Hue Histograms for Raster-to-Vector Conversion of Color Documents

James D. Beatty
Spatial technologies & Concepts Division
Science Applications International Corporation
McLean, Virginia 22102
jbeatty@gis.saic.com

Kenneth J. Hintz
Department of Electrical and Computer Engineering
George Mason University
Fairfax, Virginia, 22030
khintz@gmu.edu

ABSTRACT

This paper introduces and demonstrates a new approach to raster-to-vector conversion of color maps by identifying individual lines and features within the map based on the hue of the digitized pixels. This paper further demonstrates the effectiveness of using the HSI (hue, saturation, and intensity), as opposed to the more common RGB (red, green, and blue), color representation combined with a hue histogram for identifying different lines within a color cartographic map image. In addition to its effectiveness, this semi-automatic technique is able to successfully extract these lines from the image with significantly less processing and human interaction than the currently available methods and software. The application of this research significantly improves upon the standard methods of raster-to-vector conversion since it can simultaneously identify image components as well as significantly reduce processing time and manual labor. With only minor manual enhancements, the output of this raster-to-vector conversion technique are readily usable by GIS systems.
1. Introduction

This paper deals with a significant advancement in the state-of-the-art of vectorizing rasterized (bit-mapped) color images of color cartographic maps. Since this methodology applies to any of a number of different types of color line drawing documents from CAD drawings to nautical charts to cartographic maps, the general term source documents or just documents will be used throughout the rest of the paper. The particular documents which are used here for demonstration purposes are color charts. The term rasterized images comes from the method of converting source documents to digital form on a rotating drum scanner. The term is common in the literature and will be used throughout this paper interchangeably with another common term, bit-mapped, as opposed to vector, images. Raster images, while in digital form as pixels, are typically less informative and less useful than their vector-image counterparts. In addition to the more compact representation, the major advantages of vector representation are the ability to associate database records with the vector information, as well as the ability to perform spatial, thematic, and complex queries on the database which result in an image of the appropriate item. Additionally, for maps and other geographic information, vectors and vector images can be registered to particular latitudes and longitudes and are easily scaled while maintaining the resolution of the display system to the limit of the original document. This makes an electronic map far easier to manipulate when zooming or extracting a region.

With the advent of faster and smaller computers and increasingly less expensive mass storage, the geographic information systems (GIS) and cartographic communities have been continually moving towards the storage of data, which has been previously represented only on paper maps, in digital form without losing content or fidelity \[1,2\]. A common approach for converting legacy paper source documents into digital form utilizes a scanner to digitize the image, conversion software which converts the raster (bit-map) image to a more compact form, typically as a set of vectors, accompanied by extensive manual pre-processing of the digitized image and post-processing of the vectorized digital image to remove artifacts \[3\]. An alternative approach utilizes expensive hand-digitizing hardware in which a person manually traces over the lines with software tailored to a particular production application converts the paper map into digital vector form. Since this is a tedious and predominately manual process, the density of the image on the paper source directly relates to the amount of manual-digitizing and post-processing.

Normally, the scanning of the source document, which converts it to a raster (bit-mapped) image, is followed by extensive pre-vectorization processing. Pre-processing involves manual and/or automatic modification of the bit-map image so that it can be vectorized with fewer errors which need to be corrected later during the post-vectorization phase. This digital signal processing typically involves reducing the number of colors to two (a binary-valued image), removing scanner-induced errors or noise, and removing any extraneous data that may corrupt the vectorizing process. Much of the processing is done manually to ensure accuracy. For these reasons, many people prefer to perform minimal pre-processing and do the cleanup, albeit significantly more of it, at the end, in the post-vectorization processing. This approach is fairly commonplace in commercial software.

The vectorizing process uses relatively simple algorithms, such as thinning and edge detection, to find contiguous lines within a bitmap image \[4\]. These lines may ultimately represent
alphanumeric letters from a text document, shapes and symbols from a CAD drawing, or rivers, roads, or contour lines from a chart or map.

Post-processing corrects the vectorized image to accurately depict the source document. The correction involves removing errors and noise introduced during the digitizing and vectorizing steps. This may include using algorithms that search for and correct commonly occurring or specific anomalies, manual vector editing, line tracing, and manually comparing the vector image to the original raster. Even though image restoration techniques may not apply directly to vector images, this and other processes have proven to be beneficial for post-processing.

Errors inherent in the raster-to-vector processing of source documents and the associated time required to correct them after the fact can be significantly reduced by performing statistical analysis of the raster-scanned image before vectorization. This pre-processing consists of using a hue histogram and sequentially partitioning the image into several mutually exclusive overlays using the various histogram values. This approach, which has not been reported in the literature, nor is available in commercial software, reduces the amount of human interaction required and thus saves time and effort as well as automatically differentiates among the features making it easy for a human operator to identify the feature by type. Through this processing, post-processing is reduced to random noise removal and line tracing. Post-processing which remains after the application of this technique will not be eliminated but is greatly reduced. Post-processing of the vectorized image will not be covered here since it is beyond the scope of this research.

Similar processing techniques using RGB (red, green, and blue) representation have been attempted and are in use in a variety of specialized raster-to-vector programs with varying degrees of success. The commercial applications of such techniques tend to be limited in scope and range of images that they can deal with. No process based on the color values of the image components can be applied in general to all color line drawings.

1.1 Advantages of Vector Images

Particularly in GIS systems, the difference between bit-mapped and vector images is significant. The requirement for maintaining databases of information for each feature or individual object in an image is well known. Data size and structure are important storage issues. For example, assuming Figure 1 is a typical raster image, a bitmap image file (using the common .BMP extension) quantized to 8-bits of intensity is at least the number of rows times the number of columns, 300x300 pixels, or over 90,000 bytes. (The original image, Figure 8, is over 4,760,000 bytes at 1700 x 2800 pixels and 256 colors.) The number of colors also plays a significant role in that additional bytes will be used to store the color information and the color compression scheme. Most of the stored pixels are background and/or noise or other non-pertinent information, yet with all of the required data represented by pixels, no information on the image content or any of the features is maintained. Without being able to identify individual features in an image, the bit-mapped image is of little use in a GIS system.
1.2 Document Conversion Methods

One common method of map conversion to electronic form is the traditional approach of hand-digitizing. This practice has been common in several mapping agencies such as the National Imagery and Mapping Agency (NIMA), formally the Defense Mapping Agency (DMA), and the US Geological Survey (USGS). Hand-digitizing requires large drafting tables, sophisticated and very expensive software, and a tremendous amount of human effort and patience. Advantages of the hand-digitizing approach are its accuracy and the precision of the geographical coordinates along with the ability to enter comments or identifiers along with the data as it is entered into a database. This is the basis for maintaining a Geographical Information System (GIS) database. Each object has a selected set of features or attributes associated with it. Some features are known locally and others may be located in other documents or sources. For example, an object identified as a street or a road may have features such as name, orientation or direction, and building material, where a river may contain a name, a flow direction, and current velocity.

Another common document conversion method, which is the starting point for this research, is the use of a raster-scanning device to automatically digitize a document and convert it to a bit-mapped form. While the digital rasterization of a document is much easier and faster than digitizing it by hand, the normal processing algorithms applied to the digitized documents are only slightly less labor intensive than manual digitization. Processing of a digitized document typically involves editing it to remove obvious artifacts, running the transformation or vectorization software to perform the conversion, and then a final editing of the vector image. These algorithms and approaches are often marketed as commercial software packages that are advertised as universal raster-to-vector converters. Usually no mention is made of the significant amount of user effort required. In essence, the software is not much more than the packaging of a raster editor and, maybe, a vector editor around a thinning algorithm. Claims of universal image converting software are usually overstated even though some can handle a wide variety of image types (e.g., text documents and CAD drawings), but even then, only after some user intervention. The greatest amount of user effort is generally concentrated at the end of the raster-to-vector conversion process in the post-processing phase. Much of that effort is due to noise or gaps in lines which are introduced by their inadequate algorithms.

1.3 Hue Histogram Processing

The purpose of this paper is to present an improvement to the common method of raster-to-vector conversion by the semi-automatic pre-processing of digitized color images before the vectorization process is applied. This is done by converting the images from Red, Green, Blue (RGB) format to Hue, Saturation, and Intensity (HSI) representation and then analyzing the histogram of the hue values to identify particular features (types of lines) which are represented in documents by different colors. A histogram of the hue values which are present in the image indicates which hues should be used to produce several binary-valued images which are vectorized separately and ultimately recombined into the complete vectorized image.

This approach of using hue values to create several binary-valued images results in less ambiguous data being passed to the vectorizing process and thereby producing less-noisy and more contiguous vectors. The result is that less post-processing is required, which results in a
savings in conversion time and reductions in user interaction. This paper includes the description and development of the technique, the procedure to apply it, and a demonstration of its efficacy on a typical map. The method has been effectively applied to a variety of different types of nautical and aeronautical charts and color maps, with both colored shading and colored lines.

The following images briefly illustrate the usual vectorization technique as well as some of the common problems associated with it. The first image, Figure 1, is a small portion (300x300 pixels x 256 colors) of a raster scanned map. The second image, Figure 2, is that of Figure 1 amplitude quantized to one bit (binary valued, black and white, monochrome, or 2-color) since the thinning algorithms work only on binary images. The third image, Figure 3, is the result of applying a thinning algorithm to the binary valued image of Figure 2. Upon close inspection, the inherent problems with this unsophisticated technique can be seen in Figure 4 and Figure 5.

During the post-vectorization phase, line-tracing algorithms are applied to the thinned line image of Figure 3 and manual vector editing is performed to remove the remaining errors. This manual editing includes eliminating gap errors, ghost branches, and point noise introduced during the thinning algorithm and as a result of the simplistic pre-processing of the bit-mapped image. The images presented in this paper have no post-processing applied to them in order to best demonstrate the effectiveness of the hue histogram approach to reducing vectorization errors at the source.

Another failing of current raster-to-vector conversion approaches is the loss of uniqueness for each line, that is, its identity, type, or meaning. Lines depicting roads, streams, contour lines, and map reference markers in the original image are drawn with varying color and weight in the original image but their uniqueness disappears when they are converted to a binary-valued image for vectorization. Processing each line by its color or hue not only eliminates this problem, but also provides the distinct advantage of being able to tag each line with its appropriate type for easy incorporation into a GIS. Processing by hues also reduces gap errors and ghost branch errors by removing the interfering lines before they are vectorized.

1.4 Color Quantization

The color to binary-valued conversion is essential to the raster-to-vector conversion process since there is no known method for line thinning of colored lines. In rasterized images, colored areas and lines usually consist of more than one pixel and the digitization process is forced to quantify the color of a pixel even though it may contain more than one color. For example, the dark red line in Figure 6 (which may depict a road) when viewed pixel-by-pixel contains many different hues of red. This is exemplified in Figure 7, which is a small section taken from Figure 6 Magnified sub-section of the original image and blown up several times by pixel replication in order to show the pixel pattern. The background, when viewed at the correct scale appears yellowish, but when viewed in Figure 6, appears to be a collage of yellows and greens and other noise.

This color quantization problem is the result of color pixel manipulation and interpolation schemes used in the rasterization process and is independent of the vectorization process. In
essence, this type of scanner error is independent of the quality of the hardware or software and illustrates one of the physical constraints of the process of converting an analog document to a pixelized form. As the red line in Figure 7 transverses from left to right, notice that the hues or colors vary for each corresponding block to the right. This is the result of having two or more colors in a single block, where each block represents a pixel in accordance to the spatial quantization of the scanner. While it may be possible to digitize the image to higher resolution, this only serves to increase the quantity of data to process and may still not alleviate the problem. Figure 7 shows how a red line can be composed of several hues, however, each of these hues will deviate very little from the pure red hue.

The net result of the unavoidable errors in the color conversion process is that a line which has a single color in the original document may have a range of colors once it is digitized. This increases the processing required if the data is manipulated in the color space in which it is digitized, namely red, green, and blue (RGB). However, if a simple bilinear transformation of the data is made from RGB space to the hue, saturation, and intensity space (HSI), then it can be seen that only the one variable, hue, is spread as a result of the quantization process. Classifying by hues does not change the color of any of the pixels, only their representation.

1.5 Color representation

A digitized raster image is an image comprised of colored pixels (picture elements). Each pixel has a value which is a point in a three-dimensional color space. The axes of the three-dimensional space represent three primary colors, red, green, and blue. The RGB coordinate system is the most common form of representation, but there are other coordinate systems which may be more appropriate for different applications and are bilinearly related to the RGB system.

Since the early 1980’s [5] much research has gone into the area of raster-to-vector conversions. Most of the early research and many of the algorithm designs have been applied in the commercial world as OCR software or CAD drawing converters. All use a thinning algorithm, in one form or another, as their basis. Each varies only with the interpretation of the true single-line representation for an object or line. As with any relatively new field of research, there has been no standard or universally accepted thinning algorithm. Furthermore, many of the newer algorithms are designed using parallel computer implementations to perform many of the repetitive tasks concurrently in order to decrease the processing time. New digital cartographer mapping systems are also now coming on-line. These hybrid systems try to offer the best of both the raster and vector worlds [6]. These systems show much promise, but presently they tend to be expensive, experimental, and restricted to specific domains.

1.6 Line Thinning

H. Blum is credited for proposing the original thinning algorithm as a medial axis transformation in 1967 [4,7]. In this article, he suggests that in order to determine what an image is, it needs to be reduced to its medial axis (thinned) form. From the medial axis, and only then, can the image be processed further. His algorithm used a “nearest neighbor” approach, in that every pixel is analyzed to find its nearest neighbor as determined by an established measure. If a pixel has only one neighbor, that pixel is removed. This algorithm continues with multiple passes of the
A thinning algorithm is, by definition, a binary image-processing technique that reduces objects to sets of one-pixel wide curves. Since the development of these initial algorithms, numerous thinning algorithms have been proposed for improving computational efficiency. A typical approach of these algorithms is to iteratively delete edge points, while constantly checking to ensure end points are not removed, thus the algorithm does not break the connectedness of a line, and the algorithm does not erode the region [4]. All thinning algorithms have the same premise: reduce the edges until only a medial axis remains. The details of each thinning algorithm vary widely with the research that prompted it. Hardcore theorists insist that thinning algorithms generally occur in two categories: the traditional medial axis transformation or skeletonization, and pure thinning. Each produces a single-pixel wide line approximately depicting the center of the original line. Each is essentially similar but applied differently. In comparison, the results of the thinning algorithms may vary in some pixels, depending on the shape of the region [8]. The main problem with the thinning algorithm is that it requires binary-valued data and the algorithms only work adequately for text documents and many CAD drawings, while performing poorly with multicolored images, in particular, color map images.

The work presented in this paper uses the thinning algorithm designed by Zhang and Suen in 1984 [4,9]. It is an algorithm which searches for and marks adjacent pixels that may be edges, determines which are actually edges, removes the unwanted pixels, and then repeats until only a single-pixel wide line remains for each object in the image. The marked values and configuration determine which neighbor pixels are eliminated. This is a common iterative thinning algorithm that stops when the operation removes no additional pixels.

1.7 Line Simplification

Raster-to-vector research is not only limited to thinning algorithms, but also to line simplification (to reduce the number of points and not lose any information), clustering of information (as opposed to categorizing everything as linear points), and intentional distortion (exaggerating details for a realistic appearance). Research into other pre-processing and post-processing aspects continue to increase. Examples include “de-speckling” (the removal of random noise) the raster image prior to processing, and adding geometric algorithms and filters designed to enhance line-tracing capabilities of the post-processor for pattern matching, as well as recovery of 3-D shapes from 2-D images [10,11].

1.8 Commercial Raster-to-Vector Programs

Maps and charts are at a much higher level of complexity than text documents for which optical character recognition (OCR) is used or computer aided design (CAD) drawings because the latter are pro-forma and consist of a multiplicity of readily identifiable shapes and/or straight lines. In addition to straight lines, most map documents have many curved lines (e.g., contour lines, roads, streams, etc.) which tend be non-predictable in both location of occurrence, and in direction. Pattern recognition techniques which work well for OCR and CAD drawings do not perform well on this type of document. Even the best commercially available raster-to-vector converters require that the image be binary-valued and usually require hand-manipulation by the
user before and after vectorizing. Some of the commercially available algorithms and software packages claim to process contour lines (even contour lines with elevation markings). These fall into the same labor intense trap mentioned earlier in that they are inefficient without extensive user interaction, and they still do not discriminate among different types of lines due to the nature of the binary-valued input.

2. **Hue Histogram Processing**

The three-phase process presented here first converts the bit-mapped RGB image to its HSI representation. The hue data is then used to generate a histogram which is analyzed, currently manually, but easily automated, in order to determine the clusters of hues which represent features of significance. Thresholds are determined from the hue histogram and used to generate several binary-valued images, one for each significant hue or cluster of hues. These binary-valued images are then thinned, vectorized, and recombined. First we discuss the RGB-HSI equivalence and then the processing of the histograms.

2.1 **HSI Color Representation**

The color-coordinate system used in this technique, the hue-saturation-intensity (HSI) coordinate system, is more closely related to a human’s perception of color and has been considered the RGB color’s most natural components [17], although no color coordinate system is universally accepted as corresponding to human perception [12]. Other color-coordinate systems include YIQ (matrix manipulation of RGB to represent transmission efficiency and monochrome TV standards), LHS (luminance, hue, and saturation – similar to the HSI system with a calculation for luminance instead of the intensity component), and CMY (cyan, magenta, yellow – primary pigment system) [4, 8, 12].

Both the RGB and HSI are three-dimensional representations, but with different orientation. Figure 9 and Figure 10 show the basic representations of color in the RGB system and the HSI system. The differences between the two systems are immediately apparent. In 3-D space, an RGB pixel at the point (0, 0, 0) represents no red, no blue, and no green; in other words, the absence of light, black. Points which are located away from the RBG origin represent different colors as being the sum of different amounts of the three primary colors. Movement along any two of the RGB axes simultaneously tends towards one of the secondary colors (cyan, magenta, and yellow, the CMY system) of representation. CMY are also referred to as the primary colors of pigment. As the value of all the three axes increases simultaneously, the resultant color moves towards pure white. The transition along this R=G=B vector represents the gray-scale. In contrast to the Cartesian representation of the RGB system, the HSI system is represented by a right circular cylinder. In Figure 10 the hue (H), what we typically call color, is depicted as the angle of the vector offset from zero degrees. Zero degrees (0°) is defined as red, 120° as green, and 240° as blue. The hue varies with the spectral wavelength, from red to violet, and thus, is also known as establishing the natural coloring of the image [12]. Therefore, a normalized hue histogram can be regarded as a vector that represents image features [1].

The saturation (S) is the magnitude of the HSI vector. The magnitude of the vector characterizes
the degree of purity of the color with respect to its dilution by white light. The intensity (I) is the
height of the cylinder, perpendicular to the plane of vectors representing hue and saturation. The
intensity is defined as the brightness of the color and will vary as a function of the relative
direction of illumination [13], but will not vary with the color components. With saturation at
zero, the hue is undefined and the transition along the intensity vector is the gray-scale.

By definition, the hue and saturation combination can represent any color. Therefore, all colors
have a different hue and saturation combination. Intensity, then, must provide separate
information. With the intensity (I) value orthogonal to the hue (H) and saturation (S) vector,
vector algebra shows that the intensity value will contain no color information. As the color
varies, the hue and saturation values will change but the intensity will not necessarily change,
unless the brightness level of the color changes. Therefore, intensity can be considered
independently of the color components. Other research has verified this with histogram
operations on the illumination component (in a hue, saturation, and luminance (HSL) system,
luminance being comparable to intensity) while leaving the hue and saturation unchanged [14].
Figure 11 shows the result as two-dimensional vector space.

This principle is important for two reasons. First, the complexity of the problem has been
reduced from a three-dimensional problem to a two-dimensional problem. Second, by removing
a dimension, hue and saturation can be shown as intrinsically related to the way the eye can
perceive colors. Therefore it is a more ideal color system when compared to the RGB tricolor
coordinate system.

Recall that hue portrays the spectral wavelength relationship of every visible color and the
saturation depicts the relative purity of the color. If each pixel in a raster image were plotted
only with respect to hue, then colors that appear different in an RGB sense, but in essence, vary
only in saturation value, would be grouped together within a small range of hue values. This is
the property that this technique exploits. For example, all of the “red” pixels in the map of
Figure 1 when digitized, do not all have the single hue “red.” These “red” pixels can be
categorized into only a small range of hue values, regardless of the saturation.

2.2 HSI to RGB Conversion

Based on the representations previously discussed, it is easy to see that the conversion from RGB
to HSI can be derived by simple trigonometric and algebraic manipulation [4, 8, 12]. Assume
that the range of values for each axis is normalized to [0, 1]. In the two systems, the gray-scale
is a common bisecting vector. Align this grey-scale vector in the RGB system with the HSI
gray-scale vector by rotating the RGB system so that the gray-scale vector is the z-axis, the same
direction as in the HSI system. Assume that red lies along 0° in the HSI system and the x-axis of
a right-hand coordinate system in the RGB system. From a top-down perspective, this places the
G-axis at 120° and the B-axis is at 240°.

The maximum length of the gray-scale vector in the original RGB system is \(\sqrt{3} \). Although the
formula for the gray-scale vector length is \(\sqrt{R^2 + G^2 + B^2} \), it is equivalent to \(\frac{\sqrt{3}}{3}(R + G + B) \).
Therefore (when normalized to [0,1]),
\[ I = \frac{R + G + B}{3} \]  
(1)

\[ S = \sqrt{\frac{2}{3} \left( R^2 - RG - RB - G^2 - GB + B^2 \right)} \]  
(2)

It is common to approximate the saturation formula with,
\[ S = 1 - \left( 3 \min \left\{ R, G, B \right\} \right) \]  
(3)

\[ H = \cos^{-1} \left( \frac{x}{S} \right) \]  
(4)

which is:
\[ H = \frac{\cos^{-1} \left( \frac{1}{2} \left( 2R - G - B \right) \right)}{\sqrt{R^2 - RG - RB + G^2 - GB + B^2}} \]  
(5)

3. **Hue Histogram Analysis**

The hue histogram shows colored features as large amplitudes centered around particular hue values. Each peak of the histogram reflects the occurrence of a colored line, colored feature, or background, thus allowing one to easily chose the hue values for decomposing the image into several mutually exclusive bit-maps representing the different features and/or background.

The largest value in the histogram of most documents is caused by the background since it covers the largest area of the raster image and hence the greatest number of pixels. This background peak indicates the hue or range of hues which should not be considered for vectorizing and are discarded. It is expected that not all of the colored features in the histogram will be represented by single hue values and some features may be comprised of a cluster of hue values. The determination of which hues to consider is now done manually, but hue clustering algorithms can be used to automate this process. Figure 12 shows the procedural flow of the process and subsequent thinning, which iterates until all of the hue peaks or hue clusters have been analyzed and converted to binary valued images for thinning and vectorization.

A hue histogram measures the frequency of occurrence of hue values on an arbitrary scale from 0-359. Efficient processing of the hue histogram necessitates the need for maintaining certain threshold values and obtaining certain statistical markers [15,16]. The statistical markers that will be used are obtained from the hue histogram. Each histogram hue (or range of hues) which exceeds a frequency of occurrence threshold is used to establish partition values ($\lambda_{\min}$ and $\lambda_{\max}$). These are used to convert the color image into several binary-valued images as defined by
the formula below:

\[
P_{\text{binary}}(x,y) = \begin{cases} 
1: & \lambda_{\text{min}} < P_{\text{in}}(x,y) < \lambda_{\text{max}} \\
0: & \text{else}
\end{cases}
\]  

That is, pixels with hues between the partition values are set to 1, all others are set to 0 (zero) [17]. The partition values can be easily chosen from the histograms in such a way that the desired vector line degradation is reduced to nearly zero.

4. **Demonstration of Automatic Hue Histogram Technique**

For the demonstration, we have chosen a small section of Figure 8, namely Figure 1. It is a section of a map of Brazil produced by the USGS and scanned into raster form by an HP ScanJet II scanner with a resolution of approximately 75 dpi. Although the subimage of Figure 1 is only 300x300 pixels, the procedures described herein work on images of all sizes and have been applied to complete maps from different sources.

The procedure which was applied to produce the following images consists of the following steps:
1. Convert the raster-scanned RGB image to HSI format.
2. Compute the hue histogram.
3. Analyze the hue histogram to determine which hues or ranges of hues exceed a frequency of occurrence threshold indicative of a features of interest.
4. Convert the image into a binary-valued images based on the hues selected in step 3.
5. Thin and vectorize the binary images
6. Combine all of the vectors created from the individual binary-valued images into a single image.

For the histogram analysis, each bin of the histogram (ranging from 0 to 359) represents one degree of the angle calculated for the hue. The hue bin scale can be set arbitrarily with greater or lesser resolution however 1 degree increments perform satisfactorily in our experience. Figure 13 is the histogram from the entire image. Notice bins #40, #43, #46, and #60 have a large number of pixels in each, however other bins as #41, #42, and #44 have none. This would indicate that there was a finite amount of scanner noise throughout the image since all of the pixels are restricted to a small number of discrete bins [18] and that the 1 degree resolution of the histogram is sensitive enough to detect the scattering of hues across a range due to the pixelation previously discussed. The majority of pixels are concentrated between bins #40 through #60 and are assumed to make up the background which is of no interest. That is, pixels with a 40 < hue < 60 are discarded.

If the number of occurrences scale is expanded so as not to be dominated by the hues of the background pixels as shown in Figure 14, then one can also see that there are some hue values which have a very low number of occurrences. Since there are 90,000 pixels (300^2) in this image, bins with relatively few pixels (e.g., less than a small percentage, typically 3σ ) in the histogram can be assumed to be of no interest and, thus, also can be ignored [15,19]. The hues
or clusters of hues which are left provide the hue values for generating the several binary-valued images for thinning and vectorization.

If we use only single hue values, such as the vector images produced by hue 0 shown in Figure 15 and hue 19 in Figure 16, it can be seen that some or part of each line is missing. If the rasterization were perfect and the histogram bin size correct, then features would be characterized by only one hue. From this example, it can be seen that one bin alone does not represent the entire red line on the original image. It is apparent that more than one bin is required to accurately represent this line. Figure 17 shows the result of processing the two hue bin values as a cluster in a single hue-to-binary conversion step.

The vector image created from hue-cluster processing is much more complete and is virtually intact. Morphological filtering could be applied at this stage with an expansion and dilation operator to connect lines with gaps. If the range of hues is expanded to include the next plausible bin in the histogram at hue 31, the resulting vector image of Figure 18 shows a complete line set. More noise is introduced, however, as some of the background pixels are captured in this bin. If the hue bin count were expanded to a finer separation, say a half-a-degree per bin, the situation would not necessarily improve. It is far more likely that the missing elements in the red line also appear in the background or as noise. Another example of the hue smearing due to the digitizing process is the image of Figure 19 representing only hue 120 and that of Figure 20 generated from hue 120 and those clustered around it.

The process of selecting hue-clusters can be somewhat automated by choosing a larger bin size for the hue histogram. For example, analysis of the histogram could have started with bin sizes of two or more. For certain maps and with experience in this hue-histogram identification process, standard bin sizes could make the process more efficient and automated, however the issue of automation has not yet been investigated as the process of hue histogram vectorization has itself yielded such a significant improvement in raster-to-vector conversion quality.

Hue processing alone has identified and vectorized most of the features. Two issues remain which are covered next. These problems are lines which lie on the gray-scale (e.g., black and dark gray lines) and features with the same hue as the background.

### 4.1 Hue Processing of Black and Grey Lines

The dark-gray or black elevation lines are not directly handled by hue-histogram vectorization. First, recall that in the HSI system, black is defined as having no intensity. This implies that the values for hue and saturation could be any value. An added difficulty occurs when red and green and blue are all the same value (for example 0), the saturation becomes zero and the hue becomes undefined. Therefore, it becomes difficult to measure black or white or other gray-scale colors on a hue histogram, particularly if saturation is zero. This issue has been resolved by assigning black and the other gray-scale colors with a saturation value equal to zero special bin #361. This value is processed as if it were a hue like the others.
4.2 Hue Processing of Similar Colors

Upon processing all of the hue histogram values or clusters, no other features can be identified within the histogram. A visual inspection of the map of Figure 1 shows that the contour lines have not been vectorized, therefore, it is apparent that the contour lines have one (or more) of the same hue values as the background pixels. This indicates that a further refinement must be applied to the remaining and discarded background pixels in order to extract this information. Figure 21, Figure 22, Figure 23, and Figure 24 are binary images representing the pixels of hue bins #40, #46, #49, and #60 from the hue histogram. These bins were originally assumed to be the background pixels due to the extraordinary number of pixels in each bin.

Notice that the contour lines are visible in Figure 24 and intermixed with the background pixels indicating that additional analysis must be performed to vectorize it. Since two visually distinct colors from the color image are used on this map which (unfortunately) have the same hue value, saturation must be used as a secondary discriminant. A simple method using saturation and a saturation histogram helps eliminate much of the noise encountered when using hue value ranges, as in Figure 25 and in Figure 26 when the hue range is expanded. Fine tuning the parameters allows only certain saturation values to be considered. Each saturation value is for a particular range of hue values and within a particular saturation-threshold of number of pixels. As with the hue histogram, the saturation peaks in the saturation histogram occur where the concentration is densest. Each saturation histogram is calculated for a single hue value only.

For a range of hue values depicting the background, particular saturation histogram peaks also identify desired features from the other lines and noise within that saturation range. All other hue values may contain some of the background or other noise, but they won’t occur as a maximum. Conversely, the saturation peaks indicate a purer concentration of the actual color being sought thereby reducing noise. Figure 25 shows the variations in saturation across the background pixel hues, and Figure 26 shows the saturation histogram of hue bin #60 alone.

The saturation histograms for each of the discrete background hue bin values (bins #40, #46, #49, and #60) in Figure 25 shows little deviation in saturation values except in bin #60, where the saturation values are distributed throughout the range, as shown Figure 26. Once again, the saturation histogram range is set arbitrarily to 0 - 100. The range depicted satisfies the need to isolate each individual occurrence of saturation values without overlap. By applying the same principle to the saturation histogram as with the hue histogram to identify ambiguous features in the color raster map image, the saturation histogram can identify particular sub-features within a single hue value. Sub-features can be identified as noise, extraneous background pixels, or as in this case, contour lines. Figure 27 is an image of the pixels which have a hue of 60 and a saturation value of 100. Vectorizing the hue and saturation combination of Figure 27 produces the vector image of Figure 28.

4.3 Results of the Hue and Saturation Processing

After the saturation processing, all of the humanly distinguishable features and lines have been successfully extracted from the original color image. Figure 29 shows the image drawn with the vectors from the hue and saturation analysis with each identified vector drawn in a color distinct
from the others. It is important to note that no post-processing has been performed on the image so minor gaps and branches may be seen however there is an insignificant number relative to that which results from the usual vectorization process. For comparison, the vector output image from the conventional binary image processing is also shown in Figure 30.

In addition to the improved vectorization, an equally significant result of hue-processing is the identification and extraction of individual vector types based on hue, a result no current raster-to-vector conversion package does automatically. In Figure 29 all of the roads (in red) and streams (in green) are unique, whereas in Figure 30 each vector line intercepts every other vector, making the line identification and post-processing very difficult and highly labor intensive.

With minor post-processing of each of the vectors and data entry and other GIS processing routines, the image in Figure 31 can be easily produced at a total manpower cost significantly less than other methods. Figure 31 also shows a simple example of possible data entry attributes for each of the features located in the image. Vector image post-processing is beyond the scope of this paper, however it can be easily seen that with the extraction of individual hues many post-processing techniques can be applied with greater effectiveness.

5. Conclusion

The need for converting and storing existing paper documents to digital format is growing as computers become more powerful and as requirements for maintaining legacy information increases. Much of the need involves saving physical space and accessibility. Many of these paper-map documents, when retained digitally, can be enhanced to augment the information or enhanced to provide additional relevant information. This paper introduces and demonstrates that identification and extraction of colored features in color images can be achieved by manipulating the pre-processing stage with an analysis based on the hue-histogram. In some cases this must be followed by a second stage of saturation histogram processing for any unresolved features. This capability is neither in the literature nor in commercial software. Errors inherent in the raster-scanning processing itself can also be reduced by using hue-clusters.
6. Figures

Figure 1  Original color image of map.

Figure 2  Original image converted to monochrome.

Figure 3  Thin monochrome image.
Figure 4 Example of gap error.

Figure 5 Example of ghost branch error.
Figure 6  Magnified sub-section of the original image.

Figure 7  Typical quantization of a line in a document to colored pixels.
Figure 8  Original raster scanned image (reduced).
Figure 9  RGB three-dimensional space.

Figure 10  HSI three-dimensional space.
Figure 11 Two-dimensional HS space.

Figure 12 Procedural flow of hue histogram processing.
Figure 13  Hue histogram of original image.

Figure 14  Expanded scale of hue histogram.
Figure 15  Vector output using hue = 0.

Figure 16  Vector output using hue = 19.
Figure 17  Vector output using combined hue values.

Figure 18  Vector output of increased hue value combinations.

Figure 19  Vector output using hue=120.

Figure 20  Vector output using hue values around 120.
Figure 21  Monochrome image of hue=40.

Figure 22  Monochrome image of hue=46.

Figure 23  Monochrome image of hue=49.

Figure 24  Monochrome image of hue=60.
Figure 25 Saturation histogram for hue range 40-60.

Figure 26 Saturation histogram for hue=60.
Figure 27  Monochrome image of Hue=60 and saturation = 100.

Figure 28  Vector output of process with hue = 60, saturation = 100.

Figure 29  Colorized and reconstructed vector output from hue histogram.

Figure 30  Thinned monochome image.
Figure 31 Sample post-processed vector image.
7. References


