Road Scene Image Analysis in Lane Departure Warning Systems

Christopher A. Warner
Department of Electrical and Systems Engineering
Oakland University, Rochester, Michigan USA.

Abstract

This paper will cover many aspects of image analysis including image pre-processing, image segmentation, and contextual analysis for road scene images acquired using a digital camera mounted to the rear-view mirror of different automobiles. In particular, road marker characteristics will be extracted including those for various white and yellow longitudinal lane markers. This lane marker information will serve to provide a position with respect to the lane for the vehicle as it travels down the road. If the vehicle should transition towards a position outside the lane boundary, the image processing system will have the potential to notify the vehicle operator that a lane departure is pending. This vehicular image processing system should be relatively invariant to illumination conditions and should be able to perform during a wide range of environmental conditions. This paper will also discuss many of the aspects required to implement the software in Matlab including flowcharts and partial code listings. Preliminary output results for the preliminary road scene image processing system will also be given.

TABLE OF CONTENTS

LIST OF FIGURES

LIST OF TABLES

1. INTRODUCTION

1.1 Justification for road scene image processing systems

In the United States in the year 2000, there were approximately 6.4 million motor vehicle crashes reported to police [11]. In many of these crashes, the position of the vehicle with respect to its travel lane was an important characteristic of the situation. Although longitudinal lane pavement markings are intended to support the regulated movement of traffic within the roadway (thus reducing the likelihood of crashes), there are also many other important factors which must be considered in any traffic situation including (but not limited to) the number and type(s) of vehicle(s) involved, the vehicle(s) dynamic operating conditions, the absence or presence of an 'intended' lane transition, the type of transition (drifting, passing, intentional lane change, etc.), the physical setting, and environmental conditions. Two resources which may be referenced regarding analysis of vehicular crashes including those involving lane changes are given in [11] and [12]. It should be noted that lane change crashes sharing the characteristics of multiple vehicle, same direction, parallel path scenarios are not the only occurrences where longitudinal lane pavement markings may play a significant role in crash circumstances. Consider also crashes involving roadway departure or vehicles traveling in opposite directions (just to name two of the many others). Nonetheless, longitudinal lane markers (when present) serve to provide information including (but not limited to) boundary and guidance to be used by the motor vehicle operator on that roadway. The 2004 edition of *A Policy on Geometric Design of Highways and Streets* indicates that road-following and safe-path maintenance in response to road and traffic conditions are significant to performing vehicle guidance [26]. Thus, the ability of an individual vehicle operator to extract and interpret lane marker information is of paramount importance to the vehicle operator's efficient and safe usage of the roadway. However, situations (driver distraction, driver physical impairment, etc.) may arise which might potentially prevent the operator of a vehicle from using all of the available traffic control device resources (including lane markers). Under these circumstances, having a road scene image processing system in place as part of an overall vehicular safety system being used to minimize the potential consequences of 'missed lane markings' could potentially assist in reducing the frequency of crash occurrences, reducing crash severity, and thus helping to save lives. Additionally, a road scene image processing system might potentially provide travel lane guidance and roadway delineation where the absence or obscurement of a formal longitudinal lane marker has created some travel lane uncertainty. Note that *great care* must be taken when attempting to quantify and qualify important factors including (but not limted to) the position of a vehicle within its lane as well as when a vehicle has deviated from its lane, the intended or unintended maneuvers of a vehicle operator within the roadway, and the effectiveness in which a road scene image processing system both interprets road scene information and alerts vehicle operators to potentially dangerous situations.

There are many important factors which need to be considered governing the effectiveness of traffic control devices (including longitudinal lane pavement markers). The 2003 MUTCD lists five basic requirements as [10]:

1. Fulfill a need;
2. Command attention;
3. Convey a clear, simple meaning;
4. Command respect from the road users;
5. Give adequate time for proper response.

One of the primary needs is to provide lane delineation assisting guidance of the vehicle traveling on the roadway. Thus, a longitudinal lane marker should be conspicuous, visible, understandable, and provide sufficient response time to safely maneuver the vehicle under a wide range of conditions including (but not limited to) driver state, vehicle speed (and other dynamic conditions), vehicle type, and environmental conditions (including weather and illumination). Two key aspects of any longitudinal lane pavement marker that must be considered are the markings luminance and the contrast of the marking with the substrate (road surface). A formal definition of luminance is the luminous flux emitted from a surface per unit solid angle per unit area, projected onto a plane normal to the direction of propagation [5]. However, luminance (in general terms) may be thought of as the amount of light reflected to the vehicle operator or road scene image processing system from the longitudinal lane marker. Contrast may be generally looked at as the difference between marking luminance and local substrate luminance and the greater the contrast between the two, the greater the potential for discernibility. Thus, the materials used for both the road surface and the lane marker may play a very significant role in affecting lane

delineation. Similarly, any materials used in roadway repair and maintenance may affect discernibility of longitudinal lane markers.

Lane markers may be composed of materials including paint, thermoplastic, and preformed tape (just to name a few). Each may include glass spheres or beads to increase retroreflective properties while also relying on the pigment to enhance the markers reflective characteristics. There are many different types of paints, thermoplastics, and preformed tapes and several sources which may be referenced for further information include [31], [32], and those listed in the related websites portion of the appendix under pavement preservation. Two compositions which may be used in roadway construction are bituminous concrete and Portland cement concrete. Bituminous concrete is a road surface composed from materials including (but not limited to) hot asphalt, hot mineral aggregate and refined tar while Portland cement concrete will typically be composed of materials including (but not limited to) Portland cement, sand, water and coarse aggregate. If simply considering the luminance characteristics of the two aforementioned road surfaces, it is apparent that a 'brighter' marker located on a 'darker' surface (or vice versa) would be more discernible than a 'brighter' marker located on a 'lighter' surface under comparable conditions. However, there are many other factors which may affect lane delineation including those listed in table X.

Table X. Some potential factors affecting lane delineation

	Some potential factors affecting lane delineation	Potential examples
1.	Illumination source	Natural generated from sunlight or skylight, artificially generated light from street lights or automotive headlights, ambient, etc.
2.	Illumination characteristics	Brightness, spectrum, illumination area, etc.
3.	Characteristics of the light wave or particle	Polarization, wavelength, frequency, phase, energy, etc.
4.	Direction of illumination	Front directed, back, etc.
5.	Angle of illumination and angle of observation	CIE 45/0 (0/45) geometry with CIE 2° standard observer, etc.
6.	Distance between vantage point and marker	Length, time, etc.
7.	Light interaction between marker and substrate	Specular reflection, diffuse reflection, retroreflection, transmission, absorption, scattering, etc.
8.	Spectral selectivity	Polarizers, IR filters, spectral transmittance vs. wavelength, etc.
9.	Environmental conditions	Temperature, surface moisture, humidity, fog, atmospheric transmittance, precipitation, etc.
10.	Human factors	Age, cognitive capability, physical status, human optics system, motor skills, color perception, tendency for vehicle position within lane, etc.
11.	Vision processing system factors	Optical characteristics, electrical/mechanical/sensor characteristics, algorithms, architectures, environment, vehicle characteristics, etc.
12.	Vehicular characteristics	Vehicle front end length, vehicle width, windshield glass transmittance, height of operator or vision system above road surface, the location and type of headlights, etc.
13.	Materials and properties of road substrate, repair or maintenance materials being used	Cement, sand, water, aggregate, refined tar, rubberized asphalt, asphaltic rubber, polymer-modified liquid asphalt, conductivity, reflectivity, luminance, etc.
14.	Materials and properties of longitudinal lane marker	Paint, thermoplastic, preformed tape, pigment, glass beads, reflectivity, refraction, luminance, etc.
15.	Marker pattern, dimensions, gap-to-segment ratios	Broken, normal, double, width, speed, and context dependent characteristics, etc.
16.	Road geometry	Vertical curves, horizontal curves, junction type, etc.
17.	Surface finish	Smooth, rough, grooved, etc.
18.	Road dirt and debris	Dust, dirt, automotive fluids, paper, etc.
19.	Illumination attenuation	Shading from roadside objects, other vehicles, etc.
20.	Discernibility	Legibility, interpretability, etc.
21.	Age, condition, performance,	Intended service life, durability, appearance and visibility, etc.
22.	Application process for marker	Paint striper, melting and forming, inlaid or overlaid, etc.

Each of these factors (amongst others) must be carefully considered when analyzing how a road scene image processing system may alert a vehicle operator to potentially adverse lane positioning governed via:

1. Detecting a lane marker or change thereof;
2. Understanding the *context of* the detection;
3. Recognizing the *significance in* the detection;
4. Evaluating the cost of response versus no response;
5. Alerting the vehicle operator appropriately and (potentially) sending necessary information to other dependent vehicle safety systems.

1.2 Overview of the Preliminary Road Scene Image Processing System

The following figure shows a generalized block diagram of the monocular camera road scene image processing system.

Figure X: Block diagram of a general image processing system

The image acquisition block should use an image sensor which yields a maximal signal to noise ratio to exposure ratio. Factors including (but not limited to) linearity, sensitivity, and noise should all be considered. Readers interested in further information regarding CCD and CMOS imaging sensors may refer to [15]. The pre-processing block will adjust the digital image information into a format that makes future processing steps simpler. This may include cropping, filtering, or other measures to improve image quality or facilitate processing. However, filters have different magnitude and phase responses and thus great care must be taken when applying them. Hence, an image filter (if chosen) should increase the signal to noise ratio. This process should not adversely affect the significant edges of the original image prior to undergoing edge (transition) detection, first-level discrimination, edge connection, GEM discrimination, and determination of the forward and rearward projections.

Images are represented in many different formats and many standards are in place that specify the representation of the image information. The images of the forward scenes analyzed in this paper are JPEG images acquired using a Canon PowerShot A85 digital camera with some of the camera specifics given in table X on page XX of the appendix. The JPEG images were read into Matlab variables using the 'imread' function which creates a multi-dimensional matrix variable containing Red-Green-Blue (RGB) values. Red, green, and blue are the colors of the visible spectrum which may be combined to form most other colors (from the possible chrominance range). However, since the processing of the three matrices of information (one each for the contributions from the red, green, and blue channels in the RGB color space) may require more resources than are necessary for a given task and since color imager chips are typically more expensive than those producing grayscale images, the analysis of grayscale images may preferentially be chosen. A great deal of meaningful information may be extracted from a grayscale image and current image processing systems in the automotive industry are designed around the processing of 'grayscale' information. To convert the RGB matrix produced by Matlab to a grayscale matrix, a transformation from the RGB color space to a combined luminance and chrominance (YIQ) color space was implemented in Matlab. The luminance information was then used as the grayscale information. In fact, the NTSC television standard has previously used luminance information in its black and white broadcasts. A transformation was also performed from the RGB color space to the Hue-Saturation-Value (HSV) color space to compare and contrast luminance information.

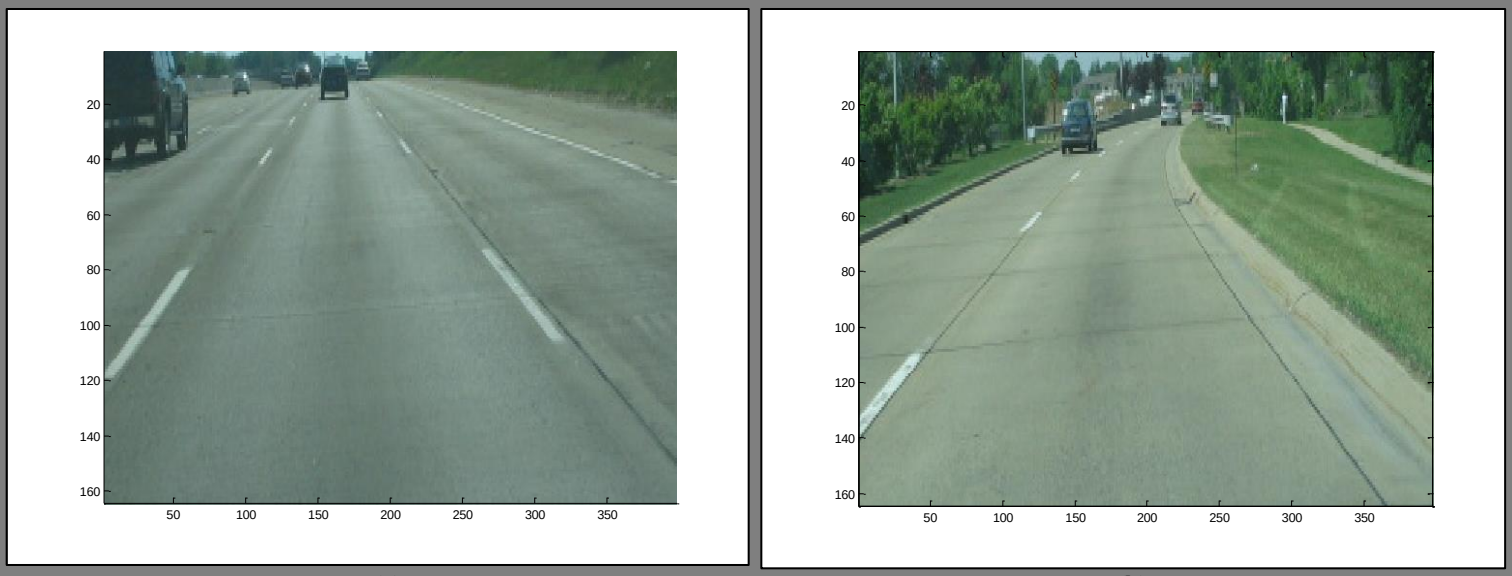
Edge based segmentation (including the connection of components) for lane markers was one of the goals of this analysis. There are a number of different edge detectors that may be used from first order gradient operators to methods based on Canny (recursive) and others. However, when considering edge detector performance, there are several criteria that should be strived for. The first is a low rate of error, meaning that true

edges are not missed and false edges are not detected. The second is that the edges should fall as close to the center of the actual edge as possible. The third is that there be only a single edge detected per real image edge [1]. Each distinct edge detector has its own unique characteristics and produces its own unique results. In many of the images that will be processed, the edges that are detected by the various operators may have discontinuous points due to a number of factors including (but not limited to) imperfections with the image processing hardware and software along with other possibilities from table X. However, methods to overcome many of these situations may be applied within the Matlab image processing to segment the image and extract the necessary features.

Once the images have been segmented and the necessary features of the lane markers have been classified, other information may be extracted from the road scene image. Because of factors including (but not limited to) the perspective of the image being taken, the camera's field of view and the specific region on interest, the range of road geometries which may be encountered while driving, the position of a vehicle with respect its lane of travel, the dimensions of a vehicle and those of the particular lane of travel, and the reason for and manner in which a driver may transition out of a lane, positive sloped marker edge transitions will typically not be found on the right one-third side of the image and negative sloped marker edge transitions will typically not be found on the left one-third side of the image. There will also be a greater likelihood that more positive sloped marker edge transitions will be found on the left half side of the image than on the right half side and a greater likelihood that more negative sloped marker edge transitions will be found on the right half side of the image than on the left half side. Many other considerations will need to be made when analyzing the various combinations of the previously mentioned factors affecting the processing of the potential marker transitions. From the transitions passing the various levels of discrimination, point-slope equations may be formulated, one each for the positive and negative sloped markers (if each is sufficiently present). These two linear equations may then be coupled by realizing that they will have an intersection point as long as they do not represent parallel line segments. This will provide a basis for approximating the vehicle's position with respect to the lane boundaries as well as providing some insight into the direction in which the road may be heading. The method that will be used to discriminate road marker edges from non-road marker edges will be a combination of structural (syntactic) pattern classification and statistical pattern classification. This method will combine structural primitives with statistical methods to assist in classifying likely road scene marker edges via their 'unique' characteristics and relationships.

The Matlab programming environment was chosen as the initial environment for the preliminary road scene image processing for a number of reasons. The first had to due with its ease of use and wide range of functions available for accessing and manipulating the large matrices associated with image processing. Another reason was the wide range of functions provided for manipulating the equations inherent in linear algebra. Another was its capability to display the generated images and results in a straightforward fashion. Although Matlab has its own image processing toolbox, this paper will show that a great deal of image processing may be performed without the tools of that toolbox and in fact, some of the tools can be 'reproduced' including those performing transformations from one color space to another and edge detection. Matlab also has the capability of controlling the processing through a graphical user interface (GUI) which provides a graphical environment for exercising the software.

The road scene images were taken using a camera mounted to the rear view mirror of different vehicles while driving on a wide variety of thoroughfares. These Michigan roadways included interstate highways, Michigan highways, and local roads (see vehicle and thoroughfare lists in table X on page XX of appendix). They were acquired at various times during the day under a variety of illumination conditions. A couple of typical road scene images are shown in figure X.



(a) (b)

Figure X: (a) Road scene with both broken lane markers; (b) road scene where curb side normal lane marker not present

The general flow of the preliminary road scene image processing software written in Matlab is shown in figure X.

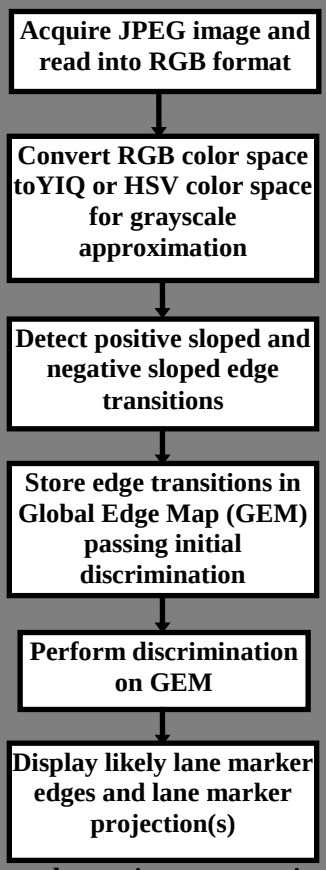


Figure X: General flow to preliminary road scene image processing software written in Matlab

Once an image has been converted from RGB to Hue-Saturation-Value (HSV) or Luminance-Hue-Saturation (YIQ) coordinates, the positive and negative sloped dark to light and light to dark edge transitions are detected. These edge transitions undergo multiple levels of discrimination on their way to being stored in the (un)discriminated global edge map. Significant points are then extracted, slope approximations are generated, and the positive and negative sloped projections (if present) are then displayed.

1.3 Current and future directions

Image processing of forward road scene information is becoming more and more possible with advances in technology. One area of technological advance is with imaging systems based on Charge-Coupled-Device (CCD) and Complementary Metal Oxide Semiconductor (CMOS) chip technologies. Another is in the area of CMOS technologies for custom applications in silicon. A third is with high dynamic range camera (HDRC) systems. Another is the vastly increased processor clock rates along with increased data throughput rates due to improved architectures, memory and bus technologies, and enhanced instruction sets. Yet another is with the advancement in operating systems, application software, and user tools available to those delving into the area of image processing. Forward road scene image processing may be particularly appealing to those in the military and automobile industries. Military uses might include leader-follower unmanned convoys and the like. Commercial automotive applications potentially include lane departure warning systems for vehicles equipped with adaptive cruise control systems or advanced safety systems.

Coinciding with the demand for these technologies is the need to have industry and federal standards in place to regulate the functionality and performance of these systems. Only in the last several years has there been a concerted effort to mandate standards for this area of technology as a result of the increased likelihood that these systems will find widespread commercial appeal leading to large scale production volumes. Several standards have been initiated including ISO 17361:2007 (with others currently being considered). However, it is not only the creation of a complete and thorough set of standards, but industries willingness to comply with the standards set forth which dictates how successful these programs turn out to be. It is also very important that there needs to be a unanimous agreement between all those governing bodies involved as to precisely how delegation of traffic control device design and use is legislated.

Currently, there are commercial vehicle systems which provide a camera and dash mounted display allowing a vehicle operator to see what is behind the vehicle. There are also commercial object detection systems that facilitate parking and radar systems used for driver drowsiness and adaptive cruise control. However, imaging systems that extract meaningful road scene information are only now being exploited due to the advances in hardware and software. These systems will find more commercial applications in various industries within the next five years or so, limited only by the imaginations of the scientists and engineers working on them and the technologies available for use in these systems. They will continue to extract more and more pertinent road scene information from the visual stream potentially leading to systems which deduce road geometry and contribute to ‘situation awareness’. Before inferring ‘road scene understanding’, some comprehension of the position of a vehicle with respect to the lane boundaries designated by the lane markers (if present) must be had. Thus, the methods used for classification of the road marker patterns will play a very important role in how efficiently the preliminary road scene image processing system performs.

2. HYBRID STRUCTURAL (SYNTACTIC) AND STATISTICAL PATTERN CLASSIFICATION

2.1 Introduction and preliminary goals

The idea behind applying a hybrid structural (syntactic) and statistical pattern classification method to lane markers results from a preliminary analysis of the problem domain and the characteristics of the other classification methods available. Some of the other methods available for pattern classification include methods based on statistical patterns, neural networks, fuzzy set theory and hybrid versions combining certain aspects of or features from the set of those available.

Many detailed books have been written on the topics of structural (syntactic) pattern recognition including those referenced in [7], [8]. This paper considers extracting the subpatterns or subcharacteristics (primitives) from a lane marker that assist in distinguishing it from other patterns which may be found in a road scene image. The primitives will then be evaluated through software written in the Matlab programming environment to evaluate their quantitative interrelationships (attributes of primitives) to either discriminate or accept the characteristic. Thus, a hybrid pattern classification method combining structural (syntactic) and statistical pattern recognition will be used.

If the goal was to classify only the broken longitudinal lane markers that are used to provide lane delineation, then it might be more feasible to pursue a purely statistical pattern recognition approach. Statistical pattern recognition has been successfully applied in areas such as optical character recognition [16] in which the problem domain is highly constrained and well-defined. However, the goals are much loftier than classifying only white broken lane markers. Some of the preliminary goals of the preliminary road scene analysis include:

1. Classification of a border transition resulting from white broken lane markers or the edges (or a significant edge component) of those markers.
2. Classification of a border transition resulting from white normal lane markers or the edges (or a significant edge component) of those markers.
3. Classification of a border transition resulting from white dotted lane markers or the edges (or a significant edge component) of those markers.
4. Classification of a border transition resulting from yellow broken lane markers or the edges (or a significant edge component) of those markers.
5. Classification of a border transition resulting from yellow normal lane markers or the edges (or a significant edge component) of those markers.
6. Classification of a border transition resulting from white wide lane markers or the edges (or a significant edge component) of those markers.
7. Classification of a border transition resulting from the innermost yellow (white) lane marker in a normal double yellow (white) lane marker combination or resulting from the innermost normal solid yellow and normal broken yellow marker combination or the edges (or a significant edge component) of those marker combinations.

It should be noted that lane delineations resulting from the sole use of raised pavement markers (without associated longitudinal markings) is not included in the preliminary road scene analysis.

Consider the two images of figure X.

(a)

(b)

Figure X: Road scene images (a), (b) showing combinations of different types of lane markers

As can be seen from figure X, a single scene in a sequence may contain a great deal of interrelated road information (painted markers, audible rumble strip, physical barriers, etc.). It might be possible to extract the white broken lane markers from figure X (a) using some of the markers geometric (area, extremal points, elongation, etc.) or shape (spatial moments) characteristics, but what about applying that same methodology to the normal yellow lane marker on the left hand side of the image in figure X (a). If attempting to apply a series of templates through some squared error method (root-mean-square error, sum-of-square error, least-mean-square error, etc.) to a gradient approximation in the locale of the broken marker, then a second methodology might be required to classify the normal yellow lane marker on the left hand side. It might be possible to apply a combined structural and statistical approach to the yellow lane marker portion, but with each additional analysis methodology incorporated comes a potential increase in system complexity. Thus, the hybrid method was chosen because it might effectively be applied to achieve a large percentage of the stated goals in a reasonably efficient manner.

2.2 Potential method overview and sample longitudinal lane marker characteristics

Figure X shows a generalized block diagram for potentially performing hybrid structural (syntactic) and statistical pattern classification.

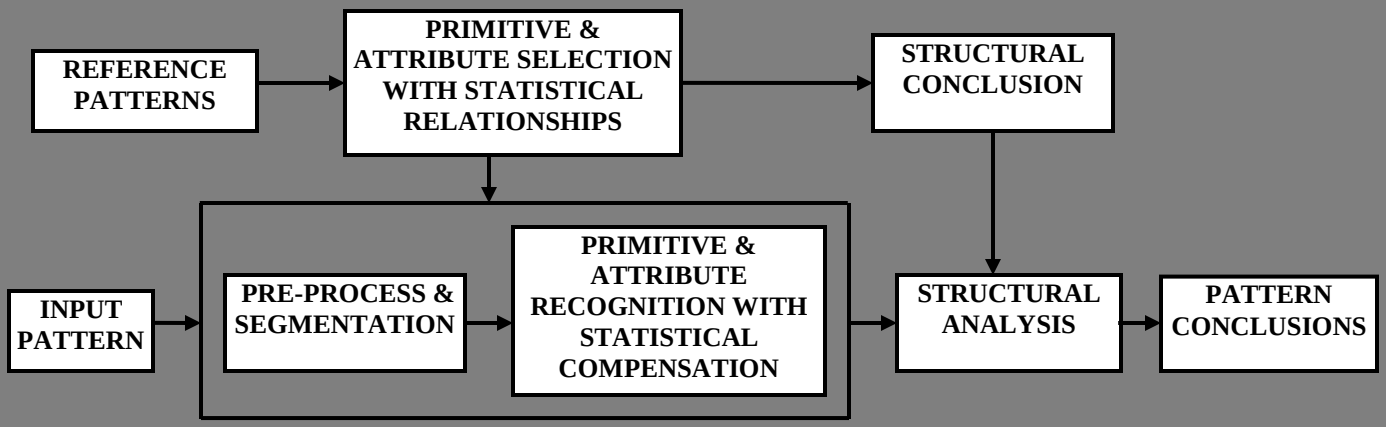


Figure X: Potential generalized block diagram for performing hybrid structural (syntactic) and statistical pattern classification

A series of reference patterns are evaluated for primitives, attributes and statistical relationships leading to a set of initial conclusions regarding the structure of the pattern. The actual input pattern is first preprocessed (if necessary) to improve the image quality or facilitate subsequent processing before the segmentation and recognition processes are performed. The input pattern may be broken down with sought after pattern primitives and attributes originating from the prespecified structural conclusions derived from the lane marker reference patterns. A preliminary stage in establishing a pattern representation may then occur during the construction of the prospective primitive (edge) map where thresholded statistical correlation is performed. Next, the individual prospective primitive components may be combined to form 'primitive segments'. Reference numerical attributes and statistical relationships may be exploited during the segmentation, recognition, and structural analysis stages to help compensate for various sources of pattern variation. Finally, from the degree of correlation between the input and reference patterns, a conclusion may be formulated.

The structure of a lane marker may be formulated via organizing rules composed of primitives, attributes, and other statistically relevant relationships. The primitives and attributes are generated by carefully considering what the constituent components are that make up that pattern and how these components are related. Consider figure X which looks at some of the characteristics of the structure of a broken white lane marker.

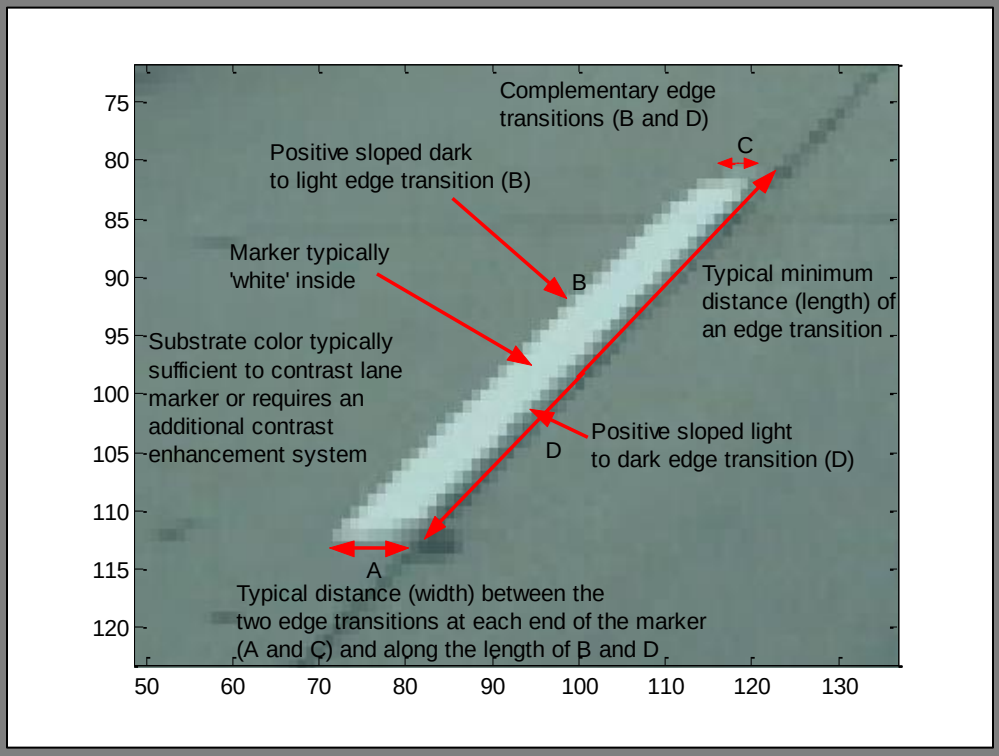


Figure X: Some characteristics of the structure of a broken white lane marker

Also note from figure X that the characteristics for a yellow broken lane marker are very similar, except that the color of the marker is yellow. Thus, some of the primitives which may be looked at for the lane marker of figure X are the edge transitions corresponding to A, B, C and D as shown in figure X. In particular, B and D correspond to the (existence of) positive sloped dark to light edge transitions and positive sloped light to dark edge transitions, respectively. Although the use of letters is typically used to represent pattern primitives, the simple patterns of figure X will have primitives described in terms of numerals (3's and 4's corresponding to B's and D's, respectively).

A couple of points should be made regarding the possible types of primitive data. First, there may be the inclusion of general quantitative data relevant to the pattern primitives *with* and *without* accommodation for noise, distortion, variation, degradation, etc. These values may include (but are not limited to) minimal and maximal distance values and typical distance values associated with the road scene marker primitives dependent on the position of the vehicle within the lane, the type of marker, the location of the marker within the field of view, the dynamic operating conditions of the vehicle, the road geometry, etc. Second, there may be the inclusion of the statistical information pertinent to the analysis of the primitive structure variations caused by noise, distortion, variation, degradation, etc. This information may include (but is not limited to) mean, standard deviation, and variance quantities or hysteresis.

The pattern conclusions attained from the structural analysis have their foundation from the establishment of the structural conclusions. Although the classification method plays an important role in the analysis of longitudinal lane markers, some preliminary comments should be made regarding the rules and regulations governing longitudinal lane markings.

3. PRELIMINARY REGULATORY INFORMATION

3.1 Lane marking function, delineation, and common patterns from the Manual on Uniform Traffic Control Devices for Streets and Highways (*MUTCD)*

Pavement markings may be used alone or in combination with other traffic control devices to convey rules, regulations, and other important characteristics. There are many factors which may affect the perception of longitudinal markings including (but not limited to) those listed in table X on page X. Nonetheless, the 2003 edition of the Manual on Uniform Traffic Control Devices for Streets and Highways (MUTCD) lists the general functions of longitudinal lines as [10]:

1. A double line indicates maximum or special restrictions.
2. A solid line discourages or prohibits crossing (depending on the specific application).
3. A broken line indicates a permissive condition.
4. A dotted line provides guidance.

The Manual on Uniform Traffic Control Devices for Streets and Highways has been adopted as the national standard as a result of federal regulation 23 CFR 655.603 for all traffic control devices installed on any street, highway, or bicycle trail open to public travel [10]. The Federal Highway Administration also plays a significant role in establishing standards for traffic control devices which need to be adhered to. However, it is the Uniform Vehicle Code which provides for individual State Highway Agencies to adopt 'state-specific' manuals and specifications for traffic control devices which meet a specified level of conformance to the most recent national standard as well as conformance to those standards required by the Federal Highway Administrator.

A code has been established for colors which may be used to convey traffic control information. The color white generally indicates regulation while the color yellow generally indicates warning. The colors white and yellow are used for longitudinal line pavement markings with table X listing some of the types of delineation provided.

Table X: Some lane marking delineation based on marker color

White Marking	Yellow Marking
1. Separation of traffic traveling in a uniform direction.	**1**. Separation of traffic flows in opposing directions.
2. Right edge of a roadway.	**2**. Left edge of roadways of divided and one-way highways.
	3. Left edge of the roadway of exit and entrance ramps.
	4. Separation of two-way left turn lanes from other lanes.
	5. Separation of reversible lanes from other lanes.

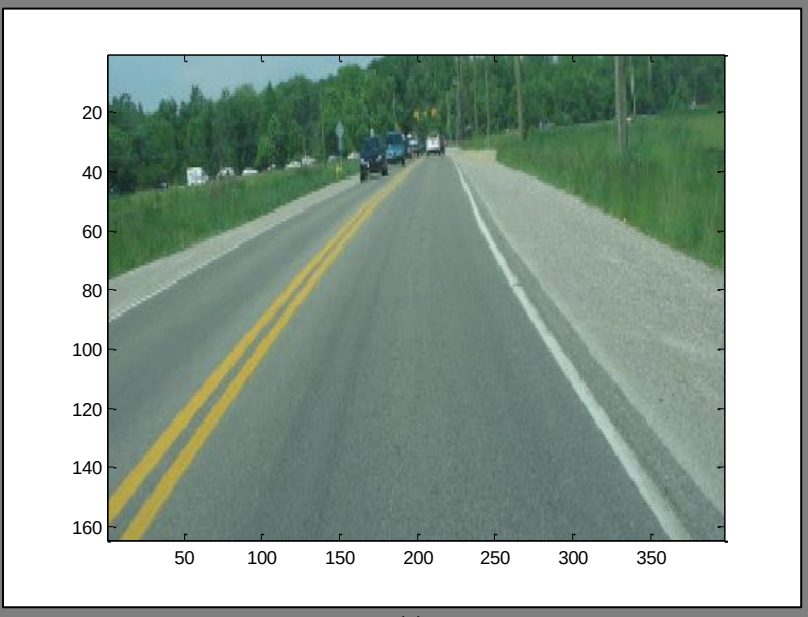
Many substrates are composed of materials which have a 'color' very similar to that specified for white longitudinal line pavement markings. The white marker superimposed on this light colored substrate may not provide a clearly discernible contrast between the two and thus black marking material may be used to provide clearer differentiation between the marker and the substrate.

Four patterns of longitudinal lines have already been mentioned (double, normal, broken, and dotted) along with their general functionality. A fifth pattern which may be encountered is termed 'wide'. These five main patterns and some of their typical associated characteristics are shown in table X.

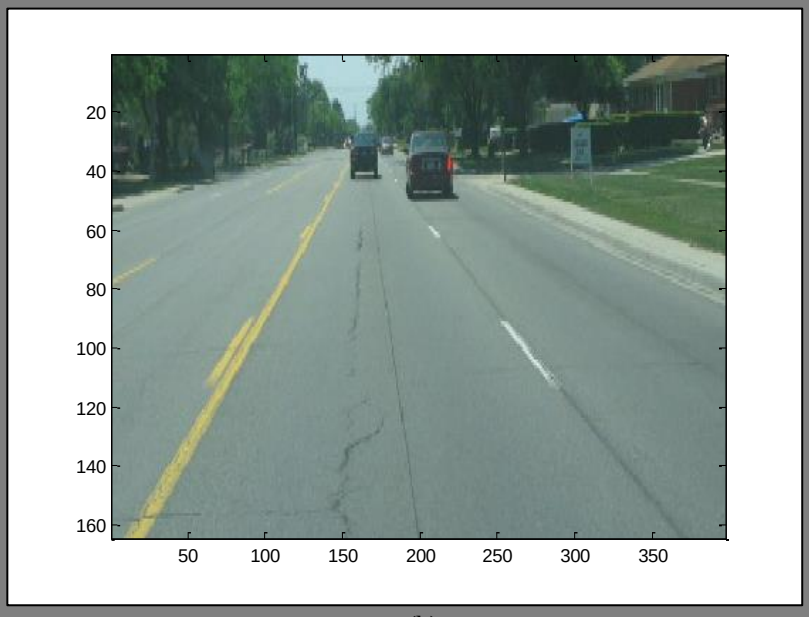
Table X: Five commonly found patterns with associated characteristics

	Pattern	Typical Width	Typical Gap	Typical Segment
1.	Normal	100 mm to 150 mm	None	Location specific
2.	Wide	At least 200 mm to 300 mm (at least twice the width of a normal line)	None	Location specific
3.	Double	2 parallel normal lines separated by a discernible space	None	Location specific
4.	Broken	100 mm to 150 mm	9 m or varies depending on need	3 m or varies
5.	Dotted	At least 100 mm	(line extension) 0.6 m to 1.8 m (lane add/drop) 2.7 m	0.6 m 0.9 m

The five patterns of table X require some elaboration. The first pattern termed 'normal' may be contrasted with the fourth pattern termed 'broken' in that they may be differentiated visually by the presence or absence of a gap or 'interruption' in the marker segment. These intended segment-gap pairs (for the broken marker) are required to appear in segment to gap ratios which provide for sufficient delineation based on factors including (but not limited to) posted speed limits. In figure X on page X, two road scene images appear, each containing a normal marker in conjunction with a broken marker. In figure X (b) the normal marker is white while in figure X (a) the normal marker is yellow. Note however, that in each of (a) and (b) of figure X, the broken marker is white. The widths of the two normal markers appear comparable nearest in the field of view and due to the perspective appear to diminish in width as the markers are observed further into the field of view. Note however that the broken markers at comparable distances in the FOV may appear to have very different lengths. Another point requiring explanation is that for the third pattern termed 'double' in which two parallel normal lines may be located adjacent to each other with a discernible 'space' found between the two as shown in figure X (a). This pattern needs to be distinguished from the potential combination of solid normal yellow and broken normal yellow two-way left turn pair markings as shown in figure X (b).



(a)



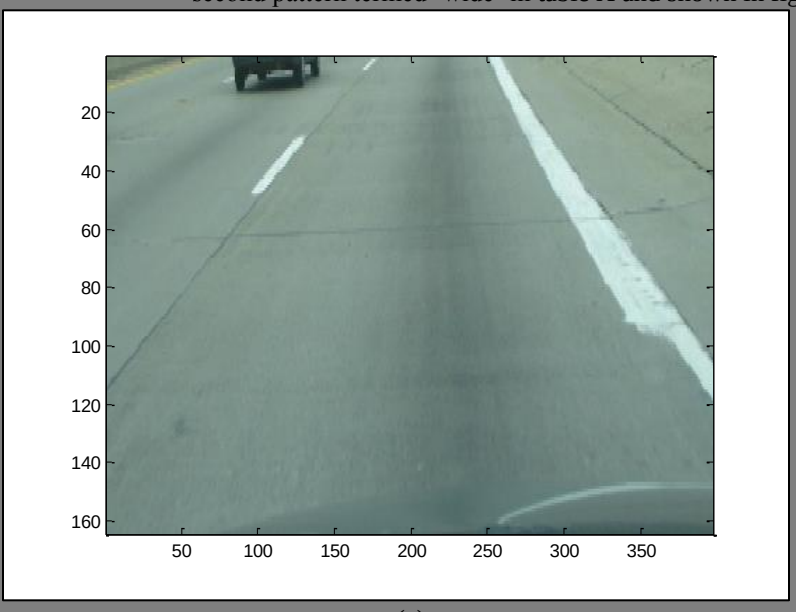
(b)

Figure X: Road scenes (a), (b) showing combinations of different types of lane markers

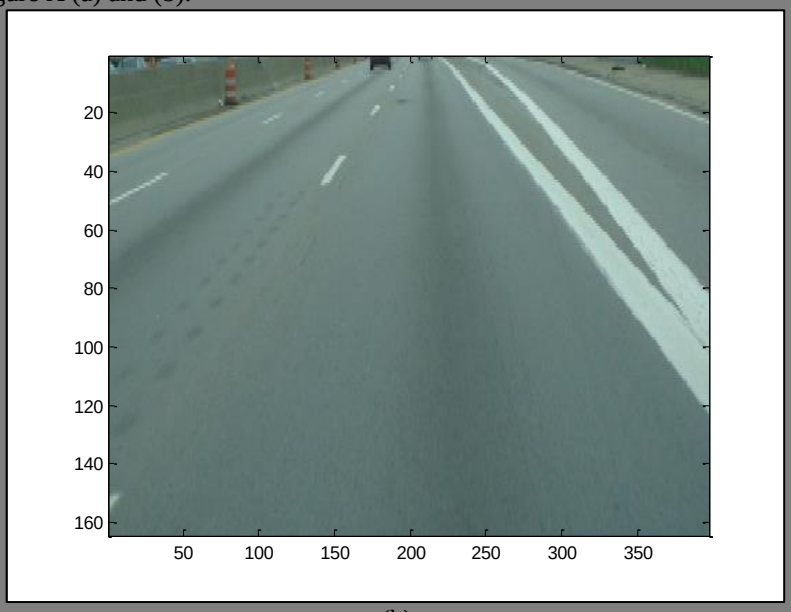
It is apparent in figure X that there is a clearly discernible space between the double yellow centerline markings (for the lower half of the image) of figure X (a) while in figure X (b) the distinction between the solid normal yellow and broken normal yellow two-way left turn lane pair is less obvious. Now consider figure X with similar patterns to those of figure X (b).

Figure X: Road scene with 'normal' and 'broken' combination of yellow longitudinal lane markers

It is clear that the delineation provided by the discernible space between the solid normal yellow and broken normal yellow left turn pair is present but substantially less pronounced than that shown in figure X (b). Now consider the second pattern termed 'wide' in table X and shown in figure X (a) and (b).



(a)



(b)

Figure X: Road scenes (a), (b) showing 'wide' channelizing lane markers

It is apparent from figure X (b) that there is a clear difference between the widths of the wide longitudinal marker used near the exit ramp appearing on the right hand side of the image and the broken white lane marker delineator on the left hand portion of the image. The distinction in widths is also evident in figure X (a) in which the beginning of a wide lane marker continued from a white normal line is evident near the beginning of an entrance ramp. It is important to note in figure X (a) that the increase in marker width due to the transition from normal white marker to wide white marker is only one possible example in which a marker width may increase as it's observed further into the field of view. It is also important to note that the degree of emphasis for a longitudinal line is related to its width. The last pattern from table X which requires some further explanation is termed 'dotted' and is shown in figure X.

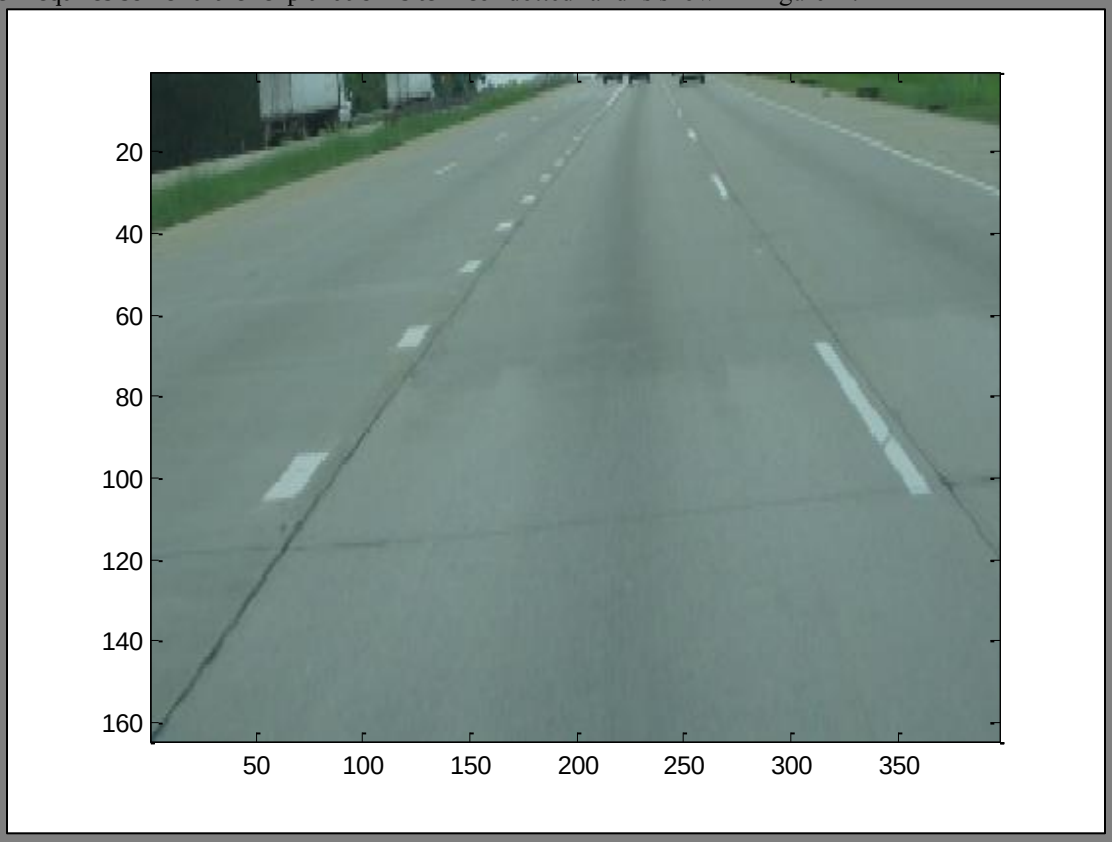


Figure X: Road scene showing 'broken' and 'dotted' longitudinal lane marker distinction

It is clear from figure X that the length of the segment used for the dotted lane marker on the left hand side of the image is substantially less than the length of the segment used for the broken lane marker on the right hand side of the image. It is also apparent that the distance of separation (gap) between the dotted lane markers is substantially less than the gap between the broken markers.

3.2 Other resources for regulatory information

There are many other sources of information which may be referenced with regards to rules and regulations pertaining to federal, state, and local policies including:

1. Uniform Vehicle Code (UVC) and Model Traffic Ordinance, 2000 Edition.
2. The Code of Federal Regulations (CFR).
3. The current individual State Highway Agency traffic control device manuals and specifications.
4. Highlights of major changes to the 2003 MUTCD Publication # FHWA-HOP-4-042.
5. U.S. Pavement Markings Publication # FHWA-OP-02-090.
6. The current national guidelines for traffic control devices used by countries around the world.

Although only a tiny percentage of the regulatory requirements have been preliminarily considered, the actual image must first be stored in a recognizable format before the actual analysis may proceed.

4. IMAGE REPRESENTATION FORMATS

4.1 Image representation formats and color spaces

There are many different file formats that have evolved over the years including the tagged image format, the motion pictures experts group format , the joint photographic experts group format, and the graphics exchange format, just to name a few. Although a major focus of this paper will be on 'approximated grayscale' image processing, there is a great deal of information contained within the color components of a road scene image. As a matter of fact, color is one of the most important properties which may be used in extracting information from an image. For further information on color, the reader may refer to [14]. Take for example the road scene of figure X. There are both yellow lane markers and white lane markers which are present in the image, with each color having a special relevance. Road signs (with distinct color attributes) may be interpretable using chrominance information but less so using only luminance information. Traffic lights with their red, yellow, and green colors may only be distinguishable when considering their chrominance characteristics. Similarly, the chrominance information of brake lights for a car may provide information to the image processing system that it cannot extract from luminance information alone. However, consider also the possible scenario in which natural or artificial light sources contribute to chrominance characteristics for a lane marker which are not indicative of the intended lane marker chrominance properties.

4.2 Generating approximated 'grayscale' data from the RGB color space

As mentioned previously, a JPEG image can be read into Matlab using the imread function where it is stored in a multi-dimensional matrix variable. It may be necessary to convert from the RGB color space to the YIQ color space to enable grayscale image processing techniques. This may be accomplished in the Matlab preliminary road scene image processing software through the use of the function rgb2yiq.m, given in the code listing portion in figure X.

```
function [image_data] = rgb2yiq(image_data)
% Programmer: Christopher Alan Warner
% RGB2YIQ is a function that will transform an image from the RGB color space to an
% approximated YIQ color space.
% ================================================================
%     INPUT LIST (in order that they appear as parameter list):
% ================================================================
% 1) IMAGE_DATA = contains image_data in RGB format
% ================================================================
%     OUTPUT LIST (in order that they appear as parameter list):
% ================================================================
% 1) IMAGE_DATA = contains image_data in approximated YIQ format
% ================================================================
%     OTHER CUSTOM FUNCTIONS USED
% ================================================================

rows = []; cols = []; depth = [];
[rows, cols, depth] = size(image_data);

yiq = zeros(rows,cols,depth);

[rgb_to_yiq_tf] = double([0.299,  0.587,  0.114,
                          0.596, -0.274, -0.322,
                          0.211, -0.523,  0.312]);
[yiq_to_rgb_tf] = double([1.000,  0.9562,  0.621,
                          1.000, -0.2727, -0.6468,
                          1.000, -1.1037,  1.7006]);
```

Figure X: Code listing portion for function rgb2yiq.m

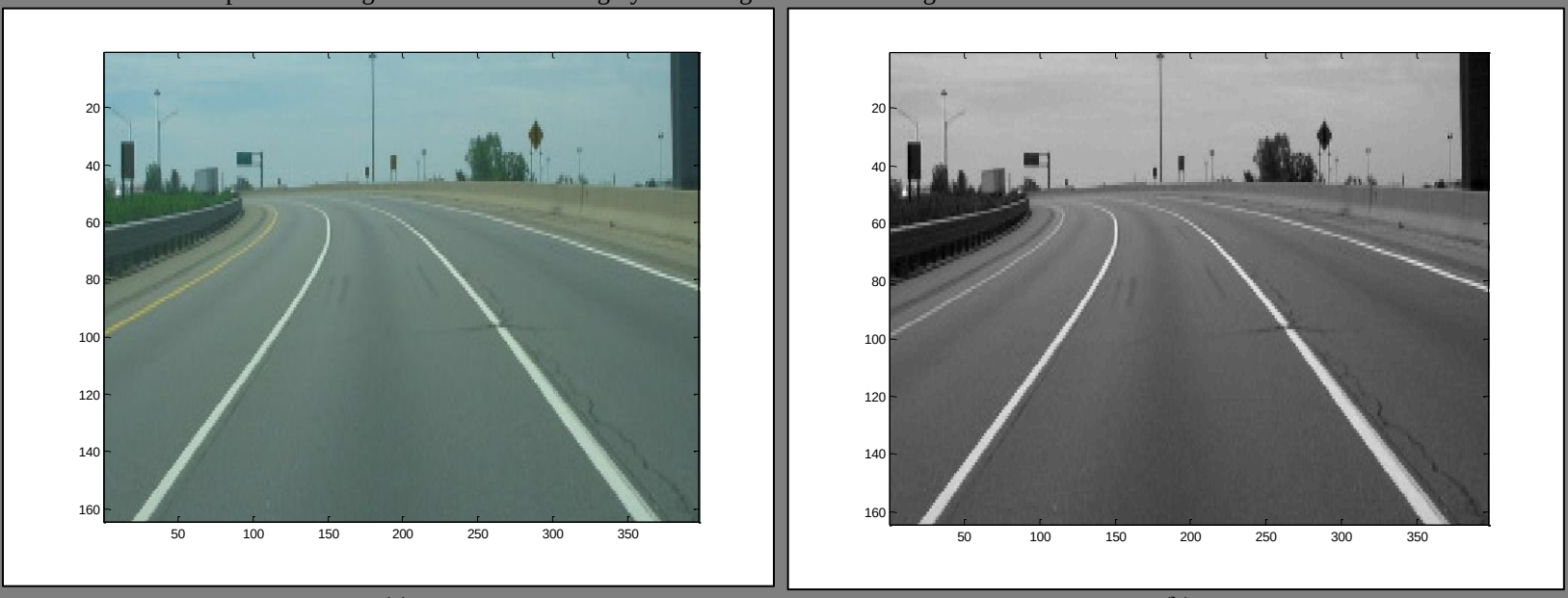
```
for c = 1:rows
   for d = 1:cols

      r = []; g = []; b = []; prod = []; rgb = [];

      [r] = image_data(c,d,1);
      [g] = image_data(c,d,2);
      [b] = image_data(c,d,3);

      [rgb] = double([r g b]);
      prod = [rgb] * [rgb_to_yiq_tf];

      yiq(c,d,1) = prod(1);
      yiq(c,d,2) = prod(2);
      yiq(c,d,3) = prod(3);

   end;
end;

image_data = yiq(:,:,:);
```

Figure X (continued): Code listing portion for function rgb2yiq.m

As can be seen from figure X, the function starts off by referencing the transfer function to conv0ert to YIQ coordinates from RGB coordinates from [23]. This linear transfer function is given by

$$[Y\ I\ Q]^t = \begin{bmatrix} 0.299 & 0.587 & 0.114 \\ 0.596 & -0.274 & -0.322 \\ 0.211 & -0.523 & 0.312 \end{bmatrix} \times [R\ G\ B]^t . \qquad \textbf{(X)}$$

It is important to note that the first row of the transfer function sums to one while the second and third rows of the transfer function sum to zero. This transfer function appears at the beginning of the function listed in figure X. Performing the necessary multiplication generates the [Y I Q] values including the luminance (or intensity) values. Once the transform of equation X has been performed, the grayscale approximated data is taken from the luminance data. A sample RGB image and its transformed grayscale image are shown in figure X.



(a) (b)

Figure X: (a) Cropped RGB image; (b) grayscale approximation to RGB image from rgb2yiq.m

Table X shows the grayscale values taken from figure X (b) from the (row,column) range (157:164,19:32). These values correspond to a small portion of the positive sloped normal white longitudinal marker found in the lower left corner of the image of figure X (b). Note that these are data of class double to two decimal points and have not been normalized to the range [0,1].

Table X: Grayscale approximated image data from figure X (b) (rows 157:164, columns 19:32)

	C19	C20	C21	C22	C23	C24	C25	C26	C27	C28	C29	C30	C31	C32
R157	119.14	116.93	119.14	114.71	115.82	118.45	114.03	114.21	120.84	121.1	128.84	191.97	217.41	211.75
R158	118.03	116.93	116.93	114.71	116.24	110.71	116.66	124.67	109.18	151.26	213.2	213.16	210.95	209.54
R159	114.71	115.82	113.61	115.82	117.35	118.45	114.45	113.61	184.39	216.51	212.09	210.65	209.54	216.18
R160	112.5	114.71	110.29	115.82	118.45	110.71	146.52	205.7	216.76	209.07	210.18	215.07	210.65	211.75
R161	120.42	118.21	111.57	115.99	115.87	158.84	213.72	213.03	207.5	208.44	216.6	208.56	214.3	207.49
R162	113.78	117.1	118.21	135.9	193.29	216.35	213.72	207.5	213.97	210.65	212.18	209.88	197.71	162.15
R163	114.89	114.89	148.07	202.26	212.9	205.12	210.82	210.65	208.44	209.96	211.28	192.18	155.51	119.43
R164	120	178.91	215.41	208.77	206.56	208.44	208.44	207.33	212.18	203.03	178.91	146.66	123.44	118.33

The YIQ color space is not the only one which includes information which may be used as grayscale image data. The Hue-Saturation-Value (HSV) color space includes intensity information which also may be used in grayscale analysis as well.

The color of blue is said to reflect ‘blue hue’ while the color green is said to reflect ‘green hue’. Each color on the hue axis falls between zero and three hundred sixty degrees with complementary colors being offset by one hundred eighty degrees. Hue is one of the chrominance components while the other is saturation. Saturation indicates the ‘level of whiteness’ or ‘purity’ within a color. A color with very high saturation has very little white (close to a pure color) while a color with very low saturation has a great deal of white (a substantially lightened version of the pure color). Note that the saturation may also be affected by the value level. Value (or intensity) indicates the level of ‘brightness’. Although luminance quantities may be analyzed in terms of value or intensity levels, it should be noted that luminance factors (percentages) have been established to assist in measurement and comparison to industry standards (ASTM, etc.). Table X from [17] shows some of the proposed daytime luminance factors for retroreflective pavement marking materials.

Table X: Proposed daytime luminance factors (%) for retroreflective pavement marking material with CIE 2° standard observer and 45/0 (0/45) geometry and CIE standard illuminant D_{65}

COLOR	Luminance Factor (Y%)	
	Minimum	Maximum
Yellow	25	
White	35	

The conversion from the RGB color space to the HSV color space for comparison of the YIQ and HSV generated intensity values as well as to observe certain chrominance characteristics of a road scene image may be performed in the Matlab preliminary road scene image processing software through the use of the function rgb2hsv.m, given in the code listing portion in figure X.

```
function [image_data] = rgb2hsv(image_data)
% Programmer: Christopher Alan Warner
% RGB2HSV is a function that will transform an image from the RGB color space to an
% approximated HSV color space.
% ================================================================
%     INPUT LIST (in order that they appear as parameter list):
% ================================================================
% 1) IMAGE_DATA = contains image_data in RGB format
% ================================================================
%     OUTPUT LIST (in order that they appear as parameter list):
% ================================================================
% 1) IMAGE_DATA = contains image_data in approximated HSV format
% ================================================================
%     OTHER CUSTOM FUNCTIONS USED
% ================================================================
rows = []; cols = []; depth = [];
[rows, cols, depth] = size(image_data);

h = zeros(rows,cols); s = zeros(rows,cols); v = zeros(rows,cols);
hsv_image_data = zeros(rows,cols,depth);

max_value = 255;
normalized_rgb = zeros(rows,cols,depth);
normalized_rgb = double([image_data]) ./ max_value;

for c = 1:rows
   for d = 1:cols

      r = []; g = []; b = []; rgb = []; min = []; max = []; i = []; j = [];

      [r] = normalized_rgb(c,d,1);
      [g] = normalized_rgb(c,d,2);
      [b] = normalized_rgb(c,d,3);

      [rgb] = double([r g b]);
```

PORTION OF FIGURE INTENTIONALLY REMOVED

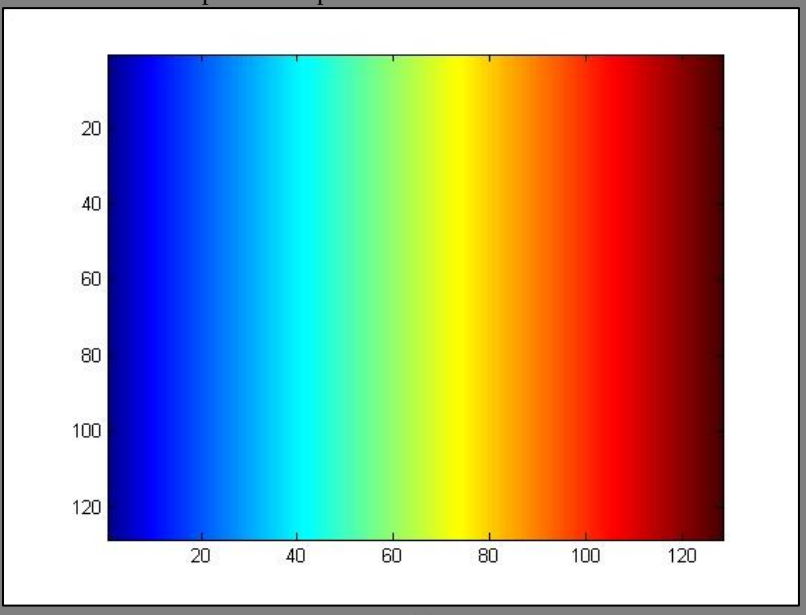
Figure X: Code listing portion for function rgb2hsv.m

PORTION OF FIGURE INTENTIONALLY REMOVED

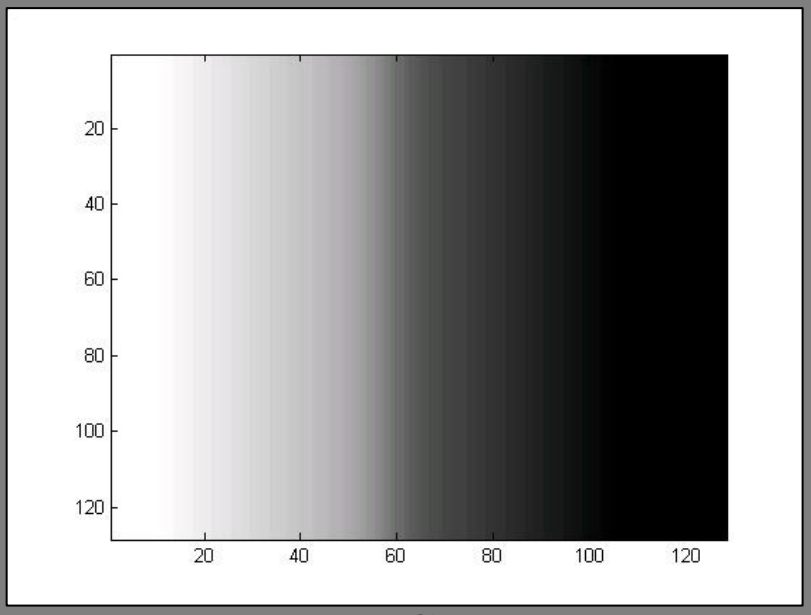
```
hsv_image_data(:,:,1) = h(:,:);
hsv_image_data(:,:,2) = s(:,:);
hsv_image_data(:,:,3) = v(:,:);

image_data = hsv_image_data;
```

Figure X (continued): Code listing portion for function rgb2hsv.m

Figure X shows the hue, saturation, and value channel outputs obtained from the code portion of figure X for a sample input image generated using code similar to the Matlab command jet which is the default colormap for several specialized plots.

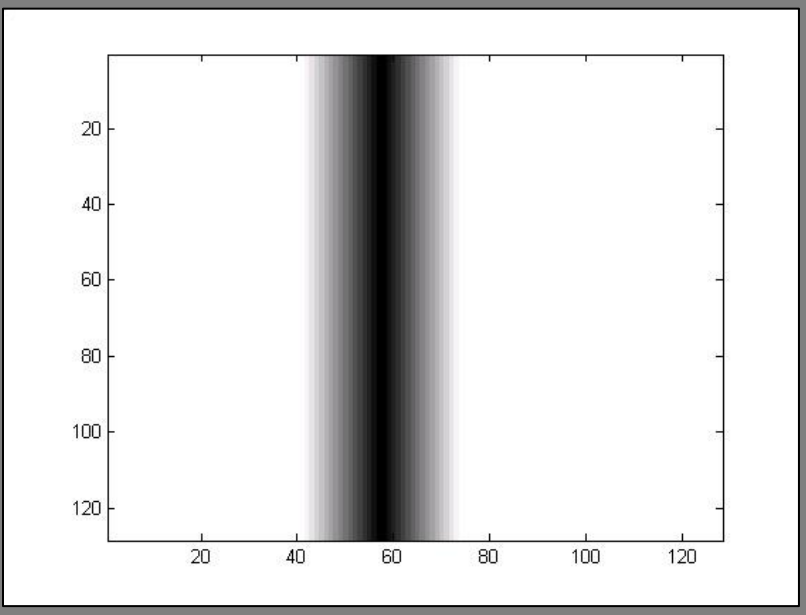


(a)

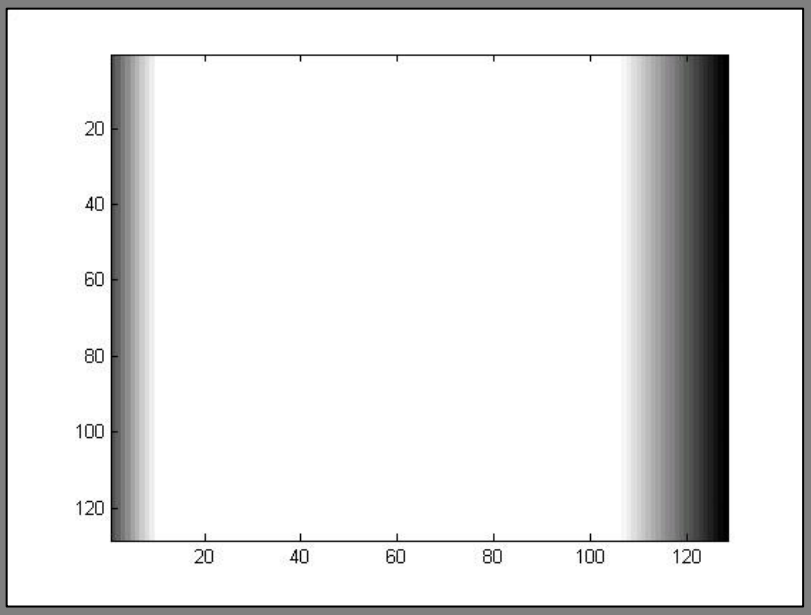


(b)

Figure X: (a) Sample image covering large range of image colors; (b) hue channel output of function rgb2hsv.m



(c)

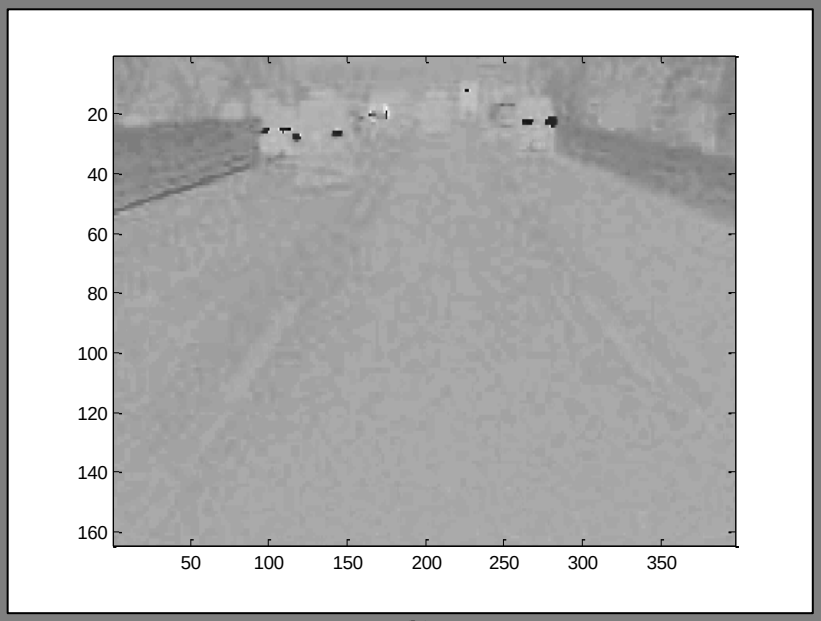


(d)

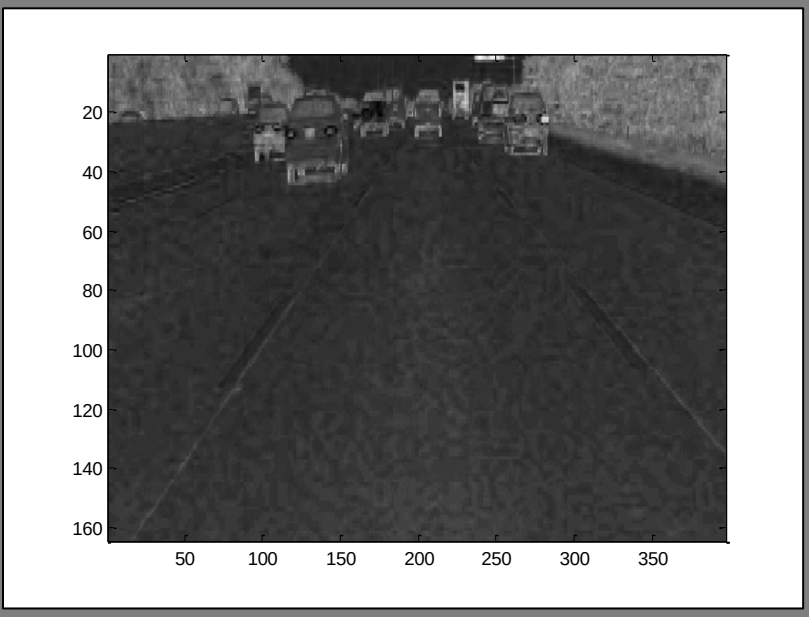
Figure X (continued): (c) Saturation channel output of function rgb2hsv.m; (d) value channel output of function rgb2hsv.m

As can be seen in figure X (b), the highest levels of hue occur furthest to the left in the figure corresponding to the colors with the strongest hues of blue. Moving from left to right across the output from the hue channel, a reasonable linear transition can be inferred visually. In figure X (c), the output from the saturation channel can be seen. Note that the darkest color shades have the highest saturation levels indicating that these colors have the least amount of 'whiteness'. Figure X (d) shows the output from the value channel which describes the level of 'brightness' within a particular color. Notice from figure X (d) that the dark blues and dark reds have the lowest level of brightness. It should also be noted that the 'color' black (although not always considered a chrominance value) has a value (or intensity) of zero. These channels may provide useful information when analyzing road scene images because longitudinal lane markers, other traffic control devices, and other objects which may found in a road scene have intentional chrominance and luminance characteristics. A sample RGB image and its HSV channel outputs are shown in figure X.

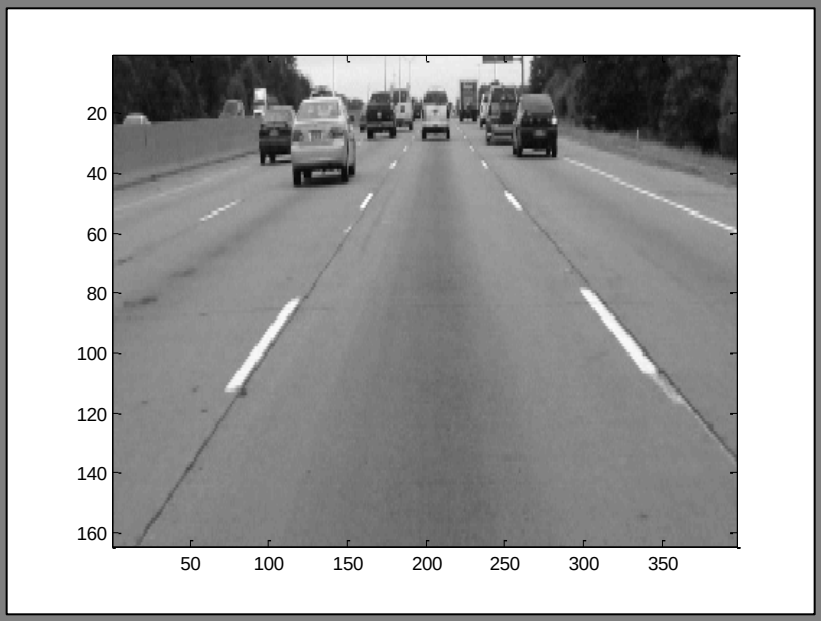
(a)



(b)



(c)

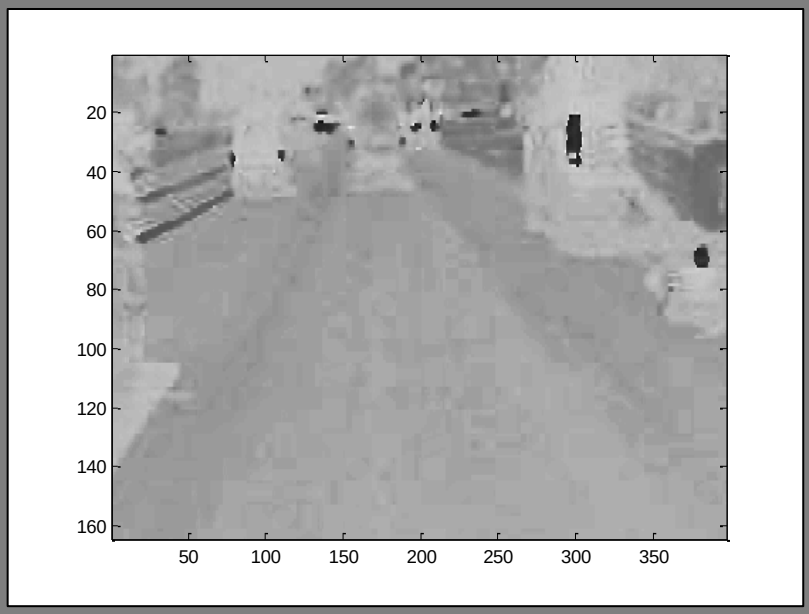


(d)

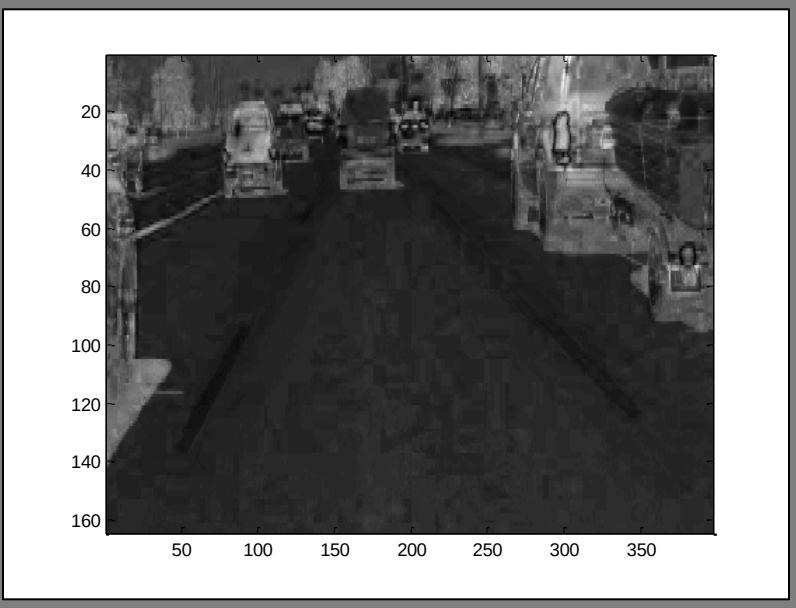
Figure X: (a) Cropped RGB image; (b) hue channel output; (c) saturation channel output; (d) value channel output

Another set of road scene images with the hue, saturation, and value channel outputs are shown in figure X.

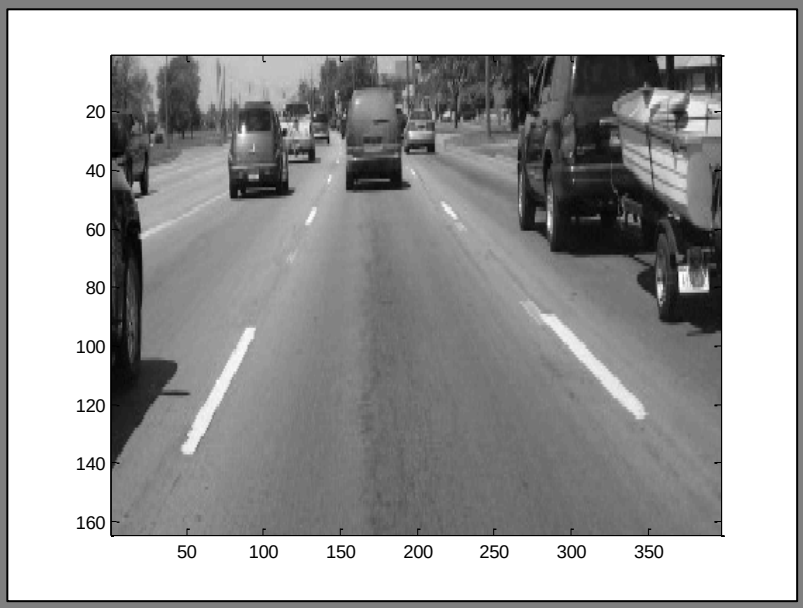
(a)



(b)



(c)



(d)

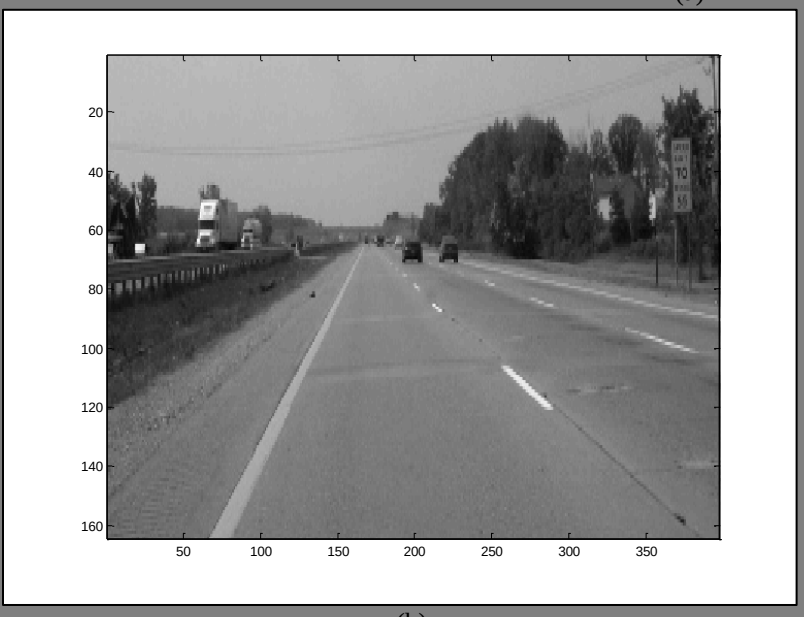
Figure X: (a) Cropped RGB image; (b) hue channel output; (c) saturation channel output; (d) value channel output

There are many different transfer functions of varying complexities for transforming values in the RGB color space to those in the HSV color space.

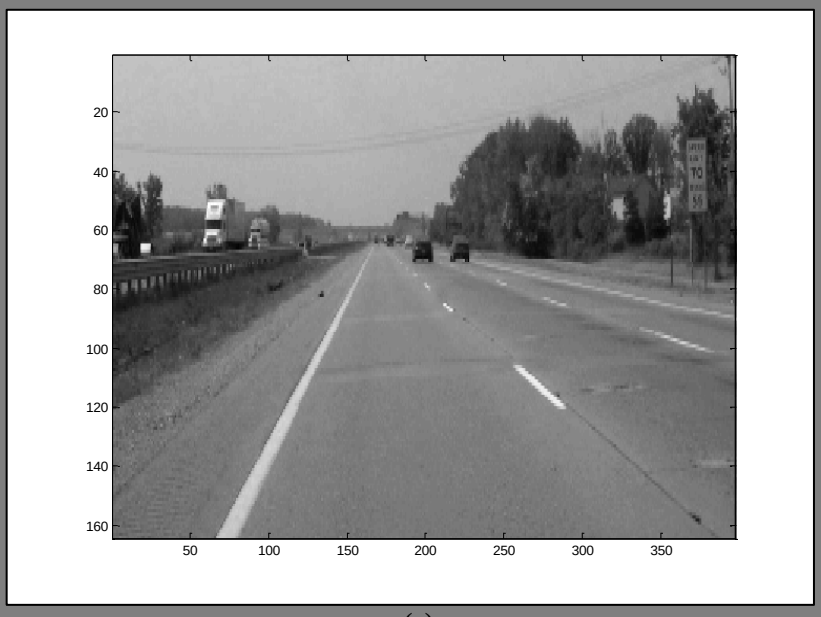
4.3 Comparison of data obtained from RGB to HSV and RGB to YIQ transforms

The luminance data obtained from the function rgb2hsv.m may be compared to the luminance data obtained from the function rgb2yiq.m. Figure X shows a cropped RGB road scene image along with the intensity values generated from each of the functions rgb2yiq.m and rgb2hsv.m for the same image.

(a)



(b)



(c)

Figure X: (a) Original cropped RGB image; (b) intensity image from YIQ transform; (c) value image from HSV transform

It is interesting to note that the two grayscale images of figure X (b) and (c) have very similar appearances. Upon visual inspection, the white broken lane markers appear to have comparable 'brightness' in each of the images. When taking a

closer look at the actual data for the white broken lane markers, it turns out that the ratio of marker to non-marker for values returned from the function rgb2hsv.m varies only slightly with the ratio of marker to non-marker for values returned from the function rgb2yiq.m. However, it is visually observable that the normal yellow lane marker from the image of figure X (b) is substantially 'less bright' than that of the yellow lane marker from the image of figure X (c). Upon taking a closer look at that (yellow marker) data, the values returned from the function rgb2hsv.m are substantially larger than the values returned from the function rgb2yiq.m. When looking closely at the data, the ratio of yellow marker to non-marker values obtained from rgb2hsv.m is generally higher than the ratio of yellow marker to non-marker values obtained from rgb2yiq.m. Table X shows a small sample of the data for the white markers from images (b) and (c) of figure X while Table X shows a small sample of the data for the yellow markers from the same images. Note that the data in Table X (a) is of class double to two decimal points and has not been normalized to values between 0 and 1.

Table X: Portion (rows 156:164, columns 67:81) of image data of figure X from approximation for normal yellow marker (a) from rgb2yiq.m; (b) from rgb2hsv.m

	C67	C68	C69	C70	C71	C72	C73	C74	C75	C76	C77	C78	C79	C80	C81
R156	122	123.28	125.19	127.16	129.76	125.56	126.77	120.68	145.73	167.48	167.85	165.65	166.75	166.32	161.6
R157	125.49	119.96	127.28	126.35	129.09	129.28	109.7	145.06	170.11	166.65	163.86	167.05	164.66	161.2	165.33
R158	127.7	129.67	127.28	127.98	131.49	111.91	153.34	170.4	169.67	162.87	164.96	164.66	161.59	165.63	165.63
R159	132.69	126.7	125.82	127.63	113.19	153.69	165.89	169.1	168.65	165.94	165.65	164.66	167.72	167.54	162.22
R160	128.26	129.98	133.58	104.38	151.38	166.79	170.09	165.37	165.63	167.65	163.1	166.75	164.79	164.4	164.23
R161	133.51	130.94	117.08	144.6	167.65	167.54	164.79	165.89	163	162.75	163.86	166.07	165.09	164.1	163.45
R162	132.98	123.96	149.9	165.9	164.69	163.3	163.68	166.61	163.3	160.54	159.26	164.53	164.44	161.2	158.67
R163	106.26	148.76	164.79	167.07	164.91	167	164.79	162.7	162.87	161.35	160.66	162.35	158.47	160.59	150.78
R164	151.99	168.07	166.65	162.58	166.88	163.68	165.09	161.59	162.57	162.75	161.47	161.93	156.71	151.24	140.78

(a)

	C67	C68	C69	C70	C71	C72	C73	C74	C75	C76	C77	C78	C79	C80	C81
R156	118	119	120	123	125	123	124	119	152	187	188	187	188	187	184
R157	122	117	123	123	126	127	110	119	181	186	185	187	183	178	183
R158	124	126	124	125	129	112	154	183	186	183	186	185	180	180	182
R159	128	123	123	126	114	156	175	182	183	186	187	185	187	185	180
R160	123	126	130	106	150	176	186	183	183	189	186	188	186	184	182
R161	130	129	117	150	179	185	186	183	183	184	185	187	185	181	176
R162	130	123	154	177	182	183	185	183	183	182	181	185	184	176	168
R163	107	153	176	186	187	188	186	183	183	182	181	183	176	173	156
R164	157	179	184	184	189	185	185	183	183	180	179	179	170	156	139

(b)

Table X: Portion (rows 112:120, columns 274:288) of image data of figure X from approximation for broken white marker (a) from rgb2yiq.m; (b) from rgb2hsv.m

	C274	C275	C276	C277	C278	C279	C280	C281	C282	C283	C284	C285	C286	C287	C288
R112	143.87	136.13	146.08	147.19	147.48	144.97	145.15	139.32	139.32	138.22	141.84	138.52	136.01	138.22	136.01
R113	225.14	189.76	147.61	137.65	144	145.1	145.82	145.22	139.32	144.98	140.73	138.51	139.41	138.3	136.09
R114	225.15	224.34	226.13	193.25	154.54	144.29	146.08	144.98	146.08	142.76	140.68	142.89	138.64	137.53	136.09
R115	214.26	228.34	227.66	224.34	225.45	195.16	153.7	132.39	130.77	117.75	121.88	132.25	138.47	140.68	136.3
R116	140.97	168.62	207.63	226.43	226.55	225.02	228.52	199.76	147.48	123.28	77.756	138.08	143.61	141.4	137.23
R117	144.17	144.17	144.29	163.09	208.44	227.54	228.76	225.15	225.45	198.9	155.35	128	134.63	142.37	143.87
R118	141.54	143.75	140.85	146.38	144.17	165.31	204.43	228.38	227.66	227.66	228.34	206.22	161.98	131.32	144.59
R119	137.41	140.73	143.75	140.43	144.85	141.96	144.29	159.6	197.5	225.15	227.87	228.98	228.47	204.47	158.67
R120	139.02	140.13	138.3	137.2	137.2	137.11	141.54	137.96	143.19	153.18	175.42	194.64	203.07	204.47	193.89

(a)

	C274	C275	C276	C277	C278	C279	C280	C281	C282	C283	C284	C285	C286	C287	C288
R112	139	131	141	142	143	140	140	134	133	132	135	132	130	132	131
R113	215	182	143	133	137	138	139	139	136	139	136	134	135	134	132
R114	214	213	215	184	149	139	140	139	140	137	134	136	133	132	132
R115	203	216	216	213	214	186	149	129	127	114	117	126	132	134	132
R116	135	160	197	214	215	214	217	191	143	120	78	132	137	135	133
R117	138	138	138	155	197	215	217	214	214	189	150	125	131	138	139
R118	136	138	135	140	138	157	194	216	216	216	217	197	156	128	140
R119	133	136	138	135	139	136	139	152	187	212	215	216	215	196	152
R120	135	136	134	133	133	132	136	131	137	145	166	183	191	192	183

(b)

As may be seen from the data of table X, the HSV transform may produce greater luminance contrast between yellow longitudinal lane markers and the surface these markers are applied to. Thus, the choice of transform used to generate the luminance data may have a significant impact on subsequent processing. However, great care must be used in choosing a transform which will maintain the inherent color traits (the reason for having color distinctions for longitudinal lane markers) as well as the integrity in those interrelationships.

The luminance approximations derived from the various transforms (applied to the RGB data) establish a range of values that a ‘grayscale’ pixel may take on. While engineering data should be normalized whenever appropriate, the choice to represent image data in a non-normalized fashion was made because it is much simpler to ‘comprehend’ and ‘distinguish between’ values such as 250, 251, and 252 than it is for values such as 0.980392, 0.984313, and 0.988235. Also, factors such as rounding and the number format used should be chosen to attain the correct level of accuracy and precision for a given set of circumstances. This portion of the paper covered some aspects of image formats, transformations from one coordinate space to another, and some numerical specifics. The next portion of this paper deals with image pre-processing which prepares the transformed image information for subsequent analysis.

5. IMAGE PRE-PROCESSING

5.1 Overview of pre-processing

Image pre-processing includes any operations performed on the 'raw' image including (but not limited to) cropping, contrast enhancement, and filtering which produce an image format more suitable for subsequent processing tasks. Note that great care must be taken when performing this step to assure that meaningful information is not degraded or eliminated. This could potentially inhibit future processing steps or change the characteristics of the original signal in an unintended fashion.

The generalized filtering process, as shown in figure X, will be considered to highlight the use of filters and some basic filtering techniques as applied in the Matlab programming environment.

Figure X. Signal filtering process

When considering the filtering of one-dimensional signals, a filter with transfer function

$$H(s) = B(s) / A(s) \qquad \textbf{(X)}$$

will have a frequency response composed of a magnitude response and phase response. The magnitude response reflects the change in magnitude of the signal as a function of frequency while the phase response reflects any change in angle of the signal (phase shift) as a function of frequency. One goal of one-dimensional noise filtering is generally to produce the desired magnitude response (removal of noise) with linear phase (also called zero phase). The removal of some component of (only) the noise will increase the signal-to-noise ratio. If the noise filter does produce a phase shift, then the signal has changed in a way that's altered its original characteristics.

Similarly, one goal of image filters is to maintain the original spatial-structural relationships without blurring or degrading relevant characteristics. If a detrimental change were to occur, then confidence in the resulting relative object properties would also be compromised. Thus, a filter which 'smoothes' non-edges but maintains significant edge characteristics is highly desirable. However, this requires some predetermined understanding of what is considered significant and moreover what is deemed insignificant (under a vast array of conditions).

5.2 Noise reduction filters and median filters

Three simple masks for noise reduction are shown in figure X [4].

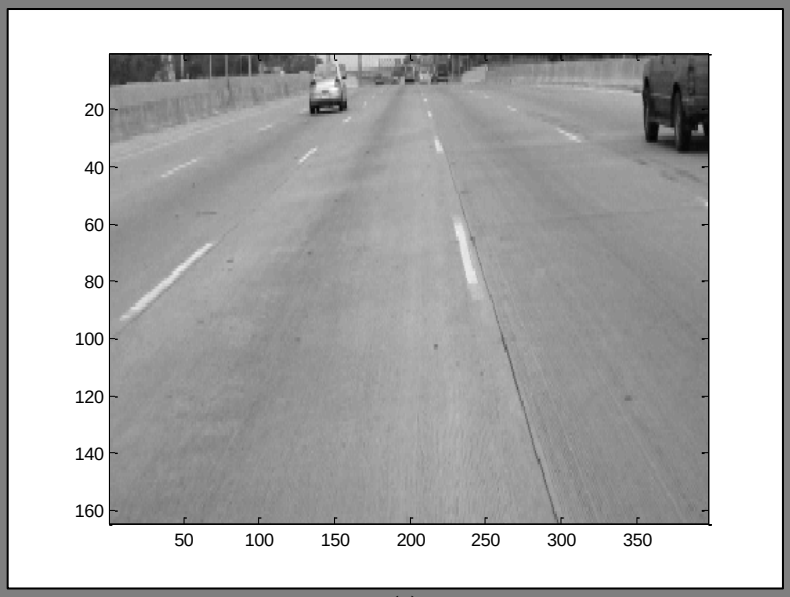
$$\begin{bmatrix} 1 & 1 & 1 \\ 1 & 1 & 1 \\ 1 & 1 & 1 \end{bmatrix} \qquad \begin{bmatrix} 1 & 2 & 1 \\ 2 & 4 & 2 \\ 1 & 2 & 1 \end{bmatrix} \qquad \begin{bmatrix} 1 & 2 & 3 & 2 & 1 \\ 2 & 4 & 6 & 4 & 2 \\ 3 & 6 & 9 & 6 & 3 \\ 2 & 4 & 6 & 4 & 2 \\ 1 & 2 & 3 & 2 & 1 \end{bmatrix}$$

w = 1/9 w = 1/16 w = 1/81

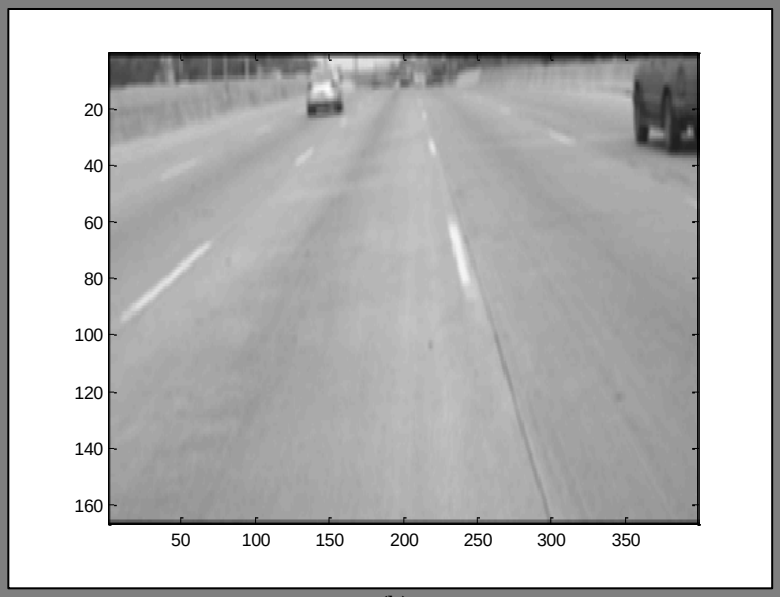
(a) (b) (c)

Figure X. Masks for noise reduction with weight value (a) 1/9; (b) 1/16; (c) 1/81

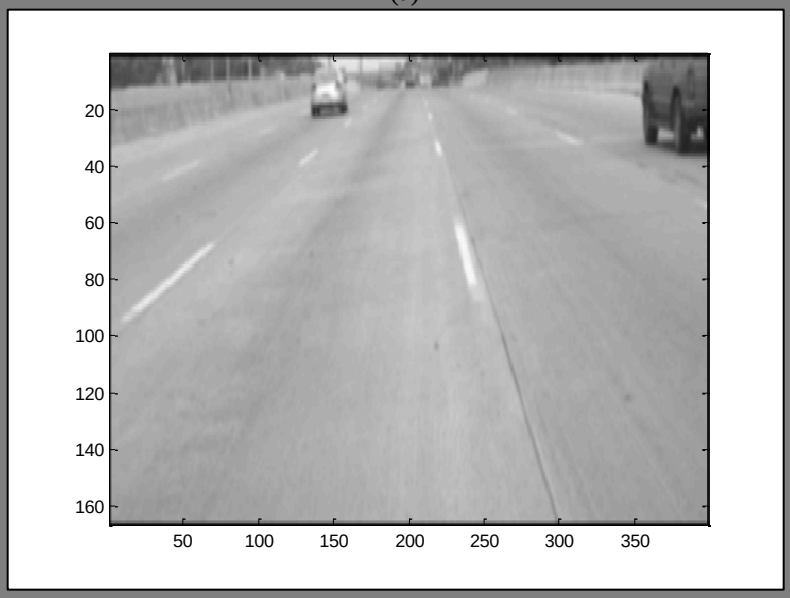
The following figure shows a grayscale road scene image and the output of the three noise reduction masks shown in figure X.



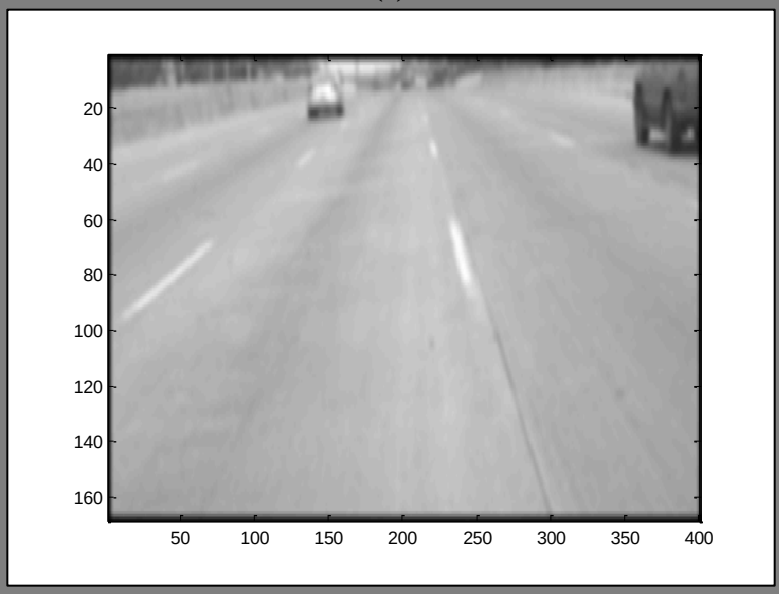
(a)



(b)



(c)



(d)

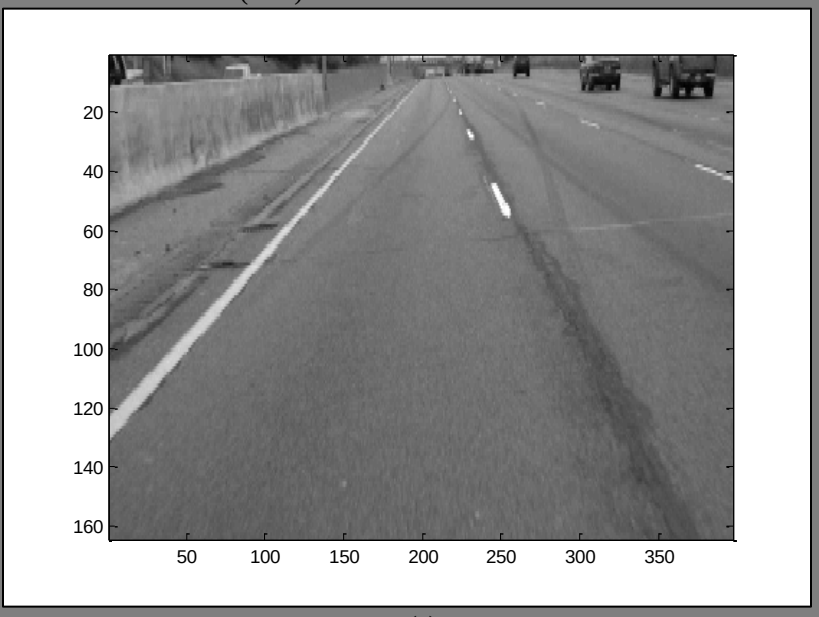
Figure X: (a) Grayscale image; (b) output of mask with weight 1/9; (c) output of mask with weight 1/16; (d) output of mask with weight 1/81

The mask of figure X (a) is considered a 'box' or 'smoothing' filter in which the weighting at each of the mask positions is equal. This filter takes the average value of the pixels in the 3x3 neighborhood and places it at the center pixel position in the resulting filtered image. The masks of figure X (b) and (c) may be considered as approximations of a Gaussian filter. Note that the weighting of the input pixels is decreased with increasing distance from the center pixel. The two-dimensional Gaussian filter may be represented as:

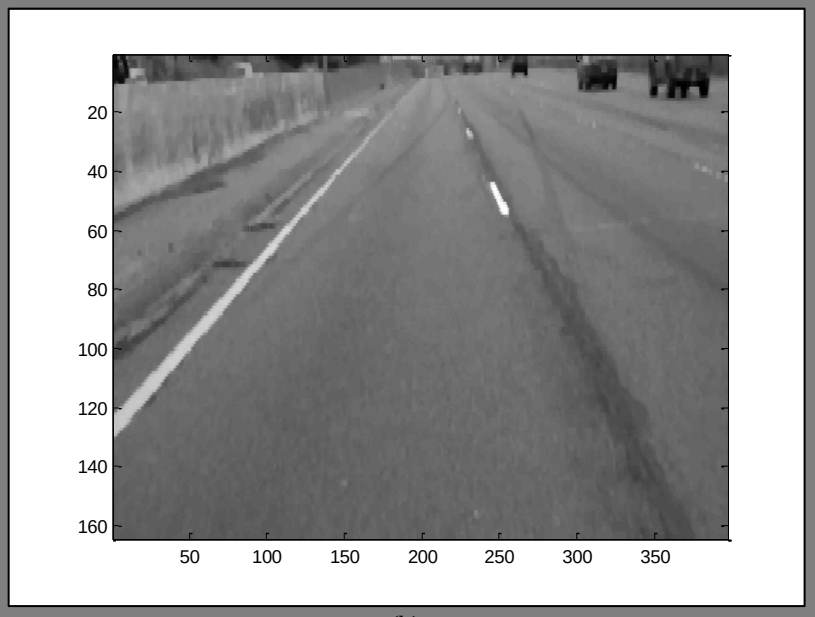
$$G(x, y) = \frac{1}{\sqrt{2\Pi\sigma}} e^{\frac{-(x^2 + y^2)}{2\sigma^2}} . \qquad \textbf{(X)}$$

Some of the potentially useful properties of Gaussian filters are the smooth decrease in weights, the degree of smoothing based on the selected value of the standard deviation 'σ', and the symmetry about the abscissa. It should be noted that Gaussian filters may be used in combination with other Gaussian filters and/or derivative operators to isolate different features within an image.

As can be seen from figure X, these filters will tend to smooth the detail within an image and different types of masks will produce different degrees of smoothing. Note that no outer edge values have been added to the filtered image, resulting in a dark shaded outer boundary to the images of figure X (b), (c), and (d). However, by selecting the weight values used in the convolution based on the spatial configuration of the neighborhood, blurring may be lessened across certain boundaries or edges of interest. There is also another class of filters (rank-value filters) in which a specific value from within a neighborhood may be chosen based upon some specific criterion. An example of a type of rank-value filter is one which will only select values from within the neighborhood that are within a specific statistical range. Another example is the median filter. Figure X shows an approximated grayscale image and the output of a (3X3) median filter.



(a)



(b)

Figure X: (a) Grayscale image; (b) output of 3x3 median filter

From a preliminary visual inspection of figure X (b), the median filter seems to smooth some of the potentially spurious detail while avoiding blurring many of the dominant lane marker characteristics. The yellow lane marker on the left side of figure X (a) appears to have comparable grayscale values to that of the median filtered output shown in figure X (b). Some of the finer detail of the road scene image has been removed but the white broken marker closer in the FOV appears to be very similar to that appearing in the original grayscale approximation. However, some of the broken white lane markers (for the current lane) further into the FOV appear to be blurred as do the broken white adjacent lane markers. This should not pose a problem as a region of interest (ROI) will be established in an upcoming section of this paper limiting the edge detection, connection, and discrimination region. Figure X shows the result of subtracting the median filtered output from the original grayscale approximation.

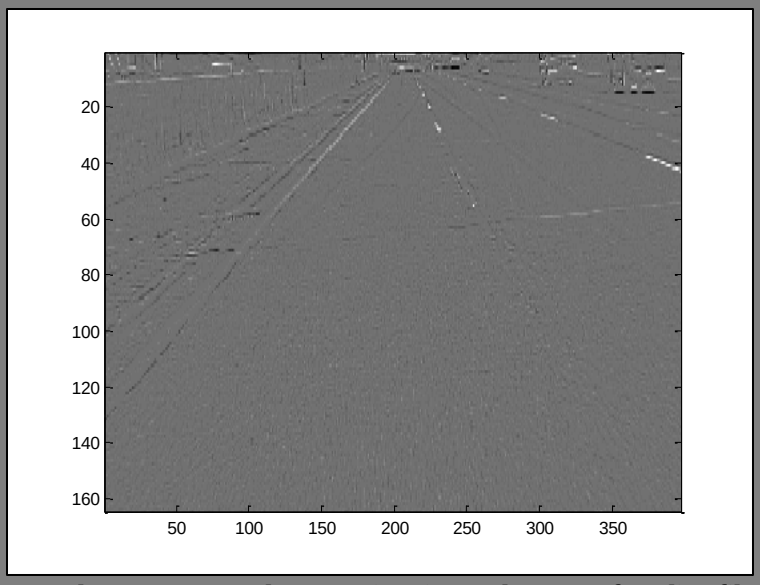


Figure X: Difference between grayscale approximation and output of median filter

Note that the larger positive differences between the grayscale approximation and the median filtered version appear 'lighter' while the larger negative differences between the two appear 'darker'. As can be seen from figure X, there is negligible difference between the actual grayscale approximation and the output of the median filter closer in the field of view while a noticeable difference is observable further into the field of view and off to the right side of the image. This has to do with the size of the block chosen for the median filter (in this case, a 3x3 block) in relation to the size and orientation of the edges further into the field of view.

It turns out that Gaussian filters used to filter noise will also filter (and smooth) non-noise characteristics pertinent to the objects within the image. Similarly, low pass and high pass filters based on linear shift invariant properties will not be able to attenuate only the noise in an image with additive noise and thus not improve the signal to noise ratios. Window methods may produce spectral distortions during the windowing process. The Matlab code portion for potentially implementing several of these filters is shown in figure X.

```
function [out_image] = sample_filt(choice_operator,image_data,opt_block_size)
% SAMPLE_FILT is a function which will perform various types of filtering on a grayscale approximated image
%=========================================================================
%    INPUT LIST (in order that they apprear as parameter list)
%=========================================================================
%   1) CHOICE_OPERATOR = used to select the particular noise reduction or median filter to apply
%   2) IMAGE_DATA = contains image data in grayscale format
%   3) OPT_BLOCK_SIZE = contains the size of the block mask for the particular filter operation
%=========================================================================
%   OUTPUT LIST (in order that they appear as parameter list)
%=========================================================================
%   1) OUT_IMAGE = contains filtered grayscale image data
%=========================================================================
min_pixel_value = 0;    max_pixel_value = 255;        rows = []; cols = [];  out_image = zeros([rows,cols]); weight = 0;
 [rows, cols] = size(image_data);

if ~isempty(findstr('nr1',choice_operator)) || ~isempty(findstr('nr2',choice_operator)) ||
~isempty(findstr('nr3',choice_operator))
   if strcmp(choice_operator,'nr1')
      op0 = [1 1 1
            1 1 1
            1 1 1];
      weight = 1/sum(sum(op0));
   elseif strcmp(choice_operator,'nr2')
      op0 = [1 2 1
            2 4 2
            1 2 1];
      weight = 1/sum(sum(op0));
   elseif strcmp(choice_operator,'nr3')
      op0 = [1 2 3 2 1
            2 4 6 4 2
            3 6 9 6 3
            2 4 6 4 2
            1 2 3 2 1];
      weight = 1/sum(sum(op0));
   end;

   row = []; col = []; upper_limit_row = []; upper_limit_col = []; row_offset = []; col_offset = [];
   [row, col] = size(op0);
   upper_limit_row = (row - 1);          upper_limit_col = (col - 1);
   row_offset = floor(row/2);            col_offset = floor(col/2);
   out_image = (weight .* conv2(image_data,op0));

elseif ~isempty(findstr('median',choice_operator))
   block_size = [];    block_size = opt_block_size;
   row = []; col = []; upper_limit_row = []; upper_limit_col = []; row_offset = []; col_offset = [];
   row = block_size;          col = block_size;
   upper_limit_row = row - 1;            upper_limit_col = col - 1;
   row_offset = floor(row/2);            col_offset = floor(col/2);
   for x = 1:rows-upper_limit_row,
      for y = 1:cols-upper_limit_col,
         out_image(x+row_offset,y+col_offset) = median(median(image_data(x:x+(row-1),y:y+(col-1))));
      end;
   end;
end;
```

Figure X: Code listing portion for function sample_filt.m

5.3 One dimensional and two dimensional convolution

As can be seen from the code segment of figure X, the function for performing filtering with different masks and different weights will generally perform a two-dimensional convolution operation with the mask over the entire image, storing the results in the appropriate location for the newly filtered image. In one dimension, the formula for performing the convolution sum can be represented as

$$y(n) = \sum_{k=-\infty}^{k=\infty} x(k)h(n-k)\,, \qquad \textbf{(X)}$$

where x(k) is the input signal at instant k and h(n-k) is the response of the system to the unit sample sequence δ(n) at a particular instant. The convolution sum may be derived from the superposition property for linear time invariant systems with input signal of the form

$$x(n) = \sum_{k=-\infty}^{\infty} x(k)\delta(n-k)\,, \qquad \textbf{(X)}$$

where the right hand side represents the sum of weighted impulses. To perform the convolution sum for one-dimensional signals, the four steps which need to be followed are:

1. Fold h(k) about k = 0;
2. Shift result of folding by n to right or left depending on whether n is positive or negative;
3. Multiply x(k) by h(n-k);
4. Sum the values resulting from the product sequence to obtain the value at the particular instant of time.

In a similar fashion, two dimensional signals may be convolved to produce the convolution sum according to

$$y(n1, n2) = \sum_{k1=-\infty}^{k1=\infty} \sum_{k2=-\infty}^{k2=\infty} x(k1, k2)h(n1-k1, n2-k2). \qquad \textbf{(X)}$$

Thus, for signals of two dimensions, the sequence of steps for performing two dimensional convolution includes the following three steps:

1. Move the adjusted mask to the point over the image where the convolution sum is to be performed;
2. Multiply each of the points in the mask by the corresponding points in the image;
3. Sum the values resulting from the product sequence to obtain the value at the particular spatial location.

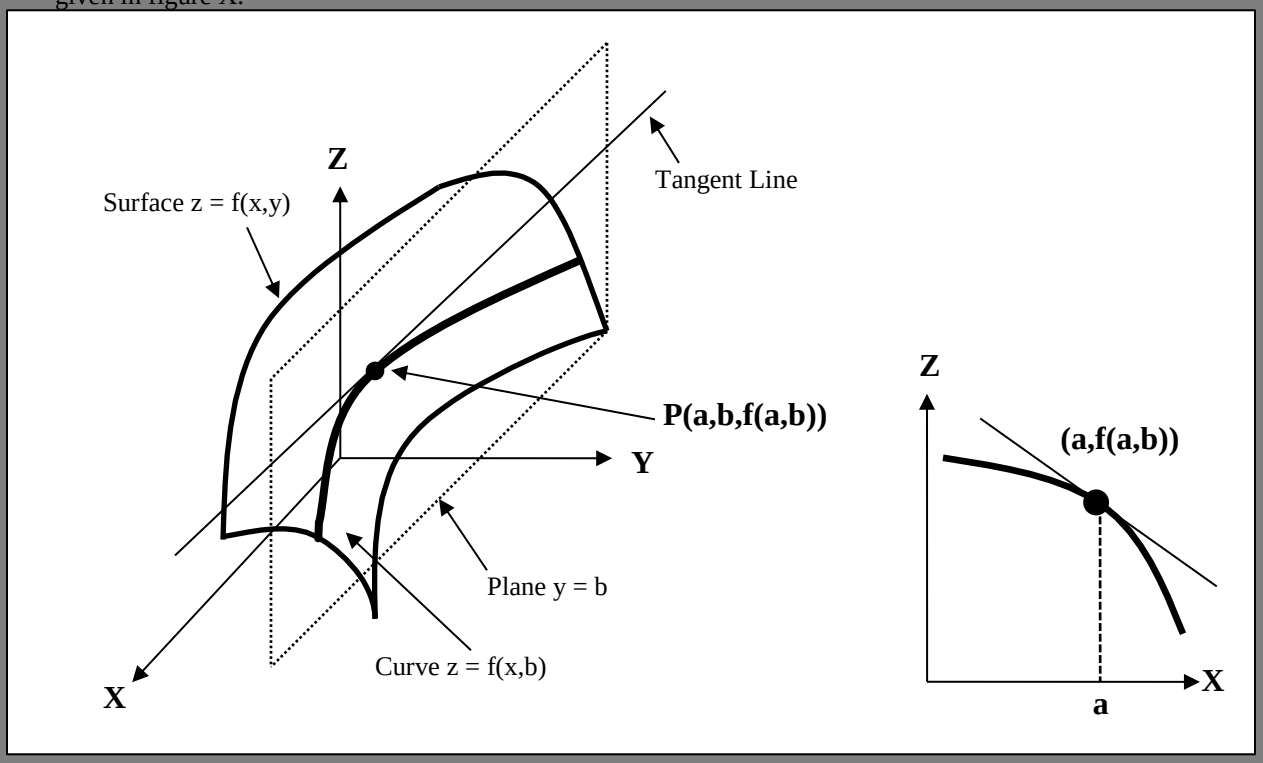
Image pre-processing may play an important role in many aspects of image processing. The edge operators used as a precursor to segmentation are in many ways ‘selective’ in the features they will isolate and are the topic of the next portion of this paper.

6. EDGE AND LINE OPERATORS AND EDGE DETECTION

6.1 Directional derivative and gradient

The images of the road scenes represent scalar functions. For each (row,column) pair within the image, there is a real number (scalar) associated with that position. One of the goals of the edge operators is to assist in isolating specifically desired characteristics. For example, the positive sloped lane markers being sought within an image may occur as edges ranging from approximately thirty degrees to just under sixty degrees when the vehicle is located in approximately the center of the lane and oriented to maintain position in the lane center with respect to the lane boundary. However, if a vehicle is transitioning within its lane, between adjacent lanes, etc., the positive sloped lane marker patterns being sought may be found at angles substantially outside the thirty to sixty degree angle range. The masks associated with specific directions of the edges being sought may be called compass masks because they can be used to isolate edges ranging from one to three hundred sixty degrees.

There exists a mathematical relationship between scalar fields (without direction) and vector fields (with direction) using the gradient function. For a function f(x,y), if the function is differentiable, the rate of change of f in the direction of the two coordinate axes is given by the partial derivatives. These partial derivatives are the slopes of the tangent lines to certain curves on the surface z = f(x,y). The geometric interpretation of the partial derivative $f_x(a,b)$ is given in figure X.



(a) (b)

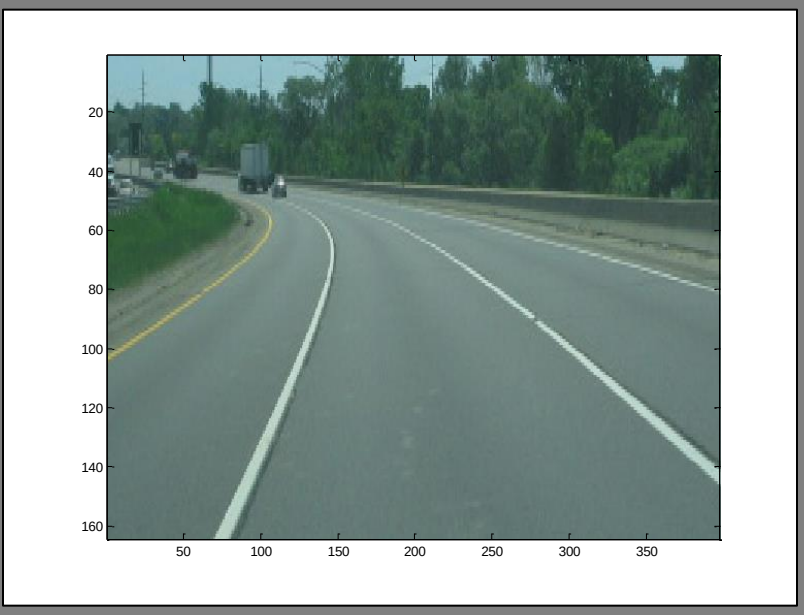
Figure X: (a) Geometric interpretation of directional derivative in the x-direction; (b) projection in the XZ plane

Then from figure X (b), it is known that

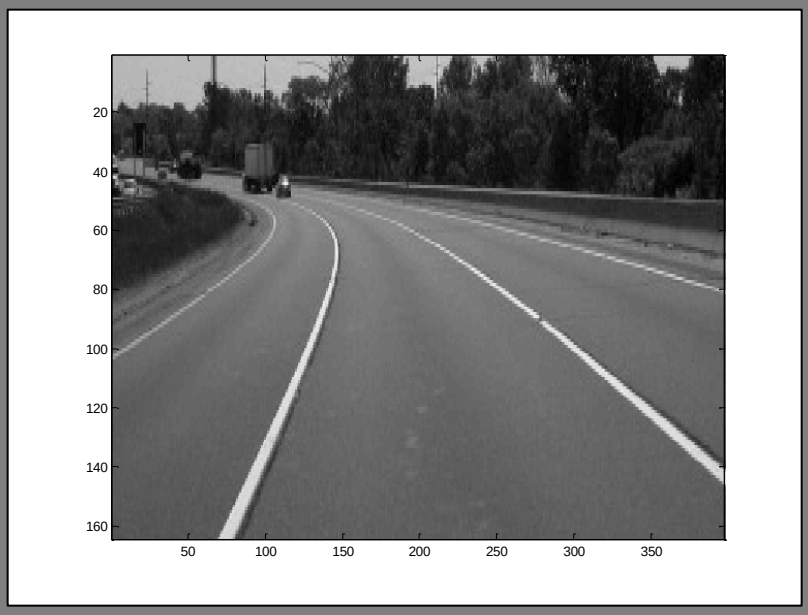
$$f_x(a,b) = \lim_{h \to 0}[f(a+h,b) - f(a,b)]/h. \qquad \textbf{(X)}$$

Similarly, the rate of change of f from any point P in any fixed direction is given by the directional derivative. The gradient of f at point P (if it is non-zero at P) indicates the direction of maximum increase. Thus, two methods have been

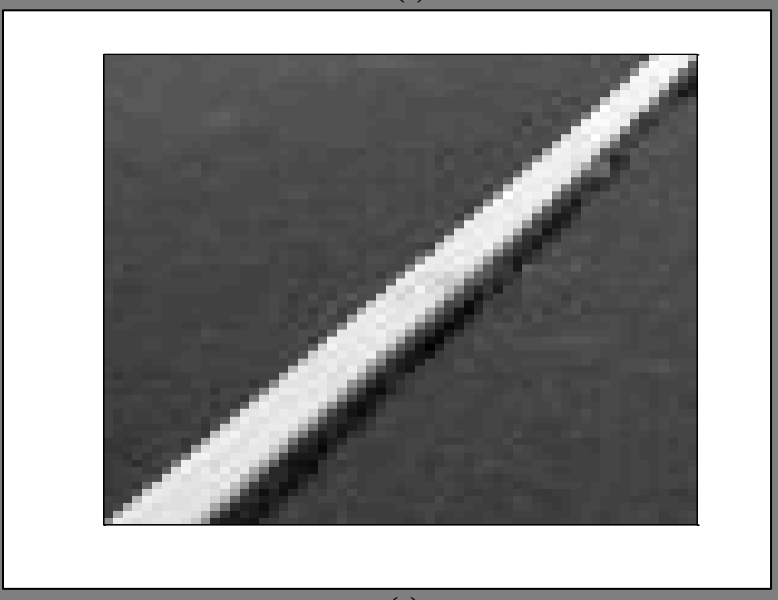
established for identifying certain characteristics within an image that may be applied in the Matlab environment. The first involves using edge operators (compass masks) and the second involves using the gradient function which provides magnitude and direction of maximal increase. A road scene image and its surface representation are shown in figure X.



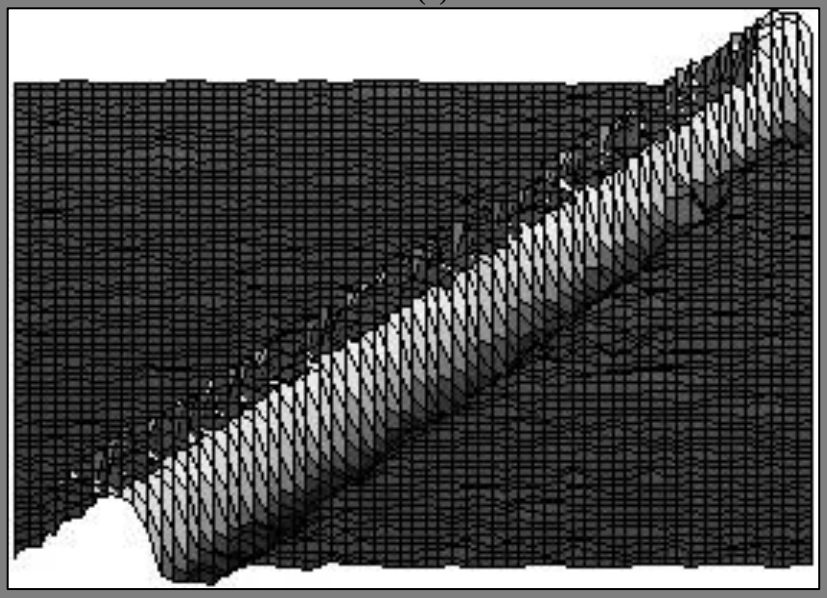
(a)



(b)



(c)



(d)

Figure X: (a) Cropped image showing various normal markers; (b) its grayscale approximation; (c) grayscale cropped from (100:164,70:130); (d) its rotated surface plot

Figure X (c) and (d) are cropped portions of the image shown in (b). The rotated surface plot of figure X (d) shows more illustratively the relevance in the geometric interpretation for the gradient of a scalar field during the analysis of longitudinal lane markers.

The gradient for a differentiable scalar function of two variables f(x,y) having continuous partial derivatives is the vector field defined by

$$\text{grad } f = \frac{\partial f}{\partial x}\underline{i} + \frac{\partial f}{\partial y}\underline{j}. \qquad \textbf{(X)}$$

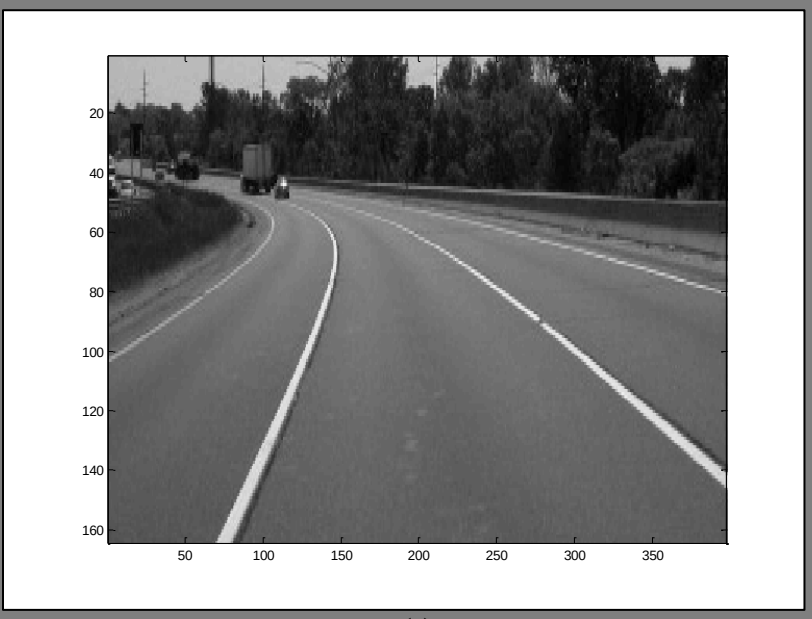
Where a large gradient magnitude is present, an edge (or portion of an edge) in the image is likely to be found. This transitional point may then be a likely candidate for a border between related objects within the image. Thus, the angle for the gradient gives the direction of maximum increase and the magnitude yields the rate of increase in that (maximimal) direction. Image data is specified in the Matlab environment in terms of (horizontal) rows and (vertical) columns, so these two 'perpendicular directions' provide useful gradient information in road scene image analysis. They may also be combined to produce gradient magnitude and direction information. If the column gradient data is denoted col(i,j) and the row gradient data as row(i,j), then the gradient magnitude may be approximated as

$$|\text{grad } f(i,j)| = [\text{col}(i,j)^2 + \text{row}(i,j)^2]^{1/2}. \qquad \textbf{(X)}$$

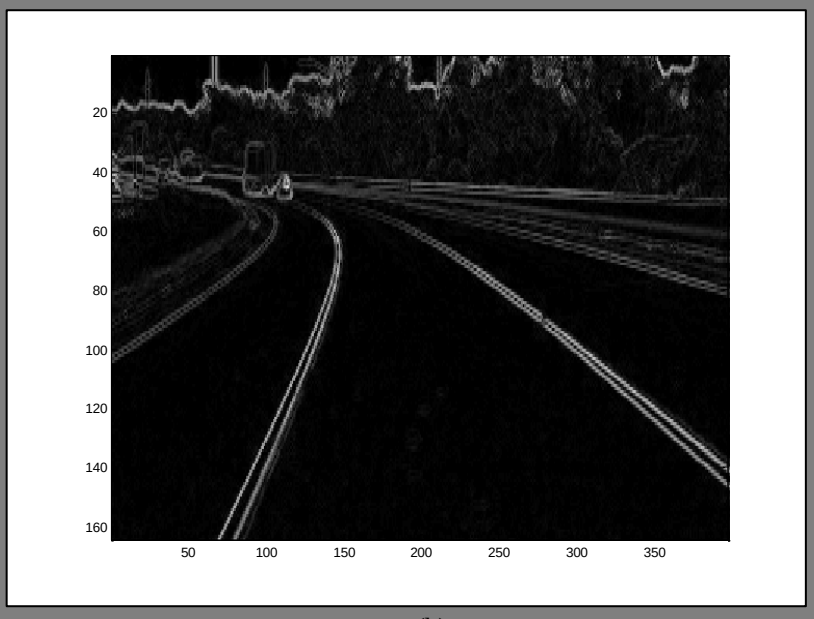
Similarly, the gradient angle may be approximated as

$$\Theta(i,j) = \tan^{-1}[\text{row}(i,j) / \text{col}(i,j)]. \qquad \textbf{(X)}$$

A grayscale approximated road scene image with the gradient magnitude calculated from Matlab's gradient function is shown in figure X.



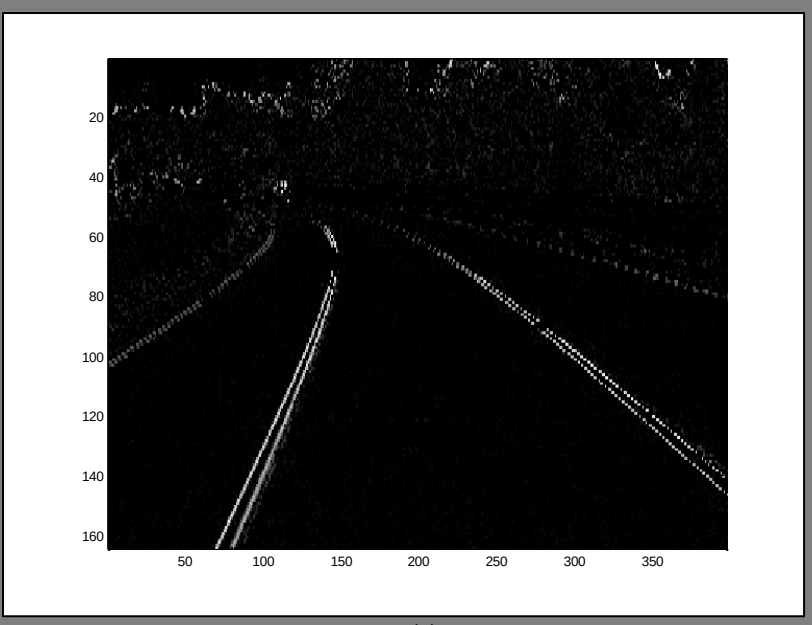
(a)



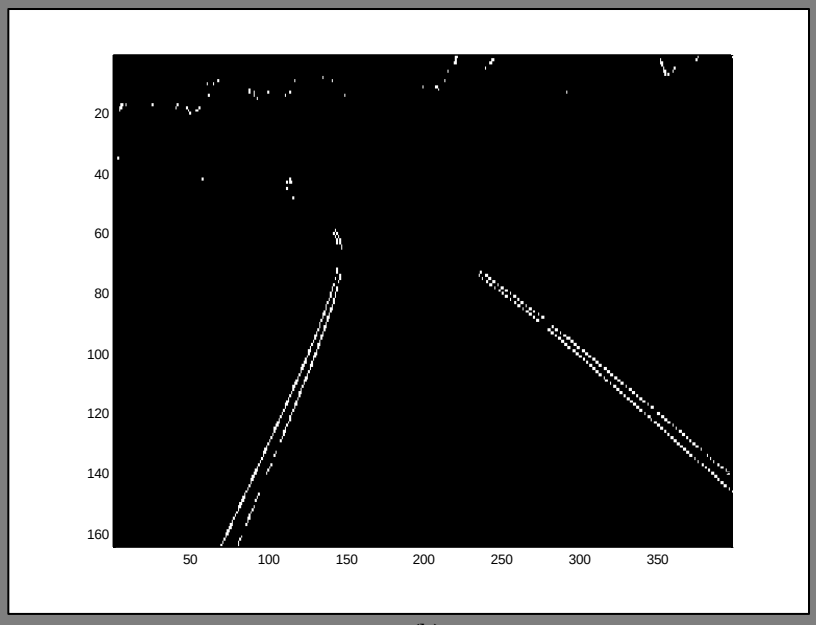
(b)

Figure X: (a) Grayscale image; (b) magnitude output from gradient function on same image

As may be seen from the figure X (b), Matlab's gradient function may be used to approximate the various gradient magnitude levels within a road scene image. These locations of largest gradient magnitude are indicated by the 'lightest' portions within the image. Figure X shows the gradient magnitude output for the gradient directions $30° < \Theta < 60°$ and $-60° < \Theta < -30°$ along with its binary thresholded counterpart.



(a)



(b)

Figure X: (a) Gradient magnitude output for gradient directions $30^o < \Theta < 60^o$ and $-60^o < \Theta < -30^o$; (b) binary thresholded gradient magnitude for gradient direction with threshold set at 50

It can be seen from figure X (b) that the thresholding of gradient magnitude information restricted to certain angles has helped to isolate certain lane marker edges.

6.2 Prewitt and Sobel edge operators and vertical edge and line operators

An example of one type of Prewitt edge operator is given in figure X (a) while one type of Sobel edge operator is given in figure X (b).

$$\underset{r1}{\begin{bmatrix} 1 & 1 & 1 \\ 0 & 0 & 0 \\ -1 & -1 & -1 \end{bmatrix}} \qquad \underset{r2}{\begin{bmatrix} -1 & -1 & -1 \\ 0 & 0 & 0 \\ 1 & 1 & 1 \end{bmatrix}}$$

(a)

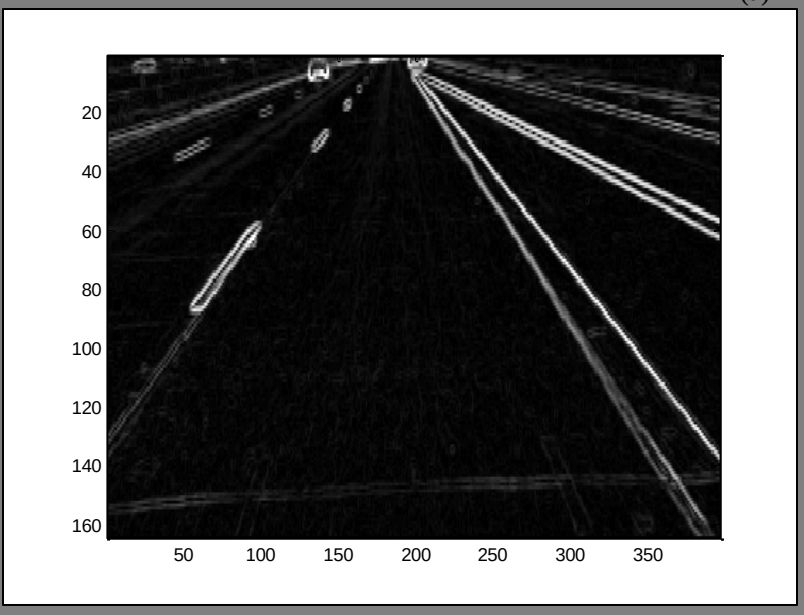
$$\underset{r1}{\begin{bmatrix} -1 & -2 & -1 \\ 0 & 0 & 0 \\ 1 & 2 & 1 \end{bmatrix}} \qquad \underset{r2}{\begin{bmatrix} 1 & 2 & 1 \\ 0 & 0 & 0 \\ -1 & -2 & -1 \end{bmatrix}}$$

(b)

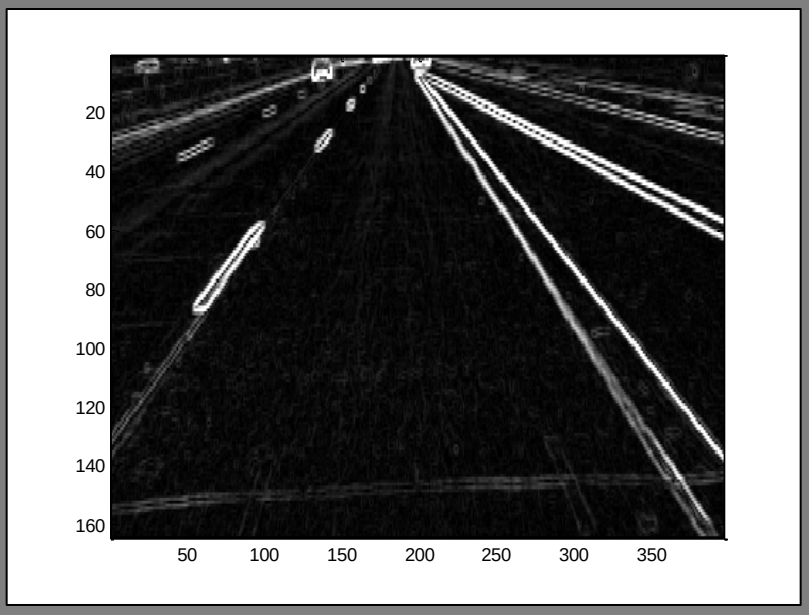
Figure X: (a) Prewitt edge operators; (b) Sobel edge operators

The Sobel edge operator is different than the Prewitt edge operator in that it weights the central pixels in the finite differences twice that of the pixels to each side. Note that the edge operators of figures X (a) and (b) may also be applied to vertical and diagonal edges by simply manipulating the positions of the individual elements of the respective masks. Figure X (a) shows a different road scene image with (b) and (c) showing the output of the Prewitt and Sobel compass masks.

(a)



(b)



(c)

Figure X: (a) Original RGB cropped image; (b) output from Prewitt compass operator; (c) output from Sobel compass operator

The term compass mask indicates that the mask will detect different edges as the mask is rotated about it center point either clockwise or counterclockwise. Thus, if rotating the leftmost Prewitt mask of figure X (a) counterclockwise one position about its center value, the mask would then detect positive sloped diagonal edge transitions from light to dark. Rotating that same leftmost Prewitt mask again counterclockwise one position about its center value, the mask would then detect vertical edge transitions from light to dark.

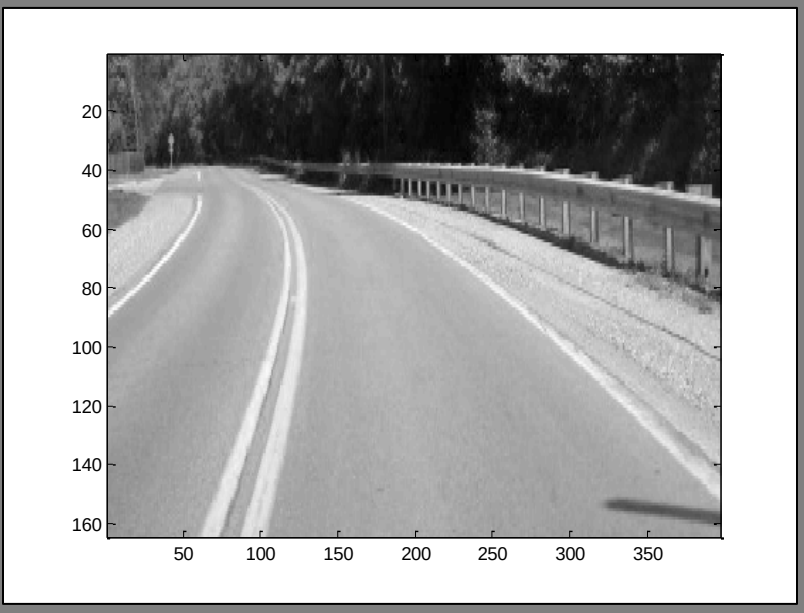
Line detectors behave in a fashion very similar to edge detectors. The line detectors look at a line as an elongated rectangle with different grayscale values on each side of the elongated rectangle. For example, a light vertical line against a dark background may contain a dark to light transition to the left of the line *and* a light to dark transition to the right of the line. Line detectors may be chosen to detect various line widths and lines at various angles. A potential example of a vertical line detector for a line 1 pixel wide is shown in figure X (a) while (b) shows an example of a vertical edge detector which will detect both dark to light and light to dark vertical edge transitions. Figure X (c) shows an approximated grayscale road scene image while (d)-(j) show various types of output from the vertical line and vertical edge detectors when applied to a road scene image.

$$\begin{bmatrix} -1 & 2 & -1 \\ -1 & 2 & -1 \\ -1 & 2 & -1 \\ -1 & 2 & -1 \\ -1 & 2 & -1 \end{bmatrix}$$

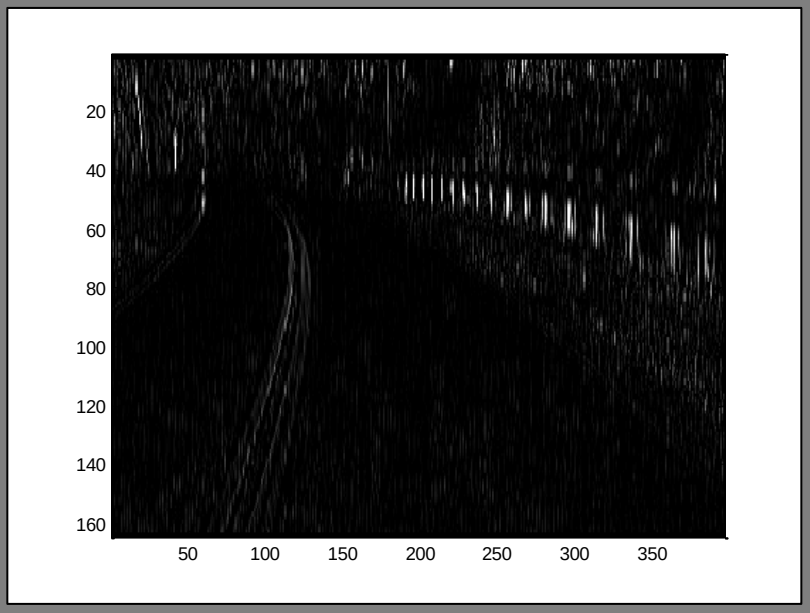
(a)

$$\begin{bmatrix} -1 & 0 & 1 \\ -1 & 0 & 1 \\ -1 & 0 & 1 \\ -1 & 0 & 1 \\ -1 & 0 & 1 \end{bmatrix} \begin{bmatrix} 1 & 0 & -1 \\ 1 & 0 & -1 \\ 1 & 0 & -1 \\ 1 & 0 & -1 \\ 1 & 0 & -1 \end{bmatrix}$$

(b)

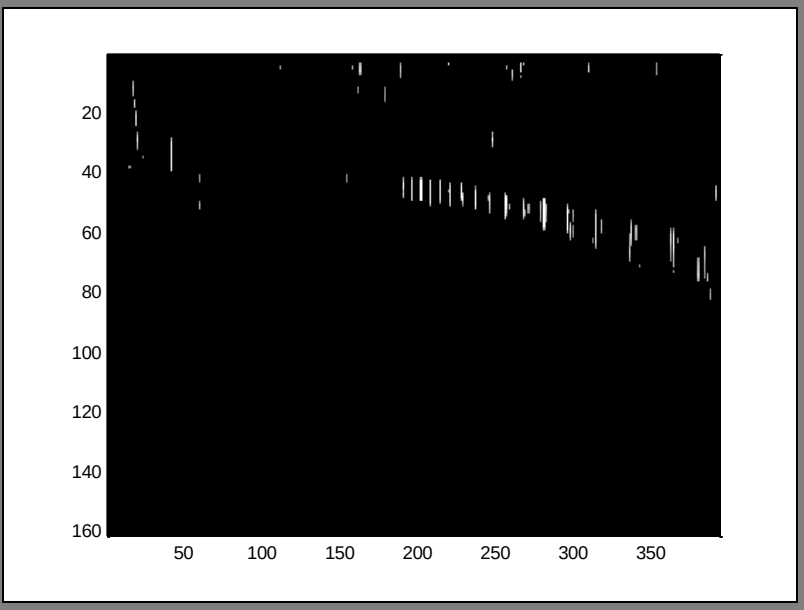


(c)

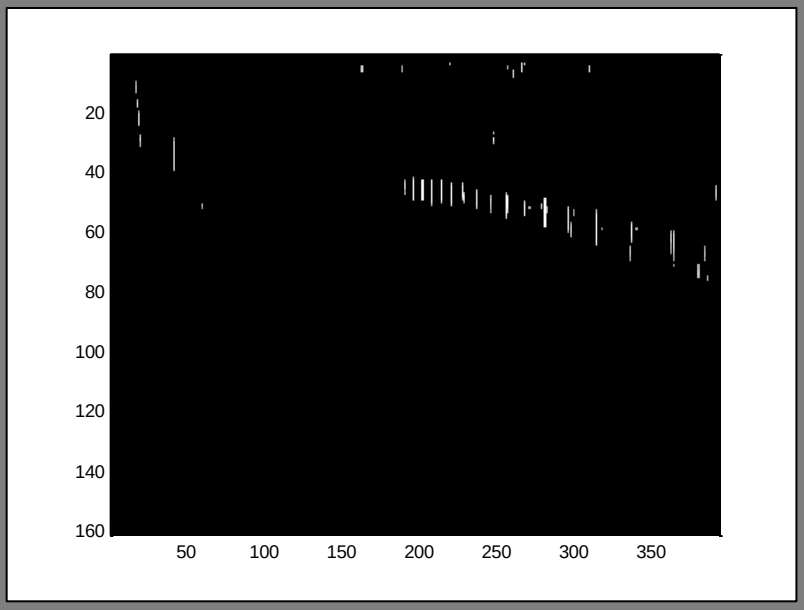


(d)

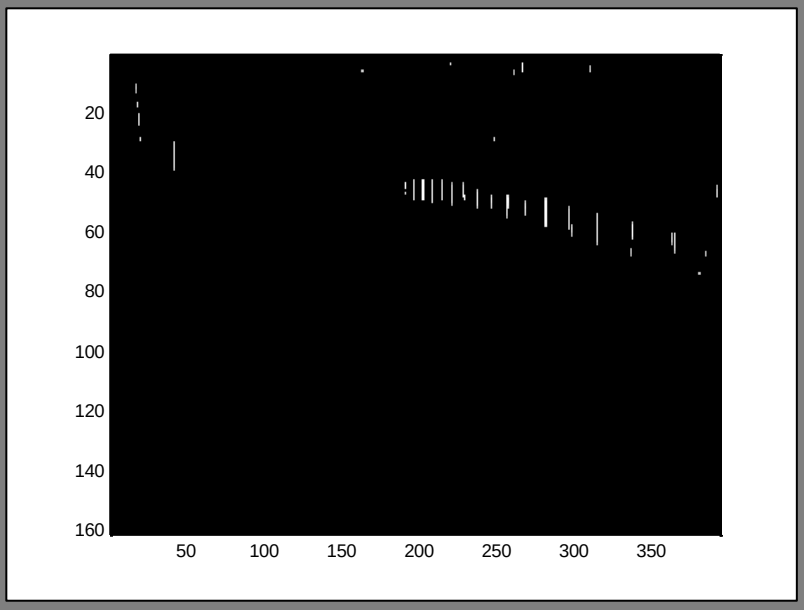
Figure X. (a) Example of 1 pixel wide vertical line detector; (b) example of vertical edge detector pair for detecting dark to light and light to dark transitions; (c) grayscale approximated road scene image; (d) output of vertical line detector when applied to road scene image



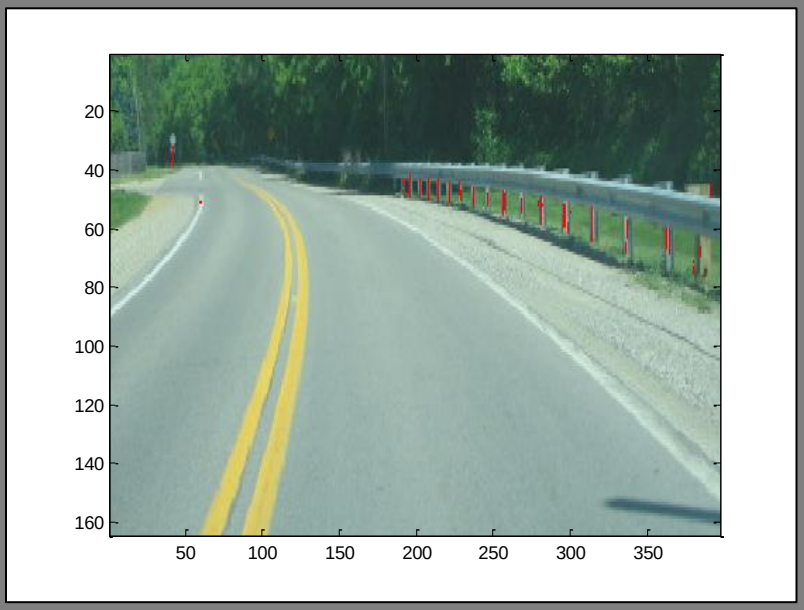
(e)



(f)

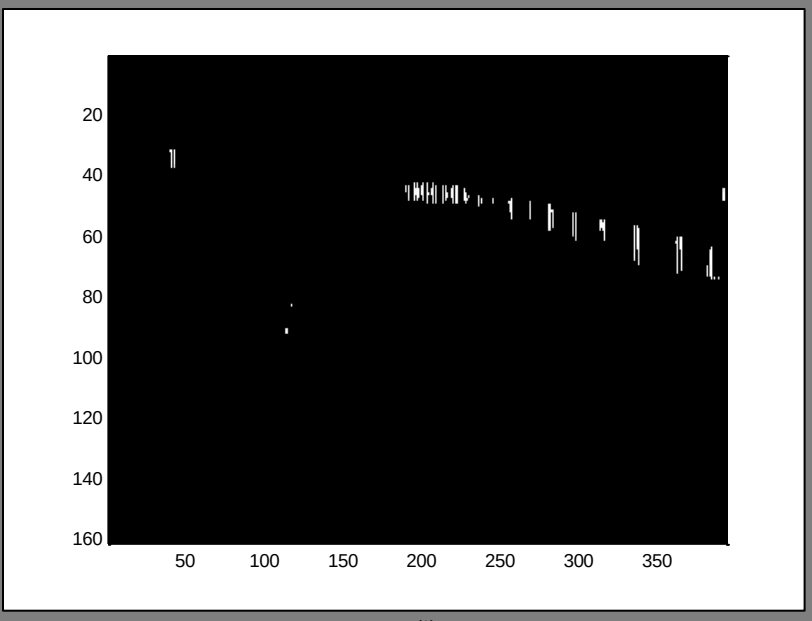


(g)

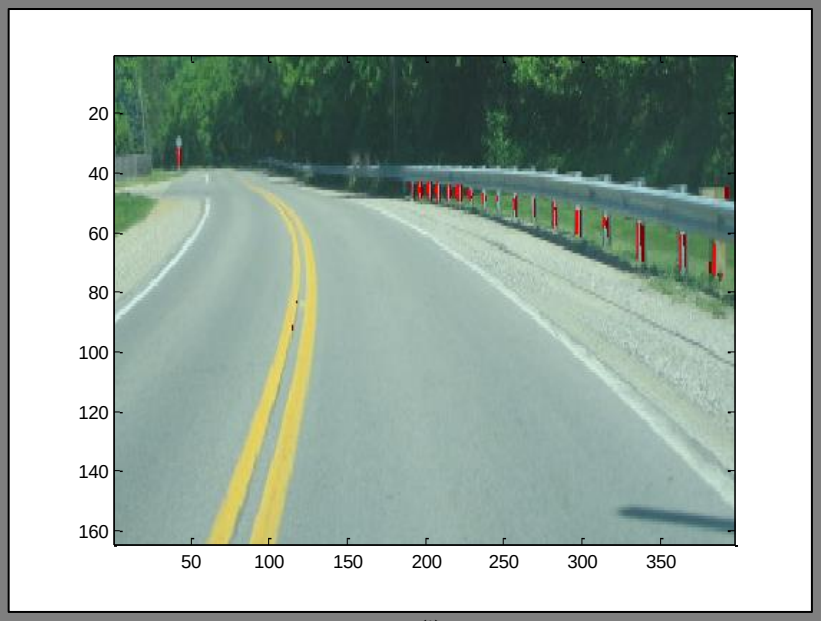


(h)

Figure X (continued): (e) Binary thresholded output of vertical line detector (threshold = 150); (f) binary thresholded output of vertical line detector (threshold = 180); (g) binary thresholded output of vertical line detector (threshold = 210); (h) thresholded vertical line detector superimposed (in red) on RGB image



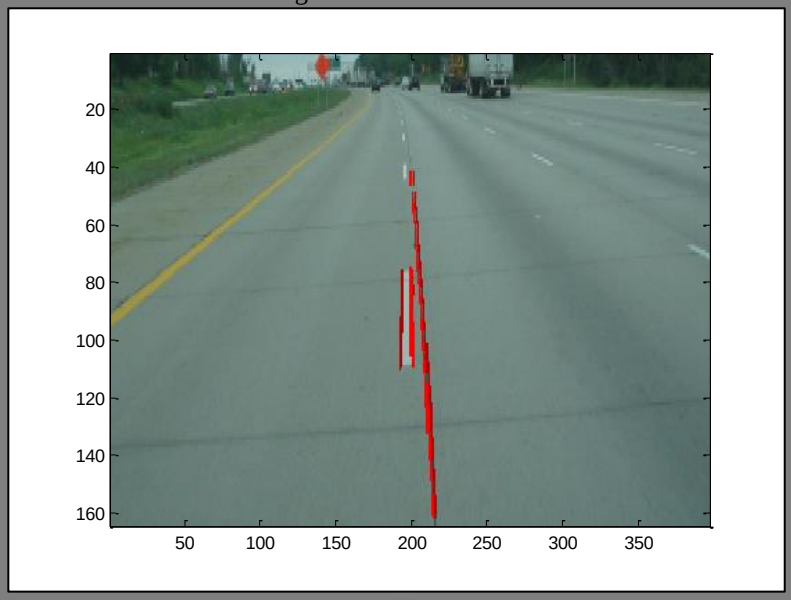
(i)



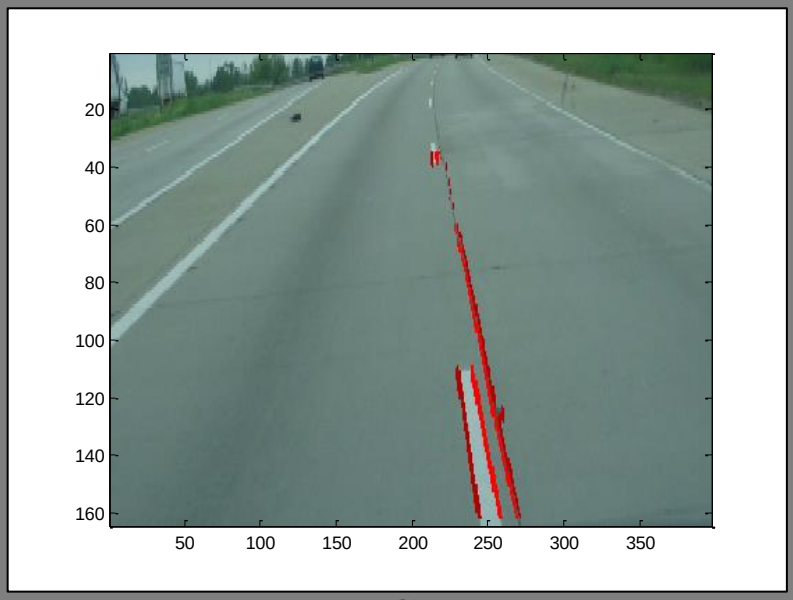
(j)

Figure X (continued): (i) Thresholded output of vertical edge detector (threshold = 235); (j) thresholded dark to light and light to dark vertical edge detector superimposed in light red and dark red, respectively, on RGB image

It should be noted that the use of the vertical edge detector may be used to detect both dark to light vertical edge transitions as well as light to dark vertical edge transitions. Depending on reasons including (but not limited to) the levels set for the thresholds, the mask characteristics, and the image aspects we are attempting to isolate, these edge detectors may also be used to detect edges varying slightly from vertical. Consider the two road scene images of figure X in which the vertical edge detector may be used to isolate both the dark to light edge transitions and light to dark edge transitions for the longitudinal lane marker and the road seam.



(a)



(b)

Figure X (a) RGB cropped road scene image with dark to light and light to dark 'nearly' vertical edge transitions highlighted in darker red and lighter red colors, respectively (b) RGB cropped road scene image with dark to light and light to dark 'nearly' vertical edge transitions highlighted in darker red and lighter red colors, respectively

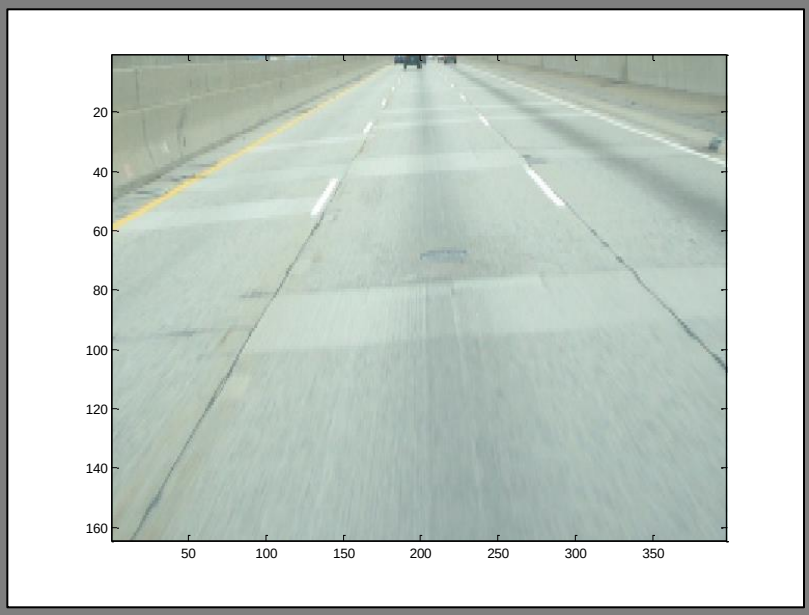
As can be seen from figure X, the vertical line detector and vertical edge detectors will indeed detect vertical or ‘nearly’ vertical transitions within a road scene image. The size and orientation (amongst other characteristics) of the edge or line being detected needs to be considered when choosing the template mask and the thresholds being compared to as this will affect what is ‘found’ and what is ‘overlooked’.

One of the issues which need to be considered when performing image processing is what precisely to do near the outer border of the image during the edge detection process. Specifically, what values should be used during the correlation process when (N-1)/2 rows and (M-1)/2 columns (for an NxM edge detector, both N and M greater than or equal to 3, N and M both odd) are required outside the outermost image border to perform the correlation. There are a number of ways which have been proposed including padding outside the image border with the outermost image pixel value or with zero. Another potential method involves considering the relationship between the Fourier transform of the image and the transfer function of the mask and then performing a periodic extension [2]. Another potential method is to simply leave the outermost image values as they are and perform the correlation up to the (N-1)/2 row and (M-1)/2 column locations. Thus, for a 7x7 edge detector, no edge components would be evaluated at the outermost three rows or columns of pixels. This scenario might create issues if only a small portion of a longitudinal lane marker was located on one of the outer image border locations. If the end of a marker did occur near the outer border of the image, then the shorter the existing marker edge, the more likely the edge would be discriminated as a non-marker (due to minimum length being one of the primitive characteristics used in the structural pattern classification). However, if only a small portion a lane marker did end near the outer border of an image (nearest to the vehicle’s front end), then it would be very likely that there would be another lane marker somewhere further in the FOV which would provide more reliable lane marker characteristics for processing. It should also be noted though that a small portion of a marker edge on the outer border of the image that might be discriminated was very likely a ‘larger edge portion further from the image border’ at the previous instant with an increased likelihood for detection. When considering whether to use edge operators or line operators or other methods to facilitate likely lane marker border classification, a number of other considerations should be made. Consider the images of figure X containing a number of potential scenarios for road scene markers that might need to be dealt with.

(a)

(b)

(c)



(d)

Figure X: (a) Normal white lane marker on 'darker colored' road surface; (b) broken white marker on 'lighter colored' road surface; (c) wide and normal white markers at exit ramp in partial shade on degraded road surface; (d) broken white markers on 'lighter multi-colored' road surface in bright sunlight

(e)

(f)

Figure X (continued): (e) Combination of normal and broken yellow and broken white markers; (f) faded broken yellow and normal white markers

It is apparent that the dimensions of a normal lane marker may be significantly different than those of a broken marker or another different normal marker at the same distance in the FOV. This may be caused by a number of factors including (but not limited to) the type of marker, the condition of the marker, the perspective projection, illumination conditions, environmental conditions, any factors affecting marker occlusion, the road geometry, the position of the vehicle with respect to the lane, image noise or distortion, or any combination of the possible causes. The color characteristics may be looked to as a common factor for the longitudinal markers in many of the images above. However, color appearance is dependent on a number of complex relationships between characteristics including (but not limited to) spectral, spatial, temporal, optical, and physical. To further complicate matters, illumination source(s) and condition(s) may vary dramatically within a single road scene image which may affect color perception. Many of these same factors will also affect the perceived luminance contrast (or difference in brightness levels) between a marker and the substrate. Thus, a white or yellow longitudinal lane marker may appear at various brightness and colorfulness levels if it has faded, is dirty, is viewed with limited illumination, is applied improperly, is affected by environmental conditions, or is somehow otherwise degraded or occluded. As will also be seen, a white longitudinal marker may appear yellow if illuminated as such. However, changes to a lane markers 'intended' color characteristics will typically also be reflected in a change to the color characteristics of the substrate.

If using edge detectors, some of the challenges due to dimensional variation inherent in the problem space might possibly be overcome as the sources of variation may now be incorporated into structural pattern attributes incorporating statistical compensation. However, edge detectors may suffer from the fact that the range of thresholds which apply to the potential marker-substrate possibilities may not be conducive to the subsequent discrimination strategy. Thus, the type of edge operator chosen will play a significant role in longitudinal lane marker edge detection.

6.3 The generalized diagonal operators

The primary mask pair used in detecting the positive sloped dark to light transitions is shown in figure X while the primary mask pair used in detecting the positive sloped light to dark edge transitions is shown in figure X.

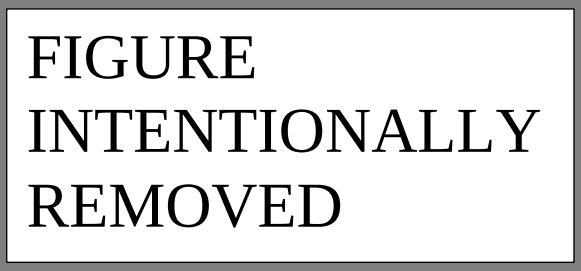


Figure X: Generalized mask pair for positive slope dark to light transitions of slope approximately one

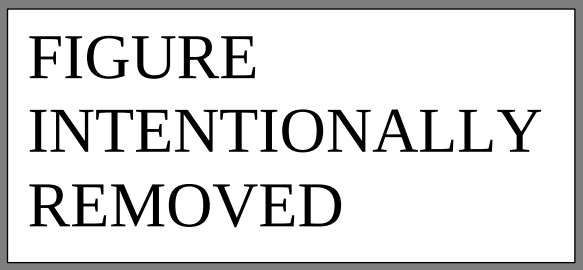


Figure X: Generalized mask pair for positive slope light to dark transitions of slope approximately one

The edge detectors shown in figure X attempt to find edges of slope approximately one within an image. This is apparent from the relationships between the positive and negative values within the mask (rise divided by run equal to one).

6.4 Edge detection for longitudinal lane markers of roads of various curvatures and the effects of threshold variation

Another factor which has yet to be elaborated on is the effect of the radius of curvature of the road (if any) on the edge detection process. Consider the following sample images taken in curves of different radii and the potential effects on the edge detection process.

(a)

(b)

(c)

(d)

Figure X: (a) Approximately straight road; (b) larger radius road curvature; (c) moderate radius road curvature; (d) smaller radius road curvature

There are a number of preliminary conclusions which may be drawn from the images of figure X. The first is that the longitudinal markers on the left and right side of the lane will typically have different percentages of prospective edges detected due to factors including (but not limited to) the varying vehicular position within the lane. Second, adjacent markers (normal and normal pairs in a center line, normal and broken pairs in a two-way left turn lane, etc.) will typically have different percentages of prospective edges detected due to their position relative to each other. Third,

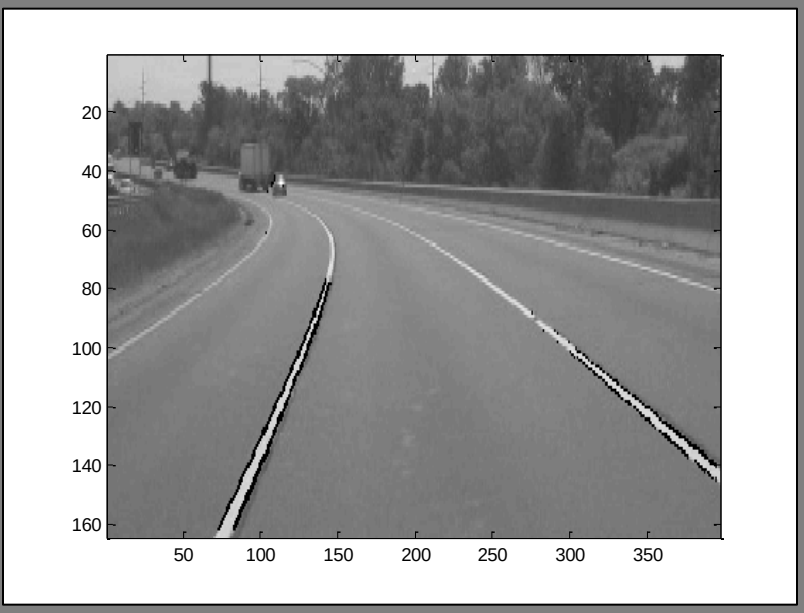
yellow and white colored lane markers will typically produce different luminance contrast due to their different brightness levels. Fourth, the longitudinal lane marker edges in the image with transitions which are more similar to that of the generalized edge detector will be more likely to be detected than marker edges which are less similar. Fifth, factors which detrimentally affect lane marker edge distinctness (lack of illumination, degradation, occlusion, etc.) will typically have a detrimental effect on longitudinal lane marker edge detection. Sixth, the lower the edge detection thresholds are set, the more effective the edge detector will be at detecting edges which are dissimilar to that of the generalized edge detector. Roads with a smaller curve radius tend to contain prospective lane marker edges with decreased levels of similarity to the generalized edge detector (further into the FOV when positioned near the center of the lane). Thus, the detection of these prospective lane marker edges will typically require a reduction in threshold levels. A comparable threshold reduction may be made for near FOV prospective lane marker edges which differ significantly from the generalized edge detector. This idea is illustrated more clearly in figures X and X for two road scene images with curves of 'moderate' radius.

(a)

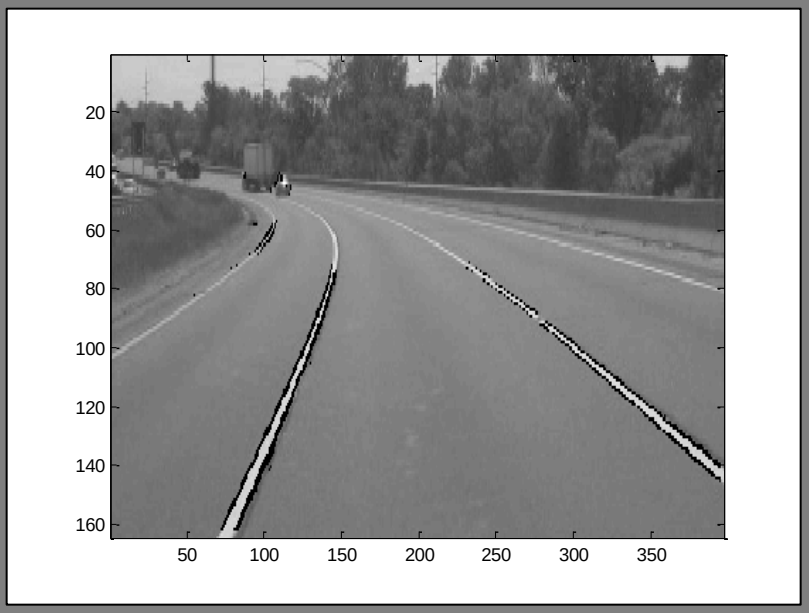
(b)

Figure X: (a) Road scene with moderate curve radius to left and (b) different scene with moderate curve radius to right

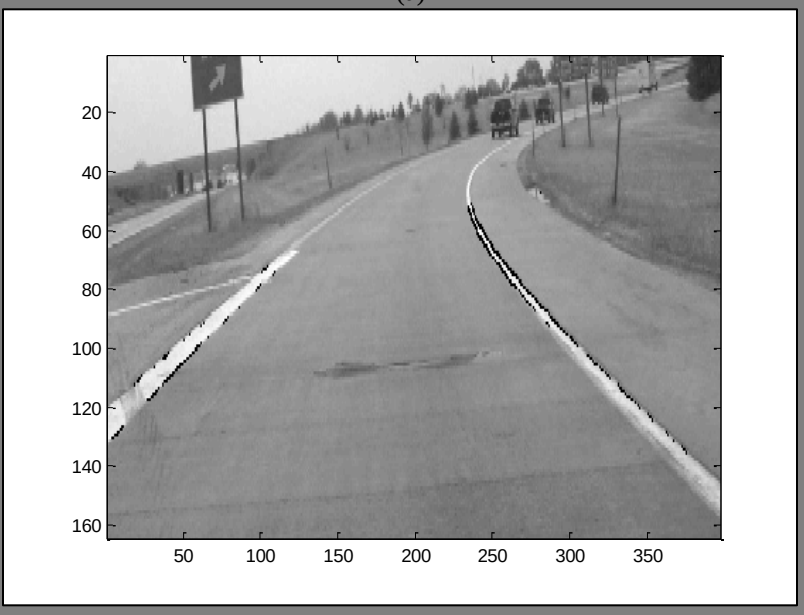
Figure X shows the results of the preliminary Matlab image processing software using two different threshold levels for edge detection in the road scene images with the resulting detected edges superimposed on grayscale approximations.



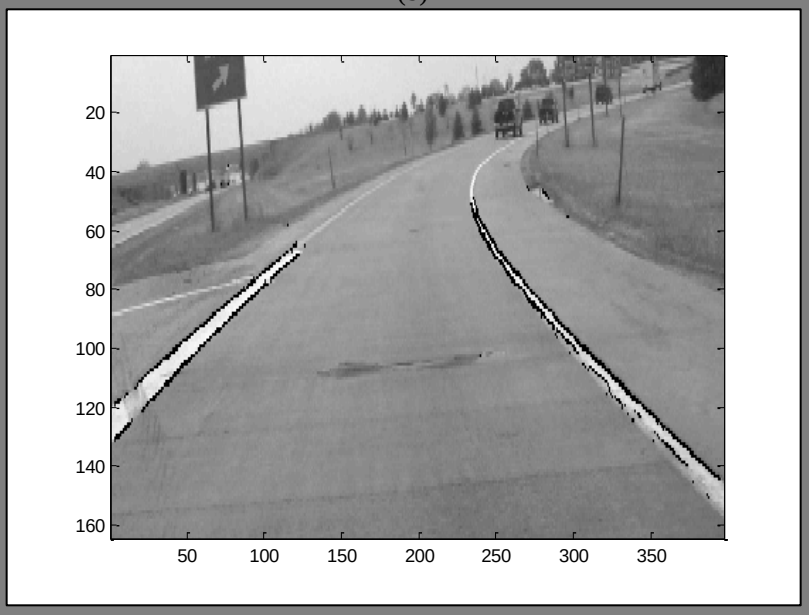
(a)



(b)



(c)



(d)

Figure X: Generalized edge detection process (with detected edges superimposed in black) (a) with threshold = 180; (b) with threshold = 70 for same road scene image; (c) with threshold = 180 for second road scene image; (d) with threshold = 70 for second road scene image

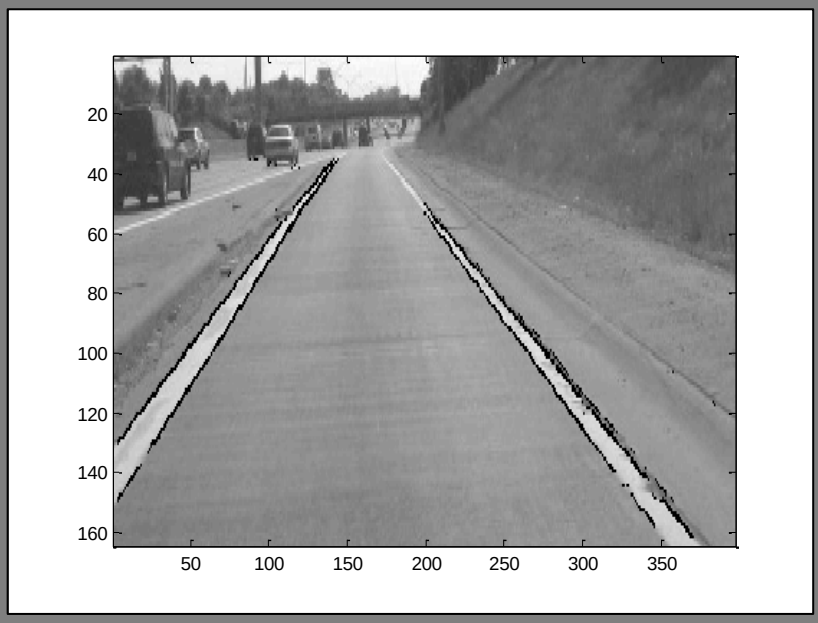
In the figure X, the different threshold levels have resulted in different portions of the lane marker edges having been detected. Note also that lowering the thresholds increases the likelihood that an unintended road scene image edge will be detected.

Thus, there are many considerations that may need to be made when considering edge detector choice(s) and threshold selection including (but not limited to):

1. How near in the FOV the lane marker edges are intended to be detected.
2. How far into the FOV the lane marker edges are intended to be detected.
3. The perspective projection of the image.
4. The 'typical' range of positions for a vehicle within a lane.
5. The 'typical' range of positions for a vehicle during and after departure from a lane.
6. The 'typical' range of positions for a vehicle entering a lane.
7. The relationship between the direction the vehicle is heading in and the delineation that the lane markers are providing.
8. The road geometry.
9. The relationships between 'typical' lane transitions and road geometry.
10. The types of lane markers being detected and variations amongst those types.
11. The locations of lane markers within an image and variations amongst those positions.
12. The need for adjacent lane marker detection.
13. Vehicle operating conditions and environmental conditions.
14. The relationships between edge detector choice(s), threshold variation, road geometry, etc., on intended and unintended edge detection.

Figure X shows how the edge detection process may detect edges further into the FOV for an entrance ramp that is approximately straight within the FOV.

(a)



(b)

Figure X: (a) RGB cropped road scene; (b) detected edges overlayed in black with threshold = 70

As may be seen from the figures above, the edge detector mask and thresholds (along with other important factors) used during the edge detection process all play a significant role in how efficiently and effectively edges are detected. A portion of the code for function detect_edge_gen_bkp.m is shown in the listing in figure X.

The function detect_edge_gen_bkp.m was written with a number of goals in mind. The first was to accept a number of different types of edge operators including (but not limited to) Prewitt, Sobel, and 'generalized' edge operators. The second was to process edge operators of different size masks. That is, a function was desired that could operate on a 3x3 mask as well as a 13x13 mask or a 25x25 mask. The third was that the function(s) would be easily modifiable to accept 'user-defined' edge operators.

```
% Programmer: Christopher Alan Warner
% DETECT_EDGE_GEN_BKP is a function that will serve to detect edges using
% various operators being applied to image_data.
% ================================================================
%     INPUT LIST (in order that they appear as parameter list):
% ================================================================
% 1) IMAGE_DATA = contains the original image data
% 3) CHOICE_OPERATOR = this operator will dictate which edge operators are used and how
% ================================================================
%     OUTPUT LIST (in order that they appear as parameter list):
% ================================================================
% 1) OUT_IMAGE_B4_DISCRIMINATION = is EDGE DETECTED IMAGE in grayscale
% ================================================================
%     OTHER CUSTOM FUNCTIONS USED
% ================================================================

function [out_image_b4_discrimination] = detect_edge_gen_bkp(choice_operator,image_data)

min_pixel_value = 0;
max_pixel_value = 255;

rows = []; cols = [];
[rows, cols] = size(image_data);                    % get size of image
col_grad = zeros([rows,cols]);
row_grad = zeros([rows,cols]);
out_image = zeros(rows,cols);
out_image_b4_discrimination = zeros(rows,cols);

total1 = zeros(rows,cols);  total2 = zeros(rows,cols);
min_total1 = zeros(rows,cols);  min_total2 = zeros(rows,cols);

min_threshold_edge_1 = 0; op0 = []; op1 = []; op2 = []; op3 = []; op4 = []; op5 = []; op6 = []; op7 = [];
generic_op_1 = [];  generic_op_2 = [];

row = []; col = []; upper_limit_row = []; upper_limit_col = []; row_offset = []; col_offset = [];

if ~isempty(findstr('Sobel',choice_operator)) || ~isempty(findstr('Prewitt',choice_operator))
            ||~isempty(findstr('Frei_chen',choice_operator))
   min_threshold_edge_1 = 55;

   if strcmp(choice_operator,'Sobel_compass')
      sobel_compass;                              % run Sobel compass script
      op0 = sm0; op1 = sm1; op2 = sm2; op3 = sm3; op4 = sm4; op5 = sm5; op6 = sm6; op7 = sm7;
   elseif strcmp(choice_operator,'Sobel_vert')
      sobel_vert;                                 % run Sobel vertical script
      op0 = sm0; op1 = sm1; op2 = sm2; op3 = sm3; op4 = sm4; op5 = sm5; op6 = sm6; op7 = sm7;
```

Figure X: Code listing portion for function detect_edge_gen_bkp.m

```
   elseif strcmp(choice_operator,'Sobel_horiz')
      sobel_horiz;                          % run Sobel horizontal script
      op0 = sm0; op1 = sm1; op2 = sm2; op3 = sm3; op4 = sm4; op5 = sm5; op6 = sm6; op7 = sm7;
   elseif strcmp(choice_operator,'Frei_chen_compass')
      frei_chen;                            % run Frei_chen_compass script
      op0 = fc0; op1 = fc1; op2 = fc2; op3 = fc3; op4 = fc4; op5 = fc5; op6 = fc6; op7 = fc7;
   elseif strcmp(choice_operator,'Frei_chen_vert')
      frei_chen_vert;                       % run frei_chen vertical script
      op0 = fc0; op1 = fc1; op2 = fc2; op3 = fc3; op4 = fc4; op5 = fc5; op6 = fc6; op7 = fc7;
   elseif strcmp(choice_operator,'Frei_chen_horiz')
      frei_chen_horiz;                      % run frei_chen horizontal script
      op0 = fc0; op1 = fc1; op2 = fc2; op3 = fc3; op4 = fc4; op5 = fc5; op6 = fc6; op7 = fc7;
   elseif strcmp(choice_operator,'Prewitt_compass')
      prewitt;                              % run prewitt script
      op0 = pw0; op1 = pw1; op2 = pw2; op3 = pw3; op4 = pw4; op5 = pw5; op6 = pw6; op7 = pw7;
   % Note many others still to add
   end;
   [row, col] = size(op0);

   upper_limit_row = (row - 1);
   upper_limit_col = (col - 1);
   row_offset = floor(row/2);
   col_offset = floor(col/2);

elseif ~isempty(findstr('gen',choice_operator))        % for generalized diagonal operator
   min_threshold_edge_1 = 150;

   gen_edge;                                % run script for loading operators for edge detection
   [row, col] = size(gen1_2_1_1);           % used because these operators will change sizes

   upper_limit_row = (row - 1);
   upper_limit_col = (col - 1);
   row_offset = floor(row/2);
   col_offset = floor(col/2);

   if strcmp(choice_operator,'gen_pos_lt_to_dk')
         generic_op_1 = gen1_3_1_2;      % positive slope
         generic_op_2 = gen1_3_2_2;
   elseif strcmp(choice_operator,'gen_pos_dk_to_lt')
         generic_op_1 = gen1_3_1_1;      % positive slope
         generic_op_2 = gen1_3_2_1;
   elseif strcmp(choice_operator,'gen_neg_dk_to_lt')
         generic_op_1 = gen1_2_1_2;      % negative slope dark to light one
         generic_op_2 = gen1_2_2_2;      % negative slope dark to light two
   elseif strcmp(choice_operator,'gen_neg_lt_to_dk')
         generic_op_1 = gen1_2_1_1;      % negative slope light to dark one
         generic_op_2 = gen1_2_2_1;      % negative slope light to dark two
   end;
```

Figure X (continued): Code listing portion for function detect_edge_gen_bkp.m

```
elseif ~isempty(findstr('Grad',choice_operator)) || ~isempty(findstr('Lap',choice_operator)) ||
~isempty(findstr('Diff',choice_operator))
   min_threshold_edge_1 = 15;

   if strcmp(choice_operator,'Grad_mag')
      [col_grad,row_grad] = gradient(double(image_data));
      out_image_b4_discrimination = sqrt((col_grad.^2) + (row_grad.^2));
   elseif strcmp(choice_operator,'Laplacian')
      out_image_b4_discrimination = del2(double(image_data));
   elseif strcmp(choice_operator,'Diff_mag')
   end;

elseif ~isempty(findstr('vert',choice_operator))          % for generalized vertical diagonal operator
   min_threshold_edge_1 = 55;
   gen_edge_1;                                    % run script for loading operators for edge detection
   [row, col] = size(gen2_2_1);                   % Negative dark to light edge detector (vertical)

   upper_limit_row = (row - 1);
   upper_limit_col = (col - 1);
   row_offset = floor(row/2);
   col_offset = floor(col/2);

   generic_op_1 = gen2_2_1;      % vertical dark to light transition
   generic_op_2 = gen2_2_2;      % vertical lt to dk transition

elseif ~isempty(findstr('horiz',choice_operator))         % for generalized horizontal diagonal operator
   min_threshold_edge_1 = 230;
   gen_edge_1_1;
   [row, col] = size(gen2_2_1);

   upper_limit_row = (row - 1);
   upper_limit_col = (col - 1);
   row_offset = floor(row/2);
   col_offset = floor(col/2);

   generic_op_1 = gen2_2_1;
   generic_op_2 = gen2_2_2;

elseif ~isempty(findstr('line_v',choice_operator))
   min_threshold_edge_1 = 200;
   gen_edge_3;

   [row, col] = size(gen3_1);

   upper_limit_row = (row - 1);
   upper_limit_col = (col - 1);
   row_offset = floor(row/2);
   col_offset = floor(col/2);
   generic_op_1 = gen3_1;
end;
```

Figure X (continued): Code listing portion for function detect_edge_gen_bkp.m

```
if ~isempty(findstr('Sobel',choice_operator)) || ~isempty(findstr('Prewitt',choice_operator))
            || ~isempty(findstr('Frei_chen',choice_operator))
   for x = 1:rows-upper_limit_row,
      for y = 1:cols-upper_limit_col,

         total1 = 0;
         total1 = sum(sum(double(image_data(x:x+(row-1),y:y+(col-1))) .* op0));
         if (total1 > max_pixel_value)
            total1 = max_pixel_value;
         elseif (total1 < min_pixel_value)
            total1 = min_pixel_value;
         end;
         if (total1 > out_image(x+1,y+1)),
            out_image(x+1,y+1) = total1;
         end;

         total1 = 0;
         total1 = sum(sum(double(image_data(x:x+(row-1),y:y+(col-1))) .* op1));
         if (total1 > max_pixel_value),
            total1 = max_pixel_value;
         elseif (total1 < min_pixel_value),
            total1 = min_pixel_value;
         end;
         if (total1 > out_image(x+1,y+1)),
            out_image(x+1,y+1) = total1;
         end;

         total1 = 0;
         total1 = sum(sum(double(image_data(x:x+(row-1),y:y+(col-1))) .* op2));
         if (total1 > max_pixel_value),
            total1 = max_pixel_value;
         elseif (total1 < min_pixel_value),
            total1 = min_pixel_value;
         end;
         if (total1 > out_image(x+1,y+1)),
            out_image(x+1,y+1) = total1;
         end;

         total1 = 0;
         total1 = sum(sum(double(image_data(x:x+(row-1),y:y+(col-1))) .* op3));
         if (total1 > max_pixel_value),
            total1 = max_pixel_value;
         elseif (total1 < min_pixel_value),
            total1 = min_pixel_value;
         end;
         if (total1 > out_image(x+1,y+1)),
            out_image(x+1,y+1) = total1;
         end;
```

Figure X (continued): Code listing portion for function detect_edge_gen_bkp.m

```
        total1 = 0;
        total1 = sum(sum(double(image_data(x:x+(row-1),y:y+(col-1))) .* op4));
        if (total1 > max_pixel_value),
           total1 = max_pixel_value;
        elseif (total1 < min_pixel_value),
           total1 = min_pixel_value;
        end;
        if (total1 > out_image(x+1,y+1)),
           out_image(x+1,y+1) = total1;
        end;

        total1 = 0;
        total1 = sum(sum(double(image_data(x:x+(row-1),y:y+(col-1))) .* op5));
        if (total1 > max_pixel_value),
           total1 = max_pixel_value;
        elseif (total1 < min_pixel_value),
           total1 = min_pixel_value;
        end;
        if (total1 > out_image(x+1,y+1)),
           out_image(x+1,y+1) = total1;
        end;

        total1 = 0;
        total1 = sum(sum(double(image_data(x:x+(row-1),y:y+(col-1))) .* op6));
        if (total1 > max_pixel_value),
           total1 = max_pixel_value;
        elseif (total1 < min_pixel_value),
           total1 = min_pixel_value;
        end;
        if (total1 > out_image(x+1,y+1)),
           out_image(x+1,y+1) = total1;
        end;

        total1 = 0;
        total1 = sum(sum(double(image_data(x:x+(row-1),y:y+(col-1))) .* op7));
        if (total1 > max_pixel_value),
           total1 = max_pixel_value;
        elseif (total1 < min_pixel_value),
           total1 = min_pixel_value;
        end;
        if (total1 > out_image(x+1,y+1)),
           out_image(x+1,y+1) = total1;
        end;
     end;    % for columns
  end;      % for rows
  out_image_b4_discrimination = out_image;
end;        % for particular operator
```

Figure X (continued): Code listing portion for function detect_edge_gen_bkp.m

```
if ~isempty(findstr('gen',choice_operator))        % for generalized diagonal operator
   for x = 1:rows-upper_limit_row,                          % for each of rows
      for y = 1:cols-upper_limit_col,                       % for each of cols
         total1(x+row_offset,y+col_offset) = 0;
         total2(x+row_offset,y+col_offset) = 0;
         total1(x+row_offset,y+col_offset) = sum(sum(double(image_data(x:x+(row-1),y:y+(col-1))) .*
           generic_op_1));
         total2(x+row_offset,y+col_offset) = sum(sum(double(image_data(x:x+(row-1),y:y+(col-1))) .*
           generic_op_2));




         if (min_total1(x+row_offset,y+col_offset) > max_pixel_value)
            min_total1(x+row_offset,y+col_offset) = max_pixel_value;
         elseif (min_total1(x+row_offset,y+col_offset) < min_pixel_value)
            min_total1(x+row_offset,y+col_offset) = min_pixel_value;
         end;
         if (min_total1(x+row_offset,y+col_offset) > out_image(x+row_offset,y+col_offset)),
            out_image(x+row_offset,y+col_offset) = min_total1(x+row_offset,y+col_offset);
         end;
      end;       % for columns
   end;         % for rows
   out_image_b4_discrimination = out_image;
end;           % for particular operator

if ~isempty(findstr('vert',choice_operator)) || ~isempty(findstr('horiz',choice_operator))
   for x = 1:rows-upper_limit_row,                          % for each of rows
      for y = 1:cols-upper_limit_col,                       % for each of cols
         total1(x+row_offset,y+col_offset) = 0;
         total1(x+row_offset,y+col_offset) = sum(sum(double(image_data(x:x+(row-1),y:y+(col-1))) .*
           generic_op_1));

         total2(x+row_offset,y+col_offset) = 0;
         total2(x+row_offset,y+col_offset) = sum(sum(double(image_data(x:x+(row-1),y:y+(col-1))) .*
           generic_op_2));

         total1(x+row_offset,y+col_offset) =
           max(total1(x+row_offset,y+col_offset),total2(x+row_offset,y+col_offset));

         if (total1(x+row_offset,y+col_offset) > max_pixel_value),
            total1(x+row_offset,y+col_offset) = max_pixel_value;
         elseif (total1(x+row_offset,y+col_offset) < min_pixel_value),
            total1(x+row_offset,y+col_offset) = min_pixel_value;
         end;
         if (total1(x+row_offset,y+col_offset) > out_image(x+row_offset,y+col_offset)),
            out_image(x+row_offset,y+col_offset) = total1(x+row_offset,y+col_offset);
         end;
      end;       % for columns
   end;         % for rows
   out_image_b4_discrimination = out_image;
end;           % for particular operator
```

Figure X (continued): Code listing portion for function detect_edge_gen_bkp.m

```
if ~isempty(findstr('line_v',choice_operator))
   for x = 1:rows-upper_limit_row,                          % for each of rows
      for y = 1:cols-upper_limit_col,                       % for each of cols
         total1(x+row_offset,y+col_offset) = 0;
         total1(x+row_offset,y+col_offset) = sum(sum(double(image_data(x:x+(row-1),y:y+(col-1))) .* generic_op_1));

         if (total1(x+row_offset,y+col_offset) > max_pixel_value),
            total1(x+row_offset,y+col_offset) = max_pixel_value;
         elseif (total1(x+row_offset,y+col_offset) < min_pixel_value),
            total1(x+row_offset,y+col_offset) = min_pixel_value;
         end;
         if (total1(x+row_offset,y+col_offset) > out_image(x+row_offset,y+col_offset)),
            out_image(x+row_offset,y+col_offset) = total1(x+row_offset,y+col_offset);
         end;
      end;       % for columns
   end;          % for rows
   out_image_b4_discrimination = out_image;
end;             % for particular operator
```

Figure X (continued): Code listing portion for function detect_edge_gen_bkp.m

As can be seen from figure X, the desired operation is passed to the function along with the grayscale approximation to the original cropped RGB image. After the string (indicating the desired edge mask) is identified, the necessary script files are called and variables are filled to facilitate the edge detection process. Note that the function can operate on compass masks, individual edge operators, line operators, built-in Matlab functions, and has the capability to be easily expanded to use other operators and functions. The function of figure X essentially performs a form of correlation to quantitatively assess how well the local image portion corresponds to the particular mask.

6.5 Preliminary edge segmentation, 'first level discrimination', and intermediate detection results

The general flow to the initial segmentation code segment is given in figure X.

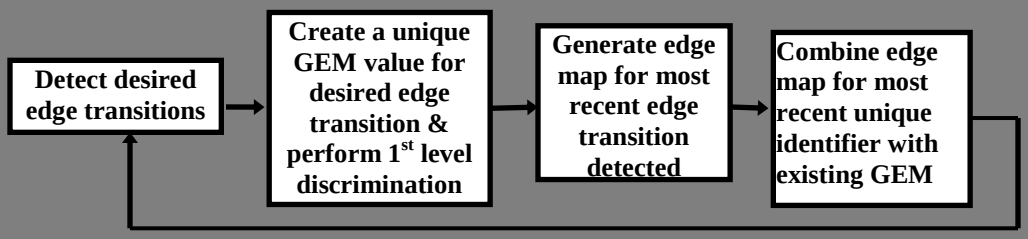


Figure X: High level flow to first portion of image segmentation

During the edge detection process, thresholds were established to compare the output of the correlation to. Preliminary values for these thresholds were initially established by looking at the value resulting from a correlation with a respective mask for a 'minimally visible' dotted, broken and/or normal lane marker edge within the ROI.

After a specific edge transition has been analyzed for a complete road scene image, those relative edge strengths are then passed to the function perform_discrimination.m to undergo first level discrimination. A portion of the function perform_discrimination.m is shown in figure X. As may be seen in figure X, the function will automatically discriminate values outside of the region of interest. If the edge location is within the region of interest, the function will next look at a number of image traits including (but not limited to) the local image color characteristics, the location of the edge within the image, the particular type of edge transition, and the edges degree of correlation to the desired mask to determine whether the edge value 'passes' the first level of discrimination.

This 'first level discrimination' provides an initial discriminating capability for the edge map being created before it is added to the GEM. As may be seen from the code of figure X, the function looks for edge transition data with a minimum specified level of correlation to the edge operator while reducing the potential amount of non-road surfaces also being considered during subsequent analysis. It is *extremely important* to make sure that the road scene image data which is considered 'outside the region of interest' does not contain *the only* reliably detectable valid lane markings within the current FOV which may provide useful lane delineation information to the preliminary road scene image processing system. Note that in general, the further a lane marker is located from the vantage point of the road scene image processing system, the less reliable its lane delineation information is in relating the current position of the vehicle to the current lane boundaries.

```
% Programmer: Christopher Alan Warner
% PERFORM DISCRIMINATION is a function that performs discrimination on the
% specific edge detected image (out_image_b4_thresholding) passed from
% detect_edge_gen (+) and returns a discriminated grayscale image.
% ================================================================
%     INPUT LIST (in order that they appear as parameter list):
% ================================================================
% 1) INITIAL_ROW_TO_SEARCH = since the entire image is not currently being analyzed, this variable limits the range
of values for processing
% 2) ORIGINAL_IMAGE_DATA = used as image being compared to to look for 'white' grayscale value image pixels for
discrimination (GRAYSCALE VALUES)
% 3) OUT_IMAGE_B4_THRESHOLDING  =  the 'edge detected' image passed from detect_edge_gen_bkp (+)
% 4) DISCRIMINATION_LEVEL = original intent to have multiple discrimination levels but not currently implemented
% 5) EDGE_OPERATOR_OFFSET = set limits for rows and cols to consider according to size of edge operator used.
% 6) YELLOW_PORTION_V = map of image containing locations with high likelihood of representing RGB 'yellow' or
'shade of yellow' characteristic corresponding to yellow longitudinal road marker.
% 7) GEM_VALUE = The edge transition that is currently being considered for discrimination

% ================================================================
%     OUTPUT LIST (in order that they appear as parameter list):
% ================================================================
% 1) IMAGE_DATA = the discriminated image (GRAYSCALE VALUES)
% 2) THRESH_MAP = a map containing region based thresholds to be used in current and subsequent discrimination
% ================================================================
%     OTHER CUSTOM FUNCTIONS USED
% ================================================================
% 1) NONE
function [image_data, thresh_map] = perform_discrimination(initial_row_to_search, original_image_data,
            out_image_b4_thresholding, discrimination_level, edge_operator_offset, yellow_portion_v, gem_value)

rows = []; cols = [];
[rows,cols] = size(out_image_b4_thresholding);

discriminated_image = zeros(rows,cols);
image_data = zeros(rows,cols);
min_pixel_value = 0;    max_pixel_value = 255;
thresh_map = zeros(rows,cols);

col_counter = 0;
initial_col_offset = 80;
final_col_offset =  []; final_col_offset = cols - initial_col_offset;
```

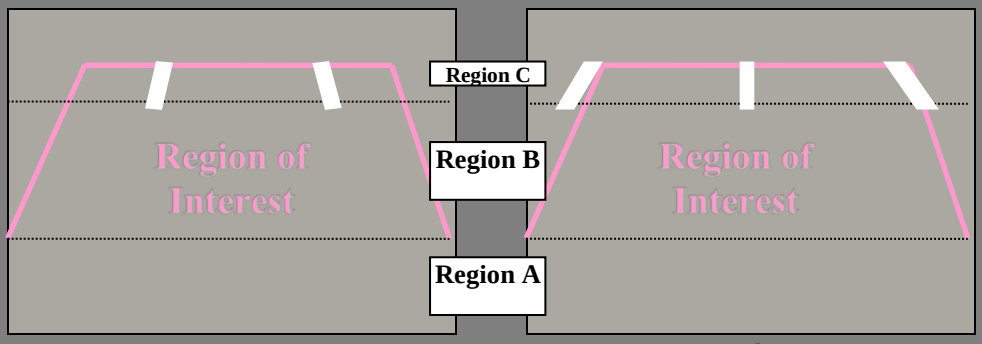
Figure X: Code listing portion for function perform_discrimination.m

```
if (discrimination_level == 1)
   for x = edge_operator_offset + 1:rows - edge_operator_offset,                    % for each of rows
      if (x > initial_row_to_search)
         col_counter = col_counter + 1;
      end;
      for y = edge_operator_offset + 1:cols - edge_operator_offset,          % Edge operator accounts for size of mask
         yellow_block_length = [];
         yellow_block_length = length(find(yellow_portion_v(x-1:x+1,y-1:y+1)));

         if (~yellow_portion_v(x,y)) || (gem_value <= 2)
            if (gem_value == 1) || (gem_value == 2)
               min_threshold_edge_1 = one_two_thresh_map(x,y);
            elseif (gem_value == 3) || (gem_value == 4)
               min_threshold_edge_1 = three_four_thresh_map(x,y);
            end;

         elseif (gem_value > 2)
            if (y < mid_point_cols)
% *** If on 'yellow' and 'significant local yellow' and 'considering only pos sloped edges'
               if (yellow_portion_v(x,y)) && (yellow_block_length >= yellow_block_length_min)
                  min_threshold_edge_1 = dom_yellow_threshold;
               elseif (yellow_portion_v(x,y)) && (yellow_block_length < yellow_block_length_min)
                  min_threshold_edge_1 = nondom_yellow_threshold;
               end;
               three_four_thresh_map(x,y) = min_threshold_edge_1;
            elseif (x >= furthest_row_mod_roi)
               three_four_thresh_map(x,y) = default_max_threshold;          % To keep uniform look of map
            end;
         end;

         if (x < initial_row_to_search)
            discriminated_image(x,y) = min_pixel_value;
         elseif (x >= initial_row_to_search) && ((y < initial_col_offset - col_counter) || (y > final_col_offset + col_counter))
            discriminated_image(x,y) = min_pixel_value;
         elseif (out_image_b4_thresholding(x,y) >= min_threshold_edge_1)    % check to see if edge strength is large
            discriminated_image(x,y) = out_image_b4_thresholding(x,y);      % if so, pixel gets ORIGINAL GRAYSCALE
         else
            discriminated_image(x,y) = min_pixel_value;
         end;
      end;
   end;
end;
image_data = discriminated_image;
```

Figure X (continued): Code listing portion for function perform_discrimination.m

Width considerations for the ROI were made taking factors including (but not limited to) road geometry and vehicle dynamic operating conditions into account. Although the widths of a lane may range from as little as 2.6 meters to as much as 4.2 meters, a typical highway lane width will be approximately 3.6 meters (12 feet). The original uncropped 640 x 480 pixel RGB road scene images showed a much more extensive FOV including the vehicle's hood, a much larger area above the 'typical' horizon line, and a wider FOV. The cropping to 164 x 397 pixels was performed because the student version of Matlab limits images to 64K pixels. Thus, tradeoffs needed to be made when considering the need to get the necessary lateral lane information while sacrificing as little longitudinal FOV information as possible. Hence, a vehicle located in approximately the center of an artificial lane as shown in figure X (a) should have a ROI including the entire width of that lane. In addition, portions of adjacent lanes further into the FOV may also be included in the ROI. A

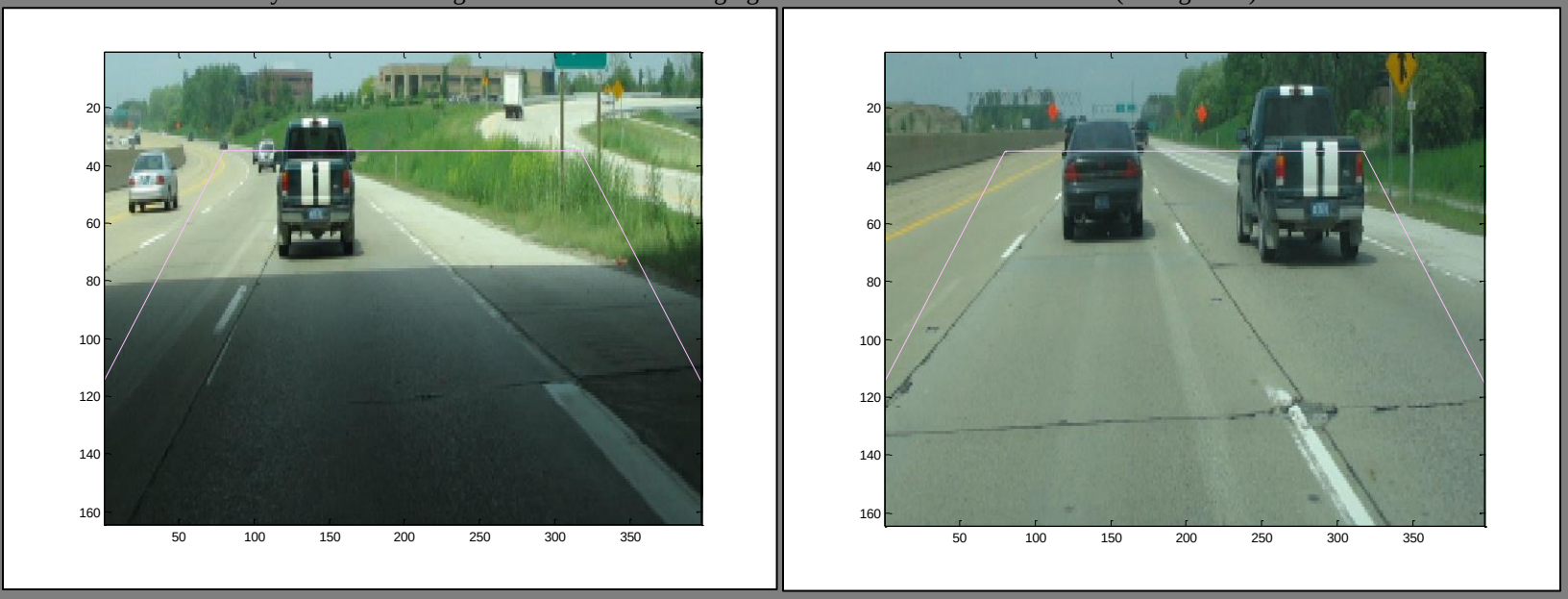
vehicle located approximately in between lanes as shown in (b) of figure X should have a ROI including a minimum of half of each lane to each side.



(a) (b)

Figure X: Artificial road scene image with sample longitudinal lane makers in FOV for (a) vehicle positioning near 'center of lane'; (b) vehicle positioning far from 'center of lane'

From figure X, region A will typically have a higher probability for true edge detection and false edge discrimination than region B and region B will typically have a higher probability for true edge detection and false edge discrimination than region C. Similarly, region A will typically have a lower probability for false edge detection and true edge discrimination than region B and region B will typically have a lower probability for false edge detection and true edge discrimination than region C. These probabilities may not apply to longitudinal lane markers with small percentages of the marker occurring within the ROI but large percentages of the same marker occurring outside the ROI. Note that there are always road scene images which will be challenging no matter what the circumstances (see figure X).



(a) (b)

Figure X: Scenes (a), (b) of vehicle with attached stripes in region of interest

The outputs from various stages of the edge detection process are shown in figure X.

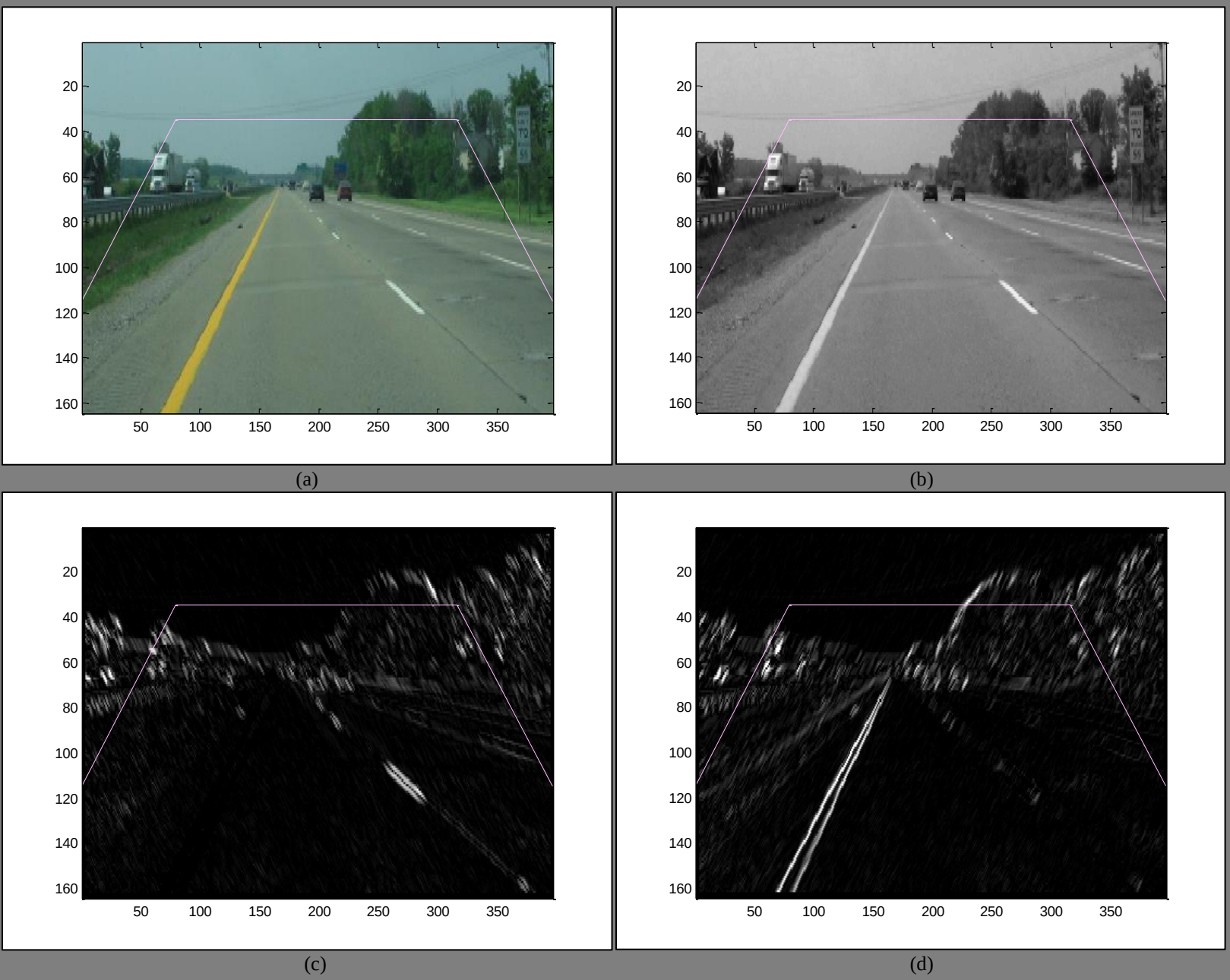
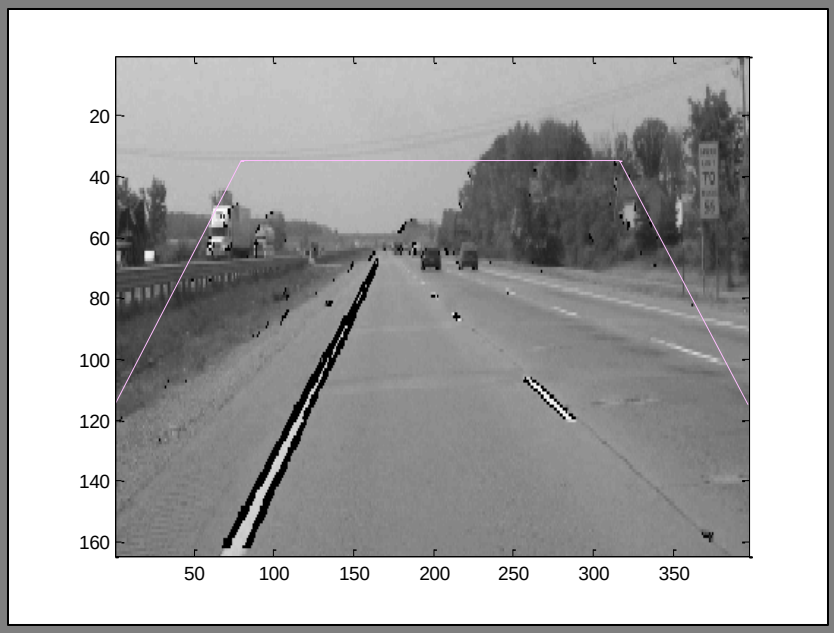


Figure X: (a) Original cropped RGB image data; (b) approximated grayscale version; (c) unthresholded output from negative slope generalized edge detector; (d) unthresholded output from positive slope generalized edge detector



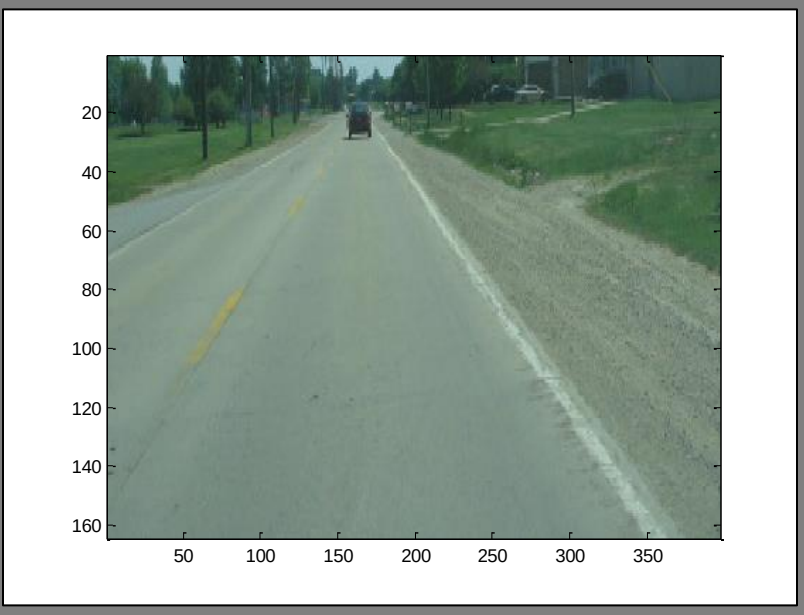
(e)

Figure X (continued): (e) Combined generalized edge detector masked output superimposed on grayscale approximation

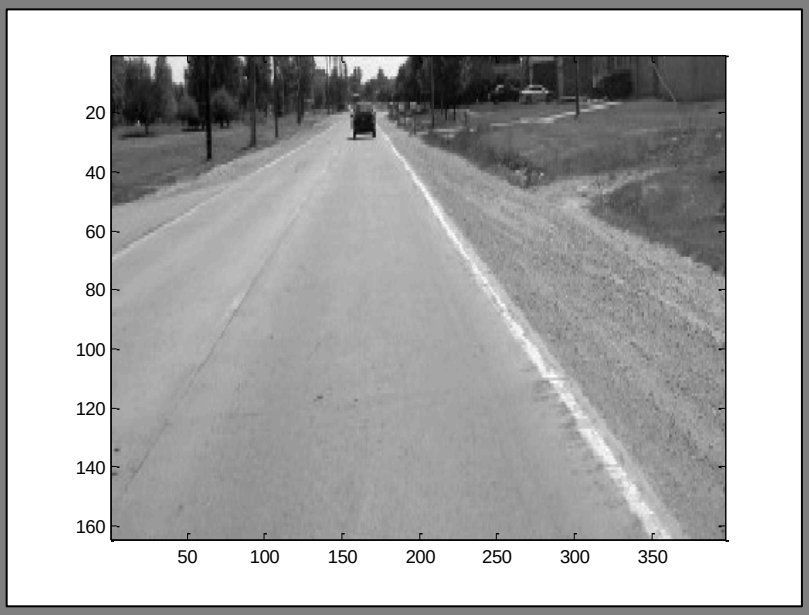
In figure X, (a) and (b) represent a cropped RGB road scene image and its grayscale approximation, respectively. Figure X (c) and (d) show the unthresholded negative and positive sloped image to mask correlations, respectively, generated via use of the generalized edge detector of figure X. Each of the unthresholded images is then edge thresholded, converted to binary, and then combined. Finally, figure X (e) shows the results of the combined binary thresholded masked edge detection superimposed on the grayscale approximation.

6.6 Preliminary explanation for color supplemented segmentation

One of the reasons color was chosen as a condition for contributing to edge discrimination can be further seen in the sequence of images shown in figure X.

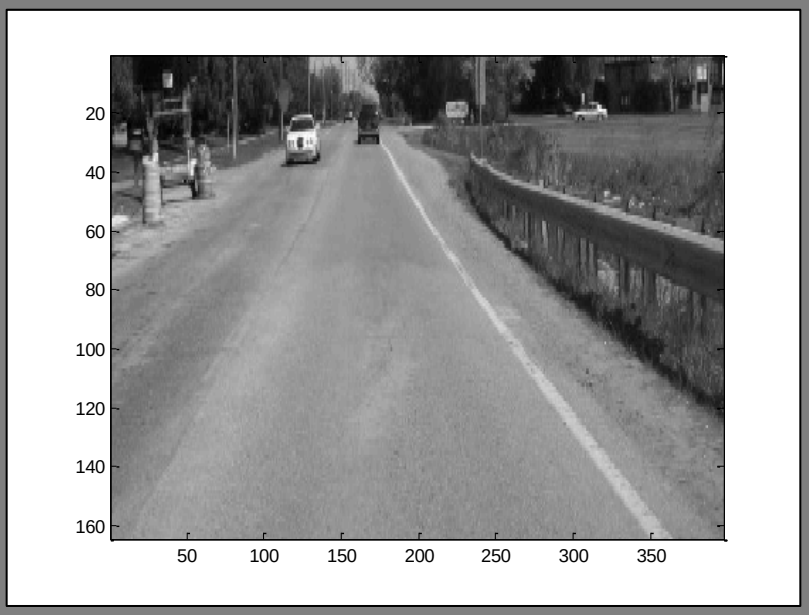


(a)



(b)

(c)



(d)

Figure X: (a) First cropped RGB road scene image with degraded broken yellow marker; (b) grayscale approximation to first image; (c) second color road scene image with degraded yellow marker; (d) grayscale approximation to second image

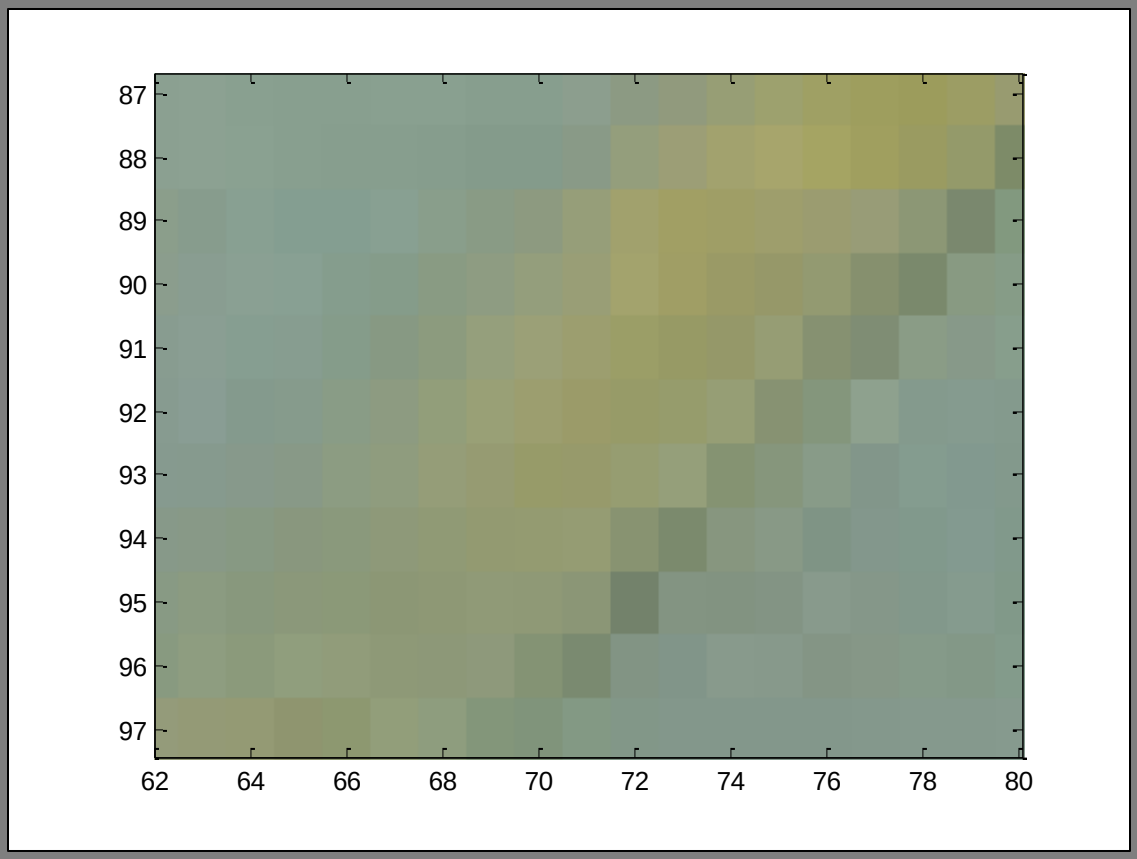


Figure X: Detail using Matlab zoom command for figure X (a) (rows 87:97, columns 62:80)

Table X: Grayscale approximated values from figure X (b) (rows 87:97, columns 62:80)

	C62	C63	C64	C65	C66	C67	C68	C69	C70	C71	C72	C73	C74	C75	C76	C77	C78	C79	C80
R87	160	161	160	159	159	160	160	158	158	158	154	154	158	161	160	158	156	157	155
R88	160	161	161	159	158	158	157	155	155	154	158	158	162	167	165	160	155	154	139
R89	158	156	160	158	158	160	158	155	154	158	161	161	159	158	156	156	151	136	153
R90	157	157	160	160	157	156	155	156	158	158	163	160	154	152	154	144	137	154	156
R91	156	158	158	157	156	153	155	159	160	158	158	154	152	157	145	141	156	153	158
R92	155	157	154	155	156	155	158	160	158	155	155	156	158	146	150	161	154	155	154
R93	154	154	153	153	156	156	157	155	155	154	157	159	147	150	155	150	156	153	153
R94	153	153	153	151	153	153	154	154	155	156	147	138	150	153	148	151	153	154	153
R95	154	155	152	152	153	151	152	154	153	150	130	148	147	148	154	151	152	155	153
R96	154	157	154	158	156	153	152	153	147	138	148	149	154	153	149	151	154	152	155
R97	155	154	154	149	152	158	157	150	148	153	151	151	151	151	151	152	153	153	154

As may be seen in table X, there is very little difference between the grayscale values for the degraded yellow broken marker and the local substrate. Also note from figure X, particularly on the rightmost edge of the yellow marker, these pixels do not appear to be shades of yellow but they do appear to be shades of brown. It is apparent that the degraded yellow marker in the lower left hand corner may or may not be distinguishable from the local substrate under the conditions using only grayscale approximated values from either of the transforms listed previously. These are only a couple of supporting reasons as to why the use of the RGB color space (or other chrominance information) to supplement edge detection of longitudinal markers may be necessary.

Now consider the road scenes of figure X where attempting to use color to assist in detecting the lane marker edges may be very challenging due to factors including the illumination source, luminous intensity, the angle of illumination and observation, illumination variation and shading, marker degradation, marker occlusion, substrate material and color, substrate surface finish, marker material and color, environmental conditions, reflectivity characteristics, application process, distance to markers being detected, etc.

(a)

(b)

(c)

(d)

Figure X: (a) Road scene image; (b) road scene image; (c) road scene image; (d) road scene image

Image (a) of figure X was taken while traveling through a tunnel with bright lights performing the illumination. Note the dominant shades of 'orange' for both the white broken longitudinal lane markers as well as the road surface, tunnel walls, etc. The image of figure X (b) was taken on an extremely sunny day on a darker colored substrate with degraded

white and yellow longitudinal lane markers while driving through the shade of trees planted along the side of the road. Figure X (c) was taken on an extremely sunny day on a light colored substrate with white longitudinal markers and the shade from an overpass while figure X (d) shows a yellow normal longitudinal marker in partial shade transitioning to bright sunlight. Clearly, important relationships exist between the light source(s) and reflective objects within a road scene image.

Figure X (a) indicates a situation in which the intended color of the white broken longitudinal lane markers has been 'visibly altered' due to the tunnel lighting. It is also apparent that the color characteristics of the road substrate have been 'visibly changed'. While this may affect certain color properties of and relationships between objects within the road scene image, note figure X which shows the grayscale approximation generated from figure X (a).

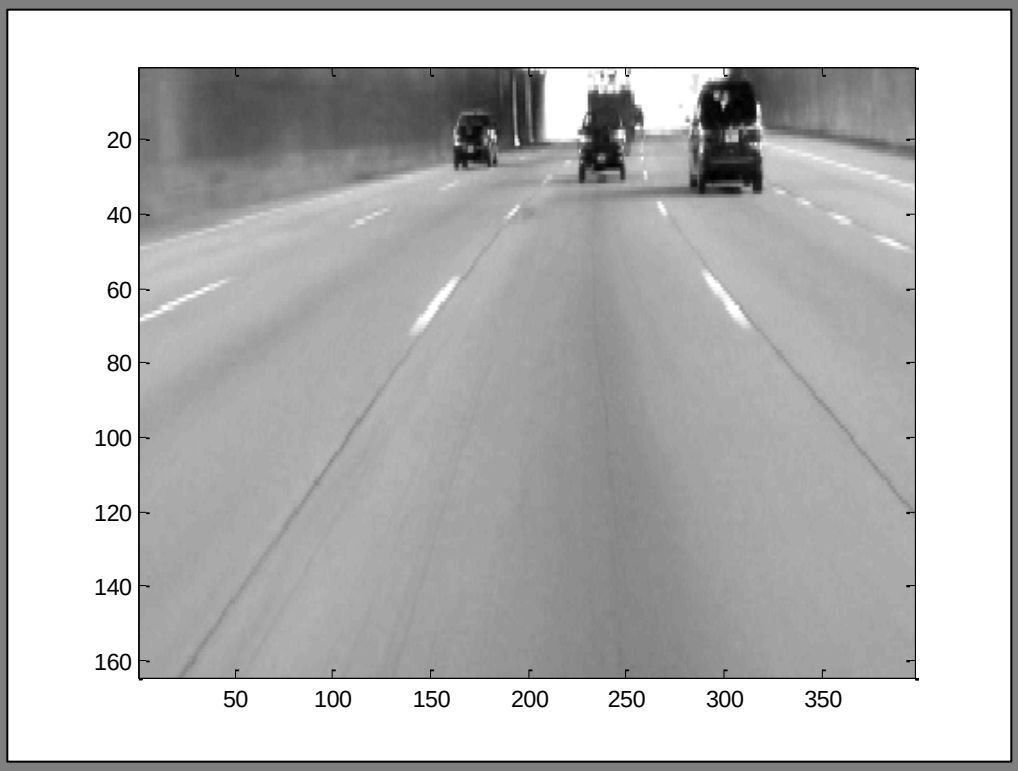


Figure X: Grayscale approximation of figure X (a)

As may be seen from figure X, using color to assist in detecting prospective lane marker edges may also produce detrimental results if not used cautiously. Up to this point, some sources of potential color variation for white and yellow longitudinal lane markers have been briefly looked at. Now consider figure X in which some of the many different possible shades for yellow longitudinal lane markers from various road scene images are presented.

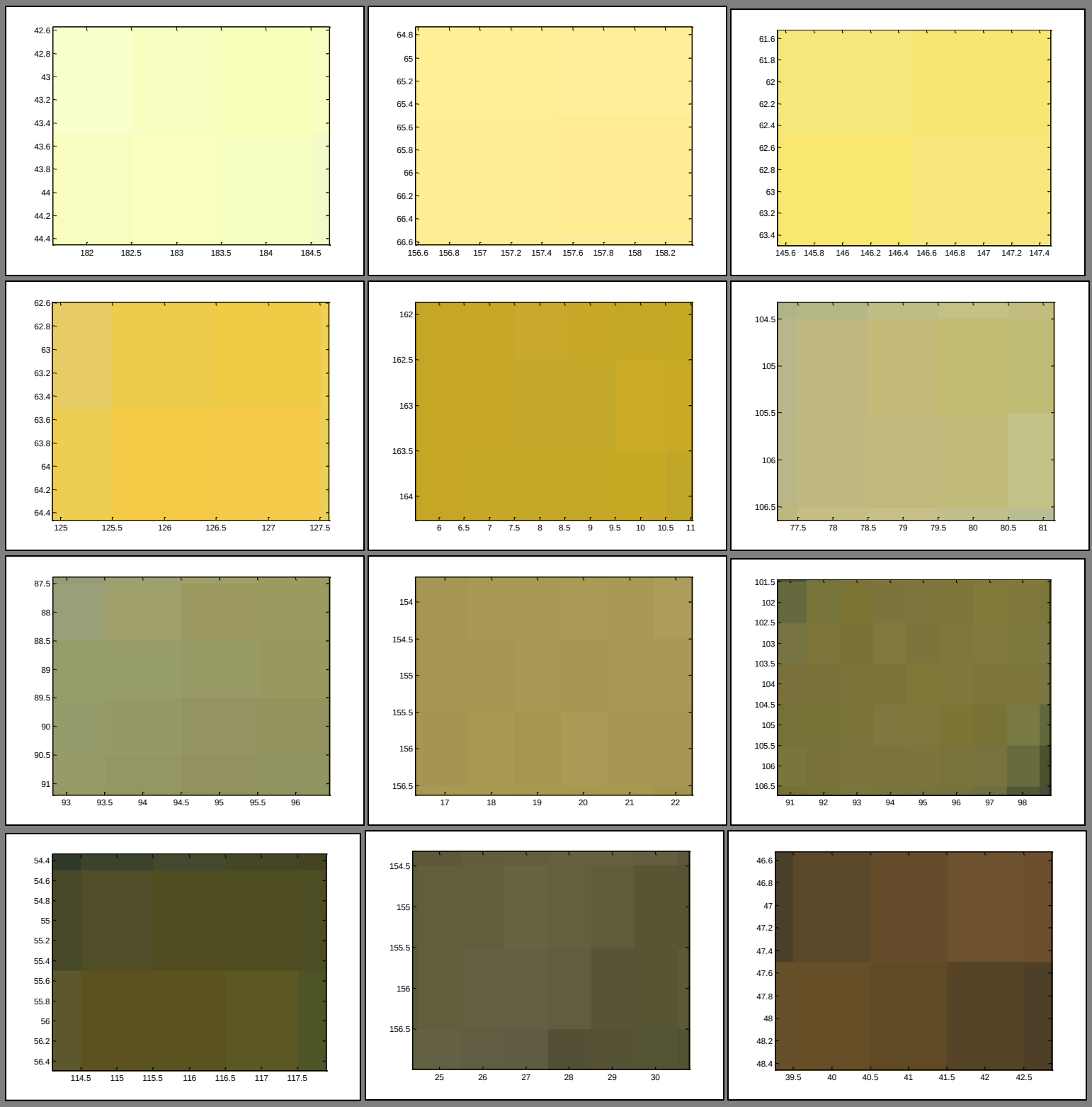


Figure X: Different shades taken from yellow longitudinal markers for various road scene images

From figure X, it is apparent that a yellow longitudinal lane marker may take on a wide range of shades. Now also consider that the transition near the outer edge of the marker (from marker to substrate) will typically have different

color characteristics than those nearer the center of the marker and will generally be affected by factors including (but not limited to) marker material, marker application process, substrate material, substrate application process, etc.

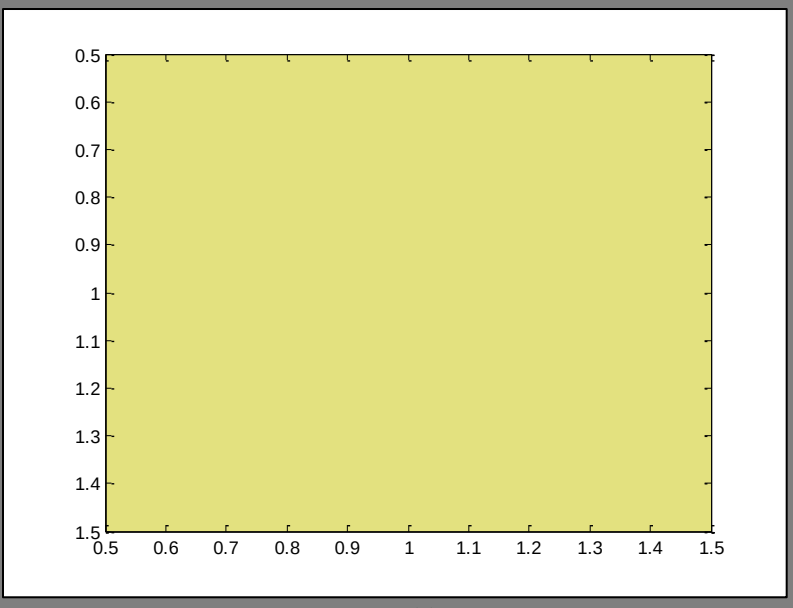
6.7 Preliminary isolation of various shades for yellow longitudinal markers and color segmentation

Isolating certain color characteristics may be performed using different methods. When considering the potential color spaces that may be used during this process, there are many important considerations to make (including but not limited to) the color range being generated by the imaging system, the color range available with the actual color space, the representation of the information within the color space, the actual collection of colors that we are attempting to isolate, the environmental conditions under which the colors are being observed, the illumination or shading conditions (or other sources of light variation) which may affect the intended color characteristics, the reflective properties of the particular objects being observed, etc. Although there are a number of different shades of yellows and browns in figure X, an example of a color specification established for yellow and white retroreflective pavement marking material with specific observer, geometry, and illuminant is given in table X [17].

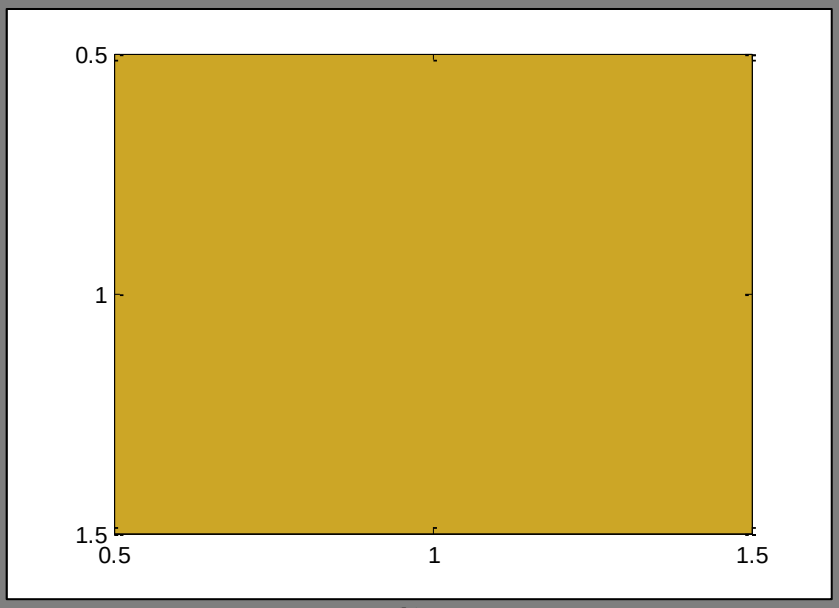
Table X. Daytime color specification limits for retroreflective pavement marking material with CIE 2° standard observer and 45/0 (0/45) geometry and CIE standard illuminant D_{65}

COLOR	Chromaticity coordinates							
	1		2		3		4	
	X	Y	X	Y	X	Y	X	Y
Yellow	0.560	0.440	0.490	0.510	0.420	0.440	0.460	0.400
White	0.355	0.355	0.305	0.305	0.285	0.325	0.335	0.375

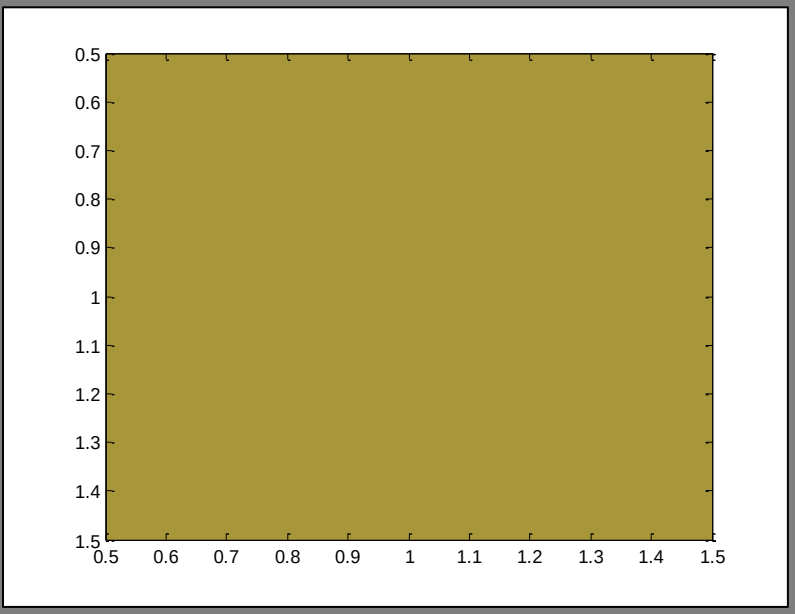
As has been mentioned previously, the Hue-Saturation-Value color space produces both chrominance (hue and saturation) and luminance (value or intensity) information. This chrominance and luminance information covering a wide range of the possible shades for a yellow longitudinal lane marker are ‘somewhat localized’ in the HSV space. However, extracting and quantifying the [H,S,V] triples associated with the complete range of yellow longitudinal marker shades would still entail generating a rather extensive and complicated representation. The interested reader may refer to references listed in the appendix for further explanation of the HSV color space. The RGB color space is another which may potentially be used to extract the range of yellow color characteristic shades associated with a yellow longitudinal lane marker. This would involve representing the [R,G,B] triples of the RGB color space in such a way as to quantify the desired color traits. This might prove even more cumbersome than using the HSV color space triples. Although each might be successfully applied to determine the necessary representation, there is a simpler approach which may be initially applied to quantify the necessary ranges for the yellow longitudinal marker color characteristic shades. This involves looking at the individual RGB values and ranges of values for ratios between the red to blue, blue to green, and green to red (row,column) image coordinate values. Several sets of [R,G,B] triples with associated green/red, red/blue, and blue/green ratios which may serve as a useful starting point for isolating some preliminary color characteristics of yellow longitudinal markers are shown in figure X.



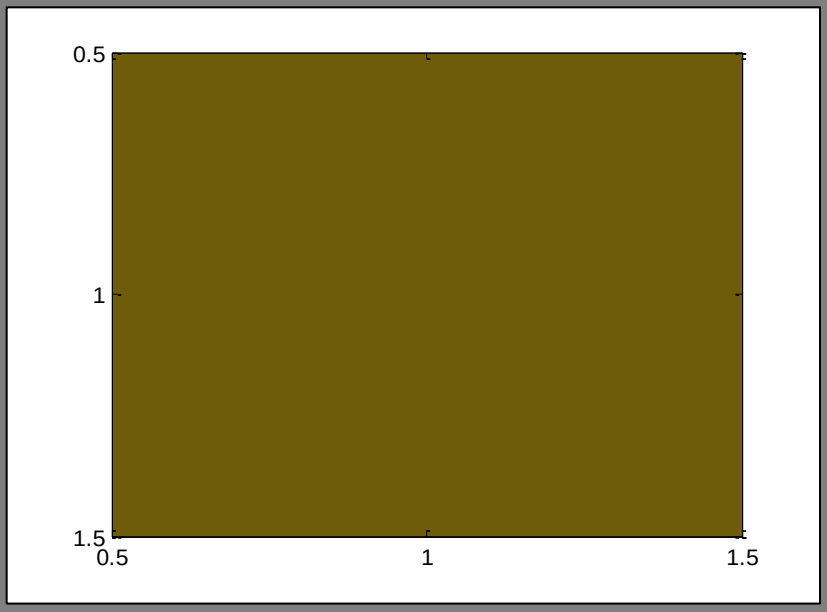
(a)



(b)



(c)

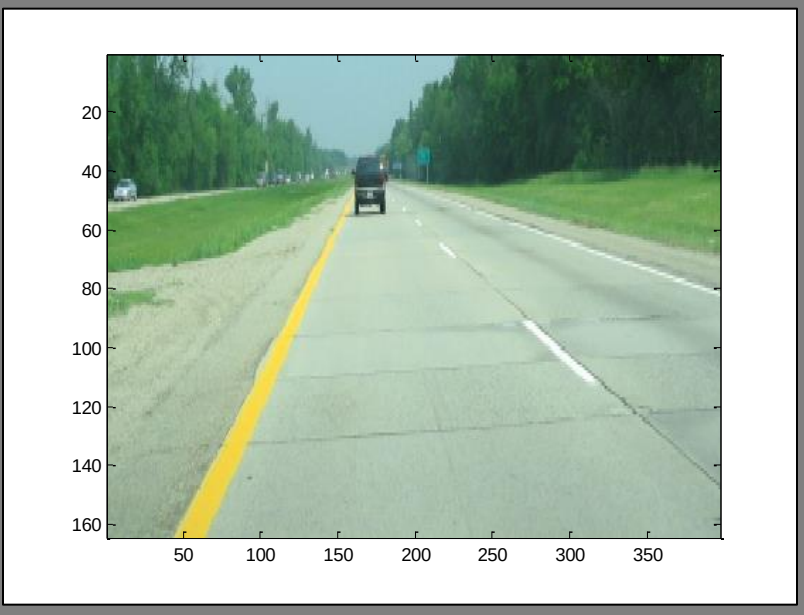


(d)

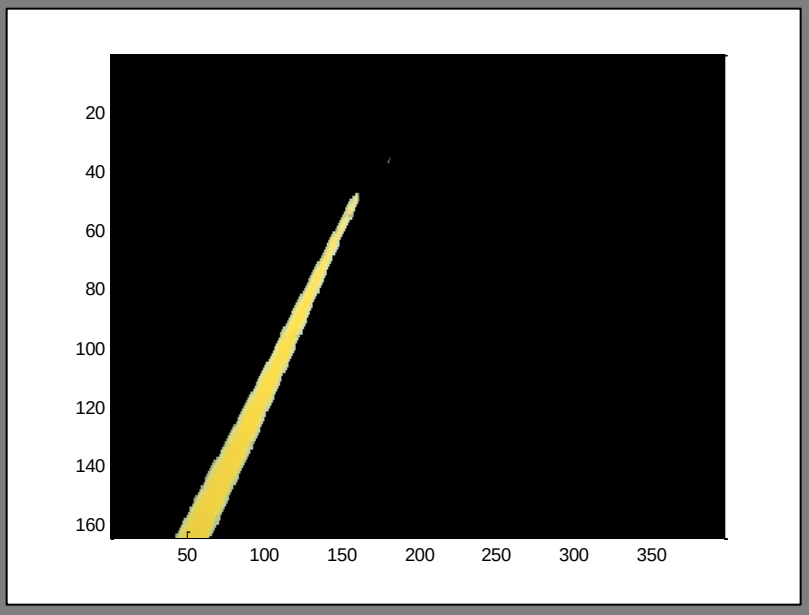
Figure X: RGB colors for (a) [R,G,B] = [0.89,0.88,0.5] with G/R = 0.98, R/B = 1.8, B/G = 0.57; (b) [R,G,B] = [0.8,0.65,0.15] with G/R = 0.83, R/B = 4.9, B/G = 0.25; (c) [R,G,B] = [0.66,0.59,0.23] with G/R = 0.89, R/B = 2.91, B/G = 0.39; (d) [R,G,B] = [0.43,0.36,0.04] with G/R = 0.84, R/B = 11, B/G = 0.11

When using [R, G, B] triples and their associated ratios, great care must be taken to properly eliminate the RGB color characteristics which are not supposed to be included in the yellow longitudinal marker color characteristic range. If performed correctly, the various yellow and brown shades which are isolated may represent those pixel locations with a high probability for correspondence to the color characteristics of a yellow longitudinal marker. *It is important to note that the presence of other objects or object portions within the FOV with very similar yellow color characteristics is possible when using only the color as a basis for isolation. These objects potentially possessing comparable yellow color characteristics include (but are not limited to) vehicles, other traffic control devices (signs, lights, curb designators, raised pavement markers, colored pavements, etc.), dry grass, 'yellow dirt', and discolored pavement.* However, it is not the yellow longitudinal marker which is being 'designated' through its color characteristic shade but a way of enhancing

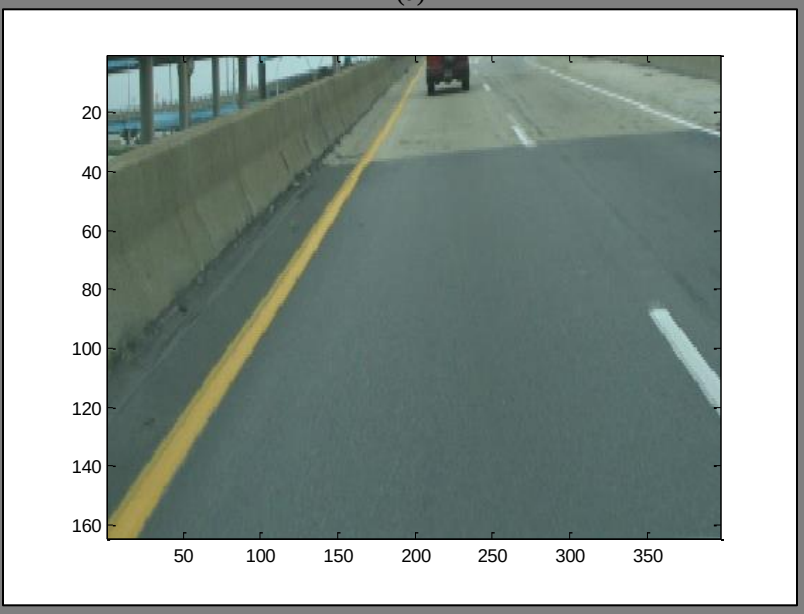
the discrimination process. Furthermore, this color information may be combined with GEM values and dimensional relationships towards a color segmented image which allows further boundary and region based feature determination and enables the potential differentiation between normal and normal, normal and broken, and broken and broken yellow pairs of close proximity adjacent longitudinal marker edges. Consider the road scene images of figure X in which yellow longitudinal lane marker *color characteristics* have been isolated independent of analysis pertaining to edge transitions or the dimensional characteristics of lane markers as they change in the FOV.



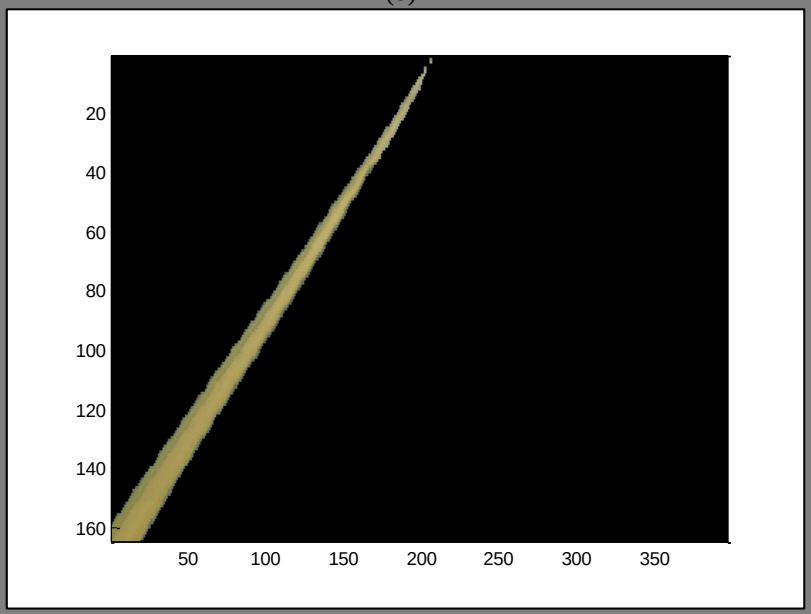
(a)



(b)

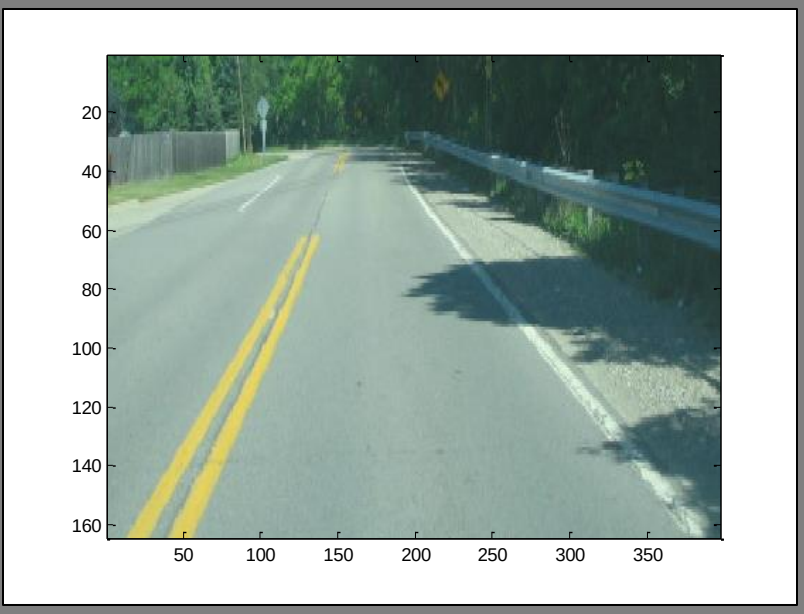


(c)

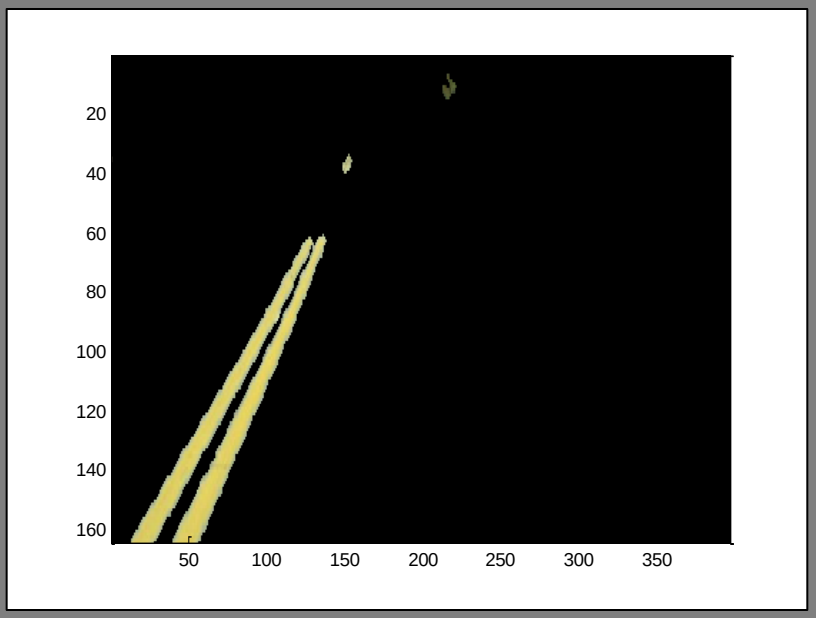


(d)

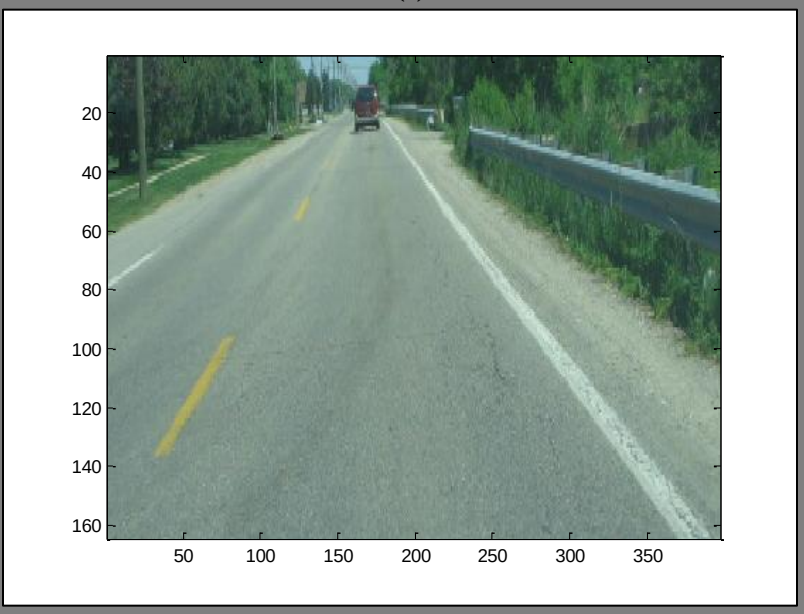
Figure X: (a) Cropped road scene image; (b) image with 'yellow longitudinal marker shades' color characteristic isolated; (c) cropped road scene image; (d) image with 'yellow longitudinal marker shades' color characteristic isolated



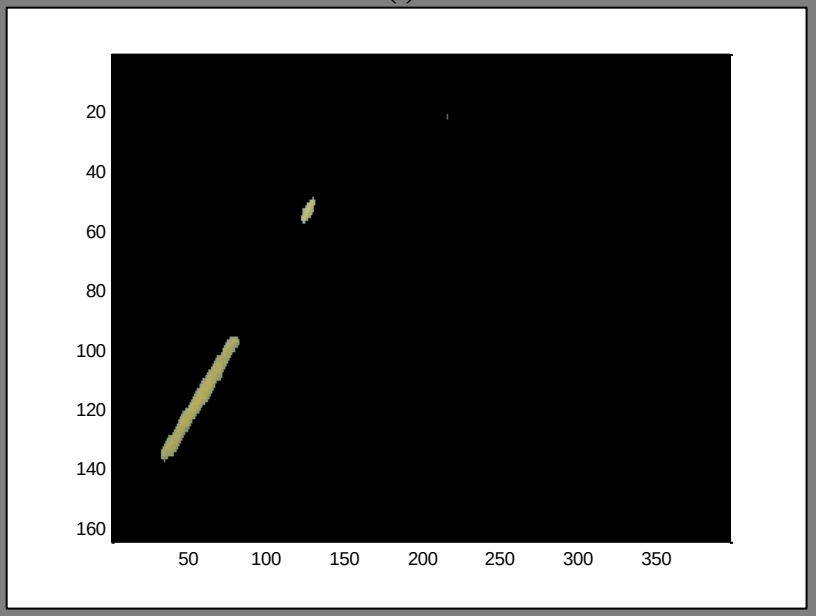
(e)



(f)



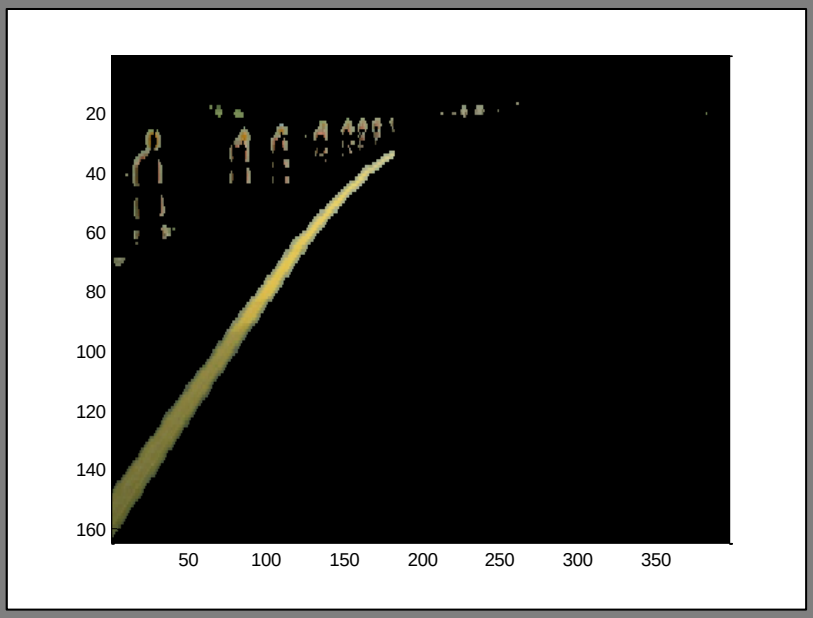
(g)



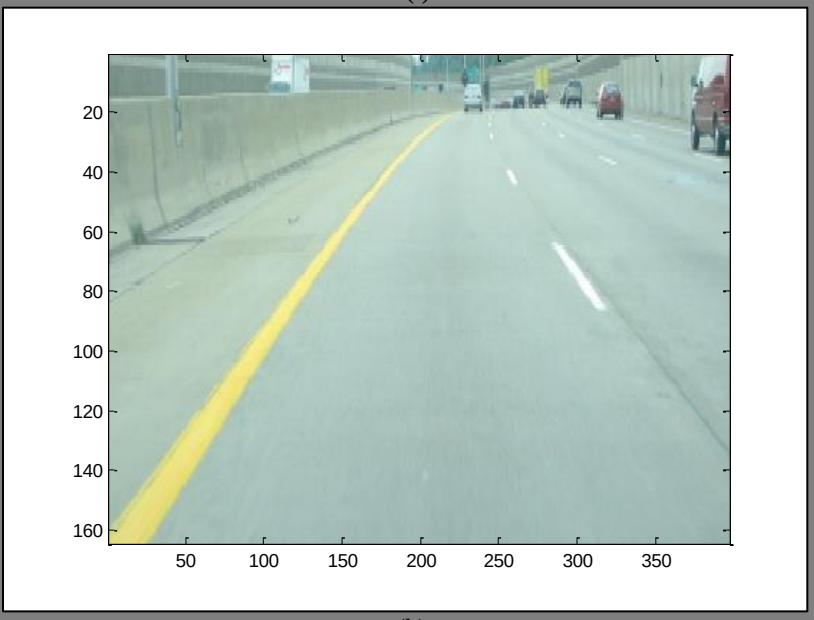
(h)

Figure X (continued): (e) Cropped road scene image; (f) image with 'yellow longitudinal marker shades' color characteristic isolated; (g) cropped road scene image; (h) image with 'yellow longitudinal marker shades' color characteristic isolated

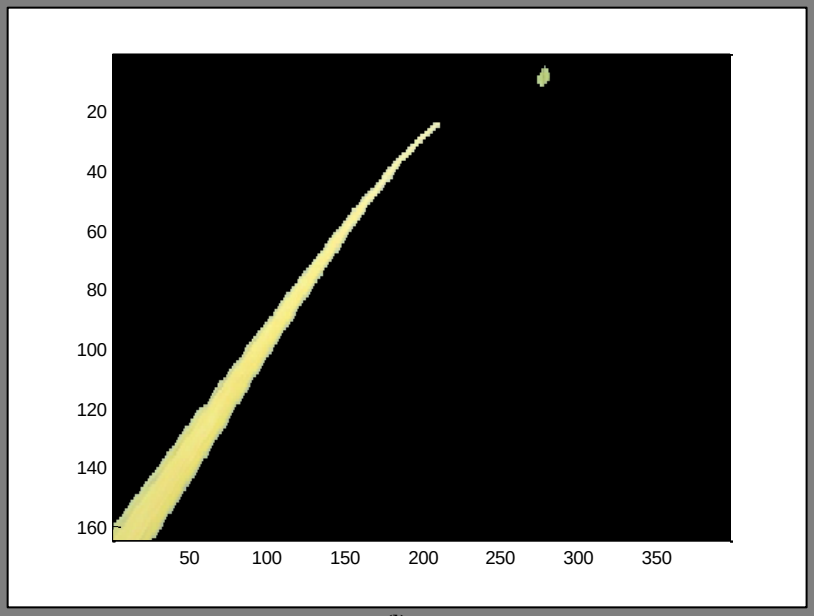
(i)



(j)



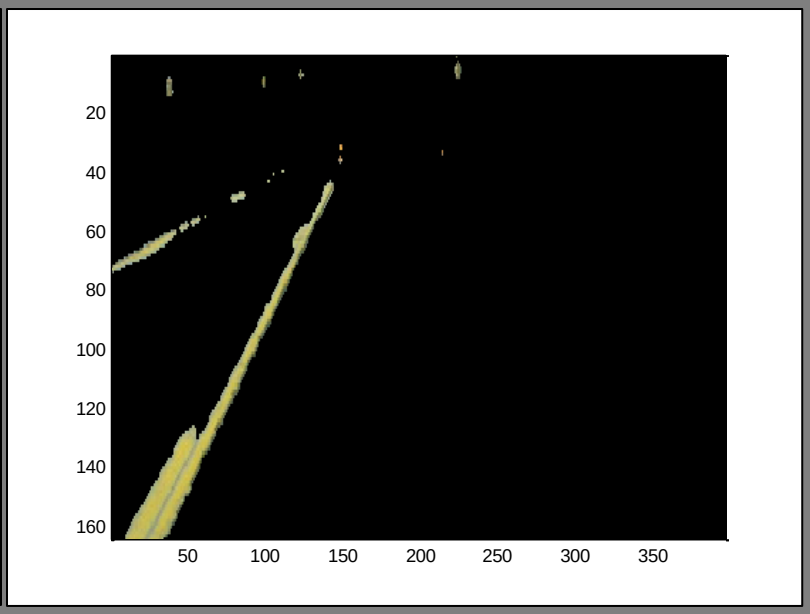
(k)



(l)

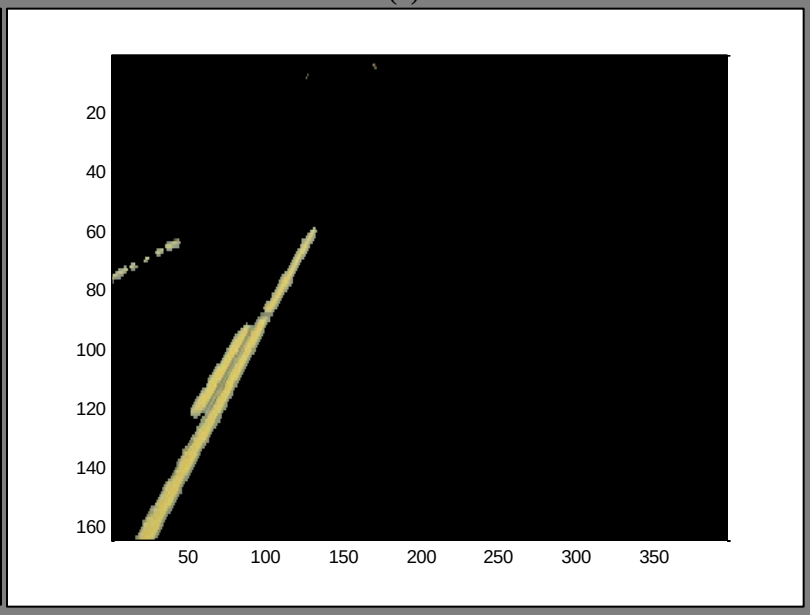
Figure X (continued): (i) Cropped road scene image; (j) image with 'yellow longitudinal marker shades' color characteristic isolated; (k) cropped road scene image; (l) image with 'yellow longitudinal marker shades' color characteristic isolated

(m)



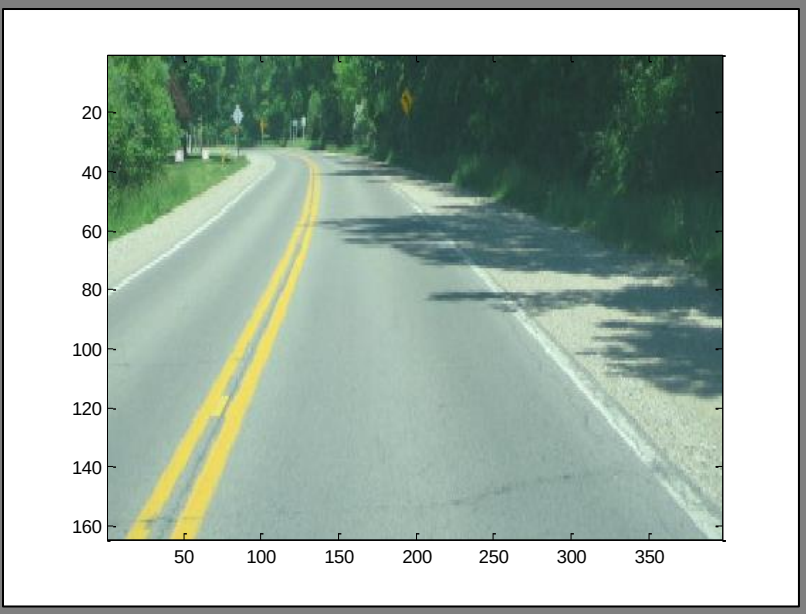
(n)

(o)

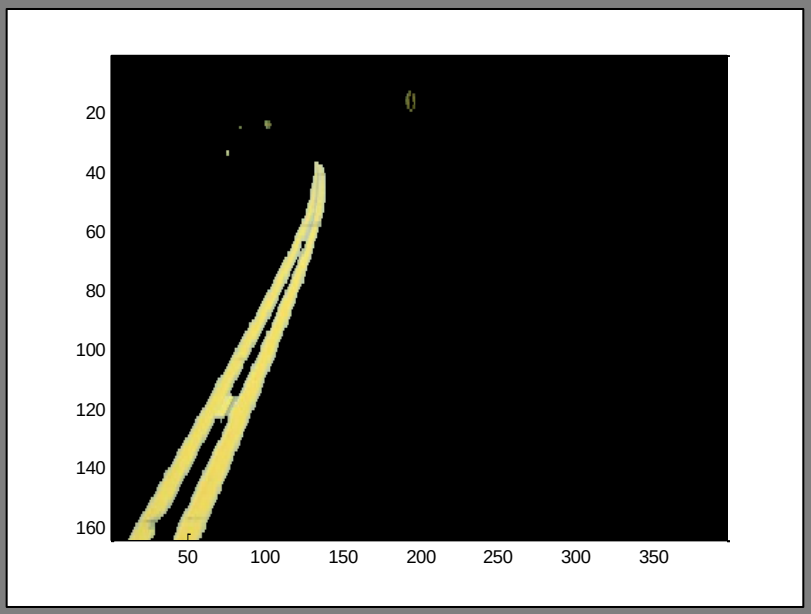


(p)

Figure X (continued): (m) Cropped road scene image; (n) image with 'yellow longitudinal marker shades' color characteristic isolated; (o) cropped road scene image; (p) image with 'yellow longitudinal marker shades' color characteristic isolated

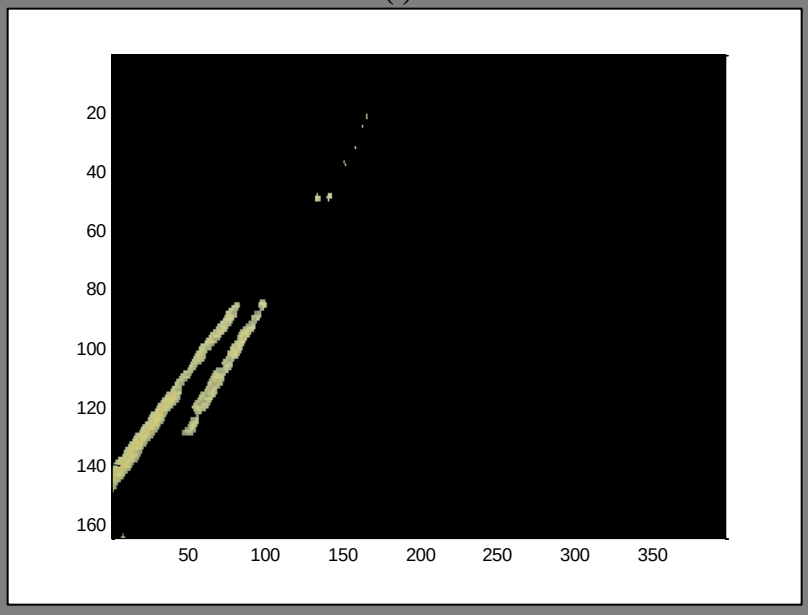


(q)



(r)

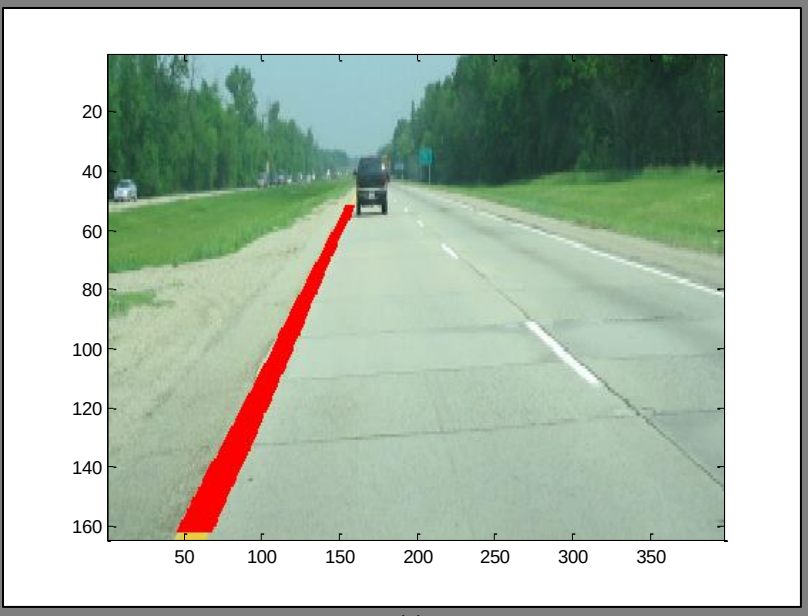
(s)



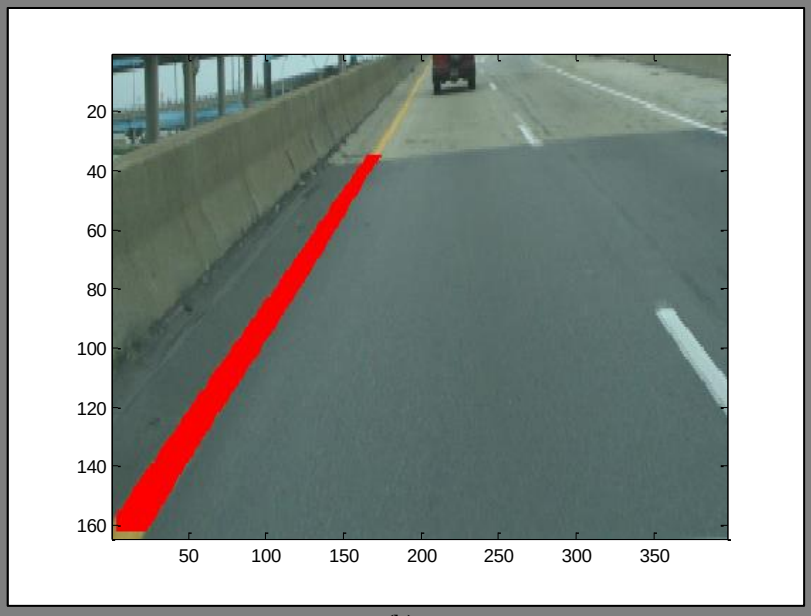
(t)

Figure X (continued): (q) Cropped road scene image; (r) image with 'yellow longitudinal marker shades' color characteristic isolated; (s) Cropped road scene image; (t) Image with 'yellow longitudinal marker shades' color characteristic isolated

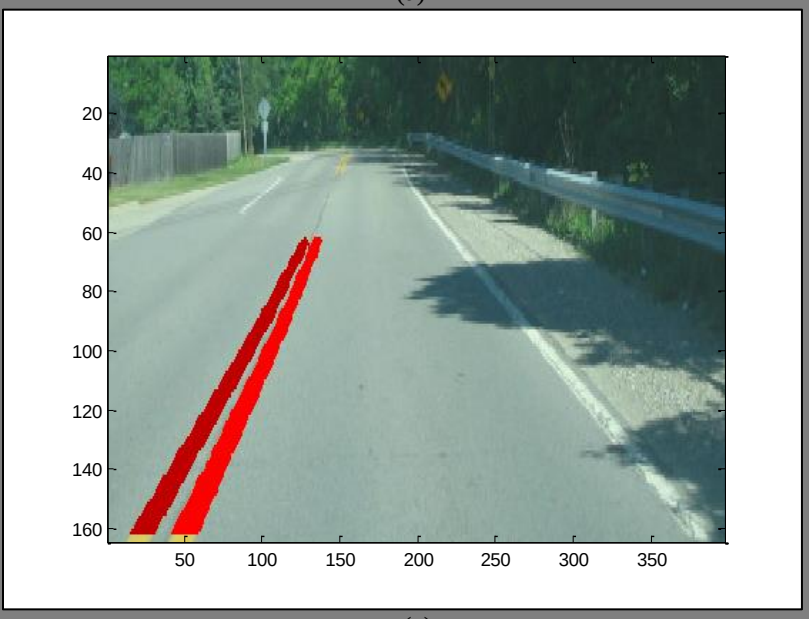
As may be seen in the previous images, using *only* the color information may not allow sufficient differentiation between yellow pixel values corresponding to lane markers and those unrelated to lane markers. Similarly, using only the color characteristics may not provide sufficient discriminating capability between yellow longitudinal lane markers in very close proximity to one another. However, using GEM values along with dimensional characteristics and those locations representing pixels with a high degree of color correspondence to yellow longitudinal lane markers may contribute to producing the distinct yellow marker segments shown in figure X. Note that the differentiation between the distinct yellow marker segments is indicated by superimposing different shades of red on each distinct marker segment.



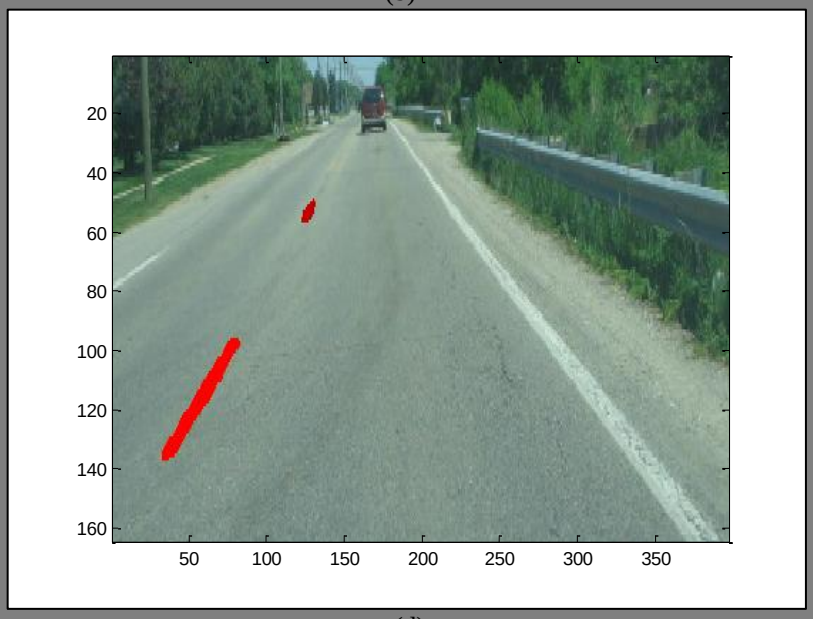
(a)



(b)



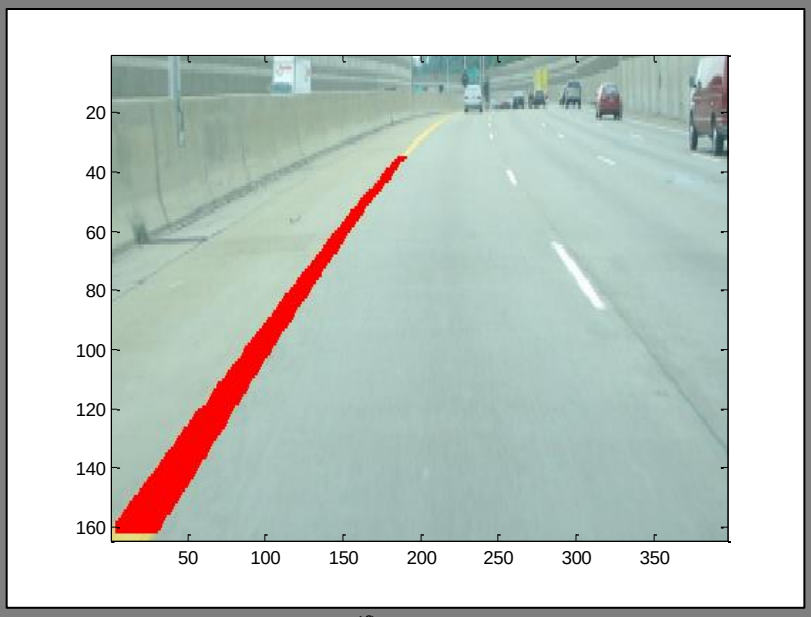
(c)



(d)

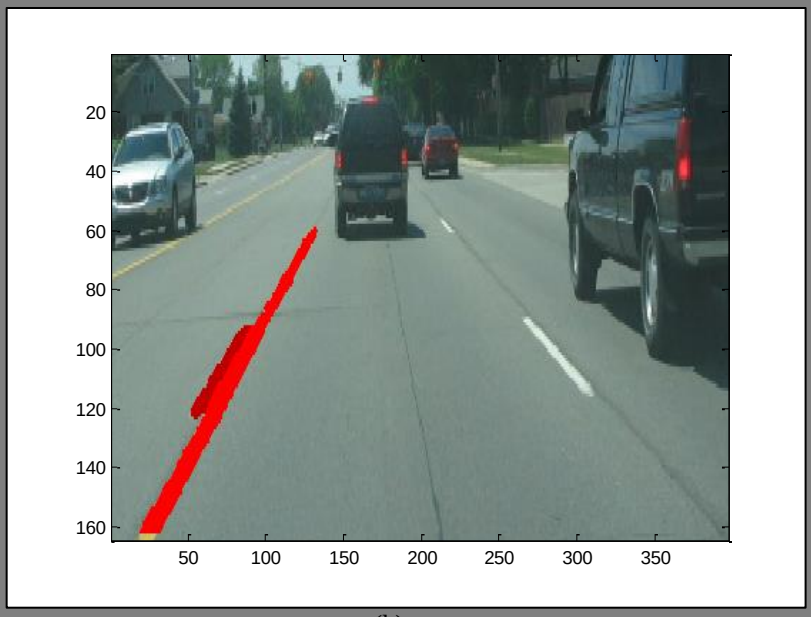
Figure X: Road scenes (a), (b), (c), (d) corresponding to figure X (a), (c), (e), and (g), respectively, showing distinct yellow longitudinal lane markers from analysis incorporating GEM values, color characteristics, and dimensional relationships

(e)



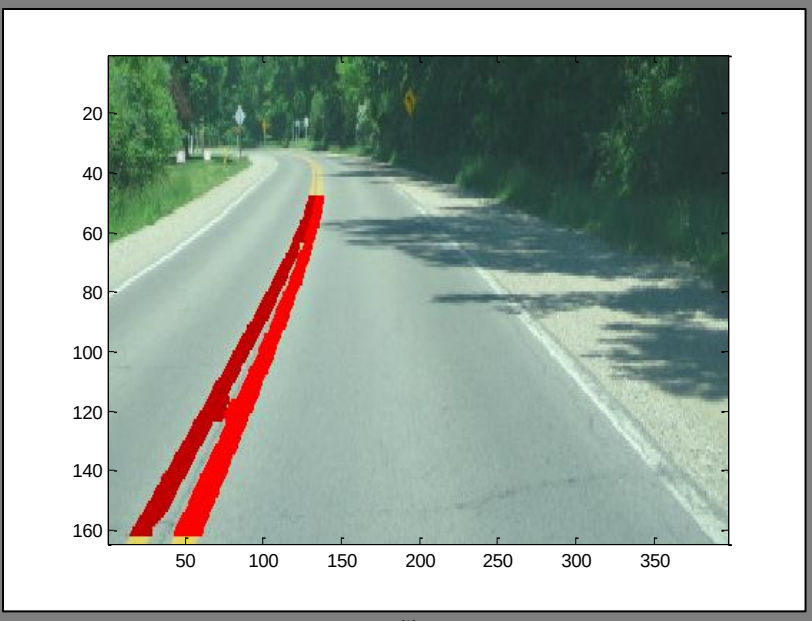
(f)

(g)



(h)

Figure X (continued): Road scenes (e), (f), (g), (h) corresponding to figure X (i), (k), (m), and (o), respectively, showing distinct yellow longitudinal lane markers from analysis incorporating GEM values, color characteristics, and dimensional relationships



(i)

(j)

Figure X (continued): Road scenes (i), (j), corresponding to figure X (q) and (s), respectively, showing distinct yellow longitudinal lane markers from analysis incorporating GEM values, color characteristics, and dimensional relationships

The edge operators, line operators, and first order gradient all provide edge information generated from the road scene image. Each operator has its unique characteristics and can be used to help find edges at specific angles. Color information may be used to further facilitate subsequent image processing tasks. However, once each distinct edge type has been detected, it must first be stored as part of a group to deduce further meaning from the complete set of edges. Thus, after passing the first level of discrimination, each distinct edge map was stored with a unique indicator in the global edge map, discussed in the next section of this paper.

7. THE GLOBAL EDGE MAP (GEM) CONCEPT, SEGMENTATION, AND CLASSIFICATION

7.1 Idea behind the Global Edge Map concept and generating an edge map

The global edge map (GEM) was required because the edge detection process would be generating many different light to dark and dark to light edge transitions for longitudinal markers of different slopes. The use of the word global is intended to convey that it is an edge map which will be constituted of (potentially) many different types of edges and would be 'comprehensive'. Thus, this single edge map could be used to represent the contributions from the positive sloped light to dark transitions, the positive sloped dark to light transitions, the negative sloped dark to light transitions, the negative sloped light to dark transitions, and potentially others. A portion of a hypothetical global edge map for the artificial image of figure X (a) might be represented as that shown in figure X (b).

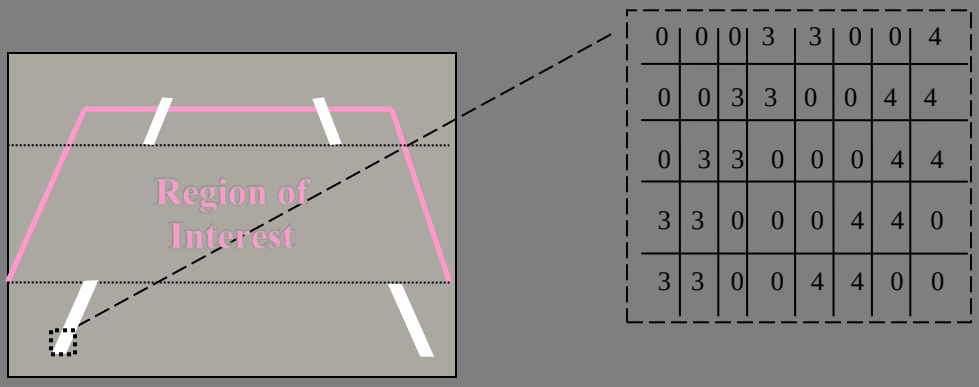


Figure X (a) Artificial road scene image with sample longitudinal lane makers in FOV; (b) sample portion of potential edge map

As may be seen from figure X, the positive sloped dark to light transition on the leftmost portion of the marker might be represented by 3's in the GEM while the positive sloped light to dark transition on the rightmost portion of the marker might be represented by 4's in the GEM. The values of three and four are used because the values of one and two are used to designate the negative sloped transitions which are detected prior to detecting the positive sloped edges.
The general flow for the process of construction of the GEM and further segmentation is given in figure X.

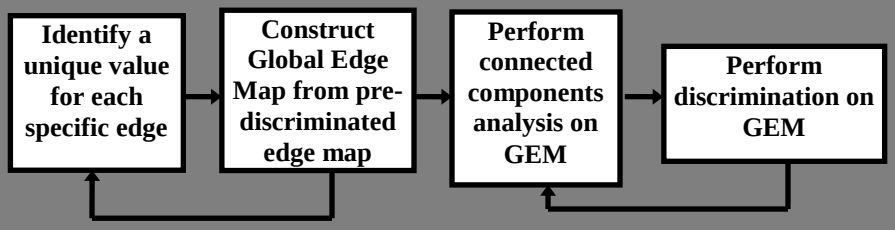


Figure X: High level flow leading to GEM discrimination

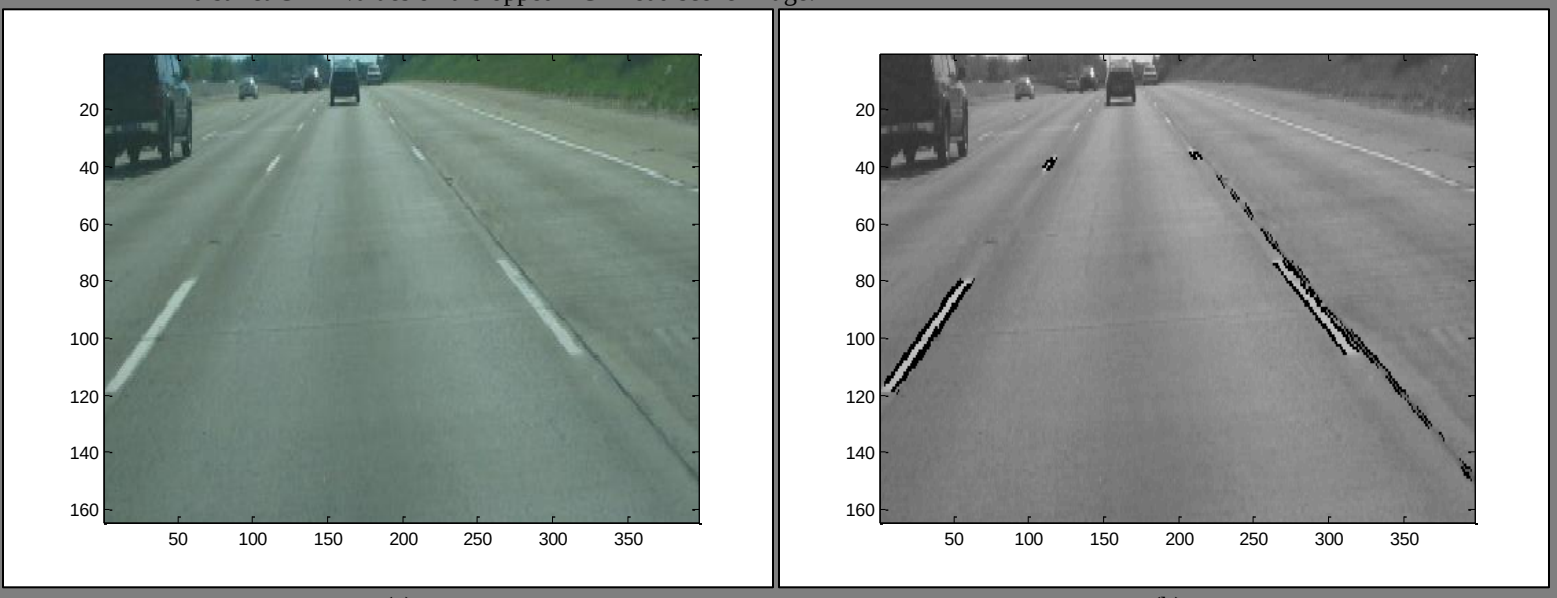
A unique method has been formulated to allow the GEM to hold more than one specific edge transition value within a single matrix element location. This is necessary because there may be pixels (or groups of pixels) within the image which contribute to more than just one edge transition. The code for the function which constructs the new GEM value from the previously existing value is shown in figure X.

```
function [global_image_map_matrix] = update_global_edge_map_matrix_array(temp_image_map, temp_global_image_map);
% Programmer: Christopher Alan Warner
% UPDATE_GLOBAL_EDGE_MAP_MATRIX_ARRAY is a function that will accept the current global edge map along with
% the newly acquired image map (to be added) and returns an updated global image map matrix .
% ================================================================
%     INPUT LIST (in order that they appear as parameter list):
% ================================================================
% 1) TEMP_IMAGE_MAP = the most recently (first level) discriminated data being added to the overall global edge map
% 2) TEMP_GLOBAL_IMAGE_MAP = the current (prior to addition of new map data) overall global edge map
% ================================================================
%     OUTPUT LIST (in order that they appear as parameter list):
% ================================================================
% 1) GLOBAL_IMAGE_MAP_MATRIX = the updated global image map matrix
% ================================================================
%     OTHER CUSTOM FUNCTIONS USED
% ================================================================
% 1) NONE

rows = []; cols = [];
[rows, cols] = size(temp_image_map);
global_image_map_matrix = zeros(rows,cols);

for j = 1:rows,
   for k = 1:cols
      if (temp_image_map(j,k) ~= 0)

         image_map_value_string = '';
         gem_string = '';
         temp_string = '';
         temp_num = [];

         if (temp_global_image_map(j,k) == 0)
            temp_global_image_map(j,k) = temp_image_map(j,k);
         else
            image_map_value_string = num2str(temp_image_map(j,k));
            gem_string = num2str(temp_global_image_map(j,k));
            temp_string = strcat(gem_string, image_map_value_string);
            temp_num = str2num(temp_string);
            temp_global_image_map(j,k) = temp_num;
         end;
      end;
   end;
end;

global_image_map_matrix = temp_global_image_map;
```

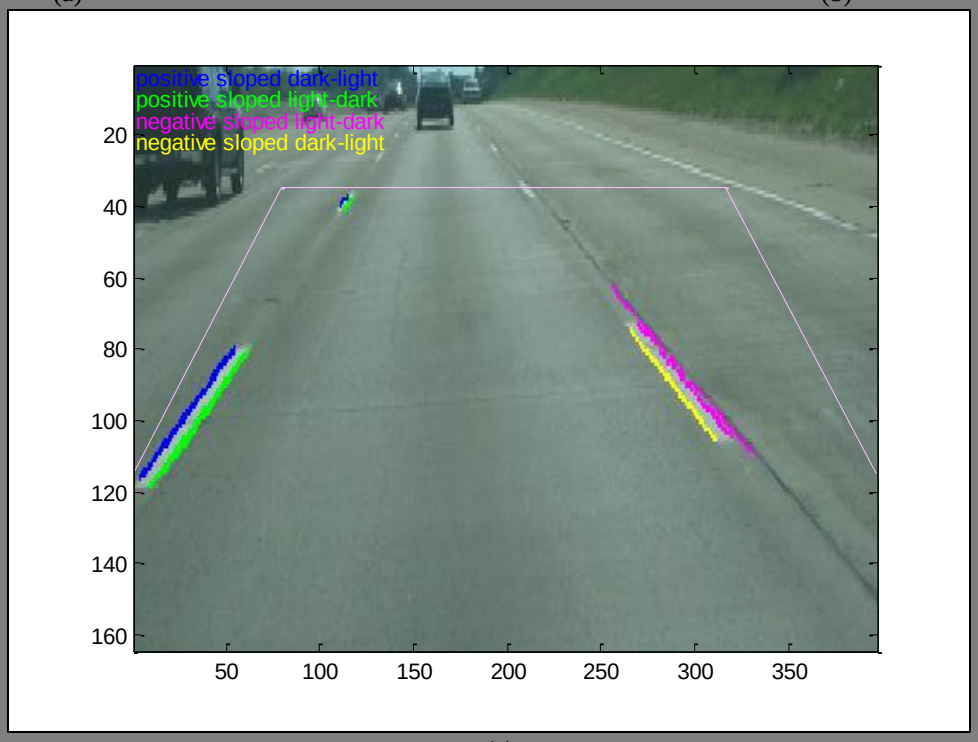
Figure X. Code listing for portion of function update_global_edge_map_matrix_array.m

As may be seen from the code listing in figure X, the specific GEM values from the individual edges are stored as numeric values within the matrix. Thus, a GEM value at a specific location may be 124 meaning that that specific location contains contributions from three different edges represented by the unique identifiers 1, 2, and 4. Once the undiscriminated values have been added to the GEM, analysis of the GEM proceeds. Figure X shows a road scene image

with edge values overlayed on a grayscale approximation as well as a color based visual distinguishment of the resulting distinct GEM values on a cropped RGB road scene image.



(a) (b)



(c)

Figure X: (a) RGB cropped road scene image; (b) grayscale approximation with edge values superimposed in black; (c) portion of global edge map values superimposed in color on RGB cropped road scene image

In figure X (c), the blue color indicates the location of the positive sloped dark to light edge transitions (or 3's) within the GEM, the green color indicates the location of the positive sloped light to dark edge transitions (or 4's) within the GEM, the magenta color indicates the location of the negative sloped light to dark edge transitions (or 1's) within the GEM while the yellow color indicates the location of the negative sloped dark to light edge transitions (or 2's) within the GEM.

7.2 Interconnection of the GEM using connected components on detected edges

There are various ways in which related values (for instance, 1's) within a map may be grouped or connected. Two methods are four connectedness and eight connectedness, the former of which is shown in figure X.

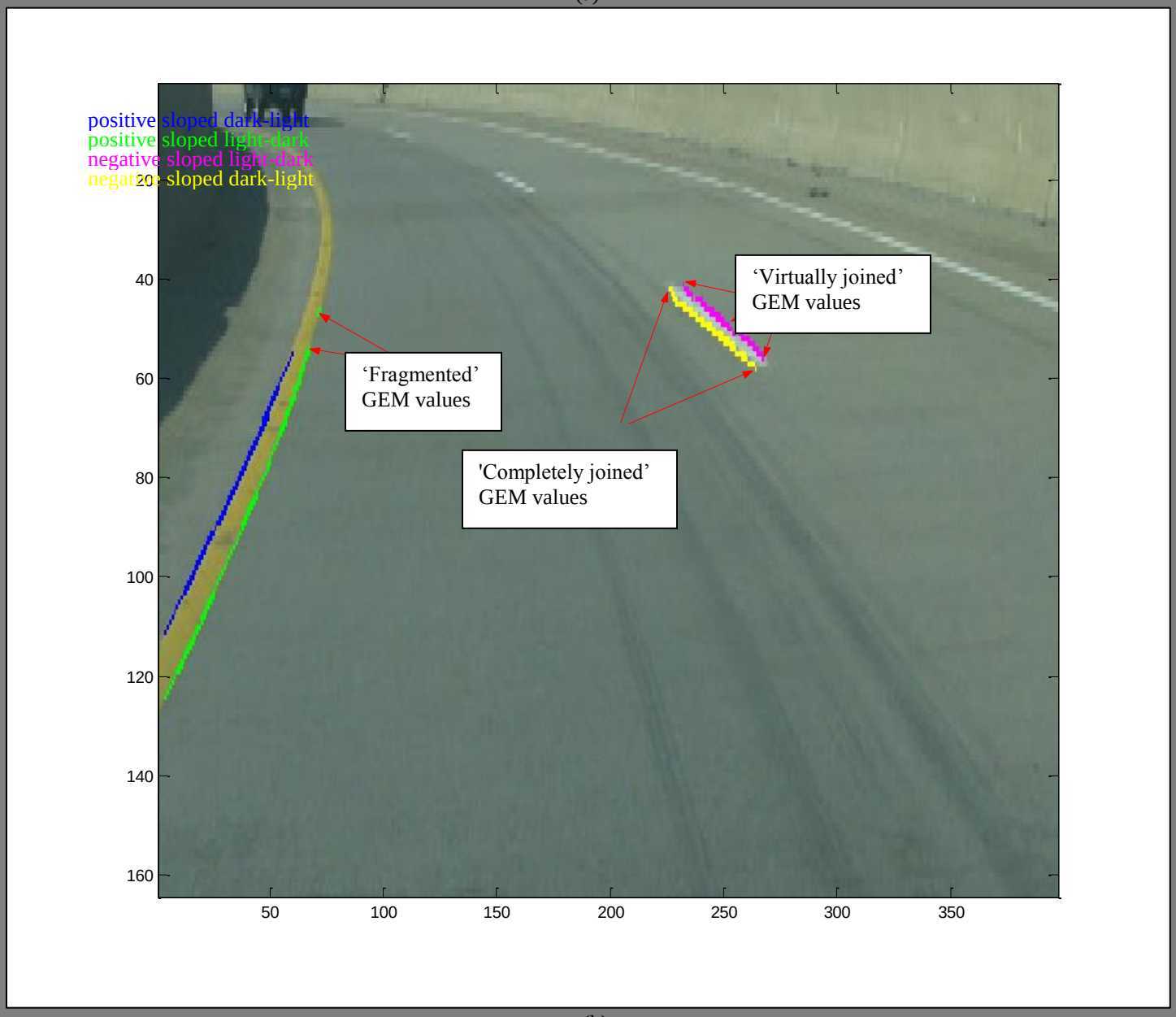
0	0	1	0	0
0	0	1	0	0
1	1	**1**	1	1
0	0	1	0	0
0	0	1	0	0

Figure X: Example of four connectedness

It is apparent that the pixel at the center location (3,3) exhibits four connectedness. Now consider the possible relationships within groups for the respective edge transitions (positive and negative sloped dark to light and light to dark) and how they may be connected or linked. As may be seen from figure X, there is an uninterrupted sequence of (blue) GEM values for the positive sloped dark to light edge transition near the prospective border between the white broken longitudinal lane marker and the substrate in the lower left corner of the image. However, there is yet to be a relationship described between the individual GEM elements other than the fact that they have met certain edge correlation requirements and have passed a preliminary level of discrimination. That is, there is no illustrated relationship between adjoining GEM values from one GEM location to the next and how the segment connected by the 'endpoints' may be evaluated more aptly for correspondence to a border between a lane marker and the substrate. Thus, the establishment of a segment to facilitate further discrimination methods is necessary. However, many factors contributing to a dispersal of related GEM edge components may prevent considering only those directly adjoining elements for interconnection. Thus, a customized connected components analysis may be required to maximize the effectiveness in linking the desired GEM values into segments. The road scene of figure X shows two different types of lane markers that have undergone the edge detection process. As may be seen in figure X, GEM values within an edge may be categorized in a number of different ways. A pair of GEM values may be considered 'completely joined', meaning that there is an uninterrupted sequence of adjoining relevant GEM values between the comparable pair. A pair of GEM values may be considered 'virtually joined', meaning that there exists at least one interruption to the sequence and this interruption includes up to two missing adjoining relevant GEM values between the comparable pair. A pair of GEM values may also be considered 'fragmented' if there exists at least one interruption to the sequence of more than two missing adjoining relevant GEM values between the comparable pair. It should be noted that there may exist individual 'completely joined' segments within a longer 'virtually joined' segment as well as within a longer 'fragmented' segment.

The completely joined GEM values will be the easiest to connect while the virtual GEM values will become more and more difficult to connect as adjoining interruptions become more prevalent and the question of which value to link next becomes more uncertain. Lowering of the thresholds is an option to reduce the dispersion of comparable GEM values within a particular lane marker transition. However, this also increases the likelihood that unintended transitions might be segmented. Thus it is desirable to have a method which will 'overlook' lesser degrees of GEM dispersion but will disconnect significantly disjoint GEM sequences.

(a)



(b)

Figure X: (a) Original cropped RGB image; (b) various types of 'interconnectedness'

Detected negative

A potential method may be preliminarily instituted with the knowledge that the prospective edges within the GEM being connected should typically have a slope in the vicinity of one. This method should also be able to connect the prospective edges resulting from the analysis of a road scene image with a vehicle positioned approximately centered between adjacent lanes (as shown in figure X) as well as prospective edges which may be encountered during lane maneuvers. The connection method should be able to compensate for movement to an inadvertent GEM neighbor as well as being able to skip to likely GEM locations in the case where no significant adjoining GEM values are found. Similarly, the method should be able to analyze local clusters of GEM values to assist with less certain traversal through highly dispersed relevant GEM regions. As an initial example of how one may traverse an edge map containing varying degrees of GEM value dispersion, the following sample arrangement was created to assist in explaining the preliminary generalized method (for a positive sloped prospective border sequence contained within the GEM).

M	**N**	**O**	**P**
I	**J**	**K**	**L**
E	**F**	**G**	**H**
A	**B**	**C**	**D**

(a)

1) Look to diagonal and if found, look to next diagonal (If A then F then K then P then ….).
2) If not found on diagonal, look sideward and also look upward (If not F then look at B and also look at E).
2a) If found both upward and sideward, look to specific elements within the next subsequent block for possible continuation points.
2b) If found only upward or only sideward, start looking at next diagonal from where it has been found (if found at E then look at J elseif found at B then look at G).

(b)

Figure X: (a) Sample search arrangement; (b) preliminary generalized method for connecting joining components

7.3 Preliminary GEM discrimination and classification

The starting and ending points (along with other pertinent prospective border information) may be determined for the ‘connected’ segment as elaborated on in the previous section. Although this segment has met certain correlation criteria and has passed a preliminary level of discrimination, it must still be compared to additional primitive and attribute information originating from the analysis of the reference patterns leading to structural conclusions. Thus, one goal is to determine whether the segment corresponds to a border between a lane marker and the substrate or a border not related to a lane marker. Typical non-lane marker transitions include (but are not limited to) those detected from road vehicles, portions of the horizon further into the FOV, edges on the road surface not originating from longitudinal lane markers, edges from objects that are not part of the road surface other than vehicles, and words or symbols which may appear as part of the local roadway traffic control devices.

The proposed primitives include (but are not limited to) the (existence of) transitions associated with the prospective borders occurring between a longitudinal lane marker and the road substrate. These prospective border transitions typically occur in complementary pairs and have attributes (structural characteristics) associated with them. Some of the relevant factors involved in primitive selection have already been elaborated on.

Up to this point, no explicit distinction has been elaborated on regarding the structural characteristics for a longitudinal lane marker pattern without a gap (normal, wide, double, etc.) and those longitudinal lane marker patterns containing a gap (broken, dotted, etc.), except for the fact that there exists certain distance criteria associated with each. This may include distance information related to the marker segment, the marker gap, the marker segment to gap ratios, etc. However, consider the fact that distance criteria are highly dependent on factors including location within the FOV, origination location both within and outside the FOV, road geometry, regulatory requirements, degradation, image noise and distortion, occlusion, application process, etc. Similarly, no formal explanation has been given regarding how the structural attributes (numerical features) are chosen and quantified during the selection and recognition phases nor have specific statistical pattern descriptions been made explicit. However, a proposed framework has been established based upon factors including the characteristics of the hybrid method chosen, the preliminary regulatory information covered, various images highlighting some important perspective relationships for parallel longitudinal lane markers for roads with different geometric properties, the chrominance and luminance aspects of a road scene image, the GEM structure selected and methodologies incorporated in facilitating the pattern classification process, etc.

Through a rigorous analysis of the problem domain and from the initial goals established for the preliminary road scene image processing system, a structural recognition method was chosen which attempts to establish a structural representation through the discrimination of unlikely pattern candidates from the set of those occurring within a road scene image. From the set of undiscriminated structural features remaining in the GEM along with flags set or cleared during the intermediate structural recognition stages, certain structural conclusions may be inferred. Thus, an overall structural analysis may encompass factors including (but not limited to):

1. The existence of a reference transition with respect to the complementary transition;
2. The relevant structural positioning of the reference transition to the complementary transition;
3. Distance measurements and comparisons;
4. Reference and complementary transition ‘strengths’;
5. Comparable slopes existing between reference and complementary transitions;
6. Comparable slopes falling within an established range;
7. Reference and complementary transition occurrence total;
8. Limited occurrences of contradictory GEM values within the locale;
9. Perspective projection characteristics;
10. Reference and complementary transitions meeting certain statistical requirements;
11. Reference and complementary transitions not exceeding certain statistical limits;
12. Map based regional proximity relationships.

Thus, during GEM discrimination, the segment designating a single prospective transition must be evaluated for its relationships to the structural conclusions derived from the reference longitudinal lane marker(s). Depending on the degree of correspondence to the reference primitives and attributes, a pattern conclusion may be formulated.

If a segmented edge transition contained within a pair of points is discriminated, the connected components (and associated transition locations) between the points will need to be added to and removed from certain variables. This is a task which requires special attention because the values in the GEM corresponding to lane marker edge transitions are stored as numeric values potentially containing the numerals 1, 2, 3, and 4. Say for example a positive sloped dark to light edge transition represented by a GEM value of ‘3’ is to be removed but the value within that specific

matrix location within the GEM also contains the values ‘1’ and ‘2’ as contributions from negative sloped edge transitions. To add and remove unique values from the specific locations, the numeric value containing the numerals 1, 2, and 3 must first be converted to a string and then the value being added is added as a string element (character) or the value being removed is removed as a string element (character). The modified string must then be converted back to a numeric value for storage in the GEM. A portion of the function for removing a specific value from a GEM location is given in the code segment of figure X.

```
% Programmer: Christopher Alan Warner
% REMOVE_CURRENTLY_UNUSABLE_DATA_FROM_GEM is a function that will serve to look for certain edge values  in temp_edge_map (also temp_edge_map in calling environment) and remove them from disc_gem (also disc_gem in calling environment).
% This function is typically found following add_currently_unusable_data_to_current_gem as a means of removing the
% information from disc_gem after it has been added to temp_edge_map. The removal is performed automatically so that the return  to the calling environment (disp_image) for perform_disc_gem returns an image with the just processed information removed.
% Note that this function may be called to operate on single or multiple rows of information depending on the values found in starting_row and ending_row
% ================================================================
%     INPUT LIST (in order that they appear as parameter list):
% ================================================================
% 1) DISC_GEM = contains the complete edge map having relevant edge values removed from
% 2) TEMP_EDGE_MAP = contains the edge map being looked to for relevant values to be removed from disc_gem
% 3) STARTING_ROW = contains the first row value of the block (1 x N for single row) being processed
% 4) STARTING_COL = contains the first column value of the block being processed
% 5) ENDING_ROW = contains the last row value of the block being processed
% 6) ENDING_COL = contains the value (starting_col + col_offset) => the final column value
% 7) SLOPE_INDICATOR = contains the value indicating which specific edge transitions (specific values) are being looked for
% ================================================================
%     OUTPUT LIST (in order that they appear as parameter list):
% ================================================================
% 1) UPDATED_DISC_GEM = the updated edge map with values from temp_edge_map having been removed
% ================================================================
%     OTHER CUSTOM FUNCTIONS USED
% ================================================================
% 1) NONE

function [updated_disc_gem] = remove_currently_unusable_data_from_gem(disc_gem, temp_edge_map,
starting_row,starting_col,ending_row,ending_col,slope_indicator)

rows = []; cols = [];
[rows, cols] = size(disc_gem);              % Get # rows and cols from disc_gem
updated_disc_gem = zeros(rows,cols);
```

Figure X: Code listing portion for function remove_currently_unusable_data_from_gem.m

```
if (slope_indicator == 3)
   for x = starting_row:ending_row        % Note that starting_row will be the smaller of the two and ending_row will be the
      for y = starting_col:ending_col     % Note that starting_col will be the smaller of the two and ending_col will be the
% If temp_edge_map has significant data in specified position that needs to be removed from specific position in disc_gem
         if (temp_edge_map(x,y) == 3 | temp_edge_map(x,y) == 13 | temp_edge_map(x,y) == 31 | temp_edge_map(x,y) ==
            23 | temp_edge_map(x,y) == 32 | temp_edge_map(x,y) == 43 | temp_edge_map(x,y) == 34 |
            temp_edge_map(x,y) == 132 | temp_edge_map(x,y) == 123 | temp_edge_map(x,y) == 213 |
            temp_edge_map(x,y) == 231 | temp_edge_map(x,y) == 312 | temp_edge_map(x,y) == 321 |
            temp_edge_map(x,y) == 143 | temp_edge_map(x,y) == 134 | temp_edge_map(x,y) == 314 |
            temp_edge_map(x,y) == 341 | temp_edge_map(x,y) == 413 | temp_edge_map(x,y) == 431 |
            temp_edge_map(x,y) == 1432 | temp_edge_map(x,y) == 1423 | temp_edge_map(x,y) == 1243 |
            temp_edge_map(x,y) == 1234 | temp_edge_map(x,y) == 1342 | temp_edge_map(x,y) == 1324 |
            temp_edge_map(x,y) == 2134 | temp_edge_map(x,y) == 2143 | temp_edge_map(x,y) == 2413 |
            temp_edge_map(x,y) == 2431 | temp_edge_map(x,y) == 2314 | temp_edge_map(x,y) == 2341 |
            temp_edge_map(x,y) == 3142 | temp_edge_map(x,y) == 3124 | temp_edge_map(x,y) == 3214 |
            temp_edge_map(x,y) == 3241 | temp_edge_map(x,y) == 3412 | temp_edge_map(x,y) == 3421 |
            temp_edge_map(x,y) == 4132 | temp_edge_map(x,y) == 4123 | temp_edge_map(x,y) == 4213 |
            temp_edge_map(x,y) == 4231 | temp_edge_map(x,y) == 4312 | temp_edge_map(x,y) == 4321 |
            temp_edge_map(x,y) == 243 | temp_edge_map(x,y) == 234 | temp_edge_map(x,y) == 324 |
            temp_edge_map(x,y) == 342 | temp_edge_map(x,y) == 423 | temp_edge_map(x,y) == 432)

            string_from_num = '';                  % Initialize to null string
            length_string = [];
            new_string = '';                       % Initialize new string being created

            string_from_num = num2str(disc_gem(x,y));   % Get information from that position in disc_gem into string
            length_string = length(string_from_num);    % Get length of string
            for z = 1:length_string                % For each of elements in string
               if string_from_num(z) ~= '3'            % If that string element IS NOT being removed
                  new_string = strcat(new_string,string_from_num(z)); % Formulate new string with each element
               end;
            end;
            if (isempty(new_string))
               new_string = '0';
            end;
            disc_gem(x,y) = str2num(new_string);        % Store new string (as number) without removed in disc_gem
         end;
      end;
   end;



updated_disc_gem = disc_gem;
```

Figure X (continued): Code listing portion for function remove_currently_unusable_data_from_gem.m

Figure X shows the output from various stages of edge detection, connection, and GEM discrimination.

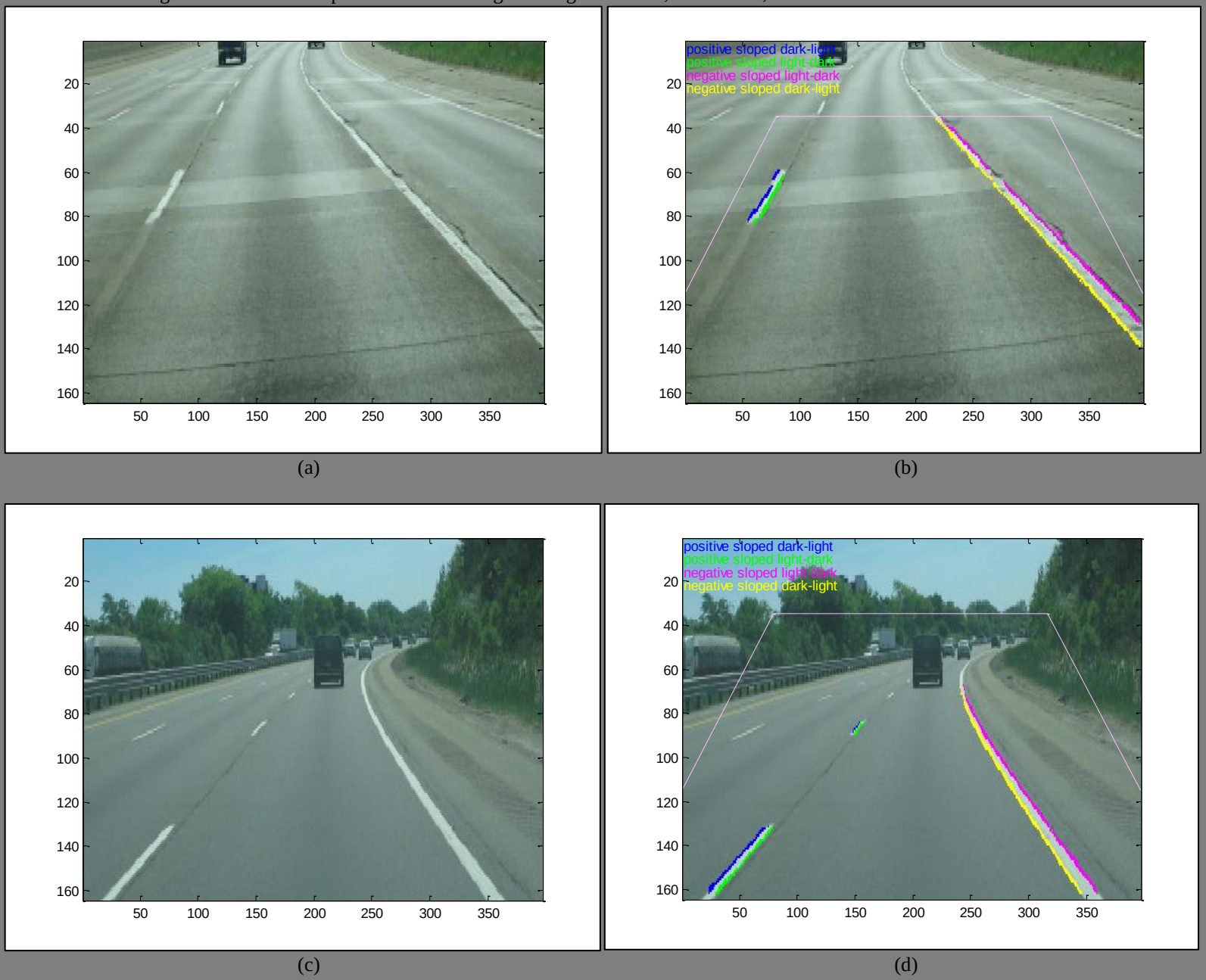


Figure X: (a) Original cropped RGB image; (b) preliminary results of detection, connection, and GEM discrimination; (c) original cropped RGB image; (d) preliminary results of detection, connection, and GEM discrimination

Figures X(b) and (d) show the preliminary results of the edge detection, connection, and GEM discrimination process with the positive sloped dark to light and light to dark edge transitions and negative sloped dark to light and light to dark edge transitions highlighted by the blue, green, yellow, and magenta pixels, respectively. The color coding legend is also shown in the upper left hand corner of the RGB images. Figure X shows the results of the edge detection, connection, and GEM discrimination process with raised thresholds that may be used to eliminate many of the false edge transition possibilities that need to be analyzed in figure X. It can be seen that the lane markers nearest the front of the vehicle have edge transitions passing the GEM discrimination process. However, the broken white lane marker further into the FOV has now seen its edge transitions go undetected.

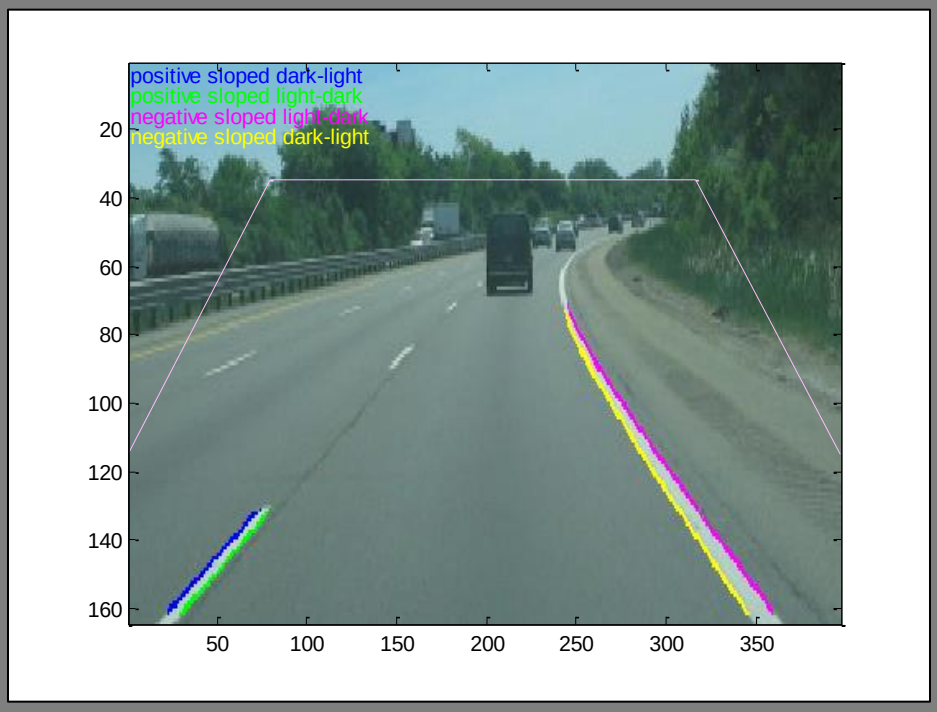


Figure X: Preliminary results of detection, connection, and GEM discrimination with raised thresholds for road scene image of figure X (c)

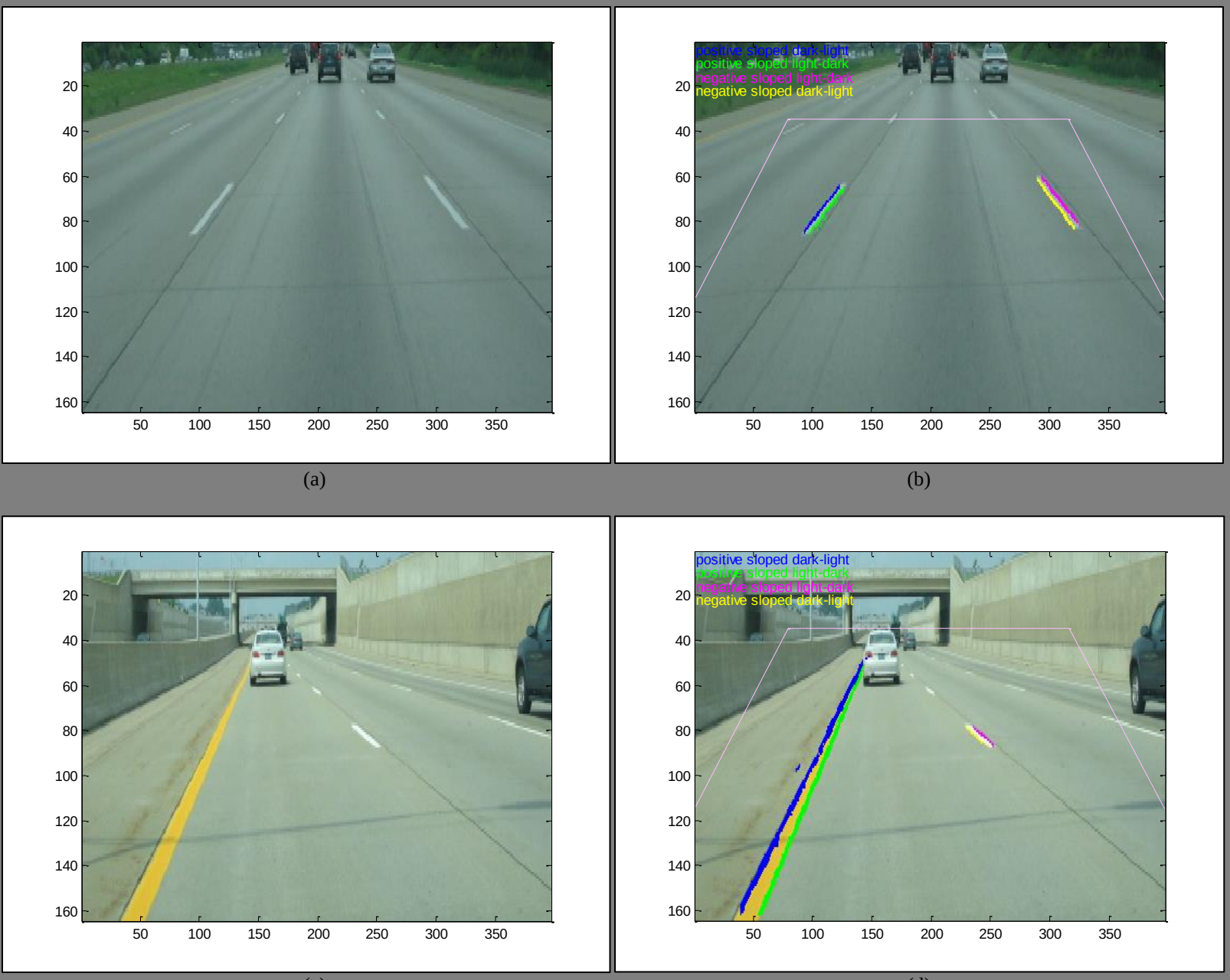
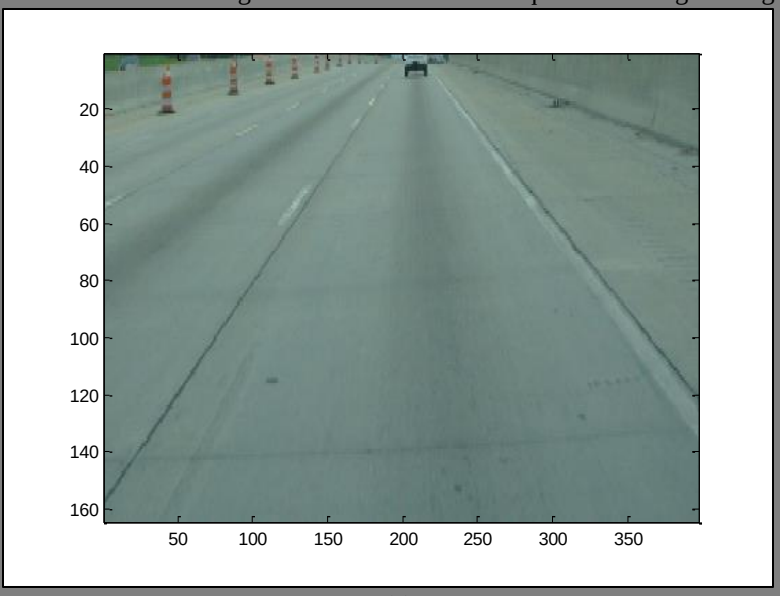
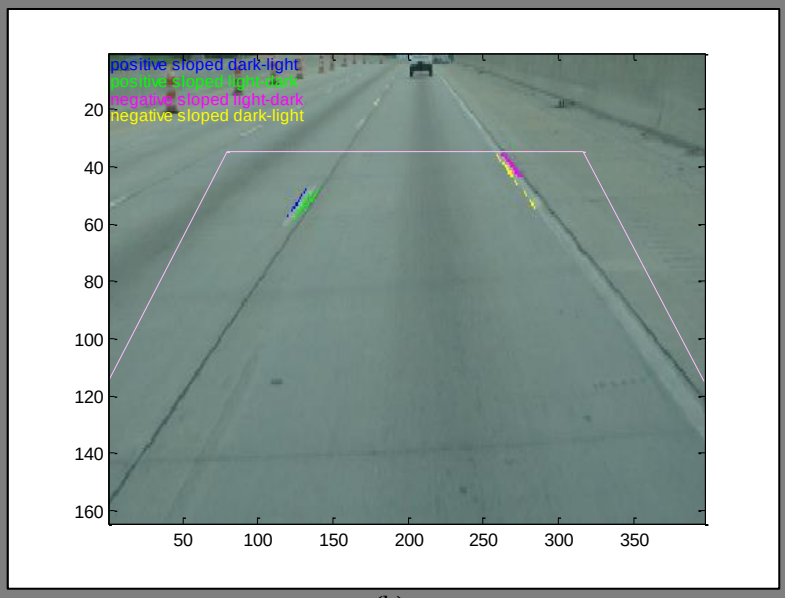


Figure X: (a) Cropped RGB image; (b) preliminary results of detection, connection, and GEM discrimination; (c) cropped RGB image; (d) preliminary results of detection, connection, and GEM discrimination

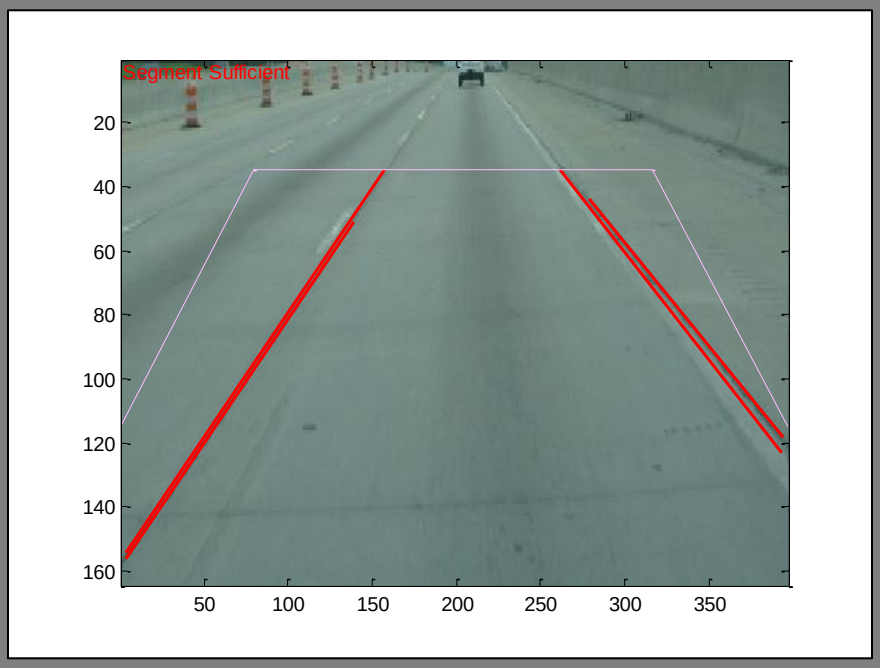
There exists another array which contains the start and end points for all edge transitions which have met certain reference edge transition requirements but do not have ‘sufficient’ complementary edge transitions. It is very important to note that this edge information may or may not contain useful road boundary information. For example, various traffic barriers (longitudinal, bridge railings, etc.) may generate a significant transition along the line (or in close proximity to) where the barrier comes in contact with the roadway. Similarly, road edges without explicit boundary markers, roadway lane seams, and curbs may produce potentially useful boundary information. Also consider the situation in which an actual white longitudinal lane marker includes one side with a distinguishable transition while the other side (the complementary transition) ‘blends into’ the substrate preventing a complementary edge transition from being detected. Consider the sequence of images in figure X.



(a)

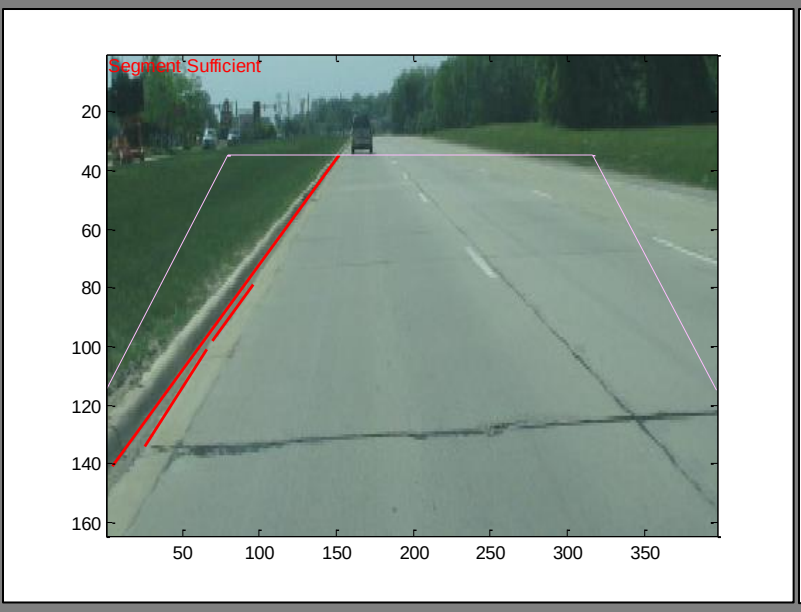


(b)

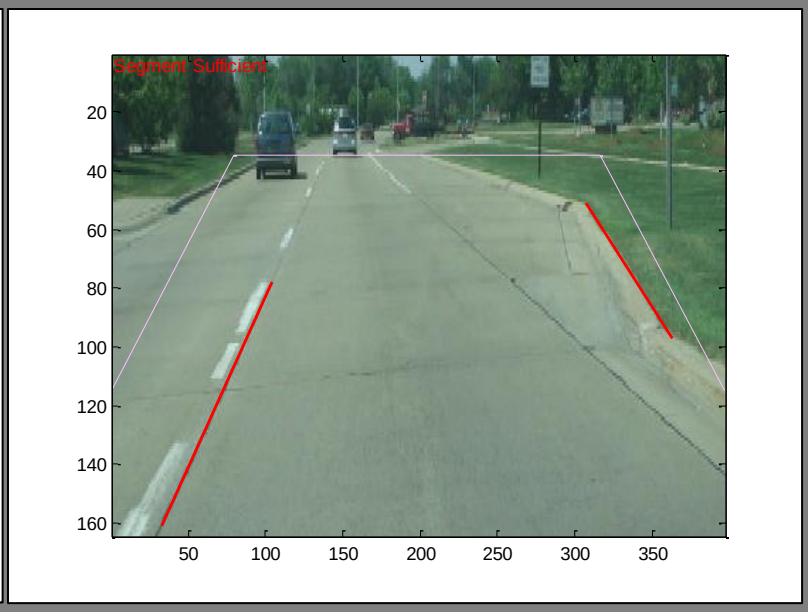


(c)

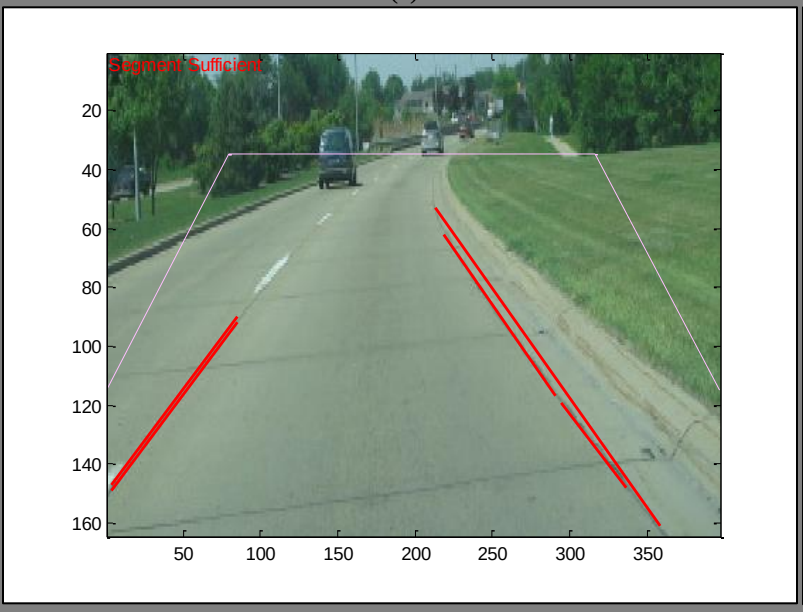
Figure X: (a) Cropped RGB road scene image with (b) preliminary results of detection, connection, and GEM discrimination; (c) Segments of sufficient length (shown in red) which provide potentially useful boundary information



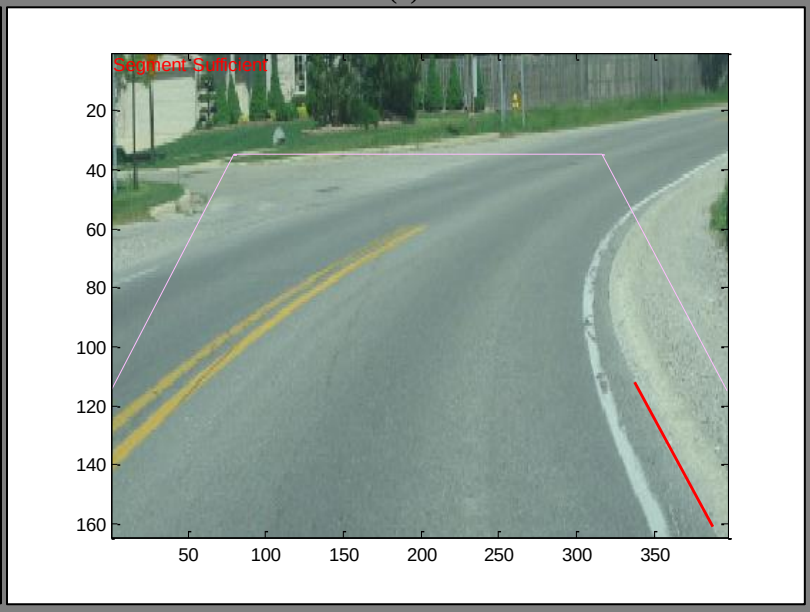
(a)



(b)



(c)



(d)

Figure X: (a) – (d) Segments of sufficient length (shown in red) which provide potentially useful boundary information

Figure X: Segments of sufficient length (shown in red) which provide potentially useful boundary information

The first level of discrimination was initiated by a call to function perform_discrimination.m, which discriminated those potential edge points if they were outside the region of interest, didn't meet correlation requirements, or didn't meet various other criteria. The next level of discrimination was based upon factors including (but not limited to) those shown in figure X on page XX and discussed further on page XX.

As may be seen in some of the previous road scene images, the shade originating from trees, vehicles, tunnels, or other road objects is a situation which is commonly incurred during the capture of road scene images. The potential lack (or overabundance) of illumination as a result of driving at night is another of the many existing illumination conditions which may prove troublesome for unspecialized camera systems. These natural and artificial illumination conditions potentially affect how the image processing is managed. Thus, systems which provide maximum resolution for the existing illumination conditions need to be considered when performing road scene analysis.

7.4 Justification for high dynamic range camera (HDRC) systems

How an imaging system deals with predominantly dark (or bright) conditions as part of the image acquisition process may play a very significant role in the effectiveness of the system. The dynamic range of a high dynamic range camera (HDRC) system must be considered when choosing road scene image processing hardware. A sensors dynamic range may be defined as [2]:

> "the ratio of maximum light intensity required to saturate a pixel to light intensity which produces an output equal to noise output".

It is apparent from many of the previous figures that the illumination supplied by the artificial and natural sources combine to provide sufficient lighting to allow the preliminary road scene image processing software to amply detect the likely longitudinal lane boundaries. Under certain circumstances, it may be seen that due to the lack (or overabundance) of illumination within the road scene (dimly lighted tunnels, extremely bright sunlight, etc.), there are insufficient edges detected to extract likely lane boundary information. It might be possible to lower the existing thresholds to a point where the correct lane marker edges are detected but it has also been previously noted that drastically lowered thresholds can have a number of detrimental effects including (but not limited to) increasing the processing time required (due to the increased number of potential edges which need to be discriminated) as well as increasing the possibility that false edges go undiscriminated.

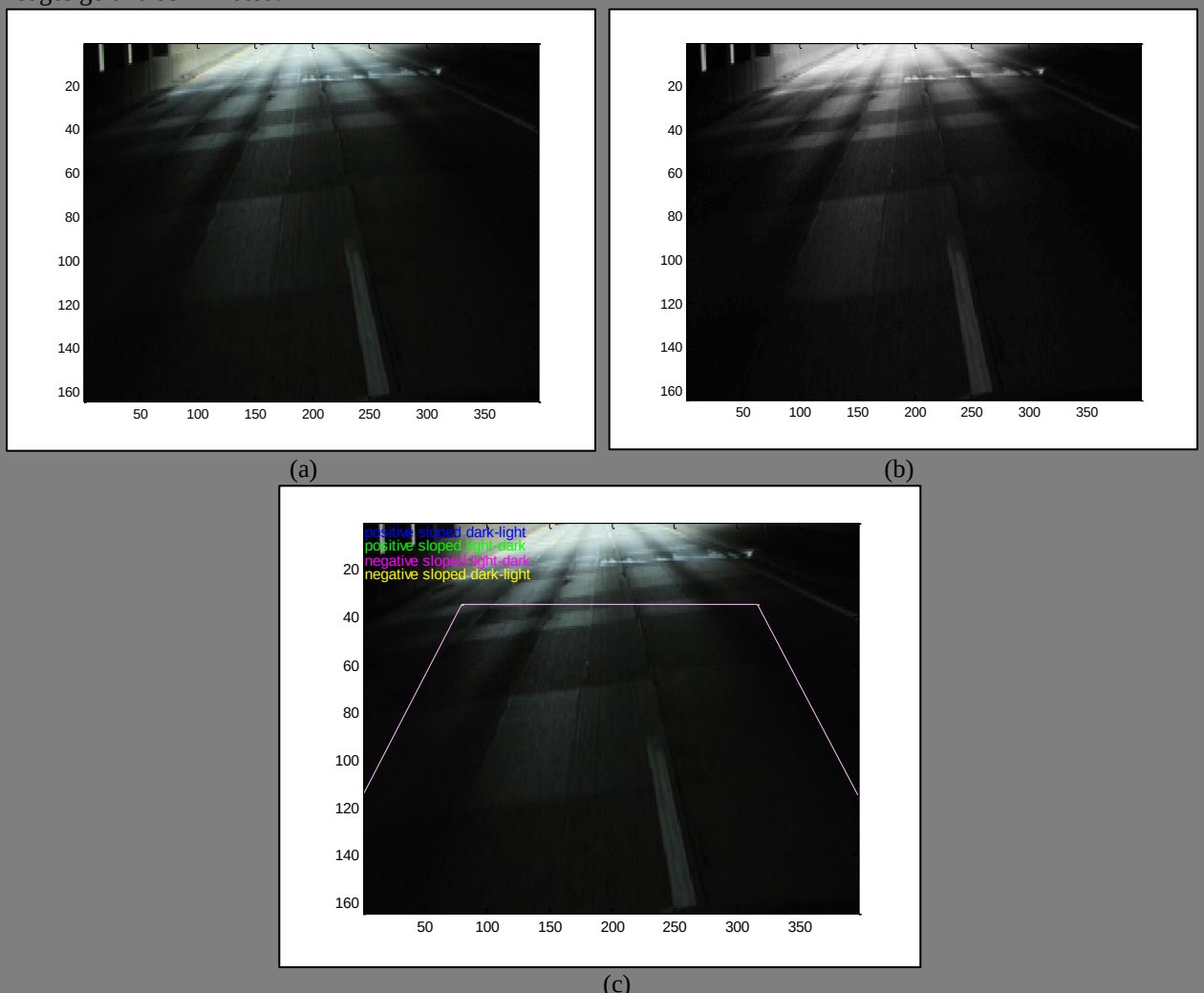
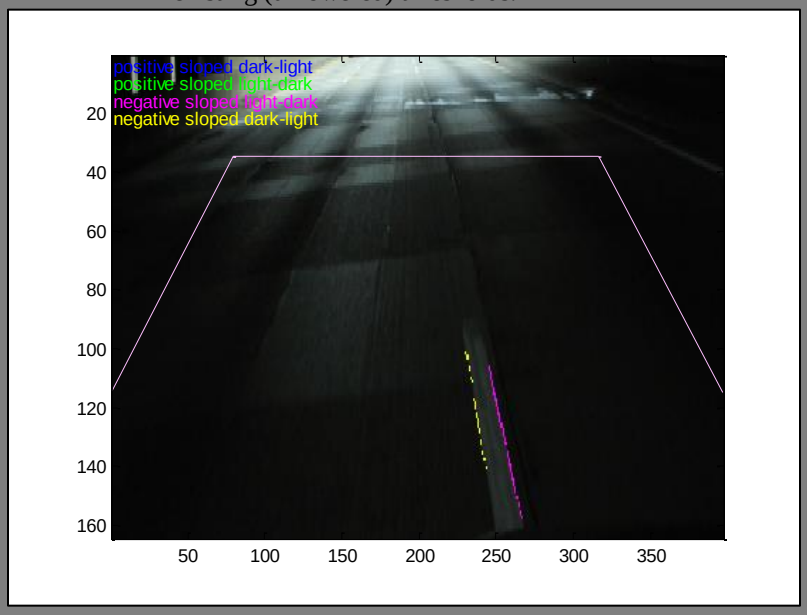
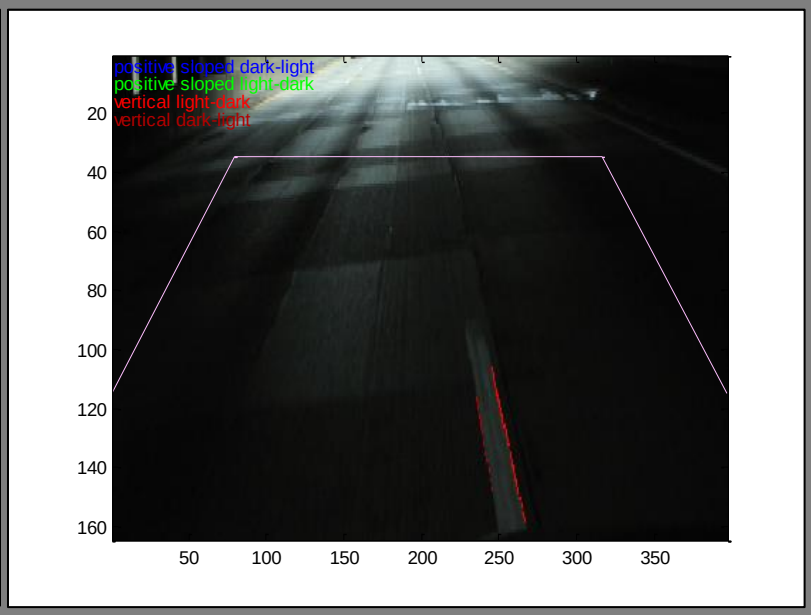


Figure X: (a) Original RGB cropped image; (b) grayscale approximation; (c) preliminary results of detection, connection, and GEM discrimination with existing thresholds using the generalized diagonal detector

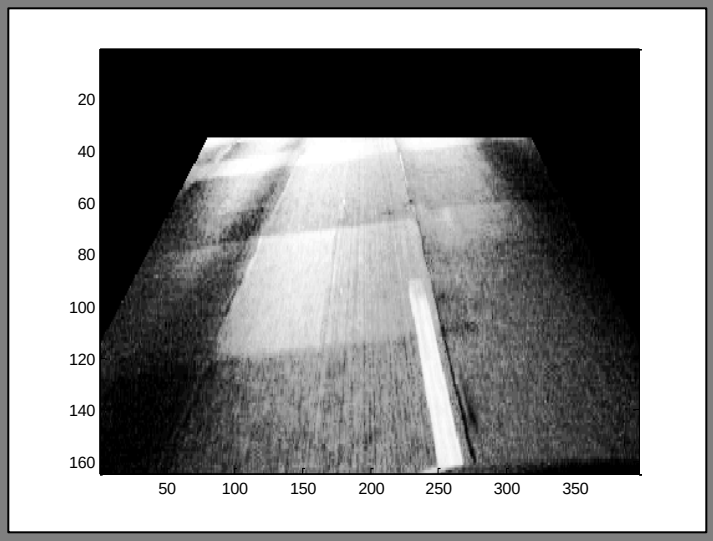
Note figure X in which the existing thresholds and use of the generalized edge detector have not detected the lane marker transitions present in the tunnel. Nonetheless, figure X shows the effect of drastically lowering the thresholds for figure X using the generalized edge detector along with the preliminary results of using the vertical edge detector with the existing (unlowered) thresholds.



(a)



(b)



(c)

Figure X: (a) Preliminary results of detection, connection, and GEM discrimination with drastically reduced thresholds using the generalized diagonal detector; (b) preliminary results of detection, connection, and GEM discrimination with existing thresholds using vertical edge detector; (c) contrast enhanced grayscale approximation within region of interest

Although the demand for HDRC systems is increasing for military and certain commercial applications, the technology still has quite a ways to go before it is financially feasible for most large scale production volumes. Unless a vehicle is nearly centered between adjacent lanes, the road scene image processing system has the potential to detect the likely lane markers to the left and the right sides of the vehicle. When lane markers on each side of the vehicle are reliably detected, relationships between the two may be used to interpret further information from the road scene image.

8. CONTEXTUAL ANALYSIS

8.1 Calculating slope approximations, slope averages, and generating line approximations

At this point, the GEM contains the border transitions which have exhibited a high level of correspondence to those occurring between longitudinal lane markers and the road substrate. Start and stop points for these GEM segments have been determined but there is yet to be a formalized representation established for these segments. Formal representations are necessary because they provide the foundation for:

1. Establishing a relationship between a border and the rest of a road scene image;
2. Establishing relationships between the individual borders within a road scene image.

Thus, generating line approximations for likely positive and likely negative sloped transitions allows further content based context to be established. Since the point-slope equation for a line requires both relevant points and a slope value, calculation of these quantities from the existing GEM constructed segment(s) is necessary. Figure X shows a portion of the code segment that performs the slope calculations (for a connected segment that has passed various levels of discrimination).

```
length_region_a_array = [];
length_region_a_array = max(find(region_a_start_end_point_array));  % Determine number of elements in start-end

slope_approx = zeros(length_region_a_array/4,1);                % Initialize array to store slope approximations
length_approx = zeros(length_region_a_array/4,1);               % Initialize array to store length approximations
slope_approx_counter = 1;                                       % Initialize indice into slope approximation array
length_approx_counter = 1;
upper_limit = 0;
upper_limit = length_region_a_array/2;

approx_num = 0;
approx_den = 0;
approximate_slope = 0;

if (~isempty(length_region_a_array))
   for a = 1:2:upper_limit                  % a index will be used to reference specific locations within arrays
      interim_a = [];
      if (a == 1)                           % first time through loop
         interim_a = 1;                     % interim_a points to first location
      else
         interim_a = (2*a)-1;               % else, interim_a points to (2*a) - 1 location
      end;

      starting_point_line = [];   ending_point_line = [];
      starting_point_line = [region_a_start_end_point_array(interim_a) region_a_start_end_point_array((2*a))];
      ending_point_line = [region_a_start_end_point_array((2*a)+1) region_a_start_end_point_array(2*(a+1))];

% *** FUNCTION APPROX_SLOPE WILL TAKE POINT INFORMATION FOR THE SEGMENT AND GENERATE
APPROXIMATIONS FOR THE SLOPE AND IT'S SPECIFIC COMPONENTS (NUMERATOR AND DENOMINATOR)
      approx_num = 0;          approx_den = 0;
      approximate_slope = 0;

      [approx_num approx_den approximate_slope] = approx_slope(starting_point_line, ending_point_line);

      point1 = []; point2 = [];
      point1 = [starting_point_line(2) starting_point_line(1)];
      point2 = [ending_point_line(2) ending_point_line(1)];
      segment_approx_length = 0;
      segment_approx_length = approx_length(point1, point2);


% *** CALCULATED SLOPE APPROXIMATIONS FROM ABOVE NOW STORED IN SLOPE APPROXIMATION
        length_approx(length_approx_counter) = segment_approx_length;
        length_approx_counter = length_approx_counter + 1;
```

Figure X: Code listing portion for calculating slope approximations on edges passing discrimination

```
        if (approx_num ~= 0 && ~isnan(approx_num) && approx_den ~= 0 && ~isnan(approx_den)) %
            slope_approx(slope_approx_counter) = -(approximate_slope);
            interesting_images_slopes(slope_approx_counter,temp_image_counter) = -(approximate_slope); %
        else
            slope_approx(slope_approx_counter) = Nan;
            disp('approx_num ~= 0 && ~isnan(approx_num) && approx_den ~= 0 && ~isnan(approx_den)) &&
                  angle_falls_within_grad_limits_flag');
            msgbox('approx_num ~= 0 && ~isnan(approx_num) && approx_den ~= 0 && ~isnan(approx_den)) &&
                  angle_falls_within_grad_limits_flag','ERROR','error');
            pause;
        end;

        slope_approx_counter = slope_approx_counter + 1;              % Increment index into slope_approx array

        starting_point_handle_line = [];    ending_point_handle_line = [];  handle_line_red = 0.0;
        starting_point_handle_line = [region_a_start_end_point_array(2*(a+1)) region_a_start_end_point_array(2*a)];
        ending_point_handle_line = [region_a_start_end_point_array((2*a)+1) region_a_start_end_point_array((2*a)-1)];
        handle_line_red = line(starting_point_handle_line,ending_point_handle_line,'color','r','linewidth',2);
    end;
end;
```

Figure X (continued): Code listing portion for calculating slope approximations on edges passing discrimination

The function of figure X calls the function approx_slope.m, shown in figure X, which calculates the approximate slope from the two endpoints.

```
% Programmer: Christopher Alan Warner
% APPROX_SLOPE is a function that will take two points passed to the function and then generate an approximate
% slope for the line which would be connected by the two endpoints [row1, col1]  and [row2, col2].
% ================================================================
%     INPUT LIST (in order that they appear as parameter list):
% ================================================================
% 1) POINT1 = contains the first point being used in the slope approximation calculation
% 2) POINT2 = contains the second point being used in the slope approximation calculation
% ================================================================
%     OUTPUT LIST (in order that they appear as parameter list):
% ================================================================
% 1) VERTICAL_DISTANCE = the difference between the two row values corresponding to the endpoints of the line
segment
% 2) HORIZONTAL_DISTANCE = the difference between the two column values corresponding to the endpoints of the line
segment
% 3) SEGMENT_APPROX_SLOPE = the vertical_distance divided by the horizontal_distance
% ================================================================
%     OTHER CUSTOM FUNCTIONS USED
% ================================================================
% 1) NONE

function [vertical_distance, horizontal_distance, segment_approx_slope] = approx_slope(point1, point2);

vertical_distance = 0; horizontal_distance = 0; segment_approx_slope = [];

vertical_distance = point1(1) - point2(1);
horizontal_distance = point1(2) - point2(2);

if (vertical_distance ~= 0)
   if (horizontal_distance ~= 0)
      segment_approx_slope = (vertical_distance/horizontal_distance);
   elseif (horizontal_distance ==  0)
      segment_approx_slope = inf;
   end;
else
   if (horizontal_distance == 0)
      segment_approx_slope = nan;
   end;
end;
```

Figure X: Code listing portion for function approx_slope.m finding the approximate slope of a line with two endpoints

Once each image has been analyzed for its positive and negative sloped dark to light and light to dark edge transitions (with associated starting and ending points) and each of these edges has passed the various levels of discrimination, it may be useful to calculate both an average slope resulting from the set of positive sloped edge transitions and an average slope resulting from the set of negative sloped edge transitions. This may be performed as shown in figure X by the function approx_slope_averaging.m.

```
% APPROX_SLOPE_AVERAGING is a function that will accept an array containing
% the slope approximations for certain specific edge transitions along with
% a group of indicators for the specific edge transitions and calculate an
% average approximate slope.
% ================================================================
%     INPUT LIST (in order that they appear as parameter list):
% ================================================================
% 1) ORIGINAL_REGION_A_START_END_VALUE_ARRAY = contains the original region A transition indicators (3's or
                                4's or 1's or 2's)
% 2) SLOPE_APPROX = contains the slope approximations for the respective transitions (3's or 4's or 1's or 2's)
% 3) VALUE_INDICATOR1 = contains a specific indicator for the absence or presence of a 3 (1)
% 4) VALUE_INDICATOR2 = contains a specific indicator for the absence or presence of a 4 (2)
% 5) FIRST_TARGET_VALUE = contains an indicator of the specific edge 3 (1) transition being compared against
% 6) SECOND_TARGET_VALUE = contains an indicator of the specific edge 4 (2) transition being compared against
% ================================================================
%     OUTPUT LIST (in order that they appear as parameter list):
% ================================================================
% 1) APPROX_SLOPE = the average of the slope approximations passed to the function
% 2) APPROX_SLOPE_FROM_FLAG1 = an indicator of which specific edge transitions contributed to the average slope
% 3) APPROX_SLOPE_FROM_FLAG2 = an indicator of which specific edge transitions contributed to the average slope
% ================================================================
%     OTHER CUSTOM FUNCTIONS USED
% ================================================================
% 1) NONE


function [approx_slope, approx_slope_from_flag1, approx_slope_from_flag2] =
approx_slope_averaging(original_region_a_start_end_value_array, slope_approx, value_indicator1, value_indicator2,
first_target_value, second_target_value);

approx_slope_from_flag1 = 0;
approx_slope_from_flag2 = 0;
approx_slope = 0.0;

orig_first_indicator = []; orig_second_indicator = [];
orig_first_indicator = find(original_region_a_start_end_value_array == first_target_value);   % Indicator used to store original
                region A start end 3 values (4 values)
orig_second_indicator = find(original_region_a_start_end_value_array == second_target_value);   % Indicator used to store
                original region A start end 1 values (2 values)
```

Figure X: Code listing portion for function approx_slope_averaging.m

```
if (~isempty(value_indicator1) && ~isempty(value_indicator2))            % If both 'blue' and 'green' edge transitions exist

   length_orig1_ind = [];  orig1_indicator_length = [];
   length_orig2_ind = [];  orig2_indicator_length = [];

   [length_orig1_ind orig1_indicator_length]= size(orig_first_indicator);  % Number of original blue indicators (magenta)
   [length_orig2_ind orig2_indicator_length]= size(orig_second_indicator); % Number of original green indicators (yellow)
   sum_orig1_slope_approxs = 0;                                  % Initialize sum of slope approx variable
   for a = 1:length_orig1_ind                                    % For number of original 'blue' slope approximations ('magenta')
      sum_orig1_slope_approxs = sum_orig1_slope_approxs + slope_approx(orig_first_indicator(a)); % Sum orig 'blue' slope
   end;

   sum_orig2_slope_approxs = 0;                                  % Initialize sum of slope approx variable
   for a = 1:length_orig2_ind                                    % For number of original 'green' slope approximations ('yellow')
      sum_orig2_slope_approxs = sum_orig2_slope_approxs + slope_approx(orig_second_indicator(a)); % Sum 'em
   end;

% *** NOW POSITIVE SLOPE APPROXIMATION EQUALS (SUM OF BLUE AND GREEN SLOPE)
   if (length_orig1_ind + length_orig2_ind) > 1
      approx_slope = (sum_orig1_slope_approxs + sum_orig2_slope_approxs) / (length_orig1_ind + length_orig2_ind);
   end;
   approx_slope_from_flag1 = 1;
   approx_slope_from_flag2 = 1;

elseif (isempty(value_indicator1) && ~isempty(value_indicator2))          % If 'green' but no 'blue' edge transitions exist

   approx_slope = mean(slope_approx(orig_second_indicator));              % Slope approximation is mean of 'green' slope
   approx_slope_from_flag1 = 1;                                  % Approx slope from green edge only flag set (yellow flag set)

elseif (~isempty(value_indicator1) && isempty(value_indicator2))      % If 'blue' but no 'green' edge transitions exist
   approx_slope = mean(slope_approx(orig_first_indicator));              % Slope approximation is mean of 'blue' slope approx's
   approx_slope_from_flag2 = 1;                                  % Approx slope from blue edge only flag set        (magenta flag
elseif (isempty(value_indicator1) && isempty(value_indicator2))
   approx_slope = 0.0;
   approx_slope_from_flag1 = 0;
   approx_slope_from_flag2 = 0;
end;

if (approx_slope == 0.0) || (isnan(approx_slope))      % If variable still zero or now NAN, generate warning and break
   msgbox('ZERO or NAN VARIABLE VALUE','WARNING','warn');
   break;
end;
```

Figure X (continued): Code listing portion for function approx_slope_averaging.m

The calculation of the slope averaging approximation is relatively straightforward. Each of the slopes which have passed the discrimination process are stored in an array with an indicator designating whether that value corresponds to a transition for a positive sloped dark to light edge, a positive sloped light to dark edge, a negative sloped dark to light edge, or a negative sloped light to dark edge. The function considers both complementary transitions during the slope

averaging approximation. It sums each of the slope values for either the positive or negative edges and divides the sum by the total number of edges that were summed. If only one of the two complementary edge transitions were present, the mean value is taken and a flag is set indicating the slope approximation is generated from only one of the two complementary edges. If there were no complementary edges present for a given positive or negative sloped edge, the slope approximation is set to zero. Note that this is not the complete manner in which the average slope may be calculated. It might be necessary to consider some statistical methods on the various slopes stored in the array before generating an average slope approximation. It may also be necessary to consider weighting certain slope values based on criterion pertinent to the properties of the road scene image or the total number of edge components being considered (or other relevant factors).

Once the significant points have been established and the slope approximations have been calculated, it may be necessary to take the points and the slope approximation and generate the equation of a line from those points and slope values. This would allow generation of both the forward and rearward projections for the (existing) positive and negative sloped connected segments (to be discussed further in the next section of this paper). The code for potentially generating the line approximations is shown in figure X. An important consideration needs to be made when looking at the code in figure X and is illustrated in figure X.

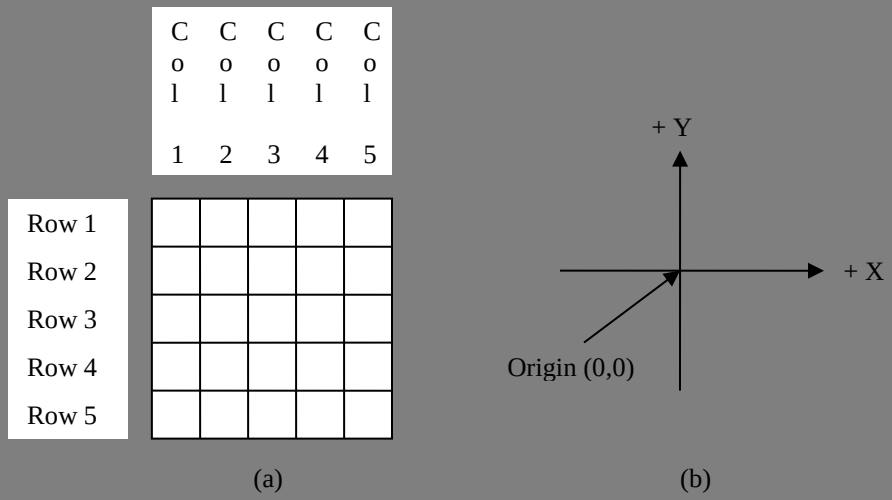


Figure X: (a) Showing (row,column) format for Matlab matrices; (b) Cartesian coordinate system

In Cartesian coordinates, (x, y) coordinate pairs 'correspond' to (column, row) values in Matlab matrices. However, Matlab matrices are actually accessed using (row, column) pairs. Thus, the point-slope equation

$$y - y_1 = m(x - x_1) \qquad \textbf{(X)}$$

would actually be

$$x - x_1 = m(y - y_1) \qquad \textbf{(X)}$$

and then solving for y would yield

$$y = \frac{1}{m}(x - x_1) + y_1 . \qquad \textbf{(X)}$$

This accounts for taking the reciprocal in figure X. The reason for taking the negative sign results from the location of the origin in figure X and the relationship between the reference directions used in each of (a) and (b). Although the positive and negative sloped borders may potentially be projected both nearer and further within the FOV, there is still no explicit relationship between the two. Knowing the position of one prospective border with respect to another may provide a number of useful characteristics including (but not limited to) a foundation for establishing geometric relationships between the two dimensional image and the three dimensional world.

```
% Programmer: Christopher Alan Warner
% GENERATE_LINE_APPROXIMATIONS is a function which will calculate a series
% of points corresponding to the line generated from the particular
% approximate slopes and specific edge point locations
% ================================================================
%     INPUT LIST (in order that they appear as parameter list):
% ================================================================
% 1) REGION_A_START_END_POINT_ARRAY = contains the region A transition points for the particular edge  transition
% 2) APPROX_SLOPE_FLAG1 = contains an indicator of the specific transition contributing the slope
% 3) APPROX_SLOPE_FLAG2 = contains an indicator of the specific transition contributing the slope
% 4) APPROX_SLOPE = contains the average of the slope approximations
% 5) INDEX_1 = contains an indicator of a specific edge point location within the region A array
% 6) INDEX_2 = contains an indicator of a specific edge point location within the region A array
% 7) INDEX_3 = contains an indicator of a specific edge point location within the region A array
% 8) INDEX_4 = contains an indicator of a specific edge point location within the region A array
% 9) TARGET_ROW = the furthest row number within the FOV (currently 35)
% 10) SLOPE_INDICATOR = an indicator for positive or negative sloped line approximation
% ================================================================
%     OUTPUT LIST (in order that they appear as parameter list):
% ================================================================
% 1) XC1 = a point from the specific edge transition used in the line approximation
% 2) YC1 = a point from the specific edge transition used in the line approximation
% 3) X1 = the target row value (currently 35)
% 4) Y1 = one point calculated from the point-slope equation
% 5) XC2 = a point from the specific edge transition used in the line approximation
% 6) YC2 = a point from the specific edge transition used in the line approximation
% 7) X2 = the target row value (currently 35)
% 8) Y2 = one point calculated from the point-slope equation
% 9) XC1_TARGET_ROW = target row calculated from target col, actual points, slope approx...
% 10) YC1_TARGET_ROW = target col 1 or 397 depending on slope indicator
% 11) XC2_TARGET_ROW = target row calculated from target col, actual points, slope approx...
% 12) YC2_TARGET_ROW = target col 1 or 397 depending on slope indicator

% ================================================================
%     OTHER CUSTOM FUNCTIONS USED
% ================================================================
% 1) NONE

function [xc1, yc1, x1, y1, xc2, yc2, x2, y2, xc1_target_row, yc1_target_col, xc2_target_row, yc2_target_col] =
            generate_line_approximations(region_a_start_end_point_array, approx_slope_flag1, approx_slope_flag2,
            approx_slope, index_1, index_2, index_3, index_4, target_row, slope_indicator)

x1 = 0; y1 = 0; x2 = 0; y2 = 0;
xc1 = 0; yc1 = 0; xc2 = 0; yc2 = 0;
xc1_target_row = 0; yc1_target_col = 0;
xc2_target_row = 0; yc2_target_col = 0;
min_col = 1;    max_col = 397;
```

Figure X: Code listing portion for function generate_line_approximations.m

```
if (approx_slope_flag1 || approx_slope_flag2)  % If 'green' or 'blue' edges exist ('yellow' or 'magenta' exist)
   if (approx_slope ~= 0)                      % If approx slope is non-zero
      if (isfinite(approx_slope))              % If approx slope is finite (non-infinite)
         if (approx_slope_flag1 && approx_slope_flag2) || (~approx_slope_flag1 && approx_slope_flag2)
                                  % If both  green and blue present OR only blue
            xc1 = region_a_start_end_point_array(index_1);          % Get first point from array for blue line (magenta line)
            yc1 = region_a_start_end_point_array(index_2);          % Get second point from array for blue line (magenta
         elseif (approx_slope_flag1 && ~approx_slope_flag2)         % elseif only green present
            xc1 = region_a_start_end_point_array(index_3);
            yc1 = region_a_start_end_point_array(index_4);
         end;

         x1 = target_row;

         y1 = 0;                                 % Initialize value being calculated
         y1 = (-1/approx_slope) * (x1 - xc1) + yc1;

         if (slope_indicator == 1)
            if (yc1 >= min_col)
               yc1_target_col = min_col;                % Target is nearest to FOV
               yc2_target_col = min_col;
            end;
         elseif (slope_indicator == 2)
            if (yc1 <= max_col)
               yc1_target_col = max_col;                % Target is furthest to FOV
               yc2_target_col = max_col;
            end;
         end;
         xc1_target_row = -(((y1 - yc1_target_col) * (-approx_slope)) - (x1));

         if (approx_slope_flag1 && approx_slope_flag2) || (approx_slope_flag1 && ~approx_slope_flag2)
           % If both  green and blue present OR only green
            xc2 = region_a_start_end_point_array(index_3);      % Get first point from array for green line (yellow line)
            yc2 = region_a_start_end_point_array(index_4);      % Get second point from array for green line (yellow line)
         elseif (~approx_slope_flag1 && approx_slope_flag2)    % elseif only blue present
            xc2 = region_a_start_end_point_array(index_1);
            yc2 = region_a_start_end_point_array(index_2);
         end;
         x2 = x1;                                % x2 size now has been modified based on presence of 'complementary' edge
         y2 = 0;                                 % Value being calculated initialized
         y2 = (-1/approx_slope) * (x2 - xc2) + yc2;

         xc2_target_row = -(((y2 - yc2_target_col) * (-approx_slope)) - (x2));
      end;
   end;
end;
```

Figure X (continued): Code listing portion for function generate_line_approximations.m

Figure X shows a sequence of images which have had their edges detected, connected, and discriminated.

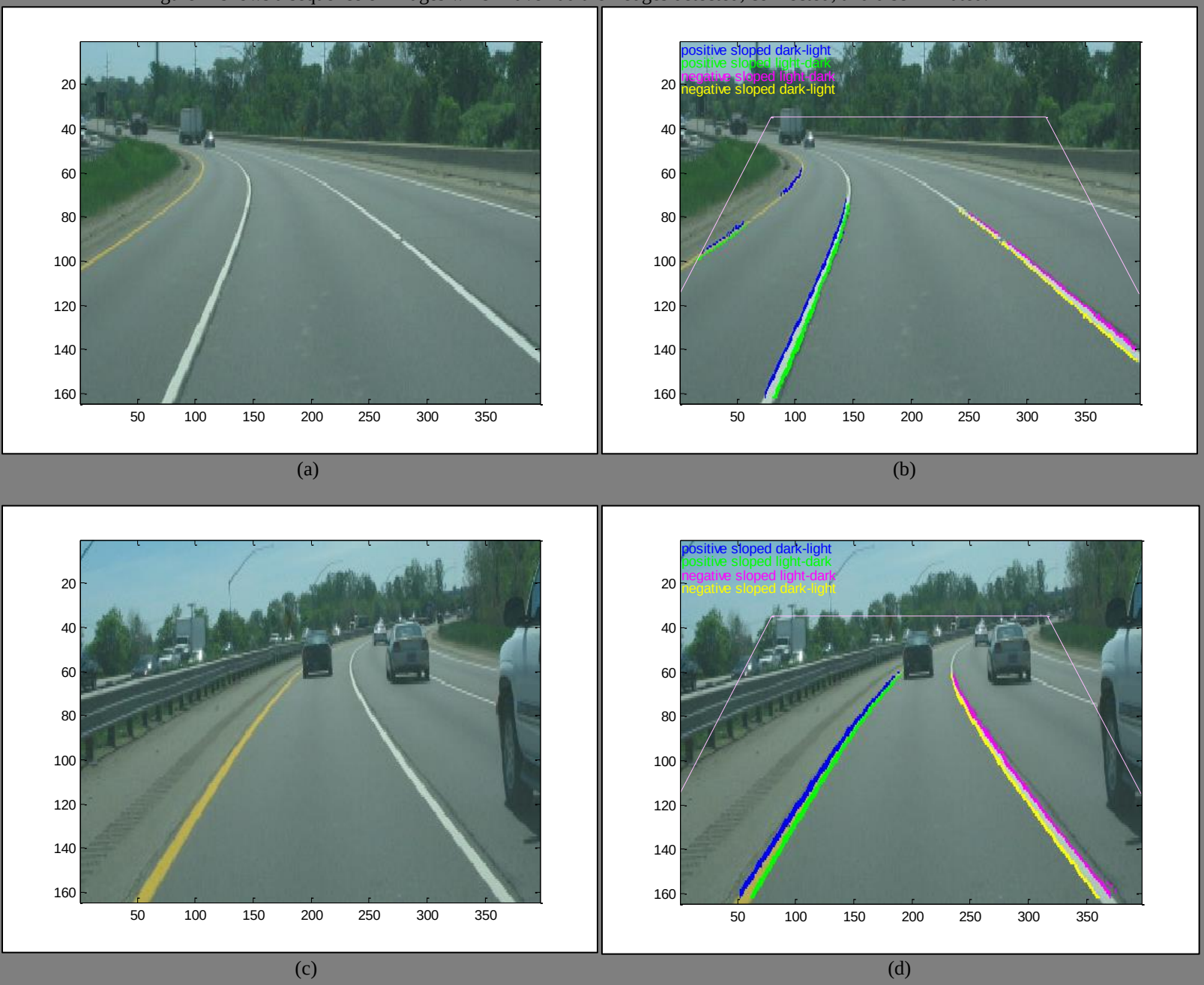
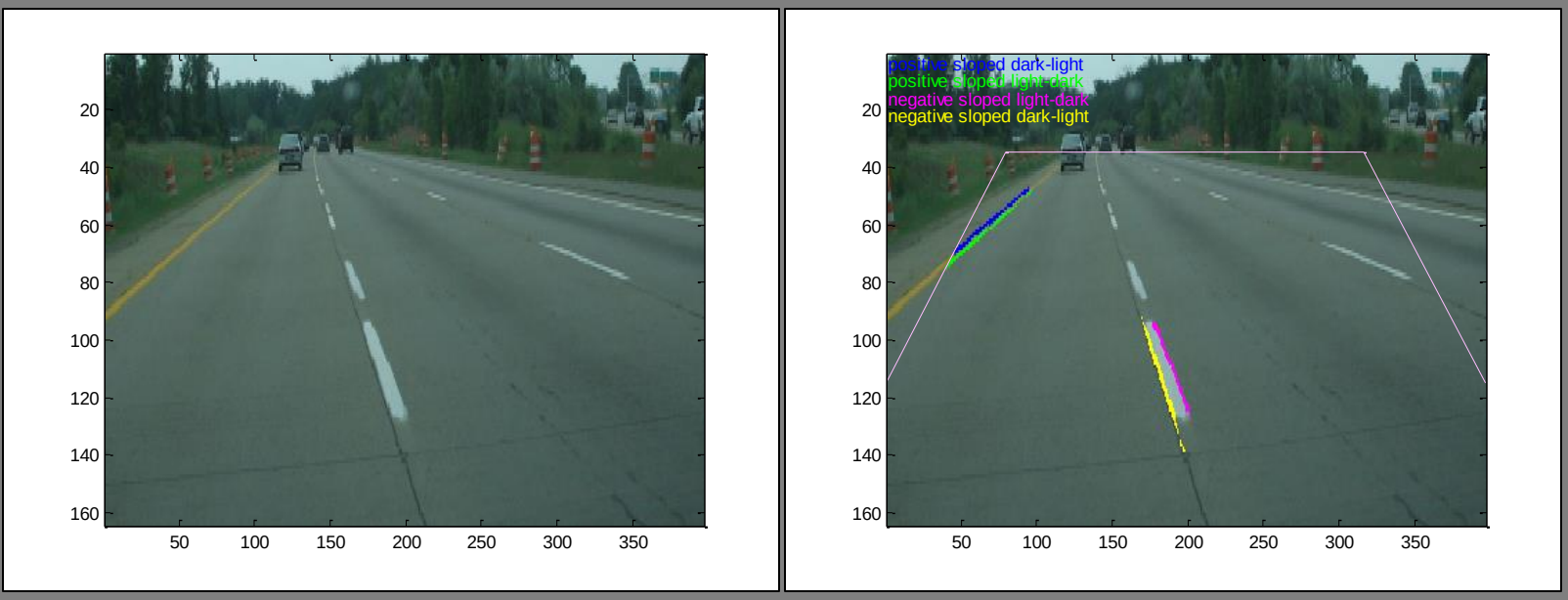


Figure X: (a) Original cropped RGB image; (b) preliminary results of detection, connection, and GEM discrimination; (c) original cropped RGB image; (d) preliminary results of detection, connection, and GEM discrimination



(a) (b)

Figure X: (a) Original cropped RGB image; (b) preliminary results of detection, connection, and GEM discrimination

8.2 Using a system of linear equations to couple likely individual border projections

The slope of a line containing two distinct points P_1 and P_2 on that line may be expressed as

$$y_1 - y_2 = m_1(x_1 - x_2), \qquad \textbf{(X)}$$

where m_1 represents the slope of the line segment L_1 and the points (x_1, y_1) and (x_2, y_2) represent the starting and ending points of the line segment L_1.

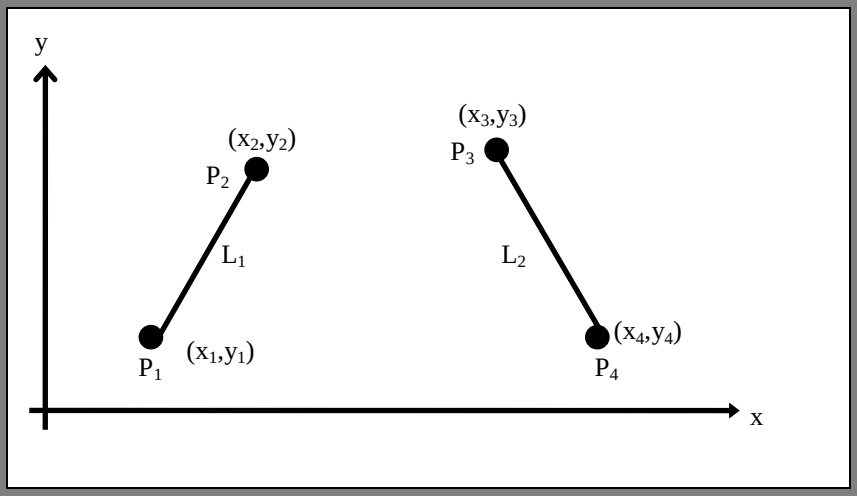


Figure X: Cartesian coordinate representation of lines L_1 and L_2 with points (x_1, y_1), (x_2, y_2), (x_3, y_3), (x_4, y_4)

Similarly, a relationship for a second line segment L_2 may be expressed as

$$y_4 - y_3 = m_2(x_4 - x_3). \qquad \textbf{(X)}$$

The slopes of these two lines may be calculated as

$$m_1 = (y_1 - y_2) / (x_1 - x_2) \qquad \textbf{(X)}$$

and

$$m_2 = (y_4 - y_3) / (x_4 - x_3). \qquad \textbf{(X)}$$

If L_1 and L_2 are not vertical and m_1 equals m_2, then the two line segments are parallel and 'forward projections' from the two would never intersect. Due to the perspective (and other important characteristics) of the road scene image processing system, there should never be more than one pair of distinct vertical complementary borders detected within the image for coupling. Similarly, there should never be positive and negative sloped border transitions detected where the relative positions of L_1 and L_2 as indicated in figure X are exchanged. Clearly, figure X indicates that forward projections from L_1 and L_2 will intersect and thus the two equations in X and X may be coupled and solved through the techniques of linear algebra yielding precisely one solution.

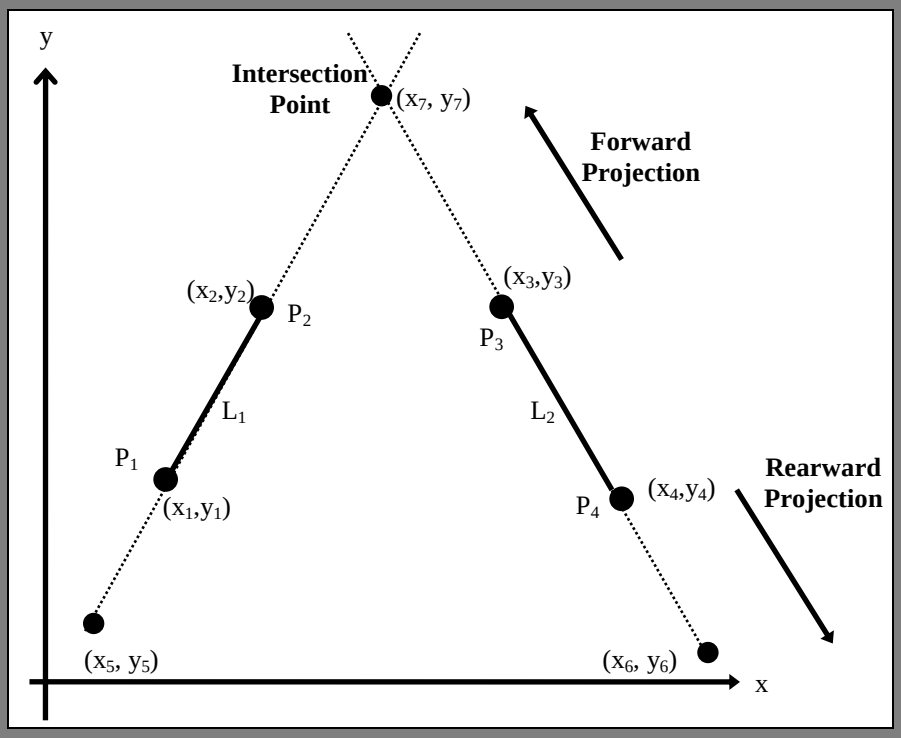


Figure X: Projections of lines L_1 and L_2 with labeling of additional points

Figure X better illustrates how the forward and rearward projections for the line segments L_1 and L_2 may be represented and shows additional points used in the analysis. If one point on L_1 is known and one point on L_2 is known, and the slopes of the lines L_1 and L_2 are known or may be approximated, then the point (x_7, y_7) may be determined by solving a system of equations. The reason that this is possible is because the system represents two equations in two unknowns.

Now assume the point (x_1, y_1) is known and that the point (x_4, y_4) is also known. Also assume that the slope values m_1 and m_2 may be calculated or somehow otherwise approximated. The intersection point (x_7, y_7) couples the two linear equations for the forward projections via

$$y_1 - y_7 = m_1(x_1 - x_7) \qquad \textbf{(X)}$$

and

$$y_4 - y_7 = m_2(x_4 - x_7). \qquad \textbf{(X)}$$

Expanding (X) and (X) yields

$$y_1 - y_7 = m_1x_1 - m_1x_7 \qquad \textbf{(X)}$$

and

$$y_4 - y_7 = m_2x_4 - m_2x_7. \qquad \textbf{(X)}$$

Converting the system of equations into matrix form then yields

$$\begin{bmatrix} m_1 & -1 \\ m_2 & -1 \end{bmatrix}\begin{bmatrix} x_7 \\ y_7 \end{bmatrix} = \begin{bmatrix} m_1x_1 - y_1 \\ m_2x_4 - y_4 \end{bmatrix}. \qquad \textbf{(X)}$$

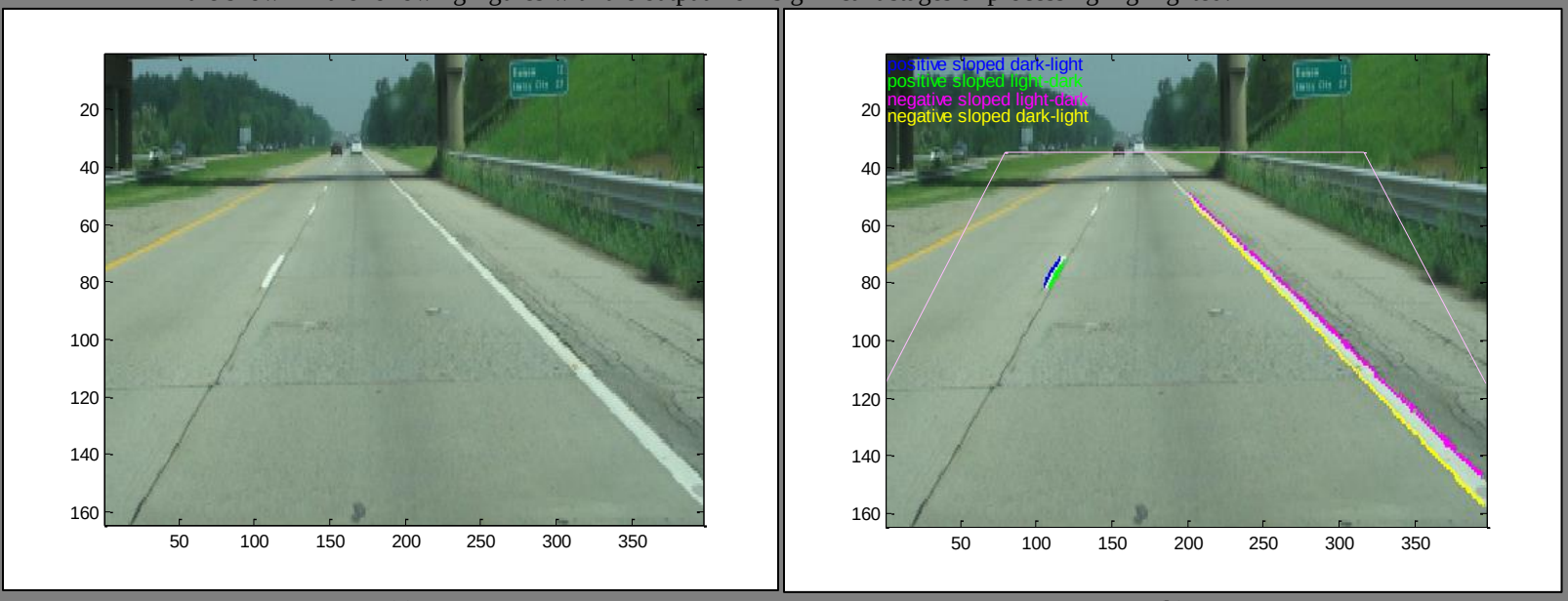
From there, x_7 and y_7 may be solved for as

$$\begin{bmatrix} x_7 \\ y_7 \end{bmatrix} = \begin{bmatrix} m_1 & -1 \\ m_2 & -1 \end{bmatrix}^{-1}\begin{bmatrix} m_1x_1 - y_1 \\ m_2x_4 - y_4 \end{bmatrix}. \qquad \textbf{(X)}$$

The system of equations may also be solved using other standard methods for solving N equations in N unknowns. It should be noted that care needs to be taken when considering distances between pixels within images because the distance that is calculated should represent the distance from the center of the pixel at one point to the center of the pixel at the next point.

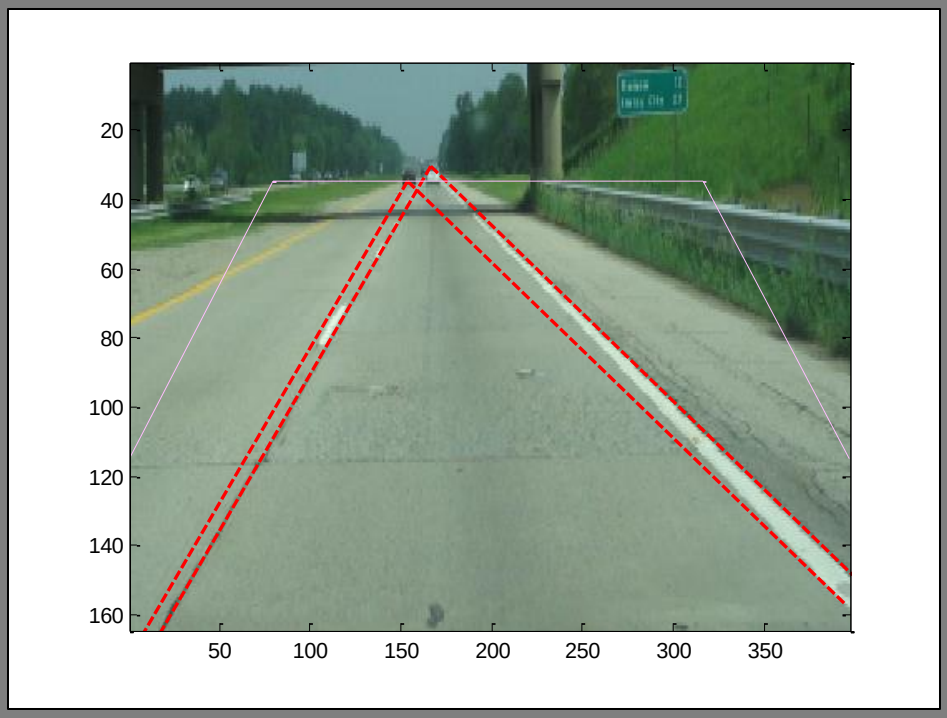
A road scene image with longitudinal lane markers delineating both the left and right boundaries of a lane will typically result in two equations in two unknowns. Solving the two equations in a fashion similar to that described previously may produce a set of related projections from which other pertinent information may be drawn.

The intersection point (if it exists) is typically the approximate location where the positive and negative sloped forward projections meet in the image plane. If only a single set of complementary border transitions is detectable, the preliminary road scene image processing system should have the capability to determine projections using that available information. Note also that if the intersection point does not fall somewhere in the current image plane, the projection lines should still reflect where in the image plane the forward projections would intersect. The previous discussions regarding equations related to line segments have revolved around the generalized diagram in figure X and have focused on the forward projections. Note that rearward projections are also possible and may be determined using similar information to that which is used to formulate the forward projections. The rearward projection points are represented in figure X by the coordinate pair (x_5, y_5) for the positive sloped transition and the coordinate pair (x_6, y_6) for the negative sloped transition. Since a relationship between distinct points on a line and the slope of that line has already been formulated to produce the forward projections, it is simply a matter of substituting the row value for the intended rearward projection point into a comparable equation to obtain the desired column value. A number of road scene images are shown in the following figures with the output from significant stages of processing highlighted.



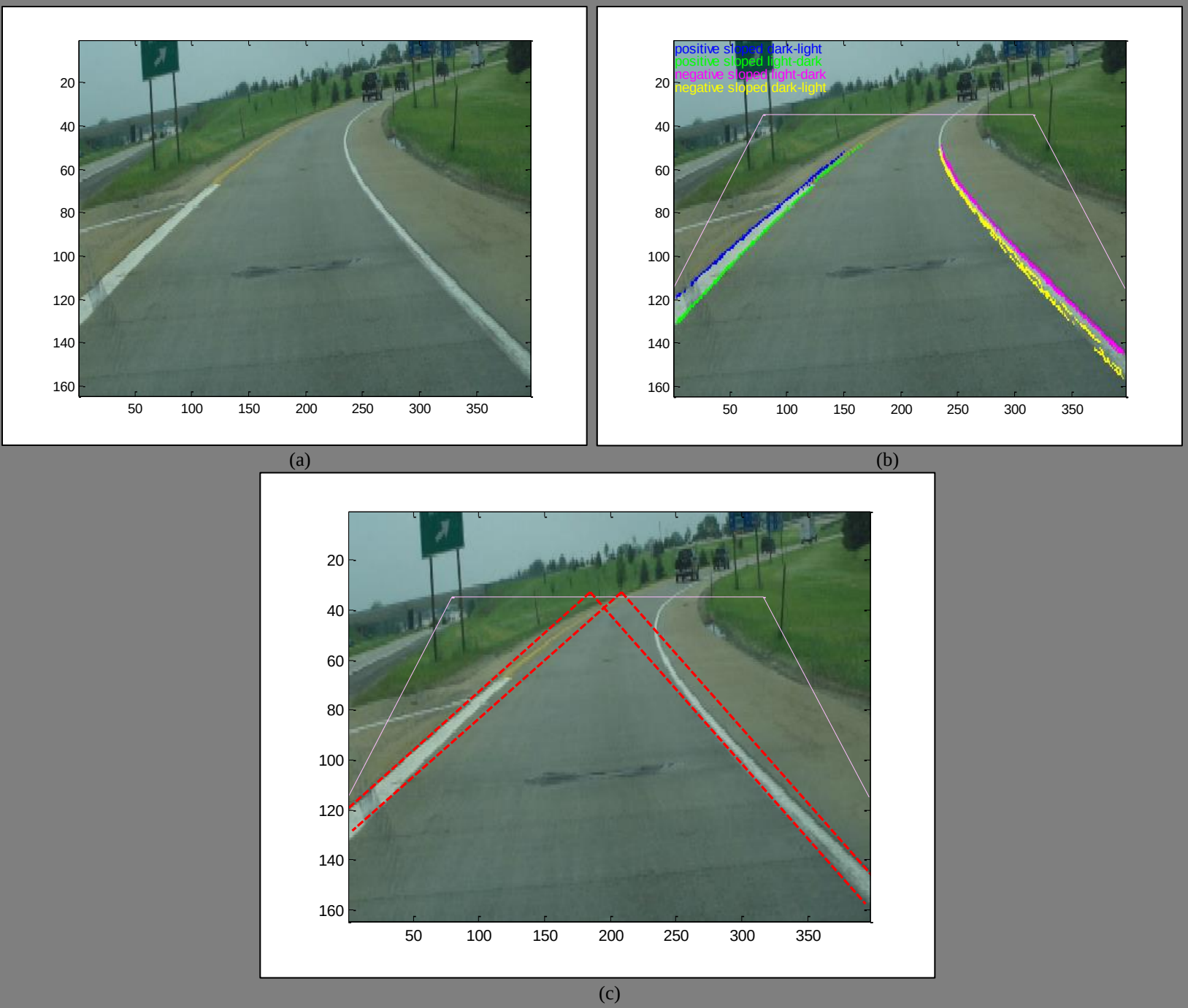
(a) (b)

Figure X: (a) Cropped RGB image; (b) preliminary results of detection, connection, and GEM discrimination



(c)

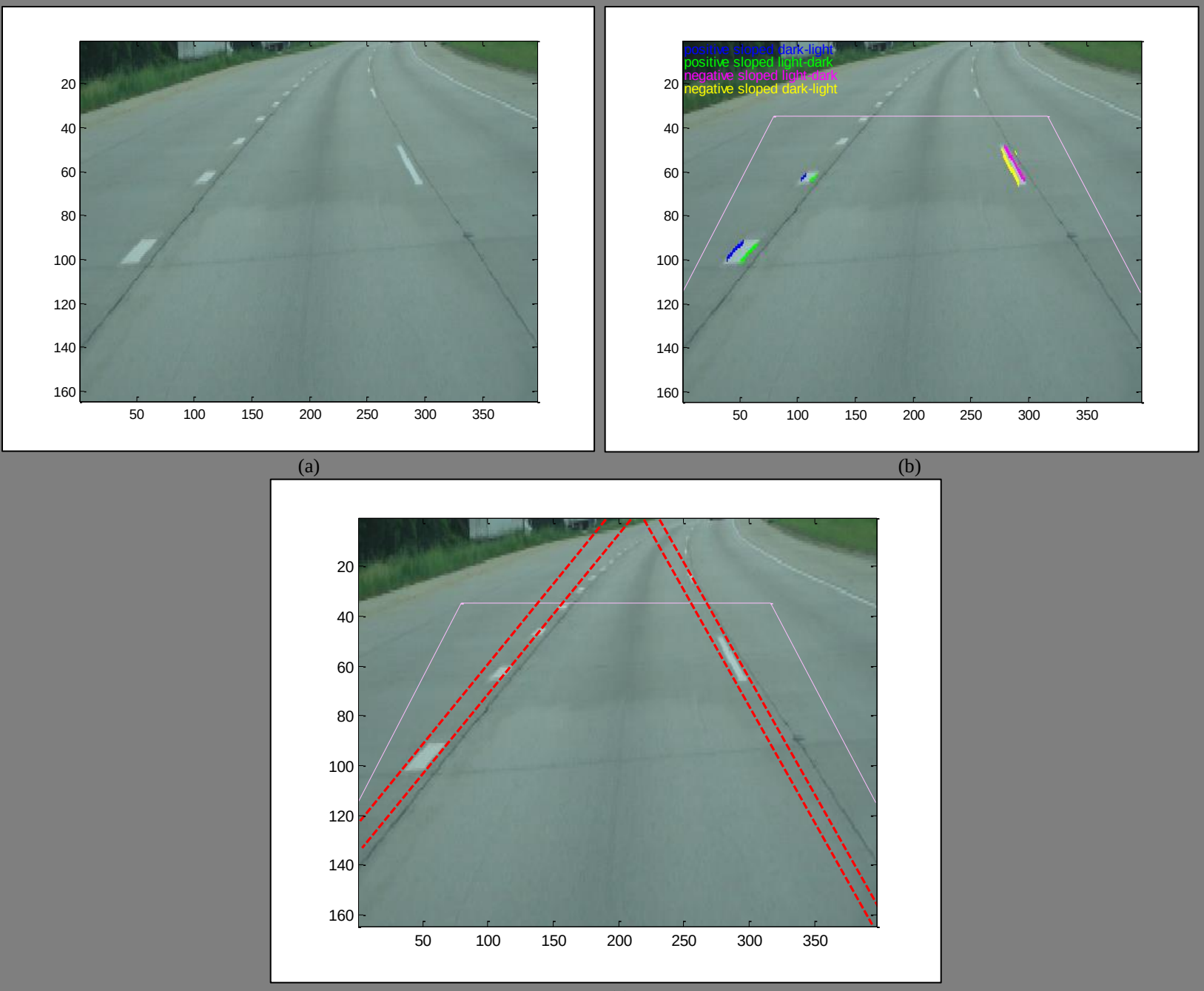
Figure X (continued): (c) RGB cropped including projections



(a) (b)

(c)

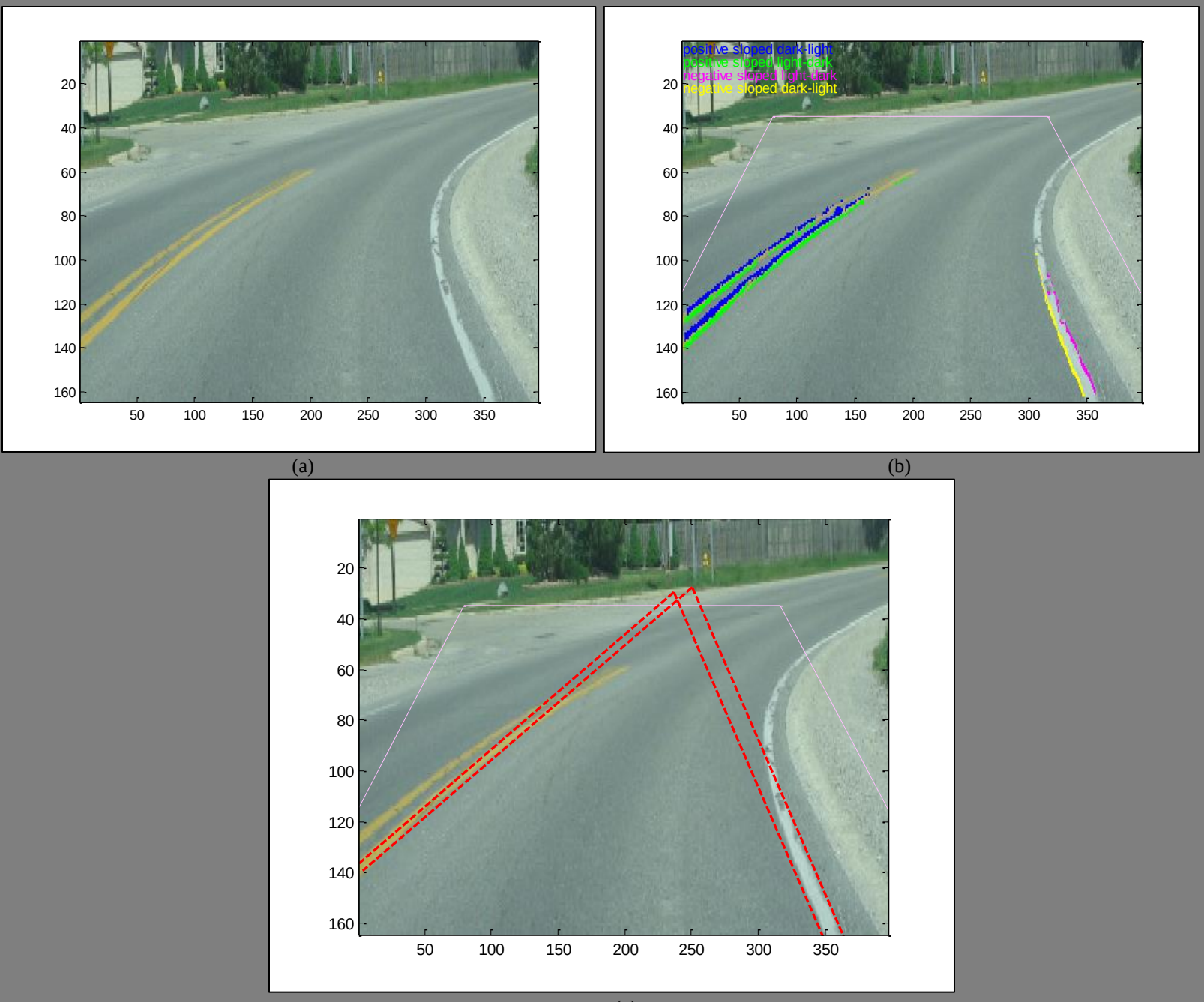
Figure X: (a) Cropped RGB road scene image; (b) preliminary results of detection, connection, and GEM discrimination; (c) RGB cropped including projections



(a) (b)

(c)

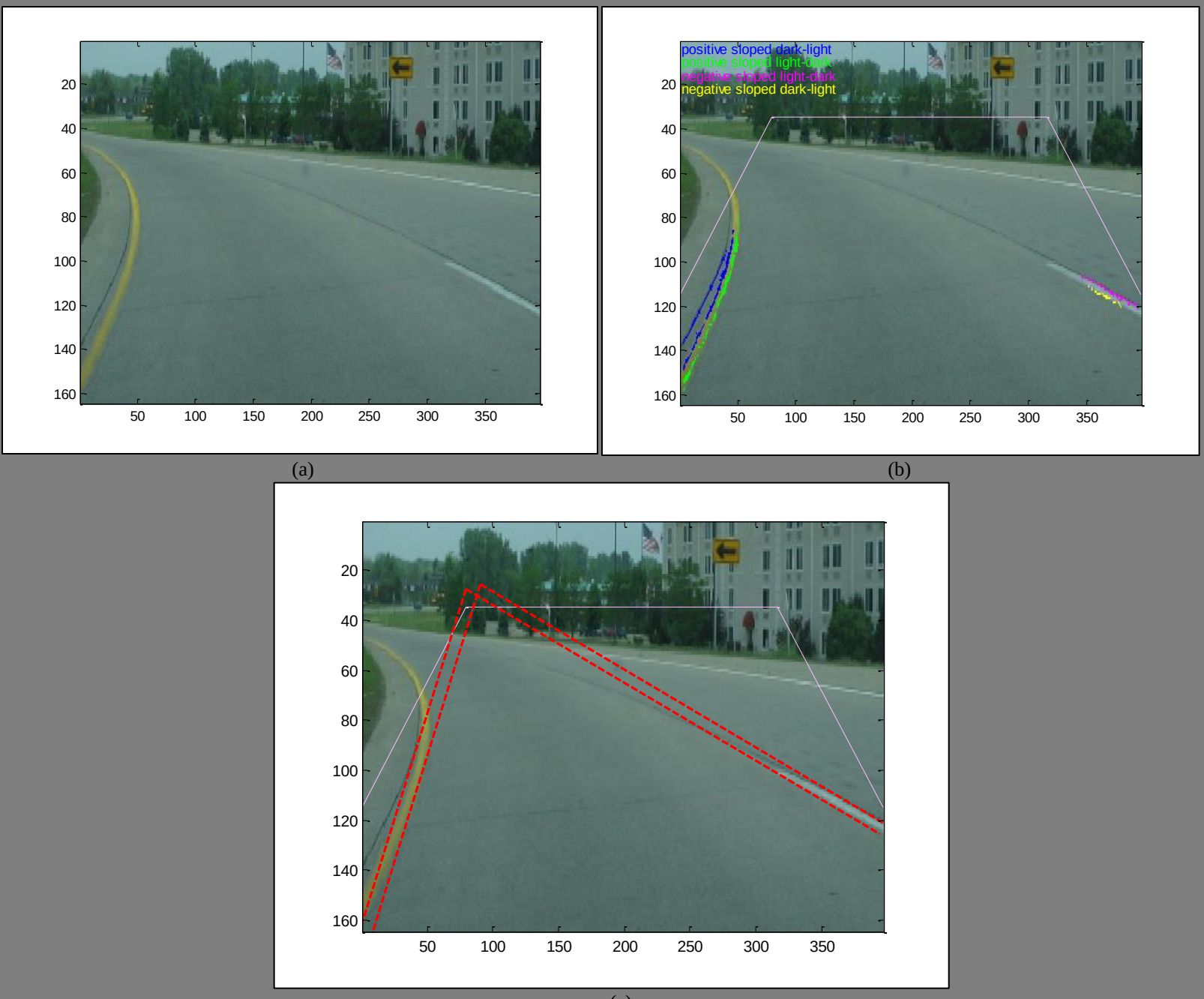
Figure X: (a) Cropped RGB road scene image; (b) preliminary results of detection, connection, and GEM discrimination; (c) RGB cropped including projections



(a) (b)

(c)

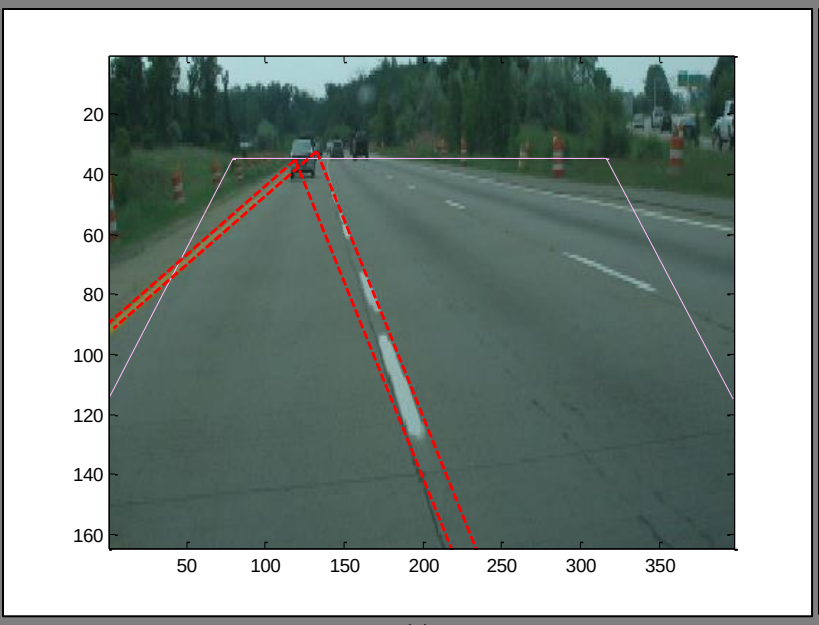
Figure X: (a) Cropped RGB road scene image; (b) preliminary results of detection, connection, and GEM discrimination; (c) RGB cropped including projections



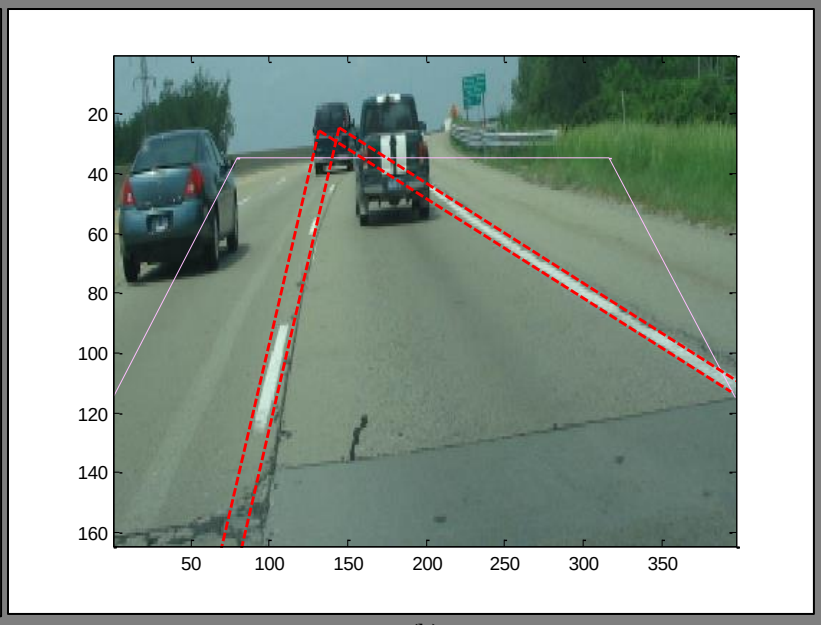
(a) (b)

(c)

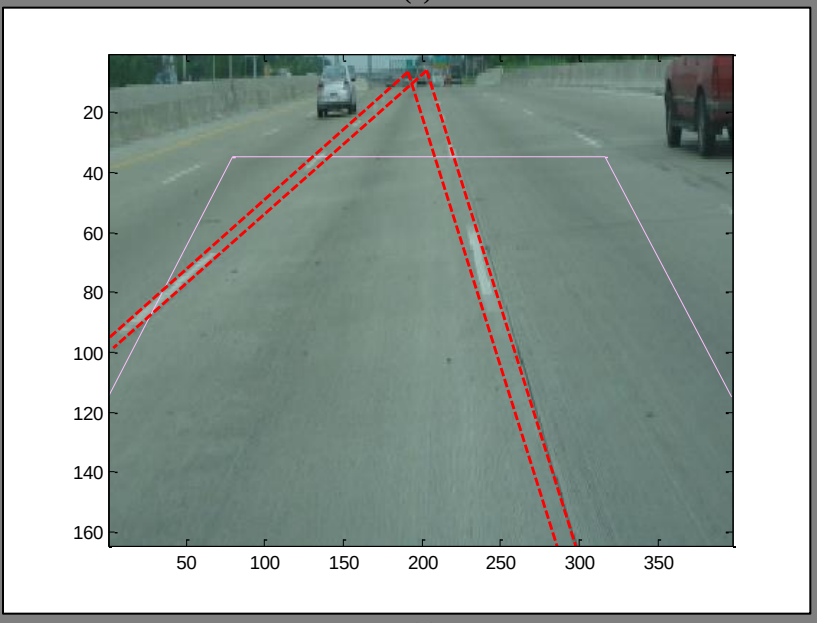
Figure X: (a) Cropped RGB road scene image; (b) preliminary results of detection, connection, and GEM discrimination; (c) RGB cropped including projections



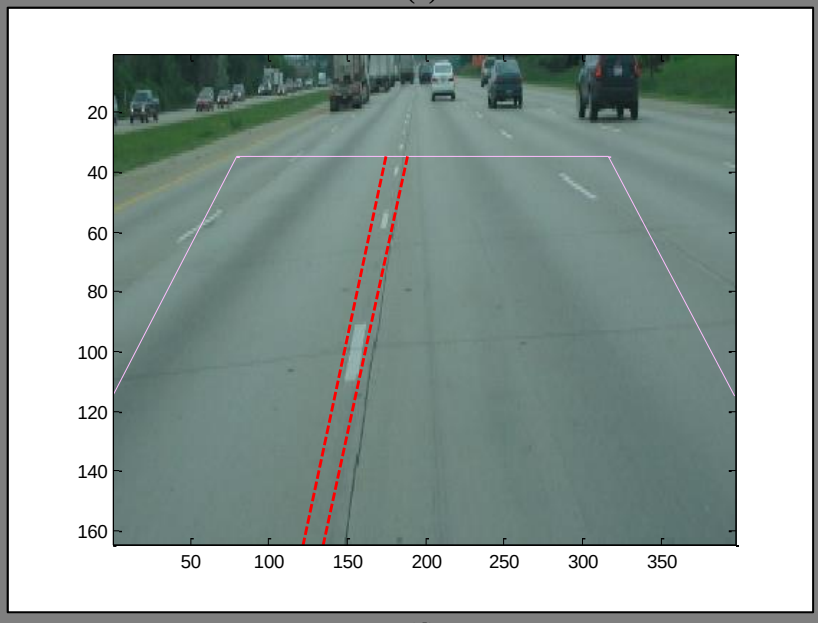
(a)



(b)

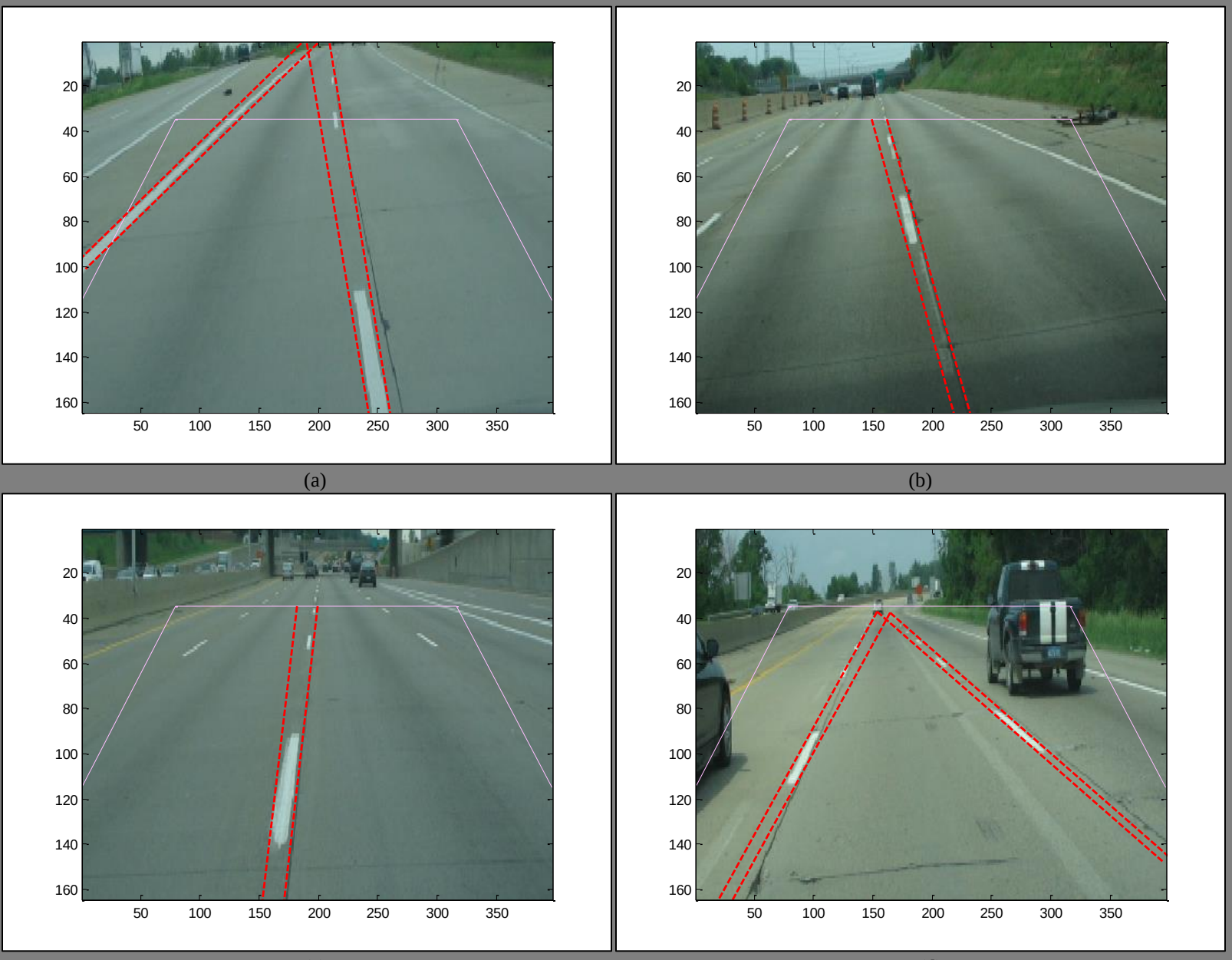


(c)



(d)

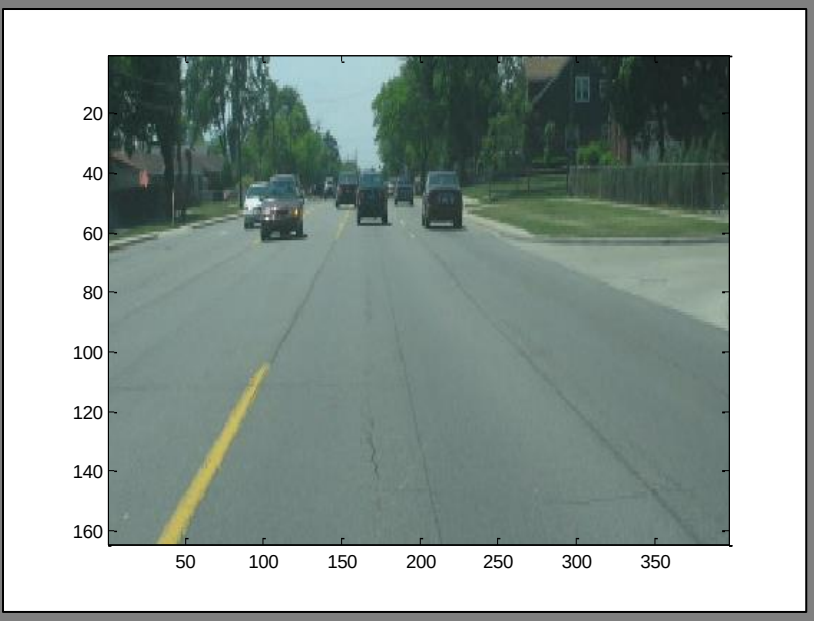
Figure X: Various RGB cropped road scenes (a) - (d) with projections



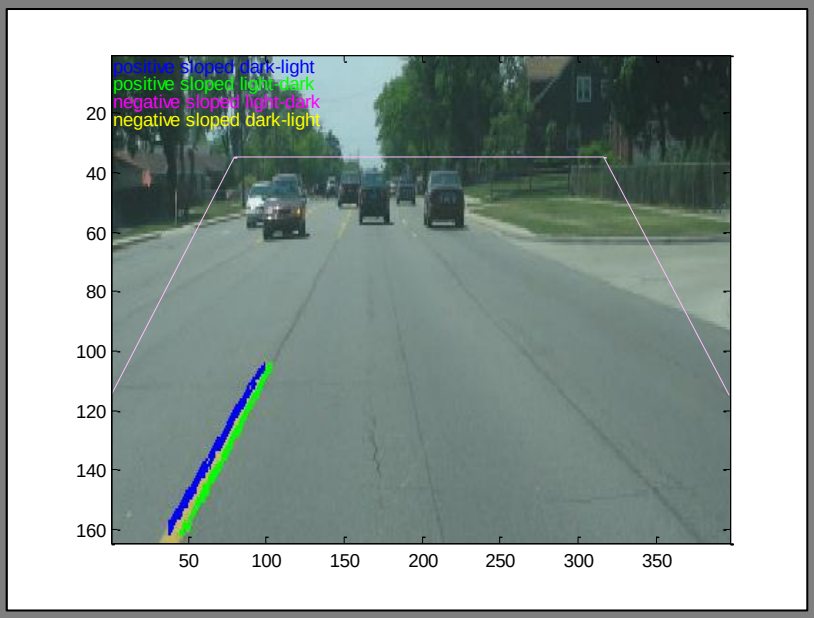
(a) (b)

(c) (d)

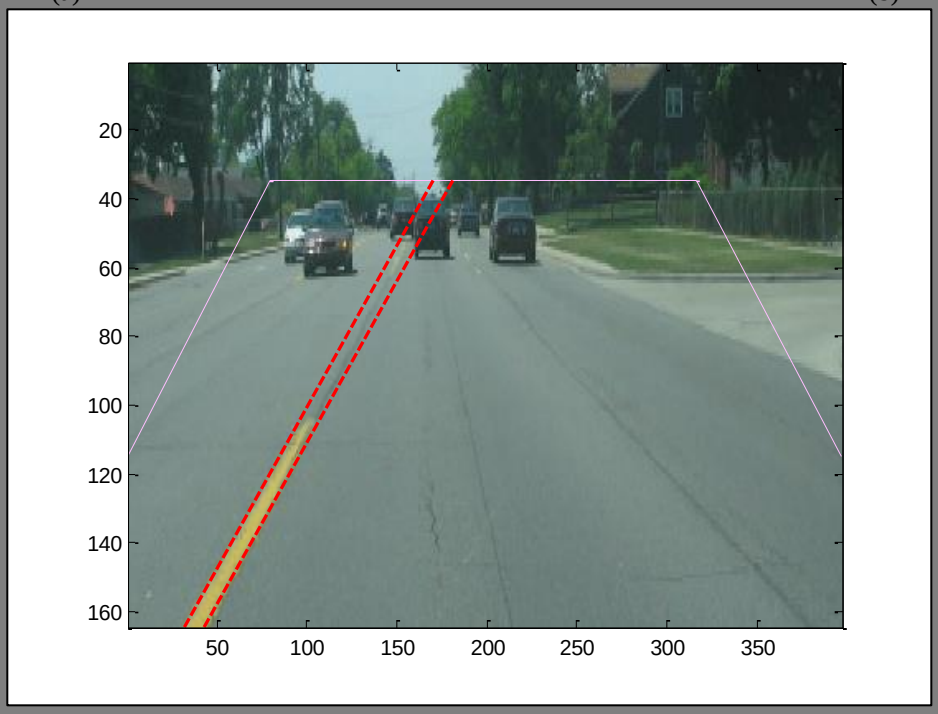
Figure X: Various RGB cropped road scenes (a) – (d) with projections



(a)



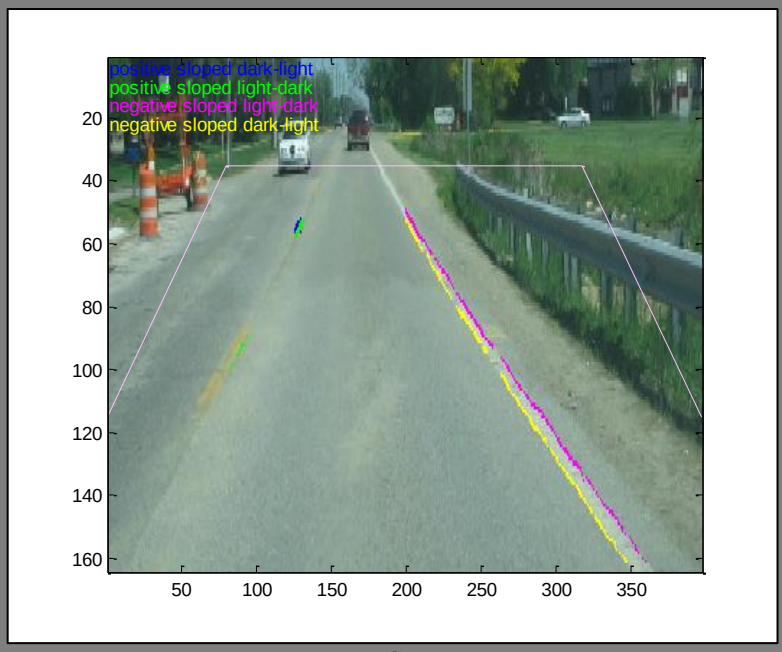
(b)



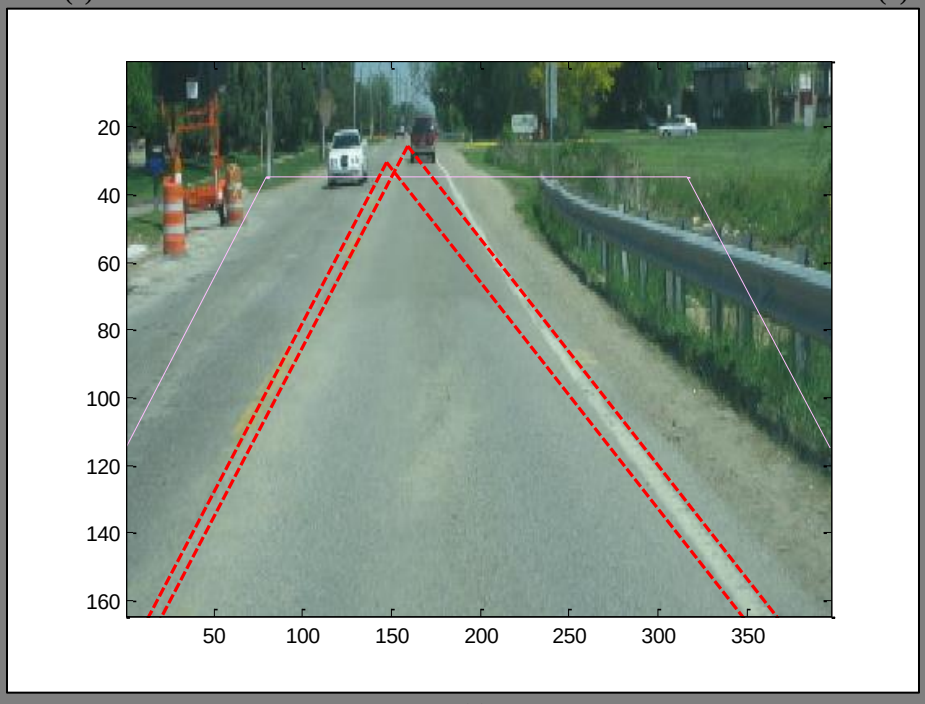
(c)

Figure X: (a) Cropped RGB image; (b) preliminary results of detection, connection, and GEM discrimination; (c) RGB cropped showing only positive sloped projections in red

(a)

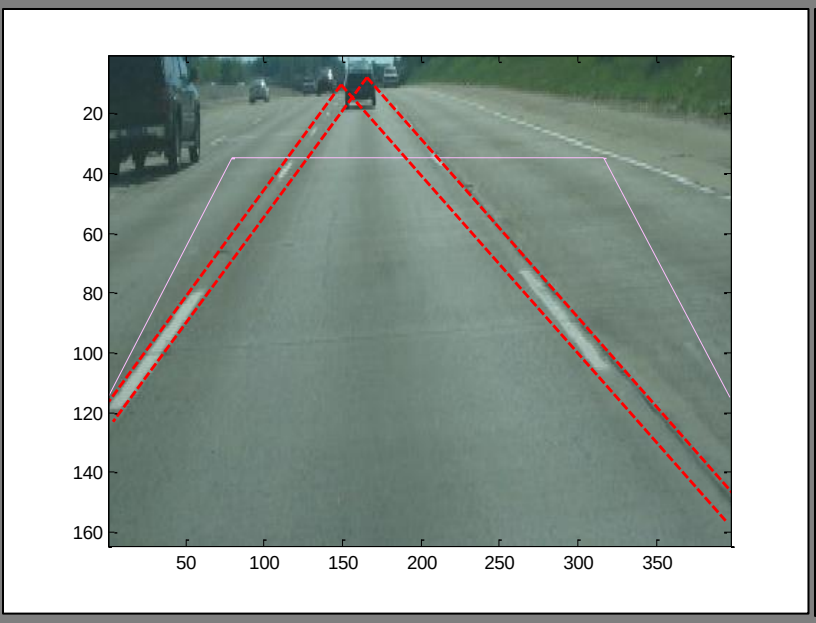


(b)

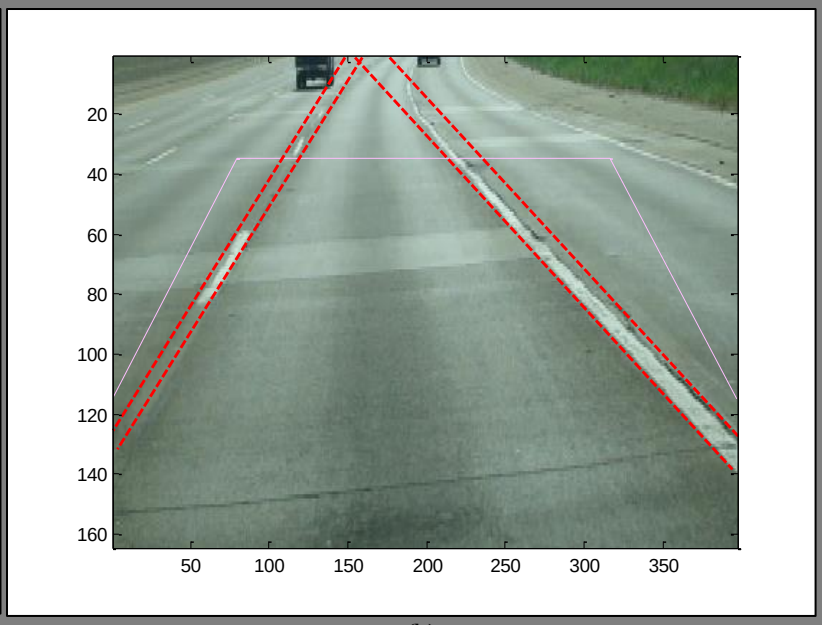


(c)

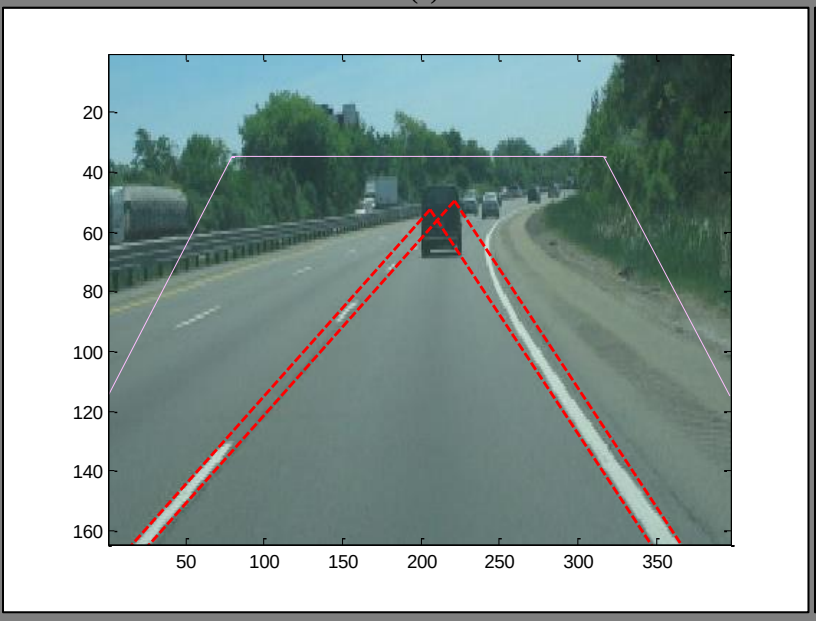
Figure X: (a) RGB cropped image with severely degraded yellow marker; (b) preliminary results of detection, connection, and GEM discrimination; (c) RGB cropped road scene image with projections in red



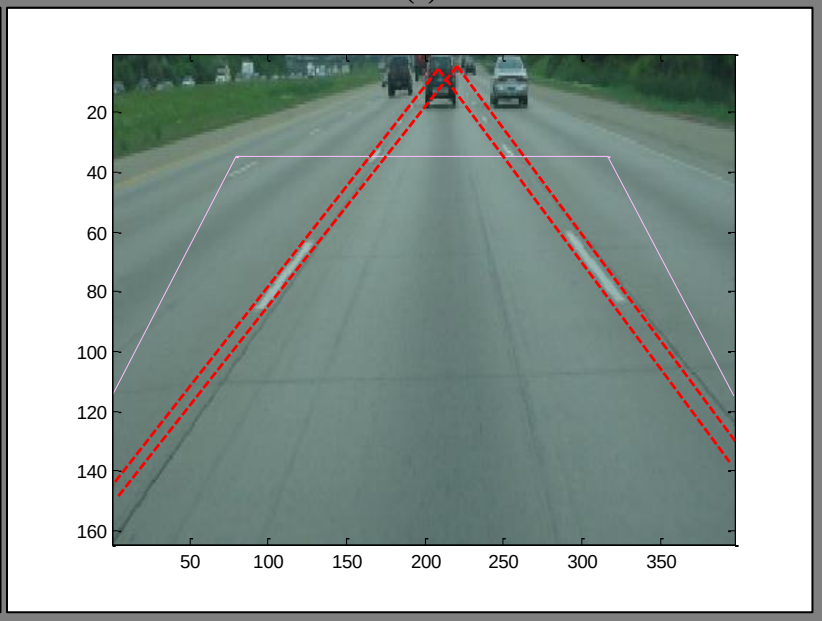
(a)



(b)

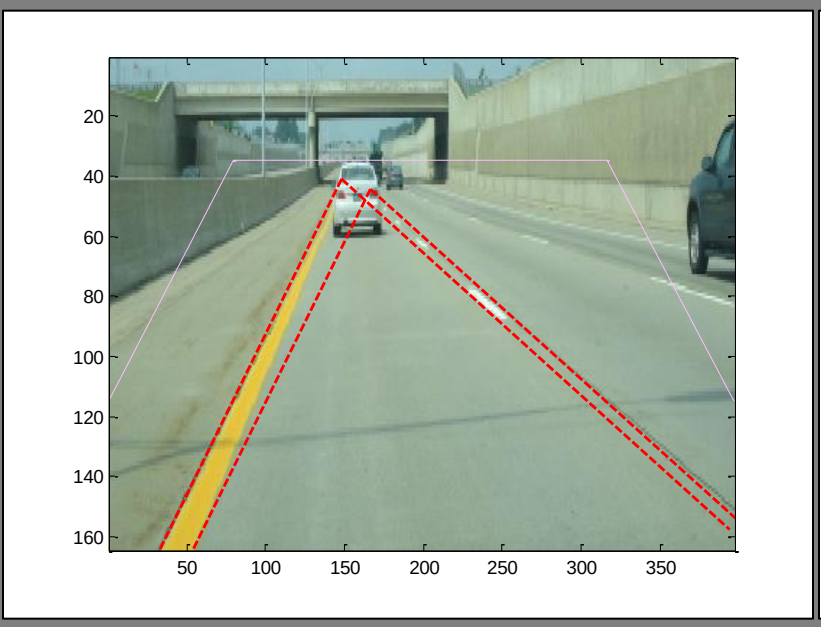


(c)

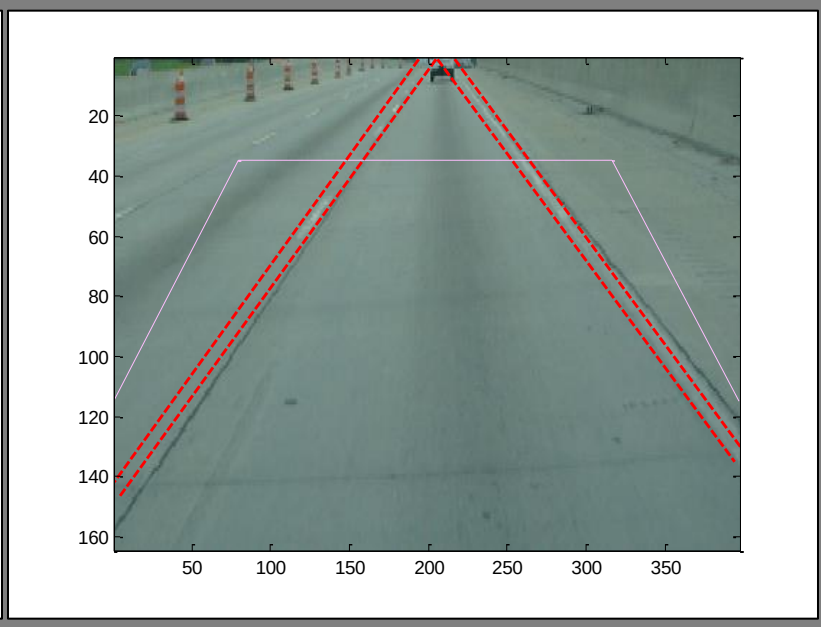


(d)

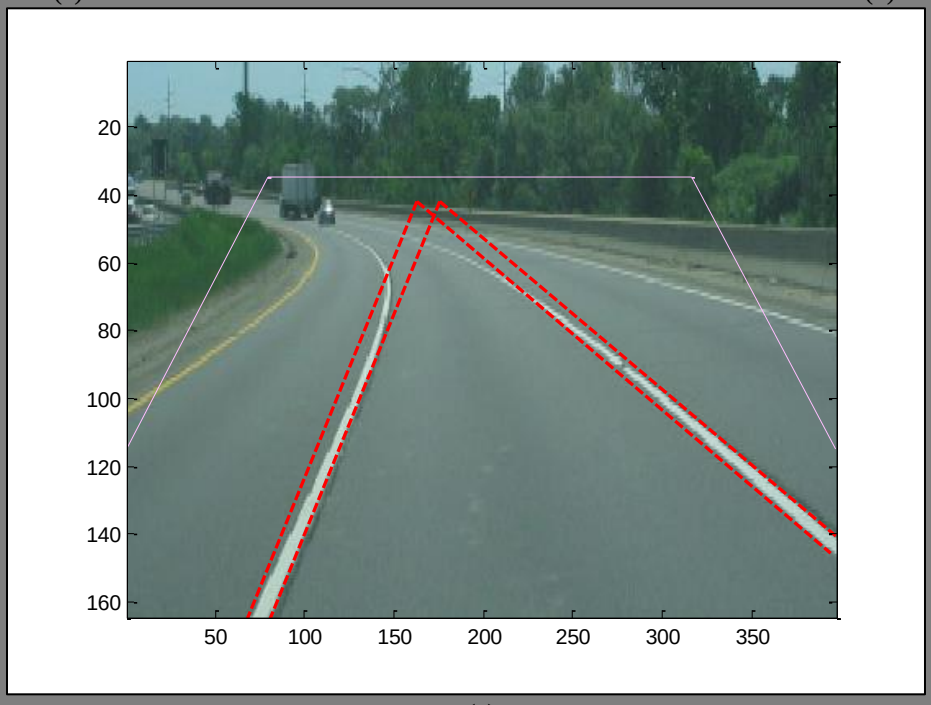
Figure X: Various RGB cropped road scenes (a) – (d) with projections



(a)



(b)



(c)

Figure X: Various RGB cropped road scenes (a) – (c) with projections

When considering calculating the inverse of a matrix, several important things need to be noted. First consider the following equation,

$$Ax = b \qquad \textbf{(X)}$$

or

$$x = A^{-1}b. \qquad \textbf{(X)}$$

Using the Matlab command 'inv' generates the explicit inverse of the square matrix while the Matlab command '\' will perform matrix division by Gaussian elimination. Both functions provide warning messages if the matrix being inverted is nearly singular or badly scaled, but the '\' matrix operator will produce residuals which are typically several orders of magnitude less than those produced from using the 'inv' command on the same matrix. Also, using the '\' command typically produces its results in substantially less time than using the 'inv' command. For more information on the algorithms used in the 'inv' and '\' commands, refer to [9].

As may be seen from the previous discussions, a great deal of information has been derived from the original RGB image. After its conversion to the YIQ or HSV color space to generate grayscale values, positive and negative sloped dark to light and light to dark transitions were detected. Color based characteristics were isolated to supplement discrimination. Before each detected edge could be admitted to the GEM with a unique identifier, it had to first pass first level discrimination. Next, each of the edge transitions within the GEM had to undergo discrimination based upon a combined structural (syntactic) and statistical pattern analysis to distinguish the non-candidate marker edges from the candidate marker edges. Then, each undiscriminated border generated a slope approximation which potentially contributed to an overall slope approximation for each associated positive and negative sloped longitudinal lane marker. Other methods (besides averaging) could also be applied to supplement the 'overall slope approximation' process. With these slope approximations and a relevant point associated with the slope approximation, four linear equations could potentially be generated. Of these four linear equations, two corresponded to the positive sloped complementary border transitions and two corresponded to the negative sloped complementary border transitions. With these four equations, two sets of coupled equations could be used to generate the intersecting dashed red projections shown in many of the previous figures.

It is evident from the previous analysis that a great deal of context derived content may be extracted from a road scene image. Detecting a potential lane boundary resulting from a likely longitudinal lane marker and the relationships between likely lane boundaries in a road scene image has been introduced. Relating this two dimensional image plane information to the three dimensional characteristics of a road scene provides the basis for approximating the location of a vehicle with respect to the likely lane boundaries. This may then contribute to the differentiation between being 'within a lane' and having 'departed a lane'.

9. PRELIMINARY STANDARDS INFORMATION

9.1 Introduction

Up to this point, the main focal point has been on the analysis of longitudinal markings leading to the calculation of forward and rearward projections for a road scene image. The 2003 version of the MUTCD defines a *longitudinal marking* as [10]:

"pavement markings that are generally placed parallel and adjacent to the flow of traffic such as lane lines, centerlines, edge lines, channelizing lines, and others."

From the previous definition, a longitudinal lane marking may delineate the separation of traffic lanes, the right or left edge(s) of the traveled way, etc.

It also defines a *traveled way* as [10]:

"the portion of the roadway for the movement of vehicles, exclusive of the shoulders, berms, sidewalks, and parking lanes,"

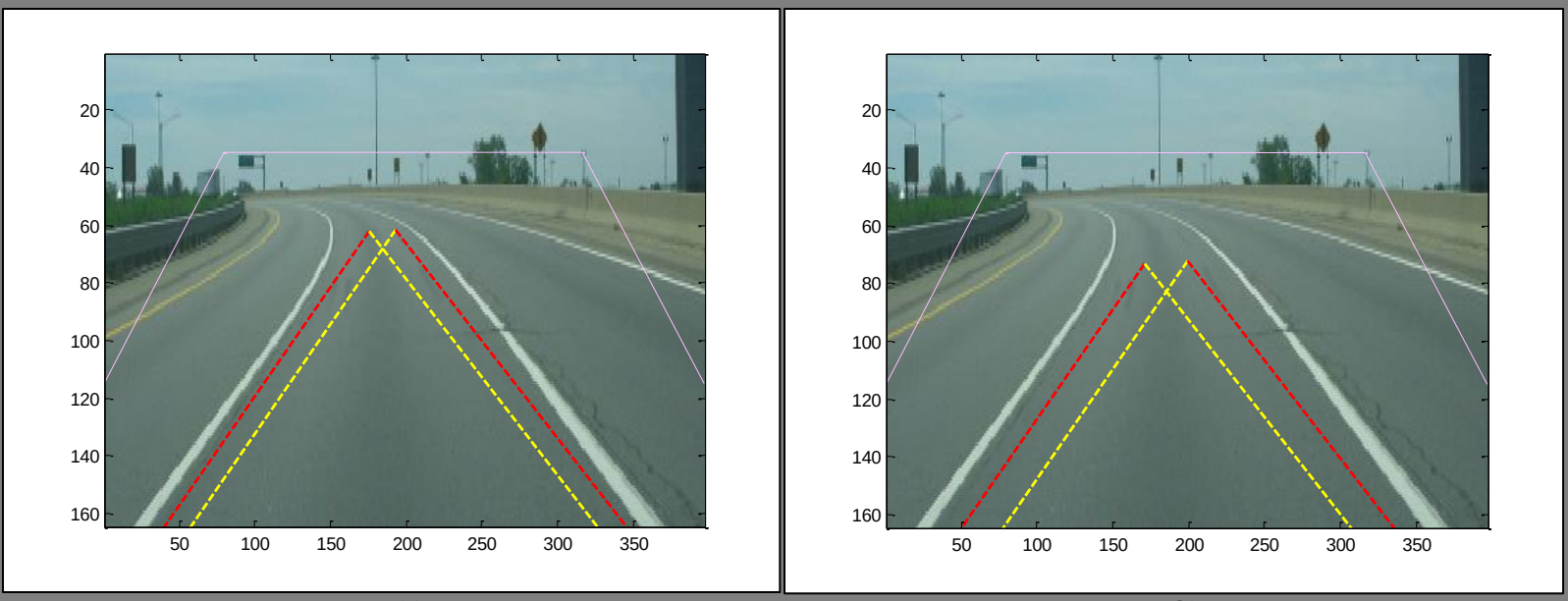
and a *highway* as [10]:

"a general term for denoting a public way for purposes of travel by vehicular travel, including the entire area within the right of way".

In many accidents which occur where the position of the vehicle relative to the lane boundary plays a significant role, the idea of 'entire area within the right of way' may come into question. **This paper has not and will not attempt to interpret where a vehicle should or shouldn't be relative to the longitudinal markings occurring within a road scene image.** However, in the course of extracting the forward and rearward projection information for a particular road scene image, it was first necessary to consider some preliminary regulatory information from documents including (but not limited to) the 2003 MUTCD pertinent to longitudinal marker widths, patterns, colors, etc. Similarly, to begin to ascertain how a 'traveled way' may relate to those longitudinal markings encountered, it is first necessary to establish what being 'within a lane' constitutes and when not being 'within the lane' constitutes a 'lane departure'. To assist in preventing unintended lane departures, warning indicators may be used to alert the vehicle operator of a pending lane transition. Various standards have been initiated to provide further information related to lane departure warning systems (LDWS).

9.2 ISO 17361:2007

The first international standard covering Lane Departure Warning Systems is ISO 17361:2007 entitled "Intelligent transport systems - Lane departure warning systems - Performance requirements and test procedures". It covers subjects including (but not limited to) terms and definitions, specifications and requirements, and test methods. Some of the potentially relevant topics introduced in this document include (but are not limited to) curvature and velocity based classification, lateral position and lane departure velocity, time to line crossing (TTLC), earliest and latest warning lines, and three test procedures including warning generation, repeatability, and false alarm. Figure X (a) and (b) shows a road scene image containing left and right white normal longitudinal markers where the forward and rearward projection lines have been modified from those typically shown in the other previous road scene images.



(a) (b)

Figure X: (a) Sample road scene image with modified projection lines; (b) Same road scene image with different projection lines

While the 2003 MUTCD contains no formal definition specific to lane boundary, it does use the phrases ‘separation of traffic lanes’, ‘separation of traffic flows’, and ‘edge of the roadway’ on many occasions when discussing markings for longitudinal lines. ISO 17361:2007 defines a *lane boundary* as [18]:

“borderline of the lane, situated at the center of the visible lane marking or, in the absence of a visible lane marking, determined by incidental visible road features or other means such as GPS, magnetic nails, etc.”

Several interesting facts should be noted. The first is that the rearward projections shown in many of the road scene images might potentially be used to establish positional relationships between a likely lane boundary location and the location on the vehicle where the camera attached to the rear view mirror. The second is that there is a distinct relationship between the location of the vehicle’s rear view mirror and the location of significant points on the vehicle’s front end. The third is that there exists a unique relationship between the dimensions of a vehicle’s front end and the pixel positions used to represent it within the image plane.

It is also important to note that this standard does not cover warnings issued regarding collisions with other vehicles nor does it cover taking vehicle control to prevent departures. Similarly, warnings occurring on roadways with temporary or irregular markers are not covered by the standard. Standards information may be used for evaluating the performance of a lane departure warning system. A graphical user interface is a common way of displaying intermediate results of the image processing tasks being performed on a road scene image and is discussed in the next section of this paper.

10. CREATING A GRAPHICAL USER INTERFACE IN MATLAB FOR THE PRELIMINARY ROAD SCENE IMAGE PROCESSING SYSTEM

Graphical user interfaces (GUI's) may be desirable for a number of reasons when performing image analysis in Matlab. The first reason is that Windows based applications have become a very common environment for performing signal analysis. This environment is highly intuitive and easily usable with little or no training in Windows programming. Another reason is that the GUI facilitates some of the important features of object oriented programming including object classes, inheritance, and function overloading. The Matlab GUI development environment (GUIDE) provides features such as object handles and methods to graphics objects for 'safely and effectively' operating on the objects within the user interface. A third reason is that the GUI provides a cohesive environment for performing the various activities common in image processing. A fourth reason is that a correctly designed user interface will prevent incorrect usage of the image processing software by allowing only a systematic sequencing of steps to exercise the software.

The preliminary user interface was designed with a number of goals in mind. The first was that it would be able to provide a wide range of functionality in an easy to use fashion. Thus, the selection of the image, the analysis on the selected image, and the display of the output resulting from the analysis would need to be straightforward. It was decided that the selection of the various road scene images would be performed using a popup menu because one of its important features is a value index which may be used to designate which of the individual JPEG images from the long sequence had been selected. Since a large number of road scene images were acquired from a limited number of different roads, the road name (or a general designator of) where the road scene image was acquired from would serve as a useful key for gaining access to the set of images for that specific road designation. The functions performed on the images including those relating to preprocessing, image segmentation, and contextual analysis were incorporated into a menu system because windows based GUI applications typically contain menus for the control of processing. These menu options allowed both single operations and groups of operations to be performed on a single image. Another reason for incorporating a menu system is that Matlab provides a Menu Editor which allows creation of context and windows menus. A menu option was added for performing file input and file output, two common tasks required for the manipulation of raw data. Finally, menu options were added to enable the display of information in various color space formats including RGB, HSV, and YIQ as well as the display of specialized road scene plots. An example of a potential user interface for the Preliminary Road Scene Image Processing software is shown in figure X.

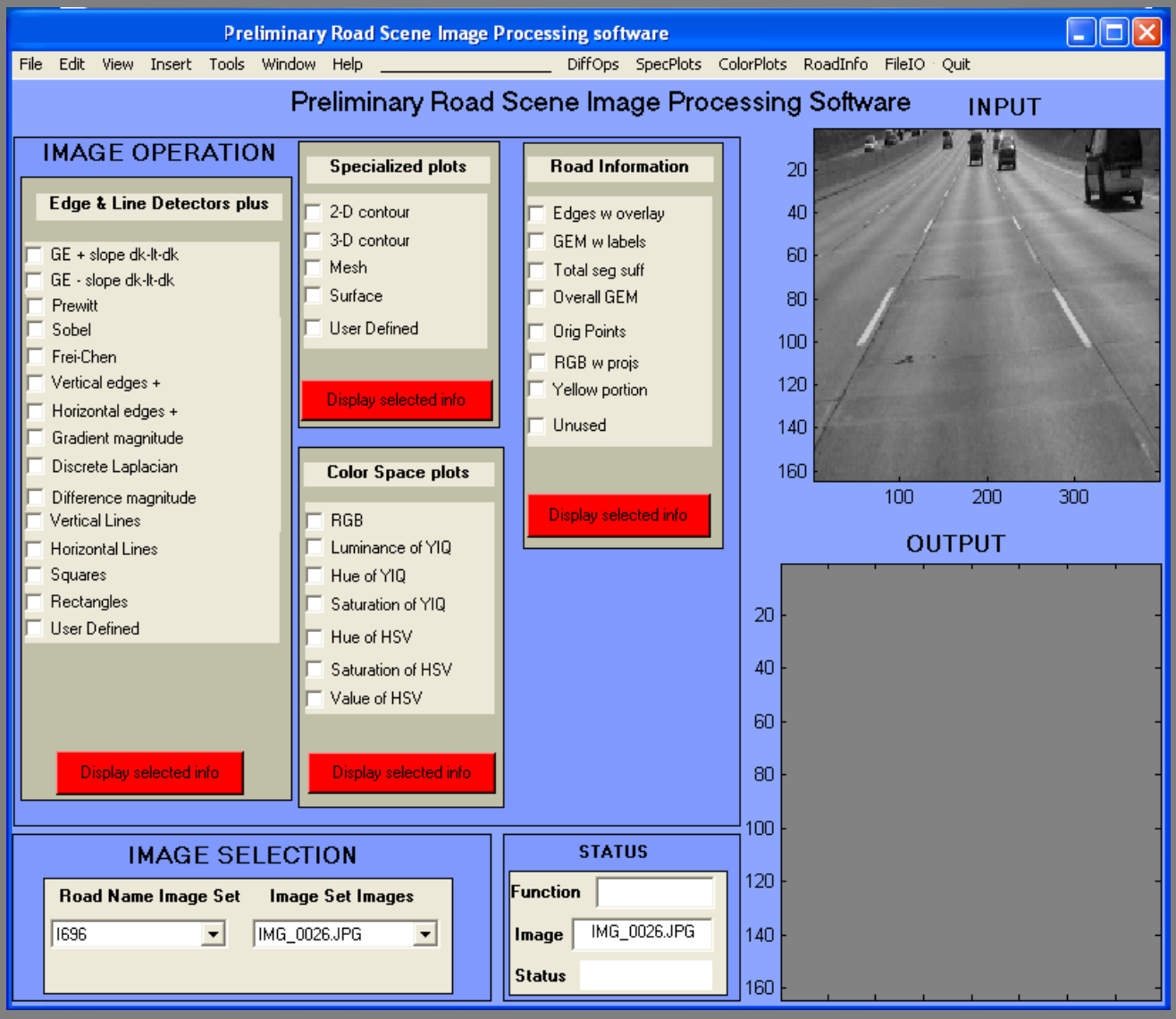


Figure X: A potential user interface for the Preliminary Road Scene Image Processing Software

As may be seen in figure X, there are two main portions to the menu system. One is for the standard 'file menu' provided for in Matlab and the other is for the various functions (or groups of functions) which may be applied. The two menu portions are separated by a long straight horizontal line at the top of the menu. The GUI portions for 'IMAGE SELECTION' and 'IMAGE OPERATION' are shown in their own frames and a small section for status information also has its own frame. This status information will show the name of the current JPEG image selected ('Image'), the last operation applied to the current image ('Function'), and a status ('Status') showing whether or not processing is currently being performed.

Use of the Preliminary Road Scene Image Processing software is initiated by selecting a road name and then an image from the 'IMAGE SELECTION' portion of the GUI. In figure X, the image 'IMG_0026.JPG' is selected from the I696 road image set. No processing is allowed until an image has been selected. The grayscale image prior to any analysis is displayed in the upper right hand corner of the GUI under the textbox labeled 'INPUT'. The menu bar option 'RoadInfo' leads to a pulldown suboption 'Extract_Road_Info' which produces the GUI output displayed in figure X.

Figure X: A potential user interface for the Preliminary Road Scene Image Processing Software

A unique feature of the GUI indicates completion of an operation with a 'red background ' and 'selected checkbox' within the 'IMAGE OPERATION' portion of the user interface. Selection of multiple operations from the 'DiffOps', 'SpecPlots', 'ColorPlots', or 'RoadInfo' menu options would result in red background selected checkboxes occurring in one or more of the four distinct image operation frames ('Edge & Line Detectors plus', 'Specialized plots', 'Color Space plots', and 'Road Information', respectively). Each of the selected outputs may then be displayed in a separate figure by choosing the 'Display selected info' button within a particular frame. To deselect an available chosen operation from being displayed in a separate figure, it is possible to simply deselect the checkbox by clicking within the white square. An example of how the selected red checkboxes will display is shown in figure X. Note that 'Extract_Road_Info' is the only menu option which will perform multiple operations on a single image, resulting in multiple red background selected checkboxes being shown. Also note that the last operation performed will have a descriptive name shown in the 'Function' status box and its resulting image shown under the textbox labeled 'OUTPUT' in the lower right hand corner of the GUI. Additionally, while processing is being performed, the 'STATUS' frame will show the status message 'Processing' until the current operation is complete. After the processing of the 'Extract_Road_Info' is completed,

selecting the 'Display selected info' pushbutton produces a separate figure containing relevant image information which has been extracted from the road scene image. Figure X shows the separate figure and results for image number 'IMG_0111.JPG' from the 'Various_local' road set after having selected 'Extract_Road_Info' from the menu system and clicking on the 'Display selected info' pushbutton.

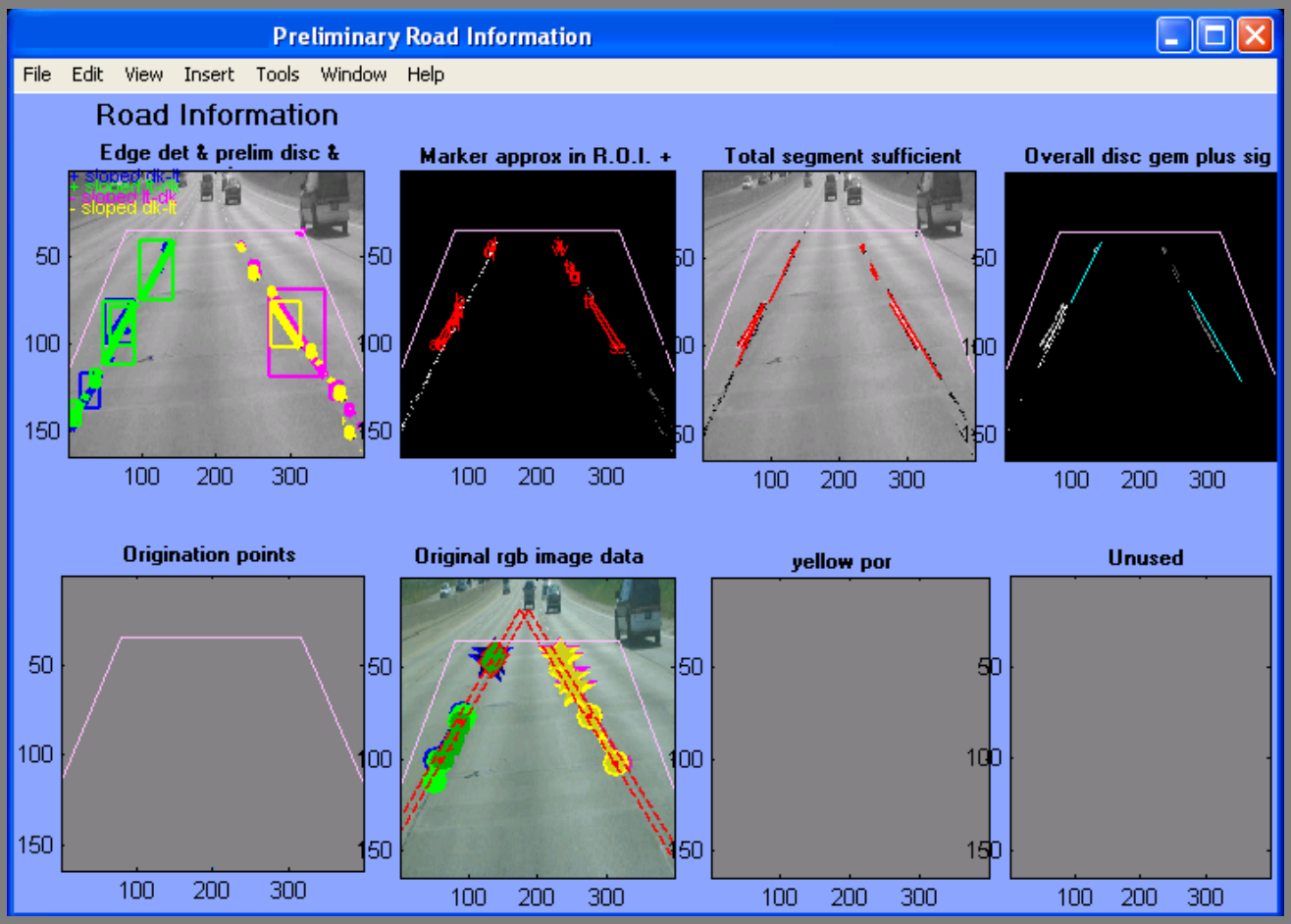


Figure X: Preliminary road information generated by selecting 'Display selected info' from main menu

Note however that the 'Preliminary Road Information' figure doesn't contain much relevant information due to the fact that there have been no forward and rearward projections obtained from within the ROI.

11. REFERENCES

[1] Canny, John, "A Computational Approach to Edge Detection", IEEE Transactions on Pattern Analysis and Machine Intelligence, VOL. PAMI-8, NO. 6, November 1986.

[2] Jähne, Bernd, *Practical Handbook on Image Processing for Scientific Applications*, 2nd edition, CRC Press, 2004.

[3] Matlab's Image Processing Toolbox, The Mathworks.

[4] Haralick, Robert M., and Shapiro, Linda G., *Computer and Robot Vision*, volume I, Addison-Wesley Publishing Company, 1992.

[5] "Handbook of Computer Vision and Applications, volume 2, Signal Processing and Pattern Recognition", edited by Jähne, Bernd, Haußecker, Horst and Geißler, Peter, Academic Press, 1999.

[6] Acharya, T., and Ray, A.K, *Image Processing: Principles and Application*, John Wiley & Sons, Inc., 2005.

[7] Fu, K.S., *Syntactic Pattern Recognition and Applications*, Prentice-Hall, 1982.

[8] "Synatic and Structural Pattern Recognition: Theory and Applications", edited by Bunke, Horst, and Sanfeliu, Alberto, World Scientific, 1990.

[9] Dongarra, J.J., Bunch, J.R., Moler, C.B., and Stewart, G.W., LINPACK User's Guide, SIAM, Philadelphia, 1979.

[10] "Manual on Uniform Traffic Control Devices for Streets and Highways", 2003 Edition, U.S. Department of Transportation – Federal Highway Administration.

[11] Najm, W.G., Sen, B., Smith, J.D., and Campbell, B.N., "Analysis of Light Vehicle Crashes and Pre-Crash Scenarios Based on the 2000 General Estimates System", DOT HS 809 573, February 2003.

[12] Sen, B., Smith, J.D., and Najm, W.G., "Analysis of Lane Change Crashes", DOT HS 809 571, March 2003.

[13] "Uniform- Vehicle Code (UVC) and Model Traffic Ordinance", 2000 Edition, National Committee on Uniform Traffic Laws and Ordinances.

[14] Lee, Hsien-Che, *Introduction to Color Imaging Science*, Cambridge University Press, 2005.

[15] "Handbook of Machine Vision", edited by Hornberg, Alexander, Wiley-VCH, 2006.

[16] Duda, R.O., Hart, P., and Stork, D., *Pattern Classification*, 2nd edition, Wiley, 2000.

[17] Federal Register, Vol. 67, No. 147, Rules and Regulations, 2002.

[18] ISO 17361:2007 "Intelligent transport systems – Lane departure warning systems – Performance requirements and test procedures", International Organization for Standardization, February 2007.

[19] Sebe, N., and Lew, M.S., *Robust Computer Vision: Theory and Applications*, Springer, 2003.

[20] Jain, Anil K., *Fundamentals of Digital Image Processing*, Prentice Hall, 1989.

[21] Gose, E., Johnsonbaugh, R., and Jost, S., *Pattern Recognition and Image Analysis*, Prentice Hall, 1996.

[22] Van Der Heijden, Ferdinand, *Image Based Measurement Systems:Object Recognition and Parameter Estimation*, Wiley, 1995.

[23] Haralick, Robert M., and Shapiro, Linda G., *Computer and Robot Vision*, volume II, Addison-Wesley Publishing Company, 1993.

[24] Highlights of major changes to the 2003 MUTCD Publication # FHWA-HOP-4-042, U.S. Department of Transportation – Federal Highway Administration.

[25] U.S. Pavement Markings Publication # FHWA-OP-02-090, U.S. Department of Transportation – Federal Highway Administration.

[26] "A Policy on Geometric Design of Highways and Streets," 2004 Edition, American Association of State Highway and Transportation Officials.

[27] Lay, M.G., "Handbook of Road Technology, volume 2, Traffic and Transport", 3rd edition, Gordon and Breach Science Publishing, 1998

[28] "Practice for Roadway Lighting," RP-8, 2001, Illuminating Engineering Society.

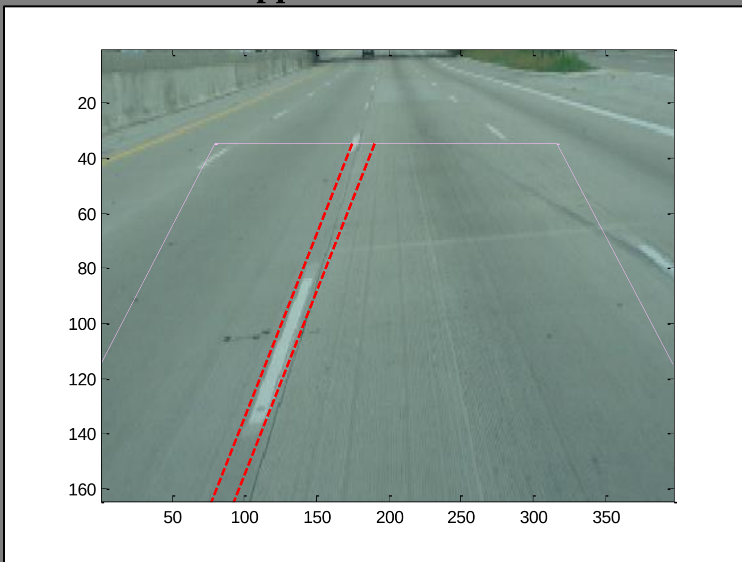
[29] Shapiro, Linda G., and Stockman, George C., *Computer Vision*, Prentice-Hall Inc., 2001.

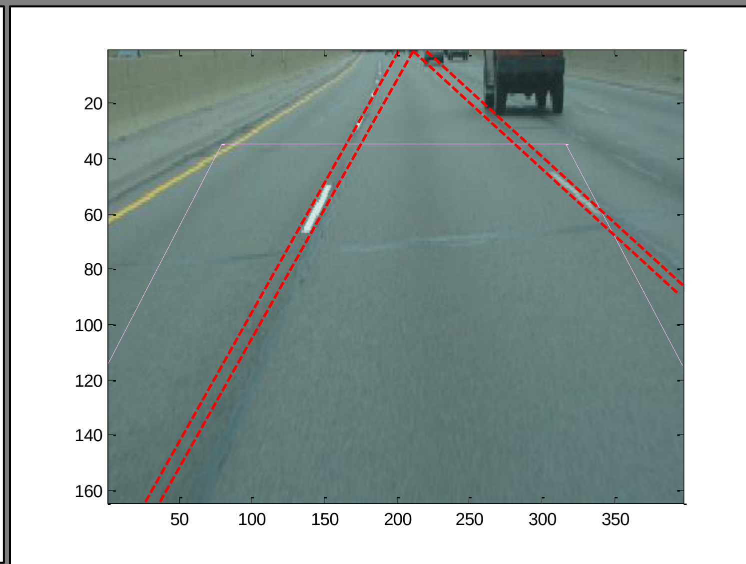
[30] Ray, Sidney F., *Applied Photographic Optics*, 2nd ed., Focal Press, 1994.

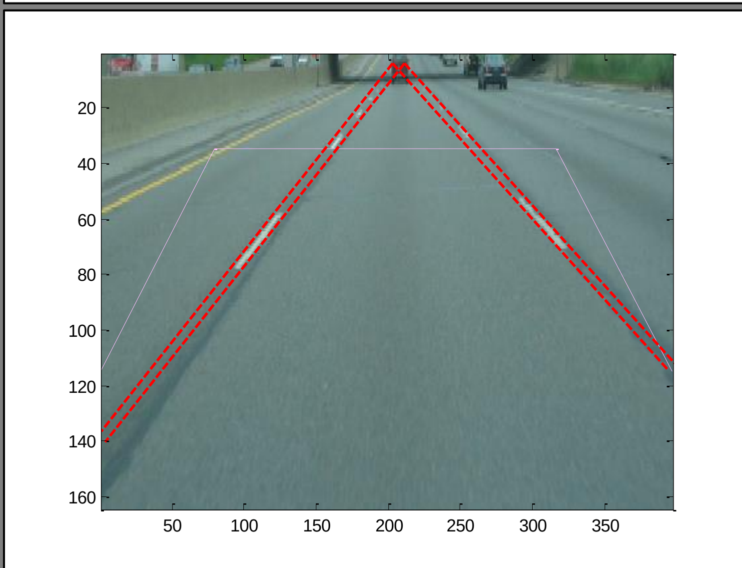
[31] Migletz, J., Fish, J., and Graham, J., "Roadway Delineation Practices Handbook", #FHWA-SA-93-001, August 1994.

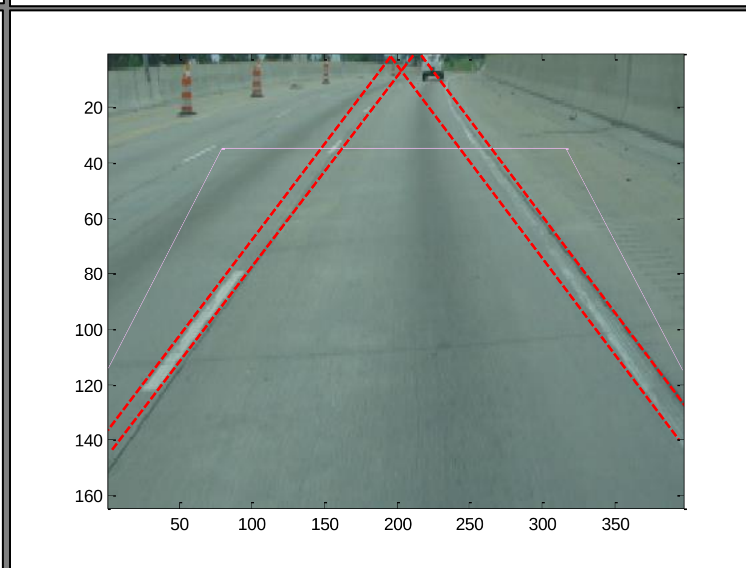
[32] Hawkins Jr., H.G., "Evolution of the U.S. Pavement Marking System", Texas Transportation Institute, College Station, TX., October 2000.

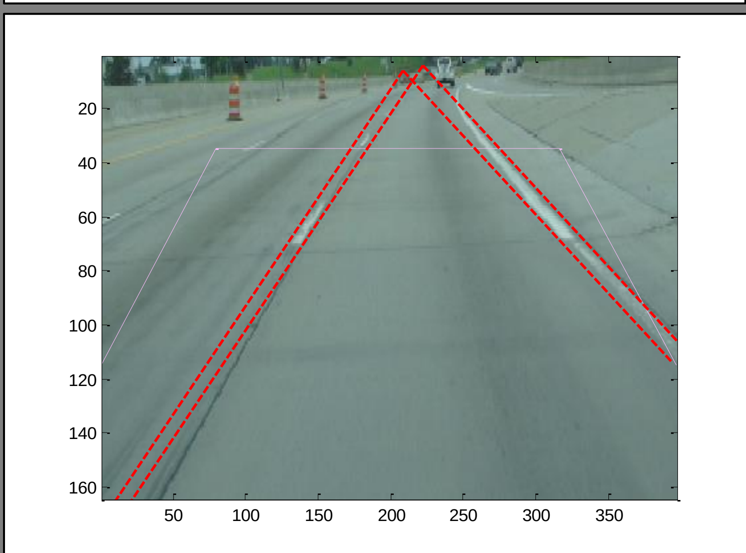
12. Appendix

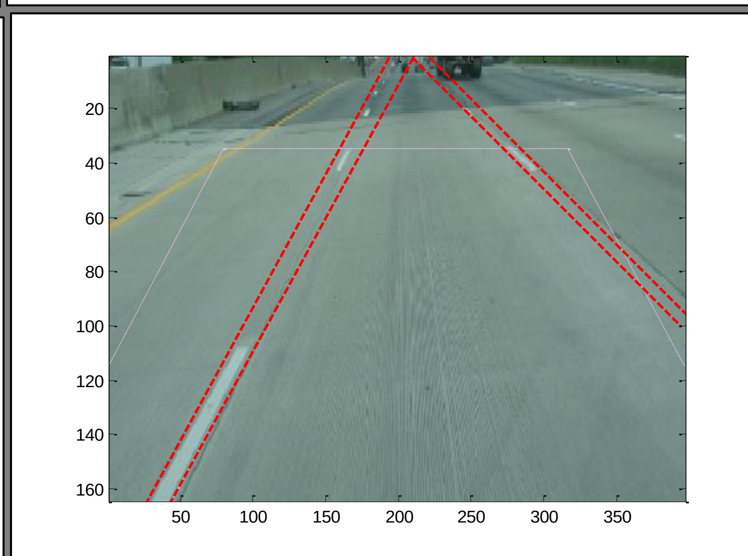


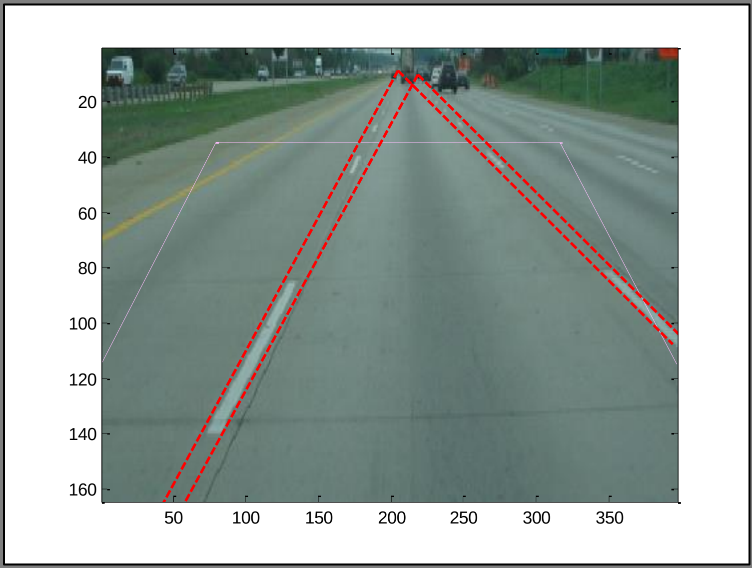


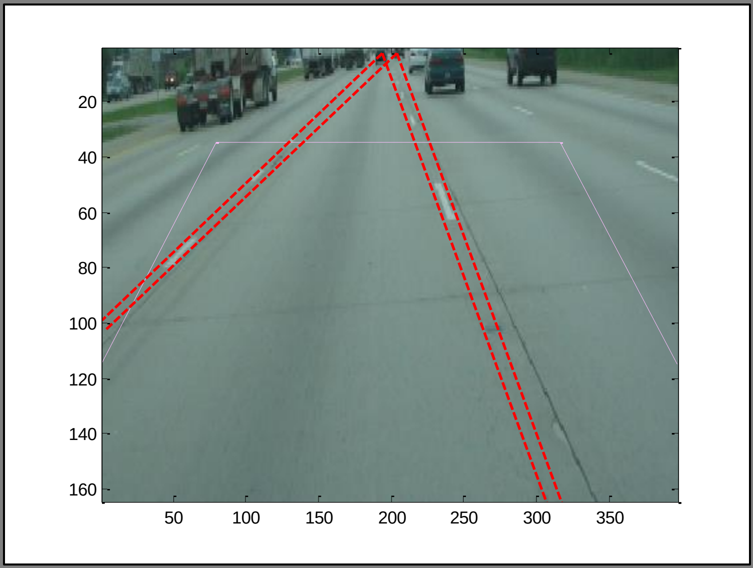


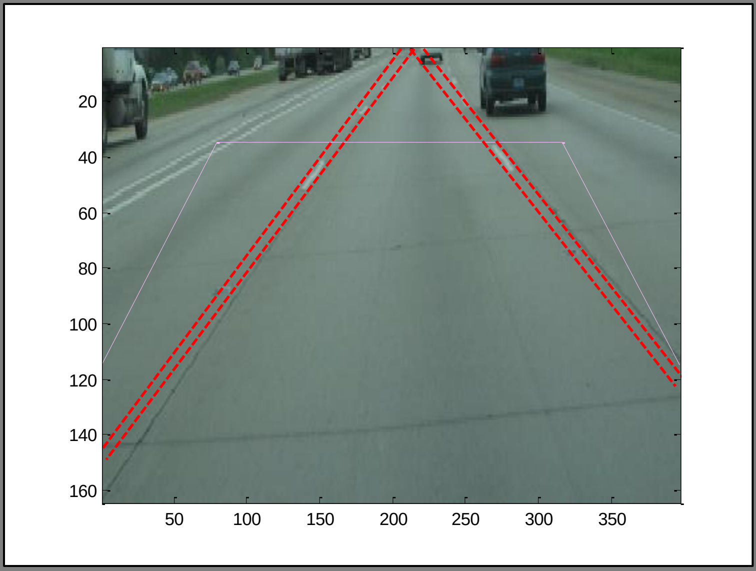


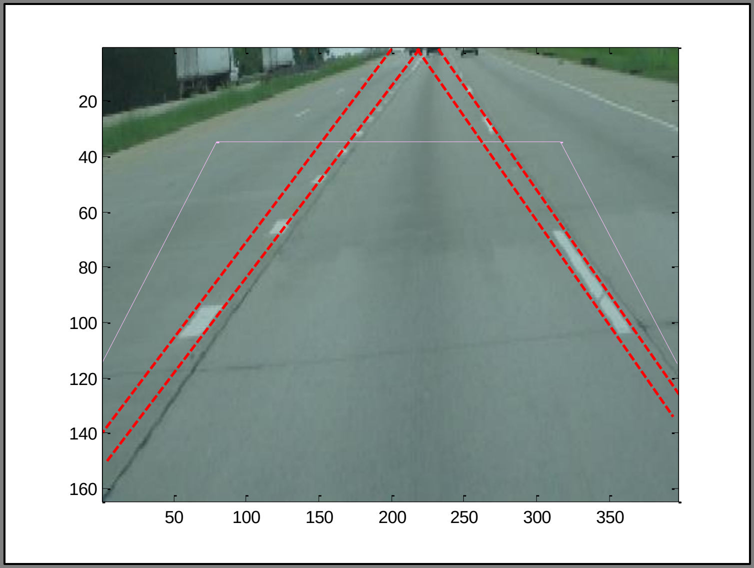


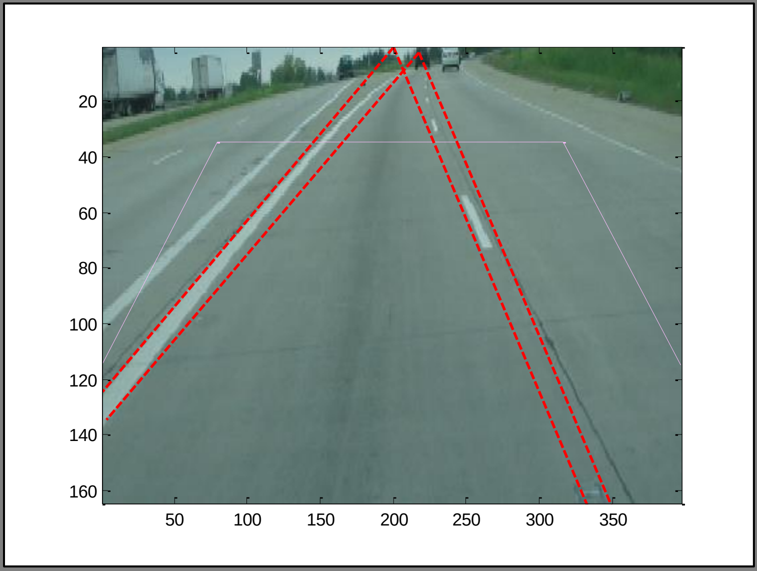


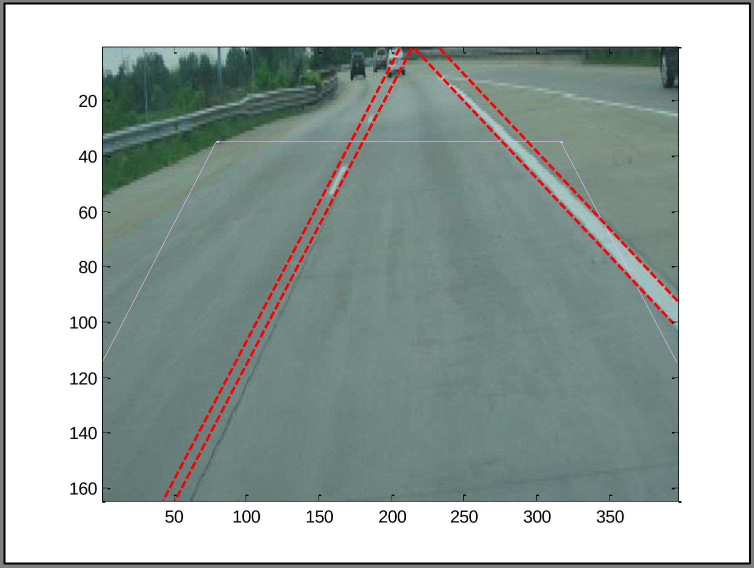




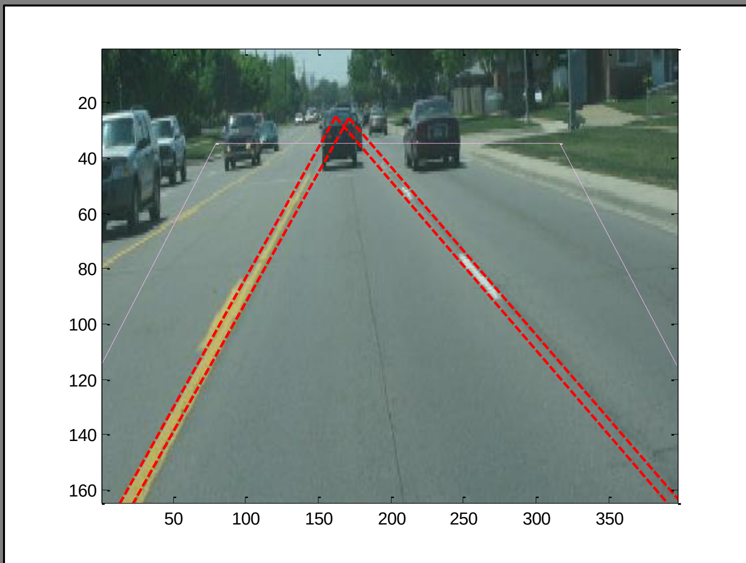


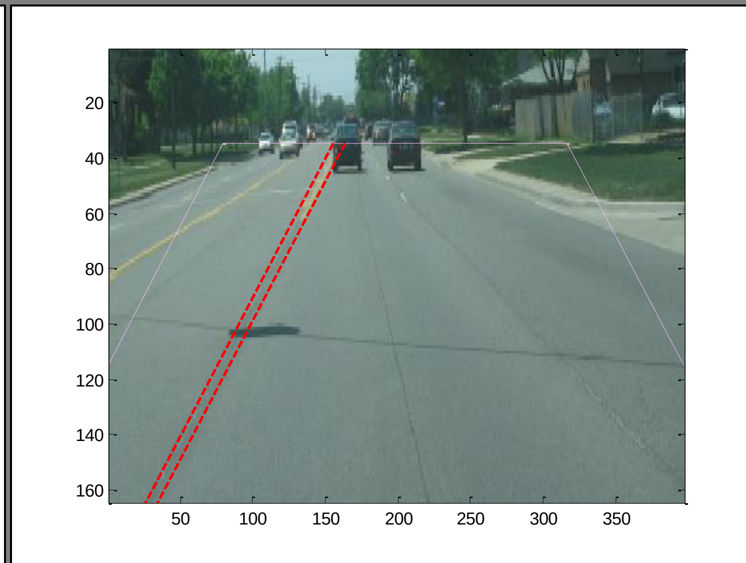


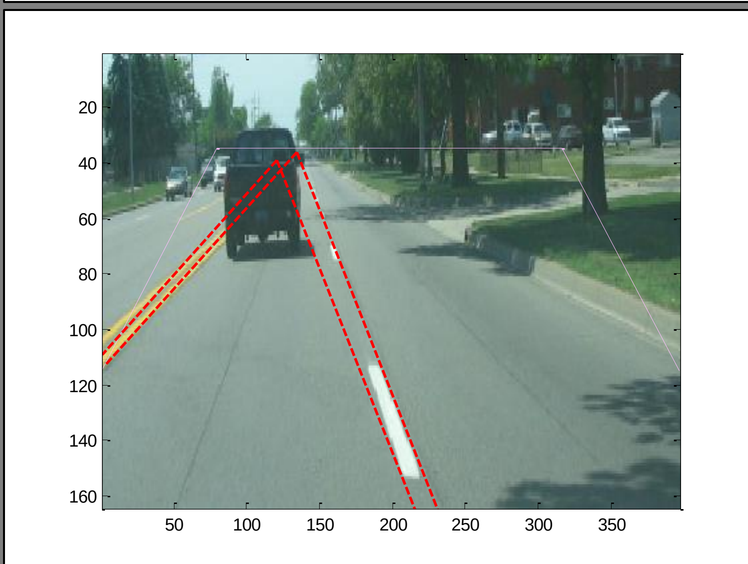


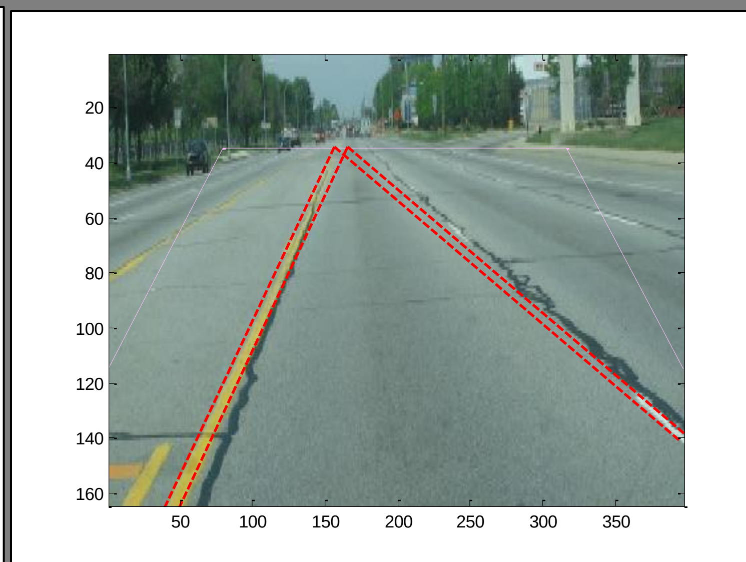


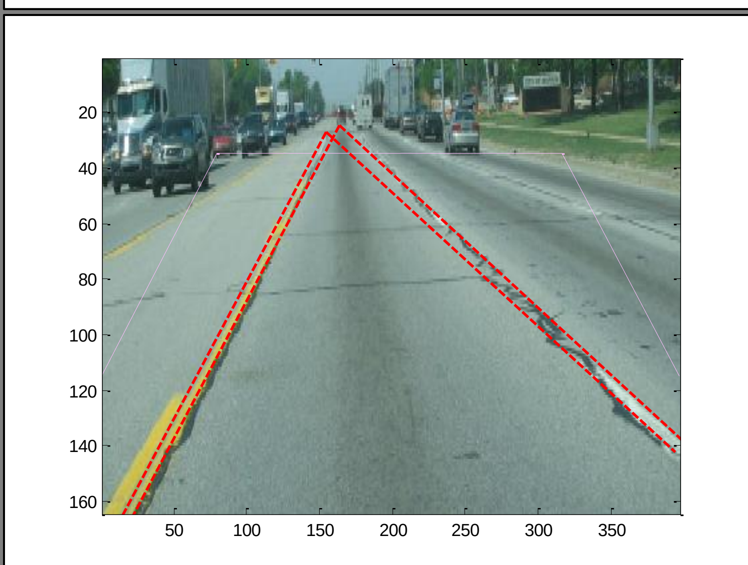
DO NOT COPY

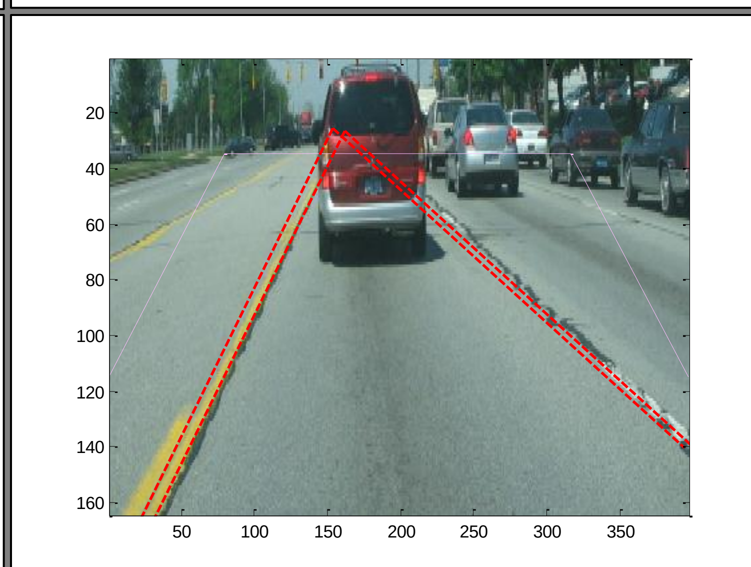


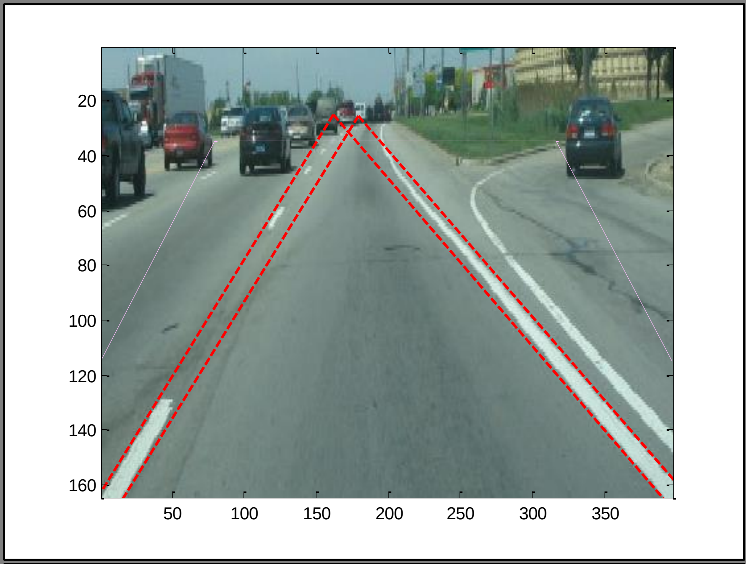


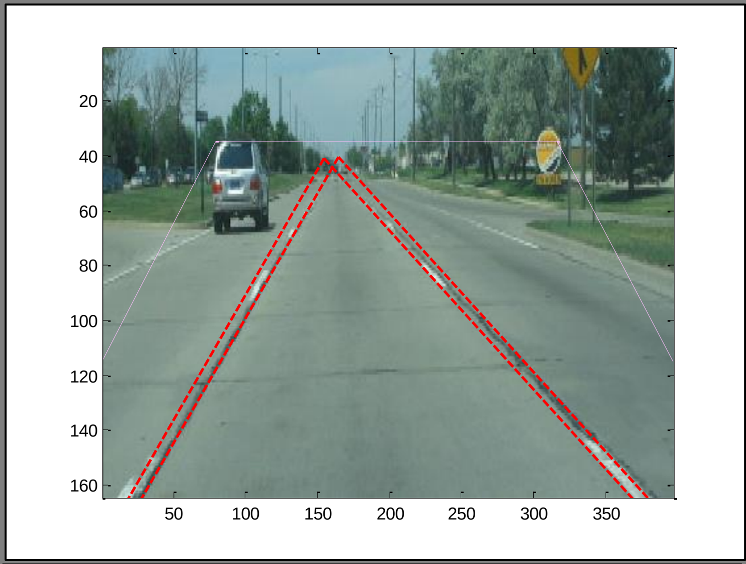


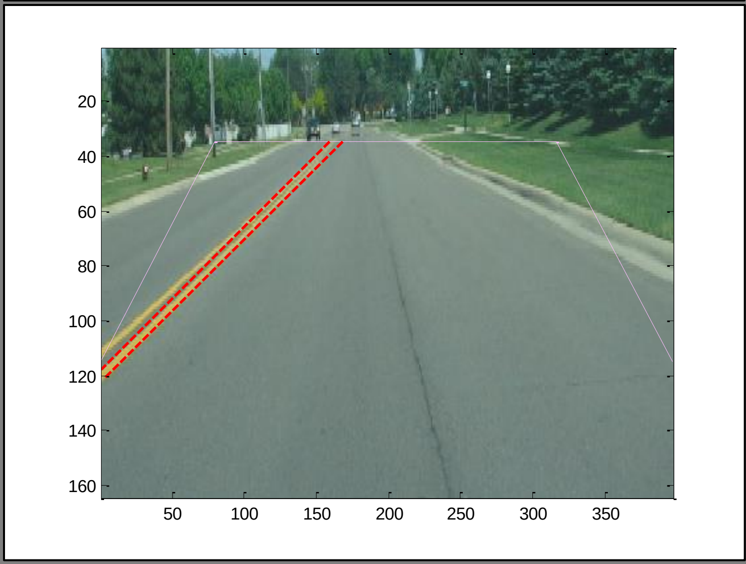


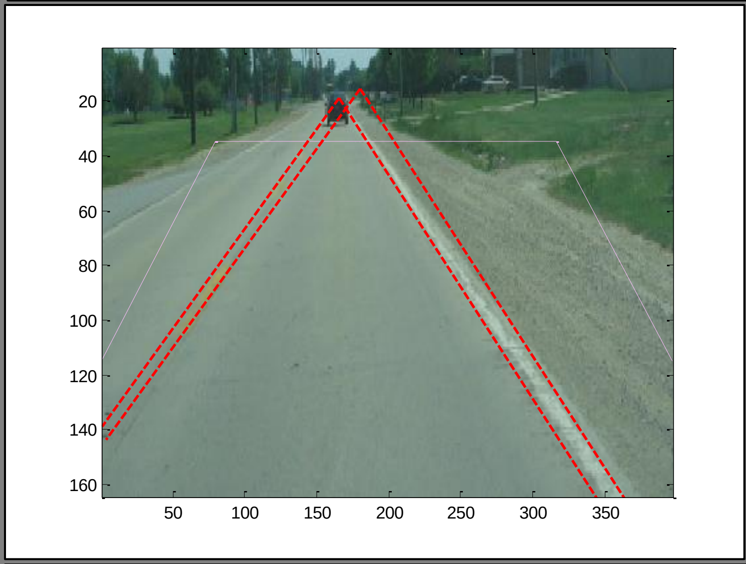


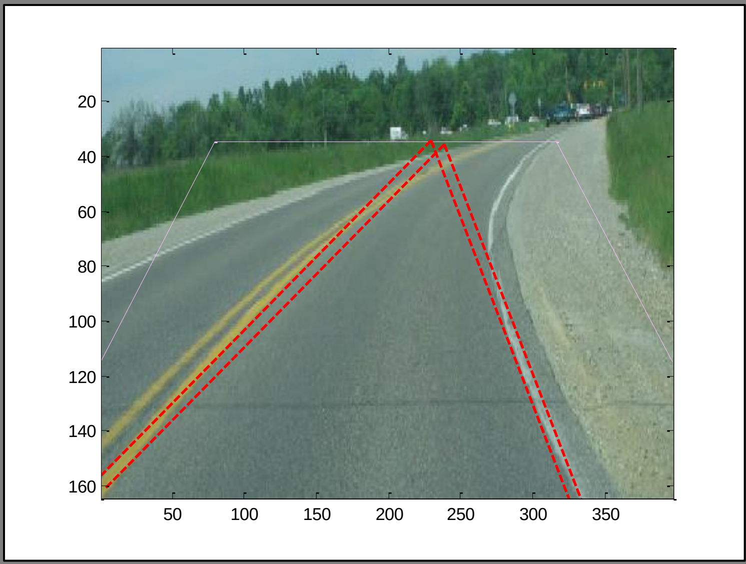


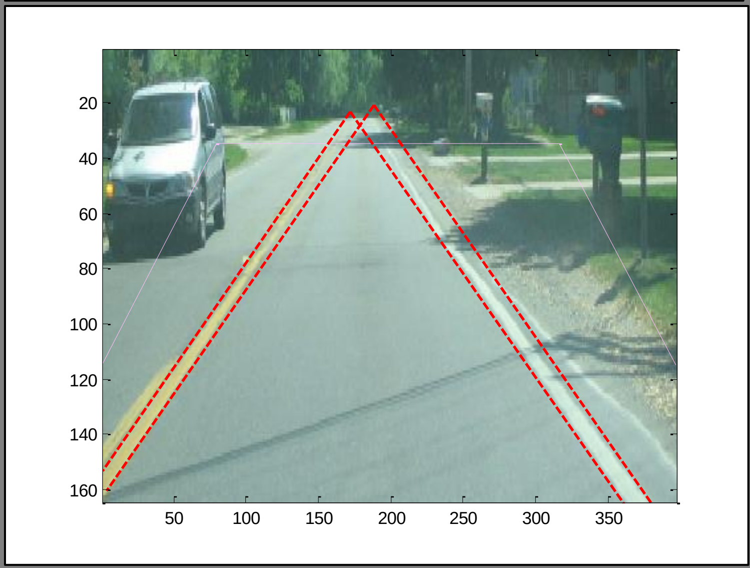


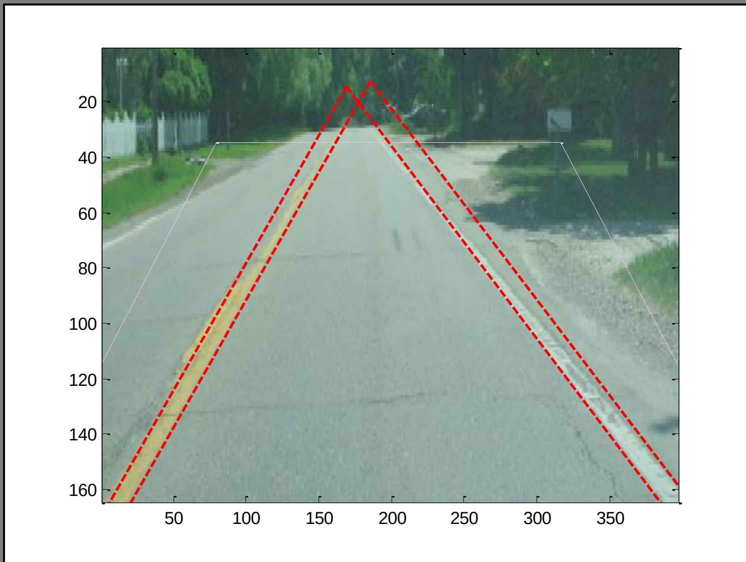


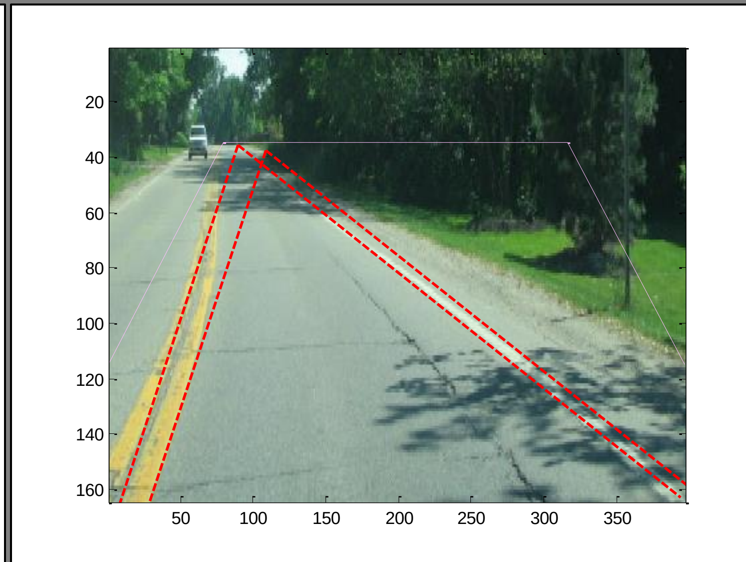


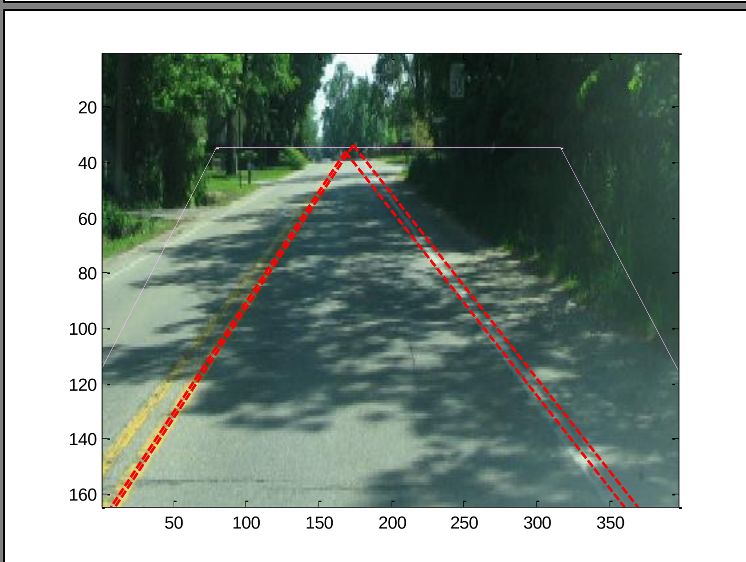


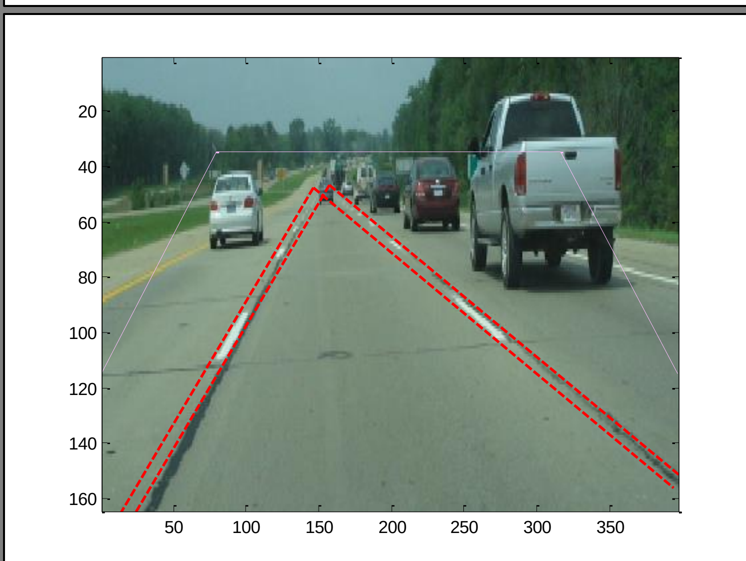


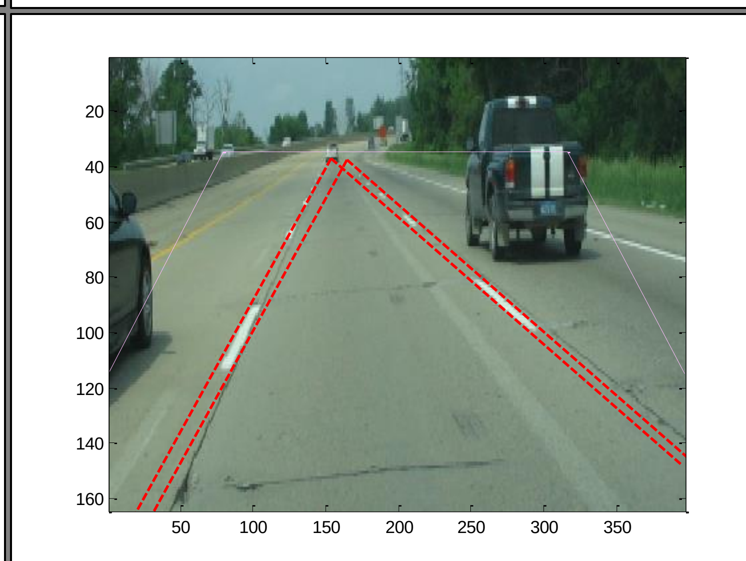


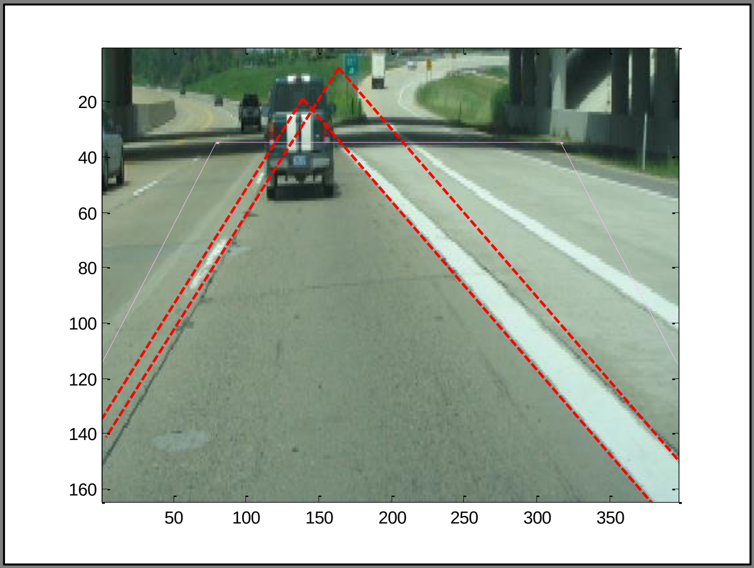


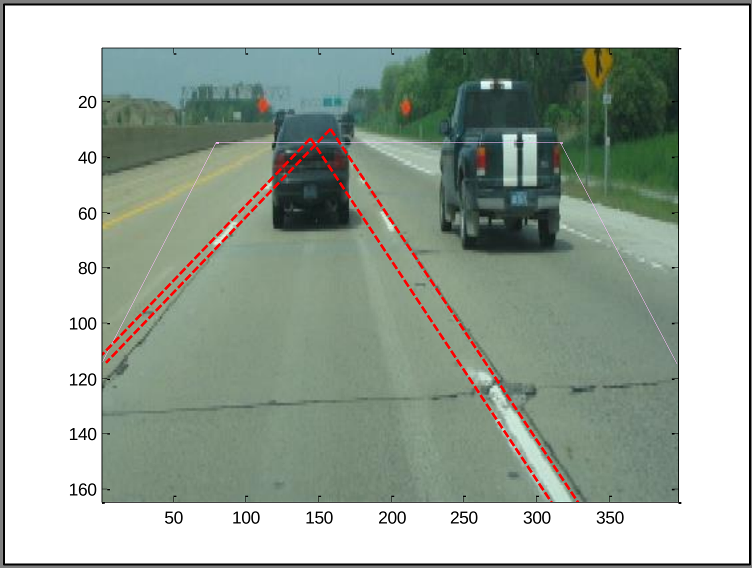


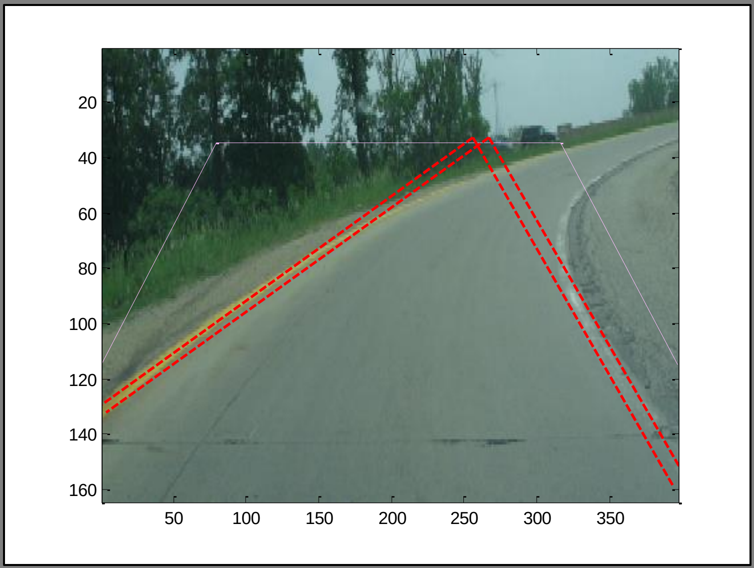


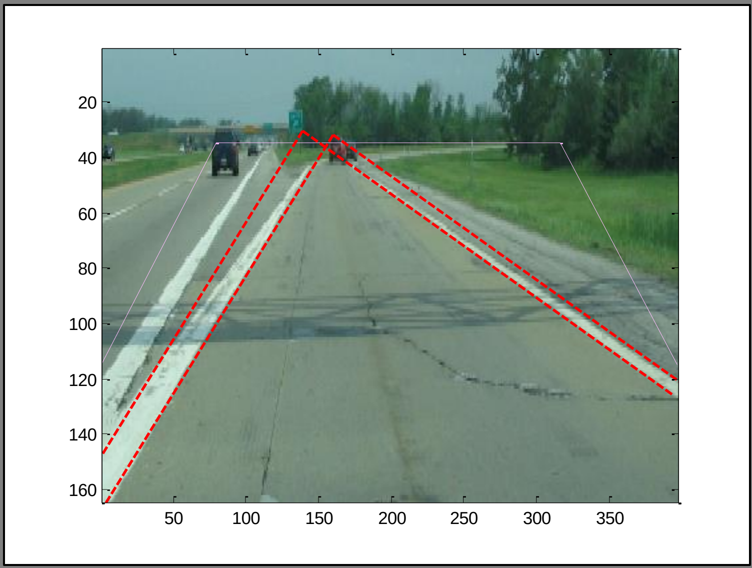


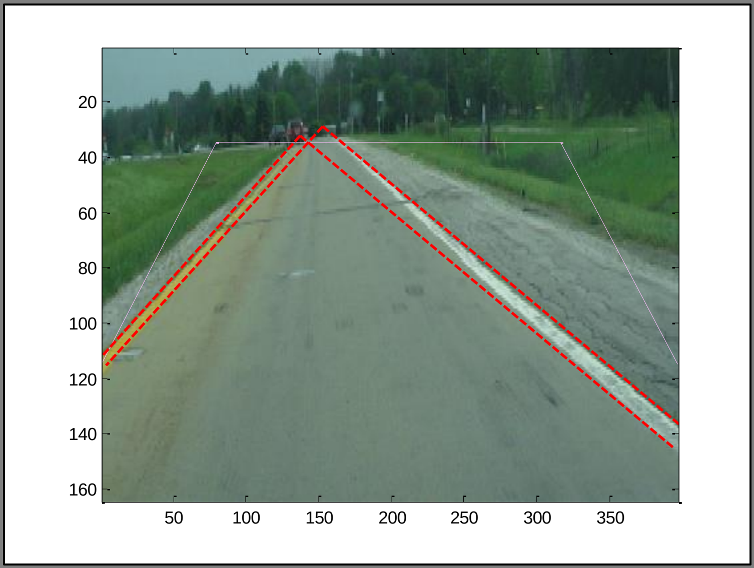


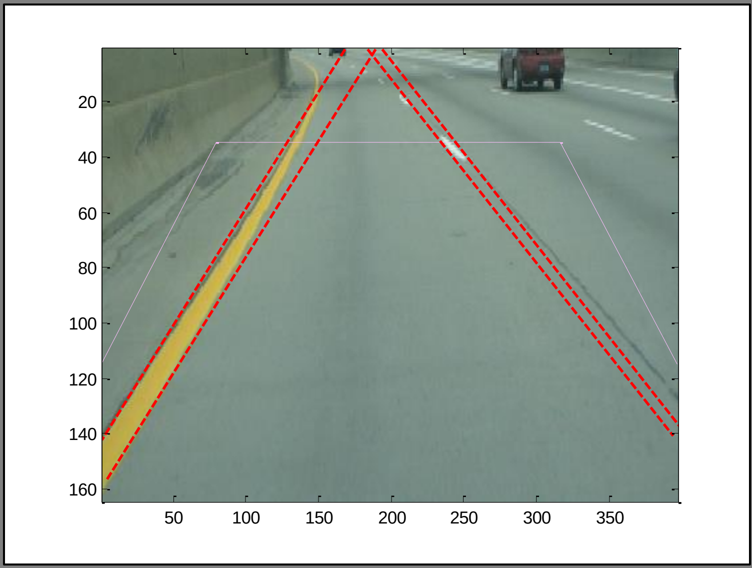


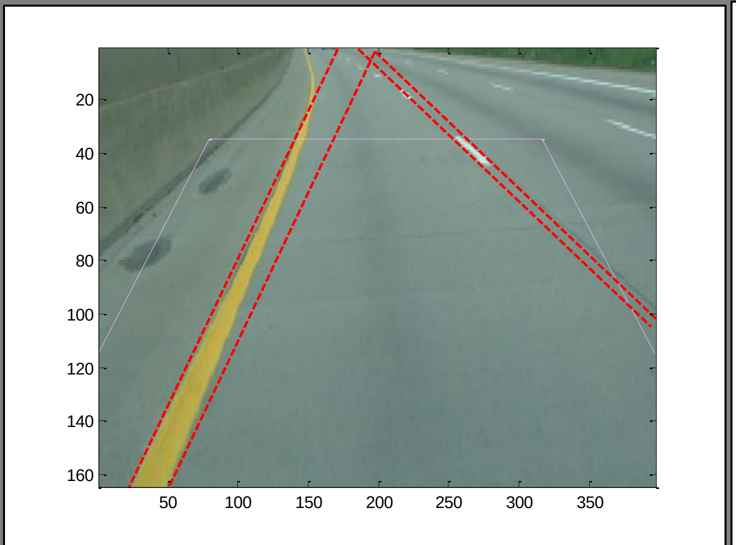


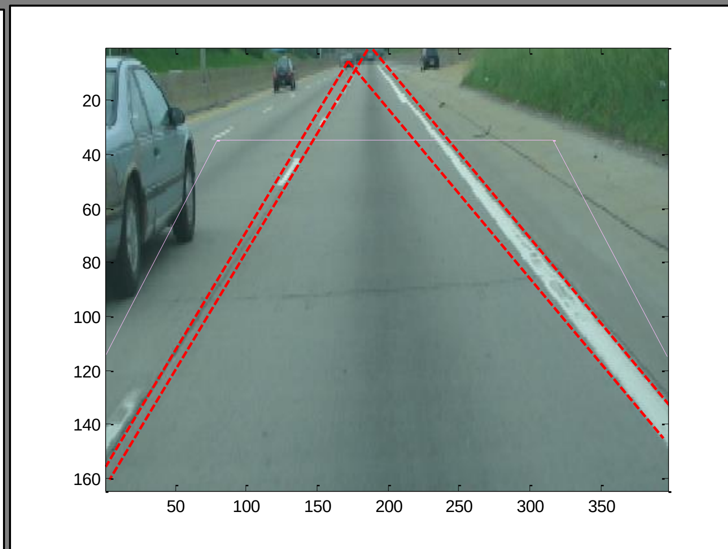


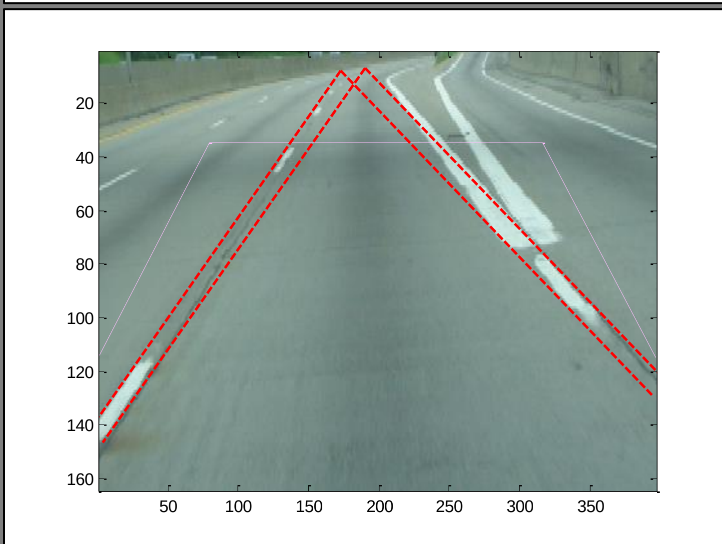


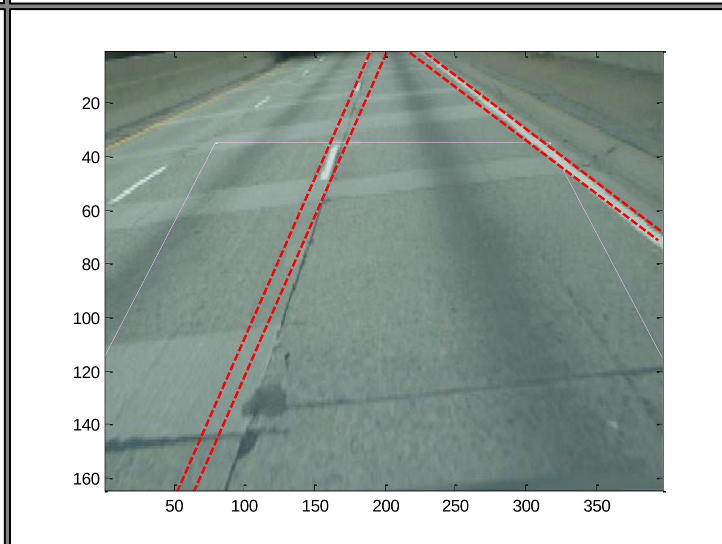


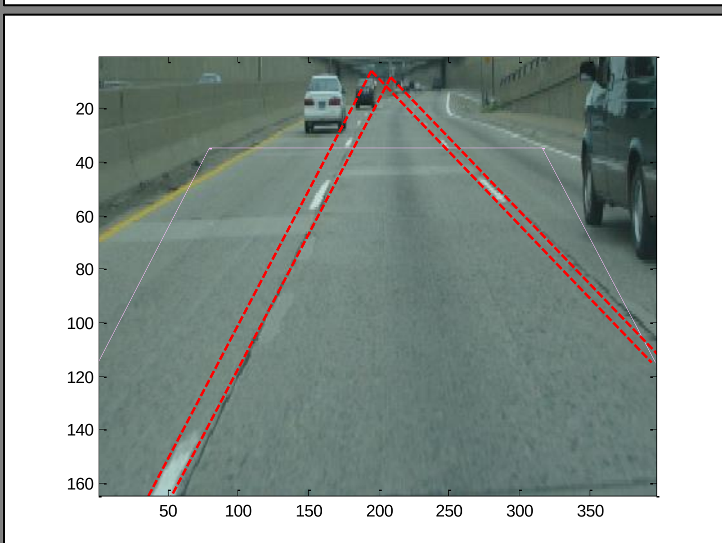


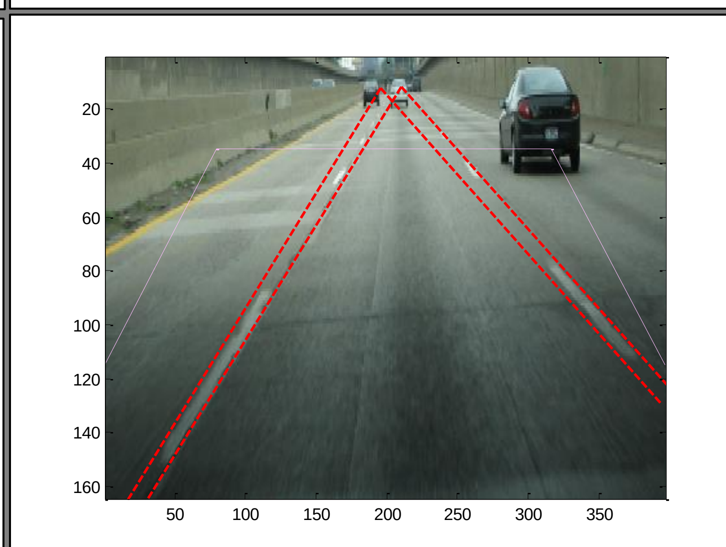




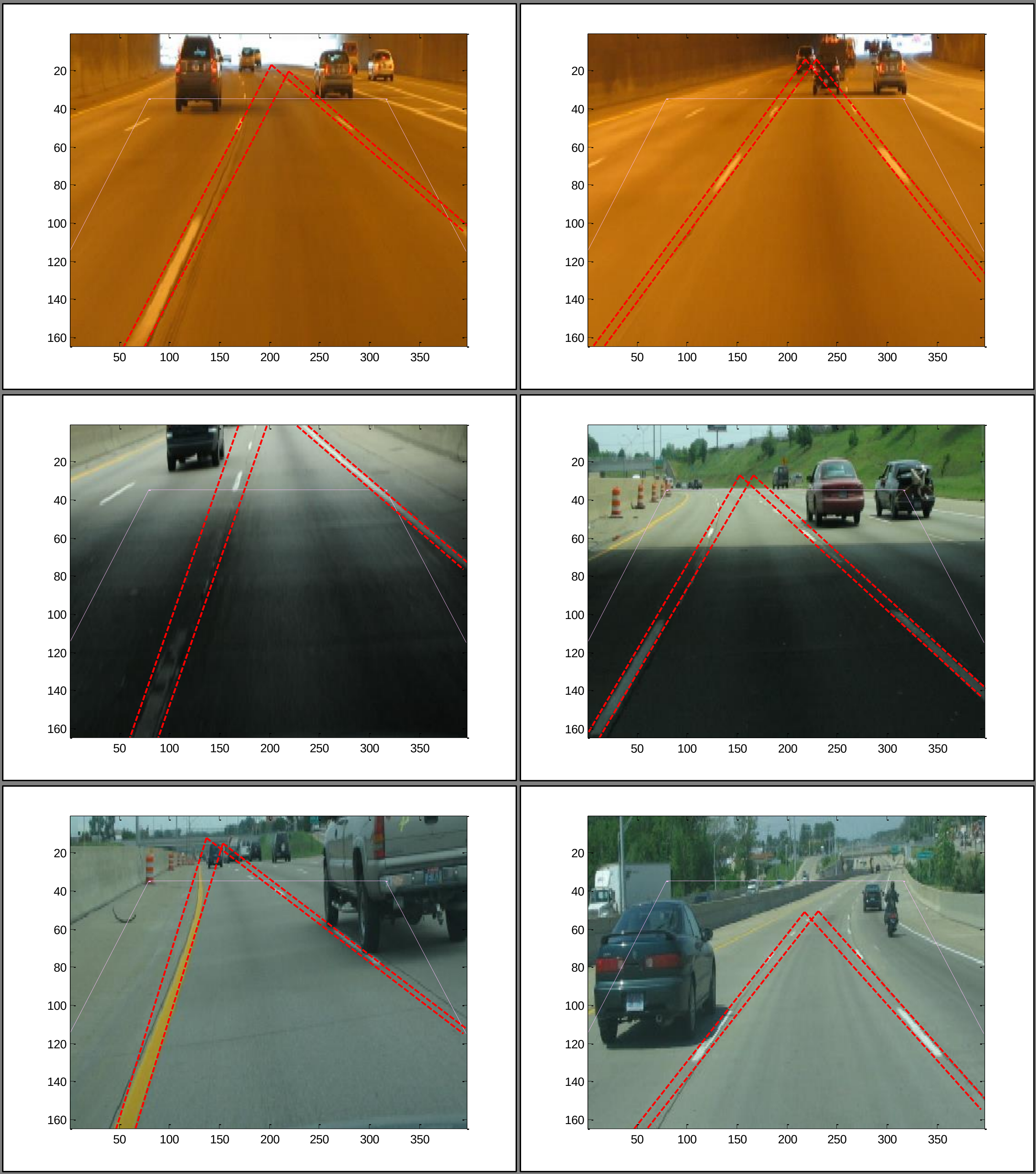


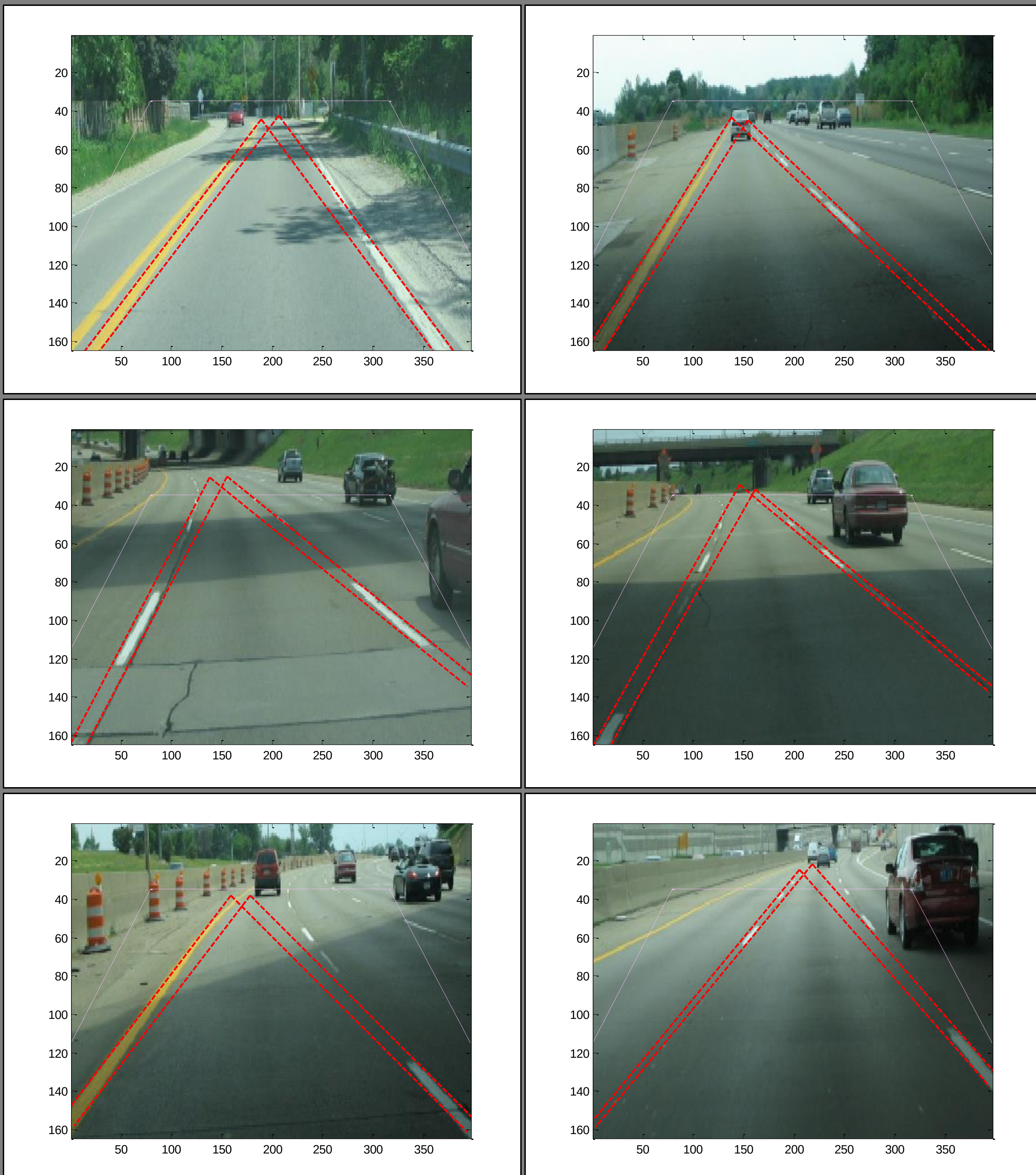


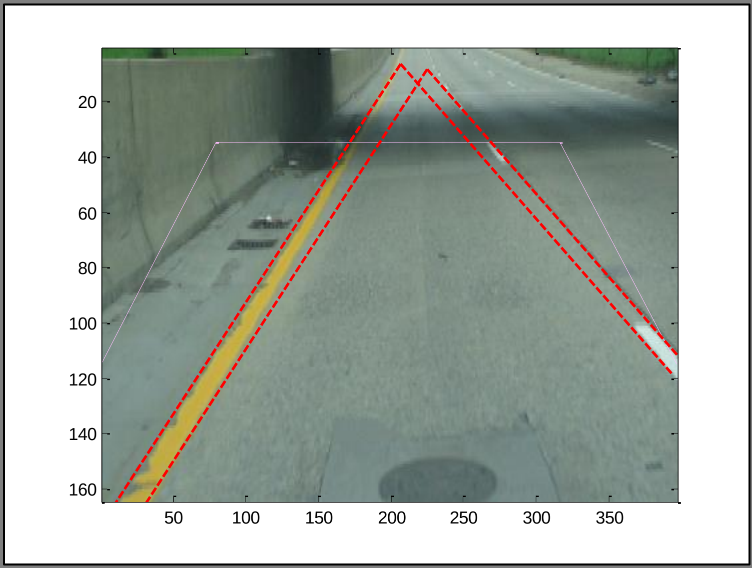


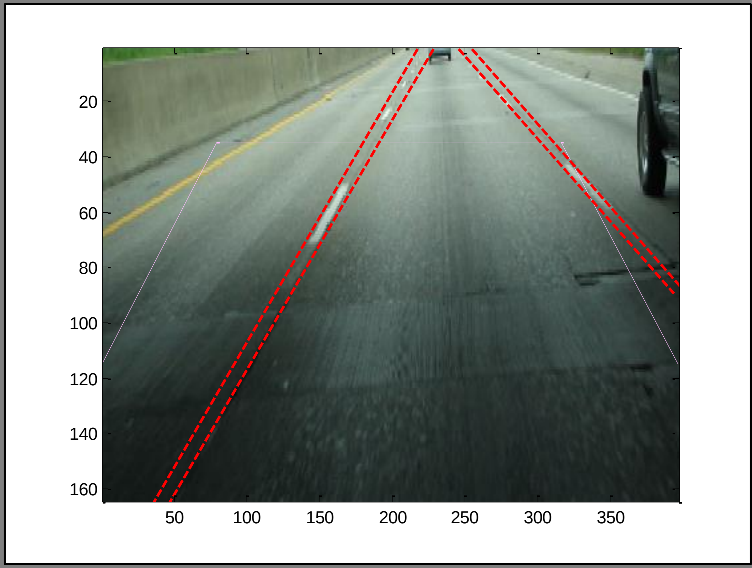


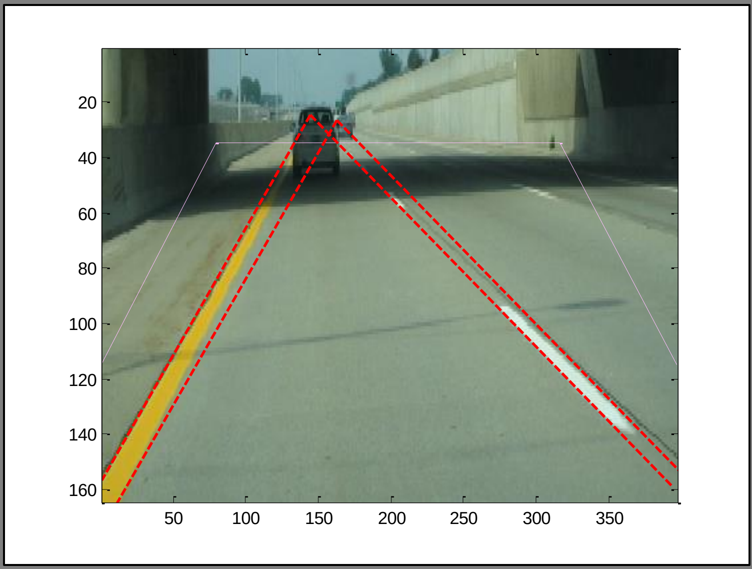
DO NOT COPY

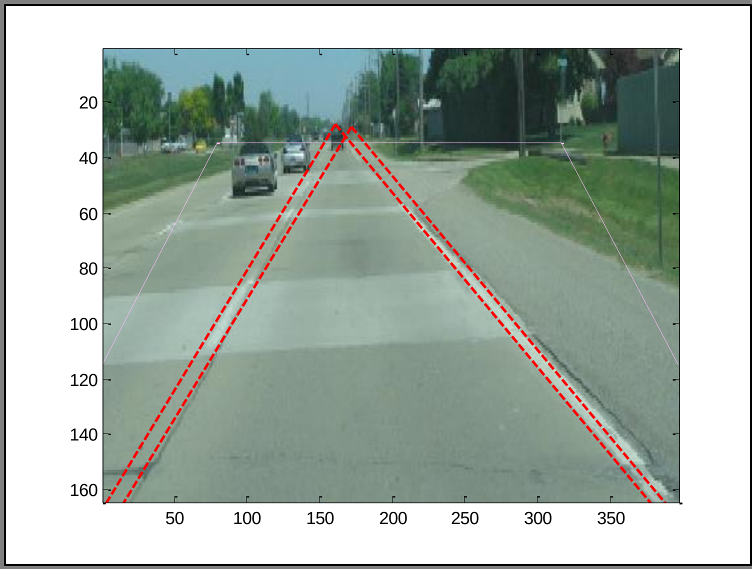


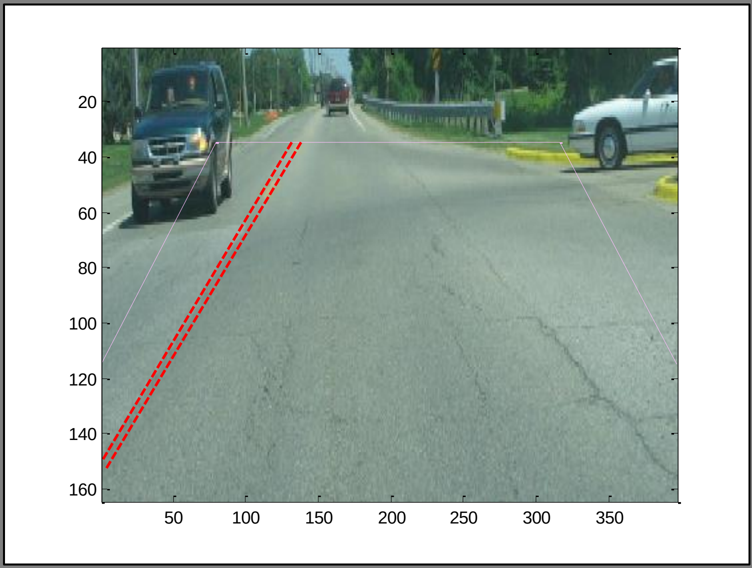


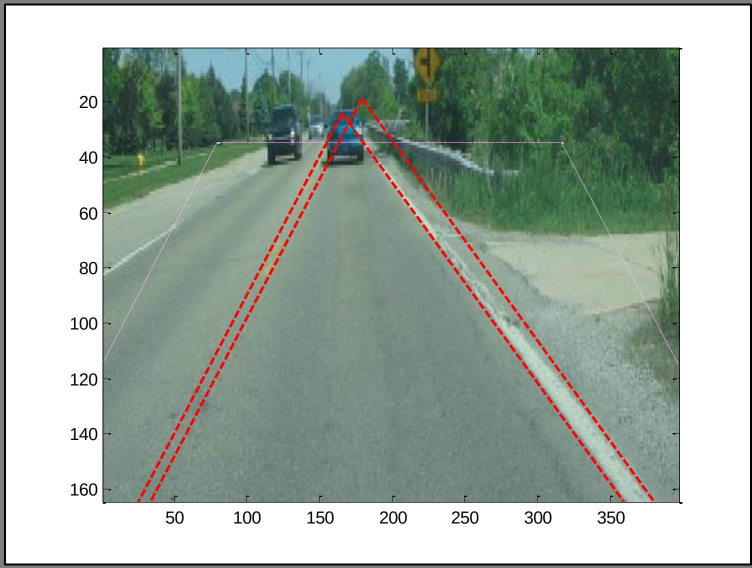




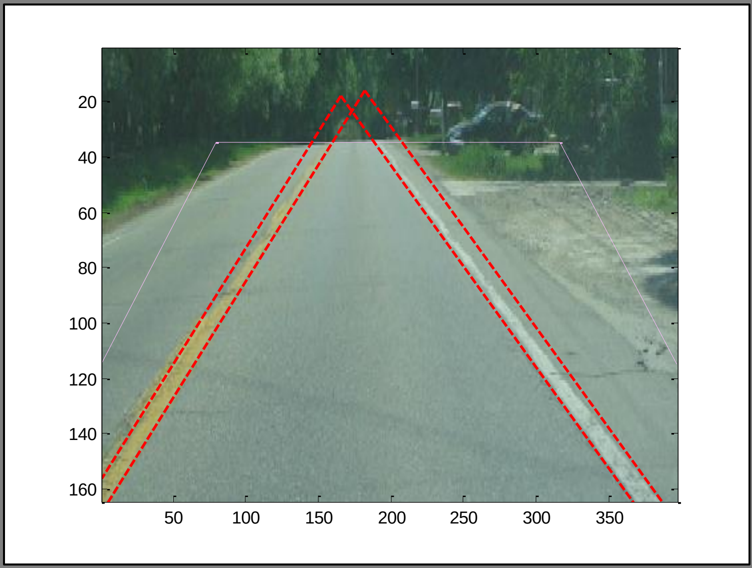


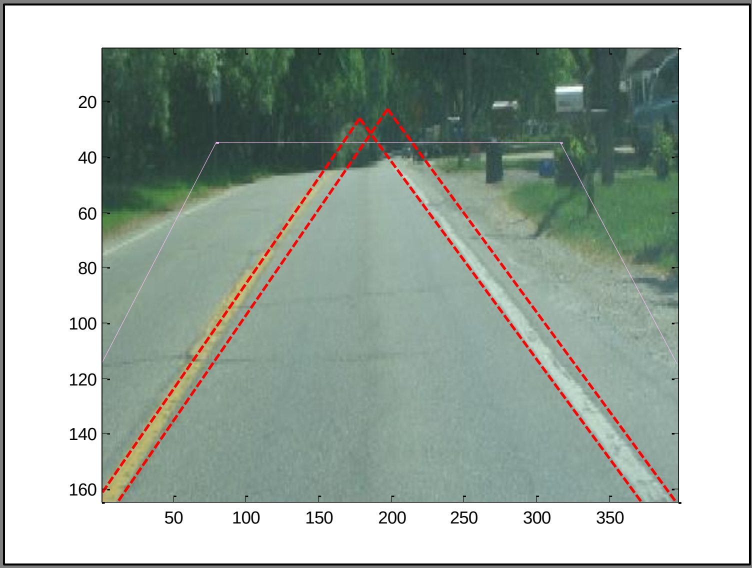


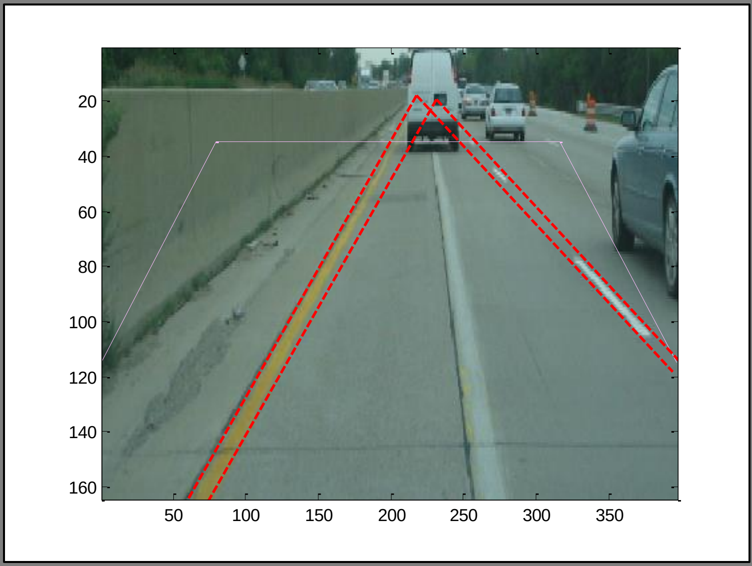


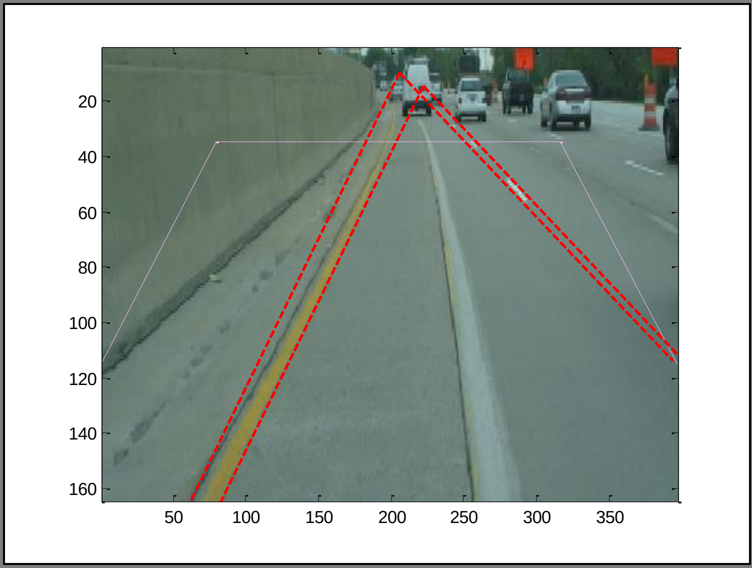


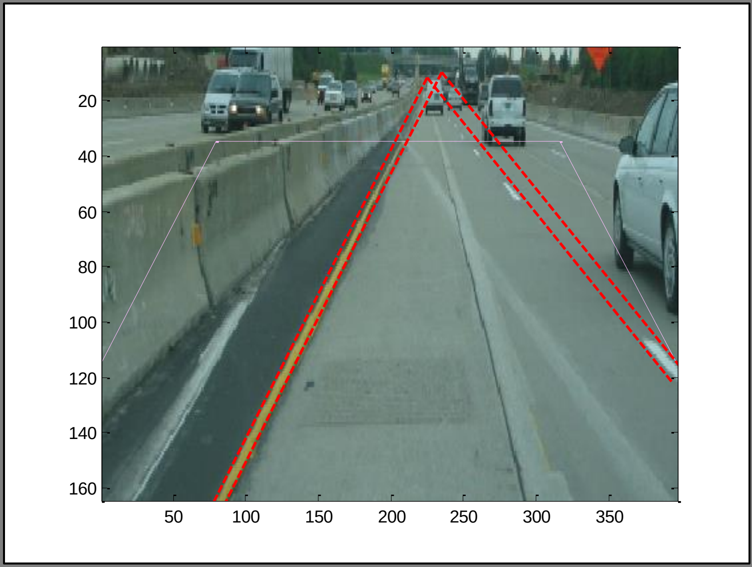
DO NOT COPY

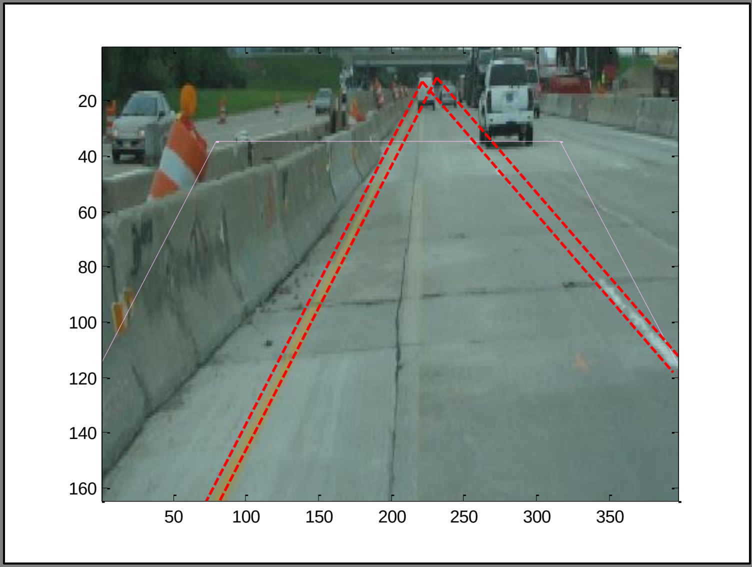


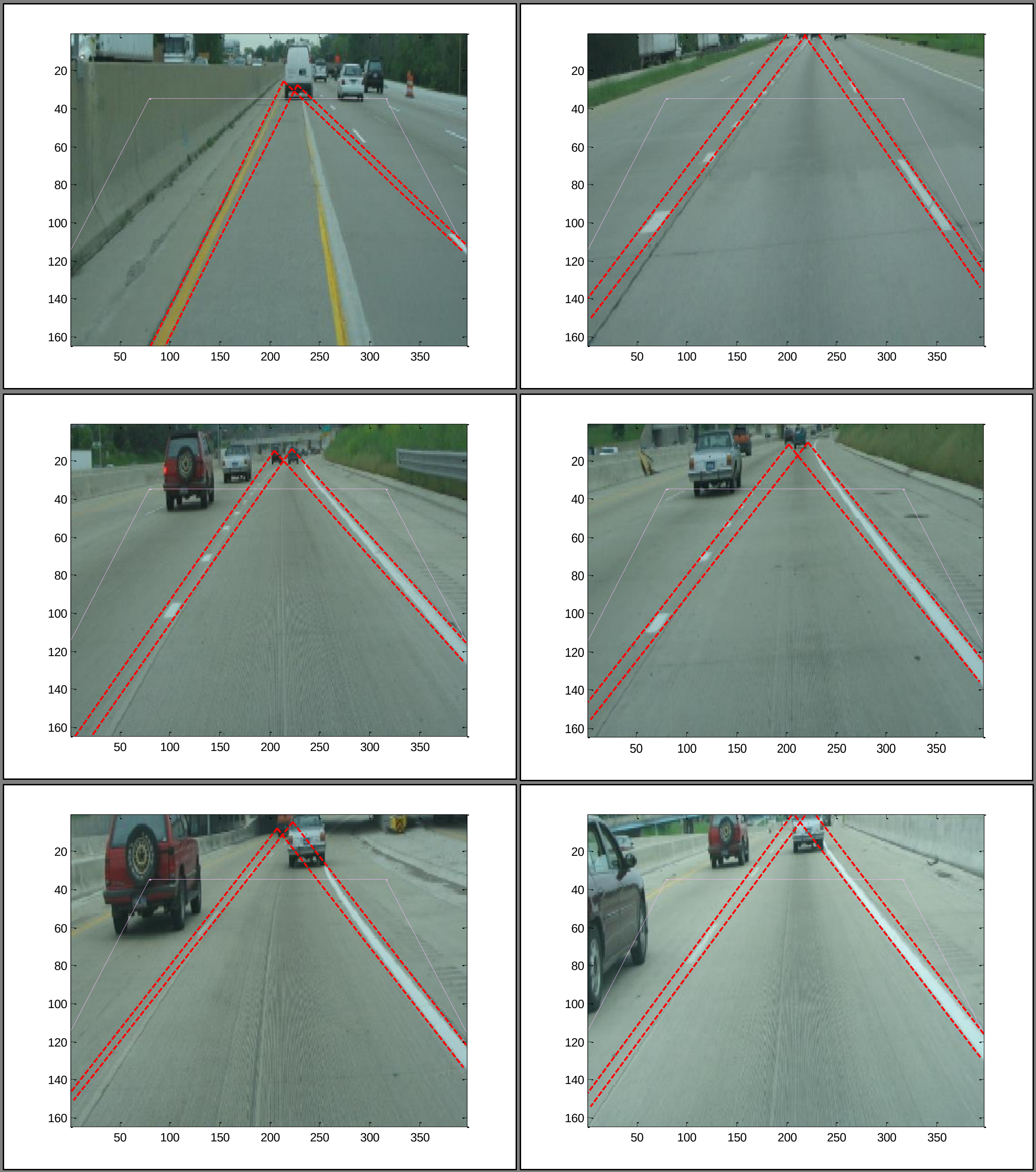


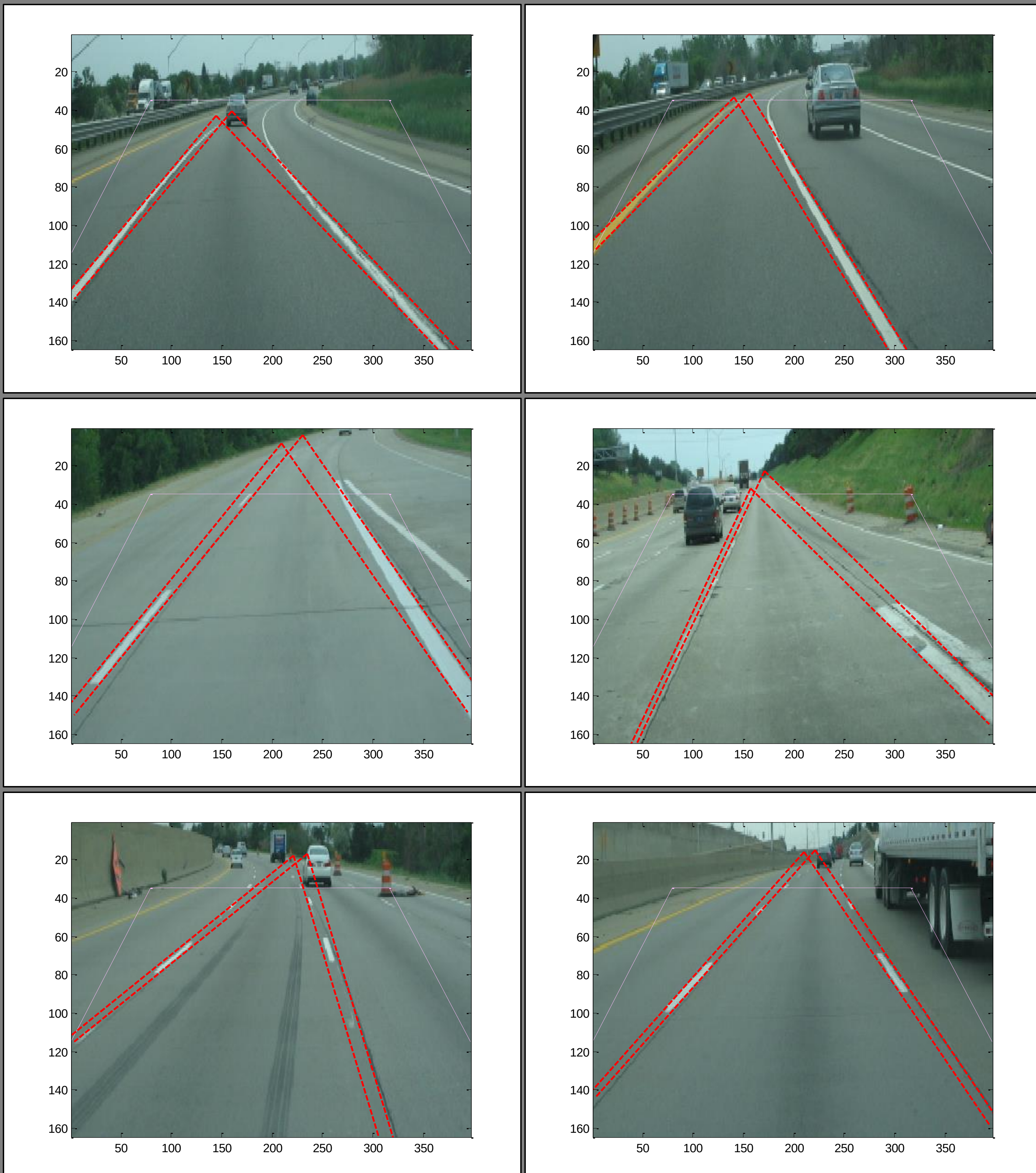


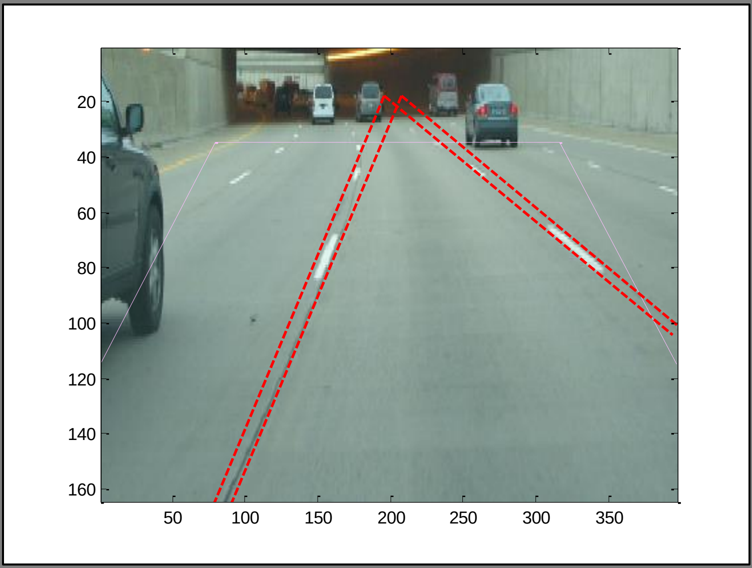


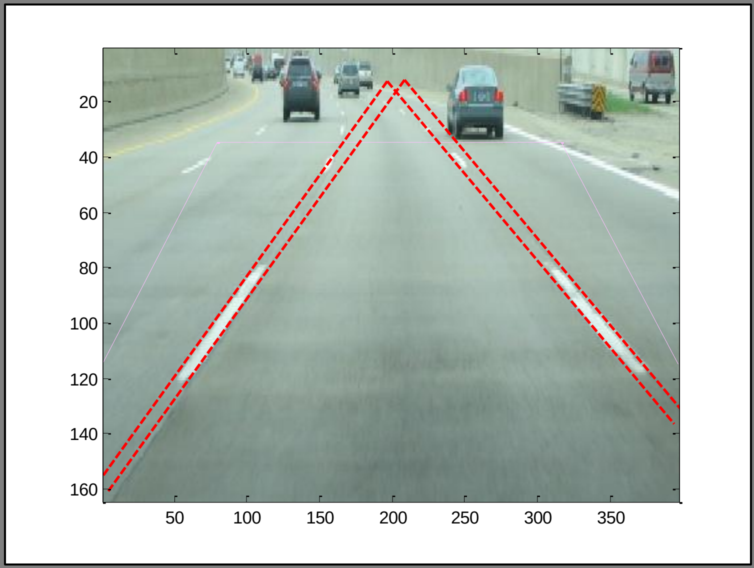


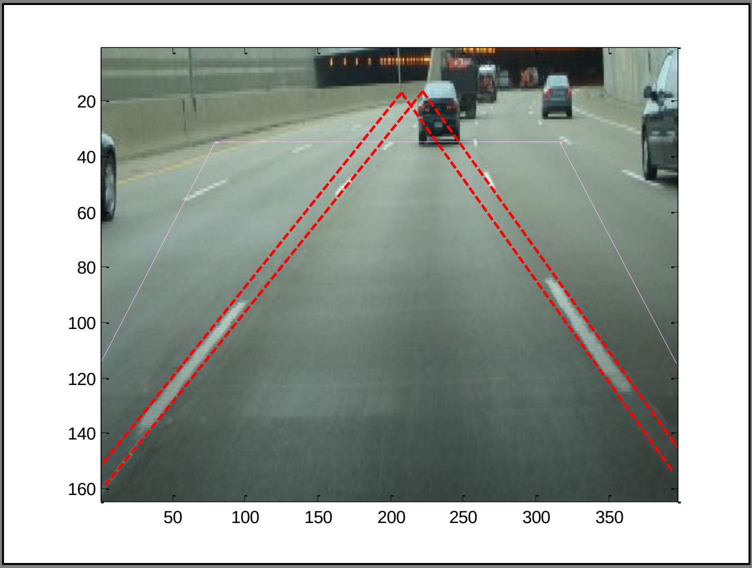


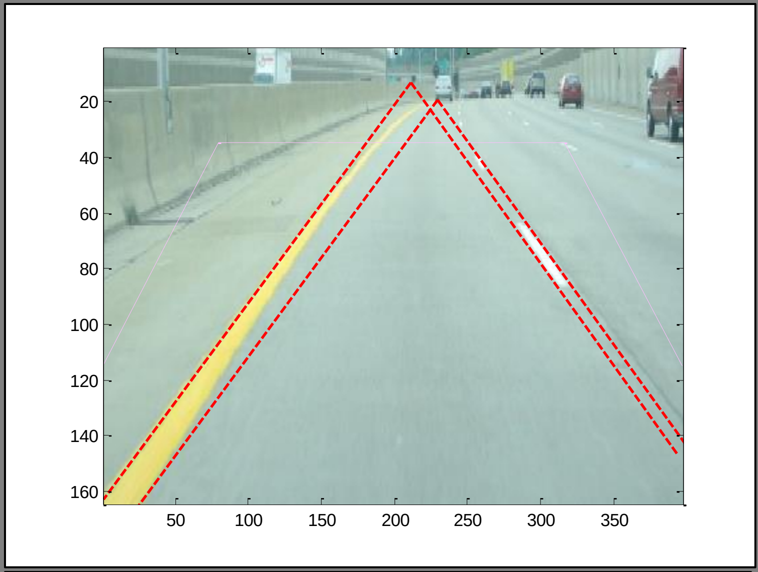


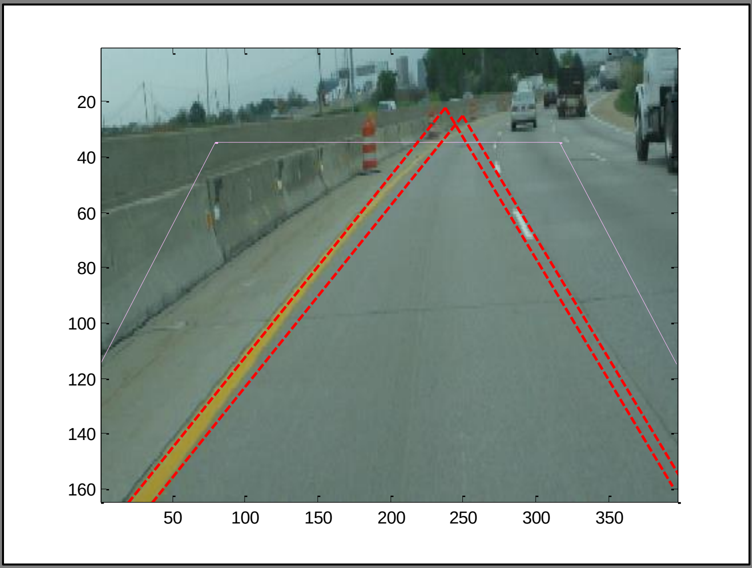


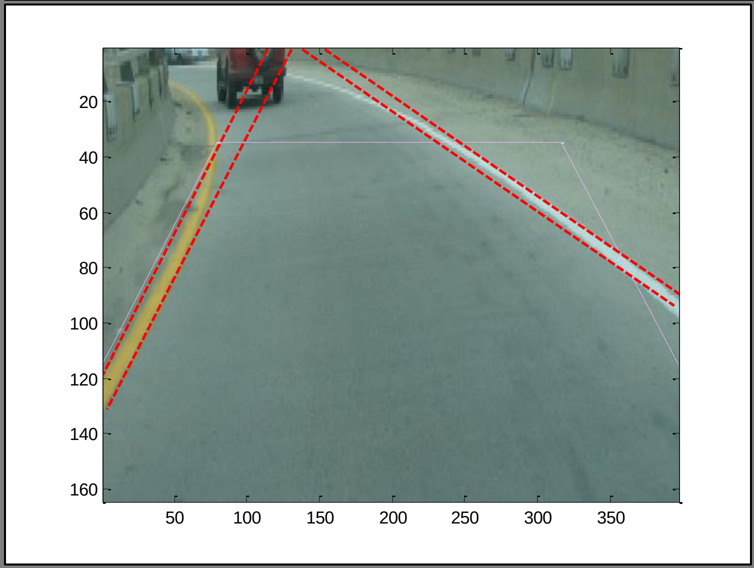




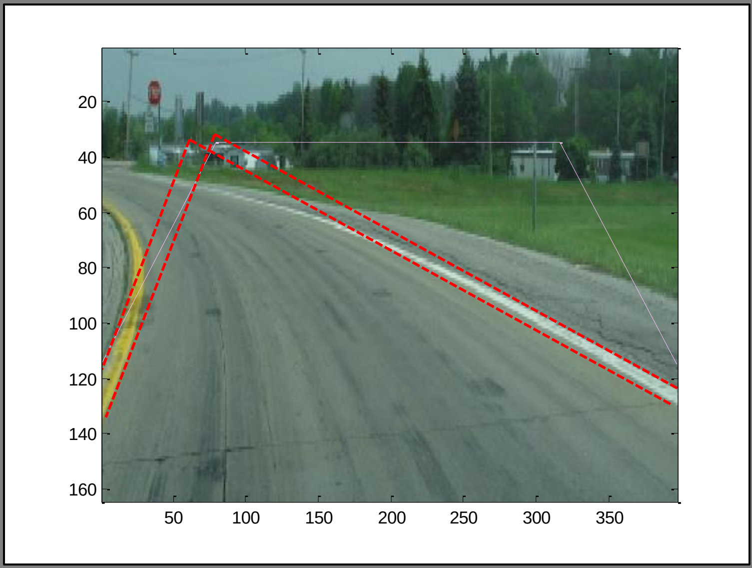


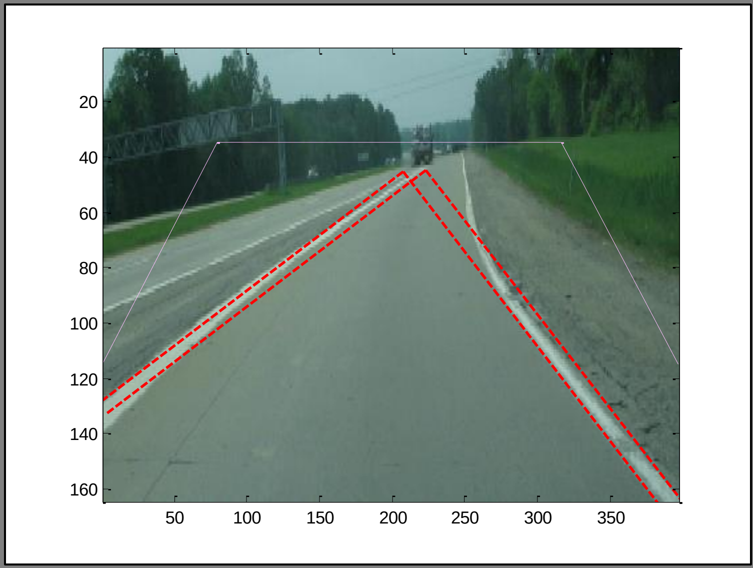


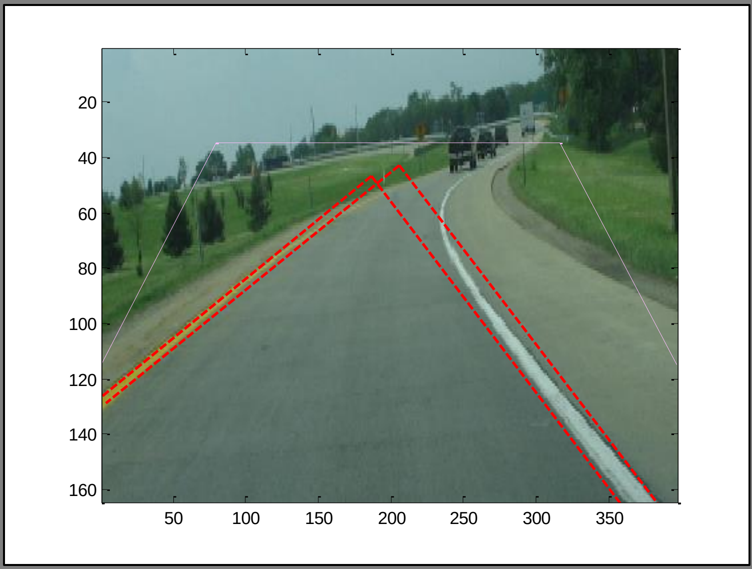


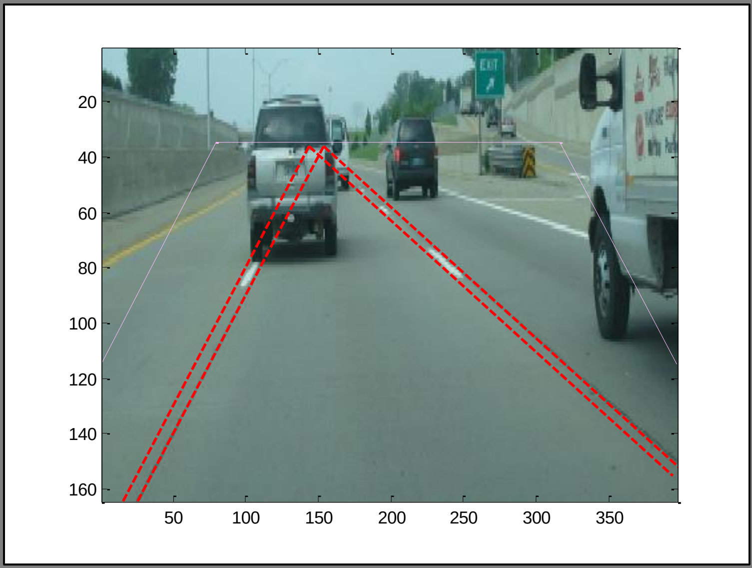


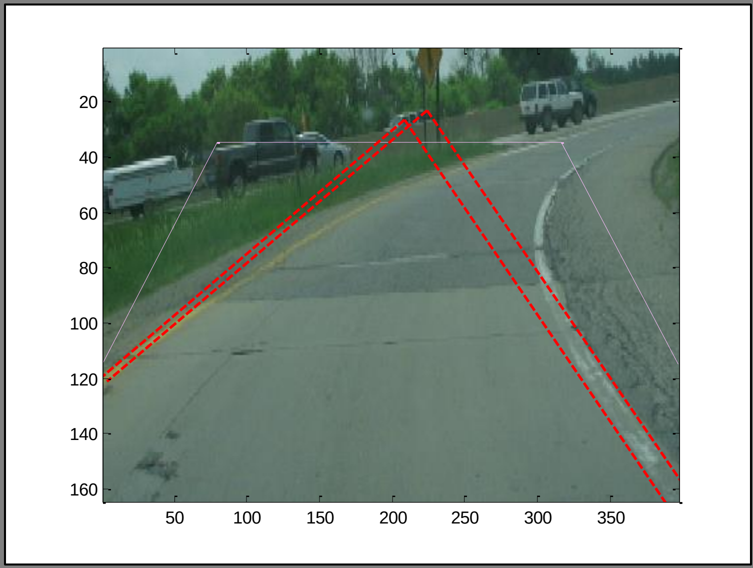
DO NOT COPY

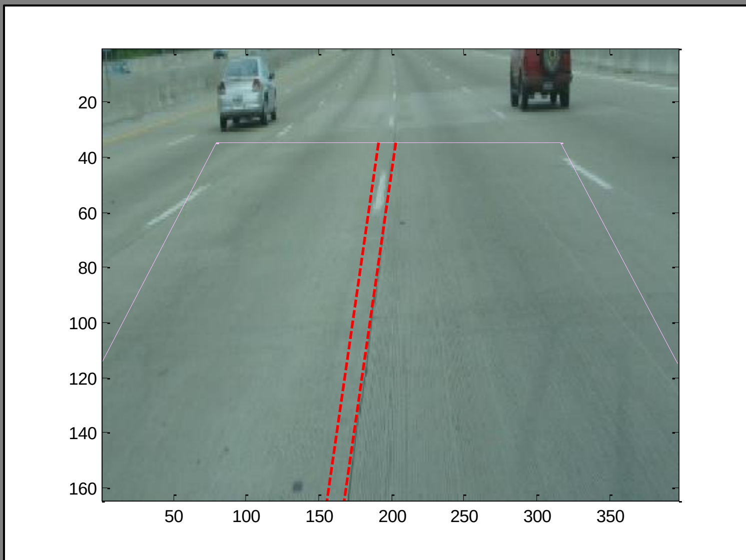


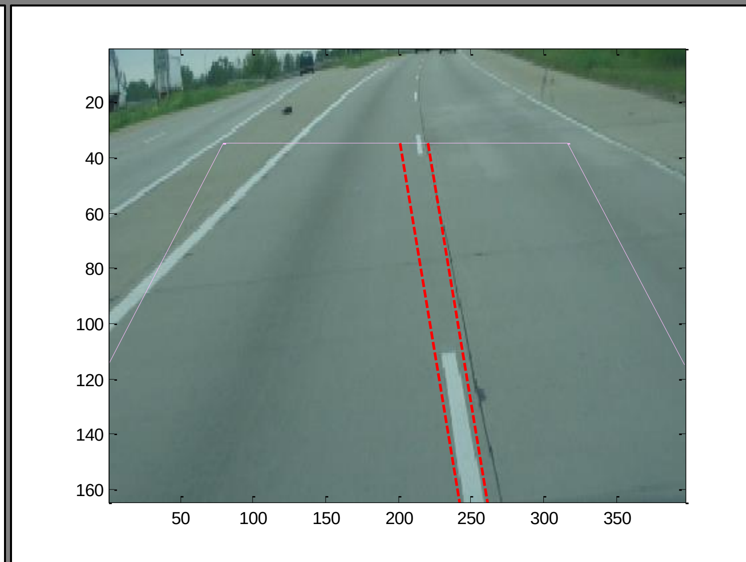


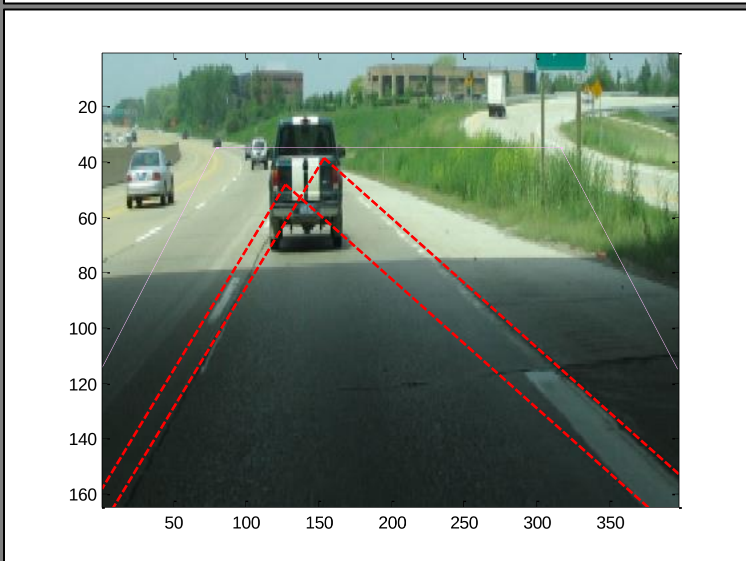


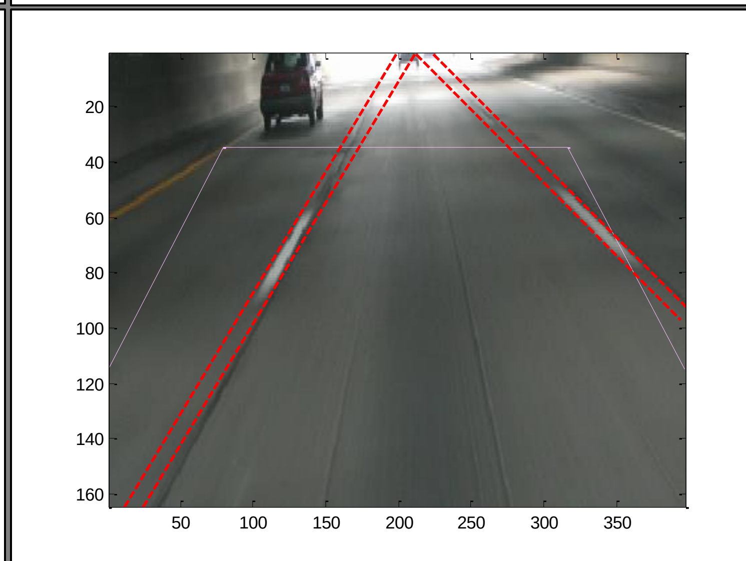


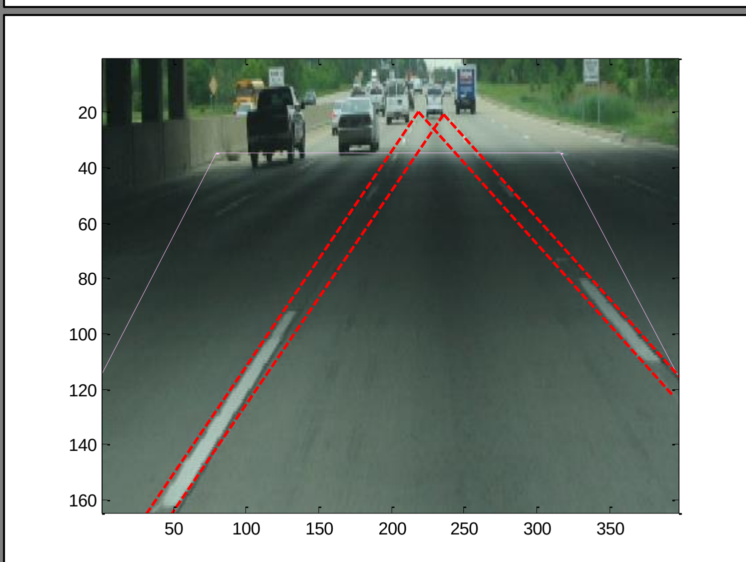


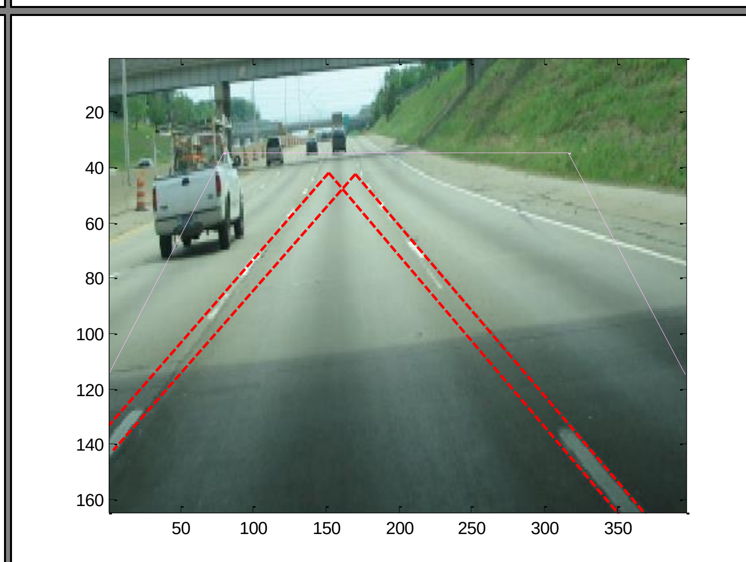




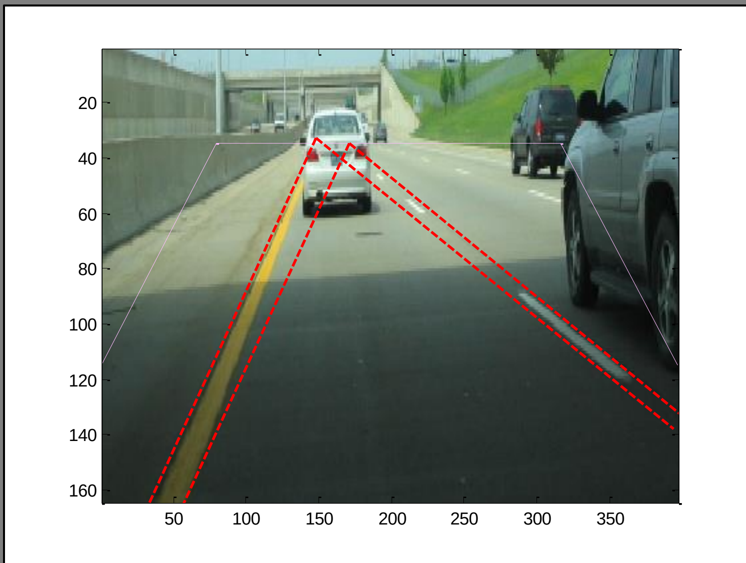


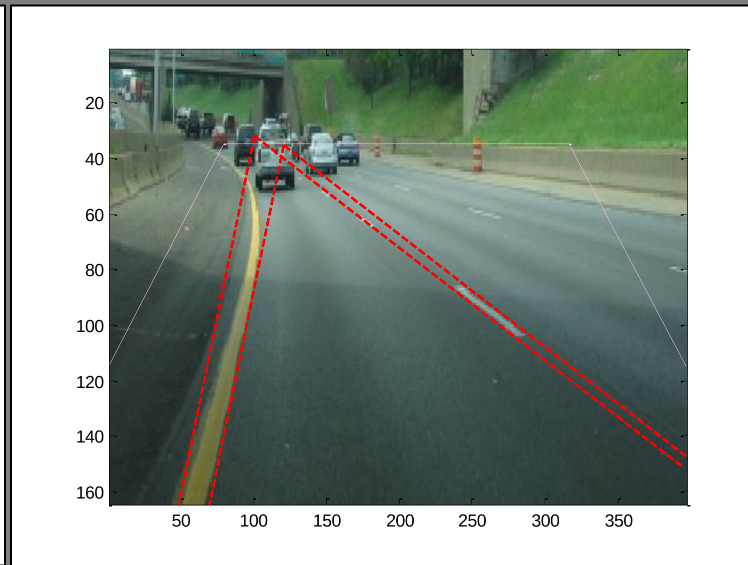


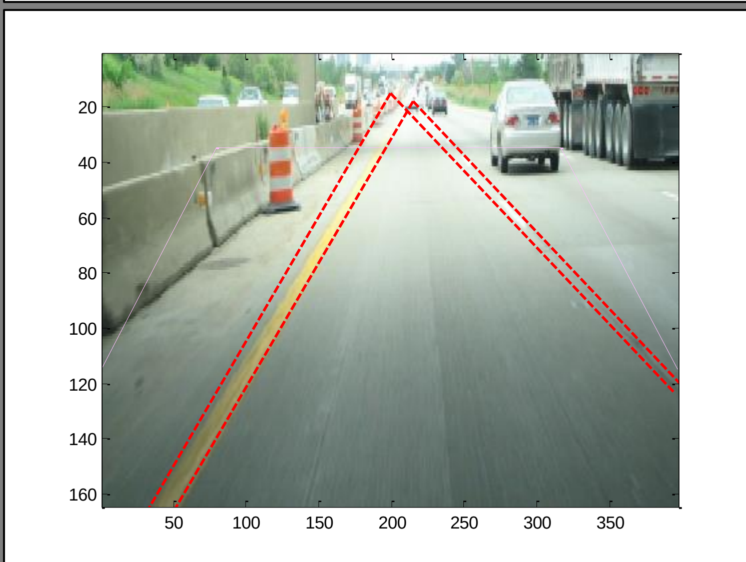


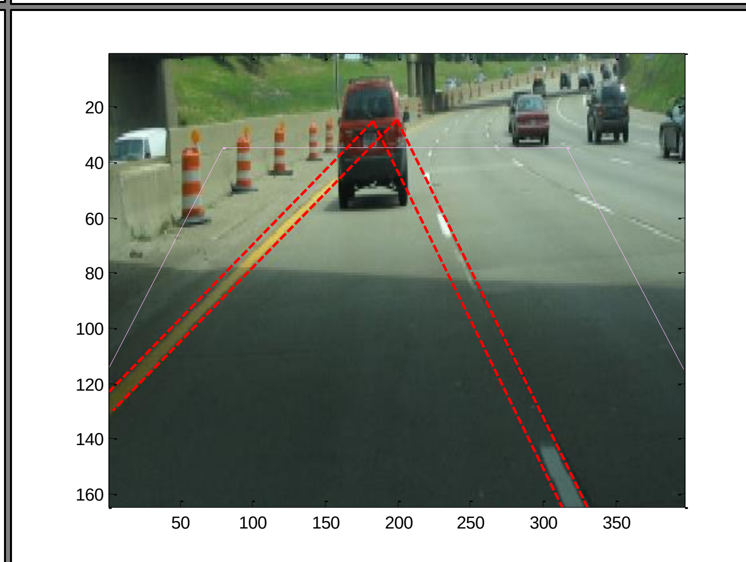


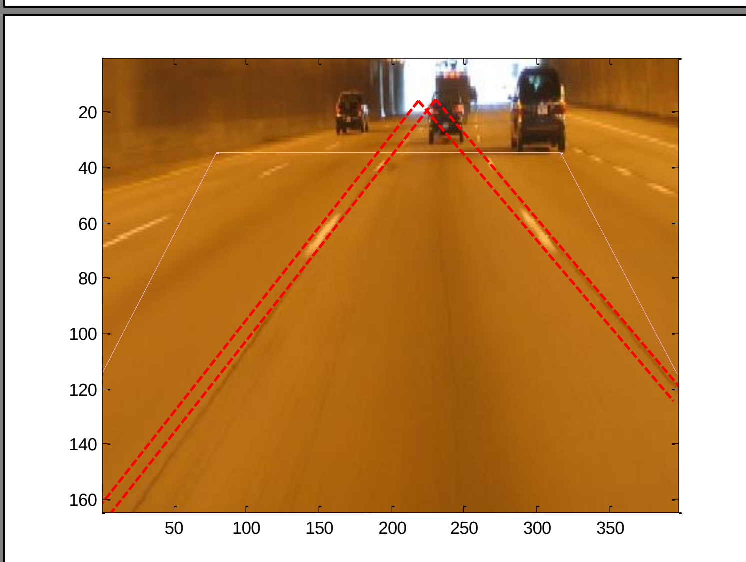
DO NOT COPY











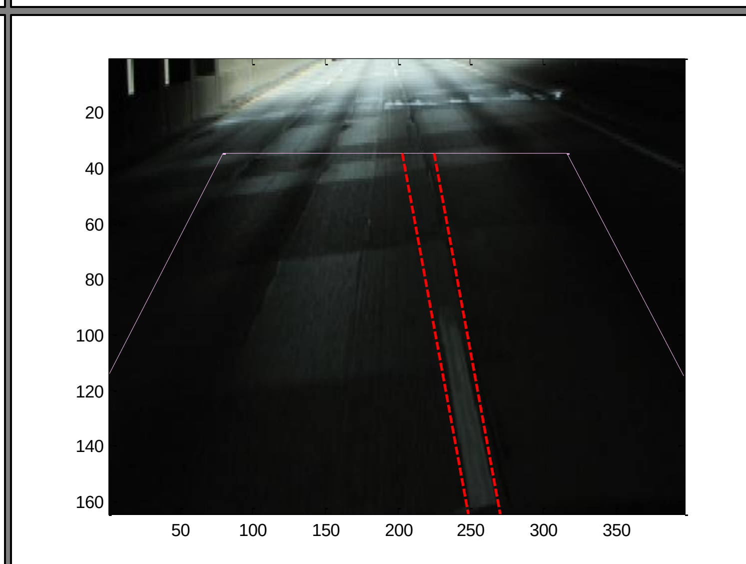


Table X. Roadways traveled during road scene image acquisition

NAMES OF SOME ROADWAYS TRAVELED FOR ROAD SCENE IMAGE COLLECTION	
1.	**Interstate 75**
2.	**Interstate 696**
3.	**Michigan 59**
4.	**Interstate 96**
5.	**Interstate 275**
6.	**Michigan 53**
7.	**Michigan 10**
8.	**Interstate 94**
9.	**Michigan 39**
10.	**Various local steets and roads**

Table X. Vehicles used during road scene image acquisition

NAMES OF SOME VEHICLES USED FOR ROAD SCENE IMAGE COLLECTION	
1.	**1999 Toyota Solara 2 door**
2.	**2005 Toyota Camry 4 door**

Table X. Partial specifications on Canon PowerShot A85 camera

	PowerShot A85	Description
1.	Camera Effective Pixels	Approximately 4.0 million
2.	Image Sensor	1/2.7-inch CCD (Total number of pixels: Approximately 4.2 million)
3.	Lens	5.4 mm; f/2.8
4.	AF System	TTL Autofocus
5.	Shooting Distance from the front of the lens	Normal: 46 cm - infinity
6.	Shutter	Mechanical Shutter + electronic shutter
7.	Shutter Speeds	15-1/2000 sec.
8.	Light Metering System	Evaluative Metering
9.	Exposure Control System	Program AE/Shutter-priority AE/Aperture priority
10.	Exposure Compensation	+- 2 stops in 1/3-stop increments
11.	Sensitivity	Auto
12.	White Balance	Auto
13.	Built-in Flash	Auto
14.	Flash Range	Normal: 46 cm - 4.2 m
15.	Image Recording Format	Still Image: JPEG
16.	Compression	Superfine, Fine, Normal
17.	Number of Recording Pixels	Small: 640 x 480
18.	Dimensions	101.0 x 64.0 x 31.5 mm (excluding protrusions)
19.	Weight	Approximately 200 g (camera body only)