2005

Creating Geo-specific Road Databases From Aerial Photos For Driving Simulation

Dahai Guo
University of Central Florida

STARS Citation

Creating Geo-Specific Road Databases from Aerial Photos for Driving Simulation

by

DAHAI GUO
M.S. University of Central Florida, 2001

A dissertation submitted in partial fulfillment of the requirements
for the degree of Doctor of Philosophy
in the Department of Electrical and Computer Engineering
in the College of Engineering and Computer Science
at the University of Central Florida
Orlando, Florida

Summer Term
2005

Major Professors: Harold I. Klee and Arthur R. Weeks

ABSTRACT

Geo-specific road database development is important to a driving simulation system and a very labor intensive process. Road databases for driving simulation need high resolution and accuracy. Even though commercial software is available on the market, a lot of manual work still has to be done when the road crosssectional profile is not uniform. This research deals with geo-specific road databases development, especially for roads with non-uniform cross sections.

In this research, the United States Geographical Survey (USGS) road information is used with aerial photos to accurately extract road boundaries, using image segmentation and data compression techniques. Image segmentation plays an important role in extracting road boundary information. There are numerous methods developed for image segmentation. Six methods have been tried for the purpose of road image segmentation. The major problems with road segmentation are due to the large variety of road appearances and the many linear features in roads. A method that does not require a database of sample images is desired. Furthermore, this method should be able to handle the complexity of road appearances.

The proposed method for road segmentation is based on the mean-shift clustering algorithm and it yields a high accuracy. In the phase of building road databases and visual databases based on road segmentation results, the Linde-Buzo-Gray (LBG) vector quantization algorithm is used to identify repeatable cross section profiles. In the phase of texture mapping, five major uniform textures are considered - pavement, white marker, yellow marker, concrete and grass. They are automatically mapped to polygons. In the chapter of results, snapshots of road/visual database are presented.

I dedicate this dissertation to my beloved wife, Jiaying, who has been giving me tremendous support during my graduate studies.

I also want to thank Dr. Klee, Dr. Weeks, Dr. Bauer, and Dr. Radwan. They brought me to the field of driving simulation and have given a lot of instructions.

Dr. Foroosh is appreciated for his service in my committee.

TABLE OF CONTENTS

LIST OF FIGURES

LIST OF TABLES

LIST OF ACRONYMS/ABBREVIATIONS

USGS	The United States Geographical Survey
DLG	Digital Line Graph
DOQ	Digital Orthophoto Quatrangle
DEM	Digital Elevation Model
LBG	Linde-Buzo-Gray

CHAPTER ONE: INTRODUCTION

Driving simulation can contribute greatly to transportation research, especially when the focus is on driving behavior. A goal of driving simulation is to make driving scenarios as close to reality as possible. The road database is one of the most important factors in deciding how close a driving scenario is to the real world. Road databases can be based on real scenes, such as the UCF campus, or just imaginary. The former type is called geo-specific and the latter is called geo-typical. Geo-typical roads are easier to create than geo-specific roads are. Manually creating geo-specific road databases is very time consuming. The cost could be much higher when a road database is for driving simulation, because, ideally, the accuracy of the data should match the dimension of the smallest object that can be seen by drivers.

Thanks to the advances in photogrammetry techniques, aerial images with high resolution are available and can provide many details of roads. The aerial images in this research have a resolution of 0.2-0.3 meters per pixel, as measured in this research. Even though aerial images can help in the process of developing geo-specific road databases, it still takes modelers a lot of work. The amount of work depends on how complex the real scene is and on how detailed data should be. For example, a straight road could simply be represented by a vector (*start, end*) in a map. However, this amount of information is far from enough to describe a road for driving simulation. Motion based simulators need to know how to react to the surface of the road at each point. This straight road could have irregular shaped medians. Therefore, motion of a simulator could vary at different lateral positions on a straight road. Most research about road extraction from aerial images is for updating maps or creating road networks such as [2], [5], [8], [10]-[13],

[15] and [19], These methods focus on recognizing center line information. This level of detail is not fine enough for driving simulation.

Most research in road extraction begins with an entire image. It is difficult and computationally expensive to extract roads due to presences of other road-like features with straight edges. The USGS has been collecting and updating geographical data for many years. This data includes road center lines, aerial photos, etc. Nearly every zip code in the United States has been documented. Fortunately, this data is either free or accessible at a small cost. USGS geographical data is very helpful in analyzing aerial photos. In this research, USGS geographical data serve as starting points for extracting road data. This is one of the major contributions to road database development in this research.

While USGS data can easily limit search space for roads in aerial photos, road segmentation is still very difficult, because of the non-uniformity of road cross sections. A suitable image segmentation method has to be chosen. Six methods have been tested. They were an edge grouping method [31], a texture-based comparison method using histogram intersection techniques [30], a feature-based comparison method using Expectation-Maximization (EM) method [26], a region-growing method [25], an active contour method [27], and the mean-shift clustering method [20]. Edge grouping methods [31] can accurately delineate road boundaries, but it is sensitive to noises, such as pavement marking and traffic. Because the aerial photos used in this research have a very high resolution of 0.2 to 0.3 meters per pixel, the noise can affect segmentation results dramatically. In order to address the noise in photos, complex mathematical models are needed, but it is difficult for them to be universal. The texture-based comparison method [30] using histogram intersection algorithms picks sample images which are usually as small as 10X10 and compares their histograms of color information with the histogram of each pixel’s neighbors in object images. This method work well when samples are taken from an image whose appearance is close to that of object images. When this condition cannot be satisfied, segmentation results will be affected. In this situation, a large amount of samples are needed. The region-growing method [25] merges each pixel with its neighbors if they are similar. This process stops when the boundary of the region is not similar enough with neighbor pixels. This method can divide an image into small uniform areas, but the problem is how to post process these small areas. The feature-based segmentation method [26] treats each pixel as a vector which may include hue, saturation and intensity. Then the EM method is used to group pixels with similar features. This method yields accurate segmentation results, however this method is computation-intensive. The mean-shift method [20] first determines the main objects in the image and uses information, such as average saturation to classify these objects. This method was adopted in this research, because it is simple and reliable. The aforementioned six types of methods will be explained in detail in Chapters 2 and 3.

The driving simulation system where the road databases are developed in this research was made by Simulation Technology Solution (STS). This company has its own format of how road information is organized and it is not straightforward to convert results of image segmentation to this format. This format requires typical cross sections which can be used repeatably. The LBG algorithm for vector quantization was used to reduce all the cross sections into a set of repeatable cross sections. Both STS road database format and the LBG algorithm will be discussed in detail in Chapter 4.

CHAPTER TWO: LITERATION REVIEW

2.1 Road Extraction from Aerial Photos

Numerous methods have been developed to extract roads from aerial photos [1]-[19]. The information that needs to be extracted could be either road center line information or road boundary information. The resolutions of the object images vary from 0.25 meters per pixel [20] to approximately 4 meters per pixel [11]. The higher the resolution, the more accurate the extracted information can be. The aerial images in this research have a resolution of about 0.25 meter per pixel which is high for aerial images.

Research of road extraction, such as [2], [5], [8], [10]-[13], [15] and [19], has concentrated on how to detect roads in images and retrieve road center line information. In [12] and [13], the authors first use two edge detection methods to extract edges from satellite images, and then fuse the results though an information combination operator. Numerous road candidates will be generated. If a true candidate is denoted as “1” and a false candidate as “0”, there would be 2^n combinations; where n is number of candidates. By doing that, the expressions of probabilities $p(D \mid L)$ and $p(L)$ can be defined, where D stands for the observation and L is an n-vector, composed of 0s and 1s. In [12], the observation field is the result of edge detection. The knowledge in $p(D \mid L)$ includes that 1) roads are long, but road edges can be any length; 2) non-road edges are more probable to be short; 3) edges should not be too short. In order to calculate $p(L)$, the authors divide a whole set of edges into subsets, each of which contains closely located edges. They use the Markovian Random Field and the order depends on how many neighbors an edge has. To maximize $p(D, L)$ that is the product of $p(D \mid L)$ and $p(L)$ with respect to L¸ many numerical methods can be chosen. This algorithm demonstrates good performance in extracting roads from satellite images. It only extracts road center information, instead of objects on the road, like curbs and medians. Another issue with this algorithm is that appearances of roads vary in different places, especially when the resolution of the aerial photo increases. The method in [8] is similar to the method just introduced.

In [17], instead of building a mathematical model, the authors created several rules. The process starts with the result of an edge detection algorithm. Two basic rules stated in [17] are that two edges within a close distance and a similar orientation have a high probability of belonging to a road. This method needs to have a training process to achieve some thresholds. The purpose of the method [17] is to extract center line information, instead of road boundaries. A similar method was tested in this research for extracting road boundaries and the results were rarely acceptable. The main problem with this method is that it is very difficult to find a universal set of thresholds and the training process has to be repeated sometimes. The method in [15] is similar to this method.

In [2], the authors use old maps as well as aerial photos to update existing maps. The outlines of the roads in the existing databases are compared to the cross profiles taken from the images at a certain position. If the information from the image deviates from what is in the existing database, a new road link is proposed. The algorithm then searches the start of the new road in addition to tracking this road. Primarily, this algorithm uses cross-correlation thresholds to detect new roads. This method is similar to the method proposed in this research because it also uses some existing geographical information combined with aerial images. The images in [2] have a resolution of one meter per pixel.

In [5], a semi-automatic method is proposed. In this method, an operator first picks several seed points near a road and decide its width. Afterwards, active contour models [18] are formulated. The internal energy in the active contour indicates the smoothness of the curve and the photometric observations formulate the grey level matching of images. Active contour models can be solved iteratively and the iteration stops when each element of the solution vector falls below a threshold. There is a trade-off in this method. If many accurate seed points are provided by the operator, there is no need to use the active contour model; instead a cubic-spline interpolation model can be used to interpolate those points. If too few seed points are given, it will not be reliable to use any mathematical model.

The method in [8] is similar to the one in [11] and [12]. It builds a model of road features. The edges are detected using a set of morphological operations. In the phase of edge linking, the Maximum a Priori (MAP) configuration of edge segments is estimated by minimizing an energy function.

The method in [10] works on low level features such as edges. Given an image, this method first segments elongated region and then a curve-fitting algorithm is used to fit a curve. This method works for satellite images which usually have a resolution much lower than aerial photos. In an image with high resolution, roads usually have many objects irrelevant to road segmentation, such as pavement markings, pavement discontinuities, and traffic. It is difficult to have good segmentation results for curve-fitting. Additionally, if all the objects are put into consideration, a very complex model must be available to handle all the situations. Furthermore, roads look different from place to place; therefore an adaptive method is needed.

Instead of extracting road center line information, some methods have focused on how to obtain road boundaries, such as [1], [3], [4], [6], [7], [9], [13], [14] and [16]. These methods put the width of roads into consideration. Some of these methods begin with no knowledge of where roads are, while some have some existing knowledge about road locations.

In [1], roads are described by a group of antiparallel pairs which are two smooth curves that are parallel pointwise to each other and have opposite gradient direction. Each antiparallel pair is expressed as

antipair(antipair_no, first_end(point(X,Y,Z), point(X,Y,Z)),
second_end(point(X,Y,Z), point(X,Y,Z)),
attribute(gradient_direction, length, width, average_intensity))

To generate antiparallel pairs, directions along a segment are determined and a search is made on either side of this segment. Thresholds are used to group these antiparallel pairs. The factors that are considered are distance, difference in direction, difference in gradient direction, difference in intensity, and difference in width. This method yields favorable results, but the road contents are very uniform, which is unlike the aerial photos in this research.

In [3] and [4], an image is first divided into pieces. For each piece, the probability of the road existence is estimated through MAP estimation. When building the mathematical model, four assumptions are made. They are 1) roads' width variance is small and road width change is likely to be slow, 2) road direction changes are likely to be slow, 3) road local average gray values are likely to vary only slowly, 4) gray level variation between road and background is likely to be large, and 5) roads are unlikely to be short. After the road candidates are chosen, they are merged using dynamic programming.

In [6], a high level method is proposed. This method emphasizes modeling context in aerial photos. A context could be shadow-occlusion, vehicle-road, and building-driveway-road-segments. For example, if a road is successfully identified, it is possible to find pavement markings. This method also uses some existing geographical information. The work in [7] is the extension of [6]. In [7], a given Digital Surface Model (DSM) is used together with images to obtain context information. A region is first classified into forest area, urban area, or rural area through a texture-based segmentation method. Then for rural area and urban area, DSMs are used to analyze context relations. Afterwards, the valleys of the DSM are hypothesized to be roads and tested in images. With the information of roads, markings could be identified in high-resolution images if they are clear. After roads are successfully extracted, they are connected to create a road network. The system in [6] and [7] is very complex and includes many different modules for different types of objects in aerial photos. It also takes advantage of existing geographical information to make it reliable. However, this system does not address some objects, such as pavement discontinuities and medians, in road areas.

In [9], a segmentation algorithm based on morphological operations is proposed to segment satellite images. This algorithm connects similar pixels locally and the connected regions keep growing until no similar pixel is found. It performs well in segmenting satellite images, but the images in [9] have a much lower resolution than the images in this research. Additionally, the method in [9] does not mention how to interpret segmentation results.

In [13], the author proposes a method which uses a growing active contour algorithm [19]. The original concept of active contours was proposed in [18] by Kass et al. It is used to interpolate given points by a geometrically smooth curve. Generically, it is a deformable model used for semi-automatic object delineation. The advantages of active contours include 1) constraints of geometric models 2) the ease in changing of external forces. The constraints of geometric models are regarded as internal forces. In order to delineate objects' contours, the energy of an active contour should be minimized. Numerically, an active contour' energy is represented as

$$E_{contour} = \int_0^1 E(v(s))ds = \int_0^1 [E_{\text{int}}(v(s)) + E_{image}(v(s)))]ds\,,\ v(s) = (x(s), y(s)) \qquad (1)$$

where $E_{contour}$ stands for the energy of the active contour, $E(v(s))$ stands for the energy at point $(x(s), y(s)$, $E_{\text{int}}(v(s))$ stands for the internal force at point $(x(s), y(s)$, $E_{image}(v(s))$ stands for the image force at point $(x(s), y(s)$.

Spline curves can be used as the geometric model in many active contours. The internal spline energy can be written

$$E_{\text{int}} = (\alpha(s)\,|\,v_s(s)\,|^2 + \beta(s)\,|\,v_{ss}(s)\,|^2)/2 \qquad (2)$$

where s, between 0 and 1 represents the displacement from the beginning point. $v_s(s)$ is the first order gradient of the curve at a certain point and $v_{ss}(s)$ is the second order gradient. $\alpha(s)$ and $\beta(s)$ controls the curve's importance of the membrane and thin plate terms. In some cases, they could be constants. Minimization of the internal energy makes the curve smooth. However, being smooth does not imply being consistent with objects' contours in images. Image forces can be defined based on individual needs. For example, active contour would tend to go to edges if the image force is defined as the reciprocals of gradients. For improved results, image forces can be defined as the projection of the image gradient on the curve’s normal at a certain point. When energy is being minimized, curve points are solved simultaneously. However this only works well when the initial points are selected close to real contours. This is because for each curve point, its local minimum is searched. In [19], the concept of growing active contours is proposed. It does not solve all the points along the curve simultaneously; instead it grows an active contour from the starting and ending points to the center. Unlike traditional active contour models, this method first applies the internal force, and then turns on the image force gradually from the ends to the center. In [13], a method for road extraction based on growing active contour was proposed. This method first performs edge extraction at the coarse level. Since in rural areas, roads are the most significant linear structures in aerial images at the coarse level; extracted edges are more likely to be roads. Then active contours grow from starting and ending points to centers. The stable width is assumed to be one of the roads' properties and is used as part of the image force, which filters out those non-road edges. In this research, another active contour-based method was tested and the results were not acceptable [27]. There are two reasons. The first is that it is hard to automatically find initial points in high-resolution images. The second is that there are many linear features in road areas, such as pavement markings, pavement discontinuities and so on. Active contour based algorithms will be affected by these noises.

In [14], the authors utilize a Digital Surface Model (DSM) and Digital Terrain Model (DTM) in image analysis. The roles of DSM and DTM are to remove unwanted image contents which are far from roads. This algorithm starts with multiple color images and then extracts features, such as straight edges and road markings. These features will be analyzed together with the DSM and DTM. The system in [14] contains a knowledge-based system to combine all the information either from images or DSM/DTM.

In [16], the authors proposed an unbiased detector of curvilinear structures. This method puts line width into consideration. When a linear feature appears as a bar-shape in the image, its width can not be ignored. In [16], good results are presented. However, the test images have relatively narrow and uniform roads, which are different with the roads considered by this research. Additionally, there is no median in the test images, while medians are often used for traffic operation purposes.

The method in [37] builds 3D models for urban environments. The modeling effort occurs at two levels: (1) ground-based modeling of building facades using 2D laser scanners and a video cameras, and (2) airborne modeling airborne laser scans and aerial images. For ground-based modeling, the processed scans are triangulated and texture mapped using the camera images.

The methods discussed above can be categorized into two classes. One is concerned with extracting center line information of roads, and the other attempts to extract road boundaries. Road center line extraction can be skipped in this research, because the USGS road center line information is used. For the second class, some of the methods are interested in delineating edges, and some involve grouping similar pixels in road areas. In high resolution images, the second approach was found to be more reliable, because there are too many linear features in roads to accurately delineate road boundaries.

2.2 Image Segmentation Algorithms

There are many image segmentation methods. Each method uses cues to segment images. Those cues can be RGB (Red, Green, Blue), or HSI (Hue, Saturation, Intensity). No matter which cues are selected, there must be a method that classifies pixels. In this section, six methods are described. These six methods have been implemented for road segmentation and their performances are analyzed in Chapters 3 and 5.

2.2.1 Edge Detection and Grouping

This research has adopted the Canny edge detection method [32]. In the Canny edge detection, the first step is to filter out any noise in the original image before trying to locate edges. The continuous Gaussian has the form of

$$G(x, y) = \frac{1}{2\pi\sigma} e^{-\frac{x^2+y^2}{2\sigma^2}} \tag{3}$$

The Gaussian filter is discrete and can be used in this case. Its mathematical representation is

$$m(i, j) = c\sum_{ii=-2}^{2}\sum_{jj=-2}^{2} m(i+ii, j+jj) * g(2+ii, 2+jj) \tag{4}$$

where $m(i, j)$ is the image value at position (i, j), $g(2+ii, 2+jj)$ is the $(ii, jj)th$ element in the discrete Gaussian mask, and c is a constant. The Gaussian mask at $\sigma = 1.4$ is shown as follows:

$$\frac{1}{115}\begin{pmatrix} 2 & 4 & 5 & 4 & 2 \\ 4 & 9 & 12 & 9 & 4 \\ 5 & 12 & 15 & 12 & 5 \\ 4 & 9 & 12 & 9 & 4 \\ 2 & 4 & 5 & 4 & 2 \end{pmatrix} \qquad (5)$$

The larger the width of the Gaussian mask, the lower the detector's sensitivity is to noise. After smoothing the image and eliminating noise, the next step is to find the edge strength by taking the gradient of the image. The Sobel operator performs a 2-D spatial gradient measurement on an image. Then, the absolute gradient magnitude (edge strength) at each point can be found. The Sobel operator uses a pair of 3x3 convolution masks, one estimating the gradient in the x-direction (columns) and the other estimating the gradient in the y-direction (rows). They are shown below:

$$Gx = \begin{pmatrix} -1 & 0 & +1 \\ -2 & 0 & +2 \\ -1 & 0 & +1 \end{pmatrix} \text{ (6) and } Gy = \begin{pmatrix} -1 & -2 & -1 \\ 0 & 0 & 0 \\ +1 & +2 & +1 \end{pmatrix} \qquad (6)$$

The magnitude of the edge gradient is calculated using the following formula

$$|G| = |Gx| + |Gy| \qquad (7)$$

Additional to the magnitude of the edge gradient, the direction of the edge gradient and the edge direction can be calculated using the following formulas

$$\theta_g = a\tan(Gy/Gx) \text{ (9) and } \theta_e = \frac{\pi}{2} - a\tan(Gy/Gx) \qquad (8)$$

Afterwards, there are two thresholds that are used to decide edge pixels. These two thresholds are denoted as T1 and T2, where T1>T2. Any pixel with $|G| > T1$ is classified as an edge pixel. Any pixel, connected with the previous edge pixel and with $|G| > T2$, is also an edge pixel.

In the phase of edge grouping, edges with short distance and similar orientation should be connected.

2.2.2 Textured-Based Image Segmentation By Comparing Histograms

The method in [30] uses color ratio gradient information as the cue for segmentation, and a histogram comparison method, which is called histogram intersection for classifying pixels. Details of color ratio gradient are explained in [30]. At least two query images taken from certain color ratio gradient images should be available for segmenting the original image. The segmentation is achieved by comparing the color gradient ratio histograms of the query images and each pixel's neighbor in the original image. The comparing scheme is called histogram intersection [33] cited in [30]. This algorithm compares the closeness of two histograms and can be written as a similar function (11) that returns a numerical measure [30].

$$D_a(H^Q, H^{I_i}) = \frac{\sum_{\vec{k}}^{N_d} \min\{H^Q(\vec{k}), H^{I_i}(\vec{k})\}}{\sum_{\vec{k}=1}^{N_d} (H^Q(\vec{k}))^2} \tag{9}$$

where H^Q stands for the histogram of query image and H^{I_i} stands for the histogram of the neighbor of i$^{\text{th}}$ pixel in image. Additionally, $\vec{k}$ stands for the bin index in a histogram and N_d stands for the number of bins in a histogram. Alternatively, this function can be defined as

$$D_x(H^Q, H^{I_i}) = \frac{\sum_{\vec{k}=1}^{N_d} H^{I_i}(\vec{k}) H^Q(\vec{k})}{\sum_{\vec{k}=1}^{N_d} (H^{I_i}(\vec{k}))^2} \tag{10}$$

The larger the returned value is, the more similar two histograms are. The query image that is the most similar to a pixel's neighbor decides this pixel's classification.

This method works well when the sample images are close to the test image. If this condition is not held, the results may be affected. As far as this research is concerned, it would be safe to have multiple sample images for each texture. But it will take a great deal of effort to create such a database. In [30], this method is used to extract data in image databases.

2.2.3 Feature-Based Comparison Using EM Method

The work in [26] presents a system called "blobworld". Its algorithm has three steps. The first is to prepare a database of blobs from other images. Then for each test image, this method extracts color and texture features, including L*a*b color coordinates, anisotropy, polarity, and contrast. Then the authors use the EM method for grouping pixels with respect to these color and texture information. The EM method is used for finding maximization likelihood parameter estimates when there is missing or incomplete data. In this case, the missing data is the Gaussian cluster to which the points in the feature space belong. The form of the probability density is as follows:

$$f(x \mid \Theta) = \sum_{i=1}^{K} a_i f_i(x \mid \theta_i) \tag{11}$$

where ***x*** is a feature vector and ***a*** is the mixing weights, $(\sum_{i=1}^{K} a_i = 1)$. The value ***K*** represents the number of Gaussian clusters. The value of Θ represents a set of parameters, such as $(a_0, \ldots.., a_k, \theta_0, \ldots.., \theta_K)$. The value of f_i represents a Gaussian multivariate density parameterized by θ_i, such as μ_i and Σ_i, which are the mean and covariance of a Gaussian cluster. And f_i can be expressed as

$$f_i(x \mid \theta_i) = \frac{1}{(2\pi)^{d/2} \det(\Sigma_i^{1/2})} e^{-\frac{1}{2}(x-\mu_i)^T \Sigma_i^{-1} (x-\mu_i)} \tag{12}$$

The equations that update a, μ, and Σ take on the following form:

$$a_i^{new} = \frac{1}{N} \sum_{j=1}^{N} p(i \mid x_j, \Theta^{old}) \tag{13}$$

$$\mu_i^{new} = \frac{\sum_{j=1}^{N} x_j \, p(i \mid x_j, \Theta^{old})}{\sum_{j=1}^{N} p(i \mid x_j, \Theta^{old})} \tag{14}$$

$$\Sigma_i^{new} = \frac{\sum_{j=1}^{N} p(i \mid x_j, \Theta^{old})(x_j - \mu_i^{new})(x_j - \mu_i^{new})^T}{\sum_{j=1}^{N} p(i \mid x_j, \Theta^{old})} \tag{15}$$

where ***N*** is the total number of feature vectors and $p(i \mid x_j, \Theta)$ is the probability that Gaussian clutter ***i*** fits the pixel x_j, given the data Θ:

$$p(i \mid x_j, \Theta) = \frac{a_i f_i(x_j \mid \theta_i)}{\sum_{k=1}^{K} a_k f_k(x_j \mid \theta_k)} \tag{16}$$

This process is iterative until the parameters converge. Initially, the mean vectors and the covariance matrices are calculated by dividing the image into two halves and calculating their statistical values. After the images are segmented, each blob is compared with the blobs in the database. The one in the database that has the shortest Mahalonobis distance decides the classification of the blob being tested.

2.2.4 Region-Growing Algorithm for Image Segmentation

Region-Growing algorithm starts with each pixel as an object [25]. At each step, pairs of image objects are merged into larger objects. The merging decision is based on local homogeneity criteria, describing the similarity of adjacent image objects. A “merging cost” is assigned to each possible merge. These costs represent the degree of fitting. The simplest way of calculating the merging cost can be evaluated by the Euclidean distance between two pixels’ RGB values, such as

$$h = \sqrt{\sum_d (f_{1d} - f_{2d})^2} \tag{17}$$

where f_{1d} and f_{2d} are one of the selected features and h indicates the similarity of two objects. Alternatively, the cost can be reflected by

$$h = \sqrt{\sum_d (\frac{f_{1d} - f_{2d}}{\delta_{fd}})^2} \tag{18}$$

which puts the standard deviation into consideration. When there is no similar object to combine, merging stops. The result of this type of methods is a number of comparably uniform regions in the image.

In [34], authors use this method to segment aerial photos and then use morphological operations and a fuzzy knowledge-based system to classify segmented regions. In detail, this method performs skeleton operation to each region. The end points yielded by the skeleton operation can define contour of region. If these end points form a rectangle-like shape, there may be a building. If these end points form an elongated object, there may be a road.

This method was not chosen for road segmentation in this research, because the aerial photos contain many irregular shaped objects, such as medians and pavement patches. These could make the post-classification very difficult.

2.2.5 Active Contour Model

The method in [17] is based on the active contour model. When updating a contour point, four types of energy are considered as follows:

1. The first is energy of continuity, which is denoted to $E_1 = | \bar{d} - | v_i - v_{i+1} ||$, where $\bar{d}$ is the average distance between each consecutive two points on a contour, and v_i is the ith point on the contour. The smaller this energy is, the more continuous the contour is at the interested point. Minimizing this energy will make contour points evenly distributed.
2. The second is energy of smoothness, which is denoted to $E_2 = | u_{i+1} - u_i |$, where u_i is the orientation at ith point. The smaller this energy is, the smoother the contour is at the interested point.
3. The third is energy of gradient, which is simply magnitude of gradient. This energy controls that the grown point should have a high gradient value. Because the sum of these four types energy will be minimized with respect to the position of the updated point, this energy is actually calculated as $E_3 = - | G_{i+1} |$, where G denotes gradient.
4. The fourth is energy of projected gradient on the normal of the line of v_i and v_{i+1}. It is denoted as $E_4 = G_{i+1} \hat{n}(v_i, v_{i+1})$. This energy reflects whether the line of v_i and v_{i+1} delineates a linear feature in the image.

In the calculation, each type of energy should be normalized to a value between 0 and 1 and be given a weight. Finally, the energy of the proposed point is expressed as follows:

$$E = \alpha E_1 + \beta E_2 - \theta E_3 + \gamma E_4 \qquad (19)$$

where α, β, θ, and γ are the weights.

In [27], a greedy algorithm is proposed. This algorithm searches each contour point in its neighborhood. The point with the least energy is chosen to be the updated point.

2.2.6 Mean-Shift Clustering Algorithm for Image Segmentation

The mean shift algorithm for image segmentation was introduced in [20]. It is a nonparametric method for searching the real estimation within a search window with radius *r*. The value of radius *r* decides the resolution of segmentation. Larger values of *r* result in lower segmentation resolution. In this paper, *r* is chosen to be the square root of the trace of the global covariance matrix in the HSI fields. The segmentation algorithm is defined as follows:

1. Given an aerial image, convert the RGB image to an HSI image. Each pixel in the RGB image maps to a point in the HSI image. The process of converting a RGB vector to its HSI vector is

$$\begin{pmatrix} Y \\ R-y \\ B-y \end{pmatrix} = \begin{pmatrix} 0.299 & 0.587 & 0.114 \\ 0.701 & -0.587 & -0.114 \\ -0.299 & -0.587 & 0.886 \end{pmatrix} \begin{pmatrix} R \\ G \\ B \end{pmatrix} \qquad (20)$$

$$\begin{aligned} & H = \tan^{-1}\left(\frac{R-y}{B-y}\right) \\ \text{and } & S = \sqrt{(R-y)^2 + (B-y)^2} \\ & I = Y \end{aligned} \qquad (21)$$

2. A 3D histogram of the HSI image is obtained.

3. *m* points in the HSI image are chosen randomly.

4. The point which corresponds to the largest value in the HSI histogram is identified.

5. Start from this point $\bar{x} = \{h, s, i\}$ in the HIS histogram, find the shift vector using the formula

$$\Delta x = \frac{r^2}{5}\frac{\nabla p(\bar{x})}{p(\bar{x})} \tag{22}$$

where $p(\bar{x})$ is the value in the HSI histogram at x.

6. Update $\bar{x} = \bar{x} + \Delta\bar{x}$. If x converges, go to step 7, otherwise go to 5.

7. Centering at $\bar{x}$, all points in the window with radius *r* in the HSI histogram will be treated as a class and taken off the image.

8. Repeat steps 3, 4, 5 and 6 until all the points in the HSI image have been classified.

9. After step 7, several classes are formed, each of which is an estimation vector $\bar{x}$ of hue, saturation and intensity. Each class also has a number of points that are classified to it. From the formed classes, drop the classes whose number of pixels is less than 5% of the total number of pixels. For each point in the dropped classes, reclassify it to one of the remaining classes which has the shortest Euclidean distance to this point.

In step 5, the formula calculates the shift vector $\Delta\bar{x}$ which points to the real estimation within the window which has radius ***r*** and centers at $\bar{x}$. When the shift vector $\Delta\bar{x}$ is calculated, $\bar{x}$ is updated. This process repeats until $\bar{x}$ converges. The resulting $\bar{x}$ is the real estimation within the window, and all the points within the result window that centers at $\bar{x}$ will be treated as one class. The classified point's corresponding pixels in the image will be removed from the image. This process is repeated until all the points in the image are classified.

2.4 Introduction to USGS Data

The United States Geographical Survey (USGS) was established in 1879 and has been collecting and updating geographical information for many years. This information can be obtained either free or at a low cost. In this research, two types of geographical data issued by USGS are used. They are Digital Line Graph (DLG) and Digital Orthophoto Qaundrangle (DOQ). Digital Line Graphs (DLGs) are digital vector representations of cartographic information derived from USGS maps and related sources. For most regions in the United States, they can be downloaded from http://www.usgs.gov. There are many layers of information in a DLG. Roads are just part of a DLG. DOQs are actually aerial photos with one meter per pixel resolution. The DOQ and DLG at a given location have been accurately matched.

In this research, DLGs are used to limit the area of image segmentation. Even though there are many existing road extraction algorithms which provide good results, USGS road information is very reliable and easy to access.

2.5 STS Road Database Formats

The STS RDB system was designed to make vehicle dynamics calculations efficient. Instead of using polygonal road descriptions, a combination of road center line points and cross sections was adopted because the cross section description could be reused at different points along the road. This scheme, similar to the method used in the MultiGen Creator Road Tool, is particularly efficient when roads are primarily composed of uniform cross sections.

Each uniform section can be defined by a cross section profile and a set of center line points. Curve road boundaries are modeled in the STS RDB system using distinct center line points and corresponding cross sections.

Identical cross section profiles can occur many times in a road network. Fig. 1 illustrates how the STS's RDB system handles this situation in an efficient manner by employing a pool of cross sections. Each road segment is spatially sampled along its center line. For each center line point, an entry in the cross section pool will be used to define the cross sectional view at this point. Entries in the pool could be used more than one time as in the case of the cross section designated as Cross Section Number 1 (see Fig. 1).

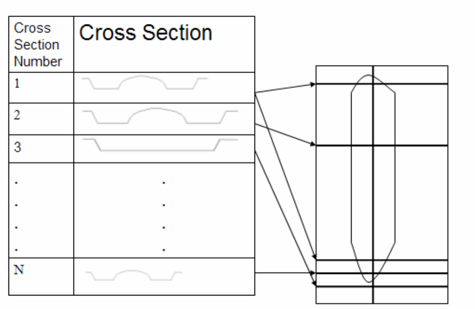

Fig. 1 STS Road Databases Requirements

During execution of the vehicle dynamics model at a specific location (centerline point) along the road, the appropriate cross section is referenced by the index associated with the centerline point. It follows that consecutive center line points along a straight line with the same cross section index is redundant. All occurrences of consecutive centerline points with identical cross section indices resulting from sampling should be replaced by a single occurrence. Compared with road polygon descriptions, memory requirements are reduced significantly, especially when there is elevation or irregular road boundaries to account for.

The process of forming a pool of cross sections is actually encoding them. In other words, similar cross sections should be identified and appear in road databases as a distinct integer.

CHAPTER THREE: EXPERIMENTS OF ROAD SEGMENTATION

Six methods for road image segmentation were introduced in the previous chapter. All of them have been implemented for road segmentation. The mean-shift clustering algorithm yields the best results and is adopted by the proposed system. In this chapter, the results of these algorithms are presented.

3.1 Edge Detection and Edge Linking

The Canny edge detection deals with how to extract edge information from an image, and it leaves the problem of interpreting these edges unsolved. In this research, the direction of Digital Line Graph (DLG) is first used to suppress irrelevant edge pixels, whose edge directions deviate from the DLG too much. In this research, if the following condition is met, the edge pixel is suppressed.

$$|\,\theta_e - \theta_{DLG}\,| >= \frac{\pi}{4} \tag{23}$$

where θ_e stands for the edge direction of an edge pixel and θ_{DLG} stands for the direction of a DLG line. This operation can remove some edges perpendicular to the direction of DLG, and also open closed contours. After this step, each edge is approximated using a cubic spline interpolation algorithm. By doing this, orientations of two edges can be compared. The similarity of orientations and closeness are the two most important factors in deciding whether two edges should be connected. Proper parameters need to be found to judge closeness and similar orientations. Based on our observation shown in Fig. 2 where the manual connection is recorded, edge pairs with short distance and similar orientation have a better chance of being correctly connected. Additionally, when the distance between two edges is the concern, the lengths of the two interested edges should also be considered. In Fig. 2, the red crosses indicate right connections, while the blue crosses indicate wrong connections. The horizontal axis stands for $\frac{D(i,j)}{L(i)+L(j)}$ and the vertical axis stands for $A(i,j)$, where $D(i,j)$ denotes the distance between edges i and j, $L(i)$ denotes the length of edge i and A(i,j) denotes the alignment of edges i and j. The line in Fig. 2 helps define a boundary for judging whether two edges should be connected or not. Numerically, if

$$A(i,j)+0.16\frac{D(i,j)}{L(i)+L(j)}-0.8<=0 \qquad (24)$$

edges i and j can be connected, otherwise they should not. Even though, there are many red crosses above the super-imposed line, it is not suggested to move this line upwards, because it will loosen the standards for connecting edges and cause false connecting. Some edges which are not qualified for being connected at the first time may become qualified at the second time, because they may have gotten longer and closer.

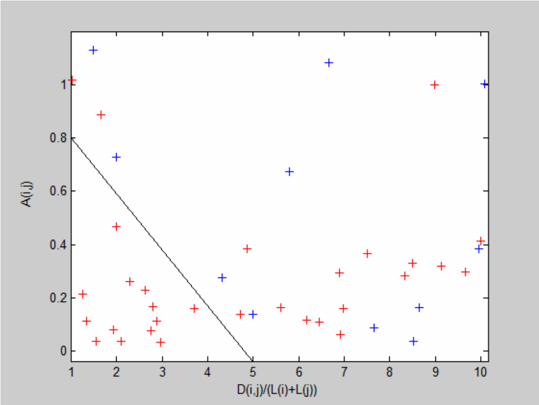

Fig. 2 The Boundary for Judge Right Connections

When the edges are grouped once, more connections could become possible, so two-time connection is suggested. Another simple rule is to connect two edges that have a same parallel partner, like Fig. 3 shows. In the first image, the upper two gray edges are approximately parallel with the bottom edge; therefore they should be connected due to the fact that they share the same parallel edge. In the second image, the two edges with a same parallel edge are at the bottom.

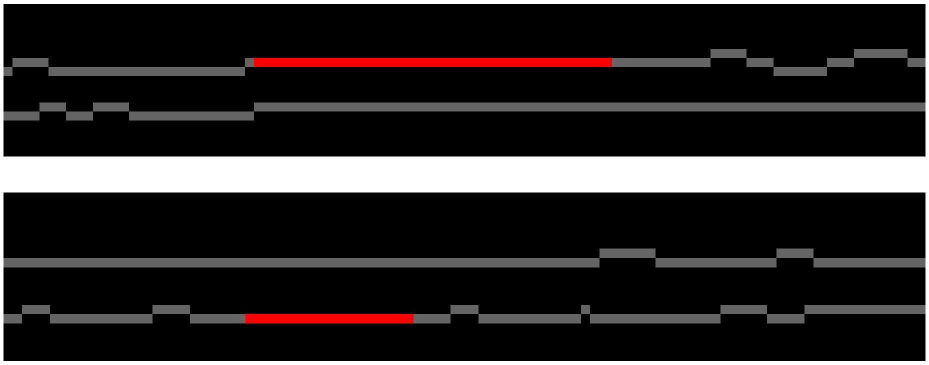

Fig. 3 Examples of Connecting Two Edges With Same Parallel Partner

The way to decide whether two edges are parallel is described as follows:

1) For a pixel in edge 1, draw a line that is perpendicular to the tangent. Tangents are approximated using cubic spline parameters.
2) If this line does not intersect edge 2, choose another pixel in edge 1 and go to step 1.
3) If the intersection in edge 2 has a similar tangent, this pair of points is regarded as a parallel pair.
4) If there are points that are unevaluated in edge 1, move to the next pixel and go to step 1.
5) If the number of the parallel pairs is less than 80% of the lengths of both edge 1 and edge 2, these two edges are not parallel; otherwise, they are parallel.

After the work of connecting is finished, this method will remove short edges, which are very likely caused by noises, such as traffic or pavement markings. Two sets of results of this algorithm are shown in the following. Each set has five images, which are the result of the Canny edge detection, the result of the filtering based on edge direction, the result of the first connection, result of the second connection, and the result of connecting edges with same parallel partners. Red edges are the results of connecting.

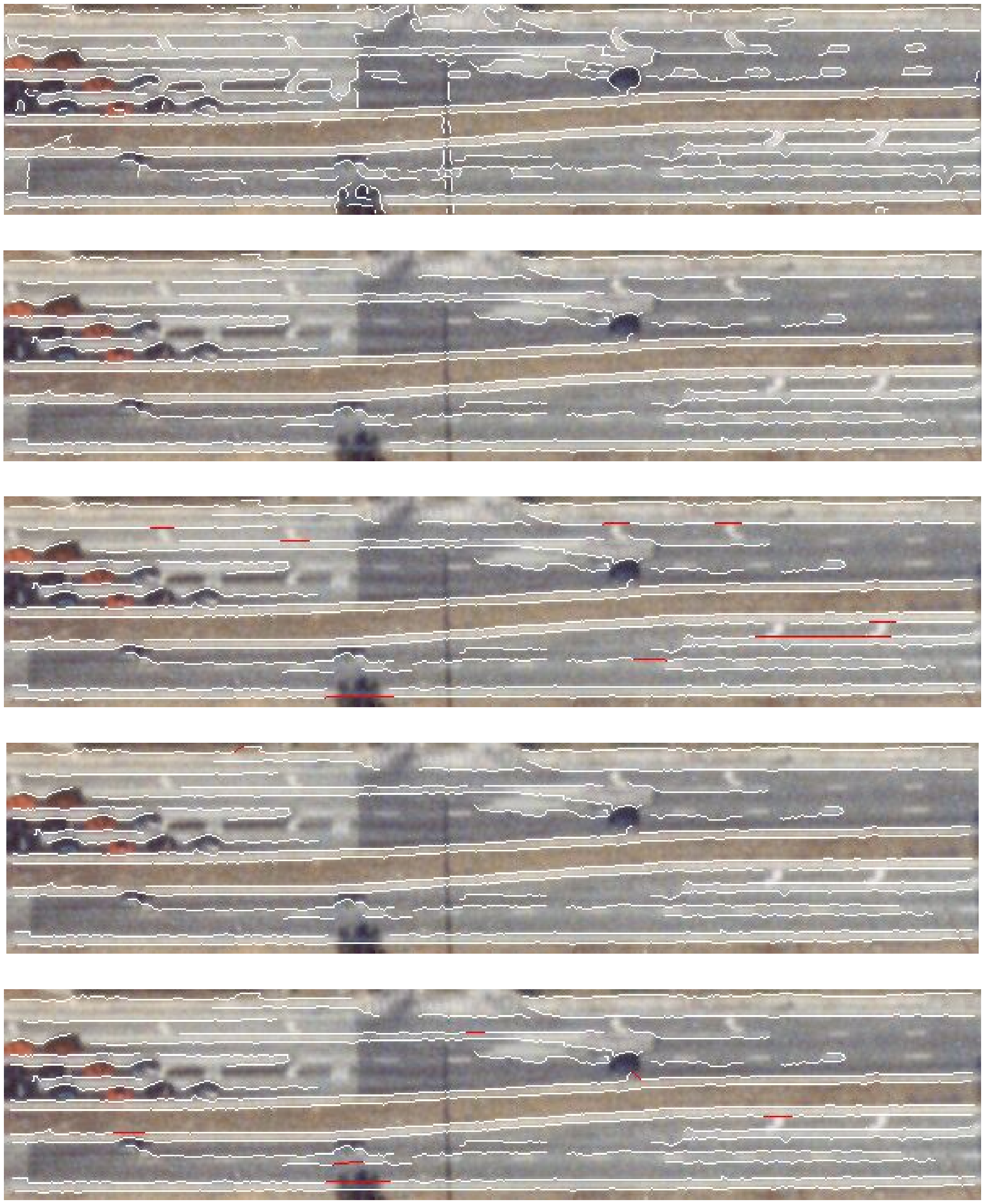

Fig. 4 Results of Edge Detection and Grouping (1)

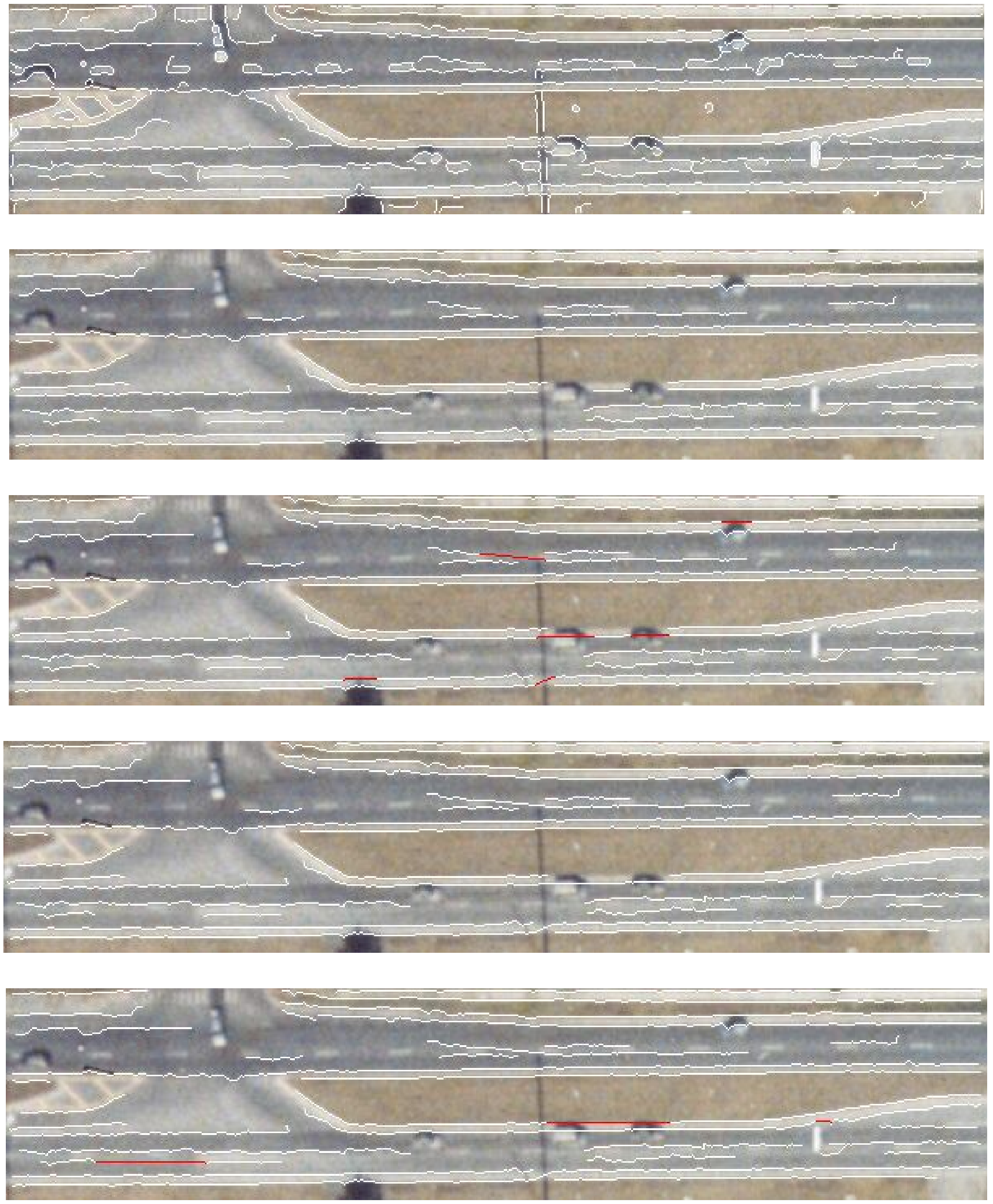

Fig. 5 Results of Edge Detection and Grouping (2)

When filtering edge points by edge direction, edges which are perpendicular to road center lines are removed. In the first connection, edges with similar orientation and close distance are connected. In the second connection, few edges get connected. This is the reason why it is suggested to connect edges no more than twice. At last, edges with a same parallel partner are connected. However, too many edges that are not road boundaries are preserved. The reason is that there are too many linear features in road areas. These noises could be traffic, pavement color discontinuities and so on. They are shown in Fig. 6.

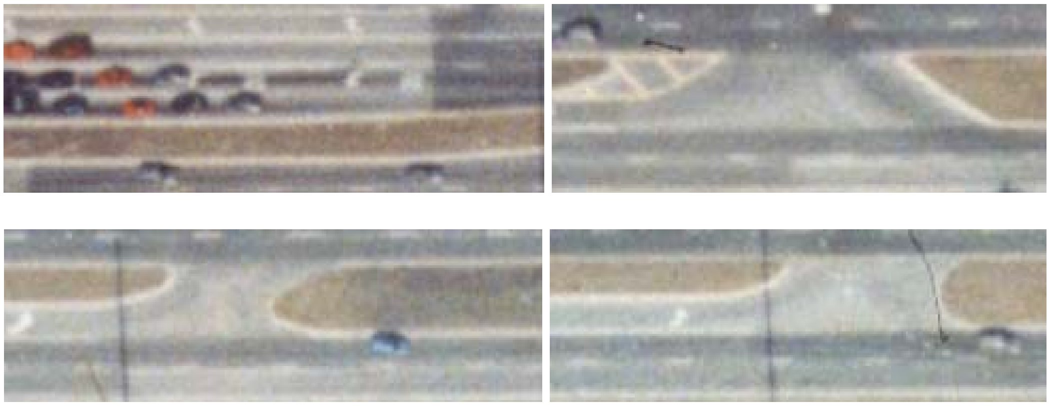

Fig. 6 Linear Features As Noises

This type of noises is very difficult to remove in an edge-detection based algorithm, because the lateral distances of these linear features are small. The edges, due to these noises, can also be very long, so length is not a good cue to separate noises in this situation.

3.2 Texture-Based Segmentation Using the Histogram Intersection Method

Texture-based methods focus on the pattern of the image. Therefore, it is good to select a cue whose pattern varies the most significantly between two textures. In [30], color ratio gradient is used as the cue. However, in our experiments, it was found that hue gradient is a better cue. In both Fig. 7 and 8, three images are demonstrated. The first one is the original image, the second one is the color ratio gradient image, and the third one is the hue gradient image. These images demonstrate that hue gradient separates two different areas, pavement and non-pavement, better than color ratio gradient does.

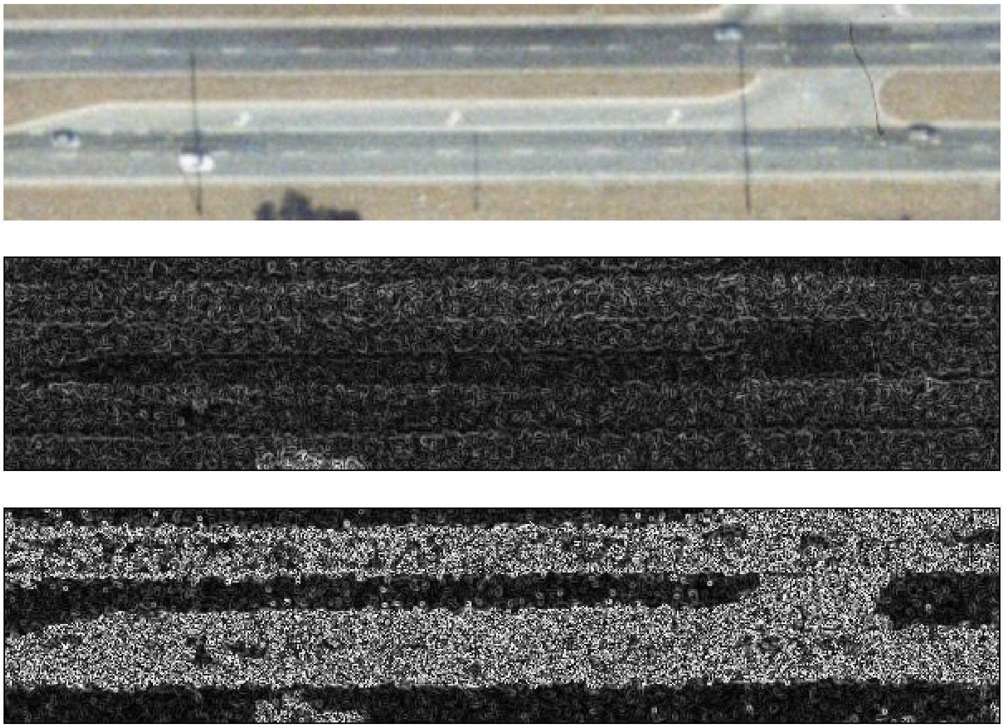

Fig. 7 Original, Color Ratio Gradient, and Hue Gradient Images (1)

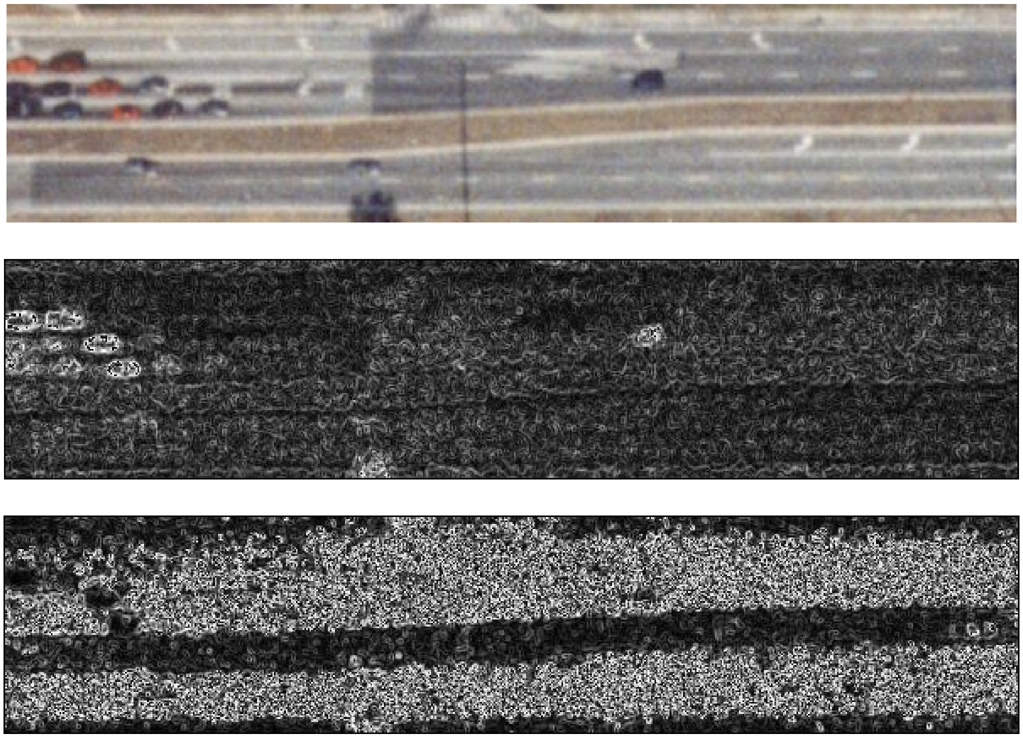

Fig. 8 Original, Color Ratio Gradient, and Hue Gradient Images (2)

As mentioned in section 2.2.2, sample images need to be chosen in order to segment an image into several classes. In this research, it is relatively simple, because a road image only needs to be segmented into two classes, which are pavement and non-pavement. Here are the sample images.

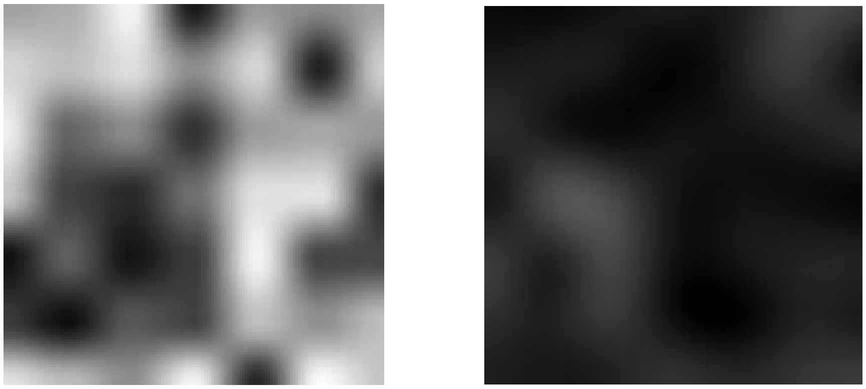

Fig. 9 Sample Images

The first image is the sample image for pavement and the second is for non-pavement. The results of the segmentation are shown below.

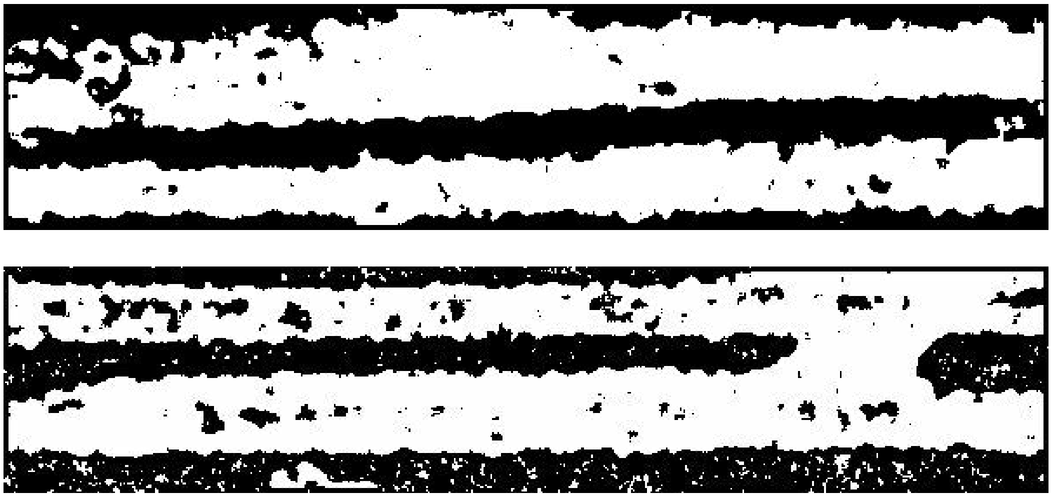

Fig. 10 Texture-Based Segmentation Using the Histogram Intersection Method

The segmentation results in Fig. 10 give strong indications where road and median boundaries are likely to exist. A simple procedure has been defined to combine the results in section 3.1 to generate cleaner edge images and nicer road delineations. This method is performed to see whether an edge is surrounded by pixels with the same classification. If it is true, that means this edge is very likely caused by noises. Our threshold is that if a 7-pixel wide strip that centers at an edge has either more than 70% pixels classified to pavement, or more than 70% pixels classified to non-pavement, this edge will be eliminated. There is a problem with the threshold, which is that it eliminates too many edges. So this method also defines that if an edge's closest parallel partner is preserved, this edge will survive as well. However, this cannot get rid of all the wrong delineations, especially those caused by the lateral discontinuity of pavement.

The fuse of segmentation and edge detection needs a certain level of intelligence. Several rules are defined to make the delineations better. Before listing the rules, the concept of area will be introduced. In this research, an area is defined to be within two edges, but without an edge point inside. The purpose of defining areas is to remove part of some edges, because it is obviously if areas on both sides of an edge have the same classification, this edge should be removed. Table 1 lists all the rules and Fig. 11 lists the improved edge images.

Table 1 Rules for Fusing Edge Grouping Results and Texture-Based Segmentation Results

Rule	Content
1	If more than 70% of area pixels are classified to pavement/non-pavement, this area is classified as pavement/non-pavement.
2	If two areas share the same boundaries (edge/marker), these two area should be classified the same.
3	If an area has two neighbors with the same classification, this area should also have the same classification.
4	If an area is small (<1000) and narrow (<20), this area should be classified to be pavement, because this area is caused by vehicles.
5	If a part of an edge or a marker is clamped by two areas with the same classification, this part should be removed.

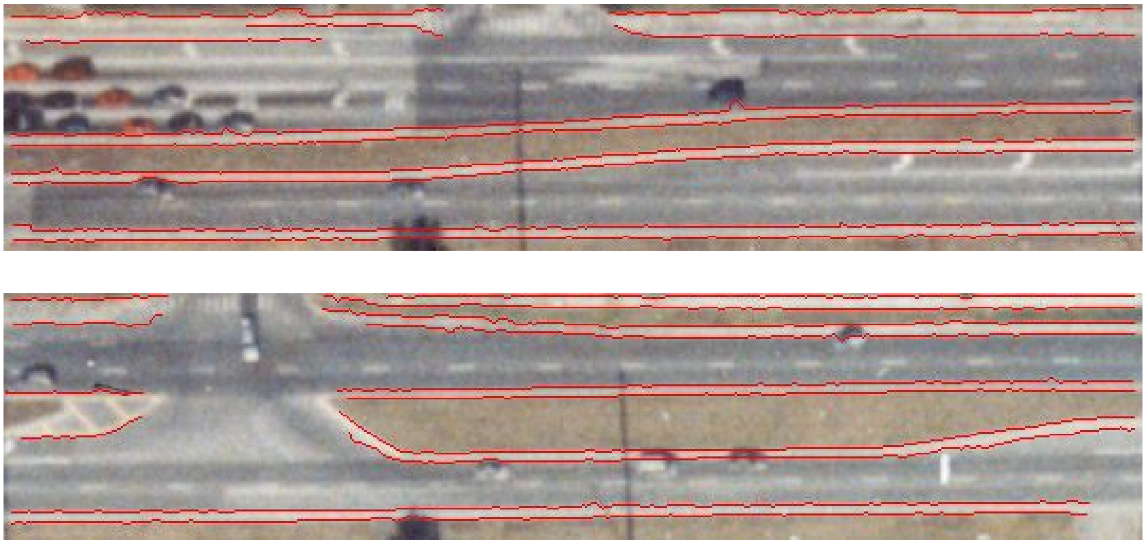

Fig. 11 Improved Edge Grouping

A serious problem with this histogram-based method is that it needs good sample images, which must be statistically close to test images. When this condition is not met, segmentation results will be significantly affected as follows:

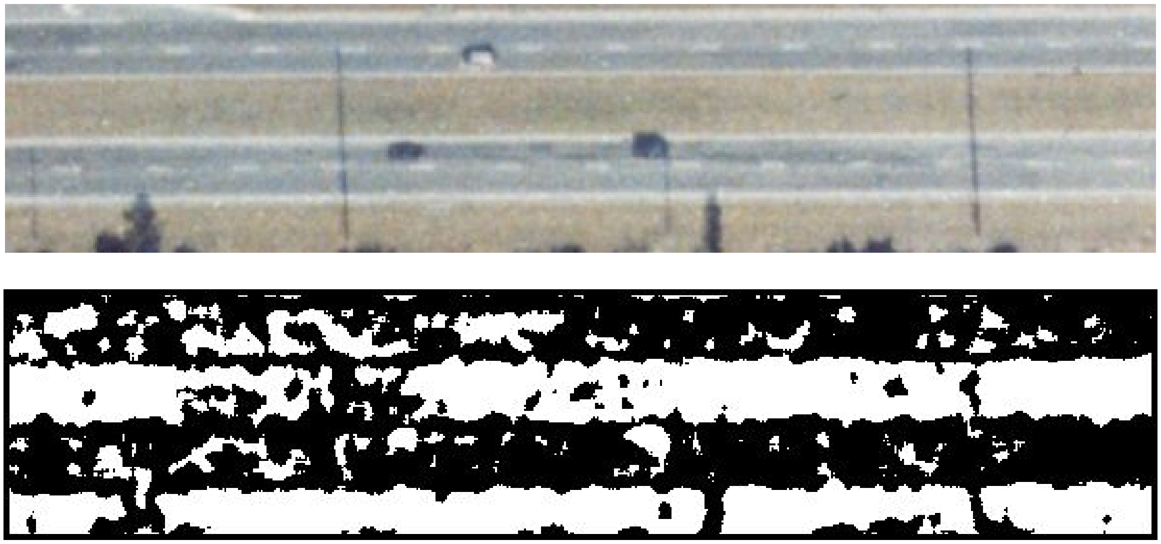

Fig. 12 Example of Failing Texture-Based Segmentation (1)

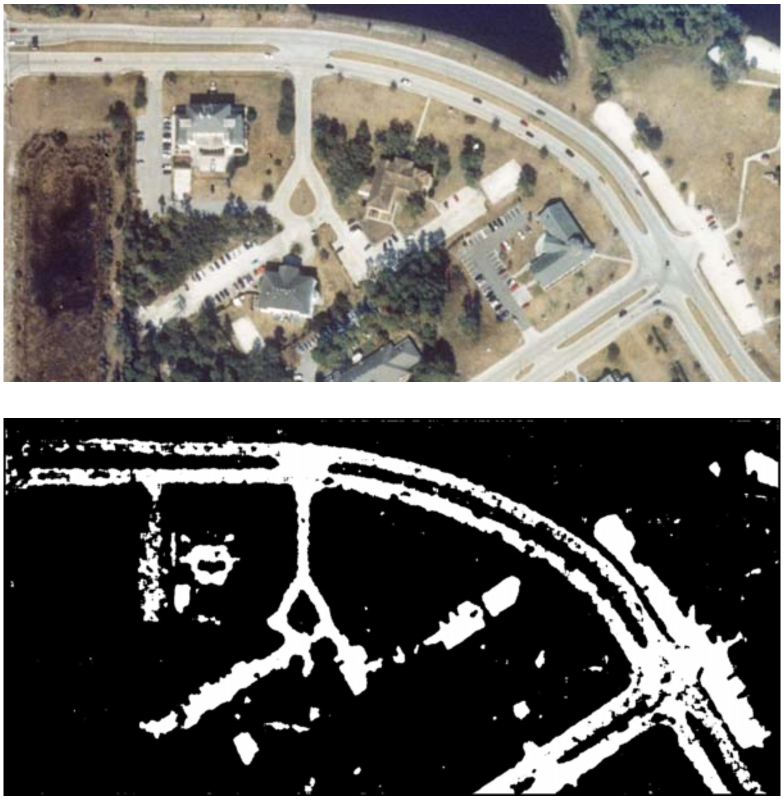

Fig. 13 Example of Failing Texture-Based Segmentation (2)

The failures are due to the heterogeneity of road appearances. When resolution of an aerial image is high, this phenomenon must be addressed. It is possible to build a database with a large number of sample images. When segmenting an image, all of them are compared with each pixel's neighbor in the test image by following the histogram intersection algorithm. The problem with this solution is twofold. The first is that road appearance in one place may look very different from that in another place. In such a case, this method is not reliable. The second is just simply too costly to build a database with many sample images.

3.3 Feature-Based Image Segmentation Using EM Method

In [26], the first step is to convert an image to its representation in a feature space. The features, considered in [26], are categorized into two types. One is based on texture and the other is based on color. The color-based features are $(si\cos(2\pi h), si\sin(2\pi h), v)$, where h, s and i are respectively hue, saturation and intensity. However, in our experiments, the color features, (h, s, i), work better. The texture-based features at a certain pixel are obtained by the following algorithm.

1. For each $\delta_k = k/2$, where $k = 1,2,....,7$, perform the following the calculations
   a. $M_{\delta}(x, y) = G_{\delta}(x, y) * (\nabla I)(\nabla I)^T$, where $G_{\delta}(x, y)$ is a Gaussian kernel and ∇I is the gradient.
   b. λ_1 and λ_2 are the two eigenvalues of $M_{\delta}(x, y)$ and $\lambda_1 > \lambda_2$
   c. $E_+ = \sum_{(x,y)\in\Omega} G_{\delta}(x, y)[\nabla I \hat{n}]_+$, where $\hat{n}$ is the unit direction of λ_1's eigenvector, Ω represents the neighborhood under consideration, and $[\nabla I \hat{n}]_+$ stands for positive value of the product of ∇I and $\hat{n}$.
   d. $E_- = \sum_{(x,y)\in\Omega} G_{\delta}(x, y)[\nabla I \hat{n}]_-$, where $[\nabla I \hat{n}]_-$ stands for negative value of the product of ∇I and $\hat{n}$.
   e. $p_k = \dfrac{|E_+ - E_-|}{|E_+ + E_-|}$

2. Choose δ_k, when $| p_k - p_{k-1} |< 0.02$ as the scale for calculating the polarity for a certain pixel.

Above is the calculation of polarity. Step a is straight forward. In step b, when λ_1 is larger than λ_2, λ_1's corresponding eigenvector reflects the dominant orientation in the neighborhood. When the two eigenvalues are similar, no preferred orientation exists. When they are negligible, the local neighborhood is approximately constant. Steps c and d calculate the positive and negative part of a gradient vector, ∇I, on $\hat{n}$ the direction of λ_1's eigenvector. Step e calculates the polarity. When p is close to 1, an edge may exist. When p decays with δ_k, there may be a 2D texture that has multiple orientations. When p takes arbitrary values, the local neighborhood may have a constant intensity. Polarity also depends on the size of the local neighborhood, so the size must be selected as polarity converges. Afterwards, the anisotropy and the normalized texture contrast for a certain pixel are calculated as follows

$$a = 1 - \frac{\lambda_2}{\lambda_1} \tag{25}$$

$$c = 2\sqrt{\lambda_1 + \lambda_2}^{3} \tag{26}$$

a represents anisotropy and c represents normalized texture contrast. Finally, the texture-based feature vector is (ac, pc, c). In the experiments, the texture-based features do not demonstrate a strong pattern. However, sometimes they greatly contribute to the segmentation. The following sets of images show the results of the experiments. Each set has six images that are the original image, the polarity image, the texture feature images, (ac, pc, c), and the segmentation result.

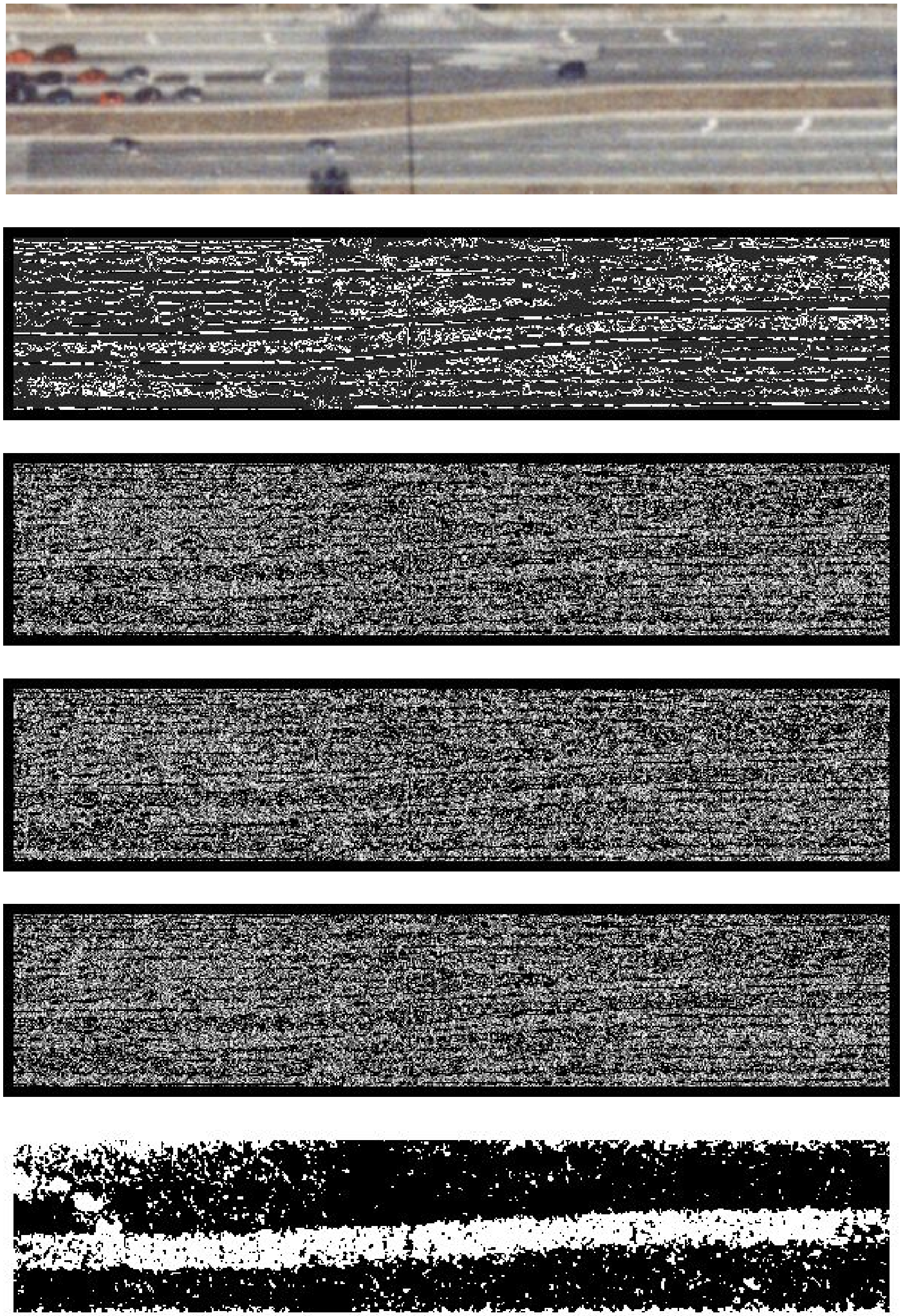

Fig. 14 Results of Feature-Based Segmentation Using EM Method (1)

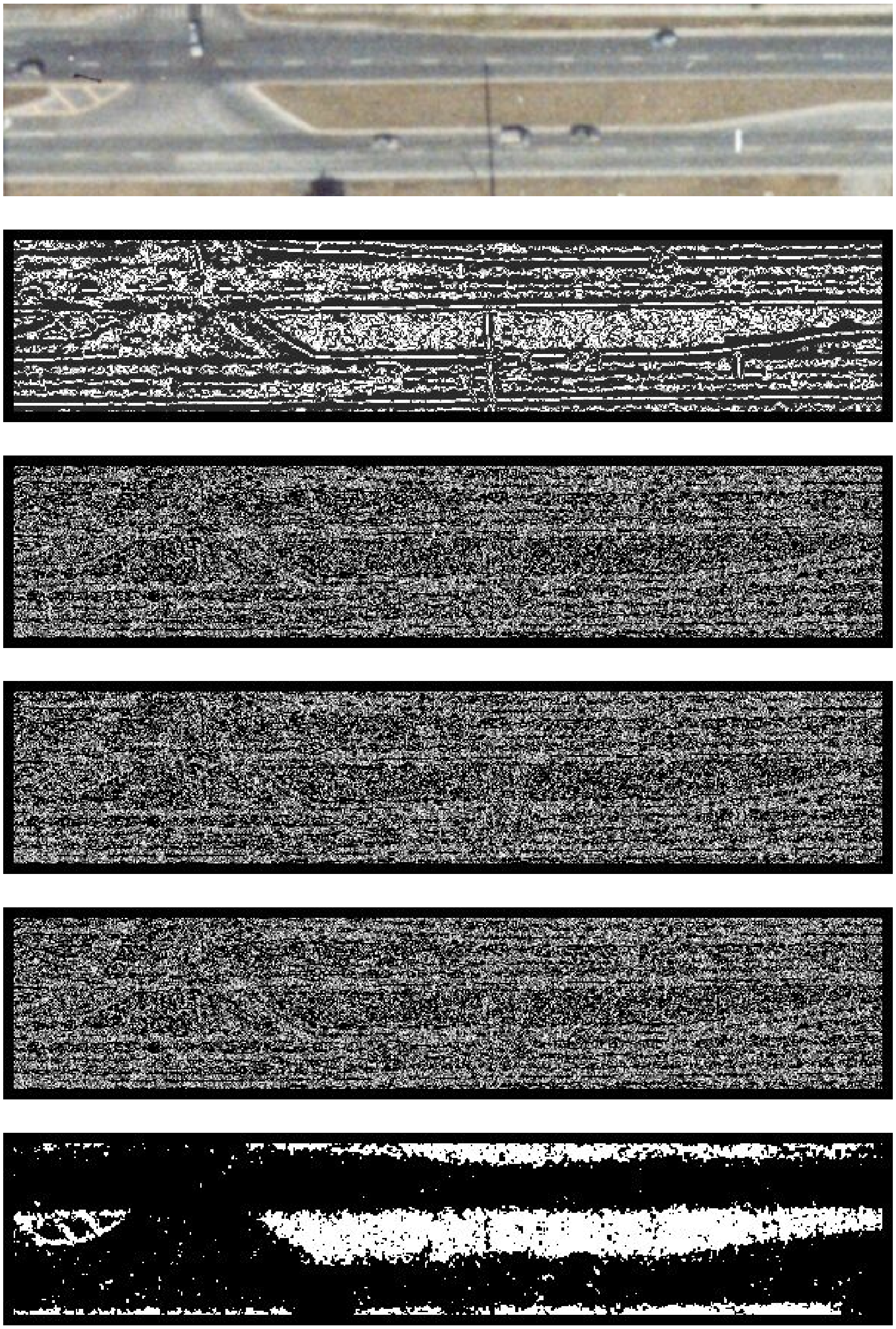

Fig. 15 Results of Feature-Based Segmentation Using EM Method (2)

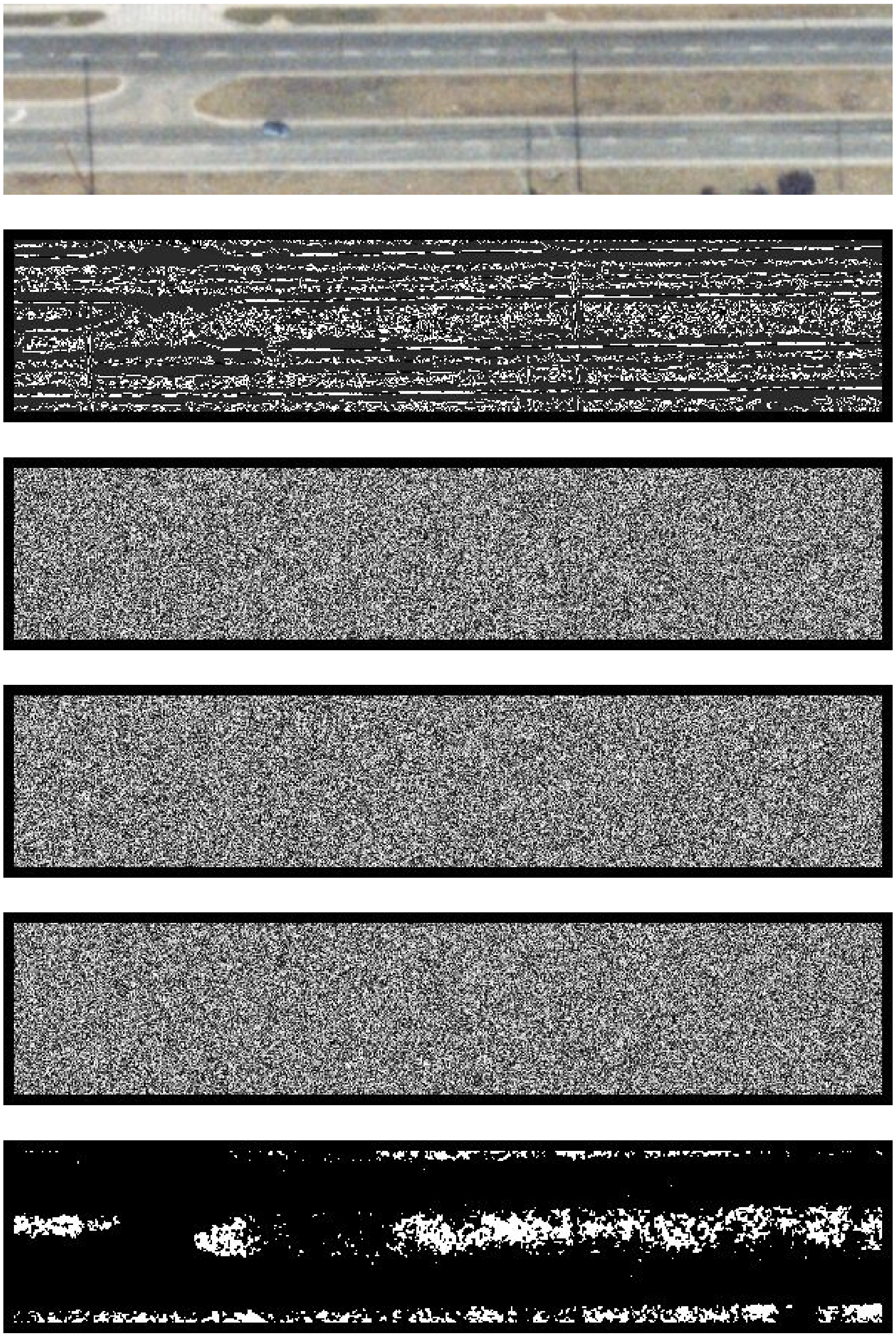

Fig. 16 Results of Feature-Based Segmentation Using EM Method (3)

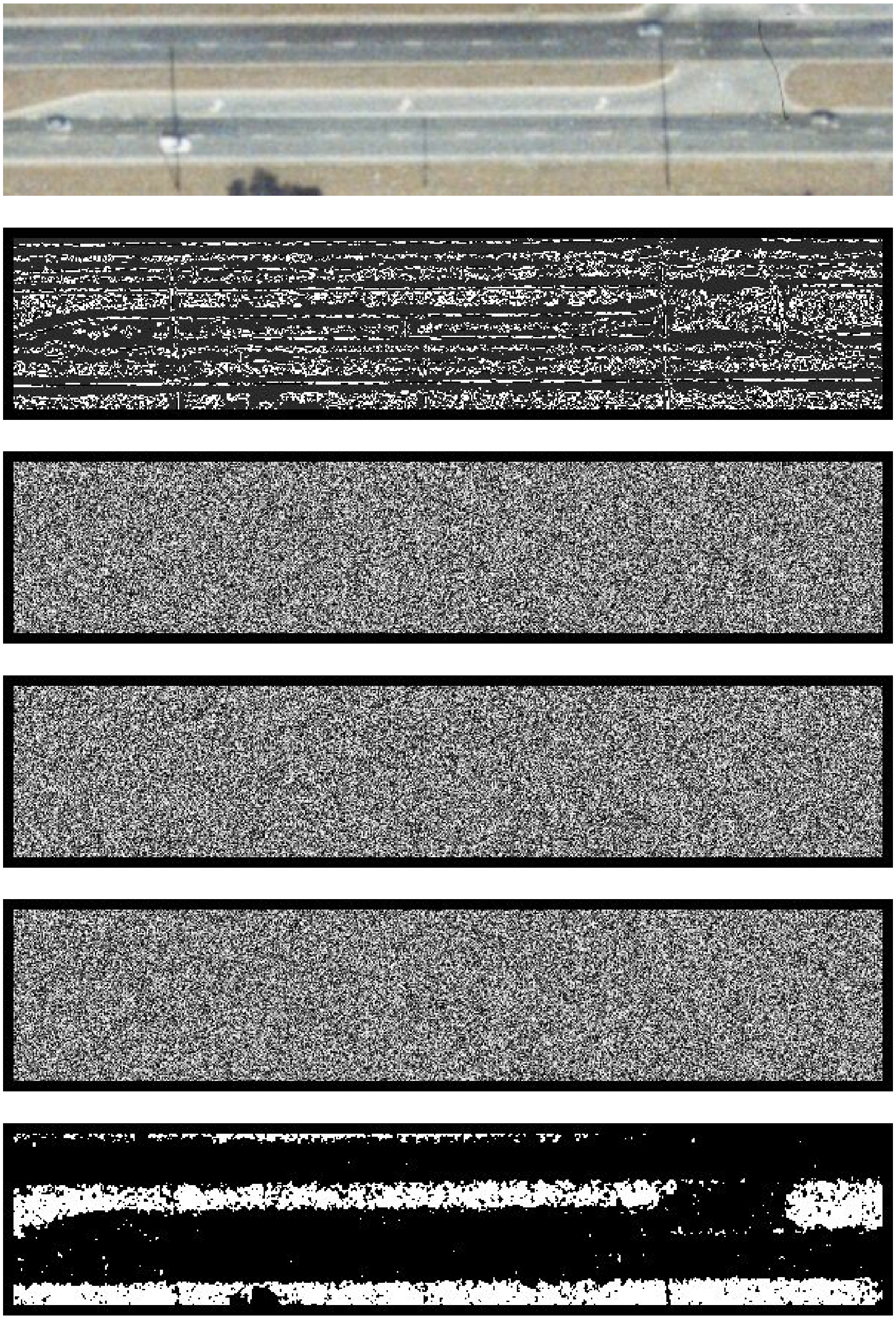

Fig. 17 Results of Feature-Based Segmentation Using EM Method (4)

In the polarity images, it is easy to see that the most salient contents in the road images are edges and constant areas, because many pixels have high and relatively high gray values in the polarity images. A few pixels having low gray values are textured area.

The visualized texture-based features are stochastic, instead of being informative. One possible reason for this phenomenon is that only gray level gradient values in the test images are considered. It may also be because the road images do not have strong textures at all. But this does not seem correct, because in the previous sub-section, the hue gradient based segmentation method provided good results when texture was the only cue for segmenting. This is interesting and is left to future research.

The results are more accurate than the results yielded by the method mentioned in the previous section. One advantage of this method is that it does not need sample images; therefore there is not any bias in selecting sample images and the cost of building a database of them.

There is another special point when applying this method for road segmentation. The initial mean vectors and covariance matrices are calculated for two halves of the image in [26]. But in this case, not all the image area is of interest, but the area along the DLGs. So for this research, the image area is defined by the 140 pixel wide stripe along the DLGs. When initializing mean vectors and covariance matrices, the first step is to divide this stripe into two halves, both of which have the same length. Then each half's mean vector and covariance matrix are calculated accordingly.

The reason why this method was not chosen as the final solution to road segmentation is because this method has to do many complex calculations, such as solving eigenvalues and eigenvectors, inverting matrices, and solving a matrix's determinant. This makes this method very difficult to be integrated into an interactive software application.

3.4 Region-Growing Image Segmentation

The region-growing segmentation method, tried in this research, is based on [25]. In this method, each pixel is treated as an object at the beginning. Then similarities (degree of fitting) among neighbor pixels are measured. Several ways of calculating degree of fitting are suggested in 2.2.4. A slightly customized formula has been used in the experiments and is $h = 1 - \sqrt{\sum_d \left(\frac{f_{1d} - f_{2d}}{(f_{1d} + f_{2d})} \right)^2}$, where f_{1d} and f_{2d} are the mean feature values of two objects. The advantage of this formula is that the higher value of h implies two objects are more likely to be merged and its value falls in a range of (0,1). If the calculated similarity is greater than a predefined threshold, these two pixels are combined into one object.

The process of merging objects is very computation intensive. If merging starts with a certain pixel, it only needs to check its neighbors; however, its neighbors can bring more pixels. This procedure repeats until no more similar pixel is adjacent. If a depth-first method is used, the complexity of forming one object is $O(n^2)$. This process has to be repeated for each pixel. Therefore, the complexity of forming all the objects is $O(n^4)$. After forming all the objects, the same process needs to be repeated and the difference is that each object is now larger. If a uniform area is of interest, it takes a shorter time to finish, because an object can be big and the number of objects is small. Another practical issue with this method is that forming an object from single-pixel objects is a recursive procedure as shown below:

```
grow(pixel p)
{
    for(each neighbor n of p)
    {
        if(n is similar to p)
        {
            n is brought to the object which p belongs to
            grow(n)
        }
    }
}
```

This recursive function may overflow the function stack in the operating system, if an area is relatively uniform. This is because a uniform area will make this recursive function run to too many levels. Therefore, a limit must be set ahead of time to control how many levels of recursion are allowed to run. In this experiment, the maximal number of recursion is 5000. This number may be enough for a complex and small image. If an image is large, this number may not be enough to form an actual object at a time; instead it may form several objects and merge them later.

Due to the intensive computation, this method is not recommended for road segmentation. It is hard to be integrated into an interactive software application. This method can be adopted for other purposes, because it is easy to understand and implement. Below some results are shown together with the original images. After the region-growing segmentation is finished, each object is represented by its average RGB. The results are color images.

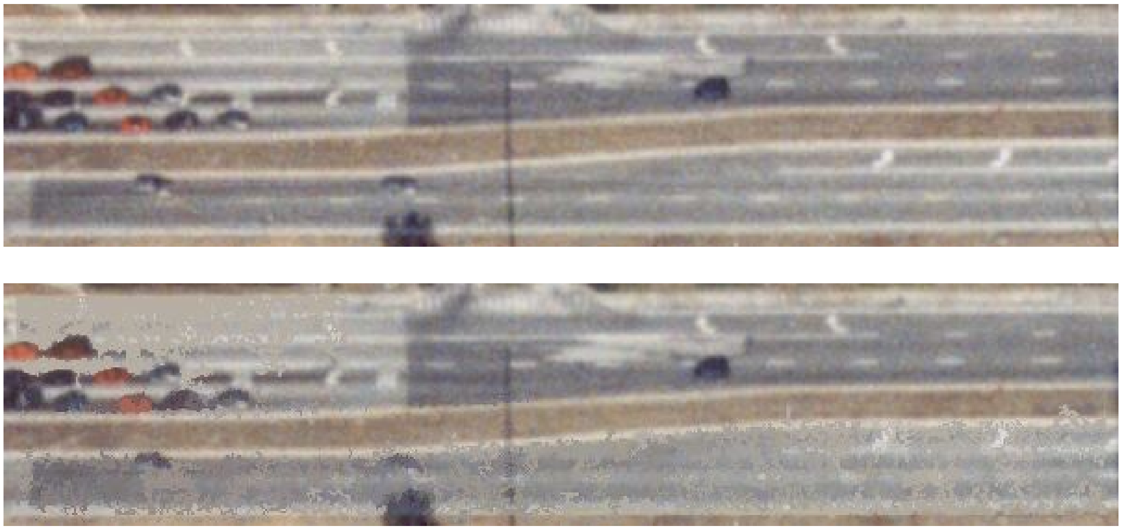

Fig. 18 Results of Region-Growing Method (1)

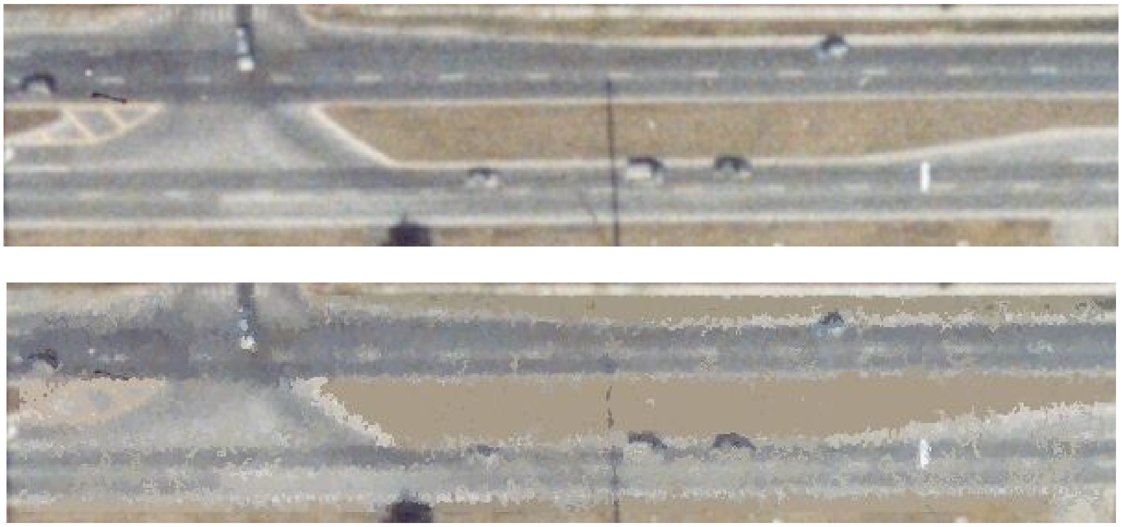

Fig. 19 Results of Region-Growing Method (2)

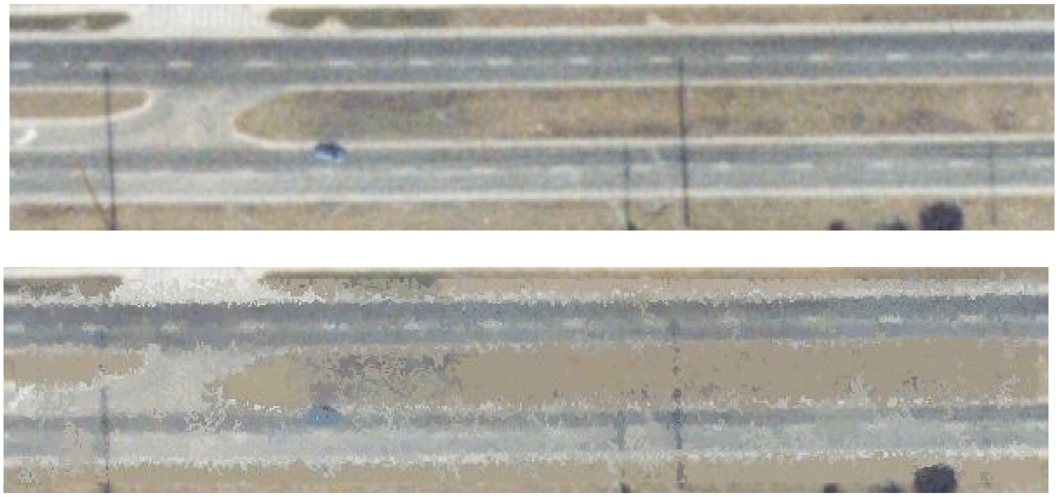

Fig. 20 Results of Region-Growing Method (3)

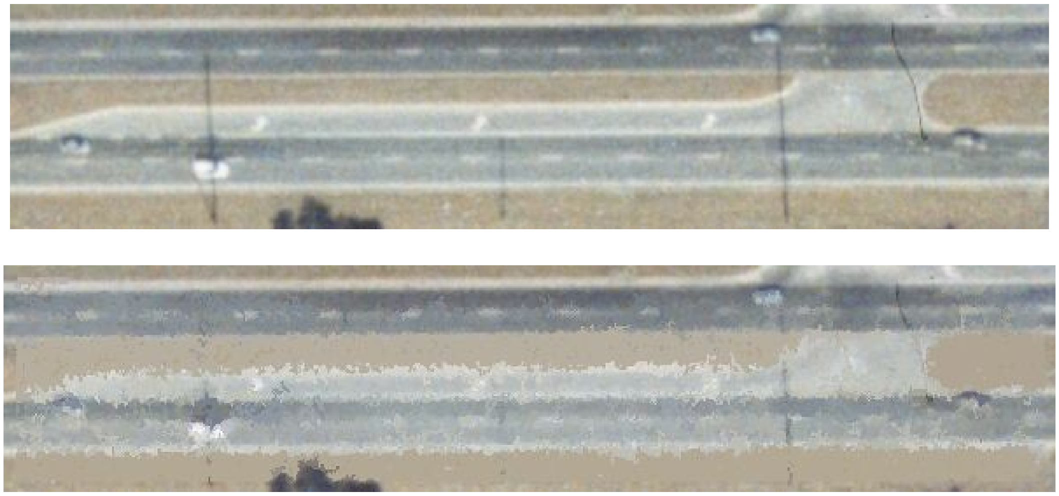

Fig. 21 Results of Region-Growing Method (4)

The results in the previous four examples give accurate results. Each object in the segmentation result is relatively uniform in the original image, which is the main goal of this algorithm. However, the interpretation of these objects is not straightforward, because located on roads there are many irregular shapes which are hard to correctly identify. Another limit with this algorithm is that two similar objects cannot be merged into one object if they are not adjacent.

That is because spatial relationships between two objects play an important role in region-growing segmentation methods. Only adjacent objects are considered for merging. This algorithm does not consider the distance within the feature space. If two objects are formed in two separate places in the image, they cannot be merged, even if they are close in the feature space.

3.5 Growing Active Contour Model

In high-resolution road images, road boundaries are not the only linear objects. Many other objects, such as pavement markings, also appear linear. Therefore, it is difficult to automatically choose initial points to fit in an active contour model. For this research, the active contour-like method, proposed in [27], is customized to grow a linear object from its two ends.

In this customized version, for each end point, the energy of each neighboring point is calculated. The one with the smallest energy is where the contour is going to grow. If any point’s energy is greater than a predefined threshold, it means the extension of the contour should be stopped. This algorithm is demonstrated as follows:

```
Initialize $\alpha$, $\beta$, $\theta$, and $\gamma$
terminate=0
end[0]=first end point of the contour
end[1]=second end point of the contour
/*loop to move points to new locations*/
for i=0 to 1                    /*grow the contour twice due to two ends*/
    while not terminate
        $E_{\min}$=BIG
        for j=0 to m-1          /*m is size of neighborhood*/
            $E_j = \alpha E_1^j + \beta E_2^j - \theta E_3^j + \gamma E_4^j$
            if $E_j < E_{\min}$
```

```
                    $E_{\min} = E_j$
                    $j\min = j$
                endif
            endfor
            if $E_{j\min} > E_{\max}$
                terminate=1
            else
                grow the contour to point j
                end[i]=point j
            endif
        endwhile
    endfor
```

The results are shown in the following figures, where red edges are grown edges and white edges are detected by the Canny edge detection method.

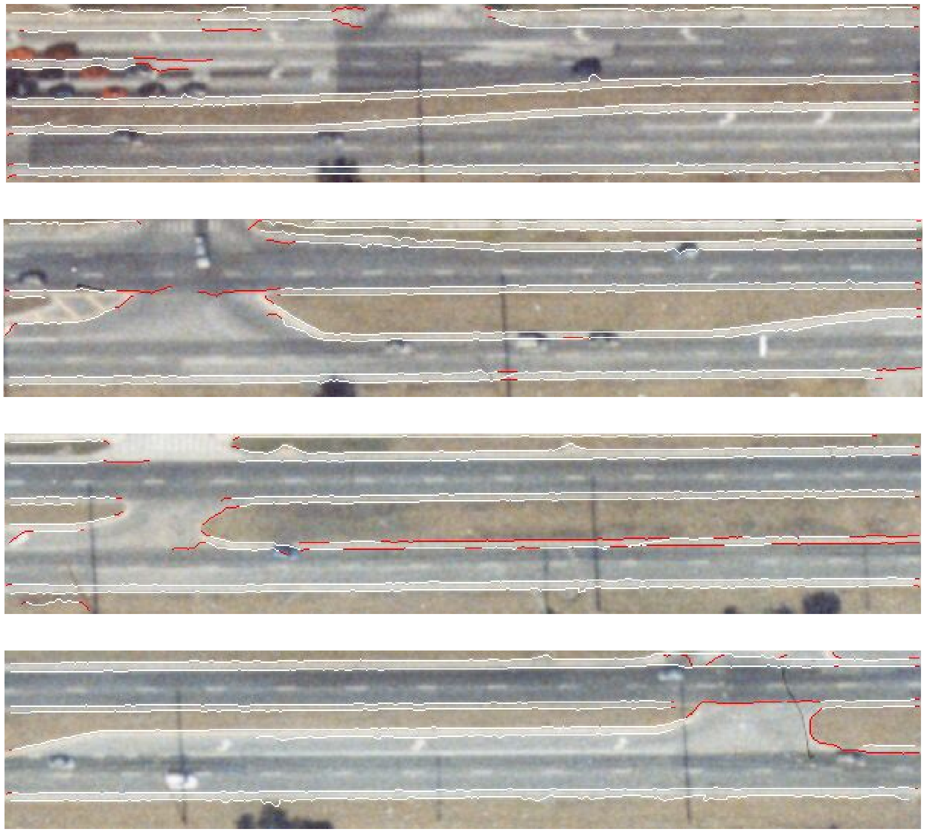

Fig. 22 Results of An Active Contour Method

The active contour method in this section grows each object from two ends. The growing follows the linear contours and is smooth. Additionally, the growing stops where the edges are weak. However, there are too many incidents of false growing. This is obvious in the second and third images, where objects near the ends of the medians extend to the pavement area. This is inevitable to this algorithm due to lateral pavement discontinuities. These discontinuities cause adjacent edges to have the same orientation. In such a case, it is not enough to only consider gradient-related factors as the image force. But one of the functionalities of active contours is to delineate objects with high gradient values. Therefore, active contour based methods are not suitable for road segmentation in high resolution images.

However, active contour based methods can be useful when developing an interactive application for road modeling. Modelers can give relatively accurate initial points around an object, such as a median.

3.6 Mean-Shift Clustering Algorithm

The mean-shift clustering algorithm segments image pixels within a feature space by searching local modes. In the following results, the first images are the original images and the second are the segmentation results. Segmented objects shown in the following images are represented by their average RGBs.

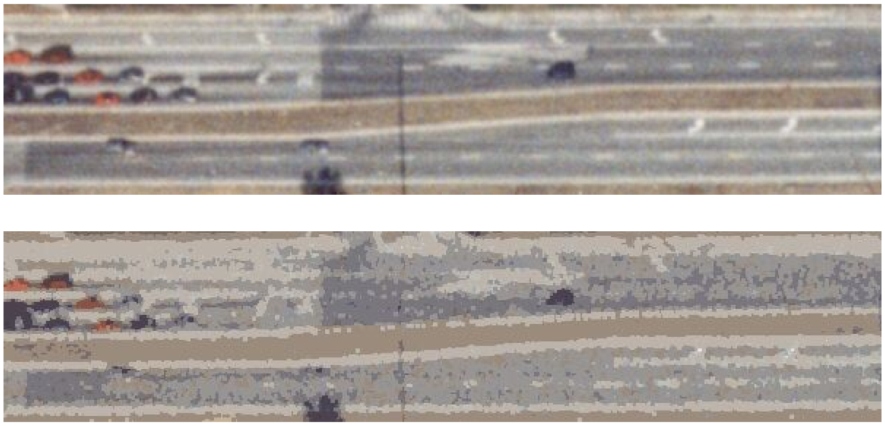

Fig. 23 Result of the Mean-Shift Clustering Algorithm (1)

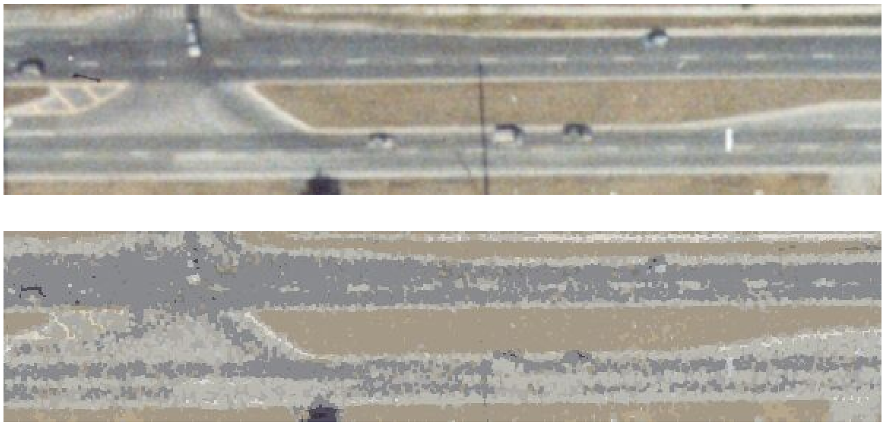

Fig. 24 Result of the Mean-Shift Clustering Algorithm (2)

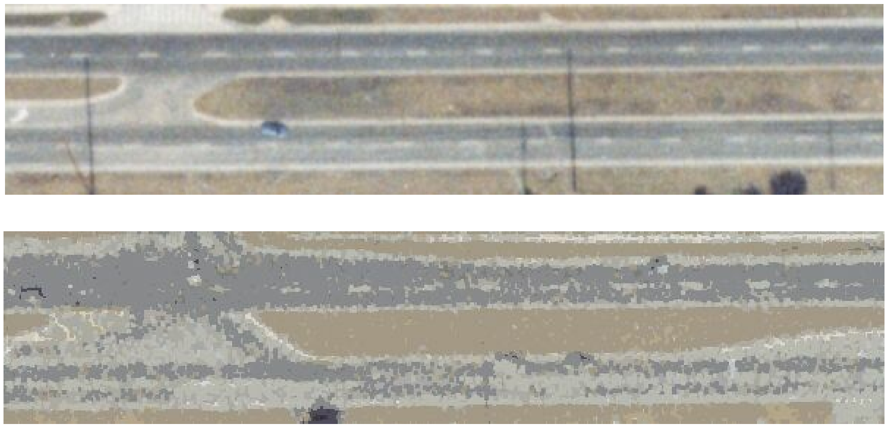

Fig. 25 Result of the Mean-Shift Clustering Algorithm (3)

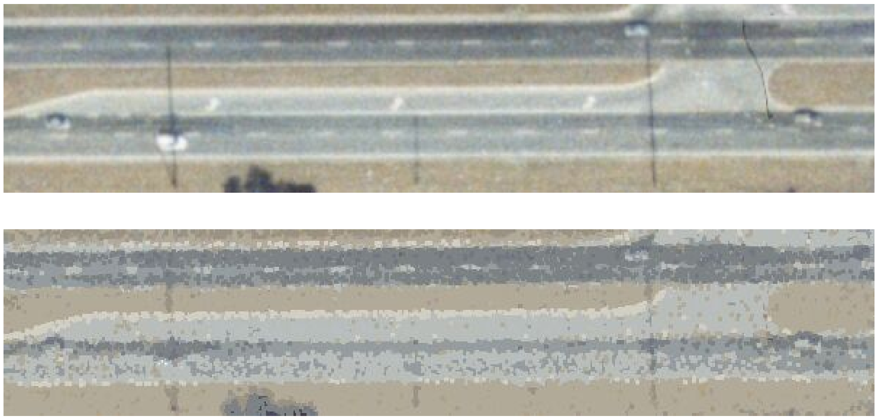

Fig. 26 Result of the Mean-Shift Clustering Algorithm (4)

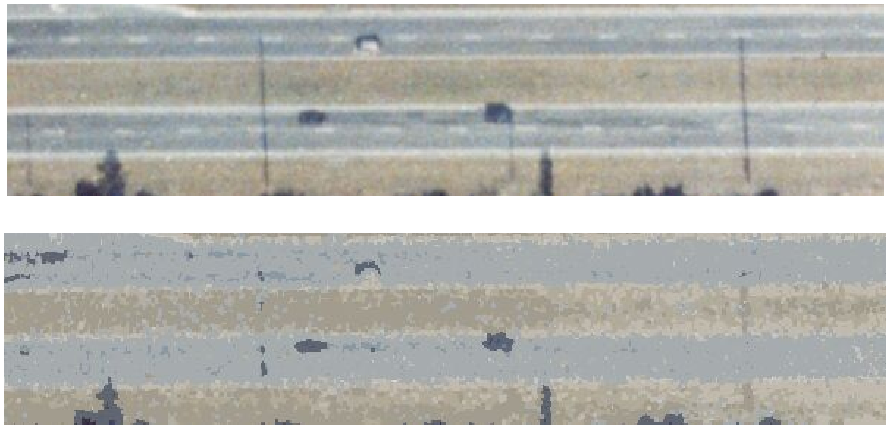

Fig. 27 Result of the Mean-Shift Clustering Algorithm (5)

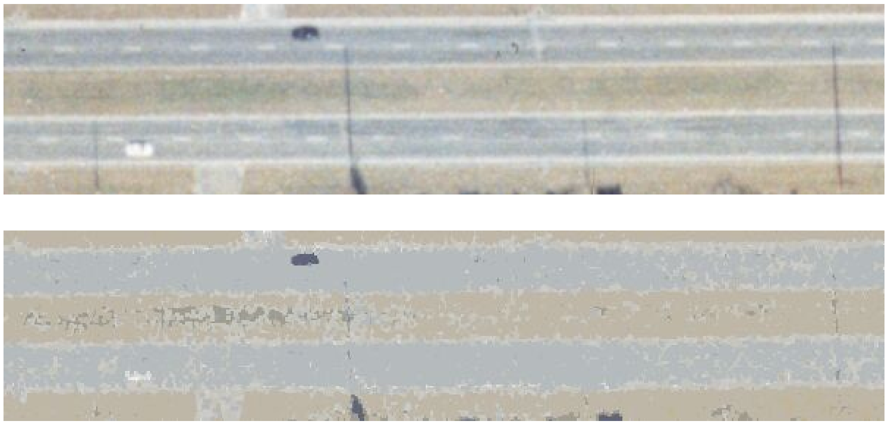

Fig. 28 Result of the Mean-Shift Clustering Algorithm (6)

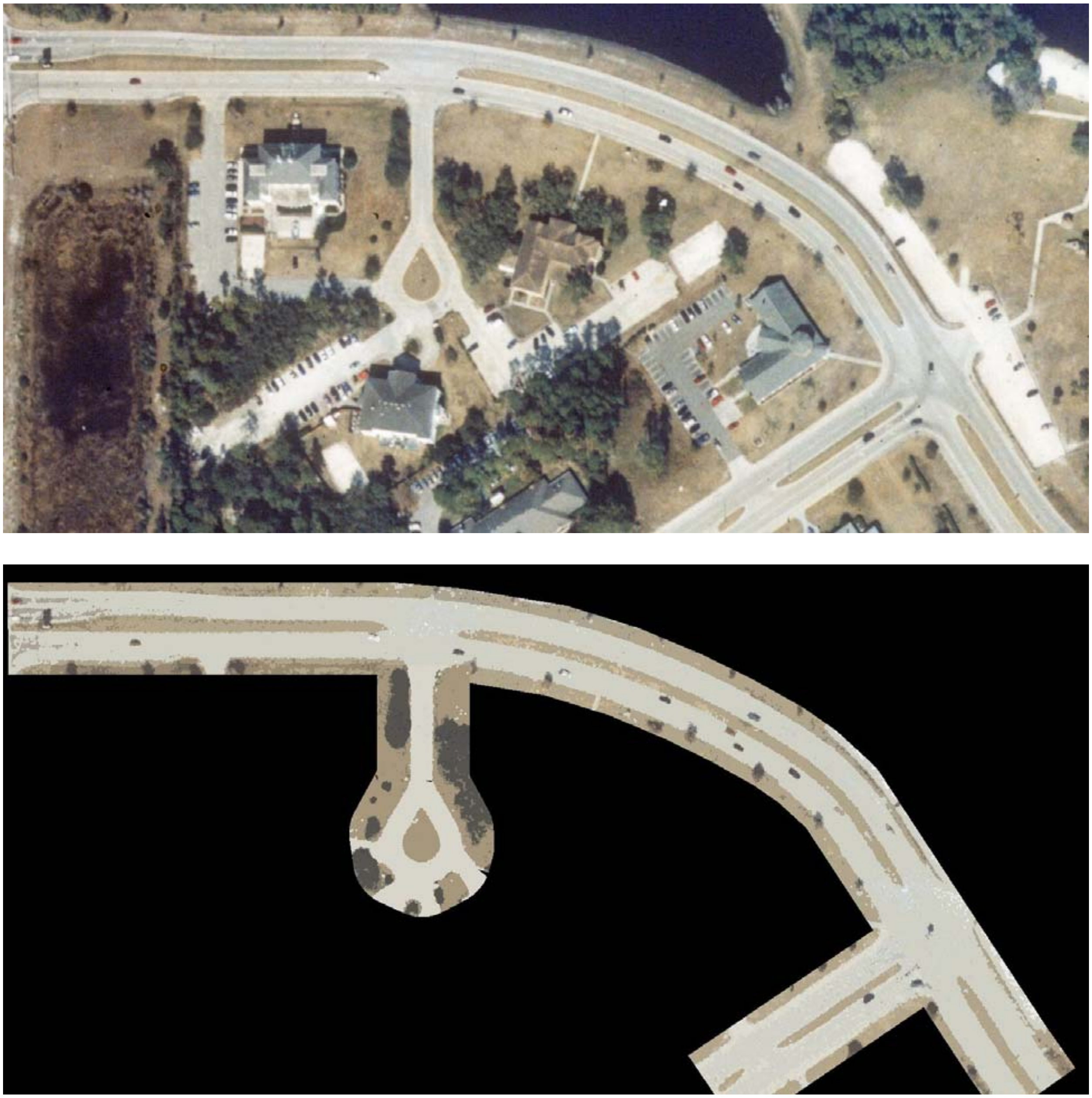

Fig. 29 Result of the Mean-Shift Clustering Algorithm (7)

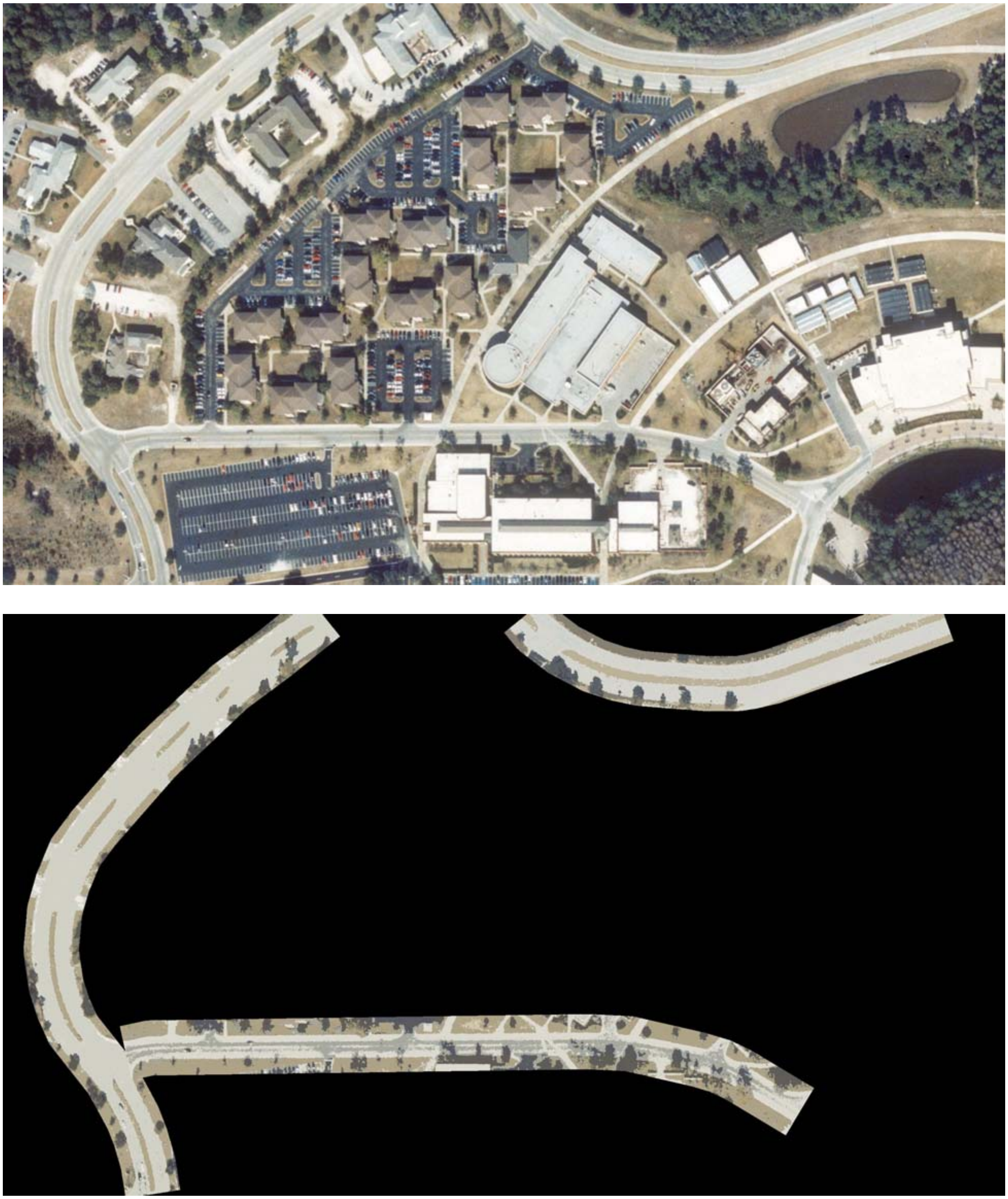

Fig. 30 Result of the Mean-Shift Clustering Algorithm (8)

These segmentation results yields accurate results. A contributing factor is that the mean-shift algorithm can find large objects prior to small objects. This can be illustrated by sizes of extracted objects in the order they are extracted.

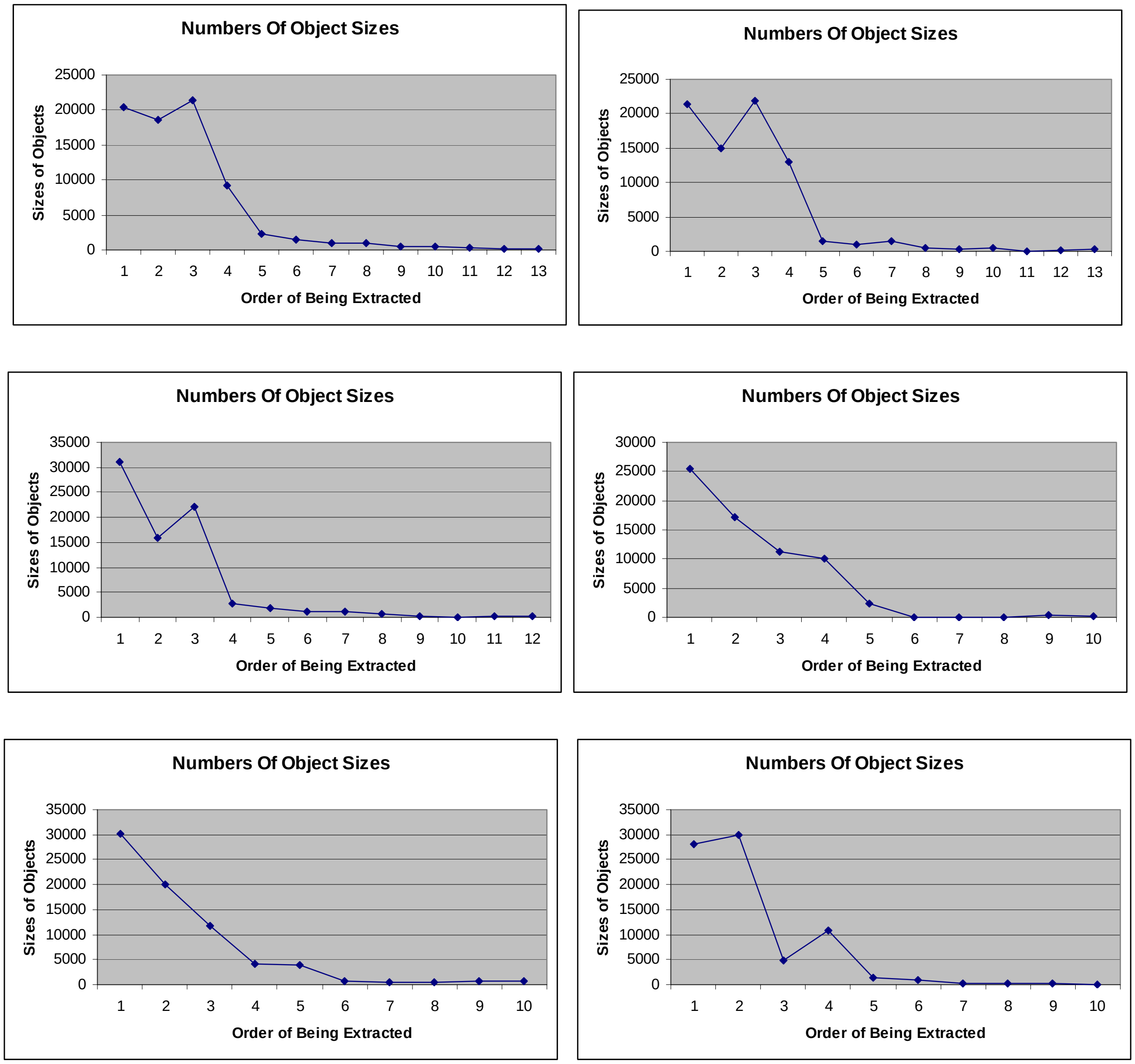

Fig. 31 Sizes of Extracted Objects

CHAPTER FOUR: METHODOLOGY

4.1 Matching DLG Information with Aerial Photos

Since the aerial images in this research and the digital road information (DLG from USGS) are from different sources, they need to be matched accurately before the recognition process begins. Two types of distortions between an image and the USGS data are considered in this research. One is linear and the other is non-linear. Fig. 32 shows the two kinds of distortions.

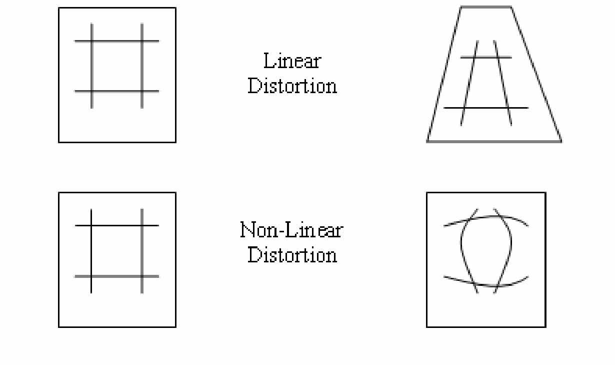

Fig. 32 Linear and Non-Linear Distortions

Linear distortion is easily identified, provided accurate correspondences between both types of data exist. It can be regarded as a planar projective transformation, which can be represented by a non-singular matrix [21] as shown below.

$$\begin{pmatrix} x' \\ y' \\ z' \end{pmatrix} = \begin{pmatrix} h_{11} & h_{12} & h_{13} \\ h_{21} & h_{22} & h_{23} \\ h_{31} & h_{32} & h_{33} \end{pmatrix} \begin{pmatrix} x \\ y \\ z \end{pmatrix} \tag{27}$$

where (x, y, z) are coordinate in the USGS data and $(x^{'}, y^{'}, z^{'})$ are coordinate in the corresponding image. Since the source image and the USGS data are planar, z and $z^{'}$ can be assigned 1. Additionally, due to the up-to-scale phenomenon [21], one of the nine elements in the non-singular matrix can be assigned a value of 1 e.g., as $H = \begin{pmatrix} h_{11} & h_{12} & h_{13} \\ h_{21} & h_{22} & h_{23} \\ h_{31} & h_{32} & 1 \end{pmatrix}$. Therefore equation (27) becomes

$$\begin{pmatrix} x^{'} \\ y^{'} \\ 1 \end{pmatrix} = \begin{pmatrix} h_{11} & h_{12} & h_{13} \\ h_{21} & h_{22} & h_{23} \\ h_{31} & h_{32} & 1 \end{pmatrix} \begin{pmatrix} x \\ y \\ 1 \end{pmatrix} \tag{28}$$

The inhomogeneous form of (28) becomes [21]

$$\begin{aligned} x^{'} &= \frac{h_{11}x + h_{12}y + h_{13}}{h_{31}x + h_{32}y + 1} \\ y^{'} &= \frac{h_{21}x + h_{22}y + h_{23}}{h_{31}x + h_{32}y + 1} \end{aligned} \tag{29}$$

Given four pairs of measured data points, the transformation becomes

$$\begin{pmatrix} x_1^{'} \\ y_1^{'} \\ x_2^{'} \\ y_2^{'} \\ x_3^{'} \\ y_3^{'} \\ x_4^{'} \\ y_4^{'} \end{pmatrix} = \begin{pmatrix} x_1, y_{1,} 1,0,0,0, -x_1 x_1^{'}, -y_1 x_1^{'} \\ 0,0,0, x_1, y_1, 1, -x_1 y_1^{'}, -y_1 y_1^{'} \\ x_2, y_{2,} 1,0,0,0, -x_2 x_2^{'}, -y_2 x_2^{'} \\ 0,0,0, x_2, y_2, 1, -x_2 y_2^{'}, -y_2 y_2^{'} \\ x_3, y_{3,} 1,0,0,0, -x_3 x_3^{'}, -y_3 x_3^{'} \\ 0,0,0, x_3, y_3, 1, -x_3 y_3^{'}, -y_3 y_3^{'} \\ x_4, y_{4,} 1,0,0,0, -x_4 x_4^{'}, -y_4 x_4^{'} \\ 0,0,0, x_4, y_4, 1, -x_4 y_4^{'}, -y_4 y_4^{'} \end{pmatrix} * \begin{pmatrix} h_{11} \\ h_{12} \\ h_{13} \\ h_{21} \\ h_{22} \\ h_{23} \\ h_{31} \\ h_{32} \end{pmatrix} \tag{30}$$

If equation (30) is denoted by $X' = Mh$, both $X^{'}$ and ***h*** are 8x1 vectors, and M is an 8X8 matrix. Therefore, $h = M^{-1}X^{'}$. Since the USGS road data are text files, it is hard to find correspondences between the USGS road data and images. The USGS issues aerial photos with one meter accuracy which have been aligned with the corresponding road centerline data. A minimum of four points are chosen on the USGS and object images followed by computation of the non-singular matrix. This produces the transformation between the USGS road data and the object images. Linear distortion between the two sets of photos for this project is represented by the non-singular matrix which is $H = \begin{pmatrix} 0.2504 & -0.0011 & -110.2432 \\ 0.0010 & 0.2491 & -117.4968 \\ 0 & 0 & 1 \end{pmatrix}$. The most significant source of distortion is 2D translation, because the first and second elements in the third row of $\boldsymbol{H}$ are zeros.

Non-linear distortion is more difficult to detect. One method is to draw several scanlines in both aerial photos and choose feature correspondences along these scanlines. For feature points along vertical scanlines, their horizontal differences are compared. For feature points along horizontal scanlines, their vertical differences are compared. This method can give a preliminary view of non-linear distortion as shown in Fig. 33. The non-linear distortion is not significant, because the dimensions of the aerial images are thousands of pixels and the total distortion is within 6 pixels. This is acceptable for segmentation requirements. Fig. 34 shows the result of matching the USGS data to the aerial photography.

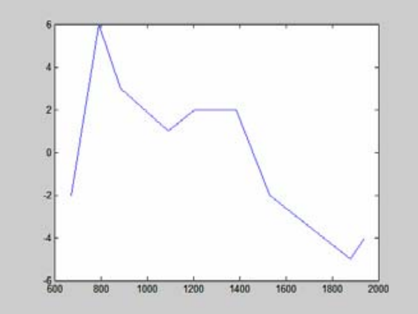

Fig. 33 The Graph of Non-Linear Distortion Measurement.

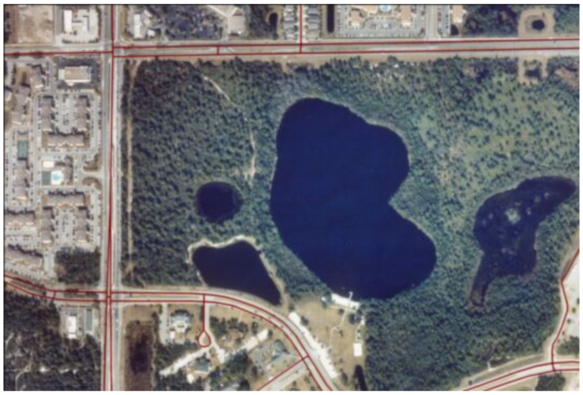

Fig. 34 Matching of the USGS DLG and UCF Aerial Photos

Projecting the USGS road data on the object images dramatically reduces the search space for segmentation.

4.2 Post Processing the Segmentation Results by the Mean-Shift Clustering Method

Usually pavements in aerial photos have lower saturation, compared with other areas. In other words, the areas that have a high average saturation are more likely to be non-pavement. It is possible to use saturation values to segment aerial images. The difficulty is that a saturation image can easily become noisy. Sometimes a saturation image is so noisy that the road boundary is not recognizable. In this case, the classification results via the mean-shift algorithm can be used. The mean-shift algorithm classifies an image into several classes. The average saturation value of each class separates pavements and non-pavements. Mathematically, the recognition process is as follows:

1. Sort the classes based on the average saturation in a descending order;

2. For class *i*, where *i* is from 1 to *n*, where *n* is the number of the classes, classify it to be non-pavement if *i*<=*t*, otherwise classify it to be pavement. Index *t* must satisfy $\frac{\overline{s_t} - \overline{s_{t+1}}}{\overline{s_t}} > 0.4$, where *s* stands for the average saturation with in a class.

The threshold value of 0.4 is to detect the first significant drop in the average saturations.

4.3 Conditional Morphological Methods

In morphological operations, an object is treated as a set. Fundamental morphological operations are the standard set operations which include union, intersection, translation, and complement. The morphological operations used to improve the segmentation results are closing and opening operations. Their mathematical expressions are shown below:

Dilation: $D(A,B) = \bigcup_{\beta \in B} (A + \beta)$ (30)

Erosion: $E(A,B) = \bigcap_{\beta \in B} (A - \beta)$ (31)

Opening: $O(A,B) = D(E(A,B),B)$ (32)

Closing: $C(A,B) = E(D(A,B),B)$ (33)

In the above four expressions, A is an object whose pixels have image coordinates, while B is a set of 2D vector, such as $[(0,0),(1,0),(0,1)]$, known as the structuring element. Operation dilation causes objects to grow in size and operation erosion causes objects to shrink. The amount of growing or shrinking depends on B. Operation opening is operation dilation followed by erosion. It removes small connected areas while preserving the shape and size of large objects in the image. Operation closing is operation erosion followed by dilation. It fills small gaps. It is important to correctly judge where to use the opening operation and where to use the closing operation for road segmentation. For a more detailed discussion about morphological operations, please refer to [36].

In order to find out where to use opening operations and where to use closing operations, USGS road information can be consulted. Along the USGS road center line, given a valid lateral offset, a path parallel to the road center line can be specified. Some pixels along each path are classified as non-pavement and some are classified as pavement by the method in 4.2. If the pavement pixels are numbered 0 and non-pavement pixels are numbered 1, the summation of all the pixels along a path reflects the likelihood of whether this path is in a pavement or non-pavement area. Fig. 35 is a graph showing this value along the road lateral offset for the segmentation result on a road image.

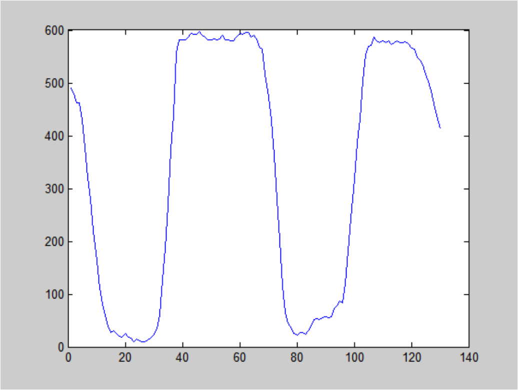

Fig. 35 The Number of Non-Pavement Pixels vs. Lateral Offset

The way to decide where the different morphological operations are to be used is to find four pairs of consecutive peaks and valleys in Fig. 35 and these four pairs should have the largest differences. The middle points of these four pairs of peaks and valleys can roughly specify the cross sectional view of the road. Due to the common knowledge, these five parts should be non-pavement, pavement, non-pavement, pavement, and non-pavement. Closing operations should be performed in non-pavement areas and opening operations should be performed in pavement areas. The conditional morphological operations remove noises caused by overlapping color information between pavement and non-pavement.

4.4 Removing Shadows and Occlusions

In aerial photos, some of the pavement area is covered by the shadow caused by trees or occluded by trees. The colors of trees and shadows are dark compared with other objects in an aerial photo. The method for removing shadows/occlusions relies on the mean-shift clustering result and the intermediate result in 4.3. In 4.2, the mean-shift algorithm classifies all the pixels into several classes. Since shadow and occlusion areas have lower intensity than anything else in the image, they will be separated into a class with the lowest intensity. However, shadow areas or occlusion areas could cover both pavement areas and non-pavement areas. In order to find out which part of a shadow/occlusion area is pavement, the intermediate result in Section 4.3 is used. Based on the graph in Fig. 35, the road cross-sectional profile can be obtained. With this information, the shadow and occlusion areas can be classified. This process is to see whether a shadow/occlusion pixel falls in a pavement area or non-pavement area. The process is described as follows:

1. The mean-shift algorithm in 4.2 segments an aerial image into several classes. The class with the lowest average intensity is the shadow/occlusion class.

2. Based on Fig. 35, a road profile is derived into five parts which are non-pavement, pavement, non-pavement, pavement, and non-pavement. Each part has a lateral range.
3. For each pixel of the shadow/occlusion class, if its lateral position is within the lateral range of a pavement area, it is classified as a pavement pixel; otherwise, it is classified as a non-pavement pixel.

4.5 Manual Adjustment

Even though the proposed method yields accurate segmentation results, it is still very difficult to automatically generate 3D road models from segmentation results. A software application with graphical user interface has been developed in this research. It has the following functionalities.

1. Loading images
2. Loading detected road boundary points in road segmentation
3. Moving images and points
4. Removing points
5. Inserting points
6. Connecting two objects
7. Saving adjusted points
8. Zooming in/out.

Below are several snapshots of this software application.

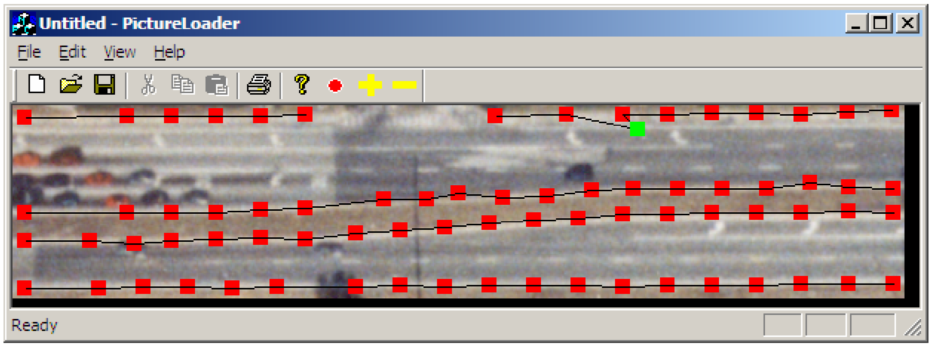

Fig. 36 Inserting a Point

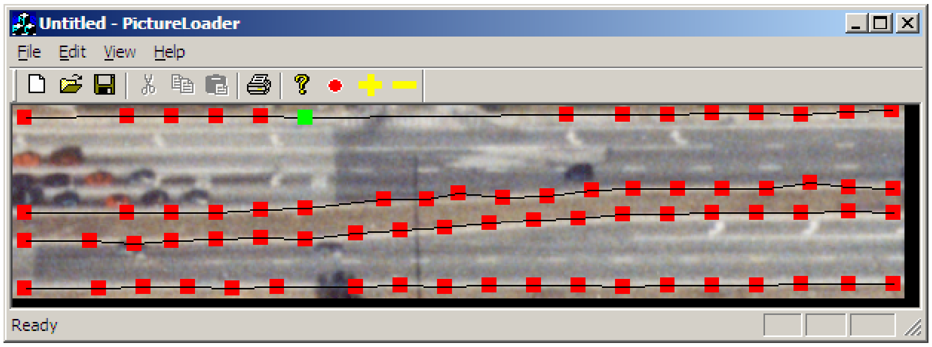

Fig. 37 Connecting Two Objects

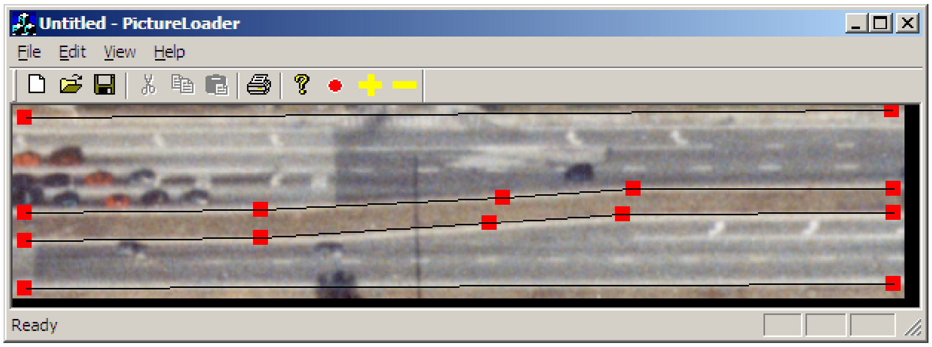

Fig. 38 Results of a Manual Adjustment

In the future research, one of the proposed works is to integrate more functions to this software. It should include automated road segmentation algorithms, 3D model generation program, and RDB generation program. Because building geo-specific roads is a major interest for any driving simulation system, such a software application is a worthwhile investment.

4.6 LBG Algorithm

In order to form a pool of cross sections, center lines are sampled. The sampling rate can vary. For each sampled center line point, a line is drawn perpendicular to its center line. This line will intersect with boundaries of roads and medians and comprise the top view of the cross section at this point. The true cross-sectional profile can be derived from the top view using basic geometric design parameters for existing roads and medians as shown in Fig. 39.

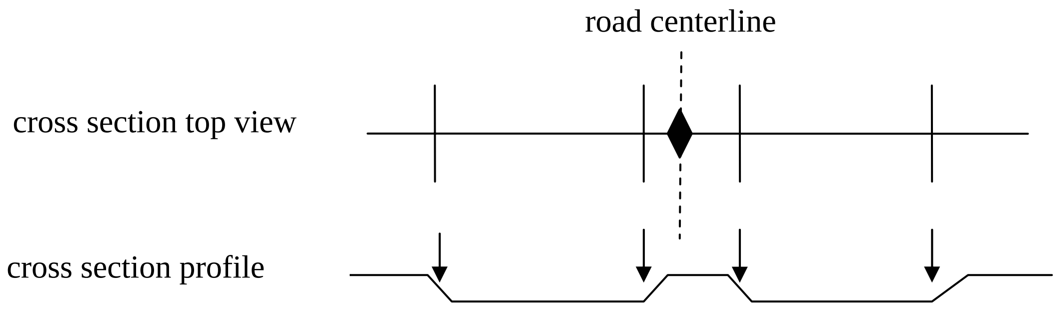

Fig. 39 Transforming Cross Sections Top Views in to True Profiles

In Fig. 39, the cross section top view line intersects with four vertical lines. The middle two vertical lines are median boundaries and the other two vertical lines are road boundaries. The four intersection points uniquely define a top view of the cross section associated with the diamond shape road centerline point. Assuming the height of curbs conforms to a six inch standard enables construction of the true cross section profile at this point.

Top views of cross sections are available at each sampled center line point with the possibility that many are very similar, if not identical. The question is how to identify similar cross sections. In this research, the Linde-Buzo-Gray (LBG) vector quantization algorithm [35] is used to identify similar top views automatically.

For example, in Fig. 39 the distances from the center line point to the four intersection points on the cross section top view are -48.0, -6.0, 7.5, and, 56.7 feet. A negative value indicates that the intersection point is located on the left side of the center line point. Consequently, the cross section vector of the center line point is (-48.0, -6.0, 7.5, 56.7).

The LBG algorithm automates the process of forming the pool of cross sections. It encodes cross section databases, identifies and removes redundant cross sections. The LBG algorithm can quantitatively evaluate the encoding error. Below is the description of the LBG algorithm training process.

1. Start with an initial set of reconstruction values $\left\{Y_i^{(0)}\right\}_{i=1}^{N}$ and a set of training vectors $\left\{X_i^{(0)}\right\}_{i=1}^{M}$. Set $k = 0$, $D^{(0)} = 0$. Select threshold $\in$.

2. The quantization regions $\{V_i^{(k)}\}_{i=1}^{M}$ are given by

$V_i^{(k)} = \{X_n : d(X_n, Y_i) < d(X_n, Y_j) \forall j \neq i\}, i = 1,2,....,N$,

where $d(X_n, Y_j)$ is the Euclidean distance between vectors X_n and Y_j

3. Compute the average distortion $D^{(k)}$ between the training vectors and the representative reconstruction value.

4. If $\frac{D^{(k)} - D^{(k-1)}}{D^{(k)}} < \in$ stop; otherwise continue.

5. $k = k + 1$. Find new reconstruction values $\left\{Y_i^{(k)}\right\}_{i=1}^{N}$ that are the average value of the elements of each of the quantization regions $V_i^{(k-1)}$. Go to Step 2.

The input to the LBG algorithm is a set of M cross section vectors. The desired number of cross sections is N and $\left\{Y_i^{(k)}\right\}_{i=1}^{N}$ will be the cross sections in the cross section pool when training stops. $\left\{X_i^{(0)}\right\}_{i=1}^{M}$ are the M cross sections before quantization and each one will point to the closest member of $\left\{Y_i^{(k)}\right\}_{i=1}^{N}$ after quantization.

Initially there is no prior knowledge of how many cross sections are needed for describing the original cross section database. An acceptable approach is to start with a small value of N, e.g. 2 and then calculate the average error among all the cross sections. If this error is not acceptable, the intermediate N vectors will be perturbed to generate another group of N vectors for retraining. The perturbing method adopted in this research is to shift each original entry by a small vector. This small vector is 0.3 times the vector standard deviation of the part of training set near an entry. For example, a 2D vector $\{y_1, y_2\}$ is chosen to represent $M^{'}$ vectors out of all M training vectors, and these $M^{'}$ vectors have standard deviations of std_1 and std_2 for the first and second items. The newly generated vector is $\{y_1, y_2\} + 0.3 * \{std_1, std_2\}$.

The relationship between average errors and numbers of vectors in the codebook is shown in Table 2. The LBG algorithm should work towards two pools of cross sections corresponding to roads with and without medians. The top views of road cross sections with medians will have four points, while those without medians will have only two points. Therefore the LBG algorithm is identifying similar 2D vectors and 4D vectors.

Table 2 shows an example of how encoding error changes with respect to the pool size. Note how the encoding errors are reduced when the pool sizes increase for both two-point and four-point cross sections. Both pool sizes were set to 128. After quantization, the number of two-point cross sections was reduced from 611 to 113 and the number dropped from 3313 to 189 for four-point cross sections.

Table 2. Encoding Errors

Two-Point Cross Section		Four-Point Cross Section	
Pool Size	Average Error (ft)	Pool Size	Average Error (ft)
2	4.30	2	4.25
4	3.80	4	3.40
8	2.90	8	2.60
16	2.03	16	1.88
32	1.38	32	1.39
64	0.82	64	1.00
128	0.40	128	0.74
256	0.17	256	0.53

4.7Automatically Creating 3D Road Model

When repeatable cross sections are identified, it is possible to convert them, combined with center lines, into the visual database automatically. One major type of information, which is not available in an RDB, is pavement markings. However this can be derived based on common knowledge, such as a lane is 12 foot wide and a marker is 6 inch wide. Then it is possible to decide which texture should be mapped at a certain place. The proposed method is to draw polygons, each of which can be mapped a uniform texture. Five textures can be made beforehand: white marker, yellow marker, concrete, grass, and pavement. This method will create many polygons, however from the experiments these polygons can easily be handled by a high quality graphics card. This procedure is shown in the following figure.

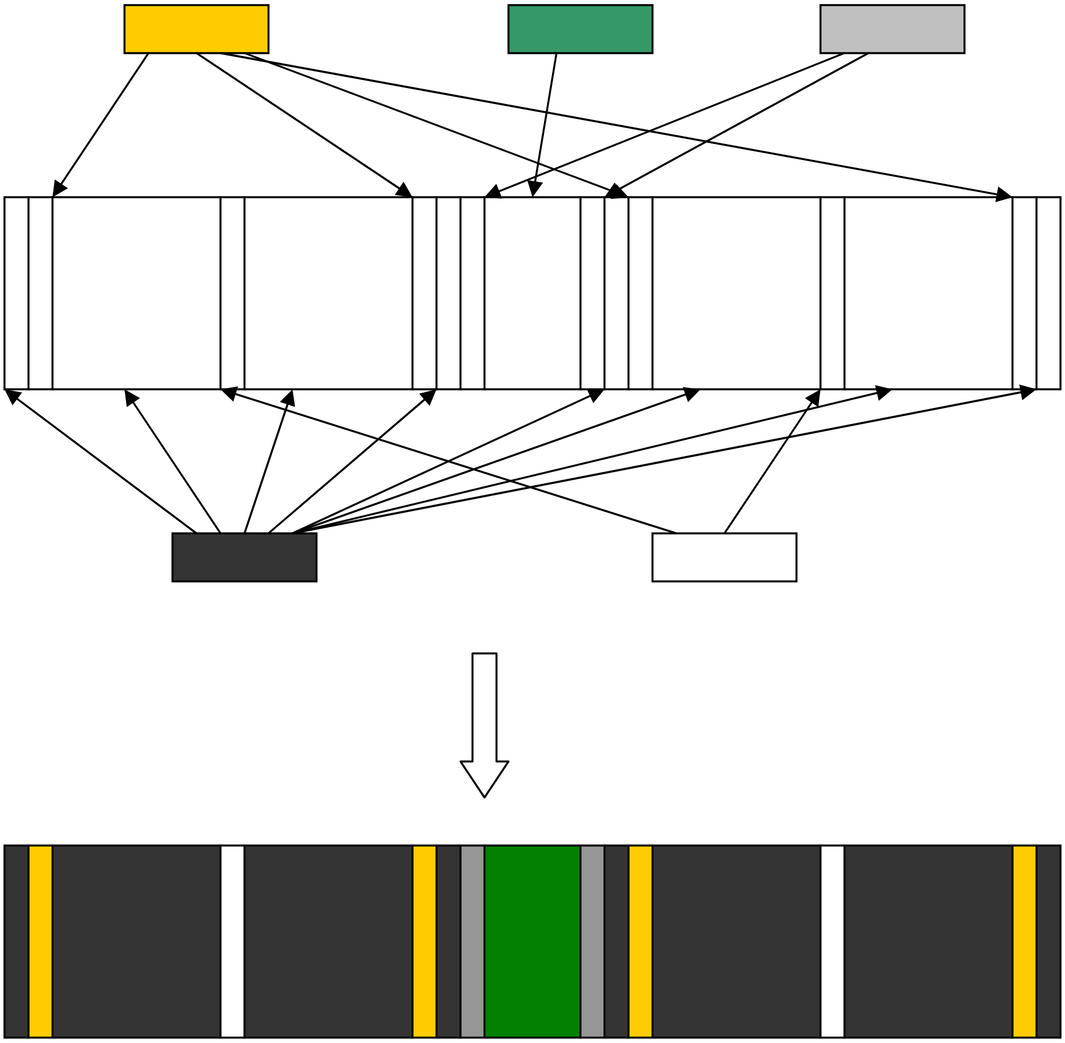

Fig. 40 Texture Mapping in Road Modeling

Polygons and five textures are placed in the upper half. Because the types of these polygons are known when they are drawn, it is straightforward to map textures. This method is not necessary to generate the final version of the visual database, but it can eliminate a great deal of work.

4.8 Summary of the Methodology

The proposed method has two main modules. The first one contains all the steps related to image recognition and its purpose is to achieve accurate road segmentation. The first step in this module is to accurately match USGS DLG information with high resolution images in order to locate where roads are. Then the mean-shift clustering method is used to segment a road area into several classes. If a class's average saturation is low, it is more likely to be a pavement area. If a class's average saturation is high, it is more likely to be a non-pavement area. The way of deciding is to sort these classes with respect to average saturation value. The first significant drop in average saturation value between two consecutive classes separates the two major categories of classes. The classes before the significant drop are non-pavement, and the ones after the significant drop are pavement. Segmentation results up to this point need to be improved using morphological opening and closing operations. It is critical to decide where to use opening/closing operation. Additionally, the effect of shadows and occlusion on road segmentation should be removed.

After achieving accurate road segmentation results, road boundary points are manually adjusted by a software application with a graphical user interface. This software enables modelers to insert, remove, connect and adjust boundary points. This step is very important and a bridge between road segmentation results and 3D modeling. Then, cross section information is sampled along each center line. In order to choose repeatable cross sections, the LBG vector quantization algorithm is used to identify repeatable cross section. Afterwards, all the information needed to build an STS compatible RDB is ready. The algorithm for building 3D models draws many polygons, each of which can be mapped a uniform texture.

The following figure shows how this method works.

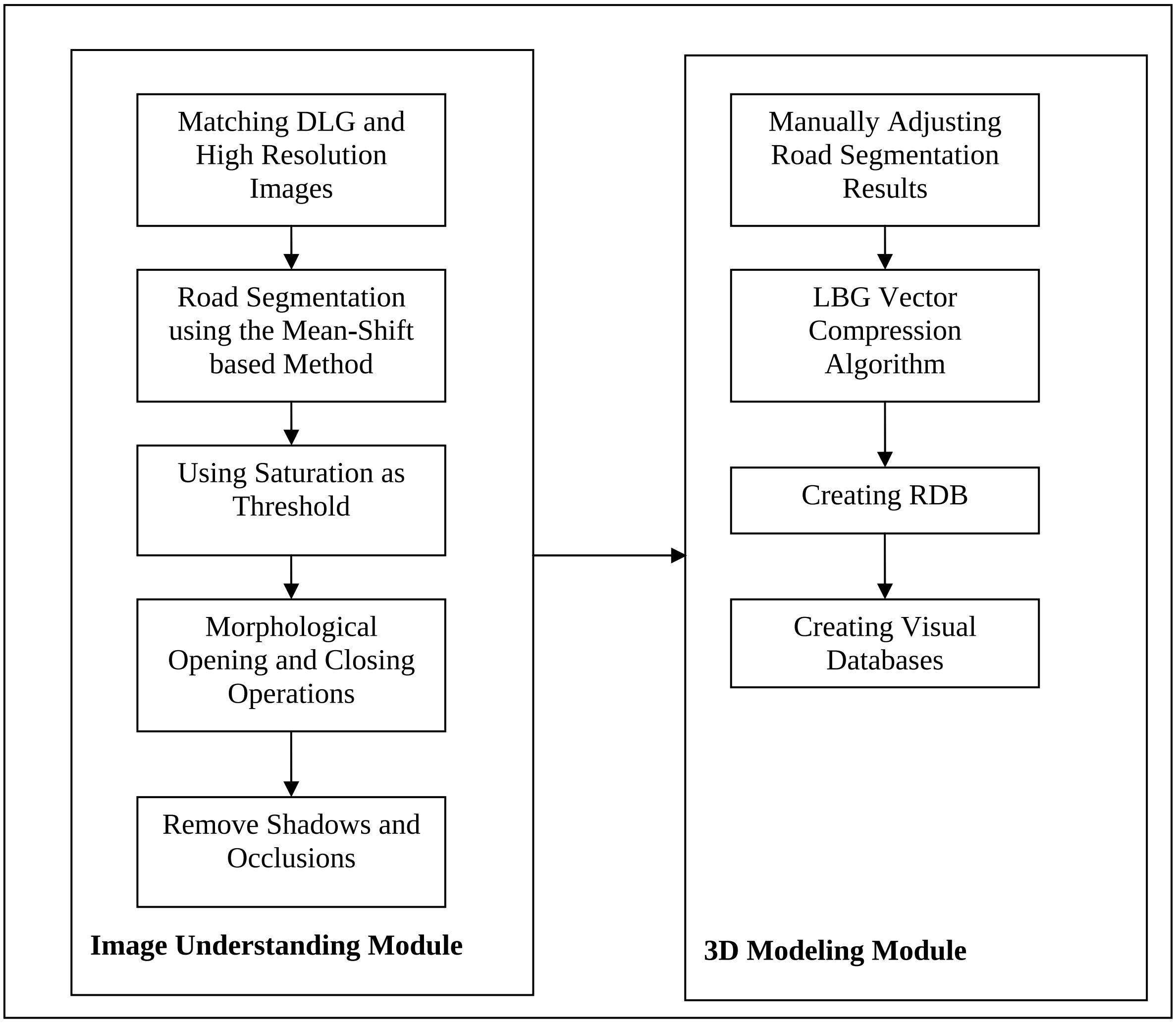
Matching DLG and High Resolution Images
Road Segmentation using the Mean-Shift based Method
Using Saturation as Threshold
Morphological Opening and Closing Operations
Remove Shadows and Occlusions
Image Understanding Module
Manually Adjusting Road Segmentation Results
LBG Vector Compression Algorithm
Creating RDB
Creating Visual Databases
3D Modeling Module

Fig. 41 Flow Chart of the Proposed Method

CHAPTER FIVE: RESULTS

5.1 The Results of Saturation-Based Segmentation

The uniform areas in the segmentation results shown in 3.6 are classes with similar HSI values. In order to have a binary segmentation, saturation images can be used. In the following saturation image (Fig. 42), pavement areas have a lower value of saturation and non-pavement areas have a higher value of saturation. If extracted objects' average saturation values are sorted in a descending order, a significant drop can normally be found in Fig. 43.

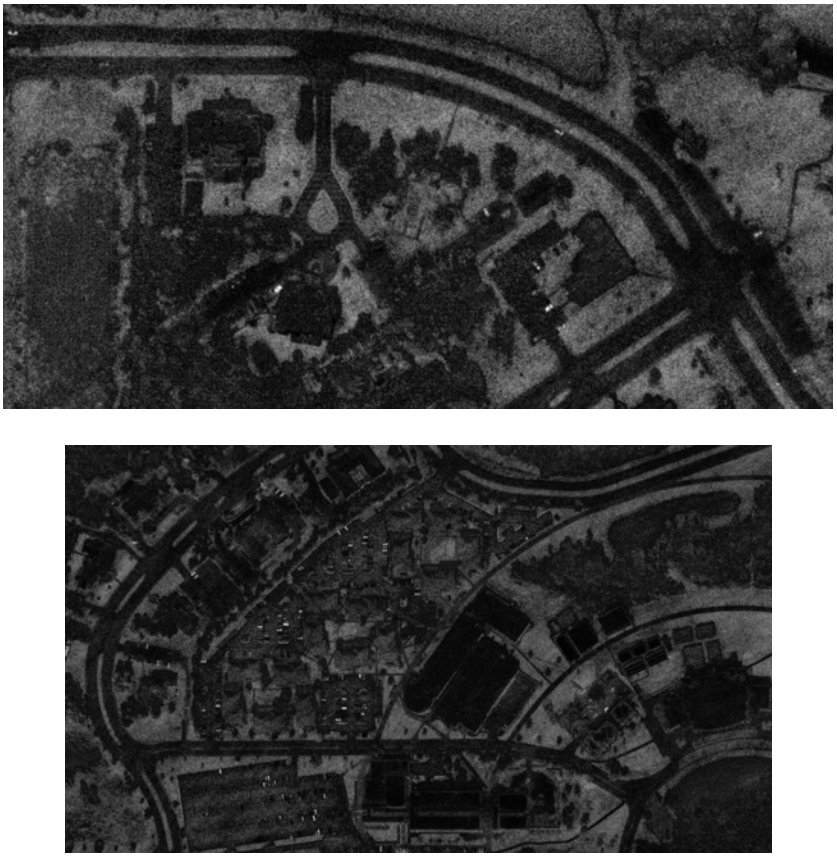

Fig. 42 Examples of Saturation Images

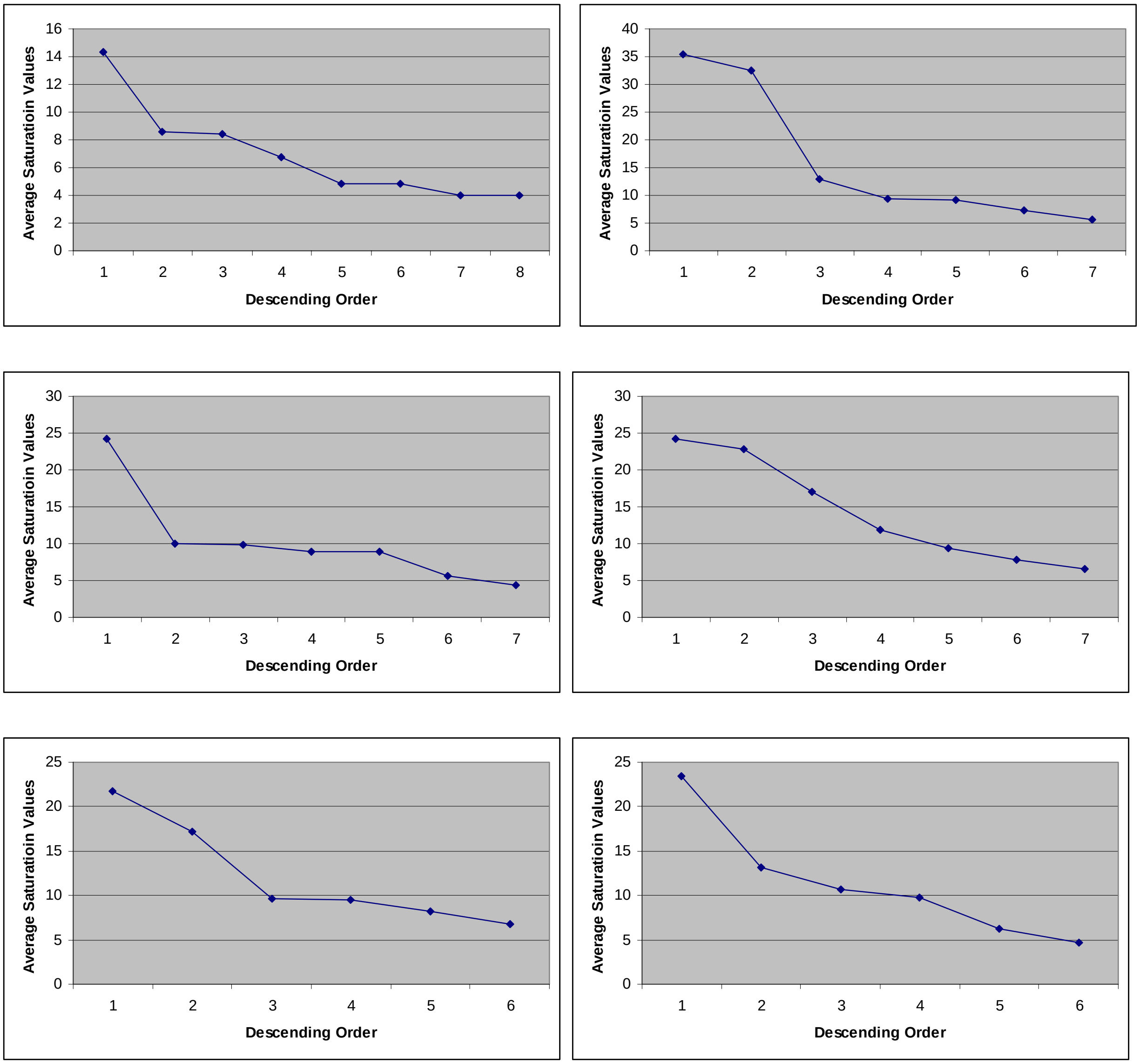

Fig. 43 Saturation Value in a Descending Order

Below are the segmentation results after using saturation as a cue to categorize the objects extracted from the mean-shift clustering algorithm.

Fig. 44 Result of Segmentation after Using Saturation as a Cue (1)

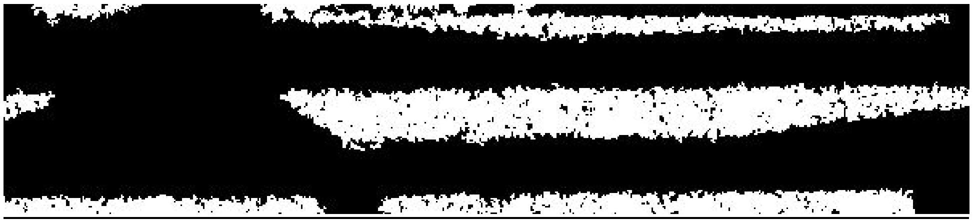

Fig. 45 Result of Segmentation after Using Saturation as a Cue (2)

Fig. 46 Result of Segmentation after Using Saturation as a Cue (3)

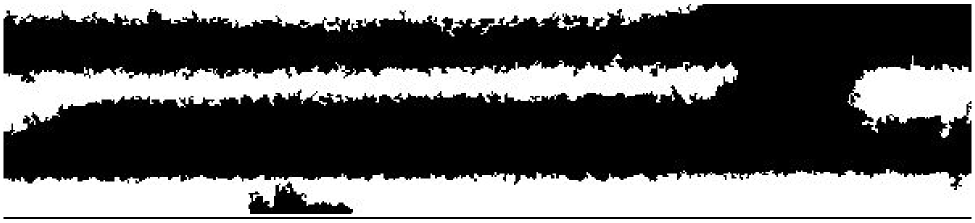

Fig. 47 Result of Segmentation after Using Saturation as a Cue (4)

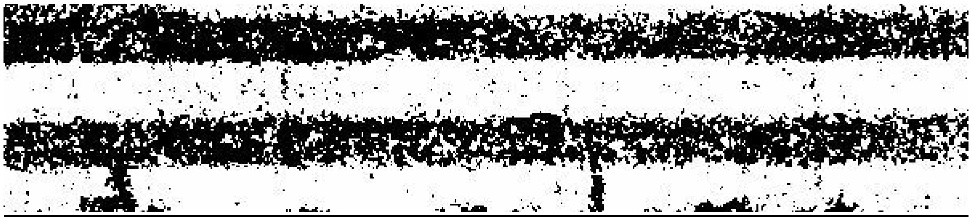

Fig. 48 Result of Segmentation after Using Saturation as a Cue (5)

Fig. 49 Result of Segmentation after Using Saturation as a Cue (6)

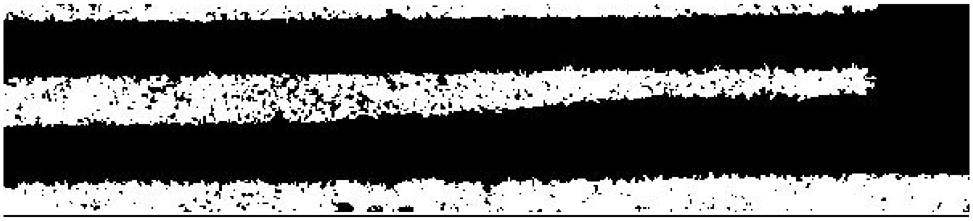

Fig. 50 Result of Segmentation after Using Saturation as a Cue (7)

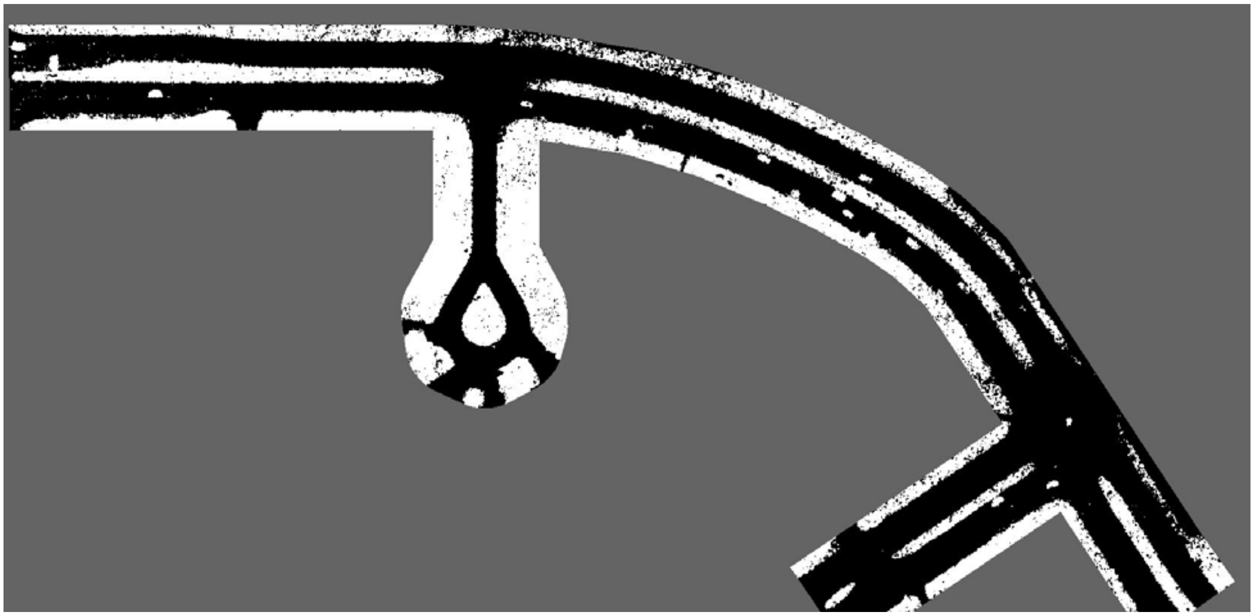

Fig. 51 Result of Segmentation after Using Saturation as a Cue (8)

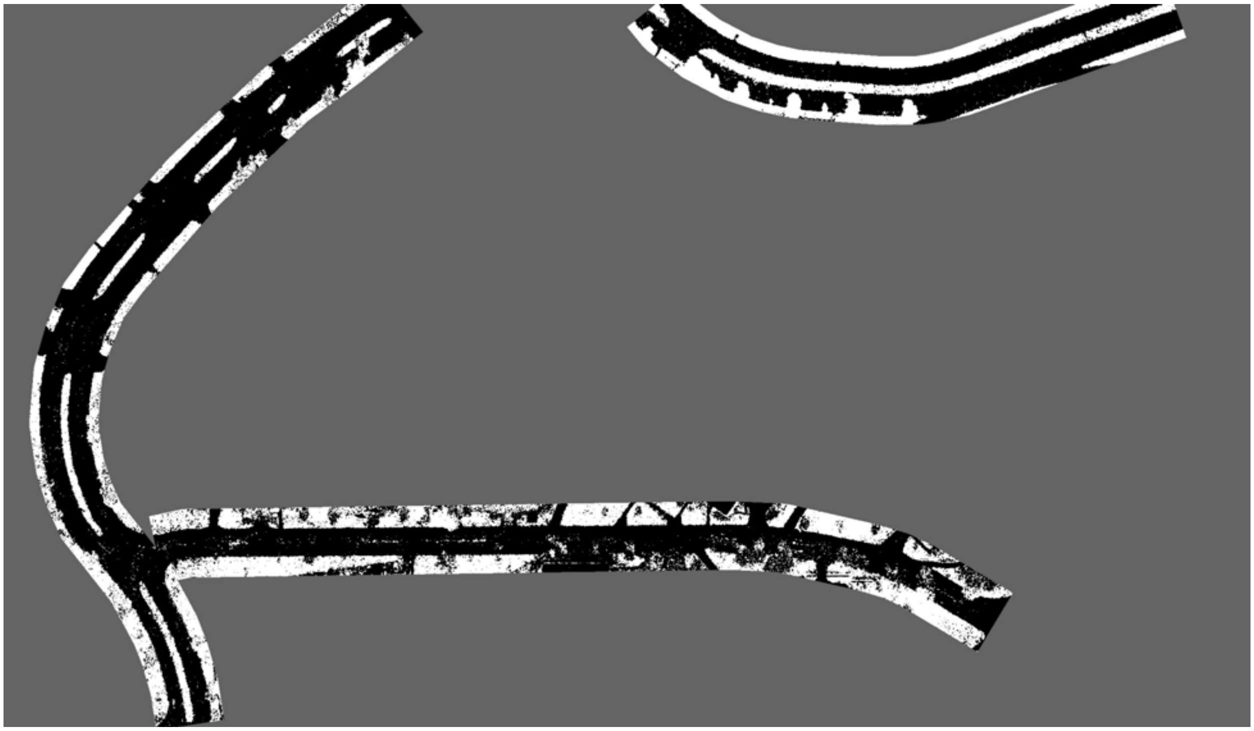

Fig. 52 Result of Segmentation after Using Saturation as a Cue (9)

5.2 The Results of Conditional Morphological Operations

The morphological operations used in this research are opening and closing operations. The opening operation should be used within pavement areas and the closing operation should be used within non-pavement areas. The following pictures are the results.

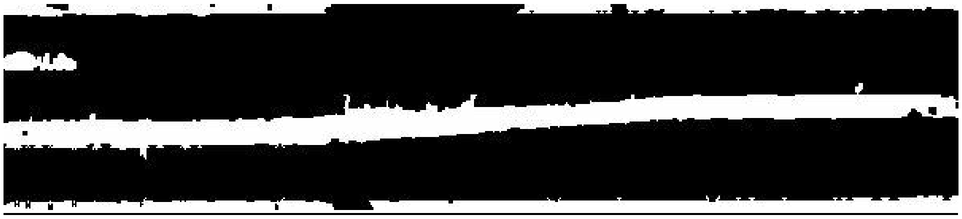

Fig. 53 Result of Morphological Operations (1)

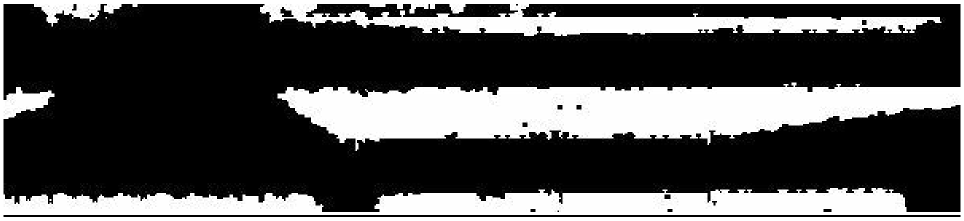

Fig. 54 Result of Morphological Operations (2)

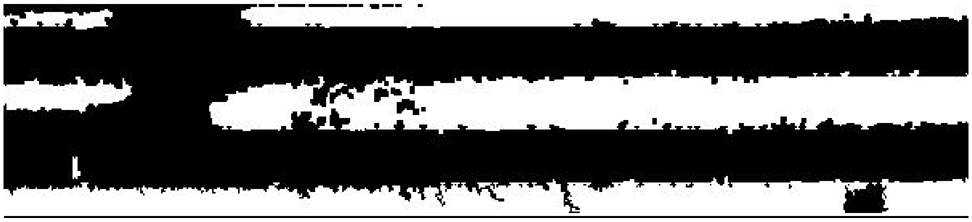

Fig. 55 Result of Morphological Operations (3)

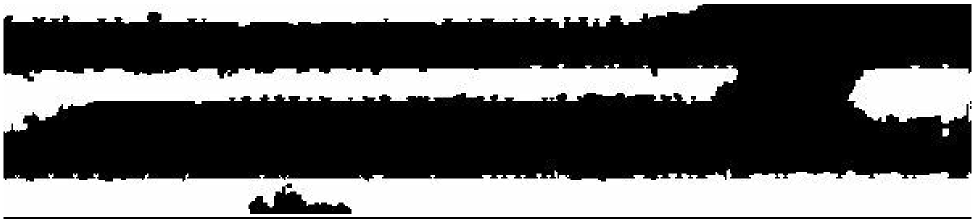

Fig. 56 Result of Morphological Operations (4)

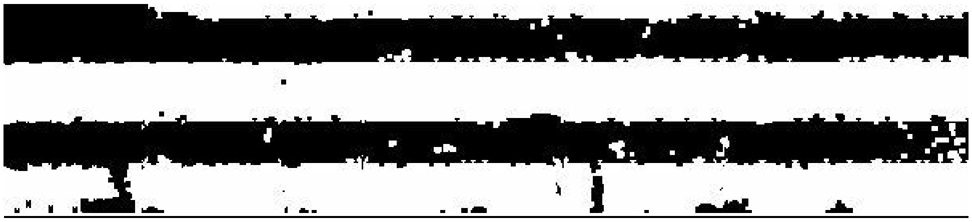

Fig. 57 Result of Morphological Operations (5)

Fig. 58 Result of Morphological Operations (6)

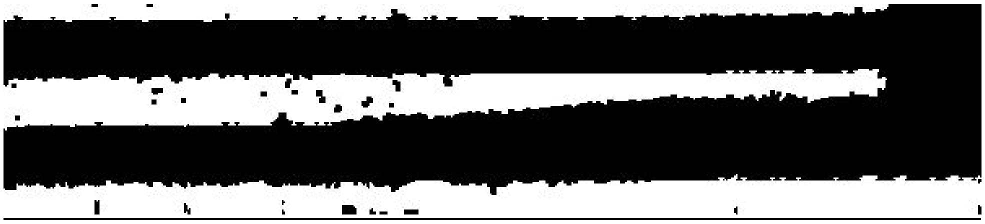

Fig. 59 Result of Morphological Operations (7)

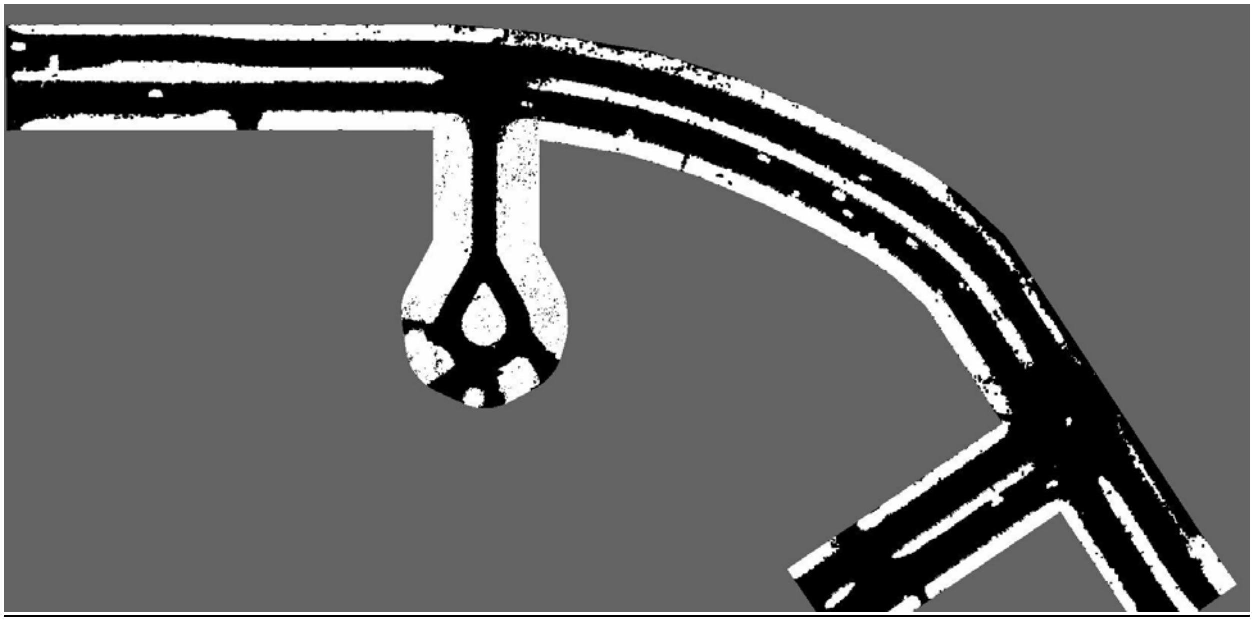

Fig. 60 Result of Morphological Operations (8)

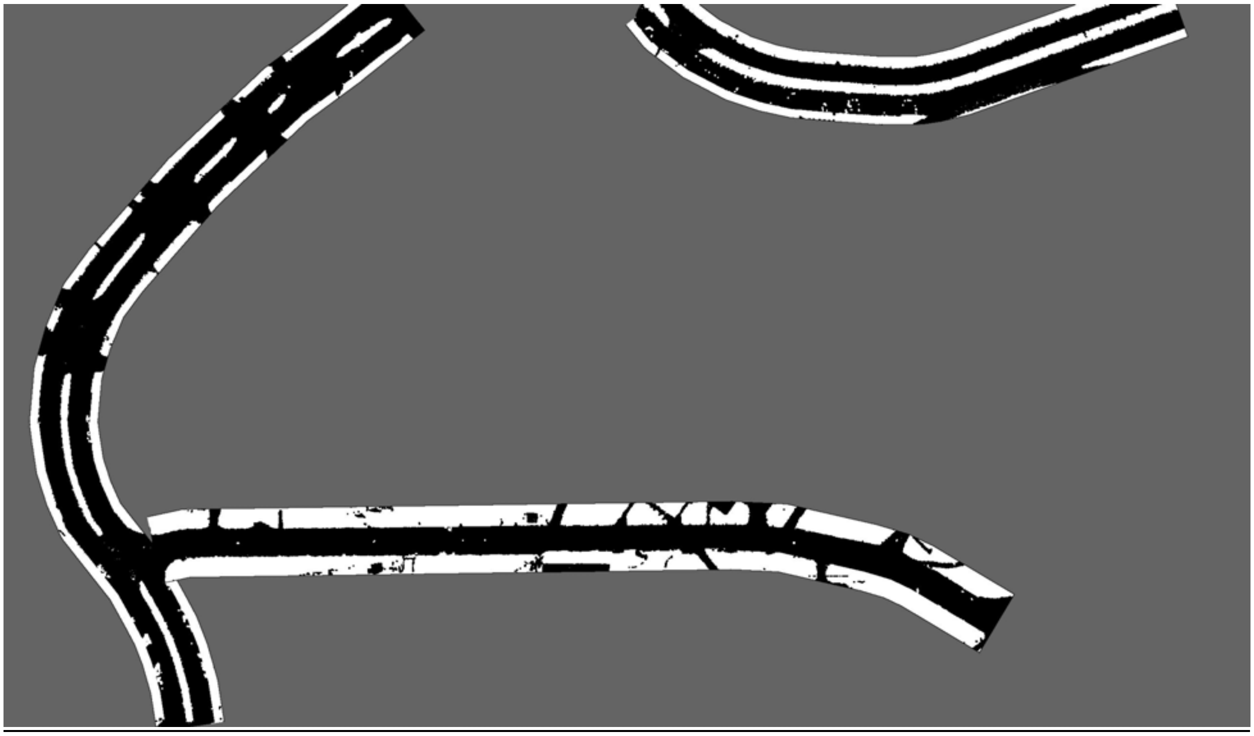

Fig. 61 Result of Morphological Operations (9)

5.3 The Results of Shadow and Occlusion Removal

In the top of Fig. 62, there are several trees on the side of the road. Some of the pavement area is covered by the shadow caused or occluded by trees. The areas covered by shadows and occlusions are classified using the results of 5.2 and the road cross sectional profile derived in 4.3. The result is shown in the bottom of Fig. 62.

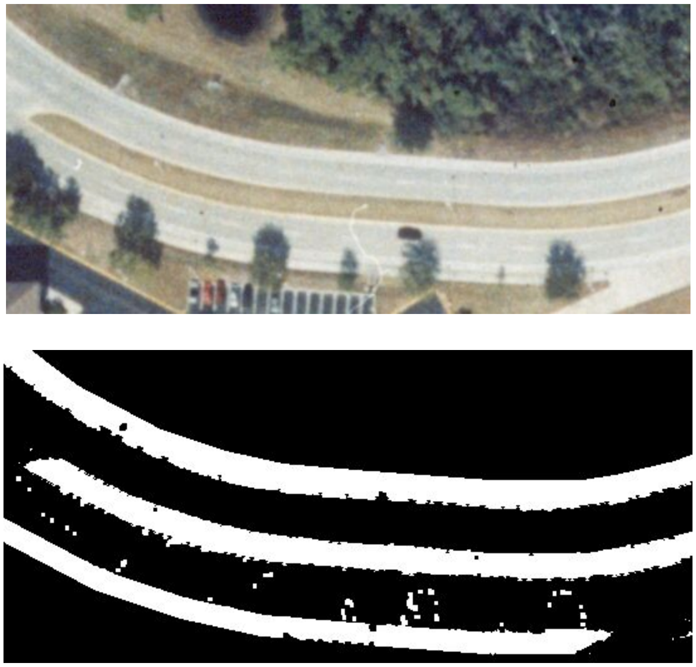

Fig. 62 The Road Image with Shadows and Occlusions, and Its Segmentation Result

5.4 The Accuracy of Road Segmentation

In order to measure the accuracies of the road segmentation results, they are compared with the results obtained by a real person using Adobe PhotoShop [22]. The results by the real person are used as the correct results. The accuracies are listed in Table 3.

Table 3. The Accuracies of the Segmentation Results

Image Size	Accuracy
600X190	86.82%
600X190	86.87%
600X190	93.65%
600X190	93.78%
600X190	87.15%
600X190	88.05%
600X190	95.75%
2542X1475	91.78%
1652X806	90.56%

In Table 2, all the accuracies are higher than 85%. The first seven images are small compared with the last two images. However, the accuracies are not affected by the image size.

5.5 The Results of Manually Adjusted Road Boundaries

Because of the high accuracies of road segmentation, it is easy for modelers to adjust the automatically recognized road boundaries. Below are the results of manually adjusted road boundaries, which are the input to the RDB and visual database generation algorithms.

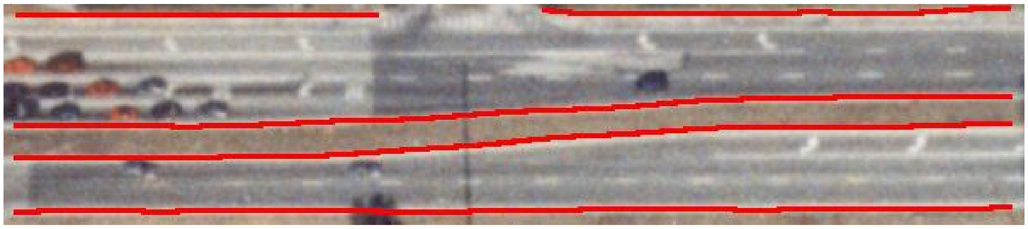

Fig. 63 Result of Manually Adjusted Road Boundaries (1)

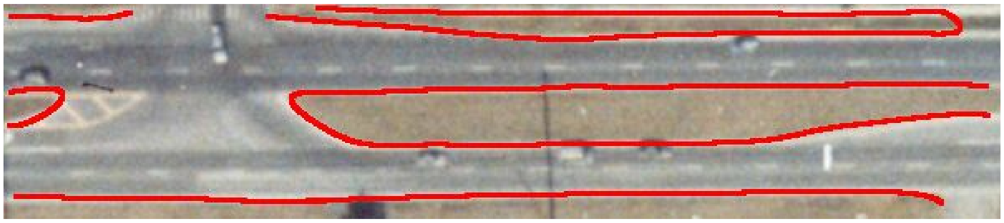

Fig. 64 Result of Manually Adjusted Road Boundaries (2)

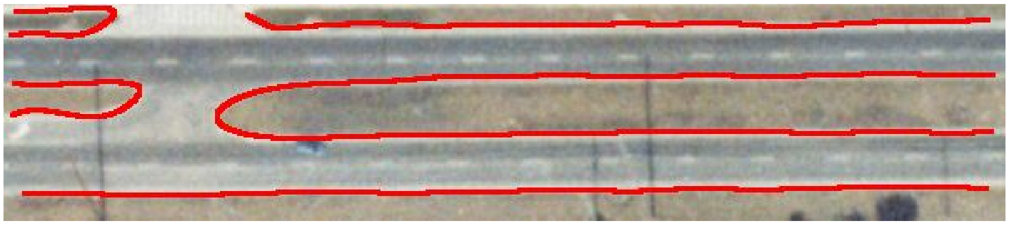

Fig. 65 Result of Manually Adjusted Road Boundaries (3)

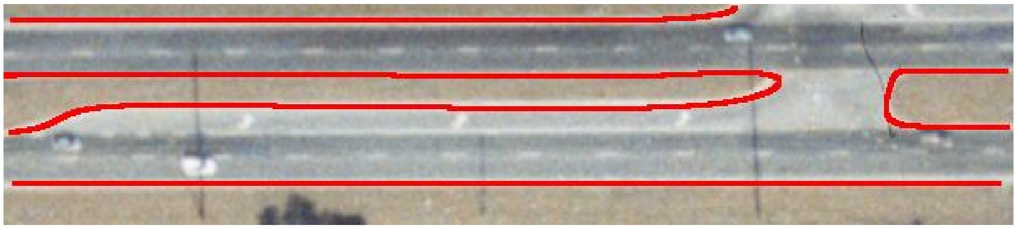

Fig. 66 Result of Manually Adjusted Road Boundaries (4)

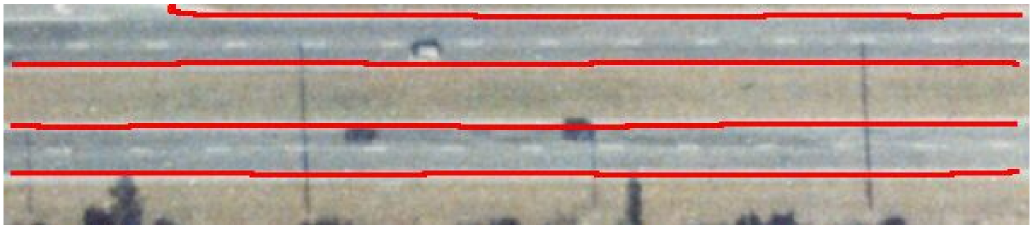

Fig. 67 Result of Manually Adjusted Road Boundaries (5)

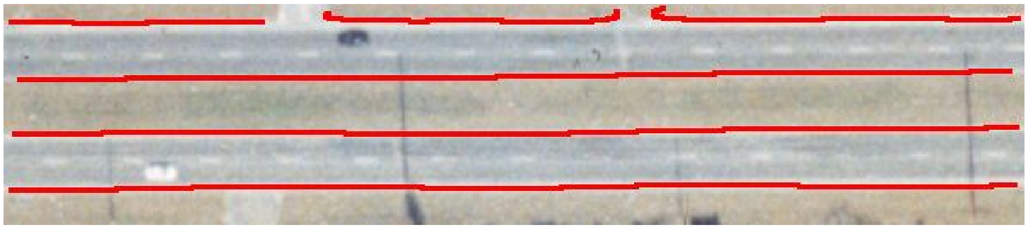

Fig. 68 Result of Manually Adjusted Road Boundaries (6)

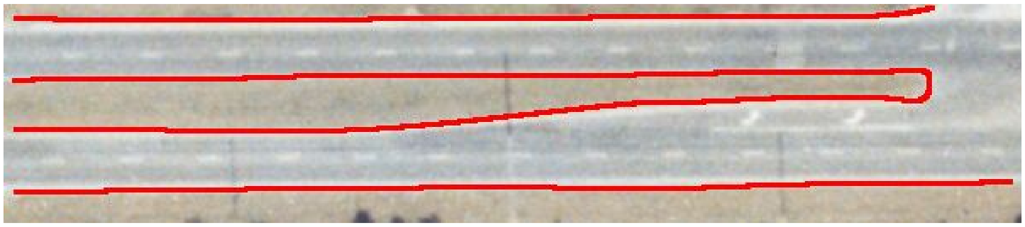

Fig. 69 Result of Manually Adjusted Road Boundaries (7)

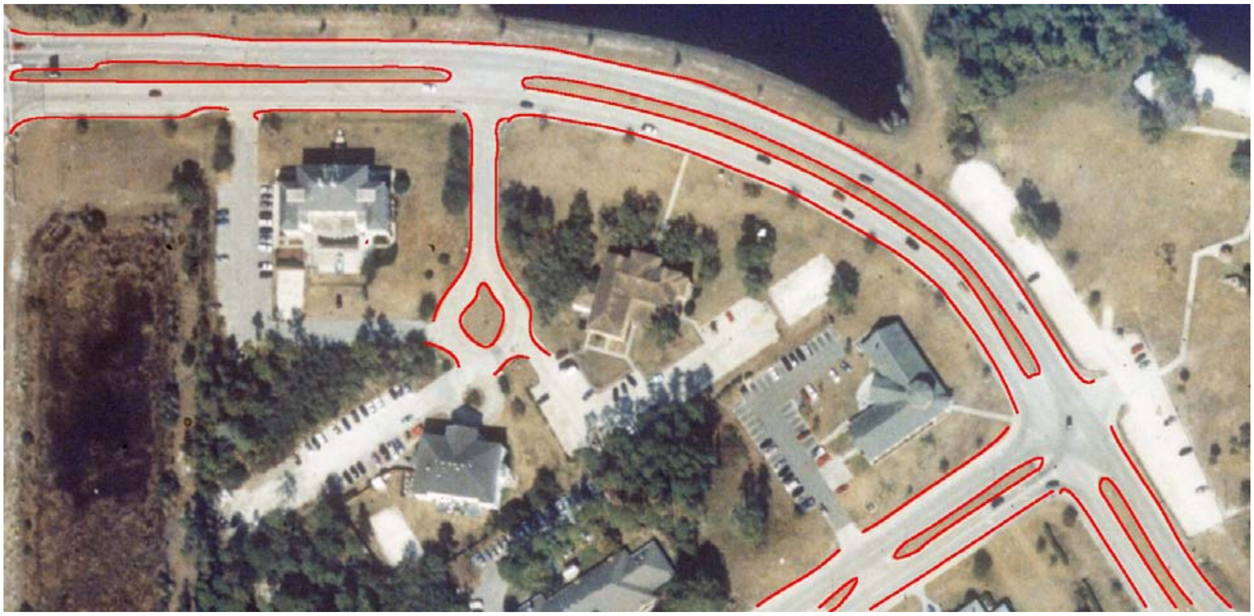

Fig. 70 Result of Manually Adjusted Road Boundaries (8)

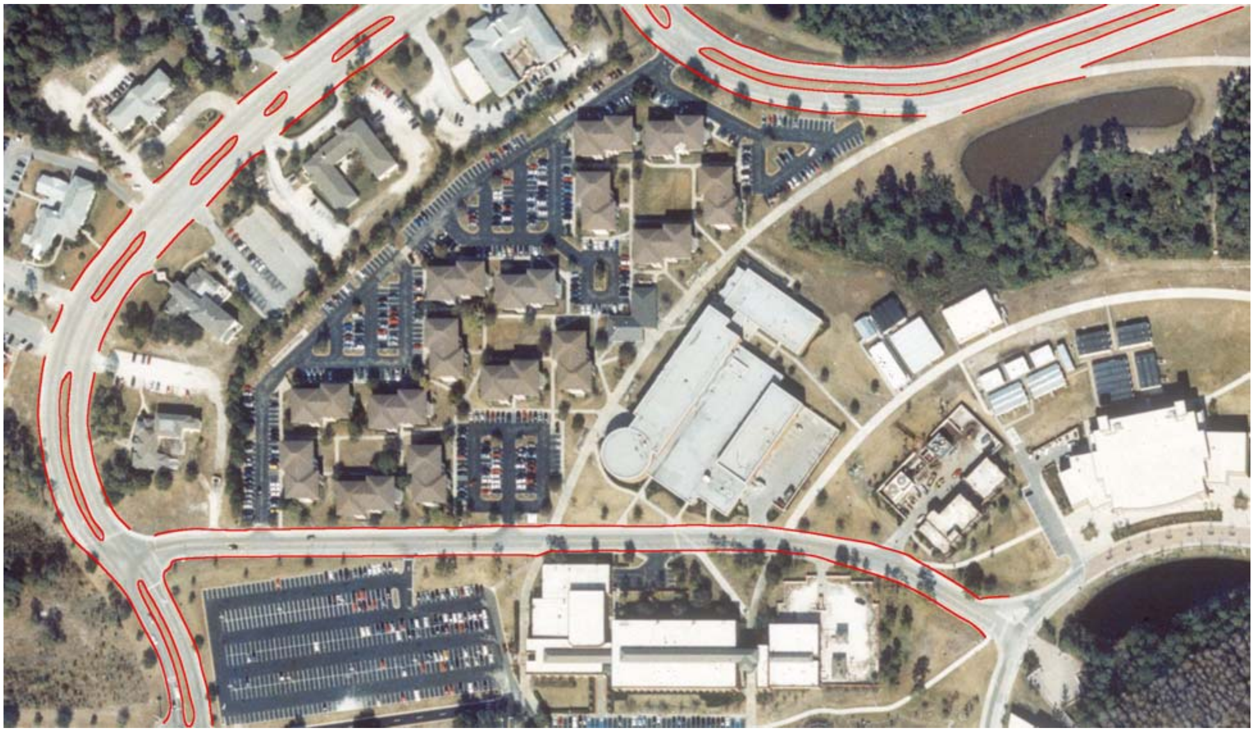

Fig. 71 Result of Manually Adjusted Road Boundaries (9)

5.6 Results of the LBG Algorithm

Three values can reflect how the LBG algorithm works. The first one is the average error between a cross section and its corresponding code entry. It is simply the Euclidean distance between a data entry and a code entry. The second and the third ones are the number of data entries before being quantized and the number of code entries after. Below is a table that shows these values.

Table 4. Results of the LBG Algorithm

Number of Cross Section Points	Average Error (pixel)	Number of Data Entries	Number of Code Entries
2	0.25	2615	275
4	0.36	6253	600

The differences between numbers of data entries and code entries are significant and the average errors are acceptable. This information can be easily converted into RDB and below are snapshots of roads in the STS Scenario Editor.

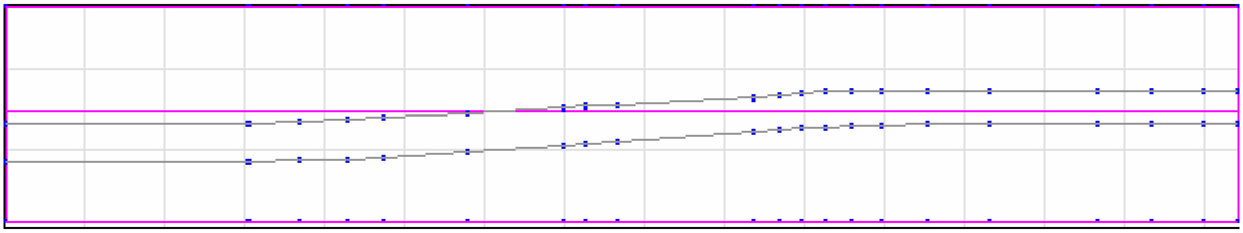

Fig. 72 Snapshot of STS Road Database (1)

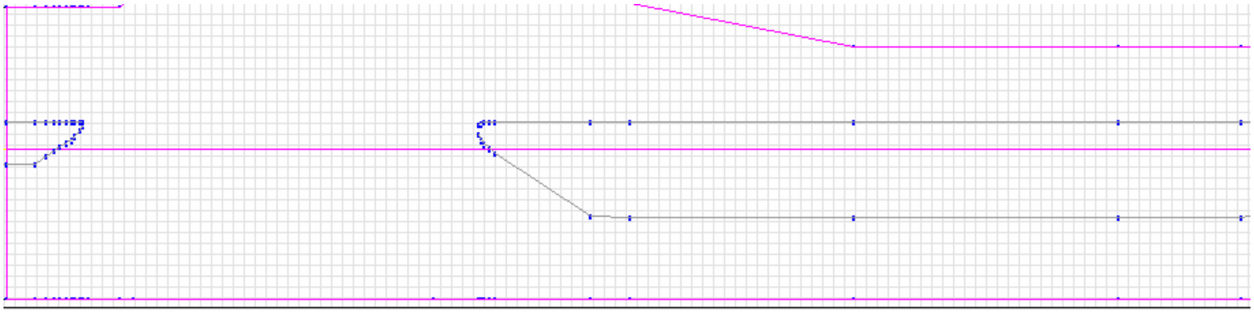

Fig. 73 Snapshot of STS Road Database (2)

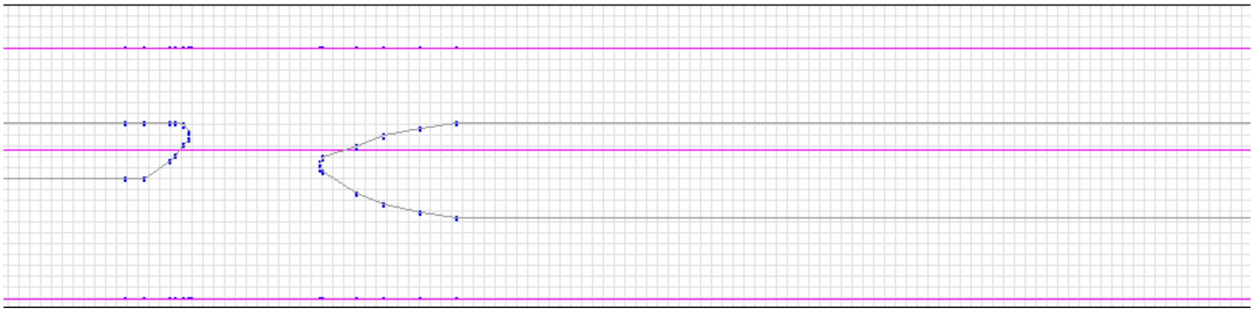

Fig. 74 Snapshot of STS Road Database (3)

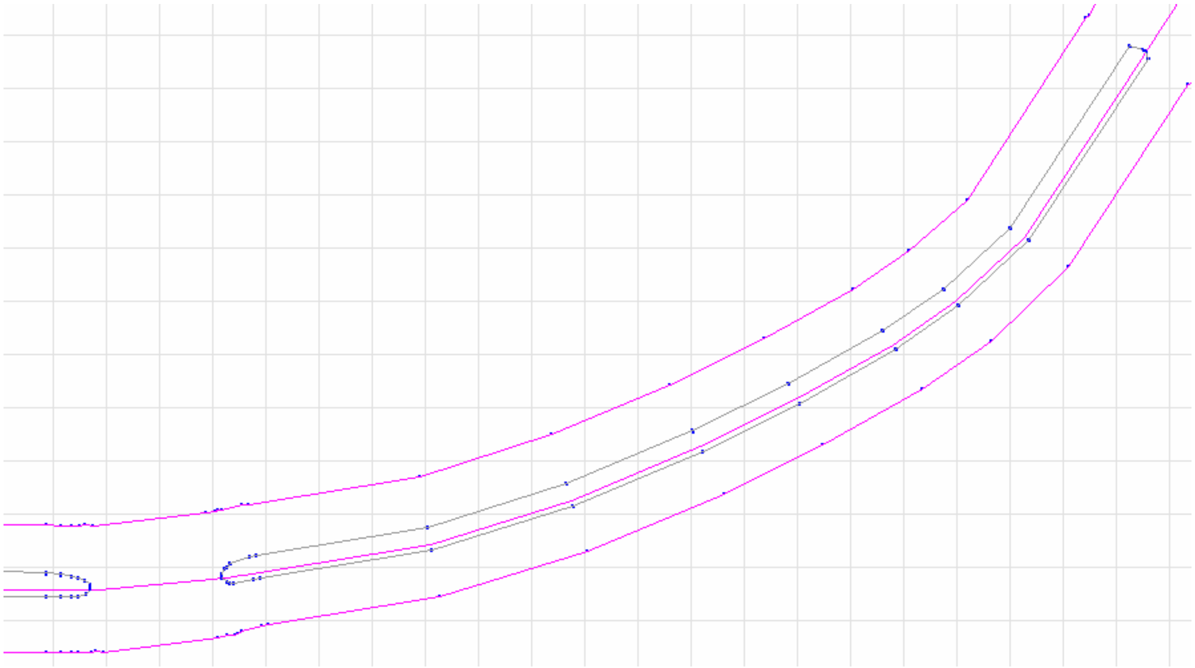

Fig. 75 Snapshot of STS Road Database (4)

5.7 Results of Building Visual Databases

Using the MultiGen APIs, road information can be converted into the MultiGen file format. Many polygons will be generated and each polygon is mapped a uniform texture. Below are snapshots of the visual database. The 3D road modeling shown in these snapshots are realistic and simulation compatible.

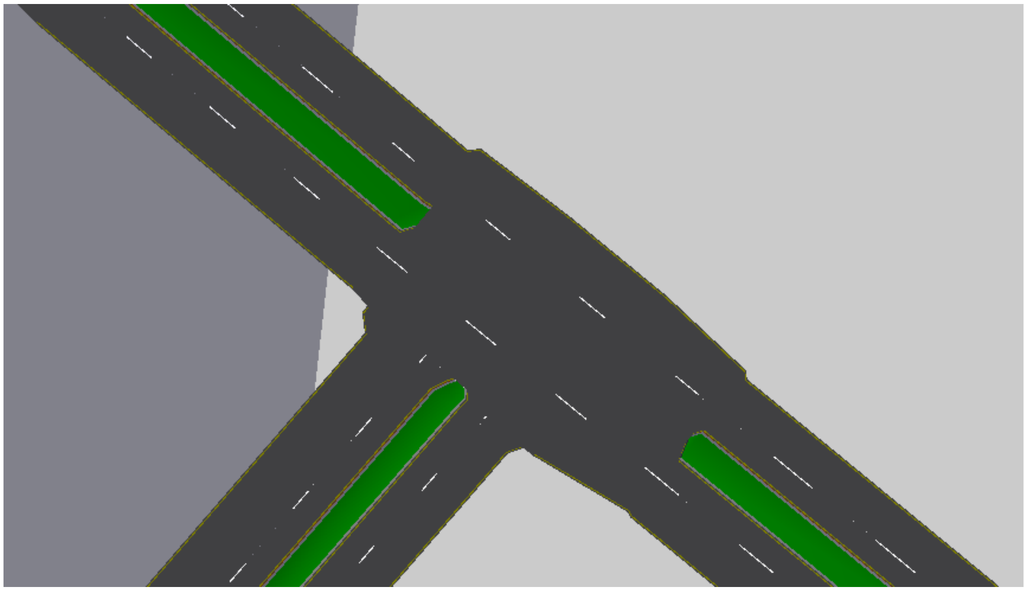

Fig. 76 Snapshot of Visual Database (1)

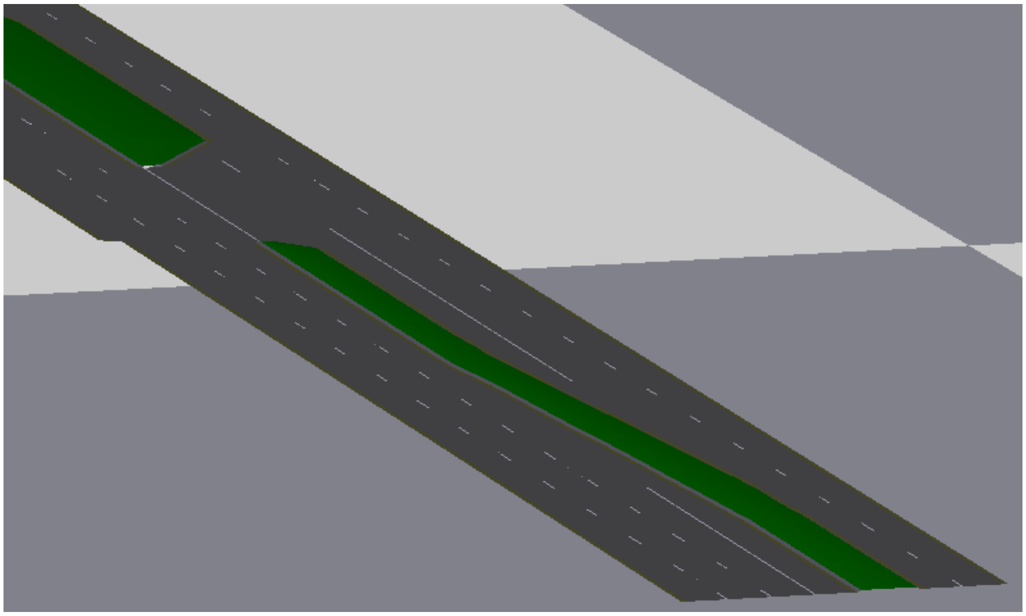

Fig. 77 Snapshot of Visual Database (2)

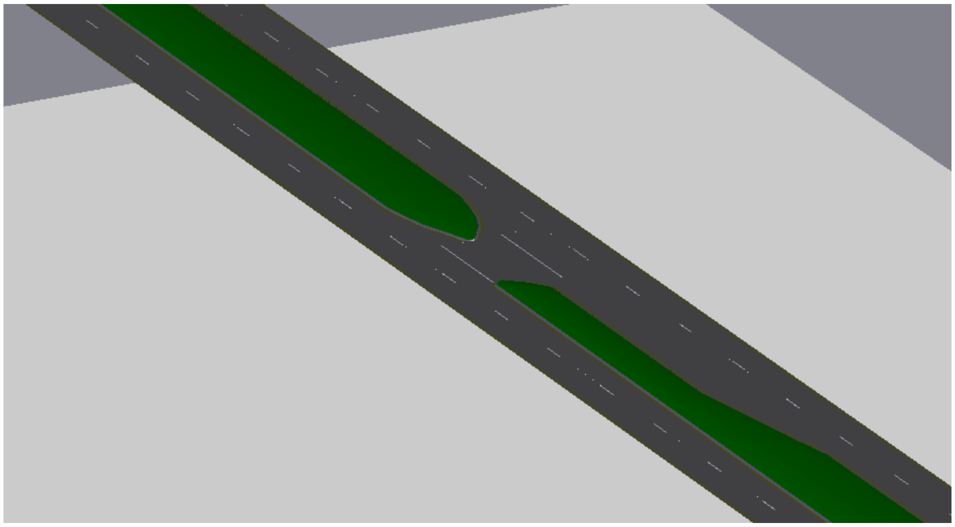

Fig. 78 Snapshot of Visual Database (3)

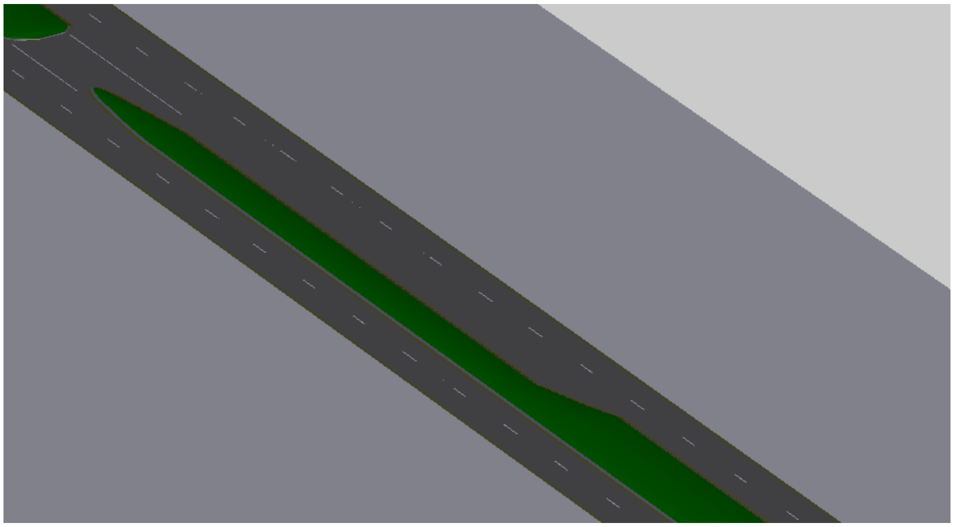

Fig. 79 Snapshot of Visual Database (4)

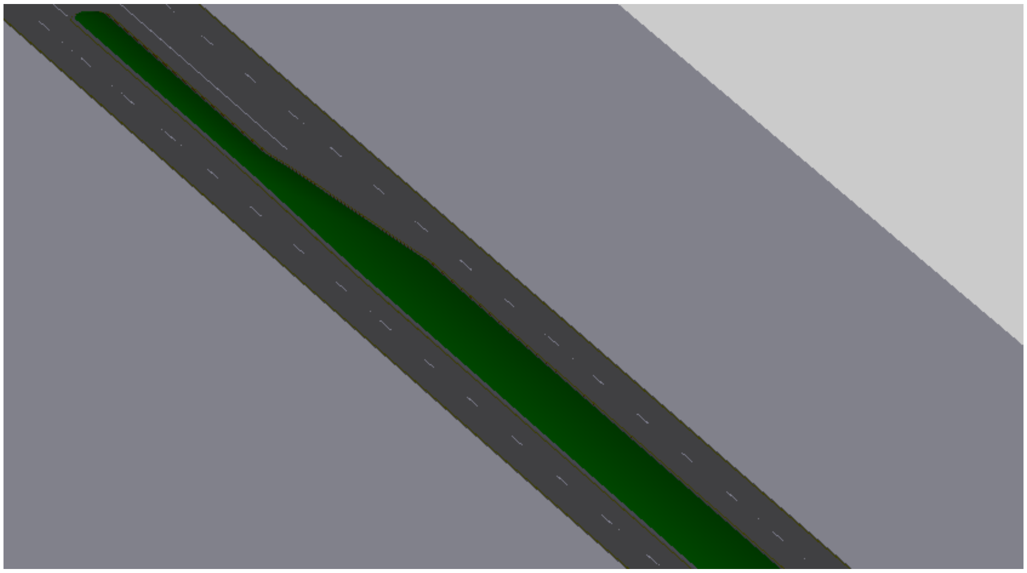

Fig. 80 Snapshot of Visual Database (5)

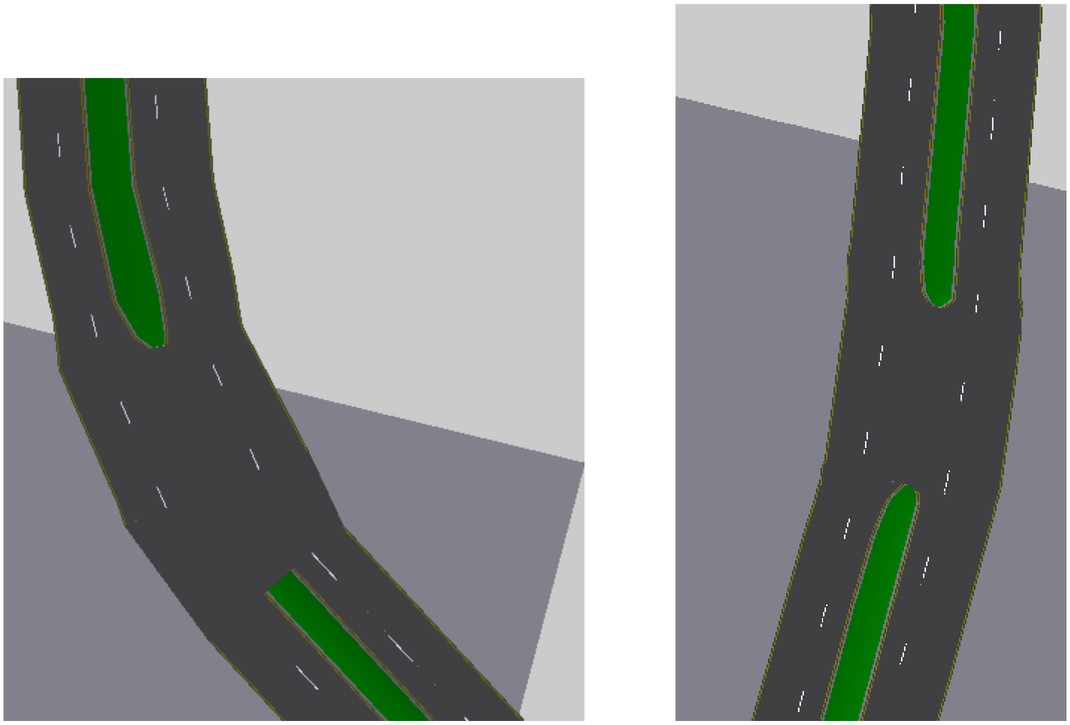

Fig. 81 Snapshot of Visual Database (6)

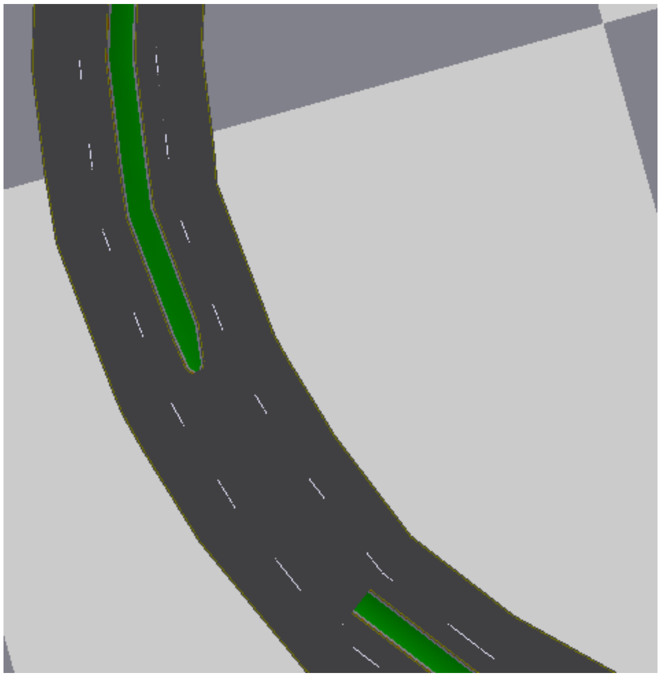

Fig. 82 Snapshot of Visual Database (7)

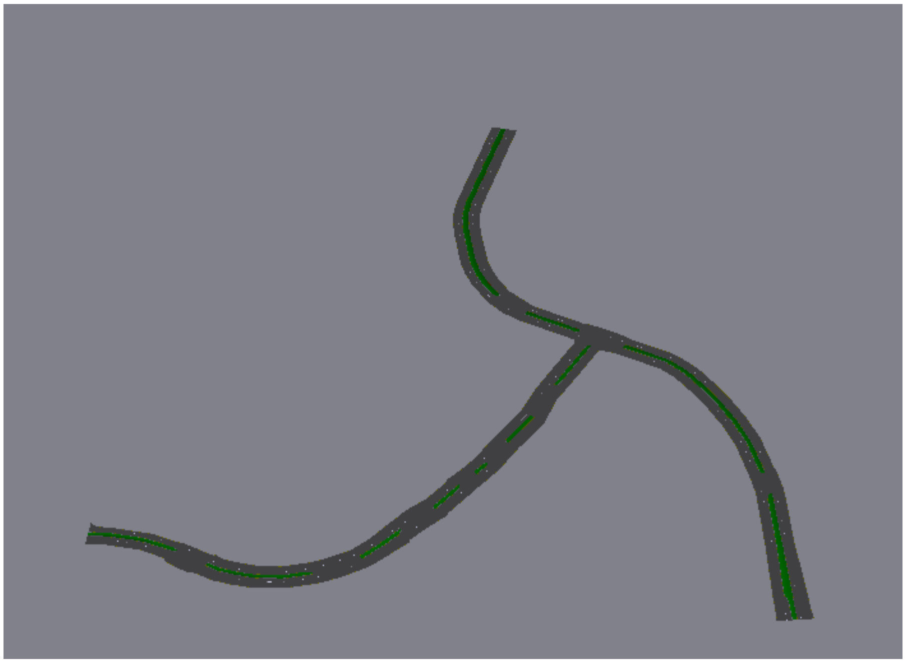

Fig. 83 Snapshot of Visual Database (8)

5.8 Limitations of the Proposed Methodology

5.8.1 Image Segmentation Problems

The proposed method first segments an image within the feature space by searching local modes. Then these segmented classes' average saturation values are compared. The ones with a low saturation are classified as pavement areas and the ones with a high saturation are classified as non-pavement areas. This procedure heavily relies on the fact that is there is a significant difference in saturation between pavement areas and non-pavement areas. In reality, there are some roads where saturation value does not change much from pavement areas to non-pavement areas, such as in the following figures. The first one is the original image and the second is the corresponding saturation image.

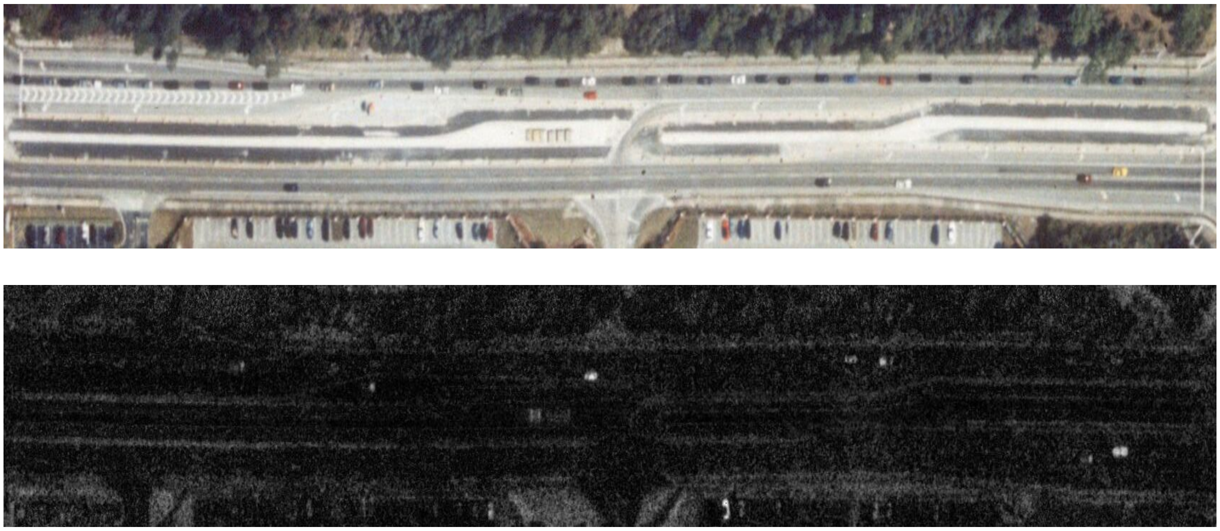

Fig. 84 A Road Image with a Relatively Uniform Saturation Value

In this situation, the classes formed by the mean-shift cluster algorithm may not be useful. The following figure is the segmentation result by the mean-shift clustering algorithm.

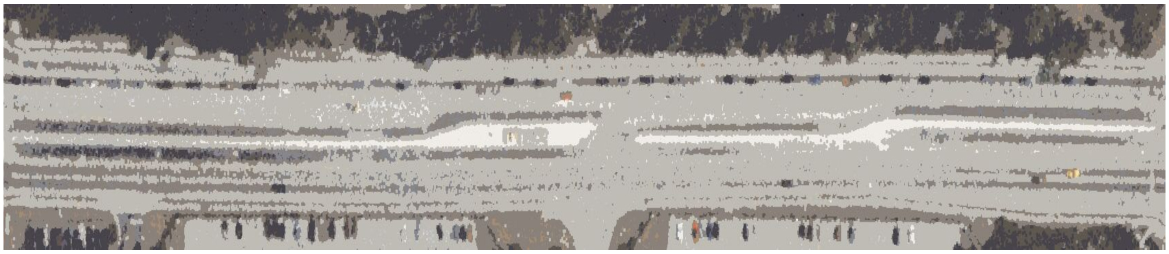

Fig. 85 Segmentation Results by the Mean-Shift Clustering Algorithm

Additionally, even though the proposed method addresses shadows and occlusions, a large amount of them may still cause considerable problems to road segmentation. This happens to narrow roads with large trees. In these areas, large amount of shadows and occlusions make road boundaries not easily recognizable. The following figures show an example of this. The first image is the original image, the second one is the segmentation result by the mean-shift algorithm, and the third one is the saturation image.

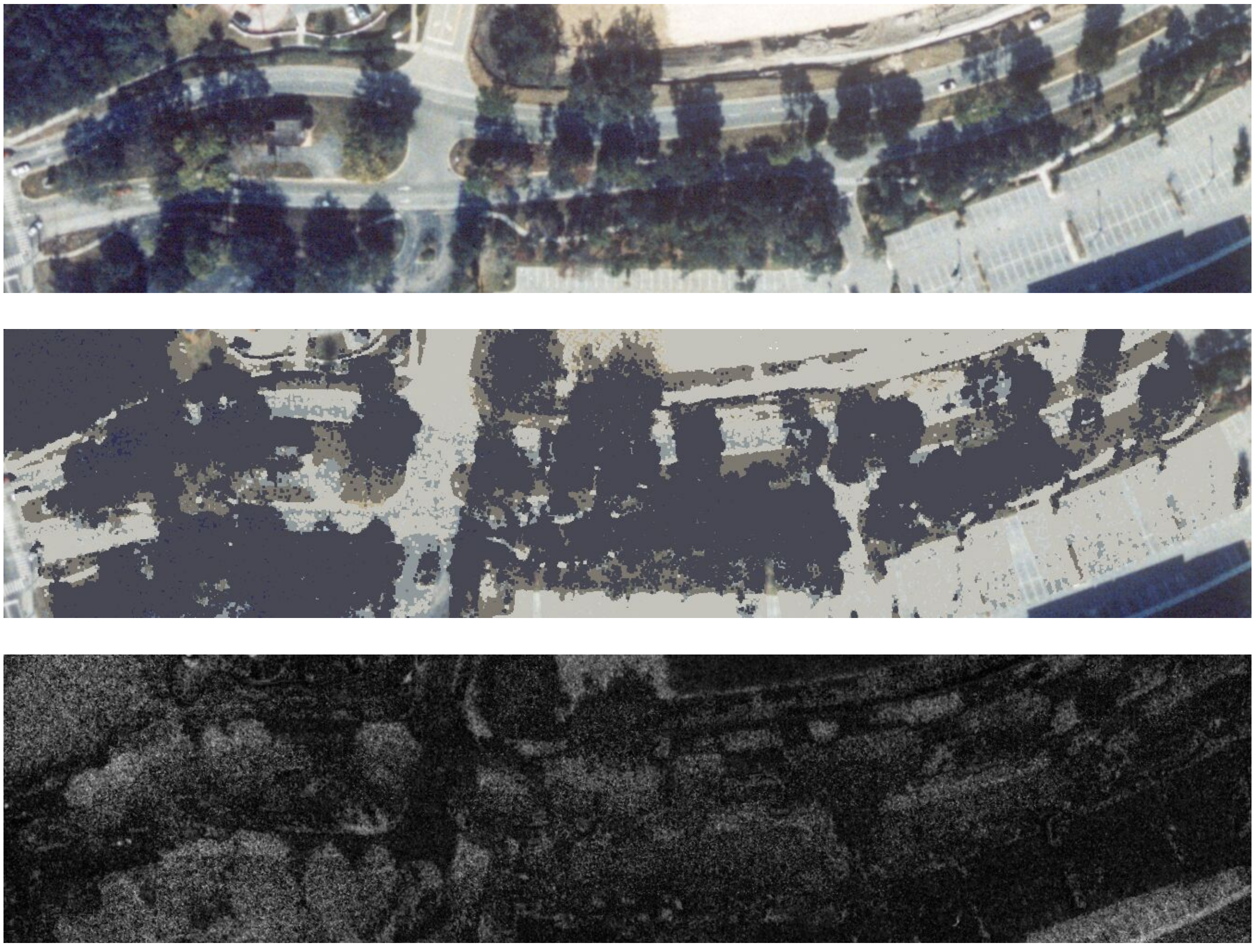

Fig. 86 Road Images with a Large Amount of Shadow and Occlusion

Both problems attribute to the major variety of road appearances. In this situation, more manual user input is required.

5.8.2 High Resolution Elevation Information Not Available

Elevation often changes along a road. It would be more realistic, if 3D road models have elevation information. However, there is no elevation information in aerial photos. The USGS provides digital elevation models (DEMs) that consist of terrain information for ground positions at regularly spaced horizontal intervals. The USGS produces five different digital elevation products. Although all are identical in the manner the data are structured, each varies in sampling interval. The one with the highest resolution is the 7.5-Minute DEM with 30X30 meter data spacing. This resolution can provide a rough reference of how elevation changes along a road, but there is a huge gap between road modeling and DEMs in terms of resolution. Therefore, it is difficult to have accurate elevation information.

Another elevation is called superelevation, which is used on road curves to provide centripetal force. This type of information is not possible to extract from aerial photos. Because the amount of superelevation needs a very high rate of accuracy, which cannot be extracted in images.

5.8.3 Less Information for Building a Complete 3D Road Modeling

A complete 3D road model should include not only road geometries, but also traffic signals, including traffic lights, traffic signs, and pavement markings. Traffic lights and signs are not obtainable from aerial photos due to the view angle. The dimensions and positions of pavement markings are only a few pixels. It is difficult to accurately extract their dimensions from images. In such a case, it is better to explore some existing documents, such as AutoCAD and MicroStation files. This topic will be discussed later.

CHAPTER SIX: FUTURE WORKS

In the future, there are two major tasks. The first is to build a GUI application that includes the proposed method. This application should give modelers the access to adjust results after each step. This software is very important, because the accuracy of image recognition algorithm is limited and manual adjustment is inevitable. It is important to let modelers screen results after each step.

The second task is to explore the possibility of combining other geographical information to have more information for road modeling. When each road is constructed, there must be a large amount of construction plans, in forms of either AutoCAD or MicroStation files. These two software applications have very good tools in building roads, including road boundaries, equipments near roads, road pavement markings, and so on. The purpose of using them is not for 3D modeling for interactive simulation, instead it is for drawing construction plans. These plans have adequate information of what needs to be constructed. These plans are interpreted by constructors during construction and it should be a straightforward procedure. The proposed future work is to develop an algorithm to automatically interpret these plans and to convert them into 3D road models. The models based on this information must be very accurate and, more importantly, very complete. The difficulty with this algorithm is how to translate these files into simulation compatible road databases and visual databases. If this algorithm is determined, it will be a huge advance in the field of geo-specific road modeling for driving simulation.

CHAPTER SEVEN: CONCLUSION

In this research, the problem of how to quickly develop 3D road databases for driving simulation is addressed. A method based on image processing techniques is proposed. This method does not extract road center line information from images, instead it matches aerial images with the USGS DLG data, which is a collection of road center line point coordinates. Doing this can exclude most unwanted contents from consideration. Before using USGS DLG data, it has to be matched with aerial photos in order to detect any distortion. The two considered distortions are linear distortions and non-linear distortions. Linear distortions are relatively easy to detect. Non-linear distortions need more manual work. After this process, significant non-linear distortions are not found and linear distortions are simply translation and scaling.

In the step of road segmentation, six methods have been tested. They were edge grouping, texture-based segmentation using histogram intersection techniques, feature-based segmentation using EM method, region-growing method, active contour method, and the mean-shift clustering method. The edge grouping method does not work, because there are too many linear objects in road areas. The texture-based segmentation method needs a database of sample images. When sample images are close to test images, good results can be obtained. If this can not be met, the algorithm will be affected. The feature-based segmentation using EM method does not have this problem, but it is computation intensive. The region-growing algorithm leaves a number of small objects to post processing. Even though each object is chromatically uniform, their irregular shapes make recognition very difficult. The active contour method also fails because too many linear objects are close to each other.

The chosen method is the mean-shift clustering algorithm for image segmentation. This method works within a feature space, where local modes are searched. An advantage of this method is that segmented objects are connected in the feature space. This is beneficial to classifying segmented classes. Formed classes in this step are categorized into two kinds, pavement and non-pavement by comparing their average saturation values. Conditional morphological operations play an important role in improving segmentation results. The proposed road segmentation method achieves high accuracies that are at least 85%. However, the segmentation results need to be adjusted by modelers. A software application has been developed for this purpose.

In the phase of creating 3D road models, repeatable cross sections are identified by using the LBG vector quantization algorithm. Afterwards, RDBs can be developed automatically. During the building of visual databases, polygons are drawn, each of which is mapped a uniform texture. Five considered textures are pavement, white marker, yellow marker, concrete, and grass.

Compared with manually creating geo-specific road databases, the proposed system has the following advantages:

1. The utilization of the USGS DLG data eliminates efforts in searching roads in aerial photos manually;
2. The accurate road segmentation results significantly reduce efforts in delineating road boundaries;
3. The automation of identifying repeatable cross sections helps organizing data for non-uniform cross-sectional roads;

4. The automation of texture mapping solves the problem of texture mapping for irregular shaped objects;
5. Adjustable parameters, such as lane width and marker width, make it easy to modify existing road models.

Due to the high accuracy in road segmentation, the proposed method can significantly reduce many hours for creating road databases for driving simulation. However, it has three limitations. The first is that the proposed road segmentation algorithm cannot handle some roads, where saturation does not change from non-pavement areas to pavement areas and there are large shadow/occlusion areas. This problem is difficult to resolve because of the major variety of road appearances. The second is that the elevation information cannot be extracted from aerial photos. The USGS DEM can provide a rough elevation profile of a certain area, which may be used only to approximate road elevations. The third limitation is that it is not possible to extract all the information needed to create road models. Objects, such as pavement markings or traffic signs, can not be accurately obtained from images.

A future goal of this research is to develop a software application with GUI. Its functionalities will include road segmentation, road database development, and visual database development. Another future work is to explore the possibility of developing a complete road model, including road geometry, pavement markings, and traffic signs from construction plans in forms of AutoCAD and MicroStation files. Because these files have complete information about how roads should be constructed, there must be adequate information. The challenge is to find out a way to interpret these files.

This research thoroughly analyzes an image understanding based approach for geo-specific road modeling. This approach can take advantage of the details in aerial photos, while it has to deal with the noise. An important contribution, made by this research, is to utilize the USGS DLG information to remove a great portion of noise, irrelevant to road segmentation. This step enables the proposed method to avoid the problem of searching roads in image. Most road extraction methods start with the whole aerial image, while there is a large amount of existing road information. Another uniqueness of this research is automatically identifying repeatable cross sections using a vector quantization algorithm. This method can reduce redundant information in creating road databases for an STS driving simulator system and quantitatively calculate average error. Additionally, this research uses geometric design standards when automatically drawing polygons for road models. This proposed system greatly benefits from existing geographical data and knowledge. In the future, more existing geographical data should be studied.

LIST OF REFERENCES

[1] J. C. Trinder et al "Artificial Intelligence in 3-D Feature Extraction" Worskshop on Automatic Extraction of Man-Made Objects from Aerial and Space Image (ASCONA) 97 P257-266

[2] G. Vosselman et al "Updating Road Maps by Contextual Reasoning" ASCONA' 97 P267-276

[3] M. Barzohar et al "Fast Robust Tracking of Curvy Partially Occlfuded Roads in Clutter in Aerial Images" ASCONA' 97 P277-286

[4]M. Barzohar et al "Automatic Finding of Main Roads in Aerial Images by Using Geometric-Stochastic Models and Estimation" IEEE Transactions on Pattern Analysis and Machine Intelligence. Vol. 18, No. 7, July 1996

[5] A. Gruen and H. Li "Linear Feature Extraction with 3-D LSB-Snake" ASCONA' 97 P287-298

[6] A. Baumgartner et al "Context-Supported Road Extraction" ASCONA' 97 P299-307

[7] S. Hinz, A. Baumgartner et al "Road Extraction Focusing on Urban Areas" ASCONA' 01 P255-265

[8] A. Kararzis et al "A Model-Based Approach to the Automatic Extraction of Linear Features from Airborne Images" IEEE Transactions on Geoscience and Remote Sensing", Vol.39, No.9, Sept. 2001

[9] M. Pesaresi et al "A New Approach for the Morphological Segmentation of High-Resolution Satellite Imagery" IEEE Transactions on Geoscience and Remote Sensing", Vol.39, No.2, Feb. 2001

[10] C. Lipari et al “Geometric Modeling and Recognition of Elongated Regions in Aerial Images” IEEE Transactions on Systems, Man, and Cybernetics, Vol.19, No.6, Nov/Dec. 1989

[11] F. Tupin et al “Detection of Linear Features in SAR Images: Application to Road Network Extraction” IEEE Transactions on Geoscience and Remote Sensing”, Vol.36, No.2, Mar. 1998

[12] F. Tupin et al “Road Detection in Dense Urban Areas Using SAR Imagery and the Usefulness of Multiple Views” IEEE Transactions on Geoscience and Remote Sensing”, Vol.40, No.11, Feb. 2002

[13] I. Laptev et al “Automatic Extraction of Roads from Aerial Images based on Scale-Space and Snakes” Technical report CVAP240, ISRN KTH NA/P--00/06--SE. Department of Numerical Analysis and Computing Science, KTH (Royal Institute of Technology), S-100 44 Stockholm, Sweden, March 2000.

[14] C. Zhang et al "Updating of Cartographic Road Databases by Image Analysis" ASCONA' 01

[15] C. Steger “The Role of Grouping for Road Extraction” ASCONA’ 97 P245-256

[16]C. Steger “An unbiased detector of curvilinear structures” Pattern Analysis and Machine Intelligence, IEEE Transactions on ,Volume: 20 , Issue: 2 , Feb. 1998 Pages:113 - 125

[17] F. Wang et al “A Knowledge-Based System for Highway Network Extraction” IEEE Transactions on Geoscience and Remote Sensing”, Vol.26, No.5, Sept. 1988

[18] Kass, M., Witkin, A. and Terzopoulos, D. “Snakes: Active contour models” International Journalof Computer Vision 1(4):321–331. 1988

[19] W. M. Neuenschwander "Ziplock Snakes" International Journalof Computer Vision 25(3), 191-201 (1997)

[20] D. Comaniciu and P. Meer "Robust analysis of feature spaces: Color image segmentation" In International Conference on Computer Vision and Pattern Recognition, Pages 750{755. San Juan, Puerto Rico,1997.

[21] R. I. Hartley and A. Zisserman "Multiple view Geometry in Computer Vision" London, U.K.: Cambridge Univ. Press, PP 12, 69-74, 2000

[22] http://www.adobe.com/products/photoshop/main.html, accessed on 11/1/2004

[23] http://edc.usgs.gov/products/map/dlg.html#description, accessed on 11/15/2004

[24] http://edc.usgs.gov/products/aerial/doq.html, accessed on 11/15/2004

[25] M. Baatz and A. Schape, "Multiresolution segmentation: An optimization approach for high quality multi-scale image segmentation," in Angewandte Geographische Informationsverabeitung. XII. Beitrage zum AGIT-Symp. Salzburg 2000, T. Strobl, T. Blaschke, and G. Griesebner, Eds., Karlsruhe, 2000, pp. 12–23.

[26] C. Carson, S. Belongie, H. Greenspan, and J. Malik "Blobworld: Image Segmentation Using Expectation-Maximization and Its Application to Image Querying" IEEE Transaction on Pattern Analysis and Machine Intelligence, Vol. 24, No. 8, August, 2002

[27] D. J. Williams and M. Shah "A Fast Algorithm for Active Contours and Curvature Estimation" CVGIP: Image Understanding, Vol. 55, No. 1, January, pp. 14-26, 1992

[28] http://www.bentley.com/en-US/Products/MicroStation/Overview.htm, accessed on 6/12/2005

[29] http://usa.autodesk.com/adsk/servlet/index?siteID=123112&id=2704278, accessed on 6/12/2005

[30] T. Gevers, “Image Segmentation and Similarity of Color-Texture Objects” IEEE Transactions on Multimedia, Vol. 4, No. 4, December, 2002

[31] J. Li, J. Randall and L. Guan “Perceptual Image Processing for Digital Edge Linking”, Electrical and Computer Engineering, 2003. IEEE CCECE 2003. Canadian Conference on, Vol. 2, May 4-7, 2003. Pages: 1215-1218

[32] A. R. Weeks “Fundamentals of Electronic Image Processing” SPIE/IEEE Press, 1996, ISBN 0-8194-2149-9

[33] M. J. Swain and D. H. Ballard “Color Indexing” International Journal of Computer Vision, Vol. 7, No. 1, pp. 11-32, 1991

[34] A. K. Shackelford, C. H. Davis “A Combined Fuzzy Pixel-Based and Object-Based Approach for Classification of High-Resolution Multispectral Data Over Urban Areas” IEEE Transaction on Geoscience and Remote Sensing, Vol. 41, No. 10, October, 2003

[35] K. Sayood 1996 “Introduction to Data Compression”, Morgan Kaufmann Publisher, Inc., San Francisco, CA

[36] A. R. Weeks “Fundamentals of Electronic Image Processing” SPIE/IEEE Series on Imaging Science & Engineering, 1996

[37] C. Früh and A. Zakhor, "Fast 3D Model Generation in Urban Environments," Int. Conf. Multisensor Fusion and Integration for Intelligent Systems, Baden-Baden, Germany, August 2001