---

### Table 7.2

Comparison of preprocessing (column) and postprocessing (row) methods
with tensor voting. Each cell contains the percentage for (Road Recall / Road Precision /
Intersection Recall / Intersection Precision)

<table>
<thead>
<tr>
<th>k-Nearest Neighbors</th>
<th>Class Conditional Classifier</th>
</tr>
</thead>
<tbody>
<tr>
<td>Knowledge Based</td>
<td></td>
</tr>
<tr>
<td>92% / 95% / 82% / 80%</td>
<td>81% / 98% / 66% / 90%</td>
</tr>
<tr>
<td>99% / 77% / 85% / 26%</td>
<td>98% / 77% / 85% / 26%</td>
</tr>
<tr>
<td>98% / 75% / 82% / 21%</td>
<td>98% / 75% / 82% / 21%</td>
</tr>
<tr>
<td>77% / 40% / 78% / 75%</td>
<td>73% / 31% / 71% / 73%</td>
</tr>
<tr>
<td>100% / 27% / 52% / 5%</td>
<td>100% / 28% / 52% / 5%</td>
</tr>
<tr>
<td>Local Thresh</td>
<td></td>
</tr>
<tr>
<td>99% / 77% / 85% / 26%</td>
<td>98% / 77% / 85% / 26%</td>
</tr>
<tr>
<td>98% / 75% / 82% / 21%</td>
<td>98% / 75% / 82% / 21%</td>
</tr>
<tr>
<td>77% / 40% / 78% / 75%</td>
<td>73% / 31% / 71% / 73%</td>
</tr>
<tr>
<td>100% / 27% / 52% / 5%</td>
<td>100% / 28% / 52% / 5%</td>
</tr>
<tr>
<td>Global Thresh</td>
<td></td>
</tr>
<tr>
<td>98% / 75% / 82% / 21%</td>
<td>98% / 75% / 82% / 21%</td>
</tr>
<tr>
<td>77% / 40% / 78% / 75%</td>
<td>73% / 31% / 71% / 73%</td>
</tr>
<tr>
<td>100% / 27% / 52% / 5%</td>
<td>100% / 28% / 52% / 5%</td>
</tr>
<tr>
<td>Local Max</td>
<td></td>
</tr>
<tr>
<td>77% / 43% / 77% / 73%</td>
<td>71% / 33% / 70% / 71%</td>
</tr>
<tr>
<td>100% / 31% / 52% / 6%</td>
<td>100% / 26% / 52% / 4%</td>
</tr>
<tr>
<td>Naive</td>
<td></td>
</tr>
</tbody>
</table>

### Table 7.3

Comparison of preprocessing (column) and postprocessing (row) methods
without tensor voting. Each cell contains the percentage for (Road Recall / Road Precision /
Intersection Recall / Intersection Precision)

<table>
<thead>
<tr>
<th>k-Nearest Neighbors</th>
<th>Class Conditional Classifier</th>
</tr>
</thead>
<tbody>
<tr>
<td>Knowledge Based</td>
<td></td>
</tr>
<tr>
<td>82% / 97% / 60% / 60%</td>
<td>80% / 98% / 57% / 83%</td>
</tr>
<tr>
<td>99% / 77% / 88% / 26%</td>
<td>98% / 77% / 89% / 33%</td>
</tr>
<tr>
<td>99% / 77% / 88% / 26%</td>
<td>98% / 77% / 89% / 33%</td>
</tr>
<tr>
<td>NA / NA / NA / NA</td>
<td>NA / NA / NA / NA</td>
</tr>
<tr>
<td>100% / 87% / 49% / 11%</td>
<td>100% / 86% / 46% / 9%</td>
</tr>
<tr>
<td>Naive</td>
<td></td>
</tr>
<tr>
<td>82% / 97% / 60% / 60%</td>
<td>80% / 98% / 57% / 83%</td>
</tr>
<tr>
<td>99% / 77% / 88% / 26%</td>
<td>98% / 77% / 89% / 33%</td>
</tr>
<tr>
<td>99% / 77% / 88% / 26%</td>
<td>98% / 77% / 89% / 33%</td>
</tr>
<tr>
<td>NA / NA / NA / NA</td>
<td>NA / NA / NA / NA</td>
</tr>
<tr>
<td>100% / 87% / 49% / 11%</td>
<td>100% / 86% / 46% / 9%</td>
</tr>
</tbody>
</table>
output from the preprocessing method, which is very easy to threshold. When used with Tensor Voting the global and local thresholds must threshold a gray-scale image to a binary image. Therefore it is somewhat expected that the results would be better for the local thresholding and global thresholding without tensor voting. The same can be said for the naive method.

However, Knowledge Based Approach benefitted from using the Tensor Voting system, simply because of its effectiveness in using the output of the Tensor Voting system to accurately identify features. On average the road recall for the Knowledge Based Approach increased by 10%, but the road precision dropped by 2%. The largest benefit is the intersection recall increased by 22%, and the intersection precision increased by 20%.

### 7.5 Postprocessing

The four postprocessing methods examined are the Knowledge Based Approach, Local Thresholding, Global Thresholding and Local Normal Maxima. The only parameter which can be varied out of these methods is the window size of the Local Thresholding. To determine the best postprocessing method, an appropriate value must be selected for the window size. To find the best value for it, the Local Thresholding method is run on the ground truth images over a range of acceptable values to determine which value produces the highest recall and precision. Since the postprocessing techniques require a preprocessing technique and tensor voting they must be used. The tensor voting uses the best value for its $\sigma$ parameter as previously determined. The results of this test can be seen in Figure 7.4. The best value for the window size is determined to be between 10 and 14.

Qualitatively the Knowledge Based System performed the best. Table 7.4 shows quantitatively that the Knowledge Based Approach performed the best. Local thresholding along with global thresholding can produce higher numbers in recall but lower in precision. This is because liberal thresholding lowers the precision, but conservative thresholding lowers recall. The local maximum had a very low recall and precision; this is expected due to the inability of the local maximum to find all the features.

### 7.6 Best Combination

To determine the best combination of preprocessing and postprocessing methods they are combined and run over the ground truth set to see which combination produces the
Table 7.4. Comparison of postprocessing methods with tensor voting and a naive (selecting all black) preprocessing method. (RR = Road Recall / RP = Road Precision / IR = Intersection Recall / IP = Intersection Precision)

<table>
<thead>
<tr>
<th>Method</th>
<th>RR</th>
<th>RP</th>
<th>IR</th>
<th>IP</th>
</tr>
</thead>
<tbody>
<tr>
<td>Knowledge Based</td>
<td>86%</td>
<td>96%</td>
<td>73%</td>
<td>80%</td>
</tr>
<tr>
<td>Local Thresh</td>
<td>99%</td>
<td>79%</td>
<td>86%</td>
<td>30%</td>
</tr>
<tr>
<td>Global Thresh</td>
<td>99%</td>
<td>78%</td>
<td>83%</td>
<td>25%</td>
</tr>
<tr>
<td>Local Max</td>
<td>71%</td>
<td>33%</td>
<td>70%</td>
<td>71%</td>
</tr>
<tr>
<td>Naive</td>
<td>100%</td>
<td>26%</td>
<td>52%</td>
<td>4%</td>
</tr>
</tbody>
</table>

best results. The four preprocessing methods - $k$-Nearest Neighbors, Class Conditional Density Classifier, Knowledge Based Classifier and the naive method (selecting all black) are each run against the five postprocessing methods - Knowledge Based Approach, Local Thresholding, Global Thresholding, Local Maxima and the naive method (selecting everything above 0 as a curve). This yields 80 results (20 combinations and 4 results for each combination).

Table 7.5 shows the results of the test. $k$-Nearest Neighbors as a preprocessing method and Knowledge Based Approach as a postprocessing method performed the best with a road recall of 92%, road precision of 95%, intersection recall of 82% and intersection

Figure 7.4. Effect of varying the window size of the Local Thresholding method on recall and precision.
precision of 80%. For a 95% confidence interval the road recall was \([92.47\%, 94.75\%]\) ± 0.14%, the road precision was \([94.13\%, 96.33\%]\) ± 0.10%, the intersection recall was \([78.91\%, 85.51\%]\) ± 3.29% and the intersection precision was \([76.31\%, 82.99\%]\) ± 2.89%.

### 7.7 Perfect Data

To determine the effects of postprocessing method and the tensor voting system on perfect data the framework was run on the set of ground truth as input. Table 7.6 demonstrates the results of each method run on the ground truth, against the ground truth. The preprocessing methods were not compared or used in the perfect data analysis since the ground truth images are already labeled with the roads. Since preprocessing methods use histogram models to determine initial estimates of labels they are not used.

The results show better recall and precision than any other method or combination of methods that are examined here. The precision of some methods are considerably low due to minor road noise which is introduced by the postprocessing methods that created significant intersection precision problems. The naive, global and local thresholding have this problem. However the knowledge based approach did considerably better at not introducing any noise.

### 7.8 No Text

Extraction of roads is complicated by features in the raster image which have the same texture as roads. Text identifying meta-data in the map has similar (if not the

<table>
<thead>
<tr>
<th>Table 7.5</th>
<th>Comparison of preprocessing (column) and postprocessing (row) methods with tensor voting. Each cell contains the percentage for (Road Recall / Road Precision / Intersection Recall / Intersection Precision)</th>
</tr>
</thead>
<tbody>
<tr>
<td><strong>k-Nearest Neighbors</strong></td>
<td><strong>Class Conditional Classifier</strong></td>
</tr>
<tr>
<td>Knowledge Based</td>
<td>92% / 95% / 82% / 80%</td>
</tr>
<tr>
<td>Local Thresh</td>
<td>99% / 77% / 85% / 26%</td>
</tr>
<tr>
<td>Global Thresh</td>
<td>98% / 75% / 82% / 21%</td>
</tr>
<tr>
<td>Local Max</td>
<td>77% / 40% / 78% / 75%</td>
</tr>
<tr>
<td>Naive</td>
<td>100% / 27% / 52% / 5%</td>
</tr>
<tr>
<td><strong>Knowledge Classifier</strong></td>
<td><strong>Naive (Selecting Black)</strong></td>
</tr>
<tr>
<td>Knowledge Based</td>
<td>77% / 98% / 65% / 86%</td>
</tr>
<tr>
<td>Local Thresh</td>
<td>98% / 87% / 84% / 29%</td>
</tr>
<tr>
<td>Global Thresh</td>
<td>97% / 84% / 83% / 22%</td>
</tr>
<tr>
<td>Local Max</td>
<td>77% / 43% / 77% / 73%</td>
</tr>
<tr>
<td>Naive</td>
<td>100% / 31% / 52% / 6%</td>
</tr>
</tbody>
</table>
Table 7.6. Comparison of postprocessing techniques with tensor voting on perfect data. (RR = Road Recall / RP = Road Precision / IR = Intersection Recall / IP = Intersection Precision)

<table>
<thead>
<tr>
<th>Method</th>
<th>RR</th>
<th>RP</th>
<th>IR</th>
<th>IP</th>
</tr>
</thead>
<tbody>
<tr>
<td>Knowledge Based</td>
<td>94%</td>
<td>100%</td>
<td>80%</td>
<td>86%</td>
</tr>
<tr>
<td>Local Thresh</td>
<td>100%</td>
<td>98%</td>
<td>86%</td>
<td>46%</td>
</tr>
<tr>
<td>Global Thresh</td>
<td>100%</td>
<td>96%</td>
<td>83%</td>
<td>36%</td>
</tr>
<tr>
<td>Local Max</td>
<td>80%</td>
<td>77%</td>
<td>75%</td>
<td>72%</td>
</tr>
<tr>
<td>Naive</td>
<td>100%</td>
<td>37%</td>
<td>52%</td>
<td>8%</td>
</tr>
</tbody>
</table>

same) texture as roads, and to extract these a method which relies on structure rather than texture is necessary. To compute the effect of text on the ground truth, the text was manually removed from each sample image and rerun on the ground truth. Table 7.7 shows the results (Recall and Precision) of the samples once text has been manually removed.

Overall removing the text from the map data increased the recall by 1% for roads and 2.5% for intersections, the precision was not significantly affected (less than 1% difference). The best combination was still $k$-Nearest Neighbors and Knowledge Based Approach for the preprocessing and postprocessing systems. The road recall was 94%, the road precision was 95%, the intersection recall was 83% and the intersection precision was 80%.

Table 7.7. Comparison of preprocessing and postprocessing methods run on ground truth images without text. (Road Recall / Road Precision / Int. Recall / Int. Precision)

<table>
<thead>
<tr>
<th>Method</th>
<th>$k$-Nearest Neighbors</th>
<th>Class Conditional Classifier</th>
</tr>
</thead>
<tbody>
<tr>
<td>Graph Search</td>
<td>94% / 95% / 83% / 80%</td>
<td>83% / 77% / 71% / 80%</td>
</tr>
<tr>
<td>Local Thresh</td>
<td>99% / 76% / 85% / 26%</td>
<td>99% / 76% / 85% / 26%</td>
</tr>
<tr>
<td>Global Thresh</td>
<td>99% / 75% / 81% / 21%</td>
<td>98% / 75% / 83% / 22%</td>
</tr>
<tr>
<td>Local Max</td>
<td>77% / 40% / 78% / 75%</td>
<td>73% / 31% / 71% / 73%</td>
</tr>
<tr>
<td>Naive</td>
<td>100% / 26% / 52% / 5%</td>
<td>100% / 27% / 52% / 5%</td>
</tr>
</tbody>
</table>

| Knowledge Classifier    | Naive (Selecting Black)        |
| Graph Search            | 80% / 85% / 68% / 80%          | 89% / 96% / 78% / 84%         |
| Local Thresh            | 98% / 87% / 84% / 29%          | 99% / 79% / 87% / 31%         |
| Global Thresh           | 98% / 84% / 83% / 23%          | 98% / 77% / 84% / 25%         |
| Local Max               | 77% / 42% / 78% / 73%          | 72% / 33% / 71% / 71%         |
| Naive                   | 100% / 30% / 52% / 5%          | 100% / 25% / 52% / 5%         |
7.9 Comments

The quantitative examination of preprocessing and postprocessing systems showed $k$-nearest neighbors and the knowledge based approach to be the best combination found. The text in the raster maps had a significant impact on precision especially on intersections but did not change the results of recall dramatically. Interestingly the Knowledge Based Classifier as a preprocessing method was found to be the best preprocessing method when no postprocessing method or tensor voting system was used. However, when run in combination with others it did not perform as well as other preprocessing methods. The reasons behind this are somewhat complicated. The Knowledge Based Classifier produced a larger amount of sparse noise in the images than any other method which caused the tensor voting system and some postprocessing systems to inaccurately identify curves. While both the Class Condition Classifier and $k$-Nearest Neighbors both were almost as effective but performed worse in the independent preprocessing analysis, neither introduced the same type of noise (it was more dense rather then sparse) which affects the tensor voting system in a way which instead of removing the noise, it actually exaggerates.

Dynamically adjusting the $\sigma$ and $c$ parameters for the tensor voting produced insignificant results. The $\sigma$ was varied from 4 to 20 by a dynamic $\sigma$ system described previously, the results of dynamically adjusting the $\sigma$, however, were somewhat irrelevant to the overall performance only affecting the average recall and precision by $\pm 1\%$. Dynamically adjusting the $c$ value for the tensor voting actually produced worse performance. The average decline in performance over all methods was 18.3%. This was due to slight variations in the tangent direction for each tensor in the field. If a slight variation occurs while the $c$ prefers a stick vote it tends to produce misaligned roads and lines between points in the final curve map produced.
CHAPTER 8

CONCLUSIONS AND FUTURE WORK

A method of $k$-nearest neighbors, tensor voting and the knowledge based approach produced the best results for both roads and intersections. $k$-nearest neighbors showed a unique ability to overcome noise, variances in texture for features and to produce smooth results for the tensor voting system. The tensor voting system showed significant capabilities for filling gaps, removing noise and finding accurate junctions in the output raster image. The knowledge based approach was most effective at producing unit width binary raster map from the output curve of the tensor voting system.

The runtime performance of the system was decent, the real time to run the tensor voting on a 200 by 200 area was 22 seconds on average on a Pentium Dual Core 2.9ghz processor under Matlab. The total runtime of the system was 60.1 seconds for a 200 by 200 area. It should be mentioned that the implementation of this was not optimized and significant runtime performances could be made through different languages and optimizations.

The largest problems with extracting features was preserving features close together within the raster map. Nearly every method investigated had an inability to properly segment features when they became closely grouped with the exception of the knowledge based classifier which qualitatively did the best job at this. Another significant problem was roads which intersect and are at a small angle to each other or have fairly thick lines that must be thinned. Figure 8.1 shows a road in the lower right hand corner which intersects with another at a small angle and has thick overlapping lines. When this occurs instead of forming one intersection the tensor voting system curves the two roads together prematurely and produces two intersections. The output of the tensor voting system is shown in Figure 8.2 which demonstrates how the intersection is broken in two.

Text is also a significant challenge to overcome in the raster maps. The text can appear in any orientation, can have different font types, seems to be bolder in some parts then others, overlaps existing features, has the same texture as features in the raster and
Figure 8.1. Sample of intersecting roads that are too close.

Figure 8.2. Sample of the output from tensor voting and postprocessing method for intersecting roads that are too close.

has various sizes. In order to overcome this a method using structural analysis would be necessary.

8.1 Future Work

To overcome the existing problems with effectively identifying features a system to extract or produce a bounding box around text to adequately remove it from the feature set is necessary. In addition to this some sort of system to produce smooth straight features when features are closely grouped together or linear features intersect at a small angle. Both of these problems had the most significant effect on the performance of the system.
APPENDIX

1. Low Level Features - Lines, curves, junctions, regions and end points

2. Semantic Features - Histogram data, color, proximity and other information combined with lower order features to extract features in terms of maps such as roads, interstates, highways, etc.

3. USGS - United States Geological Survey Maps

4. Raster - A data array of pixel values for an image, complemented with a colormap makes an image.

5. Thinning - creating a skeleton of original linear features by eroding border components to a single unit width while preserving extremities (end points).

6. Fitting - line and curve fitting, approximating a linear feature to the smoothest curve or straightest line.


8. Junction Analysis - Identifying intersections between two or more linear features.

9. Region Boundary Extraction - Identifying enclosed regions and their outlying boundaries.


11. Segmentation - A process of determining separation of low level features in the raster set.
REFERENCES


