01 May 1993

Cartographic Pattern Recognition using Template Matching

Angela G. Lammers

Ralph W. Wilkerson
Missouri University of Science and Technology, ralphw@mst.edu

Fikret Erçal
Missouri University of Science and Technology, ercal@mst.edu

CARTOGRAPHIC PATTERN RECOGNITION USING TEMPLATE MATCHING

A. Lammers* and R. W. Wilkerson and F. Ercal

CSc-93-07

Department of Computer Science

University of Missouri - Rolla

Rolla, MO 65401 (314)341-4491

*This report is substantially the M.S. thesis of the first author, completed May 1993.

ABSTRACT

In creating digital maps from paper maps, the paper map must first be scanned to produce a raster image, and then converted into vector format. Vector format allows non-graphical cartographic information to be stored along with the graphical objects. At the United States Geological Survey, the conversion from raster to vector format is performed by a commercial software package. The package also attempts to classify the graphical objects based on shape, line patterns, and other information gained from the raster file. Since the package frequently fails to classify a significant percentage of the elements in the map, manual map analysis and classification, a slow and costly process, is necessary.

The current project implements template matching in an effort to reduce the amount of manual analysis necessary for hydrography files (files containing all water data from a map). Lines appearing in a hydrography file are either solid (shoreline or perennial streams) or made up of the repeated pattern of a dash and three dots (intermittent streams). The Wise Intermittent Stream Recognition and Detection (WISRD) system has been created to identify lines having the intermittent stream pattern, thereby decreasing the amount of manual editing necessary for hydrography files.

The choice of template matching as a pattern recognition technique has proven to be quite beneficial in the classification process. The WISRD system succeeded in significantly reducing the number of unclassified lines upon its completion.

TABLE OF CONTENTS

Page

LIST OF ILLUSTRATIONS

I. INTRODUCTION

Artificial intelligence is one of the most controversial topics in computer science. Ever since thinking machines were predicted as the next big revolution in computers in the 1950's, advocates and skeptics of AI have been waging a war of words. This war is not only about the pros and cons of machines which are able to reason and learn, but also, fundamentally, about whether the creation of such machines is even possible. Even the fact that the overzealous predictions of the 1950's have not been realized adds fuel to the argument that AI in the form of machines which truly think is not possible. Steven Tanimoto (1990), a proponent of AI, suggests two additional reasons for the controversy. The first is that AI, until recently, has not been given much public attention. Actual examples of machine expertise, he says, are not common in everyday life, making it difficult for those outside the field to believe that artificial things could act intelligently. The second reason Tanimoto lists for the skepticism of some is one's own perception of human intelligence. Admitting that a machine might be able to think as well as a human would tend to lower the status of humans to the mechanical level. Tanimoto suggests that most people would have a psychological aversion to such a thought, and therefore would reject any notion which might have the effect of placing humans and machines on the same level.

While these arguments may be true, the fact remains that the thinking, reasoning machines predicted so many years ago do not at this point exist. Tanimoto says artificial intelligence is "a technology concerned with the processes of reasoning, learning, and perception" (p. 6). These three abilities must be present in a system for it to be considered truly intelligent. This is not to say that systems cannot be developed which *simulate* intelligence. A system which contains built-in knowledge used in reasoning, but which, for instance, does not learn can be said to simulate intelligence. To the outside observer the system appears to be intelligent because it responds to (predefined) situations in an intelligent and reasonable manner, but the system cannot generalize existing knowledge for use in undefined situations. Such systems might be called wise rather than intelligent--the built-in knowledge in the system is analogous to wisdom in humans[1]. Wise systems must be defined carefully so as to eliminate or at least minimize the possibility of undefined states. Thus, it is usually necessary to enumerate every possible known situation, as well as provide broad exception handling for the inevitable undefined state. For this reason, wise systems are best suited for problems in which exhaustive state definition is possible, or where undefined states are very rare.

One example of a problem which is a good candidate for a wise system occurs in cartography. In order to conserve manhours for more complex tasks, it is often desirable to have an automated system which is able to recognize different symbols or patterns which occur on maps. Such a system would have various applications, including automated map analysis and geographic information systems (GIS). The finite area covered by a map along with the finite number of symbols which can appear on a map make cartographic pattern recognition a prime candidate for a wise system application. A system could be created, for instance, which has access to all the possible symbols which might appear on a specific kind of map and is preprogrammed with all the rules for symbol classification. The current project, Wise Intermittent Stream Recognition and Detection (WISRD), represents such a system. WISRD has been designed as an

[1] Wisdom is a synonym for knowledge in the American Heritage Desk Dictionary.

enhancement to an existing system already in use at the United States Geological Survey in Rolla, Missouri. The purpose of WISRD is to shorten the manhours necessary when editing hydrography files--digital maps which contain water data (flowing water such as streams, standing water such as lakes, and wetlands such as marshes) for the area of coverage. Nearly all the elements in the hydrography file can be characterized as either perennial or intermittent. Perennial elements are represented by solid lines, while intermittent elements are represented by patterned lines consisting of the repeated sequence of a dash and three dots. WISRD implements template matching, a popular and simple pattern recognition technique. The knowledge built-in to WISRD, then, includes a dot template and the rule that an intermittent stream is characterized by sets of three dots separated by single dashes. Full advantage has been taken of any simplifying aspects of this particular application in order to increase both efficiency and accuracy.

A similar system developed by Tushar J. Amin and Rangachar Kasturi (1987) is able to classify several different patterns of lines. The lines classified consist only of various lengths of dashes. A type 1 line in the system is a solid, unbroken line, and then further line types are defined based on the relative lengths of the dashes (line segments) and gaps between dashes which make up the line. The file is initially processed to produce a list of all the individual line segments which occur therein. All line segments which are beyond a specific threshold in length are classified as type 1. Those segments which fall below the threshold in length must then be assigned to a particular line type, and then to a particular line. The length of the segment determines the line type to which it is assigned. In assigning a segment to a line, the gaps between the beginning (ending) point of the segment and the ending (beginning) point of each line of the same type already recognized are computed. These gap distances are examined to see if they fall within the inter-segment gap range of the particular line type which has been assigned to the segment. If none do, the segment is said to begin a new line. If only one of the distances calculated falls within the range, the segment is assigned to that line. If more than one line comes within the proper range, the segment is assigned to the line with closest direction to its own.

There are several factors which make the above approach unsuitable for the current project. The first is that whereas the above system classifies patterned lines which consist only of dashes, the current system must classify patterned lines of dashes and dots. Amin and Kasturi are able to determine line direction from each line segment by computing its slope, but this information is not available from a single dot. In addition, Amin and Kasturi are able to assign a dash to a line type based solely on its length. In so doing, they are relying heavily on the accuracy of the original map and the scanning device used to create its digital representation. This reliance is not always feasible in real world applications. In many of the maps used to test the current system, the length of the dashes and gaps between dashes and dots varies radically depending on the terrain being represented, so much so that it was determined immediately that relative distance between consecutive points along a line would not be an effective measure of line type. There is an additional error factor: human error. Since the original lines were placed by cartographers manually, no assumptions can be made about what the distances "should" be, as these will vary from map to map and from line to line in the same map.

Following is a more detailed description of the current project, as well as a discussion of some of the issues which surround the project. The second chapter examines the effect that computers have had on cartography. It also describes two of the ways that digital cartographic data is represented. Chapter three contains a comparison of human and computer pattern recognition and introduces three different methods of computer pattern recognition. Chapter four describes the existing system and discusses the desired enhancements that led to the creation of the WISRD system. It then specifies the aspects of the problem which make it a good application of template matching, and details the way in which template matching was applied to solve the problem of intermittent stream recognition. Chapter five gives the results of using the WISRD system on ten different digital maps selected from those created at the U. S. Geological Survey in Rolla, Missouri. It offers a rough analysis of the added efficiency afforded by WISRD versus manual classification, and discusses the specific situations in which WISRD will fail to correctly classify lines. The present paper concludes with a discussion of further work which could be undertaken to improve the recognition and classification rates.

II. COMPUTER CARTOGRAPHY

A. ADVANTAGES OF COMPUTERS IN CARTOGRAPHY

Cartography is one of the many fields which have been revolutionalized by the computer. Prior to the electronic age, maps could only be produced by physically exploring an area and hand-recording its topography. One needs only to look at maps dating from shortly after the discovery of the New World to realize how subjective early map-making was. In these maps we recognize the familiar shapes of the European continent, but find that the outline of North America is severely lacking in accuracy.

A chronological study of several maps created one or two hundred years apart will reveal a steady increase in precision. This is a natural process as the map-makers become more familiar with the area being mapped and cartographic tools become more sophisticated; maps made one hundred years ago can be said to be fairly correct. Problems are created, however, when the area mapped is changed. Any map is accurate only as long as it accurately reflects its corresponding area of the world. In the case of a handmade map, the addition of a road or the creation of a lake contributes to the overall inaccuracy of the map. Eventually, the map will become completely obsolete. When this happened before the electronic age, a new map had to be entirely remade from scratch.

Prior to the Industrial Revolution, such change was gradual enough that the accuracy of even an old map (assuming that it was fairly accurate to begin with) was not diminished significantly over time. It was infrequent that a map had to be remade due to manmade changes to the world it depicted. Today, however, this is no longer the case. One need only to imagine producing handmade maps in the months following the dissolution of the Soviet Union to realize what a boon the computer is to cartography.

But computers mean more to cartography than just quick revisions. They have also permitted mass production which, as any armchair economist knows, creates lower prices. It is due to computers that we are able to walk into any gas station and purchase a map for a few dollars. But perhaps the most significant advantage computers bring to cartography is their ability to store information. Not only can a computer store a line representing a street, but it can store the fact that the line represents a street as well as the name of the street. This ability to store information along with a map has made possible such products as geographic information systems (GIS) and cartographic databases.

In order to be symmetrical about this discussion, we should now take a look at some of the disadvantages computers have brought to cartography. According to Keith C. Clarke (1990) there are two fundamental disadvantages posed to cartography by computers. The first, he says, is the additional amount of training cartographers of today require.

> ...the amount of technical training that a cartographer has to acquire has increased enormously. The cartographer of the nineteen nineties must be a data-base expert, a user-interface designer, a software engineer, retain a sense of map esthetics, and still produce maps (p. 9).

The second disadvantage Clarke cites is that now one need not have a background in cartography to produce a map. While he says this popularizes map-making, it also causes inferior maps to be produced. Thus the computer has created a cartographic paradox: it allows us to create better maps which may also be of lower quality!

B. VECTOR VS. RASTER GRAPHICS

When a map is stored in a computer (henceforth referred to as a digital map), there are at least two ways in which the data can be stored. If the data are stored in raster format, the area for the map can be thought of as covered by a fine grid, where the resolution of the grid is equivalent to the resolution of the map (see Figure 1b.). The grid produces regular-shaped areas (pixels) in rows and columns, each of which can hold a value. If, for instance, the map is an elevation map, each pixel would hold the elevation which corresponds to that pixel's real-world location. In a color or gray-scale digital raster map, each pixel can hold a color/gray-scale value. When each of these pixels is displayed in the same row and column in which it is stored, a color/gray-scale image of the mapped area would be produced.

A simplified form of a gray-scale digital raster map is a binary (or bilevel) map. In a binary raster map, each pixel contains either a 1 or a 0. A 1 corresponds to a foreground pixel, and a 0 corresponds to a background pixel. When a binary map is displayed, the foreground is usually black and the background is usually white. An advantage of a binary digital map over a color/gray-scale digital map is that only one bit is needed for each pixel, whereas a minimum of one byte is usually necessary for each pixel of a color or gray-scale digital raster map.

One way of collecting raster data is by scanning. In the case of the current system, a binary mylar map (either negative or positive) is scanned at a resolution of 1000 pixels per inch. Since that resolution produces between 125K and 1M of data for every square inch, and the maps scanned are 19 inches by 23 inches, it is not surprising that various compression methods are used to conserve disk space. One method of compression is to store data in run-length encoded format. Under color or gray-scale run-length encoding, each scan line is stored as a series of tuples, the first number representing the number of times the second number, the data value of each pixel, is consecutively repeated along the row (Star & Estes, 1990). This method is simplified in a binary map because each of the two possible pixel values (representing foreground and background) cycles repeatedly across the horizontal scan line. All that remains is to arbitrarily pick one of the values to invariably begin the line, and each consecutive sequence of pixels with the same value can be represented as a single number. For instance, the following scan line of binary data, consisting of 50 pixels:

00000000111111111111111111110000000111110000111111

can be run-length encoded as:

8 20 7 5 4 6

It is this method of data compression that the current system uses for raster files.

A second means of storing digital maps is vector format. Whereas raster files are pixel-oriented, vector files are point-oriented (see Figure 1c.). An object in vector format is stored as a list of its vertices. A square, for instance, is stored as a sequence of four points, one for each corner. Most elements in vector files are composed of the short line segments which connect the vertices associated with each object. The exceptions to this rule are curves and the conic sections. In these cases, the coefficients necessary to define the objects (along with the center point or other points of reference) are stored. At any rate, point locations play a major role in vector graphic files.

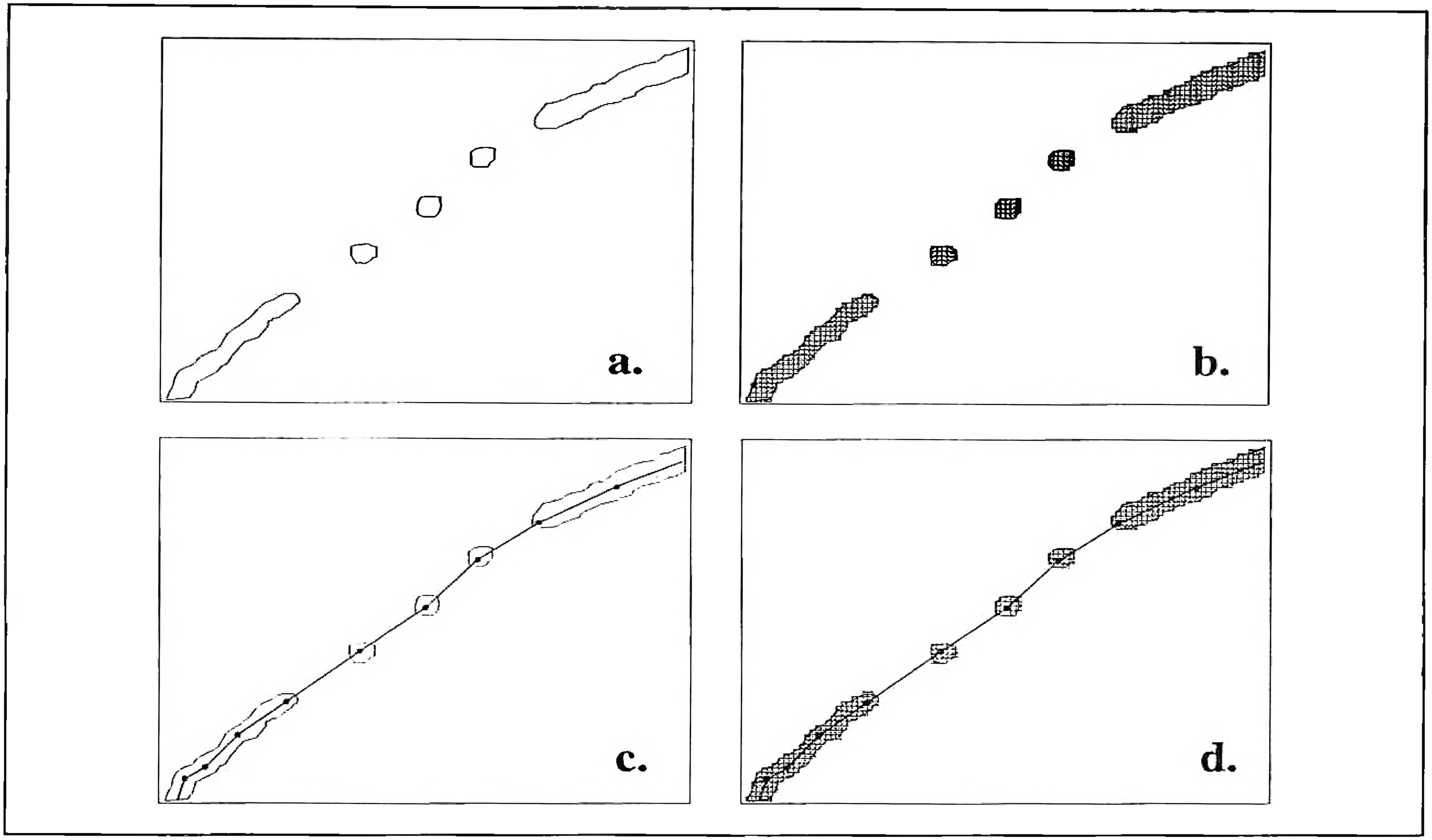


Figure 1 Raster vs. vector representation: a. the original image; b. the raster representation; c. the vector representation; d. raster and vector representations superimposed upon the original image.

As one would guess, there are advantages and disadvantages to using either raster or vector file formats. A raster file can be scanned quite easily from a printed map, whereas the vector file must either be created from the raster file (see section II.C. below), or by using a manual digitizing table. The space requirement for raster files, which has previously been addressed, is usually much greater than for vector files. Vector files are easier to edit than raster files. To shift one or more vertices of a line, all that is necessary is to change the stored coordinate values. In contrast, each affected pixel in a raster file must be edited individually to accomplish the same task. A final (and most significant) advantage is the amount of non-graphic information which can be stored in vector format. Along with each element in the vector file, information can be stored about the type of element represented and its geographic location. On the other hand, the information that can be stored in a raster file is limited to what can be quantified into a number represented by a color or gray-scale value.

C. RASTER TO VECTOR CONVERSION

According to Keith C. Clarke (1990), the use of scanners to create raster maps has amplified the need for efficient--yet accurate--methods of converting raster data to vector data. Says Clarke, "This process is very CPU intensive, and can yield topological and other errors in the resultant vector data set regardless of the quality of the input data" (p. 193f.). Clarke goes on to enumerate the three steps required in the raster-to-vector conversion process: line thinning, line extraction, and topological reconstruction.

A	A	A
0	P	0
B	B	B

A	0	B
A	P	B
A	0	B

A	A	A
A	P	0
A	0	S

A	A	A
0	P	A
S	0	A

S	0	A
0	P	A
A	A	A

A	0	S
A	P	0
A	A	A

where:
P denotes the pixel in question
S denotes a skeletal pixel
0 denotes a background pixel
A, B denote groups of pixels, at least one of which
must be set in order for pattern to match

Figure 2. Classical Line Thinning Patterns, from (Pavlidis, 1982, p. 198)

1. Line Thinning. In raster-to-vector conversion, the goal is to create thin vector lines (i.e., lines having zero thickness) which connect points lying on the raster objects. In addition, preservation of the relative shapes and sizes of all objects is required. The first step in this process is to extract thin raster lines from the raster image. Raster lines are thin if they have a width of one pixel and are completely connected. One method of line thinning is peeling, in which the edge pixels of raster objects are systematically deleted until one is left with a single line of pixels, representing the skeleton of the object. Pavlidis (1982) describes several line thinning algorithms. In the classical thinning algorithm, each pixel with a value of 1 (foreground), having a 4-neighbor with a value of 0 (background) is compared with each of six patterns (see Figure 2). If the neighborhood of the pixel matches one of the patterns, it is flagged as a skeletal pixel,

otherwise it is deleted. This process peels the edge pixels in layers until only the skeletal pixels remain.

2. Line Extraction. Once the skeletons of the raster file have been found, the individual lines must be examined to determine where they begin and end. These beginning and ending nodes will be used as the points along the vectors. Clarke (1990) and Pequet (1981) list two possible methods of line extraction. In the first, line following, each individual pixel along a line is tracked, and the beginning and ending nodes are noted. The second method, the scan-line approach, tracks pixels in a similar manner as line following, with the exception that all pixels along the current horizontal scan line are followed so that several lines can be tracked simultaneously. The beginning and ending nodes found for each line during line extraction are then used as beginning and ending points for the vectors in the vector representation.

3. Topological Reconstruction. After the individual lines of a particular object have been extracted, they must be associated with one another so as to maintain the object's original topography. To this end, the vector representation of the object contains the beginning and ending nodes of the individual lines stored, in the order in which they are originally found. This approach reconstructs the object represented in the raster file in vector file format. In addition to the topography of an object, a raster-to-vector conversion system may also examine the information in the raster file in order to store information about each object in a vector file.

It is with this step in raster-to-vector conversion that the current project deals. Most maps make use of patterned lines to distinguish different types of the same object. For instance, a paved road can be represented by a solid, unbroken line, while a dirt road may be represented by a dotted or dashed line. While this information is preserved in the raster representation of the map, it will be lost in the vector representation if steps are not taken to preserve it. This problem arises because it is more desirable to maintain the linear integrity of a patterned line (dashed, dotted, or otherwise) than it is to vectorize each dot or dash individually. The vector representation of a dashed line will look identical to that of a solid line. For this reason, we must find ways to recapture any pattern information lost in the vectorization process. In addition, we must also leave ourselves open to the necessity of differentiating between patterns. A truly useful system would be able to distinguish a dashed line, a dotted line, and a line which alternates dashes with dots.

III. PATTERN RECOGNITION

A. HUMAN PATTERN RECOGNITION

Pattern recognition is so intrinsic to ourselves as thinking beings that we hardly realize when we perform it. Say Julius Tou and Raphael Gonzalez:

> Recognition is regarded as a basic attribute of human beings, as well as other living organisms. A pattern is the description of an object. We are performing acts of recognition every instant of our waking lives. . . . A human being is a very sophisticated information system, partly because he possesses a superior pattern recognition capability. (p. 5, 1981)

Pattern recognition is one of the things that we as humans do exceptionally well, and because it is so fundamental, it is only natural that we should want to create machines with similar abilities. In the early years, computers were touted as "thinking machines." Their first uses, numerical calculation and code breaking, were offered as proof that thinking was taking place. But today we think of such processes as mechanical, rather than intelligent (Mishkoff, 1985). The progress in creating machines which truly think has been hampered by the fact that we still have not fully grasped how our own minds work. It is very difficult to model something of which one has incomplete knowledge.

Yet pattern recognition is and will continue to be an area of computer science in which a good deal of activity takes place. It would seem that anything performed so readily and easily by a human being would have real and far-reaching benefits if it could be accomplished by a machine.

B. EXAMPLE COMPUTER PATTERN RECOGNITION METHODS

Most pattern recognition techniques seek to mimic the processes which occur in the human brain. The problem with this strategy is that there exist numerous psychological and physiological models of the way in which the brain recognizes patterns. Thus, there are just as many computer pattern recognition techniques. According to Tou and Gonzalez,

> Human recognition is in reality a question of estimating the relative odds that the input data can be associated with one of a set of known statistical populations which depend on our past experience and which form the clues and the *a priori* information for recognition. (p. 5, 1981)

Such a model of human pattern recognition smacks of a statistical technique. Another technique, artificial neural networks, attempts to mimic what happens in our brains at a physiological level. Template matching, a third method, tries to mimic the process some psychologists say humans go through when we compare input received to master copies of patterns in the brain. In this case, recognition takes place when the input is matched with the template it most closely resembles (Eysenck, 1990).

That there are numerous pattern recognition techniques in use today does have some advantages. Each one, it seems, is most successful in a particular application. For this reason, most complex pattern recognition applications are only accomplished as a combination of two or more different techniques.

1. Statistical Pattern Recognition. Julius Tou and Raphael Gonzalez (1981) describe statistical pattern recognition as a "two-person zero-sum game" (p. 111) which pits nature as player one against player two, the pattern classifier. Tou and Gonzalez describe a game G as a triplet (Y, Z, L), where Y is the set of all classes in nature, ω_i,

$i=1,2,\dots,M$; Z is the set of all possible decisions by the pattern classifier, a_j, $j=1,2,\dots,N$; and L is the loss matrix where L_{ij} is the loss incurred by assigning a member of class ω_i to class ω_j. In this particular application of the zero-sum game to pattern classification, we will limit ourselves to cases where $M = N$ and $L_{ij} = 1$ when $i \neq j$, 0 when $i = j$.

The game is played when nature selects a class ω_i based on the *a priori* probability function $p(\omega_i)$ of class ω_i. From this class, a sample pattern x is produced and presented to the pattern classifier. Because the classifier does not know from which of the M possible classes x came, it must make its decisions so as to minimize the expected loss for each decision, thus minimizing the total expected loss. The expected loss for assigning pattern x to class ω_j can be expressed as

$$\ell_j(x) = \sum_{i=1}^{M} L_{ij} p(\omega_i | x) \tag{1}$$

where $p(\omega_i | x)$ is the probability that x comes from ω_i. Using Bayes' formula:

$$p(\omega_i | x) = \frac{p(\omega_i) p(x | \omega_i)}{p(x)} \tag{2}$$

with (1) and multiplying by $p(x)$, we have:

$$p(x)\ell_j(x) \equiv r_j(x) = \sum_{i=1}^{M} L_{ij} p(x | \omega_i) p(\omega_i) \tag{3}$$

But since $L_{ij} = 1$ when $i \neq j$ and 0 when $i = j$, we can say $L_{ij} = (1 - \delta_{ij})$ where $\delta_{ij} = 1$ when $i = j$ and $\delta_{ij} = 0$ when $i \neq j$. From this and (3) we now have:

$$\begin{aligned} r_j(x) &= \sum_{i=1}^{M} (1-\delta_{ij}) p(x|\omega_i) p(\omega_i) \\ &= p(x) - p(x|\omega_j) p(\omega_j) \end{aligned} \qquad (4)$$

In a game with an arbitrary number of classes, M, a pattern x will be assigned to class ω_i if $r_i(x) < r_j(x)$ for $j=1,2,\ldots M$; $j \neq i$, equivalently,

$$p(x|\omega_i) p(\omega_i) > p(x|\omega_j) p(\omega_j), \quad j=1,2,\ldots,M;\ j \neq i \qquad (5)$$

From this and (2), we can create decision functions:

$$d_i(x) = p(x|\omega_i) p(\omega_i), \quad i=1,2,\ldots,M \qquad (6)$$

$$d_i(x) = p(\omega_i|x) p(x), \quad i=1,2,\ldots,M \qquad (7)$$

where a pattern x is assigned to class ω_i if $d_i(x) > d_j(x)$ for all $j \neq i$.

Having derived two equivalent decision functions, all that remains is finding functions $p(x|\omega_i)$ and $p(\omega_i)$ (if using (6)), or function $p(\omega_i|x)$ (if using (7)). (Note that the $p(x)$ term can be dropped from (7) as it does not depend on i and is therefore not necessary.) According to Tou and Gonzalez (1981), $p(\omega_i)$ ought to be derivable from knowledge about the classes and/or from test cases. (One simple approach would be to set $p(\omega_i)$ equal to the size of class ω_i divided by the total number of possible patterns x.) They then describe various methods of arriving at $p(x|\omega_i)$ and $p(\omega_i|x)$. The interested reader is asked to consult Chapters 4 and 6 of Pattern Recognition Principles (1981) for further information.

2. Neural Network Pattern Recognition. A second method of pattern recognition is performed using artificial neural networks (ANNs). Artificial neural networks

represent an attempt to mimic the biology and physiology of the brain (human or otherwise), thus any discussion of ANNs is usually replete with terms such as neuron, synapse, inputs, outputs, and activation. According to Robert Schalkoff (1992), ANNs are popular in pattern recognition applications because, in contrast to statistical methods, little previous knowledge of class distributions in the natural world--or even of the network itself--is required. Says Schalkoff:

> To some extent, the [neural network pattern recognition] approach is a *nonalgorithmic, black box strategy*, which is trainable. We hope to 'train' the neural black-box to 'learn' the correct response or output (e.g., classification) for each of the training samples. This strategy is attractive to the [pattern recognition] system designer, since the required amount of a priori knowledge and detailed knowledge of the internal system operation is minimal. (p. 205-6)

An ANN, of which there are numerous models and paradigms, can be characterized by three qualities: the topology (interconnection pattern) of the neurons, the behavior of each neuron, and the method of training used. In Pattern Recognition: Statistical, Structural, and Neural Approaches, Schalkoff lists several reasons for using a neural network approach to pattern recognition. One advantage of ANNs he says, is that the computation process is inherently parallel. This provides us with a system which not only proceeds more rapidly, but is also more robust. Loss of a few processors should not significantly degrade system performance. Because of the parallelism built in to an ANN, the computation performed in each neuron is simple. Due to the fact that there are hundreds or thousands of neurons, each individual neuron usually only performs a summation or simple nonlinear operation. Finally, Schalkoff adds that in creating an electronic system to simulate a process that our brains perform almost constantly, it is probably desirable that the system emulates the biological process as closely as possible.

Schalkoff does note, however, that "what is not simple is the mapping of an arbitrary [pattern recognition] problem into a neural network solution" (p. 210).

3. Template Matching. Another form of pattern recognition is template matching. In this method, an input image is scanned for the occurrence of any of a known set of patterns. The image (or parts thereof) is then classified based on the presence or absence of each pattern. Each pattern in question is represented by a template, a one- or two-dimensional image which will be compared to various areas of the input image. At each comparison point, the correlation between the template and the current area of the input image is calculated. The higher the correlation of the two areas being compared, the closer the match between those areas. When searching for specific patterns within an input image, it is usually advantageous to establish a correlation threshold value below which two patterns are said not to match.

The opposite of determining correlation is to compute the mismatch of two patterns. Obviously correlation and mismatch are inversely related, so that a low (or zero) mismatch indicates a high (or perfect) correlation. Schalkoff (1992) lists two equations for calculating the mismatch (norm) of a template and an area of an input image:

$$m_1=\sum_R |f-g| \tag{8}$$

$$m_2=\sum_R (f-g)^2 \tag{9}$$

where:

f : The template.
g : The input pattern.
R : The area in g covered by f.

Again, it is usually desirable to have a mismatch threshold, above which two patterns are said to be different.

Equation (9) may be expanded to

$$m_2 = \sum_R f^2 - 2\sum_R fg + \sum_R g^2 \tag{10}$$

Since $\sum f^2$ is constant at every match, and $\sum g^2$ will be the same at that location for each template, the meaningful term from (10) is the unnormalized correlation:

$$c_{un} = \sum_R fg \tag{11}$$

Since the sign of c_{un} is negative in (10), the mismatch will be low when c_{un} is high. c_{un} therefore is directly related to the measure of correlation.

In a given pattern recognition application, the areas of the input image in which each of the template patterns is likely to occur is usually not known in advance. For this reason, it is necessary to shift the template over the entire image when searching for a specific pattern, calculating the correlation (mismatch) at each location. The following simple example from Schalkoff (1992) illustrates this process in one dimension:

Template Pattern:

$\underline{w} = 1 \quad 2 \quad 3 \quad 4$

Input Pattern:

$\underline{x} = 7 \quad 6 \quad 3 \quad 4 \quad 1 \quad 2 \quad 4 \quad 3 \quad 1 \quad 2 \quad 3 \quad 4 \quad 5 \quad 6 \quad 5 \quad 4$

In the above example there are 13 possible match locations which must be checked over the entire input image. But if the metric of correlation used is m_2, the problem of finding c_{un} at each point i becomes a convolution with n = 16, k = 4, and where the unnormalized correlation at each location is found by:

$$c_{un_i} = w_1x_i + w_2x_{i+1} + \ldots + w_kx_{i+k-1}, \quad i=1,2,\ldots,n+1-k \tag{12}$$

(See Manber, 1989).

Template matching is a popular pattern recognition technique, in part due to its computational simplicity. A second advantage of the use of templates is that it is not necessary that any *a priori* probabilities be known, in contrast to statistical pattern recognition. Care must be taken, however, in the selection of the templates to be used. The technique is best suited for small, simple patterns, although more complex patterns (such as handwritten characters or fingerprints) may be broken down into two or more simpler templates. A further drawback of template matching is that rotation is not addressed. In order to recognize a pattern which may appear at several rotations, it is necessary to have templates of the object at each one. This can significantly increase image scan time.

IV. WISE INTERMITTENT STREAM RECOGNITION AND DETECTION

A. PREVIOUS SYSTEM

The National Mapping Division of the United States Geological Survey, which produces digital cartographic data as part of the National Mapping Program (Clarke, p. 77), is currently involved in a venture which involves converting analog cartographic data into digital form. Prior to 1980, the USGS produced maps in analog format using paper, mylar and other hard copy. Since then, however, an effort has been made to build the National Digital Cartographic Data Base, a collection of the digitized data from all the previously produced analog maps. Such a database is useful not only for Federal purposes, but also for commercial GIS use. In addition, as mentioned in section II.A. above, digital data is much more quickly and efficiently revised than analog data. Analog data is taken from map separates--mylar sheets which contain all the objects (lines, characters, areas, etc.) appearing in one color for a given map. For instance, there are six map separates for a 1:24,000 scale map produced by the USGS, one for all elements occurring in blue (hydrography information), black (characters and buildings), red (roads and urban areas), brown (hypsography), green (ground cover), and purple (additions to the map since it was created). Because these separates were initially overlaid to produce the original paper map, they contain all of the information necessary to convert the map to digital form.

1. Data Collection. The USGS employs several methods to digitize the data for the National Digital Cartographic Data Base, one of which is a proprietary system called I/VEC (created by Intergraph Corporation) to create digital line graphs in Intergraph

design file format. This file format is a vector representation which allows elements (lines, shapes, cells) to be placed on different levels of the file. Placing elements on different levels facilitates grouping them into categories as all elements which belong to a particular category can be placed on the same level.

I/VEC utilizes the raster representation of the information contained in the map separate to create the map's vector representation. The raster representation is created by scanning a map separate at a resolution of 1000 points per inch. Each separate is scanned individually so that the information contained in a single raster image is limited to the features of the separate being scanned. The current project is concerned only with the objects which appear on the hydrography separate--flowing water, standing water, and wetlands.

2. I/VEC Results. The raster file generated from scanning is then used as input to I/VEC, which creates the vector file based on the information contained in the raster file. In addition, I/VEC attempts to classify each of the elements created in the vector file according to its size, shape, pattern, etc. For instance, all lines that I/VEC recognizes as solid from the hydrography separate are classified as shoreline or perennial streams. The other type of line found on a hydrography separate is the patterned line which represents intermittent water bodies and streams. It is depicted on a map by the repeated sequence of a dash and three dots (see Figure 1 on page 10). I/VEC uses an unidentified pattern recognition technique to differentiate between intermittent and perennial streams[2]. Those streams that it is able to recognize are stored on various

[2] I/VEC was shipped to the USGS in executable form only. As there was no documentation regarding the algorithms used for vectorization and pattern recognition, I/VEC was treated as a "black box". The only knowns were the input and output.

levels of the design file, the intermittent streams on one level, and the perennial streams on another. Any lines not classified by I/VEC are placed on a third level.

I/VEC's recognition of intermittent streams has proven to be only partly successful. After I/VEC completes line classification, a significant number of lines remain on the unclassified level. Hand classification of these lines is a slow and tedious process; one which would be well served by automation. An examination of the unclassified intermittent streams uncovers some clues as to the reasons for classification failure:

a. Dot Location. If the dotted area of the intermittent stream pattern falls along a sharply curved portion of the line, the vectorization component of I/VEC sometimes fails to connect all three dots (see Figure 3a.). Often, the two dashes on either side are connected through only one of the dots, and a separate line is vectorized connecting the other two dots. In this case, the pattern of dots is not preserved and therefore I/VEC does not recognize the line as an intermittent stream. Such lines are placed on the unclassified level.

b. Missed Dots. I/VEC is (justifiably) quite strict in ensuring that the desired pattern (a dash, three dots, ...) exists with no deviation along the entire length of the line. A problem occurs, however, when dots are not recognized as such. One of the user-defined parameters used as input into I/VEC defines the size below which a group of pixels in the raster image is classified as noise. This is necessary because many of the map separates used in scanning are old and dirty. Scanning such separates at low resolutions produces speckles in the raster image which are not part of the original map, and therefore should not be vectorized. I/VEC checks the size of all raster elements before vectorization, and any which fall below the threshold are classified as noise and left out of the vectorization process. Dots which are actually part of an intermittent stream, but which are either too small or not the correct shape, are classified as noise. This means that they are not vectorized as part of a line, and therefore the pattern of the line will be disrupted. When this is the case, I/VEC will not classify the line, although the correct pattern exists along other areas of the same stream (see Figure 3b.).

c. Pattern Not Long Enough. Some lines are not classified by I/VEC though none of the above cases hold. Such lines are fairly short, most with only one full occurrence of the desired pattern (see Figure 3c.). It is therefore speculated that I/VEC requires two or more instances of the dash, three dot pattern in order to classify a line as an intermittent stream.

d. Gaps Between Dots. If the space between two dots, or a dot and a dash, is too large, I/VEC will fail to vectorize the area with an unbroken line (see Figure 3d.). The current line will end at the dash or dot on one end of the large gap, and a new line will begin at the dash or dot on the other end of the gap. One of the user-definable parameters I/VEC uses is a distance tolerance, beyond which the program will end the current line and begin a new line. While this is not a problem addressed by the current project as it does not necessarily inhibit line classification, it is one which occurs in I/VEC and which must be corrected by hand.

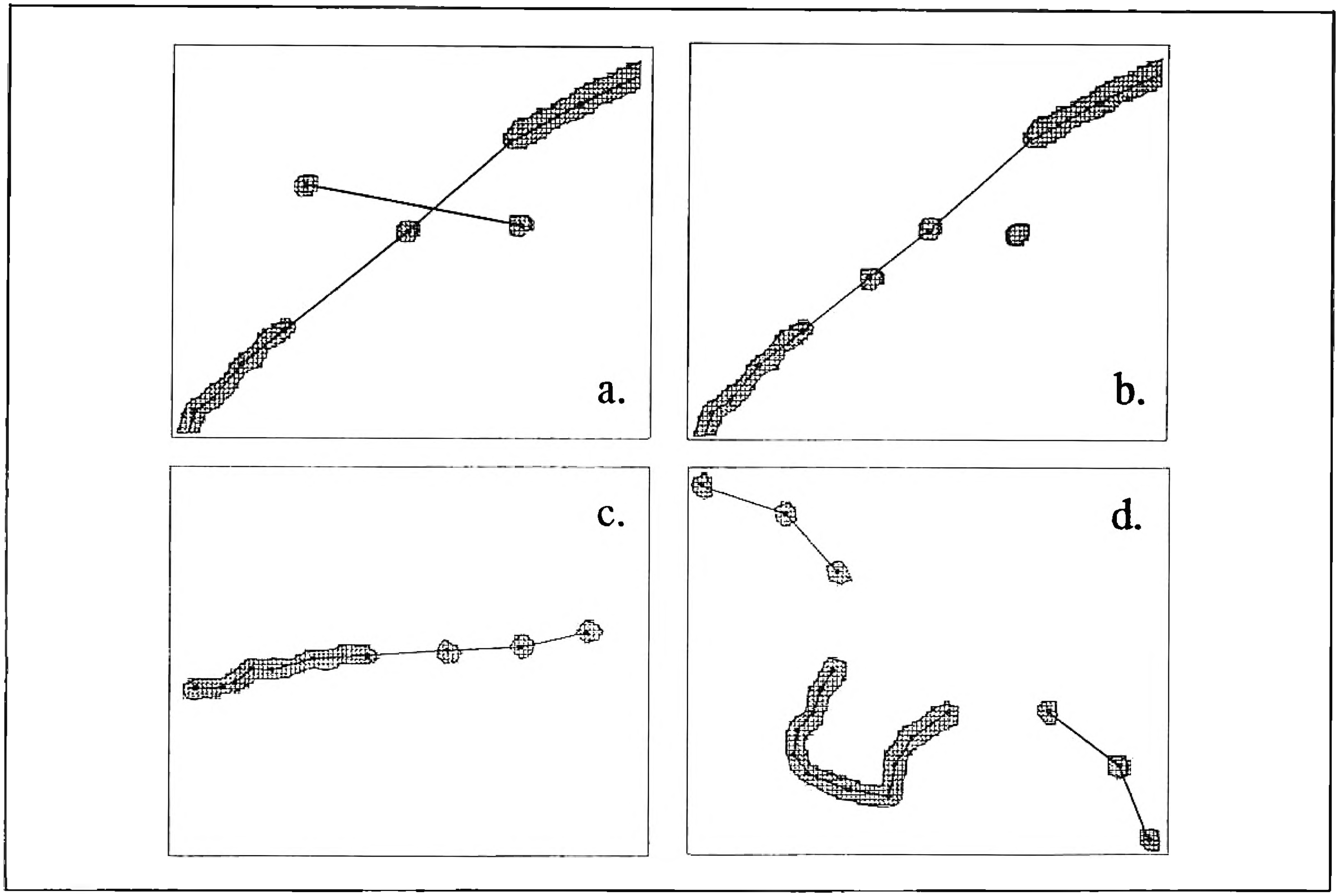
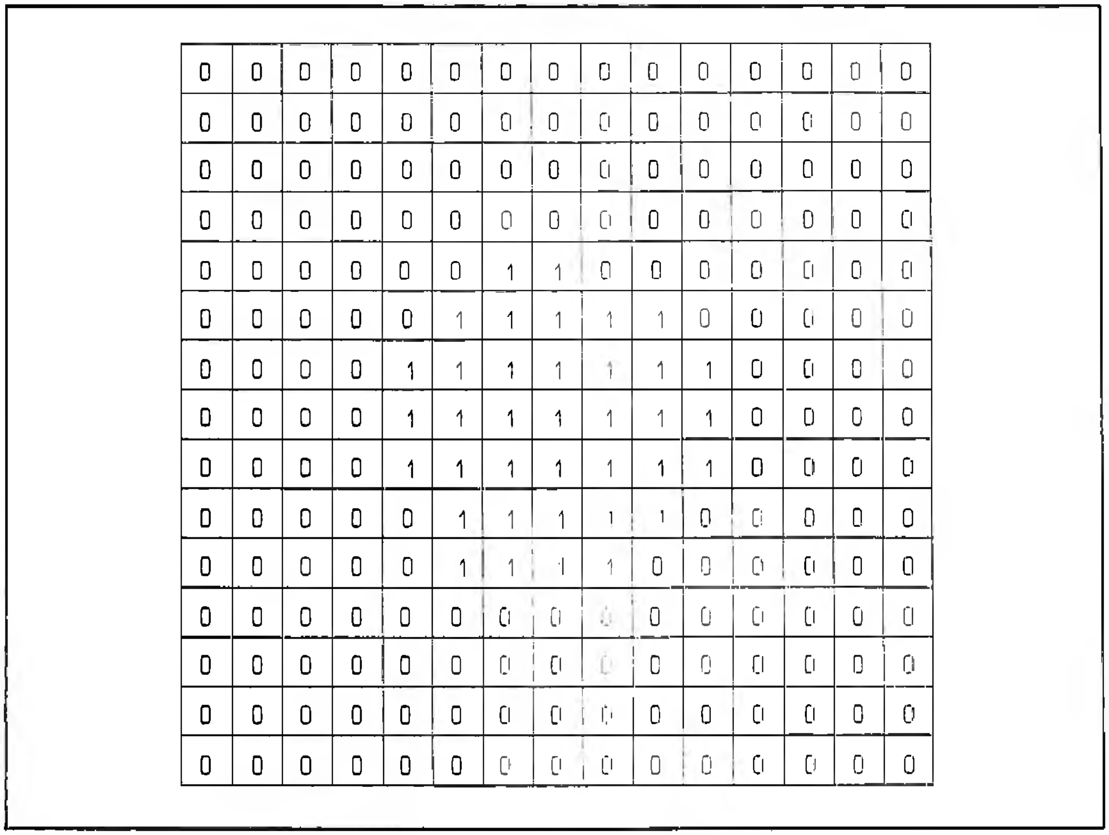


Figure 3. Situations in which I/VEC classification fails.

Although the above vectorization and classification problems occur with the use of I/VEC, there are areas in which I/VEC's performance is quite effective. With the exception of subsections a., b., and d. above, I/VEC's vectorization is very successful. Most vectors coincide perfectly with the raster lines, and dots are always vectorized with a point through the geometric center. In addition, due to I/VEC's strict classification rules (see subsections b. and c. above), any classification the program is able to make is made with great accuracy. Among the files used as test files in this project, any line classified by I/VEC as either a perennial or intermittent stream proved to be such upon examination of the raster file by the researcher (*i.e.*, there were no false positives in either category).

B. WISRD

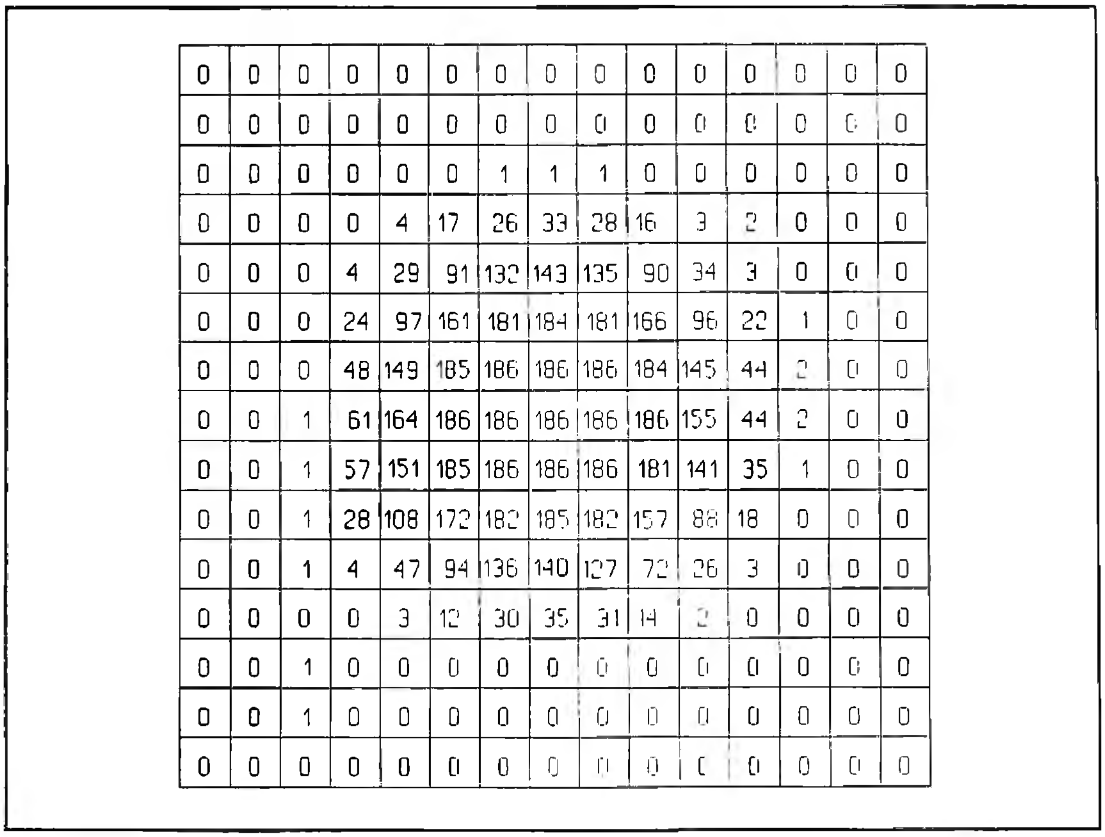
The current project was undertaken in order to decrease the amount of manual editing necessary following vectorization and classification by I/VEC. As CPU time at the USGS is cheaper than operator time, it is advantageous to produce a system which is able to classify lines in batch mode to be run at any time, requiring little or no operator interaction. Inherent to the Wise Intermittent Stream Recognition and Detection (WISRD) system is the desire to preserve the successful results produced from I/VEC (see section IV.A.2. above) while at the same time minimizing the number of unclassified lines at the completion of the system. In addition, the desire to avoid duplicating any processing already performed (to some degree of success) by I/VEC is also a driving strategy of the current project.



0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
0	0	0	0	0	0	1	1	0	0	0	0	0	0	0
0	0	0	0	0	1	1	1	1	1	0	0	0	0	0
0	0	0	0	1	1	1	1	1	1	1	0	0	0	0
0	0	0	0	1	1	1	1	1	1	1	0	0	0	0
0	0	0	0	1	1	1	1	1	1	1	0	0	0	0
0	0	0	0	0	1	1	1	1	1	0	0	0	0	0
0	0	0	0	0	1	1	1	1	0	0	0	0	0	0
0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Figure 4. Raster representation of a single dot.

1. Desired Improvements to I/VEC. The scope of the current project has been defined with the previous issues in mind. In order to decrease operator editing time, it is necessary to increase the number of classified lines in a vector file while not sacrificing classification accuracy. An increase in classified lines without a simultaneous decrease in accuracy would mean that fewer hours would be needed to classify unclassified lines, and that the entire file would not have to be scanned by hand to make sure all classifications are correct. In order to fully utilize the beneficial effects produced by I/VEC, WISRD has been designed to use as input the vector file created by I/VEC and output the same file, with modifications of line classifications.

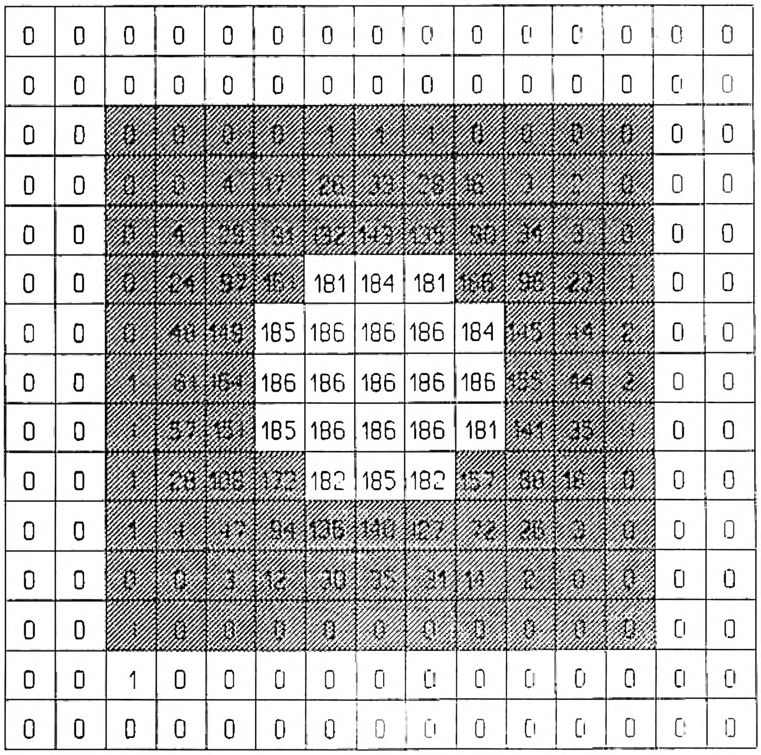


0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
0	0	0	0	0	0	1	1	1	0	0	0	0	0	0
0	0	0	0	4	17	26	33	28	16	3	2	0	0	0
0	0	0	4	29	91	132	143	135	90	34	3	0	0	0
0	0	0	24	97	161	181	184	181	166	96	22	1	0	0
0	0	0	48	149	185	186	186	186	184	145	44	2	0	0
0	0	1	61	164	186	186	186	186	186	155	44	2	0	0
0	0	1	57	151	185	186	186	186	181	141	35	1	0	0
0	0	1	28	108	172	182	185	182	157	88	18	0	0	0
0	0	1	4	47	94	136	140	127	72	26	3	0	0	0
0	0	0	0	3	12	30	35	31	14	2	0	0	0	0
0	0	1	0	0	0	0	0	0	0	0	0	0	0	0
0	0	1	0	0	0	0	0	0	0	0	0	0	0	0
0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Figure 5. Combination grid of 186 dots.

2. Template Matching. The pattern recognition technique chosen for the current project is template matching. Statistical pattern recognition is not feasible, as the probabilities necessary for its use were not *a priori* available. In addition, there are aspects of this particular project that make template matching the best pattern recognition technique based on both its success and simplicity. The success rate using template matching is so high that the additional complexity necessary for a neural network approach is not justified.

Since the purpose of the current project is to distinguish between perennial streams, denoted by a solid line, and intermittent streams, denoted by a patterned line of a dash and three dots, a search for dots along an unclassified line is the first approach.



0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
0	0	0	0	0	0	1	1	1	0	0	0	0	0	0
0	0	0	0	4	17	26	33	28	16	7	2	0	0	0
0	0	0	4	29	81	132	143	135	90	39	3	0	0	0
0	0	0	24	92	161	181	184	181	156	90	22	1	0	0
0	0	0	40	149	185	186	186	186	184	145	49	2	0	0
0	0	1	81	164	186	186	186	186	186	155	44	2	0	0
0	0	1	57	151	185	186	186	186	181	141	35	1	0	0
0	0	1	28	108	172	182	185	182	157	80	16	0	0	0
0	0	1	4	47	94	136	140	127	72	26	3	0	0	0
0	0	0	0	3	12	30	35	31	14	2	0	0	0	0
0	0	1	0	0	0	0	0	0	0	0	0	0	0	0
0	0	1	0	0	0	0	0	0	0	0	0	0	0	0
0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Figure 6. Combination grid with mask.

The choice to search for dots has proven to be a major advantage in template matching, as no compensation is necessary for rotational effects.

In order to perform template matching, the template must be compared to various areas in the raster file. In general template matching (as mentioned above in section III.B.), shifting of the template over the entire image is usually required in order to find the location(s) of the desired pattern. In this application, however, certain information stored in the vector file allows substantial trimming of the search space (the range of the raster file). In most cases, dots in the raster file can be found by centering the dot template on those points which correspond to the vectorized points stored in the vector file. Because of the vectorization accomplished by I/VEC, one is almost guaranteed that

0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
0	0	▨	▨	▨	▨	▨	▨	▨	▨	▨	▨	▨	0	0
0	0	▨	▨	▨	▨	▨	▨	▨	▨	▨	▨	▨	0	0
0	0	▨	▨	▨	▨	▨	▨	▨	▨	▨	▨	▨	0	0
0	0	▨	▨	▨	▨	1	1	1	▨	▨	▨	▨	0	0
0	0	▨	▨	▨	1	1	1	1	1	▨	▨	▨	0	0
0	0	▨	▨	▨	1	1	1	1	1	▨	▨	▨	0	0
0	0	▨	▨	▨	1	1	1	1	1	▨	▨	▨	0	0
0	0	▨	▨	▨	▨	1	1	1	▨	▨	▨	▨	0	0
0	0	▨	▨	▨	▨	▨	▨	▨	▨	▨	▨	▨	0	0
0	0	▨	▨	▨	▨	▨	▨	▨	▨	▨	▨	▨	0	0
0	0	▨	▨	▨	▨	▨	▨	▨	▨	▨	▨	▨	0	0
0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Figure 7. Binary template used. Shaded areas are not considered in correlation calculation.

the points stored as part of a vector line fall along the center of the corresponding line in the raster file when the transformation between coordinate systems is made. This process is further aided by the fact that I/VEC vectorizes dots with a single point through the geometric center of the dot, thus if a dot exists at a given point, that point is the center of the dot. The transformation coefficients necessary to produce raster coordinates from vector coordinates (or *vice versa*) are found in a file created by I/VEC. This transformation usually consists of a translation and scaling.

As mentioned above, the success of any template matching application is heavily dependent upon the selection of the template. In order to define the template for the current project, the shapes of 186 known dots were compared. A 15-pixel by 15-pixel grid around each dot produces a picture like Figure 4. Combining each grid for all 186 dots (so that if a pixel in a particular location was "on" in every grid, it would have a value of 186 in the combination grid) produced Figure 5. Higher numbers in the center indicate locations with a higher probability of being "on", while lower numbers on the edges of the combined dots indicate some blurring of the edges. Because of this blurring, a masked template was used so as to consider only those pixels which have a high probability of being either on or off, as shown in Figure 6. The template is masked because the shaded area is not considered in the calculation of the correlation. Figure 7. shows the resultant template.

3. Pass 1. Pattern recognition in WISRD is accomplished in a 2-pass approach. During the first pass, the dot template is placed in the raster file location corresponding to each point stored along an unclassified line in the vector file. The correlation between the template and the 15x15 pixel area covered by it at each point in the raster file is calculated, and if it is above the threshold percentage, the point is recognized as a dot. Once a dot has been found along a line, the number of consecutive dots which follow is recorded. If there are fewer than three consecutive dots along any area of the line other than the beginning or the end, that line is stored in a problem file for use during pass 2. If, on the other hand, three consecutive dots are found at each dotted area along the line, the line is classified during the first pass as an intermittent stream. Conversely, if no dots are found along the entire length of the line, the line is classified as a perennial stream. Classification of a line is accomplished by moving the line from the unclassified

level in the vector design file to the level designated to hold lines of the determined type (either perennial or intermittent streams).

During the first pass, each line stored on the unclassified level of the vector design file is examined. For every unclassified line, then, there are three possible outcomes: it may be positively identified as an intermittent stream; it may be positively identified as a perennial stream; or it may be classified as having one or more problem areas and stored in a problem file for later examination during the second pass. Problem areas along a line (locations where there are one or two consecutive dots, but not three) can arise for a few different reasons. Two of these reasons stem from improper vectorization. If the vector line through a dotted area only connects one or two of the dots, as described in subsection IV.A.2.a. above, that area will be flagged as a problem area as three consecutive dots were not found along the vectorized line. The same effect will be produced if one or more of the dots in an area are classified as noise prior to vectorization, as described in subsection IV.A.2.b. Again, any dot not recognized as such is not vectorized as part of the line, so it will not be found when the points which lie along the line are searched for dots during pass 1 and will therefore be recorded as a problem area. The third reason that a problem area may be found along a line reflects an inherent weakness of template matching. If a point along a vectorized line is actually a dot, but the raster image of that dot is somewhat warped or misshapen, the point may not be recognized as a dot during pass 1. When any of these situations occur, the area is flagged as a problem area and then re-examined during pass 2.

There is assuredly a tradeoff when setting the threshold value, a correlation above which indicates the presence of a dot. A high threshold value will tend to reduce false positives (points which are not dots, but which are recognized as dots), but there will be a subsequent increase in false alarms (points which are dots, but which are not recognized as such). It is therefore necessary to find an optimal threshold value, one which minimizes both false alarms and false positives. For the current project, a threshold value of 95 (95% match to the template) was chosen by trial and error.

4. Pass 2. During the second pass of WISRD, those lines which contain problem areas are examined more closely in order to find some of the dots which were missed in the original vectorization. The line problem file is used to pinpoint the areas in which the vectorized line is missing dots. These areas are searched for all noise elements within a given range. If noise elements are found in the vector file, the location of each is checked with the dot template in the raster file (see Figure 7). The correlation to the template is computed at each noise location and the locations are sorted in descending order of correlation value. Thus, when the number of noise elements found exceeds the number of dots needed, the elements most closely resembling a dot are used first. In order for a noise element to be recognized as a dot, and thus inserted into the line, the correlation percentage must be greater than the threshold of 95 (as in section IV.B.3. above).

If the number of noise elements recognized as dots meets (or exceeds) the number of dots needed to complete the pattern of the line, the necessary number of dots is added, each as a single point in the line, and consequently included in the vectorized representation. Care must be taken, however, to connect the dots in the most natural manner. Since a stream occurring in nature never crosses over itself, the line in the vector file which represents a stream must also never cross over itself. In order to ensure this to be the case, a simple algorithm is used which finds the shortest path between the five points in question: one point for each of the three dots which must be joined, as well as the last point on the preceding dash and the first point on the following dash. Since the two points on the dashes have fixed locations (one at the beginning, one at the end), only the order of the three dots is in question. The algorithm determines the total distance for each permutation of dot order and selects the path which is shortest[3]. Such an algorithm would not be efficient (or even feasible) with longer paths, but for paths with a length of five, two of whose points are fixed, it is an effective technique.

On the other hand, if the number of noise elements is less than the number of dots needed, the line will remain unclassified. Only after a problem area has been detected in pass 1, and sufficient dots failed to be recognized in pass 2, will a line fail to be classified in the WISRD system.

[3] It can be shown using the triangle inequality theorem that the shortest path between five points, two of which are fixed, will not cross itself.

V. RESULTS

A. TEMPLATE RESULTS

The use of the dot template proved to be a successful method of recognizing whether or not a vectorized point was a dot. Figure 8 illustrates the point-by-point correlation calculation for all points along a classified intermittent stream versus all the points along a classified perennial stream. Any point which gives higher than a 95% correlation to the template is classified as a dot. Such a marked difference in the correlations of dots and non-dots allowed for fairly accurate classification of intermittent streams. The ability to find the location of all dots along a line affords high confidence in line classification.

B. WISRD SYSTEM RESULTS

WISRD was tested on a total of ten digital maps produced at the Mid-Continent Mapping Center of the U. S. Geological Survey in Rolla, Missouri. These maps were created in the manner described in section IV.A.1. above. Each was scanned from a mylar separate to create the raster file, which was then used as input to I/VEC to create the vector file. The raster and vector files for each map were then used as input to WISRD, which performed the additional stream classification on those streams left unclassified by I/VEC. The ten files on which WISRD was tested had a combined total of 2393 unclassified lines following processing by I/VEC. Of these 2393 originally unclassified lines, 2165 (90.47%) were classified by WISRD during pass 1 (as either perennial or intermittent streams). An additional 106 lines (4.43%) were restored and

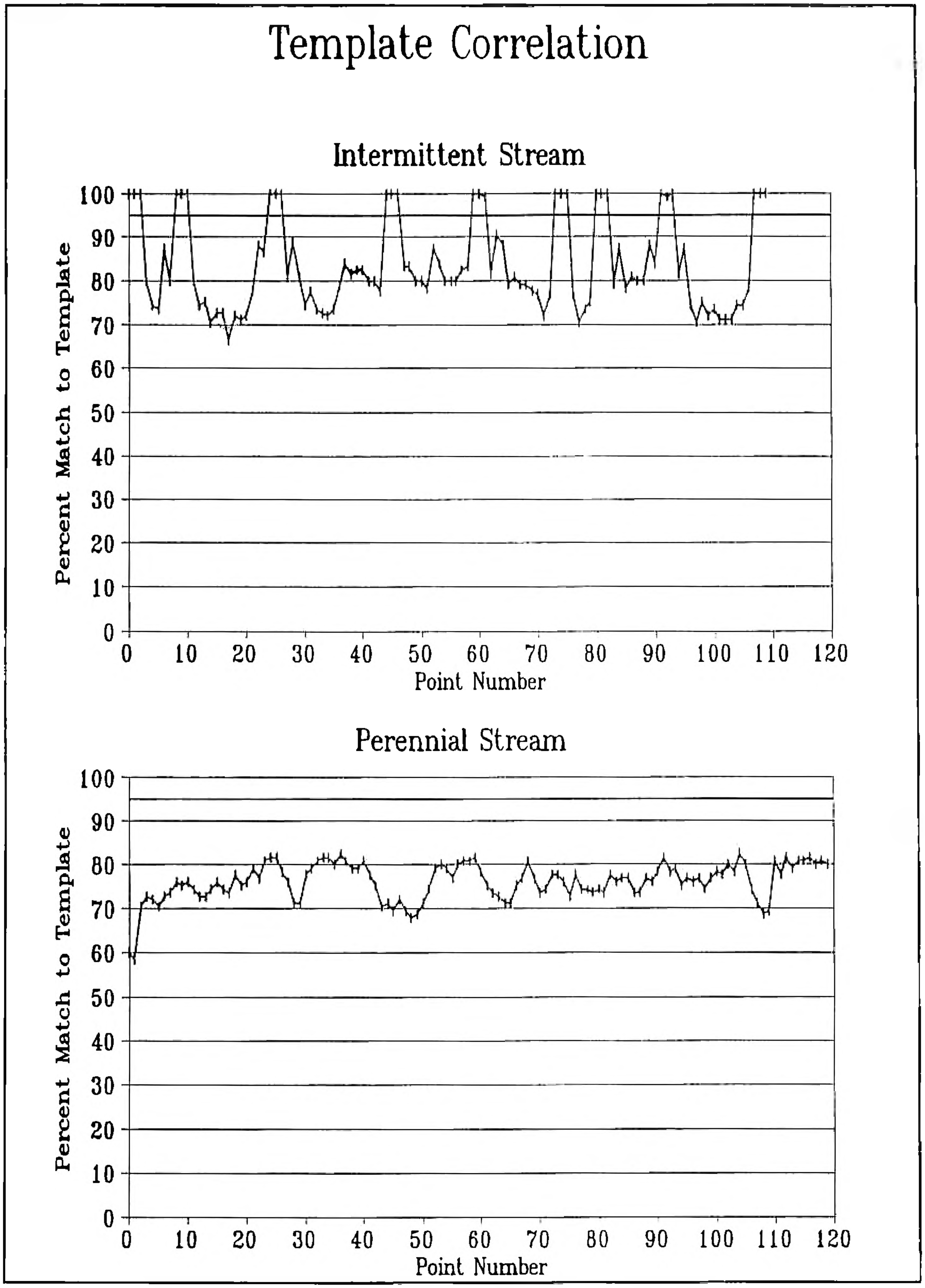


Figure 8. Point-by-point correlation to dot template for a classified intermittent stream and a classified perennial stream.

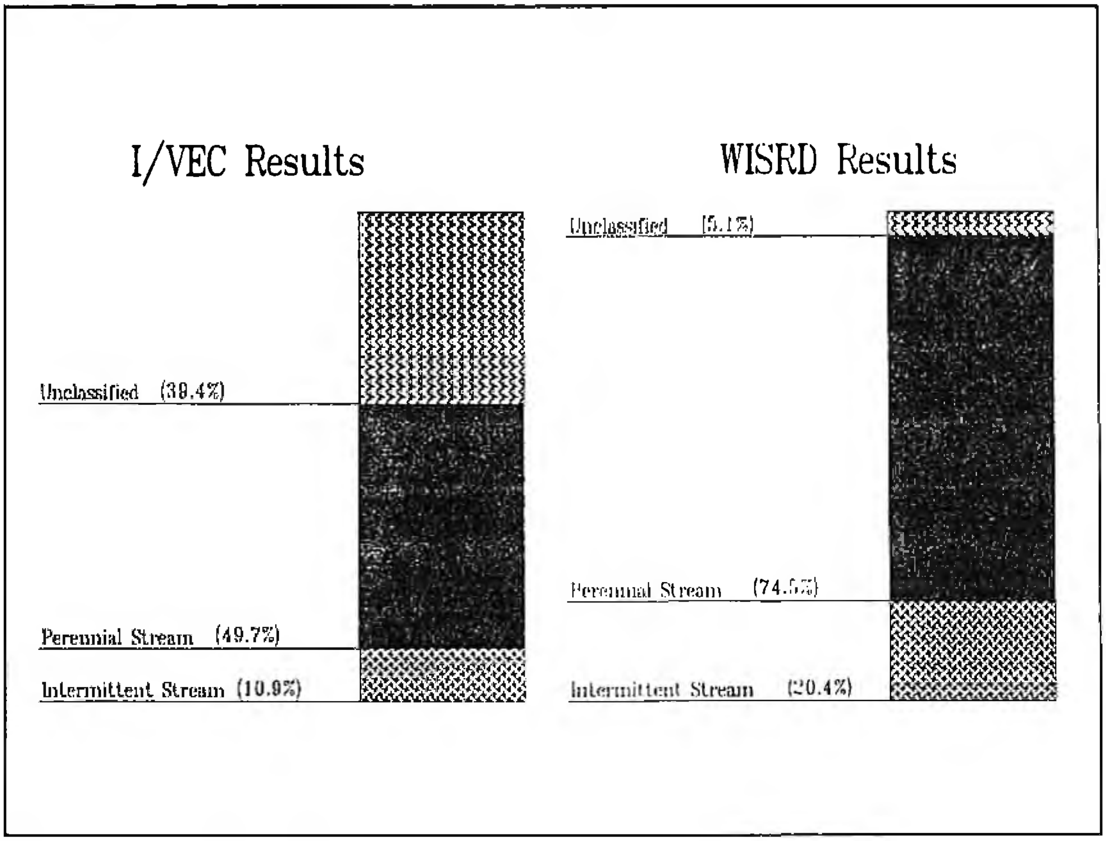


Figure 9. Comparison of the line classification percentages of I/VEC and WISRD.

classified during pass 2 as intermittent streams. This left 122 lines (5.10%) unclassified after the completion of WISRD. This compares to 39.39% of lines left unclassified after I/VEC had completed (see Figure 9).

C. WISRD CLASSIFICATION FAILURES

Among the ten files on which the system was tested, classification percentages for WISRD ranged from 80.80% to 99.23% (see Figure 10), compared to I/VEC, in which classification percentages ranged from 42.50% to 79.40% (see Figure 11). This variation in classification rates indicates the extent to which the success of WISRD

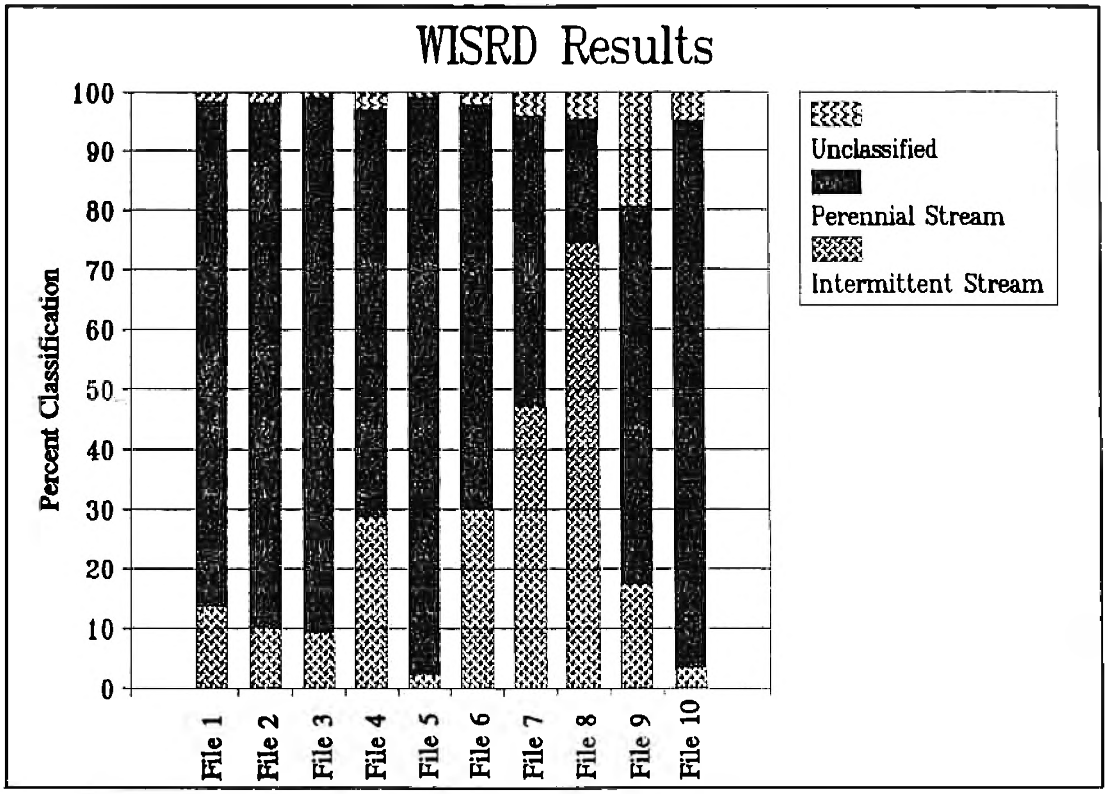


Figure 10. File-by-file comparison of WISRD line classification percentages.

depends on the quality of the map separate scanned, as well as the accuracy of vectorization. There are specific, predictable instances in which WISRD will fail to classify a line. The first of these instances occurs when the pattern of an intermittent stream is not faithfully followed in the original map separate. Because the separate was created manually, it occasionally happened that only two dots were placed between two dashes along an intermittent stream. This causes a problem in WISRD because the third dot will neither be present in the raster file, nor the vector file. Luckily this is not a prevalent problem, as it is a rare occurrence for a dot to be missing from the map separate.

A second instance in which WISRD will fail to classify a line occurs when the scanning of the map separate produces a distorted raster image. The template used to

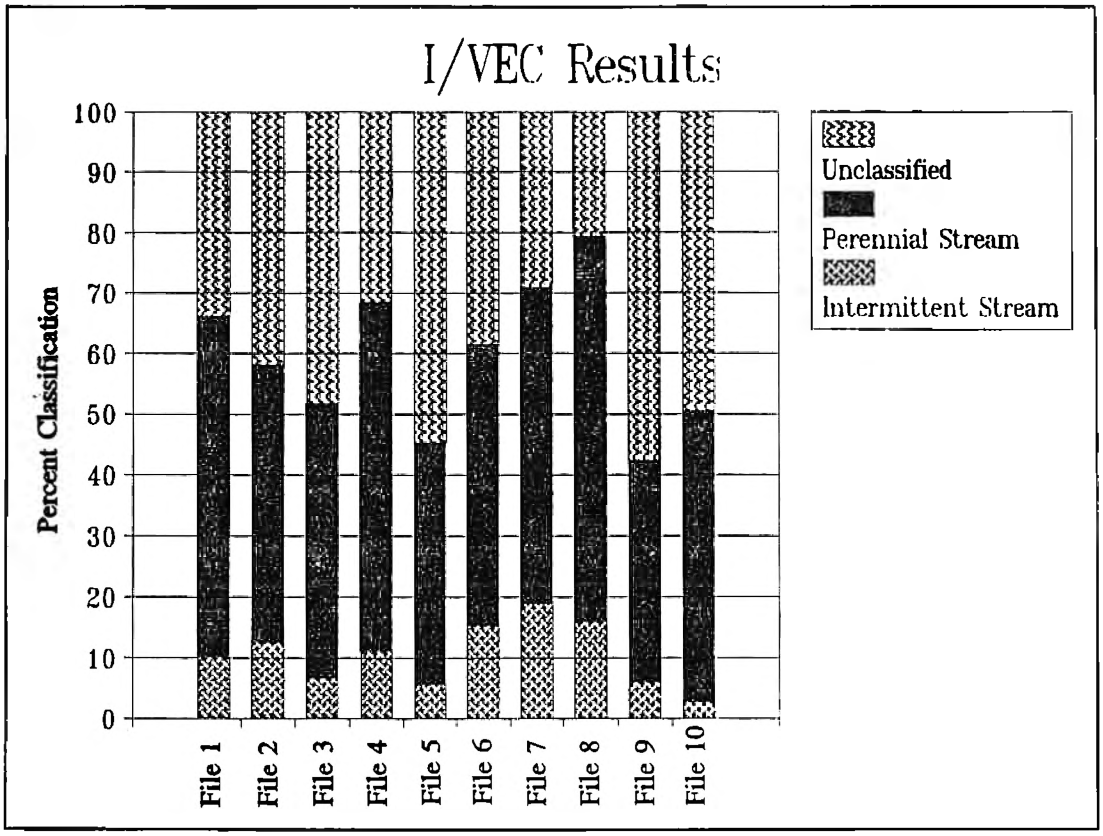


Figure 11. File-by-file comparison of I/VEC line classification percentages.

recognize dots has 125 points of correlation. With a correlation threshold of 95%, this means that a dot can vary from the template by up to six pixels. (Dots are generally eight to ten pixels in diameter, where one pixel = 1/1000 inch.) If a dot is misshapen or distorted enough to differ from the template by more than six pixels, its correlation will fall below the threshold, and it will not be recognized as a dot.

In addition to separate and scanning accuracy, WISRD is also dependent on accurate vectorization by I/VEC. The ability of WISRD to restore the proper pattern to a line by including dots which I/VEC originally classified as noise has already been discussed in section IV.B.4. above. But a problem which WISRD is currently unable to correct occurs when two of the dots in an area are vectorized with a single line which begins at one dot and ends at the other, as in Figure 3a. on page 27. In this situation, the third dot in the area is usually connected to the two dashes, one on either side. Such a problem may occur when the dotted area falls along a sharp curve, making accurate vectorization difficult. Because the line through the dashes only passes through one dot, it cannot be classified by WISRD. In addition, because of the need to correctly classify lines which begin or end with one or two dots, the line that connects the two dots will be classified as an intermittent stream.

The fourth instance in which WISRD will fail to correctly classify a line occurs because parts of a single intermittent stream are sometimes vectorized individually, due to the gaps between dashes and dots being beyond the gap-length threshold used in I/VEC, as described in subsection IV.A.2.d. above. When it happens that a single dash is vectorized individually (with no dots attached), WISRD will classify the dash as a perennial stream because no dots are found along its length (see Figure 3d. on page 27).

Of all the above failures by WISRD to correctly classify a stream, the latter is the only situation in which part of a stream is *incorrectly* classified, rather than being left unclassified. It is therefore the only problem for which the entire file must be scanned.

Any lines which cannot be classified can be flagged with circles or ellipses around each problem area. This step increases editing efficiency, as the user will be led directly to areas which require his or her attention rather than performing an exhaustive search of the file for such areas.

D. RUN TIME

It is expected that the use of WISRD following vectorization and classification by I/VEC, and before manual editing, will significantly reduce the amount of time necessary to completely edit hydrography files at the USGS. Prior to the development of WISRD, 16 to 24 hours of operator time were required to ensure that all streams had been correctly classified, and to clean up any vectorization errors. The WISRD system, which consists of passes 1 and 2, will require 30 to 45 minutes of run time on an average file (the average number of unclassified lines in each of the ten test files was 239). Following processing by WISRD, it is expected that the use of WISRD will significantly reduce the amount of time necessary to edit the file. This expected time savings is based on the fact that the user will be able to proceed directly to most of the problem areas. While it is true that it will still be necessary to scan the entire file for any occurrences of a single dash being classified as a perennial stream (as described in section V.C. above), the fact that perennial streams are displayed in the vector file in a different color than intermittent streams should aid the user in finding these situations quickly.

VI. CONCLUSION

A. FUTURE WORK

The current project has addressed only one aspect of the problems faced when creating digital maps in the manner described. As noted above, there are a few situations in which line classification will always fail. A full-scale automated map classification system must successfully deal with each situation in order to minimize manual editing and therefore maximize efficiency. Toward this end, a useful enhancement would be to shift the template around a given area in the raster file once it is determined that fewer than three dots are found along a vector line and no noise in that same area can be found. This process could have one of two effects: if the missing dot or dots do not exist in the raster file, the system would discover this fact and could take steps to remedy the situation. If, on the other hand, the missing dot(s) do exist in the raster file, but have been erroneously assigned to another line, the system could take steps aimed at correcting this error. For this search, a different, less precise template (or lower correlation threshold) could be used in order to find misshapen dots as well.

A further enhancement would be to join broken segments of a single stream into a continuous line once all lines are classified. To accomplish this, the areas around the endpoints would be searched for all lines of the same type whose endpoints are within some distance threshold from the point in question. Some rule would need to be invoked in the event that two or more such lines were found. To address a related problem, all perennial streams below a certain length could be examined to see if any intermittent streams occur within a close proximity. If so, the area should be further examined in order to determine whether or not a dash of an intermittent stream had been vectorized as a separate line, causing it to be classified incorrectly.

Finally, it would probably be beneficial to implement adaptive behavior in the system. This could require creating the dot template at run time from the set of lines classified as intermittent streams by I/VEC. Also useful would be the ability to classify lines with various patterns, rather than just the single pattern classified with the current system. This could be accomplished with a similar template matching strategy, but without many of the assumptions made in the current system.

B. CONCLUDING REMARKS

Automated map analysis and classification is an application of computers which facilitates increased efficiency of operator time. As more of this task is automated, the number of manhours needed for analysis and editing may decrease, but only when the system used meets or exceeds the degree of accuracy provided by manual analysis. A successful system must therefore mimic the behavior of a human. One way to mimic human behavior is to create a computer system with built-in domain knowledge. Cartography is a good application of computer systems with built-in knowledge (wise systems) because most of its components are well-defined.

In creating digital maps, there are two ways of representing data: raster format and vector format. Because each of these ways has its own advantages and disadvantages, both are used in the creation and maintenance of digital cartographic data, often to represent the same data in different stages of the production process. It is therefore important to ensure that no desired information is lost in the conversion from one format to the other. This project has specifically addressed the conversion of hydrographic data from raster to vector format. In order to conserve stream classification information, it is necessary that the conversion system be able to differentiate between perennial and intermittent streams based on the line patterns which appear in the original map. The USGS uses a proprietary system, I/VEC, which creates a vector file from the raster file of the map, and then makes an attempt to classify the vector lines. The success of this system, however, has been limited--with up to three days of manual editing necessary to classify those lines which I/VEC missed.

The goal of the current project was to improve upon the classification rates produced by I/VEC while maintaining the same degree of accuracy. The strategy was therefore developed whereby the vector file output by I/VEC as well as the raster file would be used in an attempt to classify the lines which I/VEC could not. Template matching was chosen as the pattern recognition technique because of it simplicity and success in recognizing the dots which are characteristic of the pattern which represents intermittent streams. This proved to be an extremely effective technique, as the shape of a dot is not affected by rotation--one weakness of template matching--and I/VEC vectorizes dots with a point in the geometric center--eliminating the need to shift the template around the raster image.

Classification rates for WISRD ranged from 80.80% to 99.23%, and averaged 94.9% over the ten test files. These figures compare favorably to an average classification rate of 60.61% by I/VEC over the same ten files. It is therefore concluded that WISRD is an effective tool in decreasing the amount of manual editing necessary for hydrography files. The editing necessary is further decreased by the fact that WISRD is able to flag all problem areas for the user, making an exhaustive search for such areas over the entire file unnecessary.

WISRD should prove to be an effective tool in map analysis. The built-in domain knowledge it contains allows it to run efficiently and accurately in the sphere for which it was created. But while further modifications would be necessary to produce a more general or adaptive system, the underlying strategies of WISRD should provide sufficient basis for the necessary alterations. It is expected that the extent of any such modifications would be limited to the addition of knowledge or the relaxing of rules.

BIBLIOGRAPHY

Amin, Tushar J. and Rangachar Kasturi. "Map Data Processing: Recognition of Lines and Symbols". Optical Engineering 26, no. 4 (April 1987): 354-358.

Clarke, Keith C. Analytical and Computer Cartography. Englewood Cliffs, New Jersey: Prentice-Hall, Inc., 1990.

Eysenck, Michael W., et al., eds. 1990. The Blackwell Dictionary of Cognitive Psychology. Cambridge: Basil Blackwell, Inc. S.v. "Pattern perception".

Manber, Udi. Introduction to Algorithms: A Creative Approach. Reading, Massachusetts: Addison-Wesley Publishing Company, Inc., 1989.

Mishkoff, Henry C. Understanding Artificial Intelligence. Indianapolis: Howard W. Sams & Company, 1985.

Pavlidis, Theo. Algorithms for Graphics and Image Processing. Rockville, MD: Computer Science Press, Inc., 1982.

Pequet, Donna J. "An Examination of Techniques for Reformatting Digital Cartographic Data/Part 1: The Raster-to-Vector Process." Cartographica 18, no. 1 (1981): 34-48.

Schalkoff, Robert. Pattern Recognition: Statistical, Structural, and Neural Approaches. New York: John Wiley & Sons, Inc., 1992.

Star, Jeffrey and John Estes. Geographic Information Systems: An Introduction. Englewood Cliffs, New Jersey: Prentice-Hall, Inc., 1990.

Tanimoto, Steven L. The Elements of Artificial Intelligence Using Common LISP. New York: Computer Science Press, 1990.

Tou, Julius T. and Raphael C. Gonzalez. Pattern Recognition Principles. Reading, Massachusetts: Addison-Wesley Publishing Company, Inc., 1981.