"Evaluation of the use of high-resolution satellite imagery in transportation applications"

Final Report
Duration of the Project: May 2002- May 2003

Prepared for
Northland Advanced Transportation Systems Research Laboratories

Rocio Alba-Flores, PI
Department of Electrical and Computer Engineering
University of Minnesota Duluth
271 MWAH, 10 University Dr.,
Duluth, MN 55812

March 2004

CONTENTS

ACKNOWLEDGMENT OF SPONSORSHIP

This work was sponsored by the Northland Advanced Transportation Systems Research Laboratories, and was conducted in the Department of Electrical and Computer Engineering of the University of Minnesota Duluth. The duration of the Project was: May 2002- May 2003

ABSTRACT

Satellite images and remote sensing data provides global information that when compared to traditional methods of data collection is able to provide larger volume of data, acquire information at difficult to reach regions, and monitor areas and events non-intrusively. In the last decade, with the advance of technology, and the development of more sophisticated remote sensing sensors systems, the use of satellite imagery and remote sensing has opened the fields of exploration and application. Among these new fields is transportation. As this imagery data becomes more readily available, transportation organizations will have to learn to exploit these useful systems, and probably make changes in their current approaches to planning and operations.

Having in mind the importance and potential that satellite imagery could have in transportation applications, the present project was developed. The main goal of this project was to explore the capabilities of using satellite images for transportation applications. First, we wanted to investigate the potential that high-resolution satellite images (IKONOS 1-m panchromatic) could have for accurate detection and identification of vehicles and roadways, and the possible applications in traffic flow, traffic congestion (traffic count, travel speed), and in urban planning. Second we wanted to explore the feasibility of using this high-resolution imagery to detect ice and snow on the highways.

In this project two sets of programs, based on image processing techniques, were developed. The first set contains algorithms that detect the roadways. The second set of algorithms detects, classifies, and counts the vehicles on the roadways. For both applications, IKONOS 1-m panchromatic imagery was used. The MATLAB™ software was used for the implementation of the programs. A meticulous literature review was performed to identify the current limitations in technology to be able to use remote sensing data to detect snow and ice on the roadways.

CHAPTER 1. Introduction and Research Approach

1.1 Introduction

Satellite imagery and remote sensing technology has been used and studied in a variety of fields since the decade of the seventies. It has been used successfully in such fields as meteorology, geographic information systems, agriculture, mining, disaster management, forestry, etc. For example, remote sensing has been used to detect the different types of vegetation, wetlands, soil types, oil spills, to do inventory of endangered species habitats by determining the extent of vegetation destroyed, weather prediction, etc.

In the last decade, with the advance of technology, and the development of more sophisticated remote sensing sensors systems, the use of satellite imagery and remote sensing has opened the fields of exploration and application. Among these new fields is transportation. The application of satellite imagery to transportation represents a relatively new challenge that is quickly gaining interest. It is not known what benefits this technology will have to transportation, and the potential benefits should be explored.

Some attractive aspects of using satellite imagery, and remote sensing, are the ability to provide large volume of information, acquire information at difficult to reach regions, and to monitor areas and events non-intrusively. Because the potential capabilities that satellite imagery and remote sensing have to provide information that can improve the efficiency and safety of transportation, on the Fall 1999, the National Consortia for Remote Sensing in Transportation was created [1]

Having in mind the importance and potential that satellite imagery could have in transportation applications, the present project was proposed. The main objective of the project was to explore the capabilities of using satellite images for transportation applications. In particular, two goals were established. First, we wanted to investigate the potential that high-resolution satellite images (panchromatic 1-m resolution from IKONOS) could have for accurate detection and identification of vehicles and roadways, and the possible applications in traffic congestion (traffic density, flow, and speed), and in urban planning. The second goal was to explore the feasibility of using this high-resolution imagery to detect ice and snow on the highways.

1.2 Summary of Literature Review

With the release to the public and the commercial availability of high-resolution satellite imagery, satellite systems have become an attractive option for gathering data for transportation applications. It is important to study and understand the characteristics of the data that can be obtained from the actual satellite systems and identify their limitations. For transportation applications, and in particular for traffic related issues, the spatial resolution and the temporal covering are two very important parameters to be considered. For example in [2][3][4] [5] the authors provide tables and figures proposing technology requirements in terms of spatial, spectral, and temporal resolution needed to distinguish specific urban and suburban attributes using space-borne technology. Specifically, the authors report that for traffic applications (count studies, congestion, velocity studies) a spatial resolution in the order of 0.25-1m, and

panchromatic visible spectral resolution are necessary. In [6] the authors performed a study of the interpretation and detection capabilities for satellite images. They provide some feel for the spatial resolution requirements to distinguish natural and man-made objects such as trees, shrubs, rocks, street, bridges, etc. The authors reported that for the detection of paved roads, at least a 2-m spatial resolution is necessary. The temporal resolution is a serious limitation since up to now, the revisit period of the satellites ranges from 16 to 26 days, however with the launch of several commercial satellite systems, it is expected that in the near future, the revisit period could be in the order of 1-3 days. [7]

In relation to ice and snow detection, besides the meteorological weather systems for predicting snow precipitation that have been used for many years, the literature reports a variety of studies to monitor snow cover over large areas. In these studies, visible, infrared, and microwave radiometers have been used. For example in [8] passive microwave radiometers are utilized to detect and map snow areas; in [9] SAR is used to discriminate between snow and ground objects. In [10] radiometers with infrared sensors have been used to distinguish between cloud top or snow cover on the ground. However, in all these studies, the particular applications require spatial resolution in the order of hundred of meters, or even kilometers. For the specific case of transportation applications, a higher resolution is needed to be able to identify snow build up on road segments. In [3] the authors report that a spatial resolution in the order of 0.25-0.5m will be needed to be able to detect ice or snow on highways. Assuming that in the near future, with the fast pace of technology, this spatial resolution could be achieve, the temporal coverage will be the main impediment to apply satellite imagery in the detection and prediction of ice and snow on road segments.

1.3 Research Approach

The activities for the development of the project were organized in four phases. In phase 1, a Data Base containing high-resolution satellite images (1-m panchromatic from IKONOS), including scenes of highways, was formed. In phase 2, using the collected images from phase 1, an algorithm to detect and extract the highways and roads was developed. In phase 3, algorithms to detect, count, and classify vehicles on those highways were implemented. In phase 4, the feasibility of using this high-resolution imagery to detect ice and snow on the highways was investigated. In the following chapters, each phase is explained in detail, and the results are presented.

CHAPTER 2 The Data Base

In phase 1, a Data Base containing high-resolution imagery with scenes of highways was formed. The type of images selected were 1-m panchromatic from the IKONOS satellite. Several companies [11][12][13] that sell satellite images were contacted. The commercial cost of this type of images was investigated. From one of the vendors, INTEC Americas [12], the cost for an IKONOS 1-m panchromatic image is in the order of $30 dollars per/sq. Km. The typical area of the images is around 10 x10 Km, this means that the cost of each image is about $300.00. However, the prices vary in accordance to the place and date. Recent images are more expensive, and for a specific region the cost could be higher. The delivery time varies, and could take between 3 to 6 weeks. The user cannot always specify a precise day and time for the desired images. The prices and conditions from vendors [11] and [13] were very similar to [12].

Because the cost of these 1-m IKONOS images is still high, and actually there is not flexibility in choosing time and day for the desired images, to create our database we decided to select some of the free images available from the websites of those companies. We downloaded about 20 images that we considered useful for our project. In our case the place and date were not very critical, the only characteristic that we were looking for was that the content of the images had views of highways and roads. Figure 2.1 shows some of the images in the database.

(a) Phoenix, AZ

(b) San Jose, CA

(c) Dallas, TX

(d) Twin Cities, MN

Figure 2.1 Test images: IKONOS 1-m panchromatic satellite images

CHAPTER 3 Detection of Highways and Roads

3.1 Extraction of Roadways

In this phase, an algorithm for the extraction of roadways from IKONOS 1-m panchromatic satellite imagery was developed. For the implementation of the algorithm MATLAB™ was utilized. In the following paragraphs, the procedures utilized and developed for the extraction of the roadways are discussed. Experimental results using the algorithm are presented.

To extract the roadways from the images, segmentation techniques were applied. To do this, two approaches were used: edge finding and thresholding. For the edge finding, the Canny and Sobel methods were used, but the results did not meet our objectives. Better results were obtained using thresholding techniques.

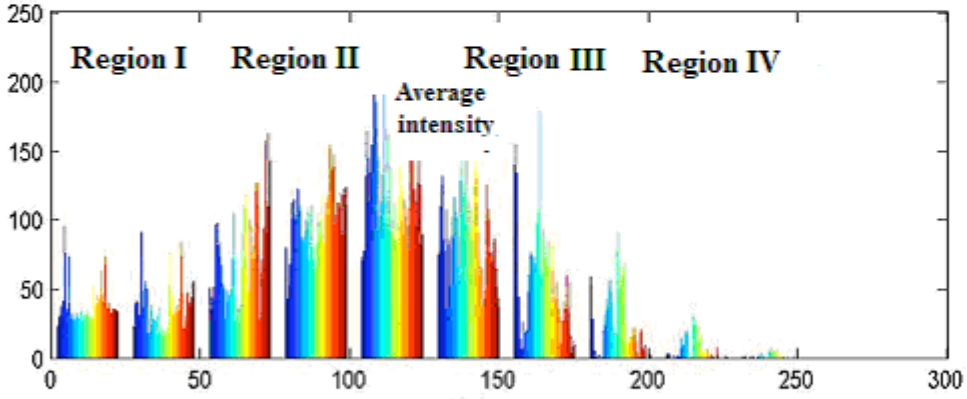
Before applying the segmentation technique, the user has to specify a region of interest, from the original image. The region of interest is a subimage, of rectangular shape, which is selected using the mouse. Working with regions of interest instead of the whole image, helps to obtain better results in the application of our algorithm.

To extract the roadways from the subimages, thresholding techniques are applied. To do this, appropriate threshold values have to be computed, and then segmentation is performed. To compute the appropriate threshold values several problems had to be analyzed. For example: the illumination conditions, the different type of pavement material, the presence of objects such as vegetation, vehicles, buildings, etc. To identify most of the possible problems, the images in the database were studied and analyzed, and several test subimages were extracted from the database. These test subimages presented various conditions under which the performance of our algorithm was evaluated. In the following section the proposed algorithm is described. Then several examples are given to illustrate the results.

3.2 The Algorithm: Detection of Roadways

The algorithm is based on global thresholding. To obtain the desired results, the histograms of several of the scenes in the database containing highways and roads, were analyzed. From the analysis, we divided the histogram in four main regions. Figure 3.1 b) shows these regions. We called Region I, starting from the lowest intensity values to midway of average intensity ***Av*** (***Av*** is the arithmetic mean computed using all the pixel values in the subimage). For most of the cases, the intensity values on this region covers dark objects of the image such as dark vehicles, shadows, etc. The Region II covers intensities that go from approximately midway of the average intensity to average intensity (***Av***), and typically covers dark gray shades objects like highways, trees, etc. The Region III goes from average intensity (***Av***), to midway of highest intensities and covers bright gray objects such as buildings, lane markers, rivers, etc. Region IV goes approximately from midway of highest intensity to highest intensity and generally covers bright objects like bright vehicles, road dividers etc. Figure 3.1 shows a typical highway scene and its histogram.

a) IKONOS 1m with a highway secene



b) Histogram of the image in a)

Figure 3.1 a) Typical scene showing highways and roads in a metropolitan area
b) Hisogram of image in a)

3.3 Computing the Threshold T

From our study of the scenes containing highways, we observed that the range of intensities for most of the roads and highways includes values that are less than the average intensity of the image being analyzed. Based on this, we decided to use the average intensity of each row of our image under study, **I = f(x,y),** to create a vector **V** . Note that the **I** is a matrix, and each element of **I** corresponds to the intensity pixel value of the image under study. Each element in this vector **V** is the average intensity value of the corresponding row of intensities of **I**. The threshold value **T** is selected as the minimum value of **V.**

The algorithm is:

```
V  is calculated by using I
V: for each row i in I
        V[i] = average_intensity[I, i]

T is calculated using V
T = minimum[ V ]
```

T is used to convert the test image **I** to binary image **B1**. A pixel at any position (x y),

if (x y) < T, then this pixel is considered as highway (black = 0)
else it is turned white = 1.

The binary image, **B1,** obtained by **T** is defined as:

```
[r c p] = size [ I ];

V = mean [ I ];
T = minimum [ V ];

 for i = 1:r,
   for j = 1:c,
         if I [ i,j ] < T
             B1[ i,j ] = 0;
        end
        else
             B1[ i,j ] = 255;
        end
   end
 end
```

Using this method to find the threshold **T**, for the test image shown in Figure 3.1a), we obtained **T** = 70.7. Figure 3.2 shows the resulted binary image **B1**.

Figure 3.2 Binary image obtained after thresholding the image in figure 2.2.1 a)

3.4 Morphological Operations

Since we are making any pixel less than **T** as **0** to detect highways, there are possibilities of detecting dark vehicles, shadows, and trees, which have intensity values less than **T** also. In Figure 3.2 can be observed that highways are dark in color. It can also be observed that there are some isolated black pixels that correspond to dark vehicles, shadows, and trees. To minimize the detection of shadows, dark vehicles, and trees, we apply four morphological operations: Clean, Majority, Fill, and Holes, to the binary image **B1**.

Clean: Remove isolated pixels (1's surrounded by 0's).
Majority: Set a pixel to 1 if five or more pixels in its 3-by-3 neighborhood are 1's.
Fill: Fill isolated interior pixels (0's surrounded by 1's).
Holes: Fills the black holes in the binary image.

As a result of these four morphological operations, most of the isolated dark pixels would be eliminated. In the particular case where large areas are covered by trees and vegetation (with intensity values in the range of highways), the possibilities of being eliminated are minimal. This is due to the fact that these large areas are not isolated set of pixels and the morphological

operations, which are restricted to a 3-by-3 block of pixels, would not eliminate them. If we try to increase the size of the block, there are possibilities of missing the highways. Figure 3.3 shows the binary image **B2** obtained after applying the morphological operations.

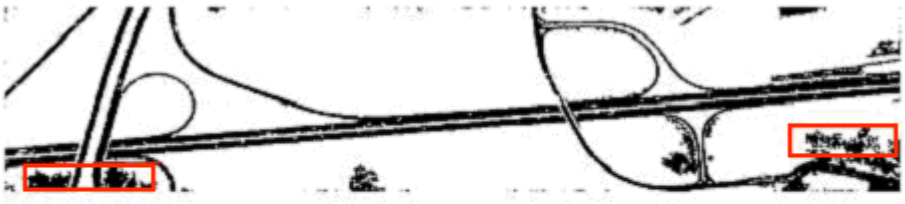


Figure 3.3 Binary image B2 obtained after applying Morphological Operations to figure 3.2

From fig. 3.3 we can observe that in the binary image **B2**, most of the isolated pixels were eliminated. Notice that in this figure the red highlighted rectangles show some vegetation that could not be eliminated by the morphological operations.

3.5 Obtaining Intensity Images of Highways

To obtain the final intensity image (**Final_I**) showing only highways, pixels with value **0** in **B2** are considered. If a pixel has value 0 in **B2**, then the intensity of the corresponding pixel in the original test image **I** is copied to **Final_I**. In this way, only the detected highways will be copied from **I** to **Final_I** using **B2**. Figure 3.4 shows the final image obtained by the algorithm 'Detection of Highways'. The **Final_I** is obtained from **B2** and **T** as:

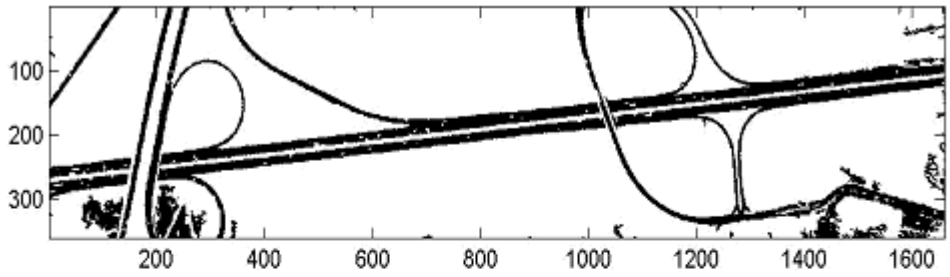
```
[row column page] = size [B2];

for i = 1:row,
  for j = 1:column,
    if B2 [i,j ] ==  0
      Final_I [i,j] = I [i,j];
    else
      Final_I [i,j] = 0;
    end
  end
end
```

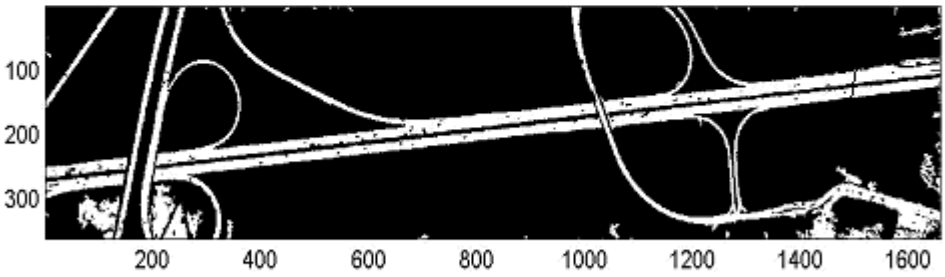
Figure 3.4 Intensity Image obtained after applying the algorithm to detect highways to the image in figure 3.1a)

3.6 Detecting Highways with Vehicles

In the previous section, no vehicles on the highways were detected in the resulting image, Final_I. In this work, we were also interested in detecting the highways with vehicles. To achieve this, the complement of the binary image **B2**, which was obtained while detecting highways without vehicles, is used. Let the complement of binary image **B2** be **B3**. Figure 3.5 shows the binary images **B2** and **B3**.



a) Binary Image B2



b) Complement of B2

Figure 3.5 a) Binary Image B2
b) Complement of B2

From image 3.5b) we can observe that the vehicles on the highway are dark holes. To detect the vehicles, these holes are filled with white pixels. In this way, we get a binary image B4, which contains only highways and vehicles in white color and remaining parts of the image in black color. Figure 3.6 shows the binary image B4 obtained by filling dark holes in B3.

Figure 3.6 Binary Image B4

3.7 Obtaining Intensity Images of Highways with Vehicles

To obtain the final intensity image, **Final_I**, (highways with vehicles), only pixels with value 1 in **B4** are considered. If a pixel has value 1 (white) in **B4**, then the intensity of the corresponding pixel in the original test image **I** is copied to **Final_I**. In this way, the detected highways with vehicles will be copied from **I** to **Final_I** using **B4**. Figure 3.7 shows the final image obtained by the algorithm 'Detection of Highways with Vehicles'. The algorithm is as follows:

```
[row column page] = size [B4];

for i = 1:row,
  for j = 1:column,
    if B4 [i,j ] ==  0
      Final_I [i,j] = I [i,j];
    else
      Final_I [i,j] = 0;
    end
  end
end
```

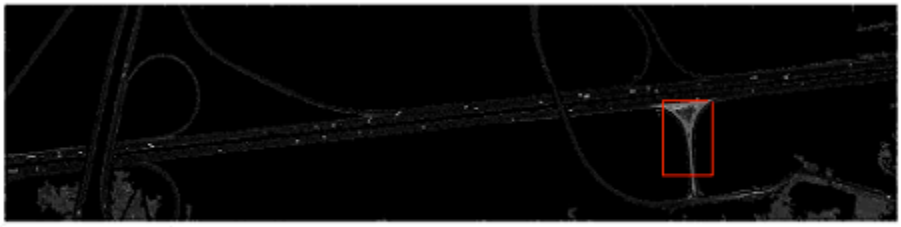


Figure 3.7 Highway with Vehicles

In Figure 3.7, the region highlighted with the red rectangle is not part of the highway, and it is not a vehicle either. The highlighted area is a black loop formed by the highway in image **B3**. While filling the black holes with white color to detect vehicles, the loop was also filled. Since the loop was a white hole in image **B4**, it was detected in the final image.

3.8 Results and Analysis

To show the results of our algorithm, two examples are given. The first example shows a highway in France, and the second one shows a highway in Baghdad. Both are IKONOS 1-m panchromatic satellite images obtained form the world wide web [11][12][13].

3.8.1 Image: Highway in France

Figure 3.8 shows an IKONOS image with a scene of highways (a highway in France). It also shows the histogram of the image. The intensity values vary in this image from 20 to 200 approximately. The intensity values of highways vary approximately from 40 to 100. The threshold **T** obtained by using the algorithm 'Calculating Threshold T' (section 3.3) is T = 99.15. Figure 3.9 shows the results of the algorithm (sections 3.3 to 3.5), where the binary and the graylevel images are presented.

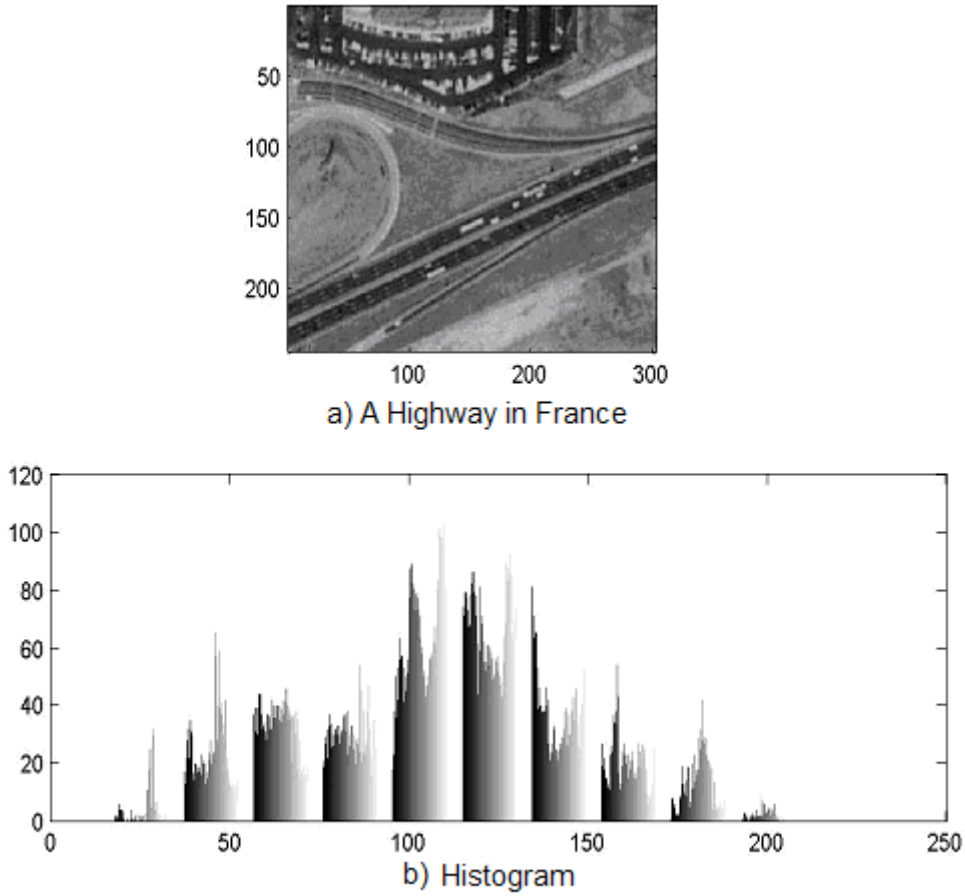


Figure 3.8 a) A Highway in France b) Its Histogram

From figure 3.9 we can see that all the highways were detected but small parts of barren land (part highlighted with red rectangles in Figure 3.9b were also detected. Barren land was detected because it had same intensity values as the highway. Also notice that the area highlighted by blue lines, a parking lot, was also detected.

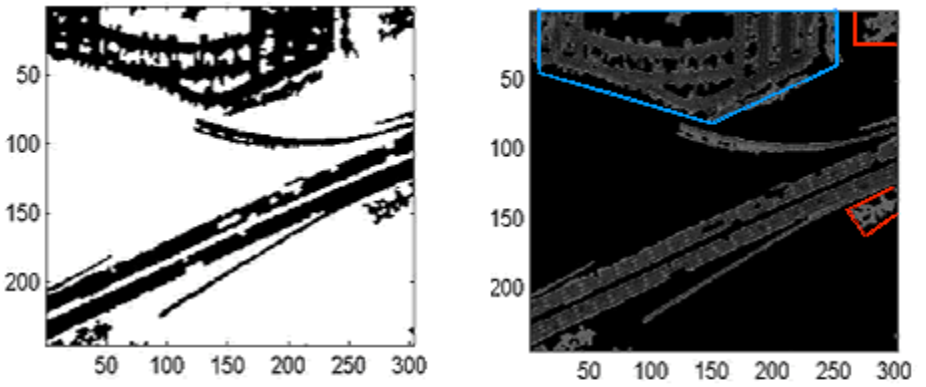


Figure 3.9 Results of applying the algorithm to detect the highways
a) The Binary Image b) The Intensity Image

Figure 3.10 shows the binary image **B4** and the highways with vehicles (sections 3.6 and 3.7). The detection of the highways is accurate but not all the vehicles on the highway were detected. The vehicles marked with red rectangles in figure 3.10a were not detected. The areas highlighted by the blue rectangles in figure 3.10b correspond to the vehicles marked in red in figure 3.10a. These marked vehicles were not detected because to detect the vehicles we filled the black holes in the binary image **B4**, but the red marked vehicles in **B4** are not holes, rather they are connected with regions that are not highway. So, these regions are not filled with white color and are not detected as vehicles. In this example also the parking lot with the vehicles on it was detected (red highlighted area in figure 3.10b).

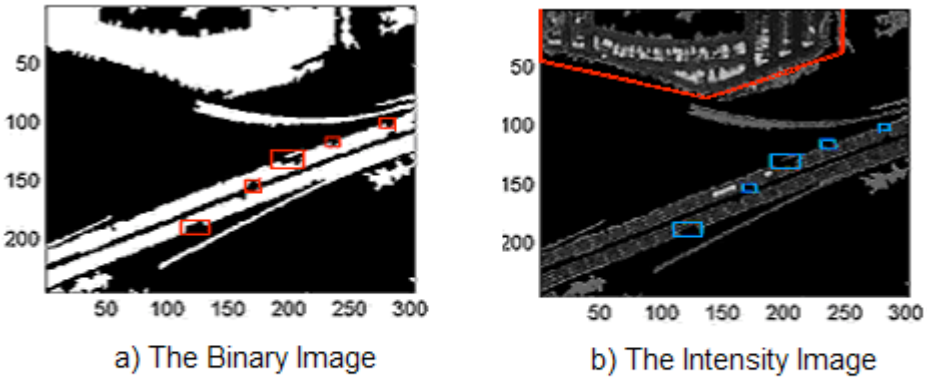


Figure 3.10 Detecting the Highways With Vehicles
a) The Binary Image B4 b) The Intensity Image

3.8.2 Image: Highway in Baghdad

Figure 3.11 shows a highway in Baghdad and its histogram. The threshold value **T** is calculated by the algorithm 'Calculating Threshold T' (section 3.3), and had the value T = 111.4.

Figure 3.12 shows the resulting binary an intensity images obtained after applying the algorithm 'Detection of Highways without Vehicles' (sections 3.3 to 3.5).

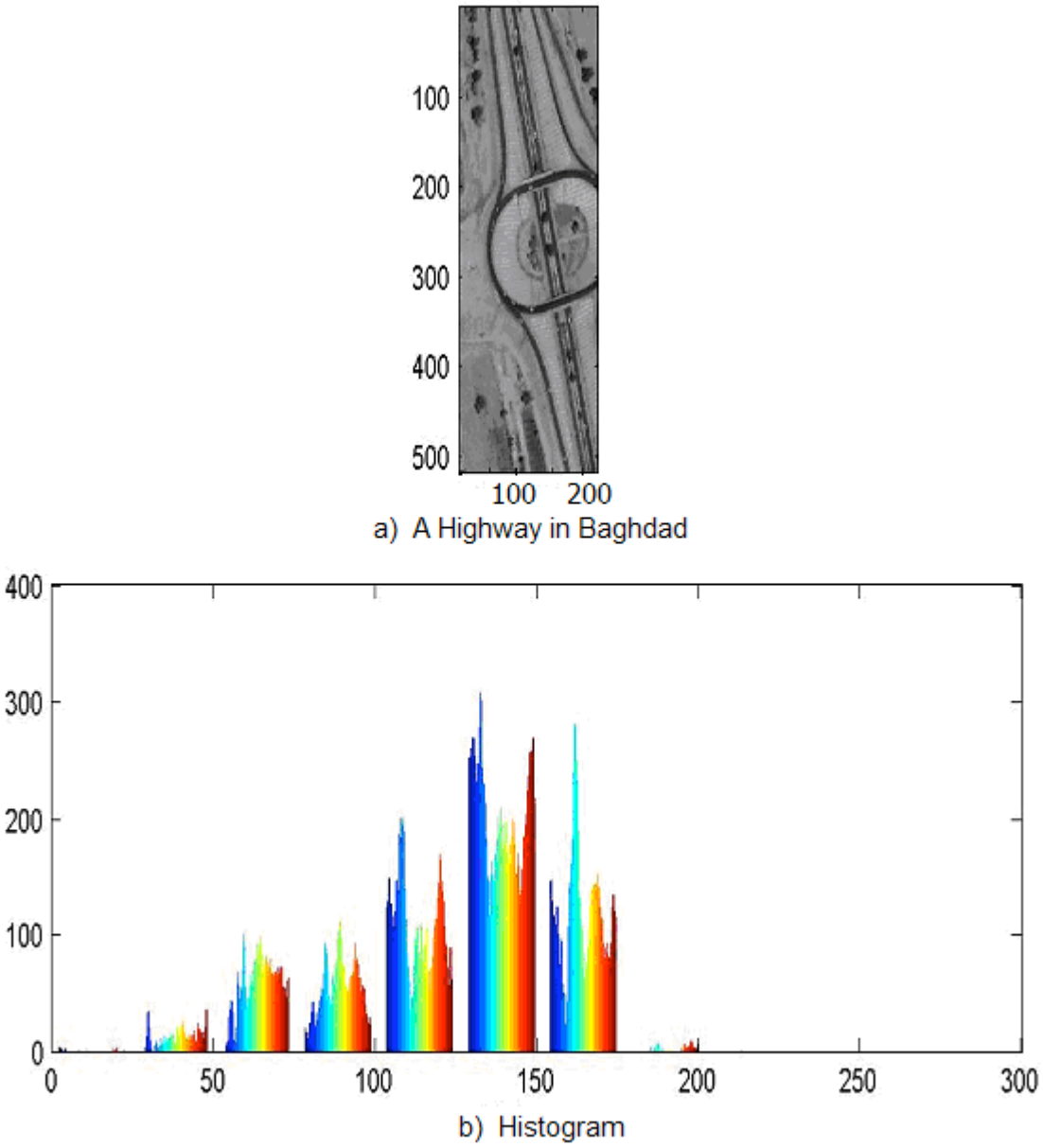


Figure 3.11 a) A Highway in Baghdad b) Its Histogram

From Figure 3.12 b) we can say that all the highways in the test image were detected. The region highlighted by the red rectangle corresponds to barren land, and the areas enclosed by the blue rectangles are trees and vegetation. The reason for not being able to avoid the detection of trees and barren land was explained in section (3.4)

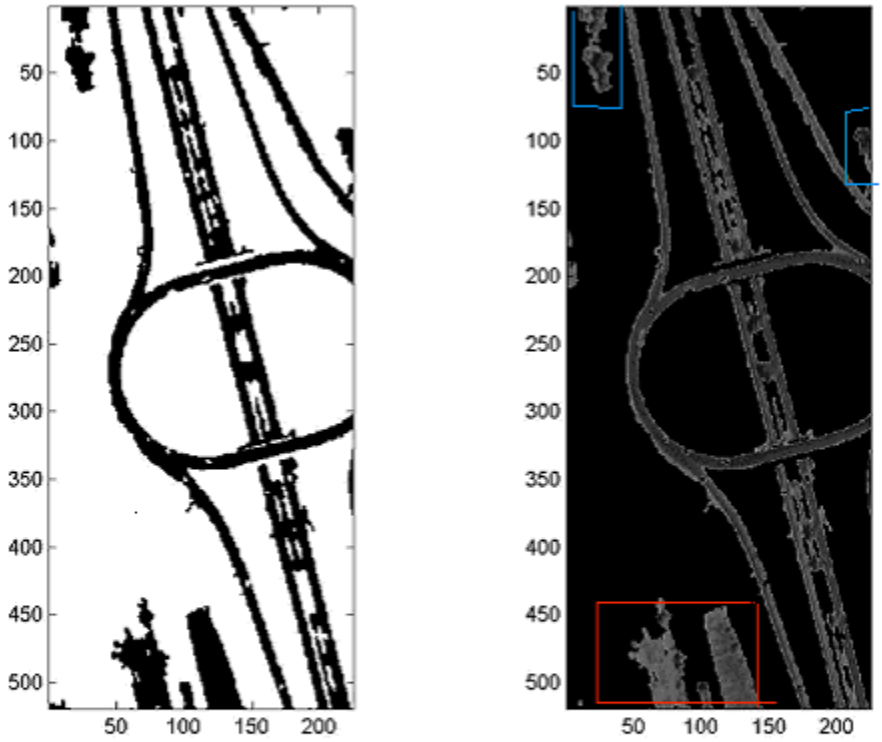
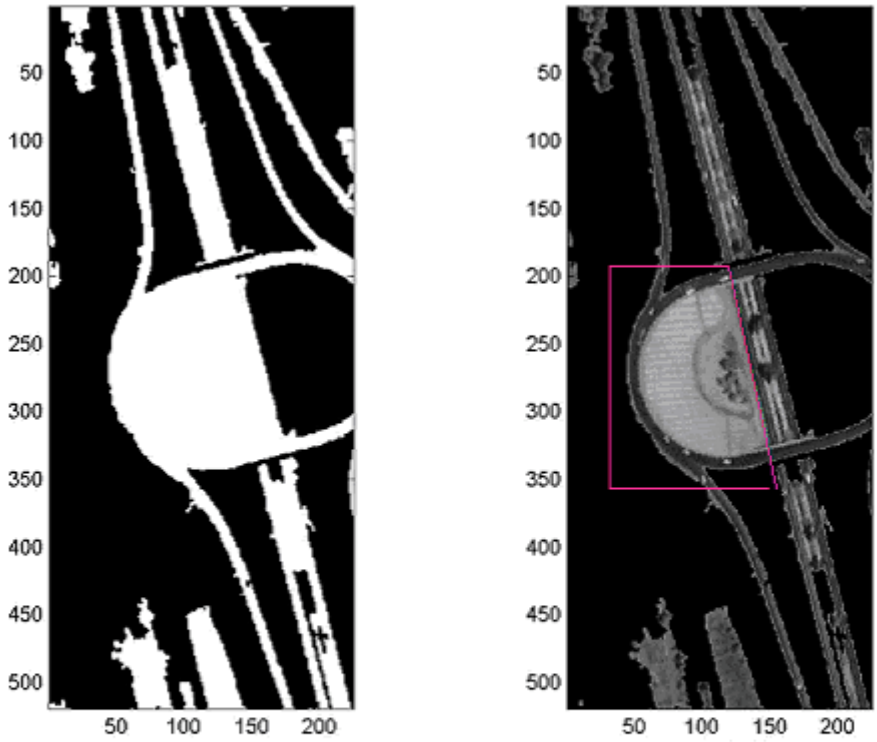


Figure 3.12 Results of applying the algorithm to detect highways
a) The Binary Image b) The Intensity Image

Figure 3.13 shows the highways with vehicles detected by the algorithm 'Detection of Highways with Vehicles'. In this example all the highways and vehicles were detected. The bright region highlighted by red color in Figure 3.13b) is not a highway, nor a vehicle. Notice that this region was not detected while detecting highways without vehicles (see Figure 3.12). The reason for detecting the highlighted region, while detecting the highways with vehicles, was explained in Section 3.7.



a) The Binary Image b) The Intensity Image

Figure 3.13 Detecting the Highways With Vehicles
a) The Binary Image b) The Intensity Image

CHAPTER 4 Detection of Vehicles

4.1 Introduction

Transportation infrastructure of highways, streets, bridges, and the number of vehicles, including cars, buses, trucks and trains are increasing rapidly. It is necessary to use the existing facilities intelligently to provide better transportation facilities. Therefore, it is required to automate the management of traffic flow by intelligent traffic control and traffic guidance systems. Traffic-related data play an important role in urban and spatial planning, e.g., for road planning and for estimation of air and noise pollution. Therefore, an algorithm that automatically detects and counts vehicles from satellite images would effectively support traffic-related applications [14].

Different techniques like installing cameras at fixed locations, weigh-in motion sensors on the pavements are being used for vehicle detection. Though they are well developed systems, they have disadvantages like cameras cannot observe the spatial progression and movement of traffic beyond the field of view [15], sensors can be affected by weather and traffic stress, maintenance of sensors may reduce the pavement lifetime etc. Using satellite images for vehicle detection can eliminate the above mentioned problems. They can cover wide areas and images of an area can be taken frequently. But, using satellite images also have problems like, atmospheric conditions which affect the visibility of vehicles in the images, daytime images provide better information that at night, etc.

Spatial resolution is an important aspect to be considered while using the satellite images for vehicle detection. Satellite images are available in different resolutions. Images with high spatial resolution provide detail information of an area. Many commercial satellites like IKONOS, QUICKBIRD, and IRS, have been launched in recent years. These satellites are capable of providing different spatial resolutions [11][12][13]. Very high spatial resolutions are in the order of 1-meter.

The aim of this research is to develop efficient algorithms for automatic detection, classification and counting of the vehicles from high-resolution satellite images. The images used for this project are 1-m panchromatic images from the IKONOS satellite. Two different Image Segmentation Algorithms, which are based on Multiple Thresholds and Otsu Threshold, were developed for vehicle detection. These algorithms were tested on several images and the results were analyzed to determine which algorithm gives better results under which conditions.

In Section 4.2 we provide a literature review referring to several authors that have been working in similar projects. Section 4.3 gives a detailed explanation of the two algorithms used for the vehicle detection. It also explains how the classification and counting of the vehicles is performed. Section 4.4 presents results of both algorithms on several test images. In section 4.5 conclusions about the results are given.

4.2 Summary of Literature Review

In the literature several authors have proposed different approaches for car detection, most of them using aerial images. For example in [14], [16] the authors propose an approach to detect and count cars using aerial images from urban scenes. They proposed a 3D car model for the detection and counting of vehicles. In [17] vertical view aerial images of urban areas are used, and the approach to detect the cars is based on a generic car model (shape of the boundary of the car, and boundary of front windshield), then the car model is used to predict if the shapes detected in the images are cars or not. In [18] a vehicle is modeled as a rectangle of a range of sizes, and convolution with edge masks are used to extract the four sides of the rectangular boundary. In [19] a model is created by example images of cars and their statistics are recorded in vectors; then by computing the features vector from image regions and testing them against the statistics of car models, the vehicles are detected. In [20] the authors extract and group image features to construct structures similar to a car model.

The goal of this work was to develop efficient algorithms for the detection, classification, and counting of vehicles on highways using high-resolution satellite images. Because our focus is on highway scenes, and not complex urban areas, our approach is much simpler, it is based on grayvalue intensities and thresholding, and it does not use any car models. The data used for this project are 1-m panchromatic images from the IKONOS satellite, and image processing techniques are used for the implementation of the algorithm. The algorithm is divided in two parts. The first part detects bright vehicles and utilizes two methods, Multiple Thresholds and Otsu Threshold. The second part detects the dark vehicles based on the Otsu method [21]. The MATLAB™ software was used for the implementation of the algorithms. The test images were obtained from the World Wide Web sites listened in references [11][12][13].

4.3 Technique used for the Detection of the Vehicles

The detection of vehicles in an image is performed by extracting the foreground objects. That is, the image is divided as foreground and background. Subdividing an image into its constituent regions or objects is known as image segmentation. Image segmentation is performed by using the common properties of the pixels. One of the common properties possessed by pixels is intensity. Therefore, intensity property of pixels is used in developing image segmentation algorithms.

Image segmentation algorithms generally are based on one of the two basic properties of intensity values: discontinuity and similarity. In the first criteria, the image is partitioned when there is an abrupt change in the intensities of neighboring pixels, such as edges. In the second criteria, the image is partitioned into regions which are similar according to some predefined conditions [22]. A simple way to do image segmentation is by using thresholding techniques, and this is the technique that is used for the detection of the vehicles in the present project.

4.4 Thresholding Technique

Thresholding is one of the widely methods used for image segmentation. It is useful in discriminating foreground from the background. By selecting an adequate threshold value T, the gray level image can be converted to binary image. The binary image should contain all of the essential information about the position and shape of the objects of interest (foreground). The advantage of obtaining first a binary image is that it reduces the complexity of the data and simplifies the process of recognition and classification. The most common way to convert a gray-level image to a binary image is to select a single threshold value (T). Then all the gray-level values below this T will be classified as black (0), and those above T will be white (1). The segmentation problem becomes one of selecting the proper value for the threshold T. A frequent method used to select T is by analyzing the histograms of the type of images that want to be segmented. The ideal case is when the histogram presents only two dominant modes and a clear valley (bimodal). In this case the value of T is selected as the valley point between the two modes. In real applications histograms are more complex, with many peaks and not clear valleys, and it is not always easy to select the value of T.

In the present work our sample images contain scenes of highways, and our goal is to apply segmentation techniques to detect, count, and classify (cars or trucks) the vehicles on the highways. Before applying the segmentation technique, the user has to specify a region of interest, from the original image. The region of interest is a subimage, of rectangular shape, which is selected using the mouse. Working with regions of interest instead of the whole image, helps to obtain better results in the application of our algorithm.

To extract the vehicles from the subimages, two algorithms based on thresholding were applied. The first algorithm uses Multiple Thresholds and it is described in section 4.6.1. The second algorithm is based on the Otsu method [21] and is explained in section 4.6.2.

4.5 Multiple Thresholds

The ideal case is a bimodal shape histogram, but such histograms are usually unavailable in real applications. In general, an intensity image have to be divided into several sub-ranges to perform thresholding, but these ranges usually overlap one with another and this makes thresholding difficult.

In this project the images that are used for vehicle detection present a histogram of the type shown in figure 4.1. From this figure we can see that two peaks are clearly defined. The peak on the left (dark side) represents the background. The peak on the right (bright side) are pixels that represent the foreground (vehicles). Notice that the histogram presents an overlapping region between the background and foreground pixels. This overlapping means that some of the objects or regions of background like lane markers have similar intensities as foreground objects (vehicles). Therefore to detect vehicles only, multiple thresholds are used. Section 4.6.1 describes the procedure followed to compute three different threshold values, and the way that they are used to detect the vehicles.

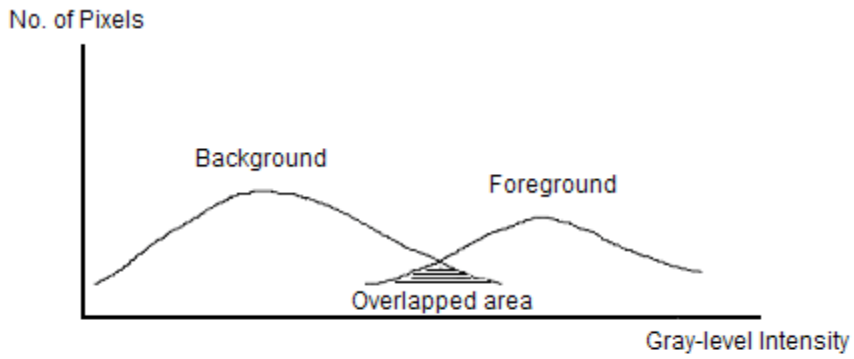


Figure 4.1 Bimodal Histogram with an Overlapped Area

To achieve our goal of detecting all the vehicles on the test images, we have divided the algorithm in two parts:

- Detection of vehicles which are bright with respect to the background.
- Detection of vehicles which are dark with respect to the background.

Figure 4.2 gives an example of dark and bright vehicles in a test image. In the following sections both parts are explained in detail.

Figure 4.2 Test Image Showing Bright and Dark Vehicles

4.6 Detection of the Bright Vehicles

Two different methods are proposed to identify the bright vehicles:

- **Multiple Thresholds**
- **Clustering by Otsu Method**

4.6.1 Computing Multiple Thresholds

By analyzing several of the sample images in the database, we can say that the intensity values of bright vehicles are greater than the intensities of the background, and sometimes there is a region where they overlap. Because the overlapping, some objects or regions on roads, such as lane markers and road dividers, may have intensity values similar to the intensity of some of the bright vehicles. Also, each bright vehicle may not have same range of intensity because of atmospheric conditions and different color of the vehicles. Therefore, to identify only the vehicles and to avoid the detection of irrelevant objects like lane markers, three different thresholds **T1, T2,** and **T3** are used.

Computation of the Threshold Values

In MATLAB, an intensity image can be stored in a two dimensional matrix. Let this matrix be

M1. Each element of the matrix **M1** corresponds to the intensity of each pixel of the image. A vector **M2** is formed by using **M1**. Each element of **M2** contains the maximum intensity of the corresponding row of intensities of **M1**. We know that in an intensity image, on a highway, bright vehicles have maximum intensity levels than any other objects. Therefore we are considering the maximum intensity in each row of **M2** to calculate the three thresholds **T1, T2,** and **T3**.

M2 is calculated by using M1

M2: for each row i in M1

M2[i] = maximum_intensity[M1, i]

T1, T2 and T3 are calculated by using M2

T1: T1 = Mean[M2]

T2: T2 = Minimum[M2]

T3: T3 = Mean[T1,T2]

Thresholds **T1**, **T2**, and **T3** are used to convert the test image to three different binary images **Image1**, **Image2**, and **Image3**. For a pixel at coordinates (x,y), if its intensity **I(x,y) > T**, then it is considered as an Object Pixel (1) else Background Pixel (0). A binary image is defined as:

$$\text{Image}(x,y) = 1 \quad \text{if } f(x,y) > T$$
$$= 0 \quad \text{if } f(x,y) < T$$

Figure 4.3 shows a highway in Oklahoma. To detect the vehicles using the proposed algorithm, the user selects first a region of interest. This region of interest has a rectangular shape, which is selected using the mouse, and should contain only one-way on the highway. If the highway under study is not vertical or horizontal on the image, the user should rotate the image first to make easier the selection of the region under study. Also, if the selected region of the highway contains two-ways, factors such as road dividers, sign boards, etc, will affect the result of the algorithm. Therefore to obtain better results, the region under study is selected in such a way that contains only one-way highway. Selection of road shoulders and ramps should be avoided, and the regions under study should contain only the highway with the vehicles. Figure 4.3b) shows an example of the selection of a region under study. This region is the one highlighted in Figure 4.3a). The selected one-way has ten bright vehicles and no dark vehicles.

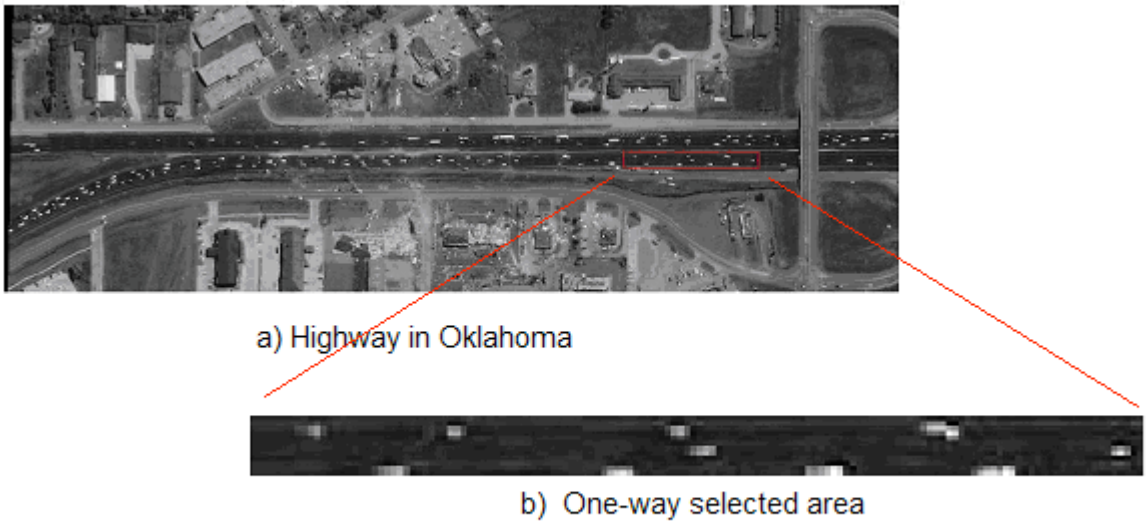


Figure 4.3 a) Highway in Oklahoma b) One-way Selected Area

By analyzing the histogram of Figure 4.3b) we observed that there are two significant ranges of intensities: 0 to 100 and 150 to 240. The region from 100 to 150 represents overlapping of intensity ranges of foreground and background. By applying the Multiple Thresholds algorithm on this test image, the values T1 = 199.67, T2 = 149.00, and T3 = 174.34 are obtained as thresholds. Three binary images **Image1, Image2**, and **Image3** are obtained by using these thresholds as shown in Figure 4.4.

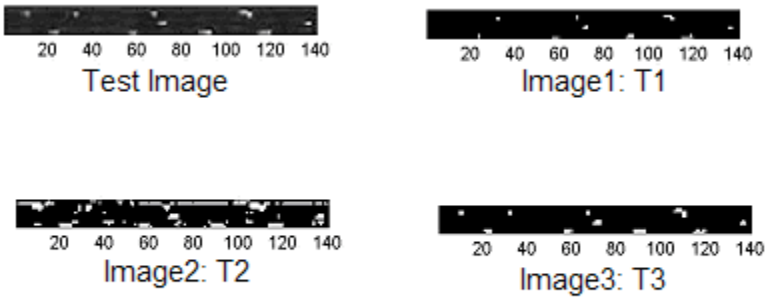


Figure 4.4 Binary Images Obtained by Using the Thresholds T1, T2, and T3

Reducing the detection of irrelevant objects such as lane markers and road dividers

From this example, fig. 4.4, we can see that each threshold detects some or all vehicles in the region under study, but lane markers and road dividers are detected by some of the threshold values too (due to the overlapping of the intensities of foreground and background). To avoid or reduce the detection of irrelevant objects such as lane markers and road dividers, logical operations are performed among the binary images. For the case of detecting bright vehicles, best results are obtained by taking common objects from pairs of the binary images generated by the thresholds **T1**, **T2**, and **T3**. This means, common objects from the pairs: (**Image1, Image2**), (**Image2, Image3**), and (**Image1, Image3**) are taken.

We can observe from Figure 4.4 that **Image1** contains some of the vehicles, **Image2** contains all vehicles and also lane markers, road dividers, etc., and **Image3** contains all the vehicles present in the test image. To detect all the bright vehicles, and to reduce the detection of irrelevant objects, the LOGICAL AND operation is performed to extract the common objects among the binary images. By applying the AND operation on the three combinations of binary images, three new binary images: **New_Image1**, **New_Image2**, and **New_Image3**, are formed which are shown in Figure 4.5.

New_Image1 = bitwiseAND[Image1, Image2]
New_Image2 = bitwiseAND[Image2, Image3]
New_Image3 = bitwiseAND[Image1, Image3]

a) Image1 AND Image2

b) Image1 AND Image3

c) Image2 AND Image3

Figure 4.5 Common Objects Taken From Pair of Binary Images
a) New_Image1 = Image1 AND Image2
b) New_Image2 = Image1 AND Image3
c) New_image3 = Image2 AND Image3

Detecting all the vehicles on the test image

Notice that the last three binary images: New_Image1, New_Image2, and New_Image3 contain only vehicles. Each of these binary images may contain some or all vehicles. These three images, New_Image1, New_Image2, and New_Image3 are added to obtain all the vehicles. The addition of these three images is performed by using the LOGICAL OR operation. From Figure 4.5 we can observe that New_Image1 contains some of the vehicles, New_Image2 also contains some of the vehicles and New_Image3 contains all the vehicles. Figure 4.6 shows the final result, this is, all the bright vehicles detected by the Multiple Thresholds Vehicle Detection Algorithm.

Final_Result = bitwiseOR(New_Image1, New_Image2, New_Image3)

Figure 4.6 Bright Vehicles Detected by the Multiple Threshold Method

4.6.2 Clustering by the Otsu Threshold

The Otsu threshold [21] uses class separability, and maximize the between-class variance to find an optimal threshold value **k*.** This threshold value is used to extract objects from their background. MATLAB has a built-in function that evaluates the Otsu threshold **k*.** Applying directly the Otsu threshold to the test image, will detect the bright vehicles, but also some of the lane markers and road dividers that are present on the highways. To reduce the problem of lane markers and road dividers, a preprocess step is applied first. The preprocess step involves the application of a sliding neighborhood operation to the test image. The sliding neighborhood operation consists of assigning to each pixel of the test image, the maximum intensity of its neighborhood (this is a rectangular area of 3-by-3 pixels, being the center pixel the one that is being processed by the operation).

Sliding Neighborhood Operation:

```
f = inline('max(x(:))');
slide_image = nlfilter(test_image,[3 3],f);
```

where nlfilter(testiamge,[3 3], f) applies the function 'f' to each 3-by-3 sliding block of test.

By applying this sliding neighborhood operation to the test images, the bright pixels corresponding to large objects, such as vehicles, become brighter, but the bright pixels corresponding to irrelevant objects such as lane markers stay about the same bright. This preprocessing step will help to highlight the vehicles, and dim some of the irrelevant objects such as the lane markers. After applying the sliding neighborhood operation, the Otsu Threshold is computed, and a binary image is generated. Figure 4.7 shows a test image, the resulting image after the sliding neighborhood operation is applied, and the binary image generated after the computation of the Otsu Threshold. Note that the Otsu Threshold detects all the bright vehicles.

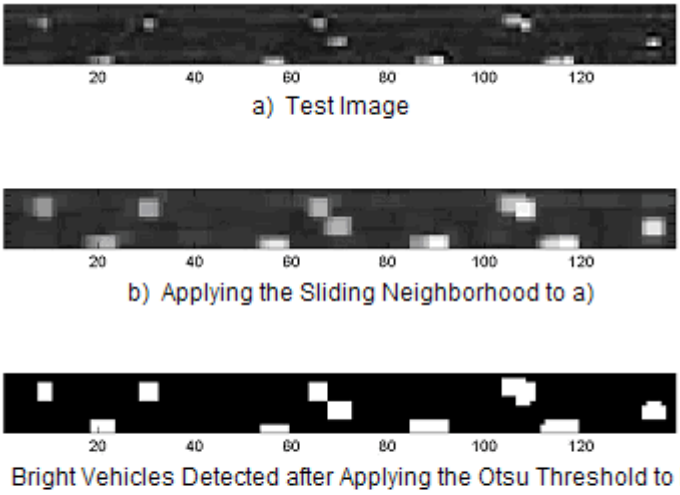


Figure 4.7 Detection of the Bright Vehicles Using the Otsu Threshold. a) Test Image (same as in fig 4.3) b) After applying the Sliding Neighborhood c) After applying the Otsu Threshold

4.7 Detection of Dark Vehicles

For the detection of dark vehicles, the Otsu Threshold is also employed. Before applying the Otsu Threshold, a sliding neighborhood operation is applied to the test image. Because in this case we want to detect dark vehicles, each pixel is assigned with the **minimum** intensity of its neighboring pixel in a rectangular neighborhood of a 3-by-3 matrix. As a result, dark vehicles become darker when compared to the background. After applying the sliding neighborhood operation, the Otsu Threshold is used to convert the test image to a binary image. Applying this procedure to Detect the Dark Vehicles to the test image of Fig. 4.3 we obtain the following results (Figure 4.8)

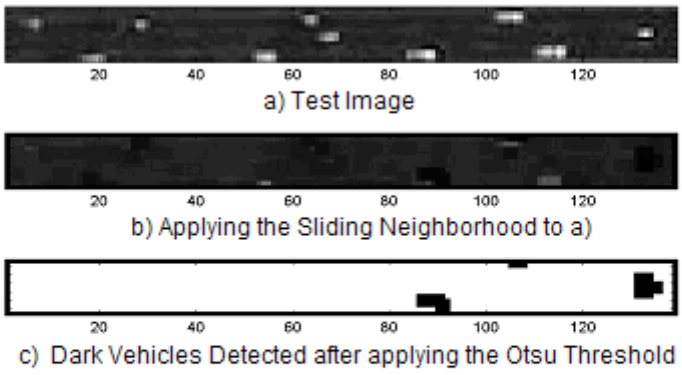


Figure 4.8 Detection of the Dark Vehicles Using the Otsu Threshold a) Test Image (same as in fig 4.3) b) After applying the Sliding Neighborhood c) After applying the Otsu Threshold

Note that the binary image, fig. 4.8c), displays all the dark vehicles as black patches. To display the dark vehicles as white patches, the complement of the binary image is taken. Figure 4.9 shows the complement of the binary image.

Figure 4.9 Dark Vehicles Detected by the Otsu Threshold
(Negative of fig 4.8 c))

Looking carefully at the original test image, fig. 4.3 , we can notice that the image does not contain dark vehicles. The white patches in the above binary image (fig. 4.9) represent the shadows of vehicles in the test image. To avoid considering vehicle's shadows as dark vehicles, the results of applying both algorithms (detection of bright vehicles, and the detection of dark vehicles) are added. This is carried out by performing a LOGICAL OR operation to the two binary images: the first one obtained by the bright vehicles detection algorithm (Otsu or Multiple Thresholds technique), and the second one obtained by the dark vehicles detection algorithm (Otsu Threshold technique). As a result of the addition, shadows of the bright vehicles, which are very close to the bright vehicles, are combined as a single vehicle. This process is presented in Figure 4.10, where the first image shows the binary image resulting after applying the bright vehicles detection algorithm (Otsu Threshold); the second image shows the binary image obtained after applying the dark vehicles detection algorithm; and the third image shows both images combined to give the final result.

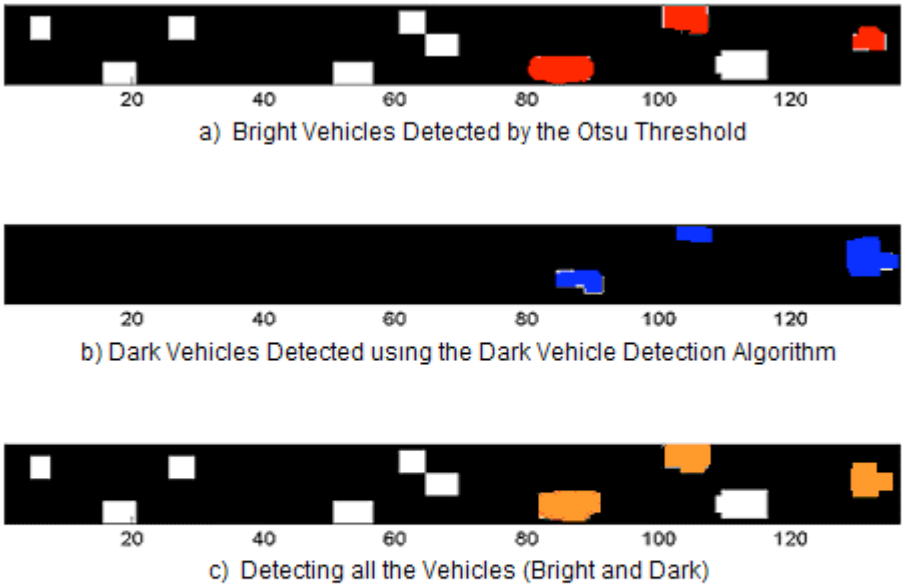


Figure 4.10 Applying the Vehicle Detection Algorithm to the Test Image of Fig. 4.3
a) Detecting the Bright Vehicles using the Otsu Threshold
b) Detecting the Dark Vehicles using the 'Dark Vehicle Detection Algorithm'
c) Detecting all the Vehicles in the Test Image

In fig. 4.10a), the vehicles colored in red have shadows, which are colored in blue in fig. 4.10b). In fig. 4.10c), the orange patches are the vehicles combined with their shadows. If the test image has dark and bright vehicles, then adding the two binary images (the one with the bright vehicles and the one with the dark vehicles), would not affect the final detection of the

vehicles as long as the detected vehicles in each binary image are not very close to each other.

In cases where dark and bright vehicles are placed very close to each other, it is possible that a bright vehicle could be combined with a dark vehicle and counted as a single vehicle, giving an error in the result. This is also true when two or more bright or dark vehicles are very close to each other, then the final result would combine this group of very close vehicles as a single vehicle, causing an error in the final count.

4.8 Classification of Vehicles as Cars and Trucks

To classify the vehicles, as cars or trucks, three parameters: width, height, and area, of the detected vehicles are considered. First the average of each of these parameters is computed. This is carried out by taking into account all the detected vehicles on the test image. Then, the three parameters for each of the detected vehicles are compared with the average values. Any vehicle with width, height, and area greater than the average values is considered as a truck, otherwise it is a car. The algorithm for the classification is as follows:

```
Mean_Size = mean [All Vehicles Size]
Mean_Width = mean [All Vehicles Width]
Mean_Height = mean [All Vehicles Height]

Car_Count = 0;
Truck_Count = 0;

For i = no_of_vehicles
If(Each_Car.Size > Mean_Size & Each_Car.Width > Mean_Width & Each_Car.Height >
Mean_Height)
        Truck_Count = Truck_Count + 1;
        Else
        Car_Count = Car_Count + 1;
        End
End
```

Special Case: At a given time, it is possible that a road may contain only cars (no trucks). In this case, there is a possibility that some cars may be classified as trucks. To minimize this problem, the values of the three average parameters (width, height, and area) are increased by 10%.

4.9 Counting the Vehicles

To compute the final count of vehicles in the test image (number of cars and number of trucks), the MATLAB function: BWLABEL is used. This function returns the number of connected objects in a binary image.

4.10 Special Cases that will affect the Results

To test our algorithm, we applied it to several of the images in the database. From the results we can say that the algorithm gave excellent results in the detection and classification of vehicles. However, as in all image processing algorithms, the performance strongly depends on the particular application, and there will be cases where the algorithm will not give the desired results. In order to obtain acceptable results in the application of our algorithm, the images under study should satisfy the following conditions.

- The region of study should include only one-way highway segments, and road shoulders and ramps should be avoided.
- The presence of bushes and trees decrease the accuracy of the results since those objects could be detected as vehicles.

For the cases where heavy traffic is present, the algorithm will cluster together the vehicles that are very close to each other, causing an error in the final counting of the vehicles.

4.11 Example. A Highway in Phoenix, AZ

In this section we show the results obtained by applying the proposed algorithm to one of the images in our Database. The bright vehicles are detected by applying both techniques: Multiple Thresholds and Otsu, and the results are compared. The dark vehicles are detected using the Otsu Threshold.

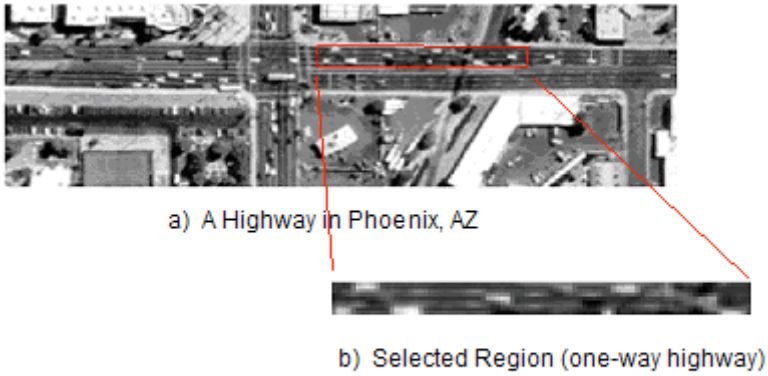


Figure 4.11 a) A Highway in Phoenix, AZ b) Selected region

4.11.1 Bright Vehicles detected by Multiple Thresholds

Figure 4.12 shows the three binary images obtained by applying the different thresholds T1, T2 and T3. These three threshold values were computed using the Multiple Thresholds method explained in section 4.3.2. From fig. 4.12 we can observe that T1 and T3 detected all bright vehicles except the vehicle highlighted in red. Notice that T2 detected all the bright vehicles, however lane markers (white patch with blue boundaries) were detected too. To detect only vehicles, and to avoid the detection of lane markers and other unwanted objects, only common objects are taken from each combination of these three binary images. Figure 4.13 shows the common objects of each combination.

Figure 4.12 a) Test Image (same as in fig. 4.11)
b), c), d) Binary Images Obtained Using the Thresholds: T1, T2, and T3

Notice that during the process of eliminating the lane markers and road dividers, there is a possibility of losing some of the vehicles. In Figure 4.12c) we can observe that while trying to eliminate the lane marker (white patch with blue boundary) in IMAGE2, we lost a bright vehicle (marked in red).

Figure 4.13 Common Objects taken from Pair of Binary Images (from fig. 4.12)
a) New_Image1 = Image1 AND Image2
b) New_Image2 = Image1 AND Image3
c) New_Image3 = Image2 AND Image3

In Figure 4.13, the three binary images have all the bright vehicles that are present in the test image, except one, the vehicle highlighted with a red box in figure 4.12a). Notice that no lane markers or road dividers are present in any of the three images. To obtain all the bright vehicles, the three binary images (New_Image1, New_Image2, New_Image3) are added. Figure 4.14 shows the bright vehicles detected by the Multiple Thresholds method.

Figure 4.14 Bright Vehicles Detected by the Multiple Threshold Method

Notice that from the 8 bright vehicles present in the test image, 7 were detected using the Multiple Thresholds technique. From this example we can see that the Multiple Thresholds technique can detect only those vehicles that are much brighter than the background. Vehicles which are just a little brighter that the background, are not detected by the Multiple Thresholds technique.

4.11.2 Bright Vehicles Detected by Otsu Threshold

Figure 4.15 shows the binary images obtained after applying the Otsu Threshold to the test image. Figure 4.15a) shows the result of applying the Otsu Threshold without the preprocessing step (Sliding Neighborhood). Figure 4.15b) shows the result obtained when a Sliding Neighborhood operation is applied first.

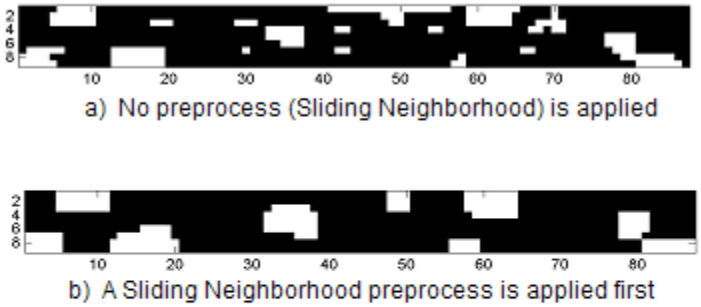


Figure 4.15 Binary Images obtained by applying the Otsu Threshold a) No preprocess step is applied b) A preprocess step is applied first

Notice from Figure 4.15a) that if the Sliding Neighborhood operation is not applied, the resulting binary image (after applying Otsu Threshold) will contain all the bright vehicles, but also lane markers. In Figure 4.15b), the preprocess step (Sliding Neighborhood operation) was applied before the Otsu Threshold. Notice that all the bright vehicles that are present in the test image were detected. To detect all the bright vehicles, common objects from these two images, fig. 4.15a) and b), are taken. Figure 4.16 shows all the bright vehicles detected by the Otsu Threshold Technique.

Figure 4.16 Bright Vehicles Detected by the Otsu Threshold

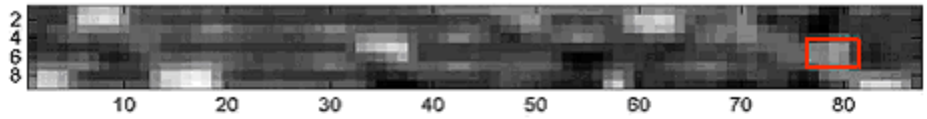
4.11.3 Dark Vehicles Detected by the Otsu Threshold

Figure 4.17 shows the results obtained after applying the Otsu Threshold to detect the dark vehicles, section 4.7. From this figure we can observe that all the dark vehicles were detected. As it was explained in section 4.7, a preprocessing step is applied first, this is, a Sliding Neighborhood operation that applies a minimum intensity function to a 3 x 3 neighbor.

Figure 4.17 Dark Vehicles Detected by the Otsu Threshold Technique

4.11.4 Detecting All the Vehicles

To obtain the final image showing all the vehicles, the binary image containing the bright vehicles, and the binary image containing the dark vehicles are added. Figure 4.18 presents the resulting image showing all the vehicles (bright and dark). Fig. 4.18a) display the test image, same as in fig. 4.11. Figure 4.18b) shows the result obtained by applying the Multiple Threshold technique to detect the bright vehicles, and the Otsu Threshold to detect the dark vehicles. Figure 4.18c) presents the result obtained by applying the Otsu Threshold to detect both types of vehicles, bright and dark. Comparing Figures 4.18b) and 4.18c) we can observe that the Multiple Threshold technique was able to detect all the vehicles, but one (the one highlighted by the red box). The reason of the missed vehicle is because that particular vehicle has pixel values very similar to the background intensities. From fig. 4.18c) it can be seen that the Otsu Threshold technique has detected all the vehicles, but also a lane marker, the white patch highlighted by the blue box. Note also that the Otsu threshold is clustering two different vehicles (red rectangle in fig. 4.18c), and this will give an error in the final count.



a) Test Image

b) Bright vehicles detected by Multiple Threshold,
Dark vehicles by the Otsu Threshold

c) Bright adn Dark vehicles detected by the Otsu Threshold

Figure 4.18 Detecting all the vehicles. a) Test Image b) Bright vehicles detected by the Multiple Threshold and Dark vehicles by the Otsu Threshold c) Bright and Dark vehicles detected by the Otsu Threshold

4.12 Results

Two main algorithms were developed for the detection and classification of vehicles. The first algorithm uses Multiple Thresholds for the detection of the bright vehicles, and the Otsu method [21] for the detection of the dark vehicles. The second algorithm uses the Otsu method for the detection of both, bright and dark vehicles. Both algorithms were tested on several of the images in the database. To measure the performance of the algorithms, the results obtained from each algorithm were compared with a manual count of the vehicles; this is, by visually inspecting the region under study.

From the results, we can say that the algorithms gave very good results in the detection and classification of vehicles. However, as in all image processing algorithms, the performance strongly depends on the particular application, and there will be cases where the algorithm will not render the desired results.

From the performed tests we found out that in order to obtain acceptable results (an error of less than 10% in the detection and counting of the vehicles) in the application of the algorithms, the regions under study should satisfy certain conditions. For example, the region under study should include only one-way highway segments, and road shoulders and ramps should be avoided. The presence of bushes and trees decrease the accuracy of the results since those objects could be detected as vehicles. For the cases where heavy traffic is present, the algorithms would cluster vehicles that are very close to each other, causing an error in the final count of the vehicles.

As future work to try to improve the results, some statistical approaches, such as the Bayesian classifier [23], could be used to evaluate the threshold values and see if more robust results can be obtained. Also, multispectral imagery could be combined with the panchromatic information, and apply techniques base on color properties to detect the vehicles and try to avoid irrelevant objects. To make the detection and classification fully automatic, without requiring from the user to rotate the image first, and then to select a region of study (only a highway segment), techniques based on pattern recognition should be investigated.

CHAPTER 5 Conclusions and Suggested Research

In this project two main sets of programs were developed. In the first set, two algorithms were implemented to detect highways and roads from the 1-m panchromatic IKONOS satellite images. The first algorithm detects roadways only , "Detecting highways without vehicles", sections 3.2-3.5. The second algorithm, "Detecting highways with vehicles" is explained in sections 3.6 and 3.7. The MATLAB code for these algorithms is given in Appendix A. Both algorithms were tested with several images in our Database (Chapter 2). From the results we can say that most of the highways were detected, but in some cases, small parts of barren land and parking lots were also classified as roadways. The detection of some unwanted objects is because our technique is base on the pixel intensity values, and in some cases the barren lands and parking lots have the same pixel values as the roadways. To improve the results of the algorithm, other techniques should be added. For example Digital Elevation Models (DEM) could help to avoid using pixels that are part of buildings or other above-ground features. Also feature extraction procedures with artificial intelligence methods could be included.

The second set of programs includes two algorithms that detect, count, and classify vehicles on the roadways. 1-m panchromatic IKONOS satellite images containing highways scenes were used. The first algorithm uses Multiple Thresholds Techniques to detect the vehicles (section 4.6.1). The second algorithm is base on the Otsu Threshold (sections 4.6.2 and 4.7). The algorithms were tested using the images from our Database (Chapter 2). From the results we can say that the algorithms provided satisfactory results, but the performance highly depends on several factors such as the presence of vegetation and buildings, the type of pavement material, and the traffic conditions in the particular scene under study. From the analysis of the tests, we found out that in order to obtain results with a maximum error of 10% in the detection and counting of the vehicles, the regions under study should satisfy the following conditions: 1) The region under study should include only one-way highway segments, and road shoulders and ramps should be avoided. 2) The presence of bushes and trees decrease the accuracy of the results since those objects could be detected as vehicles. 3) For scenes with heavy traffic, the algorithms would cluster vehicles that are very close to each other, causing an error in the final count of the vehicles. To obtain a more robust algorithm, other techniques should be added to the current algorithms. For example multispectral imagery could be combined with the panchromatic information, and techniques base on color properties could be applied to detect the vehicles and try to avoid other objects. Some statistical approaches such as the Bayesian classifier could be used to evaluate the threshold values and see if more accurate results can be obtained. To make the detection and classification of the vehicles fully automatic, without requiring from the user to select a region of study (only a highway segment), techniques based on pattern recognition should be investigated.

In relation to ice and snow detection, we found that most of the studies in the literature are related to monitor snow cover over large areas. The main application of these studies is to help in weather forecasting models identification of snow cover. In particular for transportation purposes, the detection of ice and snow has to be in a much more smaller scale. For example we would like to be able to detect snow or ice buildup on the highways, and be able to inform the travelers about the hazardous condition of the roads, and dispatch snowplows to those areas. From our meticulous literature review in this topic, we found that there are still several technological limitations for using remote sensing as a means for acquiring information to detect and assess the presence of ice and snow on the highways. The main limitations are in the temporal and spatial resolutions. For example, the current 1-m spatial resolution of IKONOS could be useful to detect a snow patch size of 2-by-2 m, base on a roadway width of 4-m. To detect smaller ice patches, a higher spatial resolution, in the order off 0.25-m, would be necessary. With respect to the temporal resolution, a 5-60 minutes could provide a reasonable performance for the detection of snow or ice buildup on the roadways. However, the current temporal resolution is of about 3-6 days. With respect to the spectral resolution, the technology is available. For example passive microwave (PM) radiometers, with its near all-weather capabilities has been used widely in snow cover detection. Also multi-spectral and hyper-spectral remote sensing systems could be used effectively to detect the crystalline structure and thermal absorbing characteristics of the snow and ice.

REFERENCES

[1] URL: http://www.ncrst.org/reseach/ncrst_home.html , National Consortia for Remote Sensing in Transportation, last retrieved: Dec. 2003

[2] Jensen, J.R, Cowen, D.J., "Remote sensing of urban/suburban infrastructure and socio-economic attributes", Photogrammetric Engineering and Remote Sensing, Vol. 65, No. 5, May 1999, pp. 611-622

[3] Coulter, L., et al, "Deriving Current Land-use information for metropolitan transportation planning through integration of remotely sensed data and GIS", Photogrammetric Engineering and Remote Sensing, Vol. 65, No. 11, November 1999, pp. 1293-1300

[4] McFarland, B., " Image resolution requirements for metropolitan transportation applications", Proceedings of Pectora 14 –Land Satellite information in the Next Decade III Conference, Denver , CO, December 1999

[5] Hung, M., "Urban Land Cover Analysis From Satellite Images", Proceedings of Pecora 15-Land Satellite Information IV/ISPRS Commision I/FIEOS 2002 Conference, Denver Colorado, Nov.10-15, 2002.

[6] Kim, Y., Seo, B., Oh, J., "The effect of the resolution of satellite images on the interpretability and detectability of geographic information", Proceedings of Pecora 14-Land Satellite Information in the Next Decade, III Conference, Denver Colorado, Dec. 6-9, 2000.

[7] InfoTerra, Satellite Imagery Worldwide, IKONOS, http://www.infoterra-global.com/ikonos.htm, accessed May 2003

[8] Kelly, R., and Atkinson, P. , " Modeling and Efficient Mapping of snow Cover in the UK for Remote Sensing Validation", Advances in remote sensing and GIS analysis, Sep. 1999, pp.75-95

[9] Shi, J. Hensley, S., Dozier, J., "Mapping Snow Cover With Repeated Pass Synthetic Aperture Radar", IEEE 1997 International Symposium on Geoscience and Remote Sensing, Singapore, Aug.3-8, 1997, pp. 628-630

[10] Standley, A., Barrett, E., "The use of coincident DMSP SSM/I and OLS satellite data to improve snow cover detection and discrimination", International Journal of Remote Sensing, Vol. 20, No. 2, 1999, pp. 285-305.

[11] Space Imaging Inc., Gallery [Online] http://www.spaceimaging.com/gallery/, accessed May 2003

[12] INTEC Americas Corp, Gallery [Online] http://www.intecamericas.com/english.htm, accessed May 2003

[13] ORBIMAGE [Online] http://www.orbimage.com/prods/index.htm, accessed May 2003

[14] Schlosser, C., Reitberger, J., and Hinz, S., "Automatic Car Detection in High Resolution Urban Scenes Based on an Adaptive 3D-Model",Proceedings of the IEEE/ISPRS joint Workshop on Remote Sensing and Data Fusion over Urban Areas, Berlin, 2003, IEEE, Piscataway, pp. 167 - 171.

[15] Angel, A., Hickman, M., et al, "Methods of Analyzing Traffic Imagery Collected from Aerial Platforms", IEEE Transactions on Intelligent Transportation Systems, Vol. 4, No.2, June 2003, pp. 99-107

[16] Hinz, S., "Detection and Counting of Cars in Aerial Images", Proceedings of the International Conference on Image Processing, Vol. 3, Barcelona, Sep. 2003, pp. III-997-1000

[17] Zhao, T., Nevatia, R., "Car Detection in Low Resolution Aerial Image", Proceeding of the 8th IEEE International Conference on Computer Vision, Vol.1, Vancouver, Canada, July 2001, pp. 710-717

[18] Moon, H., Chellappa, R., Rosenfeld, A., " Performance Analysis or a Simple Vehicle Detection Algorithm", Proceedings of the 3th ARL Federal Laboratory Symposium, Feb. 1999

[19] Schneiderman, H., and Kanade, T., "A statistical Method for 3D Object Detection Applied to Faces and Cars",

Proceedings of the Computer Vision and Pattern Recognition, Vol.1, Hilton Head, SC, June 2000, pp. 746-751

[20] Liu, G., Gong, L., and Haralick, R., "Vehicle Detection in Aerial imagery and Performance Evaluation" ILS Technical Report, Intelligent Systems Laboratory, University of Washington, Seattle, WA, 1999

[21] Otsu, Nobuyuki, "A threshold selection method from gray-level histograms", *IEEE transactions on systems, man, and cybernetics*, vol. SMC9, no.1, January 1979, pp.62-66

[22] Gonzalez, R., and Woods, R.., " Digital Image Processing", 2nd Ed., Prentice Hall, 2002

[23] L. Shapiro L., and G. Stockman, G.," Compute Vision", Prentice Hall, 2001, p. 114

[24] Kim, Y., Lee, J., Lee, B., "Congestion Data Acquisition Using High Resolution Satellite Imagery and Frequency Analysis Techniques", Proceedings of IGARSS'97, Vol. 1, Aug. 3-8, 1997, pp. 331-334

[25] Jaffe, Valerie, "Bozeman Banks on 1-meter Imagery", Imaging Notes, Vol. 15, No. 4, July/August 2000. On line at: http://www.imagingnotes.com, last retrieved: Dec. 2003

Dissemination

R. Alba-Flores, "Detection of Small Objects from High-Resolution Satellite Images Using Matlab Software", Presentation at the Viz Lab Presents Series, Visualization Digital Imaging Laboratory, UMD, Duluth MN, Nov. 4, 2003.

S. Kuthadi, R. Alba-Flores, "Detection of Small Objects from High-Resolution Satellite Images", Poster Presentation at the Research Day, Northland Advanced Transportation Systems Research Laboratories, Duluth MN, Nov. 13, 2003.

R. Alba-Flores, S. Kuthadi, "Detecting Vehicles and Roadways with Satellite Images", Presentation at the Research Day, Northland Advanced Transportation Systems Research Laboratories, Duluth MN, Nov. 13, 2003

R. Alba-Flores, "Using High-Resolution Satellite Images for Detecting Roadways and Vehicles", *Northland Transporter Newsletter,* Northland Advanced Transportation Systems Research Laboratories, Duluth MN, Issue 2, Fall 2003

R. Alba-Flores, "Vehicle Detection with Satellite Images", Presentation at the Graduate Colloquium, UMD Department of Mathematics and Statistics, Duluth, MN, Feb. 27, 2003.

R. Alba-Flores, S. Kuthadi, P. Jain, "Detecting and Counting Cars from Satellite Images", Presentation at the Research Day, Northland Advanced Transportation Systems Research Laboratories, Duluth MN, Nov. 14, 2002

Appendix A. Algorithm to Detect the Highways

Menu interface is provided to the users. Options in the menu are the following:

1. **Location:** The path of the image file must be entered here.

2. **Angle:** If the part of the image that we are going to consider is in slant position, then the image must be rotated so that we can select required part.
   a. To rotate image in clockwise direction, enter negative angle.
   b. To rotate image in anti clockwise direction, enter positive angle.
   c. **NOTE:** If the rotation of image is not required, then enter zero degrees. Don't avoid this option.

3. **Coordinates:** This option has two sub-options:

- **Select Points:** In this option we select the region of interest. When this option is selected, the image will appear on window. Select the upper left corner and lower right corner of the rectangular part that we are interested in.
- **Same Points:** This option is used to use the coordinates of region of interest of last case. When this option is selected, cropped image will appear on the window.
  - If the image is rotated by some angle in last case, image in current case will also be rotated by the same angle and then cropping is done.
- **NOTE:**
- If same points are not used and new points are to be selected, then **CLEAR** all the variables before selecting new points.
- While selecting the points, care should be taken that regions other than road should not be selected i.e, avoid road dividers etc.

4. **Region of Interest:** This option is used to crop the region of interest. Using this option we can crop only rectangular regions (not circular regions).When this option is selected, immediately the cropped part of the image will appear on the window.

5. **Points:** This option displays the points that we have selected in crop option.

6. **Road Detection:** This option has three sub options:
   - **Hor_Road:** To detect horizontal road in the image.
   - **Ver_Road:** To detect vertical road in the image.
   - **All Roads:** To detect all the roads in the image.

   Note:
   1. Hor_Road and Ver_Road options are used to find only horizontal and vertical roads in the image.
   2. After selecting Hor_Road (Ver_Road) option, if Horizontal (Vertical) road is selected along some other region, then select the same option Hor_Road (Ver_Road). Repeat this until only road is detected.

7. **Clear all Variables:** This option will clear all the variables under use i.e. all the variables are made empty.

8. **Continue:** It asks the user whether he wants to continue the program or not. It gives two choices to the users. They are:
   1. Yes: If this option is selected, then location for another image is asked.
   2. No: Quits the program.
   3. Cancel: Cancels the option selected.

9. **Exit:** It exits the Matlab.

Source Code For The Menu:

```
clc
copy_x = zeros(2,1);
copy_y = zeros(2,1);
f = uimenu('Label','MENU');
    uimenu(f,'Label','Location','Callback','loc');
    uimenu(f,'Label','Angle','Callback','ang');

    h = uimenu(f,'Label','Coordinates','Callback','coordinates');
  uimenu(h,'Label','SelectCoordinates','Callback','selectcoordinates');
    uimenu(h,'Label','Same Coordinates','Callback','samecoordinates');
    uimenu(f,'Label','Region of Interest','Callback','crop');
    uimenu(f, 'Label','Points Selected','Callback','points');

    h = uimenu(f,'Label','Road Detection','Callback','ROAD_DET');
    uimenu(h, 'Label','Hor_Road','Callback','hor_road');
    uimenu(h, 'Label','Ver_Road','Callback','ver_road');
    uimenu(h, 'Label','All_Roads','Callback','allroads');

    uimenu(f,'Label','Clear all variables','Callback','clearing');
    uimenu(f,'Label','Continue','Callback','continue');
    uimenu(f,'Label','Quit','Callback','exit');
```

Source Code For All The Options Listed In The Menu:

Location:

loc.m:

```
prompt = {'Enter the location of the image: '}; %Displays the text in the
quotes in dialog box.
title = 'LOCATION';  %gives title to the dialog box
location = inputdlg(prompt, title); %Dialog box appears on the screen which
asks user to enter the location of the image.
                                    %Location of image file will be stored in
variable 'location'. Type of variable
                                    %'location' is 'cell string'

location1 = char(location);   % type conversion from cellstring to string
image = imread(location1);    %reads image into array 'image'.
rimage = image;
figure(1);
subplot(1,1,1),imshow(image);   %displays image
```

Angle Of Rotation:

ang.m:

```
prompt = {'Enter the angle of rotation: '}; %Displays the text in the quotes
in dialog box.
title = 'ANGLE OF ROTATION';  %gives title to the dialog box
Angle = inputdlg(prompt, title);     %Dialog box appears on the screen which
asks user to enter the angle of rotation.
```

```
                                    %Angle of rotation will be stored in
variable 'Angle'. Type of variable
                                    %'Angle' is 'cell string'
Angle2 = char(Angle);      % type conversion from cellstring to string
Angle1 = str2num(Angle2);  % type conversion from string to numeric
rimage = imrotate(image, Angle1); %rotates image by angle Angle1
image = rimage;
imshow(rimage);   %dispalys rotated image
```

Select Coordinates:

selectcoordinates.m:

```
imshow(rimage);
PIXVAL ON    %diplays a bar on the screen which shows the coordinates and
pixel value
            % of the pixel pointed by the mouse.
[x y] = ginput(2);   %x and y gives coordinates of the points selected by the
mouse
[r c] = size(x);      %dimensions of x are stored in r and c

if r < 2 %we have to select two coordinates. This condition is used to avoid
any error entries.
   errordlg('Insufficient Data','Error Box');
end
```

Same Coordiantes:

samecoordinates.m:

```
button = questdlg('Do u want to use the coordinates used in last case?');
%displays a question dialog box which dispalys
                                        %the text in quotes. It also has
three options: 'yes', 'no' and 'cancel'. The
                                        %option selected will be stored in
variable 'button'.


if strcmp(button , 'No') %if selected option is 'No', user is allowed to
select new coordinates
   selectcoordiantes
end
if strcmp(button, 'Yes') %if selected option is 'Yes', user can use the
previously used points.
   rimage = imrotate(rimage, Angle1);
   imshow(rimage);
   [n_rows n_col] = size(xc);
   for i = 1:2:n_col
      x(1,1) = xc(i);
      x(2,1) = xc(i+1);
      y(1,1) = yc(i);
      y(2,1) = yc(i+1);
      crop
    end

end
```

Region Of Interest:

crop.m:

```
 imshow(rimage);
 PIXVAL ON    %diplays a bar on the screen which shows the coordinates and
pixel value
             %of the pixel pointed by the mouse.
 %[x y] = ginput(2);   %x and y gives coordinates of the points selected by
the mouse
 %[r c] = size(x);    %dimensions of x are stored in r and c
 %if r < 2    %we have to select two coordinates. This condition is used to
avoid any error entries.
 %   errordlg('Insufficient Data','Error Box');
 %end
 width = x(2,1) - x(1,1); %to crop the image, height and width are required.
 height = y(2,1) - y(1,1);
 cropimage = imcrop(rimage,[x(1) y(1) width height]); %crops the region of
interest
 figure(1), imshow(cropimage);
```

Points Selected:

points.m:

```
z = cat(2, x, y); %concatenates arrays x and y.
z1 = num2str(z);  % type conversion from numeric to string
msgbox(z1, 'x      y'); %displays the points selected for region of interest.
```

All Roads:

allroads.m:

```
cond = 1; %checking variable for RoadThreshold.m file
RoadThreshold  %this file is used to convert intensity image to binary
               %using three thresholds.
figure(1), imshow(tempimage5);

tempimage5  = bwmorph(tempimage5, 'clean'); %remove isolated pixels (1's
            %surrounded by 0's).
tempimage5 = bwmorph(tempimage5, 'majority');% Set a pixel to 1 if five %or
             more pixels in its 3-by-3 neighborhood are 1's.
tempimage5  = bwmorph(tempimage5, 'fill'); %Fill isolated interior %pixels
             (0's surrounded by 1's).
tempimage5  = bwfill(tempimage5, 'holes'); %Fills the holes in the %binary
            image.


figure(2),imshow(cropimage);
figure(3),imshow(tempimage5);

[r c p]= size(tempimage5);
%checks each pixel in the final binary image. If it is white, then it is made
black, else it is made equal to the corresponding intensity in the original
image.
for i = 1:r,
   for j = 1:c,
      if tempimage5(i,j) ==  0 %black pixel represents road.
         newimage(i,j)  = cropimage(i,j); %pixel intensity is made equal to
                                          corresponding intensity in original
                                          image.
      else
```

```
         newimage(i,j) = 0;
      end
   end
end
figure(4), clf;
imshow(newimage);
clear newimage;
```

Horizontal Road:

hor_road.m:

```
hor = 1; %checking variable for horizontal roads.
ver = 0; %checking variable for vertical roads.
cond = 0; %checking variable for all roads.
RoadThreshold
clear y;
k = 1;

[ROWS COLS PAGES] = size(tempimage);
inc = 0.02;
cond_hor = cond_hor - inc; %each time this file is selected, con_hor reduces
                              by "inc".
for i = 1:ROWS
   FLAG = 1; %for each row, FLAG is set to 1. Flag=1 represents that %the row
              should me made white.
   COUNT = 0; %for each row, COUNT is set to 0.
   for j = 1: COLS
      if tempimage(i, j) == 0;
         COUNT = COUNT + 1;
         if COUNT > (COLS*cond_hor) %if number of black pixels are %greater
            FLAG = 0;      %row is made black.
         end
      end
   end

  if FLAG == 1 %row should me made white
     y(k) = i; %storing the coordinates of white rows in array y.
     k = k+1;
     for j = 1: COLS
       tempimage(i, j) = 255;
     end
  else  %row is made black
     for j = 1: COLS
       tempimage(i, j) = 0;
     end
  end
end

 [r1 c1] = size(y);
height = y(c1) - y(1) ;
width = COLS;


tempimage2  =  imcrop(cropimage1,  [1  y(1)  width  height]);  %cropping  the
                                                          horizontal road.

figure(1);
PIXVAL ON
```

```
subplot(2,2,1), imshow(cropimage);
subplot(2,2,2), imshow(final_image);
subplot(2,2,3), imshow(tempimage); %binary horizontal road
subplot(2,2,4), imshow(tempimage2);%final horizontal road

cropimage1 = tempimage2;
```

Vertical Road:

Ver_road.m:

```
ver = 1; %checking variable for vertical roads.
hor = 0; %checking variable for horizontal roads.
cond = 0; %checking variable for all roads.
RoadThreshold
clear x;
k1 = 1;
[ROWS1 COLS1 PAGES1] = size(tempimage1);
inc = 0.02;
cond_ver = cond_ver - inc; %reduces cond_ver by inc each time Ver_Road %
                           option is selected.

%vertical roads
for i = 1:COLS1
   FLAG = 1; %for each column, FLAG is set to 1.
   COUNT1 = 0; %for each column, COUNT1 is set to 0.

%In each row, counting the number of black pixels. If it is greater than
checking condition, them that column is made black.
   for j = 1: ROWS1
      if tempimage1(j, i) == 0;
         COUNT1 = COUNT1 + 1;
         if COUNT1 > (ROWS1*cond_ver)
            FLAG = 0; %0 represents that column should be made black.
         end
      end
   end

  if FLAG == 1 %1 represents that column should me made white.
     x(k1) = i; %coordinates of white columns are stored in array x.
     k1 = k1+1;
     for j = 1: ROWS1
       tempimage1(j, i) = 255;
     end
  else
     for j = 1: ROWS1
       tempimage1(j, i) = 0;
     end
  end
end

[R1 C1] = size(x);
height1 = ROWS1;
width1 = x(C1) - x(1);

tempimage3 = imcrop(cropimage2, [x(1) 1 width1 height1]);%vertical road is
cropped from original image.
figure(1);
subplot(2,2,1), imshow(cropimage);
subplot(2,2,2), imshow(final_image);
```

```
subplot(2,2,3), imshow(tempimage1);%binary vertical road
subplot(2,2,4), imshow(tempimage3);%final vertical road

cropimage2 = tempimage3;
```

Road Threshold:

RoadThreshold.m:

```
if hor == 1 %for detecting horizontal roads
  testimage = cropimage1;
  testimage1 = cropimage1;
  testimage2 = cropimage1;
end
if ver == 1 %for detecting vertical roads
  testimage = cropimage2;
  testimage1 = cropimage2;
  testimage2 = cropimage2;
end
if cond == 1 %for detecting all roads
  testimage = cropimage;
  testimage1 = cropimage;
  testimage2 = cropimage;
end
  [r c p] = size(testimage);

 MEAN = mean(testimage, 2); %1*#rows matrix with average intensities in each
row.
 average = mean(MEAN) %average of the elements in the above matrix
[row column page] = size(testimage); %gives the size of the matrix which
contains the region of interest
 for k=1:page,
  for i=1:row,
    for j=1:column,
       if testimage(i,j) < average
          testimage(i,j) = 0;
      end
      if testimage(i,j) >= average
         testimage(i,j) = 255;
      end
    end
  end
end

minimum = min(MEAN); %minimum value in the above matrix
temp_array(1) = average;
temp_array(2) = minimum;
avg_min = mean(temp_array)
for i=1:row,
    for j=1:column,
       if testimage1(i,j) < avg_min
          testimage1(i,j) = 0;
      end
      if testimage1(i,j) >= avg_min
         testimage1(i,j) = 255;
      end
    end
 end

 MEAN1 = mean(testimage);
```

```
 average1 = mean(MEAN1) %average of above two values.

 for i=1:row,
    for j=1:column,
       if testimage2(i,j) < average1
          testimage2(i,j) = 0;
      end
      if testimage2(i,j) >= average1
         testimage2(i,j) = 255;
      end
    end
 end

%taking common objects from the above resultant images
add01 = bitand(testimage,testimage1);
add02 = bitand(testimage,testimage2);
add12 = bitand(testimage1,testimage2);

%combining the add01, add02 and add12 images.
final_image = bitor(add01,add02);
final_image = bitor(final_image,add12);

imshow(final_image);

tempimage = final_image;%hor
tempimage1 = final_image;%ver
tempimage5 = final_image;%all
```

Clearing:

clearing.m:

```
clear; %clears all variables
clear figure(1); %clears the figure window
clc; %clears the command window
```

Continue:

continue.m:

```
button = questdlg('Do u want to continue'); %displays a question dialog box
which dispalys the text in quotes. It also has
                                            %three options: 'yes', 'no' and
'cancel'. The option selected will be stored in
                                            % variable 'button'.

if strcmp(button , 'No') %compares selected option with 'No'
   quit;      %if selected option is 'No', we get quitted from the command
window
end
if strcmp(button, 'Yes') %compares selected option with 'Yes'
                         %if selected option is 'Yes', program runs from the
beginning
   clc;                  %clears command window
   clf;
   menu                  %calls menu file
   loc                   %calls loc file
end
```

Appendix B. Algorithm to Detect, Count, and Classify Vehicles

Menu interface is provided to the users. Options in the menu are the following:

1. **Location:** The path of the image file must be entered here.

2. **Angle:** If the part of the image that we are going to consider is in slant position, then the image must be rotated so that we can select required part.

To rotate image in clockwise direction, enter negative angle.
To rotate image in anti clockwise direction, enter positive angle.
NOTE: If the rotation of image is not required, then enter zero degrees. Do not avoid this option.

3. **Coordinates:** This option has two sub-options:
    1. **Select Points:** In this option we select the region of interest. When this option is selected, the image will appear on window. Select the upper left corner and lower right corner of the rectangular part that we are interested in.
    2. **Same Points:** This option is used to select the coordinates of region of interest of the last case. When this option is selected, cropped image will appear on the window. If the image was rotated by some angle in the last case, then the image in current case will also be rotated by the same angle and then the cropping process is carried out.

NOTE: If same points are not used and new points are to be selected, then **CLEAR** all the variables before selecting new points.
While selecting the points, care should be taken that regions other than road should not be selected i.e, avoid road dividers etc.

4. **Region of Interest:** This option is used to crop the region of interest. Using this option we can crop only rectangular regions (not circular regions).
When this option is selected, immediately the cropped part of the image will appear on the window.

5. **Points:** This option displays the points that we have selected in crop option.

6. **Intensity:** This option has two sub options. They are:

1. **Manual Intensity:** When this option is selected, a dialog box appears on the screen which asks for the intensity threshold. Manual intensity threshold should be entered by analyzing the cropped image.

2. **Intensity Thresholds:**

7. **Count:** This option gives the count of the vehicles in the cropped image in a dialog box.

8. **Clear all Variables:** This option will clear all the variables under use i.e. all the variables are made empty.

9. **Continue:** It asks the user whether he wants to continue the program or not. It gives two choices to the users. They are:
    1. Yes: If this option is selected, then location for another image is asked.
    3. No: Quits the program.
    3. Cancel: Cancels the option selected.

11. **Exit:** It exits the Matlab.

BETTER RESULTS:
Better results are obtained from pictures of good quality and high resolution. If the objects are differentiated properly from the ro
then perfect results can be obtained.

Source Code For The Menu:

menu.m:

```
f = uimenu('Label','MENU');
```

```
%uimenu(f,'Label','New Figure','Callback','figure');
%uimenu(f,'Label','Save','Callback','save');
uimenu(f,'Label','Location','Callback','loc');
uimenu(f,'Label','Angle','Callback','ang');
uimenu(f,'Label','Region of Interest','Callback','crop');
uimenu(f, 'Label','Points Selected','Callback','points');
g = uimenu(f,'Label','Intensity Threshold','Callback','intensity');
uimenu(g,'Label','Manual Intensity','Callback','manual');
uimenu(g,'Label','Average Intensity','Callback','avg');
uimenu(f,'Label','Count','Callback','roi');
uimenu(f,'Label','Clear all variables','Callback','clearing');
%uimenu(f,'Label','Calendar','Callback','calendar');
uimenu(f,'Label','Print','Callback','printdlg');
uimenu(f,'Label','Continue','Callback','continue');
uimenu(f,'Label','Quit','Callback','exit');
```

In the function uimenu, the third parameter gives the name of the option that appears to users on menu and the last parameter g the name of the source code file for that option. The parameter Callback calls the file given in the last parameter which performs required actions.

Code of all the options that are given in Menu:

loc.m:

```
prompt = {'Enter the location of the image: '}; %Displays the text in the quotes in
dialog box.
title = 'LOCATION';  %gives title to the dialog box
location = inputdlg(prompt, title); %Dialog box appears on the screen which asks
user to enter the location of the image.
                                    %Location of image file will be stored in
variable 'location'. Type of variable
                                    %'location' is 'cell string'

location1 = char(location);   % type conversion from cellstring to string
image = imread(location1);    %reads image into array 'image'.
rimage = image;
figure(1);
subplot(1,1,1),imshow(image);   %displays image
```

ang.m:

```
prompt = {'Enter the angle of rotation: '}; %Displays the text in the quotes in
dialog box.
title = 'ANGLE OF ROTATION';  %gives title to the dialog box
Angle = inputdlg(prompt, title);    %Dialog box appears on the screen which asks
user to enter the angle of rotation.
                                    %Angle of rotation will be stored in variable
'Angle'. Type of variable
                                    %'Angle' is 'cell string'
Angle2 = char(Angle);      % type conversion from cellstring to string
Angle1 = str2num(Angle2);  % type conversion from string to numeric
rimage = imrotate(image, Angle1); %rotates image by angle Angle1
image = rimage;
imshow(rimage);   %dispalys rotated image
```

selectcoordinates.m:

```
imshow(rimage);
PIXVAL ON   %diplays a bar on the screen which shows the coordinates and pixel
value
```

```
            % of the pixel pointed by the mouse.
[x y] = ginput(2);   %x and y gives coordinates of the points selected by the mouse
[r c] = size(x);      %dimensions of x are stored in r and c

if r < 2 %we have to select two coordinates. This condition is used to avoid any
error entries.
   errordlg('Insufficient Data','Error Box');
end
```

samecoordinates.m:

```
button = questdlg('Do u want to use the coordinates used in last case?'); %displays
a question dialog box which dispalys
                                          %the text in quotes. It also has three
options: 'yes', 'no' and 'cancel'. The
                                          %option selected will be stored in
variable 'button'.


if strcmp(button , 'No') %if selected option is 'No', user is allowed to select new
coordinates
   selectcoordiantes
end
if strcmp(button, 'Yes') %if selected option is 'Yes', user can use the previously
used points.
   rimage = imrotate(rimage, Angle1);
   imshow(rimage);
   [n_rows n_col] = size(xc);
   for i = 1:2:n_col
      x(1,1) = xc(i);
      x(2,1) = xc(i+1);
      y(1,1) = yc(i);
      y(2,1) = yc(i+1);
      crop
    end

end
```

crop.m:

```
 imshow(rimage);
 PIXVAL ON   %diplays a bar on the screen which shows the coordinates and pixel
value
             %of the pixel pointed by the mouse.
 %[x y] = ginput(2);  %x and y gives coordinates of the points selected by the
mouse
 %[r c] = size(x);   %dimensions of x are stored in r and c
 %if r < 2   %we have to select two coordinates. This condition is used to avoid
any error entries.
 %   errordlg('Insufficient Data','Error Box');
 %end
 width = x(2,1) - x(1,1); %to crop the image, height and width are required.
 height = y(2,1) - y(1,1);
 cropimage = imcrop(rimage,[x(1) y(1) width height]); %crops the region of interest
 figure(1), imshow(cropimage);
```

points.m:

```
z = cat(2, x, y); %concatenates arrays x and y.
z1 = num2str(z);  % type conversion from numeric to string
```

```
msgbox(z1, 'x      y'); %displays the points selected for region of interest.
```

manual.m:

```
PIXVAL ON    %diplays a bar on the screen which shows the coordinates and pixel
value
             %of the pixel pointed by the mouse.
prompt = {'Enter the intensity threshold:  '};
title = 'Intensity Threshold';
intensity_threshold1 = inputdlg(prompt, title);
intensity_threshold2 = char(intensity_threshold1);
intensity_threshold3 = str2num(intensity_threshold2);
 tempimage = cropimage;  %storing cropped image into a temporaray variable
[row column page] = size(tempimage); %gives the size of the matrix which contains
the region of interest

%converting cropped intensity image to binary (B & W) image.
 for k=1:page,  % for each pixel
  for i=1:row,
    for j=1:column,
       if tempimage(i,j) <= intensity_threshold3 %if pixel intensity is less than
given threshold, it is made black.
          tempimage(i,j) = 0;
      end
      if tempimage(i,j) >= intensity_threshold3 %if pixel intensity is greater than
given threshold, it is made bright.
         tempimage(i,j) = 255;
      end
    end
  end
 end
 %displays both intensity and binary versions of the cropped image.
 subplot(1,2,1),subimage(cropimage);
 subplot(1,2,2),subimage(tempimage);
```

3 intensity thresholds:

```
add01 = bitand(tempimage, tempimage1); %takes common objects of tempimage and
tempimage1
add02 = bitand(tempimage, tempimage2); %takes common objects of tempimage and
tempimage2
add12 = bitand(tempimage1, tempimage2); %takes common objects of tempimage1 and
tempimage2

se = ones(2,2); %structuring element

final = bitor(add01, add02); %takes union of add01 and add02
final = bitor(final, add12); %takes union of final and add12

%final = dilate(final, se); %dilates the final image
% dialte has some advantages and also some disadvantages:
% Advantage: If a vehicle gets splitted into parts after applying the thresholds,
dilation can
                    combine these parts into a single vehicle.
%Disadvantage: If two vehicles are closer, then they can get combined into a single
vehicle.

figure(2);
subplot(2,2,1),imshow(add01);
subplot(2,2,2), imshow(add02);
```

```
subplot(2,2,3), imshow(add12);
subplot(2,2,4), imshow(final);

[label num] = bwlabel(final,4);
cardata = imfeature(label,'Area','MajorAxisLength','MinorAxisLength');
allcars = [cardata.Area];
majoraxis = [cardata.MajorAxisLength];
minoraxis = [cardata.MinorAxisLength];
meansizecar = mean(allcars);
meanlength = mean(majoraxis);
meanminor = mean(minoraxis);
q = num2str(meanlength);
r = num2str(meanminor);
carcount = 0;
truckcount = 0;
len = length(allcars);
for i = 1:len
         if(allcars(i) > meansizecar & majoraxis(i) > meanlength & minoraxis(i) >
meanminor)
      truckcount = truckcount + 1;
   else
      carcount = carcount + 1;
   end
end
num1 = num2str(carcount);
num2 = num2str(truckcount);
msgbox(num1, 'Car Count');
msgbox(num2, 'Truck Count');
```

clearing.m:

```
clear; %clears all variables
clear figure(1); %clears the figure window
clc; %clears the command window
```

continue.m:

```
button = questdlg('Do u want to continue'); %displays a question dialog box which
dispalys the text in quotes. It also has
                                            %three options: 'yes', 'no' and
'cancel'. The option selected will be stored in
                                            % variable 'button'.

if strcmp(button , 'No') %compares selected option with 'No'
   quit;    %if selected option is 'No', we get quitted from the command window
end
if strcmp(button, 'Yes') %compares selected option with 'Yes'
                         %if selected option is 'Yes', program runs from the
beginning
   clc;                  %clears command window
   clf;
   menu                  %calls menu file
   loc                   %calls loc file
end
```