VECTOR-BASED GROUND SURFACE AND OBJECT REPRESENTATION USING CAMERAS

By

JAESANG LEE

A DISSERTATION PRESENTED TO THE GRADUATE SCHOOL
OF THE UNIVERSITY OF FLORIDA IN PARTIAL FULFILLMENT
OF THE REQUIREMENTS FOR THE DEGREE OF
DOCTOR OF PHILOSOPHY

UNIVERSITY OF FLORIDA

2009

To my dear parents, whose continuous love, prayer, and support made my journey possible
and
To my lovely wife and daughter, Youngah Kim and Jeanne

ACKNOWLEDGMENTS

I thank my advisor, Dr. Carl Crane, for his kind support and advice throughout my whole graduate education. I also thank my committee, Dr. Warren Dixon, Dr. John K. Schueller, Dr. Antonio Arroyo, and Dr. Douglas Dankel, for their support and guidance.

I also thank Center for Intelligent Machines and Robotics (CIMAR) manager, Dave Armstrong, and friend and leader, Shannon Ridgeway, for their valuable discussions. This work was made possible by the Air Force Research Laboratory (AFRL) at Tyndall Air Force Base, Florida. Thank you to all the AFRL staff.

Finally, I thank all my colleagues and friends at CIMAR. The Defense Advanced Research Projects Agency (DARPA) Grand Challenge 2005 team members, Tomas Galluzzo, Danny Kent, Bob Touchton, and Sanjay Solanki, with helped and guided my first steps of the research journey. And thanks to the DARPA Urban Challenge 2007 team members, Antoin Baker, Greg Garcia, Nicholas Johnson, Jean-Francois Kamath, Eric Thorn, Steve Velat, and Jihyun Yoon, who all worked well and had an exciting time together, and Ryan Chilton for helping with both testing and discussion.

TABLE OF CONTENTS

page

LIST OF TABLES

LIST OF FIGURES

LIST OF ABBREVIATIONS

DARPA	Defense Advanced Research Projects Agency
DGC 2005	DARPA Grand Challenge 2005
DUC 2007	DARPA Urban Challenge 2007
GPS	Global Positioning System
IMU	Inertial Measurement Unit
INS	Inertial Navigation System
JAUS	Joint Architecture for Unmanned Systems
LADAR	Light Detection and Ranging
LFSS	Lane Finder Smart Sensor component software
LFSSWing	Lane Finder Smart Sensor Wing component software
LFSS Arbiter	Lane Finder Arbiter Smart Sensor component software
NFM	North Finding Module
PFSS	(Vision-based) Path Finder Smart Sensor component software
RN	Roadway Navigation component software
TSS	(LADAR-based) Terrain Smart Sensor component software
WMVKS	World Model Vector Knowledge Store Smart Sensor component software

Abstract of Dissertation Presented to the Graduate School
of the University of Florida in Partial Fulfillment of the
Requirements for the Degree of Doctor of Philosophy

VECTOR-BASED GROUND SURFACE AND OBJECT REPRESENTATION USING CAMERAS

By

Jaesang Lee

December 2009

Chair: Carl D. Crane III
Major: Mechanical Engineering

Computer vision plays an important role in many fields these days. From robotics to bio-medical equipment to the car industry to the semi-conductor industry, many applications have been developed for solving problems using visual information. One computer vision application in robotics is a camera-based sensor mounted on a mobile robot vehicle. Since the late 1960s, this system has been utilized in various fields, such as automated warehouses, unmanned ground vehicles, space robots, and driver assistance systems. Each system has a different mission, like terrain analysis and evaluation, visual odometers, lane departure warning systems, and identification of such moving object as other cars and pedestrians. Thus, various features and methods have been applied and tested to solve different computer vision tasks.

A main goal of this vision sensor for an autonomous ground vehicle is to provide such continuous and precise perception information as traversable paths, future trajectory estimations, and lateral position error corrections with small data size. To accomplish these objectives, multi-camera-based Path Finder and Lane Finder Smart Sensors were developed and utilized on an autonomous vehicle at the University of Florida's Center for Intelligent Machines and Robotics

(CIMAR). These systems create traversable area information for both an unstructured road environment and an urban environment in real time.

Extracted traversable information is provided to the robot's intelligent system and control system in vector data form through the Joint Architecture for Unmanned Systems (JAUS) protocol. Moreover, a small data size is used to represent the real world and its properties. Since vector data are small enough for storing, retrieving, and communication, traversability data and its properties are stored at the World Model Vector Knowledge Store for future reference.

CHAPTER 1
INTRODUCTION

Motivation

Computer vision plays an important role in many fields these days. From robotics to bio-medical equipment to the car industry to the semi-conductor industry, many applications have been developed for problems using visual information. Since the price of camera sensors is falling and they are now less expensive and gather more information than light detection and ranging (LADAR) sensors, their use for applicable problem solving seems assured.

One computer vision application in robotics is a camera-based sensor mounted on a mobile robot vehicle. Since the late 1960s, this system has been utilized in various fields, such as automated warehouses, unmanned ground vehicles, space robots, and driver assistance systems [Gage 1995, McCall 2006, Matthies 2007]. Each system has a different mission. For example, systems provide terrain analysis and evaluation, visual odometers, lane departure warning systems, and moving object like other cars and pedestrians identification.

Thus, various features and methods have been applied and tested for solving different computer vision tasks. A main goal of the vision sensor for autonomous ground vehicle sensor development is to provide continuous and precise perception information, such as traversable path, future trajectory estimation, lateral position error correction, and moving and static object classification or identification. The prior work associated with the tasks of terrain analysis and the identification of roadway lanes is discussed in the literature review sections.

Literature Review

This chapter describes the various methods and algorithms that were developed to accomplish terrain analysis, lane extraction, and tracking. A terrain analysis covers feature

selection, classifier selection, and sensor fusion. A lane tracking covers feature selection, lane extraction, lane model assumption, and tracking method.

Terrain Analysis

Feature selection

Terrain analysis and estimation is an essential and basic goal of autonomous vehicle development. For off-road situations, a lack of environmental structure, plus hazardous situations, and difficulty of prediction inhibit an accurate and constant path evaluation. The key step of terrain estimation starts with proper feature selection, which is applied to the classifier.

In addition to Light Detection and Ranging (LADAR) distance information, the visual information is a good source for analyzing traversable terrain. Consequently, image intensity, various color models, texture information, and edges have been suggested and utilized for the main input for vision-based terrain estimation.

One primary feature, an intensity image, is used for terrain estimation [Sukthankar 1993, Pomerleau 1995]. The intensity image is easy to understand and requires only a small processing time, but lacks information. Many edge-based path finding systems [Behringer 2004] and stereo vision systems have used gray images to find disparities between two images [Bertozzi 1998, Kelly 1998].

The next and most commonly used feature is the red-blue-green (RGB) color space. The RGB color space is the standard representation in computer and digital cameras; therefore, it is widely known and easily analyzed. The Carnegie-Mellon Navlab vehicle used a color video camera and the RGB color space as a feature for following a road [Thorpe 1987]. Road and non-road RGB color models are generated and applied to color classification algorithms.

Another color vision system is the Supervised Classification Applied to Road Following (SCARF) system that detects unstructured roads and intersections for intelligent mobile robots [Crisman 1993]. This system can navigate a road that has no lanes or edge lines, or has degraded road edges and road surface scars.

The most common and challenging camera problem is that data are affected by changes in illumination from one time to another, which results in an inconstant color source. This is the biggest challenge to an outdoor vision-based robot system. Normalized RGB is one method used to overcome lighting effects in the color-based face recognition system [Vezhnevets 2003] and in many other fields [Bräunl 2008]. Normalization of the RGB color space makes the system less sensitive to certain color channels. In terms of hardware, attaching a polarizing filter on the camera has been considered.

Other research in vehicle sensor development introduces different color spaces, like hue and saturation, as additional features within the RGB color space [Bergquist 1999]. The RGB color space is also used as a primary feature for object detection and identification, as well as terrain estimation.

Another approach to road segmentation is a texture-based system. Zhang [1994] utilized road texture orientation using a gray image as one road segmentation feature, as well as image coordinate information. Chandler [2003] applied texture information to an autonomous lawn mowing machine. The discrete cosine transform (DCT) and discrete wavelet transformation were applied to distinguishing tall grass areas and mowed grass area.

Classifier

An autonomous vehicle is a real-time, outdoor application, and its color camera input size is relatively large. For these reasons, a simple and strong algorithm is demanded when selecting a classifier.

The Bayesian algorithm used with the RGB color space is applied as a classification algorithm in the Navlab vehicle [Thorpe 1988, Crisman 1993]. Road and non-road Gaussian models were generated using color pixels and applied to whole image pixels for road classification.

Davis [1995] implemented two different algorithms: the Fisher Linear Discriminant (FLD) applied to two-dimensional RG color feature space and the Backpropagation Neural Network was applied to a three-dimensional RGB color feature space.

Monocular vision, stereo vision, LADAR, or sensor fusion

Different types of perception systems have been developed using different type sensors, such as a monocular vision camera, stereo vision camera, Light Detection and Ranging (LADAR) sensor and camera-LADAR fusion sensor [Rasmussen 2002]. The monocular vision system is found in the Carnegie-Mellon Navlab and SCARF system [Thorpe 1988, Davis 1995]. A camera is mounted on the front in the middle of the car and faces the ground. The source image is resized for computation efficiency. Sukthankar [1993] used a steerable camera to improve the camera field of view in sharp turn situations.

Unlike the monocular vision system, a stereo vision system can detect not only terrain area, but also obstacle distance. The Real-time Autonomous Navigator with a Geometric Engine (RANGER) uses stereo vision for determining the traversable area of the terrain [Kelly 1997].

The Generic Obstacle and Lane Detection (GOLD) stereo vision system detects obstacles and estimates obstacle distance at a rate of 10Hz [Bertozzi 1998].

Lane Tracking

Feature selection

Urban areas have artificial structures for example, road lane markings, traffic signals, and information signals. The first step for a vision-based road-following system in an urban area is road line marking extraction. To meet this goal, many systems use different features. For instance, the edge extraction filter method [Broggi 1995a], morphological filtering [Beucher 1994, Yu 1992], template matching [Broggi 1995b], or frequency-based methods using a gray image or certain single channel image are utilized.

For the edge extraction method, different spatial edge filters are applied to extract lane markings. The Yet Another Road Follower (YARF) system and POSTECH research vehicle (PRV) II used a Sobel spatial filter [Schneiderman 1994, Kluge 1995, Yim 2003] and the CIMAR NaviGator III vision system used a Canny filter for extracting edge information [Apostoloff 2003, Wang 2004, Velat 2007]. The lane-finding in another domain (LANA) system applied a frequency domain feature to a lane extraction algorithm [Kreucher 1999]. The Video-based Lane Estimation and Tracking (VioLet) system used a steerable filter [McCall 2006].

Lane extraction

After an edge-based filtered image is generated, two steps are required to extract the lane. One is grouping lane pixels among the edge data, which contains many noise pixels, and the other is computing lane geometry from the grouped lane pixels.

The Hough transform is applied to overcome imperfect line segment detection caused by noise, a natural property of a road. The Hough line transform is a nonlinear transform from the

image pixel (X, Y) into a parameter space (rho, zeta). It searches for the local maximum to find the most dominant line segment. Yu [1997], Taylor [1999], Lee [2002], and Velat [2007] all applied the Hough transform for dominant line detection of an image. Chapter 4 further explains the Hough transform.

The RANdom SAmple Consensus (RANSAC) algorithm is another good tool for lane extraction in a challenging situation [Davies 2004]. The RANSAC is an iterative method to estimate mathematical model parameters from given data set that contains many outliers. It is a very robust algorithm with many outlier pixels for most hypothesized line models, although it needs many iterative steps to reach the hypothesized lane model. Kim [2006, 2008] uses the RANSAC as a real-time lane marking classifier.

Lane model

A lane model is created using an assumption based on the nature of the structured road. This lane model assumes that the road lane is a linear or parabolic curve on a flat plane and that road lane width does not change dramatically. The linear lane model is satisfied in most cases for automated vehicle control systems and lane departure warning systems in both highway and rural environments [McCall 2005]. The trajectory estimator for autonomous ground vehicles needs a parabolic curve model for accurate results [Schneiderman 1994, Kluge 1995, and Yu 1997].

Many spline curve models are utilized to represent a curved lane, for example the Cubic-spline line, B-spline, and Catmull-rom spline. Originally, the spline method was developed by the computer graphics field for efficiently representing curves. Thus, each spline model has its own character and advantage. For instance, different initial assumptions, control point locations, number of control points, and knots are suggested. The spline is also a good tool for representing

a curved lane geometry model. Wang [1998, 2000] used the Catmull-rom spline, then the B-spline [2004], and [Kim 2006, Kim 2008, Wu2009] uses the Cubic-spline for lane structures.

Tracking method (estimator)

A structured road environment does not always provide human-made line information. When a vehicle passes an intersection area, or a blurred road line area, or when other vehicles obstruct part of the line segment, the camera cannot see the road line segment. Therefore a lane tracking system is required to overcome this limitation.

The YARK system uses a least median squares (LMS) filter for estimating road model parameters [Kluge 1995]. A Kalman filter is applied for estimating the curvature of the highway [Taylor 1999, Suttorp 2006]. The Kalman filter estimates the state of a linear system from a noisy source.

A Particle filter is also applied for tracking the road lane. The Particle filter is also known as the Condensation algorithm and it is an abbreviation of **Con**ditional **Dens**ity Propag**ation** [Isard 1998]. The Particle filter is a model estimation technique using probability distribution over the state space with given information. The advantage of the Particle filter is we can apply it to a non-linear model unlike the Kalman filter. Apostoloff [2003] and Southall [2001] used the Particle filter for estimating their lane model.

CHAPTER 2
RESEARCH GOAL

Problem Statement

There is a need for a vision sensor that will enable autonomous ground vehicles to provide continuous and precise information, such as traversable paths, future trajectory estimation, and lateral position error corrections by the GPS drift, combined with small data size. The purpose of this research is to construct a vision sensor that meets these needs, yet requires minimal data.

Following are the given items or assumptions relevant to reaching that goal. First, a vehicle moves manually or autonomously. Second, The position and orientation of the vehicle is measured with respect to a global coordinate system by a Global Positioning System (GPS) and Inertial Navigation System (INS). Global position information is used to convert local information into global information. Third, an autonomous vehicle mode is provided by behavior specialist software [Touchton 2006]. Therefore the vision sensor software will work differently based on the current vehicle behavior as for example traveling a road, passing an intersection, negotiating an N-point turn, and the like. Fourth, three cameras are used to capture different fields of the view source images. The source image's quality is reasonably clear to see the environment. Last, the test area is an outdoor environment that can include an urban environment and an unstructured environment, for example, a desert road.

Development

Two computers are used for different ranges and different fields of view. One computer executes a lane finder system and the other executes a path finder system. A long-range camera, which is mounted at the center of the sensor bridge, shares its source image with the land finder

and path finder. A system based on two short-range cameras is designed for the high resolution lane tracking system.

A lane finder computer vision system can detect a road lane in an urban environment using a short-range field of view camera and a long-range field of view camera. It also identifies lane properties, for example, lane width and lane color to better understand the surroundings. Different field of view cameras essentially calculate and perform the same task. Since the confidence and resolution of each camera result are different, each generates a confidence value.

A path finder computer vision system can detect the traversable area both in an unstructured road environment and a structured environment, such as an urban road. The system uses an RGB color as a feature and builds probabilistic models for road and non-road areas. A segmented image is converted to a global coordinate view for assisting the robot's path planning. The path finder component software can create both vector and raster output.

These vision systems are applied to a real robotic vehicle at update rates of at least 10 Hz. They use a Joint Architecture for Unmanned Systems (JAUS) compatible software, so these vision components can communicate with any JAUS compatible system or subsystem.

Further Assumptions

- The road is relatively flat.
- A camera-captured source image is reasonably clear; therefore a human also can see the road lane from the source image. To meet this assumption, auto-exposure control functionality is used to obtain clear source images for various illumination conditions.

CHAPTER 3
PATH FINDER SMART SENSOR

Introduction

The Path Finder Smart Sensor (PFSS) is a perception element and consists of a single color camera at the front of a vehicle that is oriented facing the terrain ahead. Its purpose is to classify the area in the camera's range for terrain that is similar to that on which the vehicle is currently traveling and then translate that scene information into a global coordinate system. It uses a probabilistic model of a training road area by using color pixel data. Also, it computes properties of the traversable area, for example road type like, asphalt, grass, or unpaved road. The PFSS works for both unstructured roads and structured roads.

Figure 3-1 (A) shows team CIMAR's DARPA Grand Challenge 2005 vehicle, called NaviGator II. The NaviGator II was designed for the desert autonomous vehicle racing competition, the DGC 2005. The Path Finder camera of the NaviGator II is shown in Figure 3-1 (B) and its computer housing is shown in Figure 3-1 (C). Figure 3-2 (A) and (B) show sample source images of the unstructured road environment and figure 3-3 (A) and (B) show sample source images of structured road environments.

The PFSS output supports different types of perception elements, such as LADAR. The PFSS output is fused by intelligent elements of the robot system for outdoor environment autonomous vehicle driving.

Feature Space

When a human drives down a road, even if the road does not have any artificial information, such as lane marks or road signs, the human perception system naturally tries to find the best traversable area using its visual sense and any other sense or input information. In

addition, when a human uses visual information, previous experience is added to increase the estimation ability. Even though a computer vision system does not have a human's complicated and precise perception system, it can judge the traversable area using a limited amount of information.

Like human visual systems, most vision-based systems use three major visual features: color, shape, and texture [Apostoloff 2003, Rand 2003]. The PFSS is designed for both unstructured and structured road environments, which means shape information is not available all the time. The texture of a road also differs from non-road areas. Even if texture is a good feature in this application, texture requires high computation power and it works best on a clear focused image. For these reasons, the primary feature used for analytical processing is RGB color space or a variant version of RGB color space.

RGB Color Space

The RGB color system is the standard in the world of computers and digital cameras and is a natural choice for color representation. Furthermore, RGB is the standard output from CCD/CMOS-cameras. Therefore, it is easily applied to a computer system. Figure 3-4 shows the RGB color space cube [Gonzalez 2004]. The RGB color space-based system provides fairly successful results in most instances, but this feature is not robust enough for the real world outdoor environment.

Normalized RGB Color Space

Since the RGB color space does not have illumination-associated color elements, selecting the RGB color system has a disadvantage with respect to illumination variation, such as in outdoor environment applications. The saturation and hue in the hue, saturation, and value (HSV)

color space are relatively strong color elements, but these color elements also need additional element intensity for overall classification in the PFSS.

The Normalized RGB is a variant version of the RGB color space that shows less sensitivity to various light conditions [Vladimir 2003, Bräunl 2008]. It is insensitive to surface orientation, illumination direction, and illumination intensity. The Normalized RGB is only dependent on the sensor characteristics and surface albedo [Lukac 2006]. Eq. (3-1) and Eq. (3-2) show two different versions of the normalized RGB equation. In this research, Eq. (3-1) is utilized.

$$\begin{aligned}
\text{Normalized Red} &= \frac{Red}{\sqrt{Red^2 + Green^2 + Blue^2}}, \\
\text{Normalized Green} &= \frac{Green}{\sqrt{Red^2 + Green^2 + Blue^2}}, \\
\text{Normalized Blue} &= \frac{Blue}{\sqrt{Red^2 + Green^2 + Blue^2}},
\end{aligned} \tag{3-1}$$

or

$$\begin{aligned}
\text{Normalized Red} &= \frac{Red}{Red + Green + Blue}, \\
\text{Normalized Green} &= \frac{Green}{Red + Green + Blue}, \\
\text{Normalized Blue} &= \frac{Blue}{Red + Green + Blue}.
\end{aligned} \tag{3-2}$$

Training Area

The PFSS classification algorithm uses one or more sub-images as a training area for building a probabilistic model. These training areas are selected on the assumption that the vehicle drives on the road. In an unstructured environment, like the desert, since traversable road

width and area are relatively narrower than in an urban environment, the camera can see both road and non-road areas. Therefore, obtaining both traversable areas and non-traversable areas helps to increase the classification rate. In the structured road environment, many background areas were observed to be similar to the drivable road. For example, when a vehicle is undergoing a sharp turn at crossroads or a vehicle drives in a more than two lane road environment, like a highway, or another vehicle that has an asphalt-like body color drives by the PFSS training area, the non-road model will be similar to the road model. In such cases, the classification algorithm only relies on the drivable sub-image's probabilistic model. Figure 3-5 (A) shows a training area in a desert environment and Figure 3-5 (B) shows a training area in an urban road environment.

Classifier

Maximum Likelihood

The Maximum Likelihood (ML) algorithm, which is fundamentally a probabilistic approach to the problem of pattern classification, is selected for this application. It makes the assumption that the decision problem is posed in probabilistic terms, and that all the relevant probability values are known. While the basic idea underlying the Maximum Likelihood theory is very simple, this is the optimal decision theory under the Gaussian distribution assumption. Therefore, most pixel classification is done using the Maximum Likelihood approach.

In unstructured road environment, the decision boundary that was used is given by

$$\frac{1}{(2\pi)^{d/2}|\sum_1|^{1/2}}\exp\left[-\frac{1}{2}(\mathbf{x}-\boldsymbol{\mu}_1)^T\sum_1^{-1}(\mathbf{x}-\boldsymbol{\mu}_1)\right] = \frac{1}{(2\pi)^{d/2}|\sum_2|^{1/2}}\exp\left[-\frac{1}{2}(\mathbf{x}-\boldsymbol{\mu}_2)^T\sum_2^{-1}(\mathbf{x}-\boldsymbol{\mu}_2)\right], \quad (3\text{-}3)$$

where

$$\mu_1 = [\mu_{Red} \;\; \mu_{Green} \;\; \mu_{Blue}]^t, \tag{3-4}$$

$$\Sigma_1 = \begin{bmatrix} \sigma^2{}_{RR} & \sigma^2{}_{RG} & \sigma^2{}_{RB} \\ \sigma^2{}_{RG} & \sigma^2{}_{GG} & \sigma^2{}_{GB} \\ \sigma^2{}_{RB} & \sigma^2{}_{GB} & \sigma^2{}_{BB} \end{bmatrix}. \tag{3-5}$$

Eq. (3-4) and Eq. (3-5) are the mean vector and covariance matrix respectively of the drivable-area RGB pixels in the training data and μ_2 and Σ_2 are those of the background pixels.

The decision boundary is simplified as follows:

$$(\mathbf{x}-\boldsymbol{\mu}_1)^T \Sigma_1^{-1}(\mathbf{x}-\boldsymbol{\mu}_1) + \ln|\Sigma_1| = (\mathbf{x}-\boldsymbol{\mu}_2)^T \Sigma_2^{-1}(\mathbf{x}-\boldsymbol{\mu}_2) + \ln|\Sigma_2|. \tag{3-6}$$

In most pixel classification problems, the logarithm components in Eq. (3-6) are not a dominant factor for classification. Therefore, to save time, these two values are not computed since this application requires a real time implementation. So RGB pixels of X, which belong to a class, are computed based on the power of the Mahalanobis distance $(\mathbf{x}-\boldsymbol{\mu}_1)^T \Sigma_1^{-1}(\mathbf{x}-\boldsymbol{\mu}_1)$ [Charles 1988].

Mixture of Gaussians

The Maximum Likelihood-based classification algorithm used for the 2005 DARPA Grand Challenge was limited in many situations because its basic assumption was that training areas have only one Gaussian distribution. However in most cases, the properties of the road training area do not change rapidly and its distribution is Gaussian, background sub-images do change at every scene and it is inappropriate to assume the data distribution is Gaussian. Also, even if the road training area model has a Gaussian distribution, the overall road scene is not always a Gaussian distribution because light conditions change and the road image can be contaminated

by leaked oil and so on. For this situation, the Expectation-Maximization (EM) algorithm is implemented for road classification.

The RGB distribution of the unstructured scene shown in Figure 3-2 (A) is portrayed as a three-dimensional distribution plot in Figure 3-6 (A), (C), and (E). Figure 3-2 (A) is a test area scene at Citra, FL, and Figure 3-2 (B) is a sample scene from the 2005 DARPA Grand Challenge course in Nevada. Like most of the outdoor image systems, the NaviGator vision system is susceptible to extreme changes in lighting conditions. Such changes can create shadows in captured images that can result in changes in the image color distribution. Such shadows can be seen in Figure 3-2 (B).

From the 3-D RGB plots in Figure 3-6 (A) and (B), it is clear that most of the road training-area distribution is well clustered and can be evaluated with a single Gaussian distribution. However, for the background, there is no common distribution in the data. Figures 3-6 (C) and (D) show 3-D RGB plot of background areas. Thus, if a single Gaussian distribution is assumed for these areas, a large number of classification errors will be introduced. For this reason, the Gaussian-based classifiers possess limited performance ability in real world scenes. This argument is further evidenced by the statistical model distribution for the background case in which the distribution is poorly defined. Therefore it is clear that a more sophisticated modeling approach is needed, namely a mixture of Gaussian models. In the mixture model, a single statistical model is composed of the weighted sum of multiple Gaussian models. Consequently, a mixture modeling classifier represents more complex decision boundaries between the road training sub-image and background training sub-images. However, computing a complex mixture model requires more processing time than computing a single Gaussian model, choosing the proper number of mixture Gaussian models for a real-time application is

critical. In this project, a single Gaussian model for the road training sub-image and two mixtures of Gaussian models for the background sub-image were selected empirically.

Expectation and Maximization (EM) Algorithm[1]

The Expectation-Maximization (EM) algorithm consists of three steps. The first step is to decide on the initial value of $\Theta_i = \{\mu_i, \Sigma_i, P(\omega_i)\}$ (the mean vector, covariance matrix, and probability for ith Gaussian distribution, respectively). The second step is the expectation step, which calculates the expected value $E[y_{ij} \mid \Theta]$ for the hidden variable y_{ij}, given the current estimate of the parameter Θ. The third step calculates a new maximum-likelihood estimate for the parameters Θ^k assuming that the value taken on by each hidden variable y_{ij} is its expected value $E[y_{ij} \mid \Theta]$. The process then continues to iterate the second and third steps until the convergence condition is satisfied.

$$\Theta^k = \arg\max_{\Theta} Q(\Theta, \Theta^{k-1}), \tag{3-7}$$

where

$$Q(\Theta, \Theta^{k-1}) = E[\log p(x, y \mid \Theta) \mid x, \Theta^{k-1}]. \tag{3-8}$$

It is given that x_j is the known image pixel RGB vector and the labeling of the Gaussian distribution is given in the hidden variable y_i. By completing the data set for z_j, one can let

$$z_j = \{x_j, y_j\}, \tag{3-9}$$

where $j \in \{1, 2, ..., n\}$, and n is the number of background data pixels.

[1] This section is referred in [Lee 2006]

In the mixture modeling, it has to compute the value of the hidden variable vector y_i, where $i \in \{1, 2\}$. The value y_i can be decided by two simple binary random variables

$y_{ij} = 1$ when x_j belongs to Gaussian ω_i,

$y_{ij} = 0$ otherwise,

so that

$$y_j = \{y_{1j}, y_{2j}\}. \tag{3-10}$$

The vector y_j can only take two sets of distinct values: {1, 0}, {0, 1}.

By applying the K-mean clustering method for two Gaussian distributions' initial $\{\mu_i, \Sigma_i, P(\omega_i)\}$, the algorithm clusters the RGB pixels based on attributes into k partitions. Since it was decided to employ a two-mixture Gaussian model for the two background sub-images, the clustering uses two means to compute the covariance and each Gaussian value's probability.

The principal difficulty in estimating the maximum-likelihood parameters of a mixture model is that it is hard to know the labeling y_i of each data pixel. From the value by the k-means clustering algorithm, one can compute $E[y_{ij} \mid \Theta]$, where the value y_i is given by

$$y_i = \frac{P_i(x \mid \phi)}{\sum_{i=1}^{2} P_i(x \mid \phi_i)}, \tag{3-11}$$

where

$$p(x \mid \phi_i) = p(x \mid \mu_i, \Sigma_i) = \frac{1}{(2\pi)^{3/2} \left|\Sigma i\right|^{1/2}} \text{ex} \left[\text{p} - \frac{1}{2}(x - \mu_i)^T \Sigma_i^{-1} (x - \mu_i)\right]. \tag{3-12}$$

Next, one can compute the new $\{\mu_i, \Sigma_i, P(\omega_i)\}$ in terms of the complete data set $z_j = \{x_j, y_j\}$. Finally, a mixture-of-Gaussian model is computed from the new mean vector

$$\mu_i = \frac{\sum_{j=1}^{n} y_{ij} x_j}{\sum_{j=1}^{n} y_{ij}}. \tag{3-13}$$

Similarly, a new covariance matrix is obtained from

$$\Sigma_i = \left(\sum_{j=1}^{n} y_{\partial}(x_j - \mu_i)(x_j - \mu_i)^T \right) / \left(\sum_{j=1}^{n} y_{ij} \right). \tag{3-14}$$

Likewise, the probability of a single Gaussian distribution is obtained from

$$P(\omega_i) = \frac{n_i}{n} = \left(\sum_{j=1}^{n} y_{ij} \right) / n. \tag{3-15}$$

Finally, the solution to the mixture-of-Gaussian is found as

$$P(x \mid \Theta) = \sum_{i=1}^{2} p(x \mid \phi_i) P(\omega_i) = p(x \mid \phi_1) P(\omega_1) + p(x \mid \phi_2) P(\omega_2). \tag{3-16}$$

The EM algorithm was simulated on test images to gauge its performance in classifying images. For the purpose of the simulation, a single image was empirically chosen from the test site at Citra as a representative “easy case” and a second image from the 2005 DARPA Grand Challenge as a representative “hard case.” The Bayesian-based classification result and the EM-based classification result are shown in Figure 3-7. In considering the DARPA Grand Challenge 2005 image (shown in Figure 3-3 (B)) it is clear that there is a considerable affect of shadow/illumination on the road surface leaving the left portion of the road many shades darker than the right. Since slightly more of the road is covered in shade, the resulting sample

contained in the training segment is biased to the left half of the image, as shown in Figure 3-3 (B).

Figure 3-8 shows the EM error of the scenes at Citra and the DARPA Grand Challenge 2005 where the X-axis shows the number of Gaussian distributions for the road training region and background training region. For the DARPA Grand Challenge 2005 scene with only one Gaussian distribution, an error of 24.02% is obtained. However, if one Gaussian model for the road training region and two Gaussian models for the background training region are used, an error of 6.12% is obtained. For the Citra case, the RGB distribution of the road training region and the background training region shows that the two distributions do not overlap or intermix. As a result, a single Gaussian distribution for both the road and background training regions yields an error of 9.31%. Similarly, by applying one Gaussian for the road training region and two Gaussians for the background training regions, the error is 6.42%. From these results, it is clear that the EM classification algorithm provides better classification performance. Furthermore, it is clear that the EM algorithm can dramatically reduce the classification error over the Maximum Likelihood, in particular in cases of images obscured by shadow, adverse lighting, or vibration-induced motion blur.

Since the EM algorithm relies on an iterative approach, setting the correct iteration condition is critical to reducing processing time. In this application, the mean RGB value is used to control the iterative process wherein the algorithm will continue to iterate until the difference between the previous mean RGB and the current mean RGB of the image is less than a pre-defined threshold value N. The N is determined heuristically and a value of 0.1 was used in this research

$$\left|\mu^{k}{}_{i}\right| - \left|\mu^{k-1}{}_{i}\right| < N \quad . \tag{3-17}$$

It should be noted that the variable *k* in Eq. (3-17) represents the current step of the iteration. Figure 3-9 shows the two Gaussian distributions' absolute mean values over the iteration step for the DGC 2005 image.

Sub-Sampling Method

Pixel-Based Segmentation

After building the probability model of the training areas, each pixel in a source image is classified as drivable road or non-drivable background. Even if this is a simple procedure, applying each pixel to the classifier requires high computation power and is time consuming. Also, it can lead to results that have much less particle noise. Figure 3-10 (A) is the source image at the Gainesville Raceway and Figure 3-10 (B) shows a pixel-based classification result. Therefore, a block-based sub-sampling procedure is suggested to both increase processing speed and reduce noise pixels.

Block-Based Segmentation

A block-based segmentation method is used to reduce the segmentation processing time. Regions of N × N pixels are clustered together and each is replaced by its RGB mean value:

$$\mu^{L}_{(x,y)} = \frac{1}{N^2}\sum_{i=1}^{N}\sum_{j=1}^{N} P^{L}{}_{(i,j)} \quad , \tag{3-18}$$

where μ is the new pixel mean value for the N × N block, P is raw pixel data, (i, j) is raw pixel orientation, (x, y) is new block orientation, $L \in \{1, 2, 3\}$ for RGB, and N is block size.

These new (x, y) blocks are computed from top left to bottom right of the image and the cluster or blocks are then classified. This result has less noise and it decreases processing time

dramatically, since fewer numbers of pixel data are applied to the classifier. For example, if image size is 320 × 108, 34,560 pixels are processed to a classifier. However, if 4 × 4 blocks are computed and then applied to a classifier, only 2,160 blocks are applied to the classifier. Therefore, a 4 × 4 block-based method means 16 times less computation time is demanded.

In the current computation environment, the pixel-based segmentation processing time is 15 milliseconds. The block-based method spends less than 1 millisecond, which is 15 times faster than a pixel-based classification with a 320 × 108 source image. This processing time depends on computer CPU speed, image size, and block size, but it is clear that block-based sub-pixel classification method is faster than the pixel-based classification. One disadvantage is that edges are blurred and are not as distinct.

Figure 3-10 (C) and (D) show the 4×4 block and 9×9 block-based classification results. In the NaviGator II vision system, a 1 pixel offset corresponds to 1.1 cm in the bottom part of the image, and in the NaviGator III vision system, a 1 pixel offset corresponds to 3.73 cm at a distance 10 meter ahead of the North Finding Module (NFM).

Coordinate Transformation

After classification of the image, the areas denoted as drivable road are converted by a perspective transformation into global coordinates used for the raster-based traversability grid map. The raster-based traversability grid map tessellates the world around the vehicle into a 2-D grid. The grid is always oriented in a North-East direction with the vehicle positioned in the center of the grid [Solanki 2006]. Figure 3-11 illustrates the traversability grid map definition.

In a computer vision system, there is a similarity between a traversability grid map and a single channel image, except for the coordination system. An image uses a local image

coordination system and a traversability grid map uses a north heading and a global coordinate system, which is the same as a GPS coordinate system. Therefore, a perspective transformed image is generated by using reference points that match the same points both in the traversability grid map and the image, and then applying the current GPS position and rotating it by the inertial navigation system (INS) yaw data. Figure 3-12 (A) shows the camera view and world view, and Figure 3-12 (B) shows the relationship between a camera coordinate system, vehicle coordinate system, and world coordinate system.

Figure 3-13 (A) shows reference points in a calibration image. Figure 3-13 (B) shows reference points in a 40 × 40 meter, 1 meter resolution traversability grid map. A perspective transformation is applied to convert image domain pixels to traversability grid map pixels. The perspective transformation matrix is calculated based on camera calibration parameters [Hartley 2004]. Table 3-1 shows the location of four reference points in an image and a 60 × 60 meter, 0.25 meter resolution grid map.

Table 3-1. Reference point locations in image domain and grid map domain. Grid map size is 60 x 60 meters and resolution is 0.25 meter.

Reference Point #	Image coordinate (x,y)	Grid map
0	(20, 49)	(101 = 121-20, 161 = 121+40)
1	(84,80)	(101 = 121-20, 201 = 121+80)
2	(227,85)	(141= 121+20, 201 = 121+80)
3	(303,53)	(141 = 121+20, 161 = 121+40)

This relationship can be described as

$$X = Hx, \tag{3-19}$$

where X is a vector of traversability grid map coordinates, x is a vector of image plane coordinates, and H is a transformation matrix. In a 2-D plane, Eq (3-19) can be represented in linear form by

$$\begin{bmatrix} X_1 \\ X_2 \\ X_3 \end{bmatrix} = \begin{bmatrix} h_{11} & h_{12} & h_{13} \\ h_{21} & h_{22} & h_{23} \\ h_{31} & h_{32} & h_{33} \end{bmatrix} \begin{bmatrix} x_1 \\ x_2 \\ x_3 \end{bmatrix}. \tag{3-20}$$

Eq (3-20) can be rewritten in inhomogeneous form,

$$X = \frac{X_1}{X_3} = \frac{h_{11}x + h_{12}y + h_{13}}{h_{31}x + h_{32}y + h_{33}}, \tag{3-21}$$

and

$$Y = \frac{X_2}{X_3} = \frac{h_{21}x + h_{22}y + h_{23}}{h_{31}x + h_{32}y + h_{33}}. \tag{3-22}$$

Since there are eight independent elements in Eq (3-20), only 4 reference points are needed to solve for the H matrix.

$$\begin{bmatrix} x_1 & y_1 & 1 & 0 & 0 & 0 & -X_1x_1 & -X_1y_1 \\ 0 & 0 & 0 & x_1 & y_1 & 1 & -Y_1x_1 & -Y_1y_1 \\ x_2 & y_2 & 1 & 0 & 0 & 0 & -X_2x_2 & -X_2y_2 \\ 0 & 0 & 0 & x_2 & y_2 & 1 & -Y_2x_2 & -Y_2y_2 \\ & & & & \cdots & & & \\ x_4 & y_4 & 1 & 0 & 0 & 0 & -X_4x_4 & -X_4y_4 \\ 0 & 0 & 0 & x_4 & y_4 & 1 & -Y_4x_4 & -Y_4y_4 \end{bmatrix} \begin{bmatrix} h_{11} \\ h_{12} \\ h_{13} \\ h_{21} \\ h_{22} \\ h_{23} \\ h_{31} \\ h_{32} \end{bmatrix} = \begin{bmatrix} X_1 \\ Y_1 \\ X_2 \\ Y_2 \\ \cdots \\ X_4 \\ Y_4 \end{bmatrix} \tag{3-23}$$

Eq (3-23) is in $A = \lambda b$ form. For solving Eq (3-23) equation, a pseudo-inverse method is applied:

$$\lambda = (A^T A)^{-1} A^T B. \tag{3-24}$$

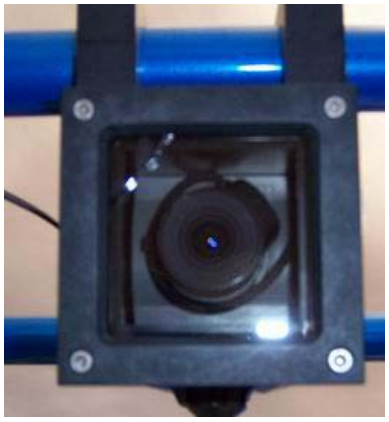
Finally, the H transformation matrix is calculated using Eq (3-24) and it is used to convert the segmented image to a traversability grid map image. Figure 3-14 (A, B) shows the classified

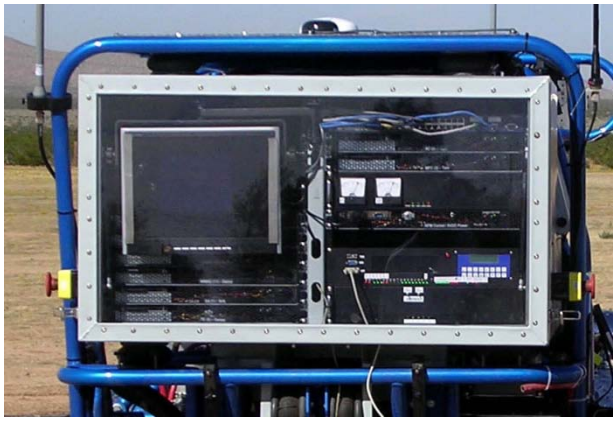
image, Figure 3-14 (C, D) shows the transformed image without pixel interpolation, and Figure 3-14 (E, F) shows the transformed image with pixel interpolation. In each subfigure (Figures 3-14 C, D, E, F), the vehicle is located at the center of the image (blue square) with its direction indicated by a thin black line.

Since a wide-angle lens is used in the camera assembly, a broad swath of the road is captured. However, as a result of that wide angle, there is an appreciable distortion in distant regions of the image. This distortion results in only a small amount of pixels representing most of the distant portion of the image. This fact results in the transformation generating a mapped image with “holes” in the distant regions of the map. These holes can then be filled by linear interpolation with respect to the row number of each pixel (see Figure 3-14 (C, E)). After creating a traversability grid map, the GPS and INS yaw data are applied to convert local coordinates into global coordinates.

A

B





C

Figure 3-1. CIMAR Navigator II Path Finder system for DARPA Grand Challenge 2005. A) camera assembly, B) computer and electronics enclosure, and C) computer housing.

A

B

Figure 3-2. Sample unstructured environment. A) Citra, FL., B) DARPA Grand Challenge 2005 course, NV.

A

B

Figure 3-3. Sample structured road environment. A) Gainesville Raceway, Gainesville, FL., B) DARPA Urban Challenge 2007 Area-C course, CA.

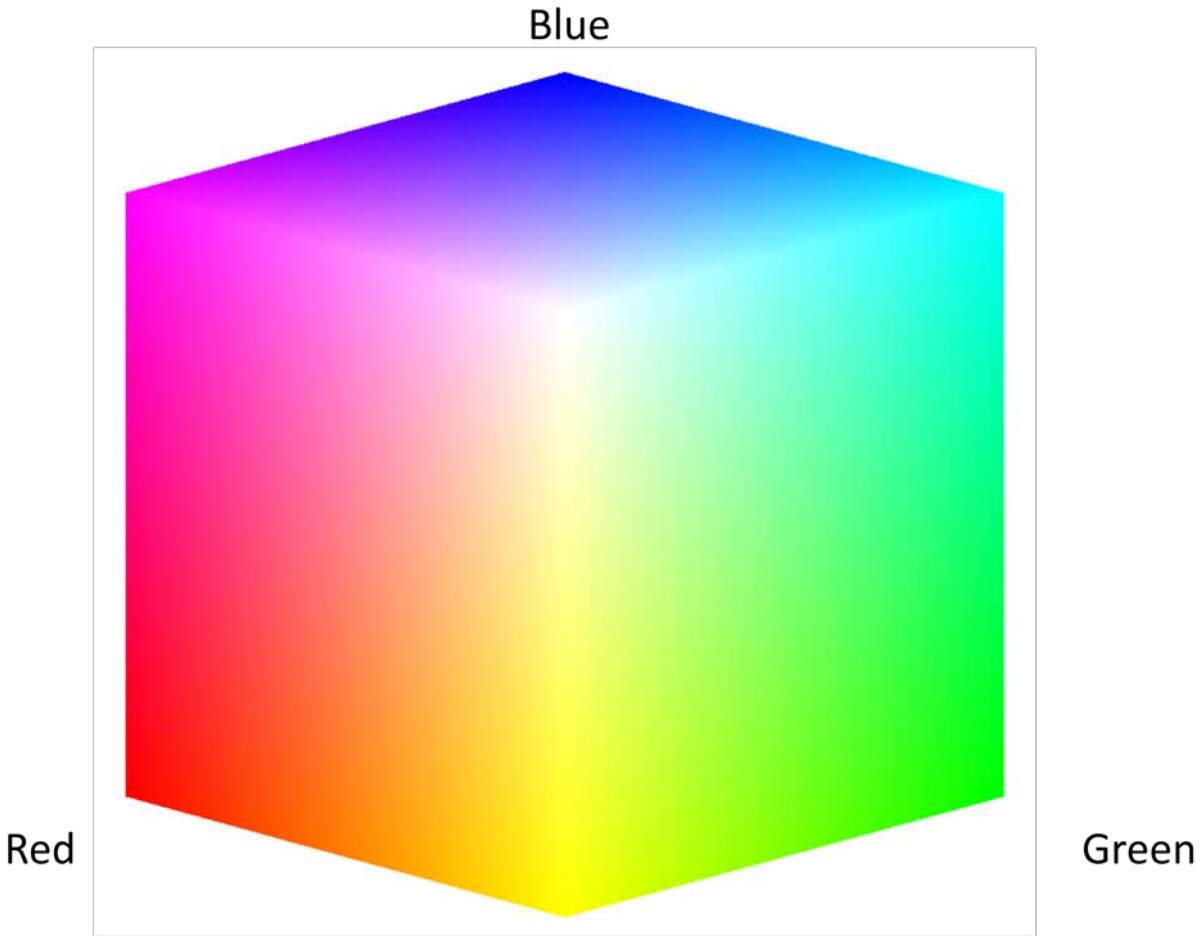


Figure 3-4. RGB (red, green, blue) color space.


A


B

Figure 3-5. Training area selection. A) Unstructured road, B) structured road.

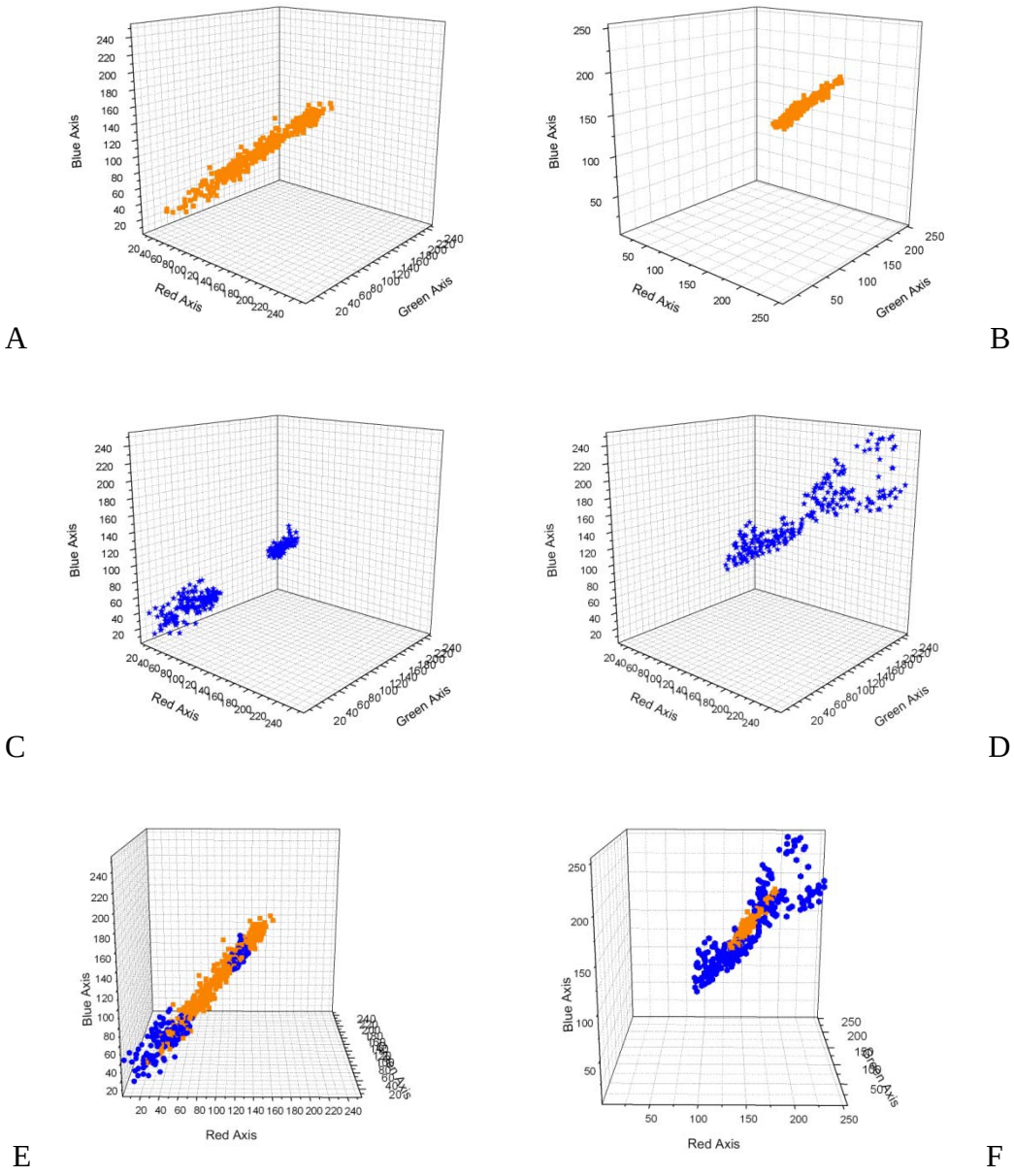


Figure 3-6. RGB distribution of road training area and background training area. A) Road at Citra, B) DARPA Grand Challenge 2005 course road, C) background at Citra, D) DARPA Grand Challenge 2005 course background, E) road and background at Citra, and F) DARPA Grand Challenge 2005 course road and background

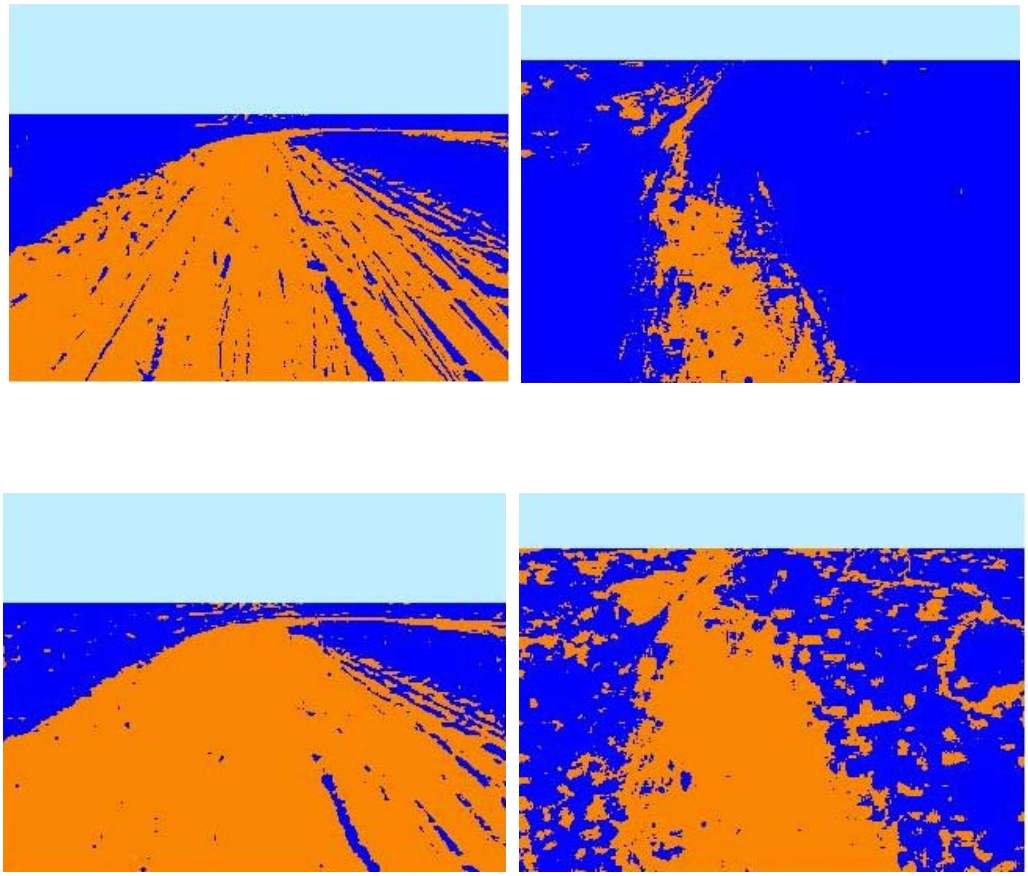


Figure 3-7. Classified road images. A) Bayesian classification result of Citra, B) Bayesian classification result of DARPA Grand Challenge 2005 course, C) EM classification result of Citra, and D) EM classification result of DARPA Grand Challenge 2005 course.

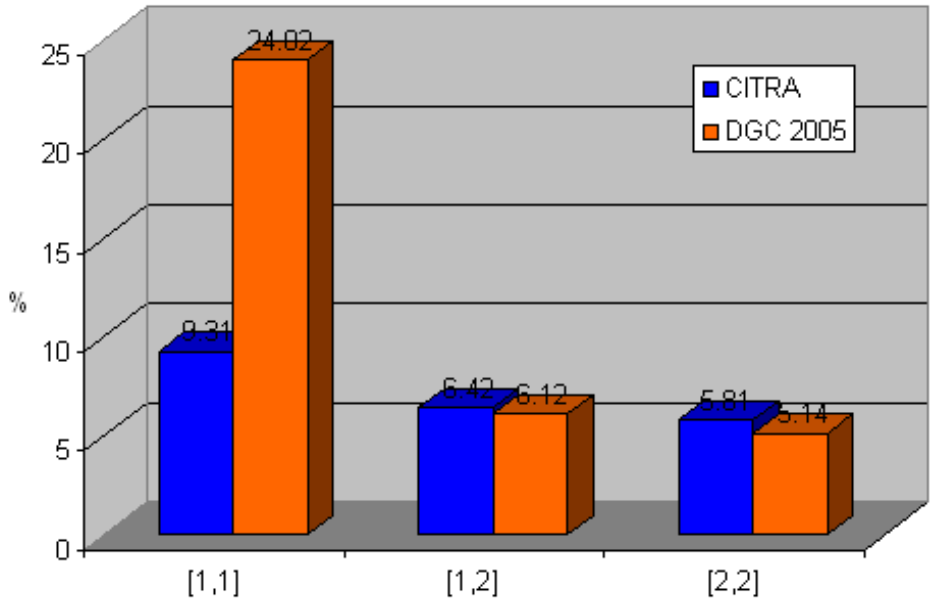
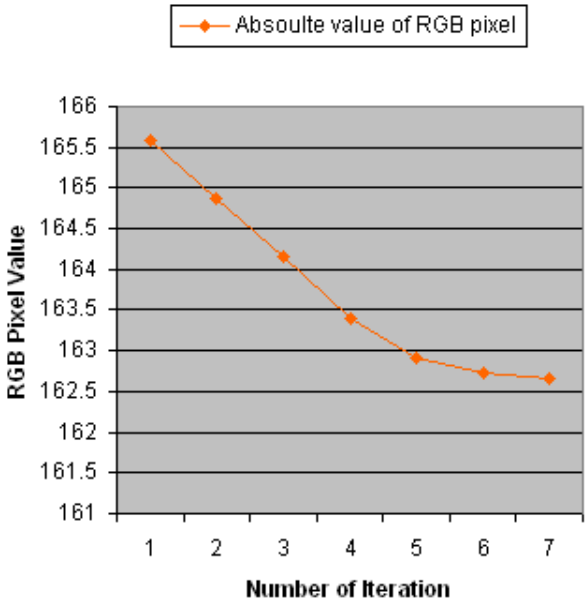
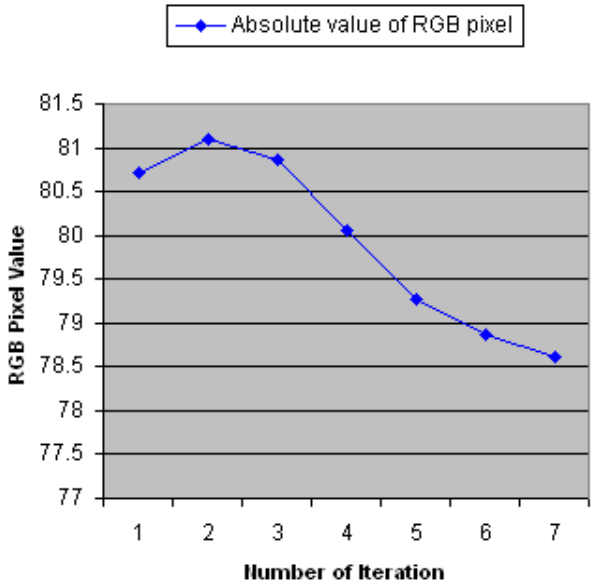


Figure 3-8. Classifier error for Citra and DARPA Grand Challenge 2005 scene with varying numbers of mixture-of-Gaussian distributions. The X axis shows the number of Gaussian distributions for the road training region and background training region.



A



B

Figure 3-9. Two Gaussian distribution' absolute mean values over the iteration step for the DARPA Grand Challenge 2005 image. A) First Gaussian mean, B) Second Gaussian mean.

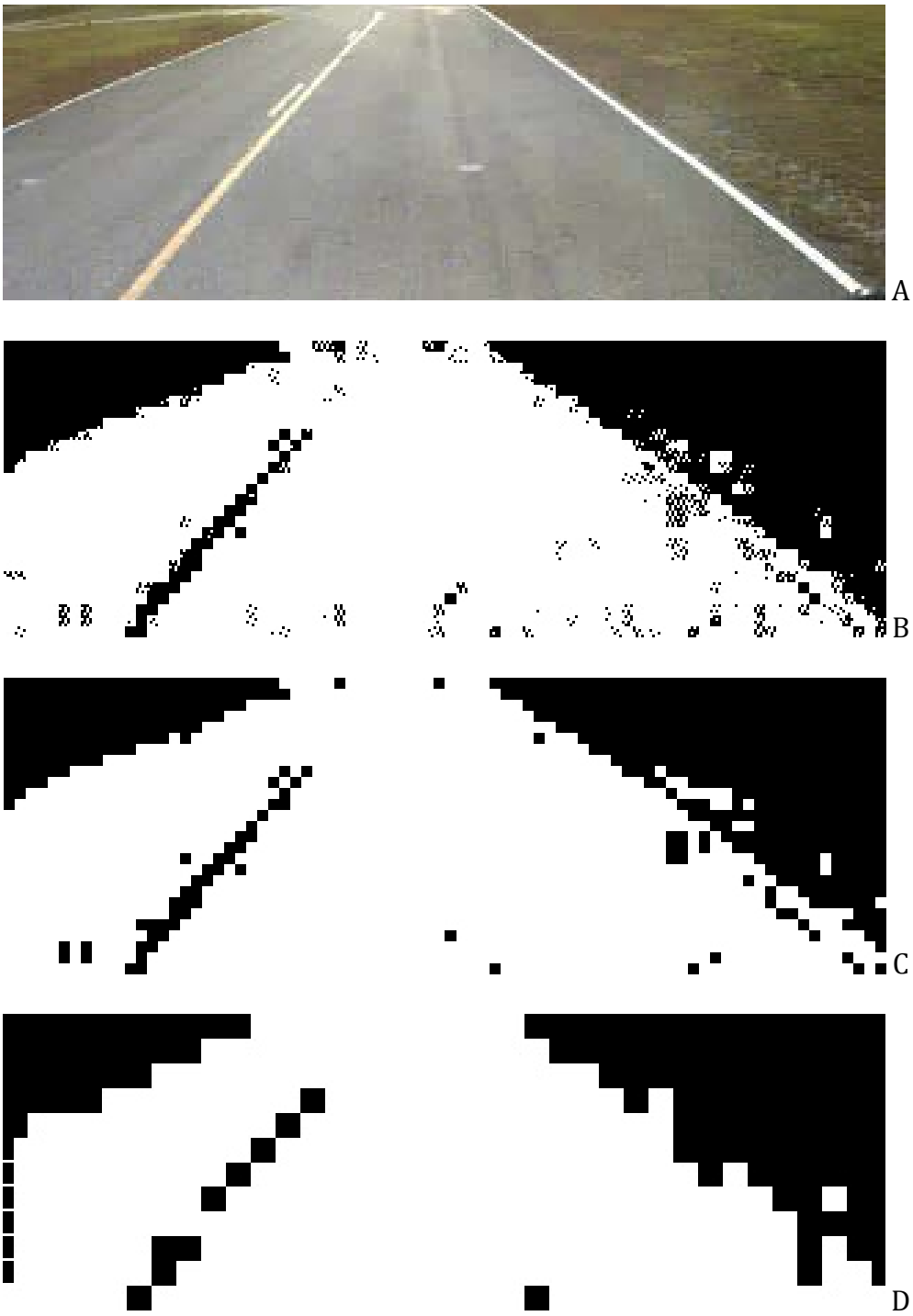


Figure 3-10. Classification result. A) Source image, B) pixel-based classification result, C) 4x4 block-based classification result, and D) 9x9 block-based classification result.

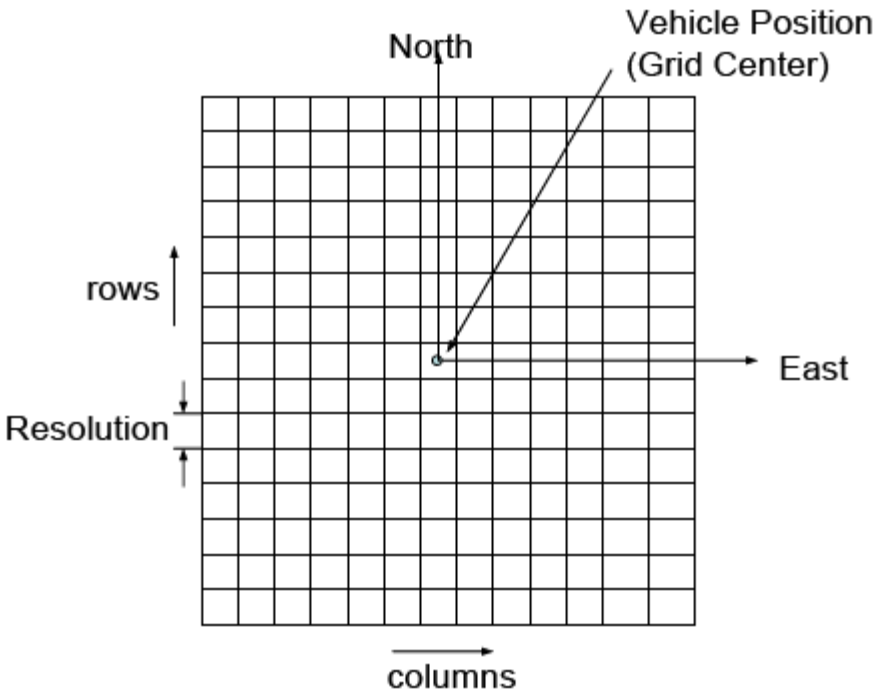
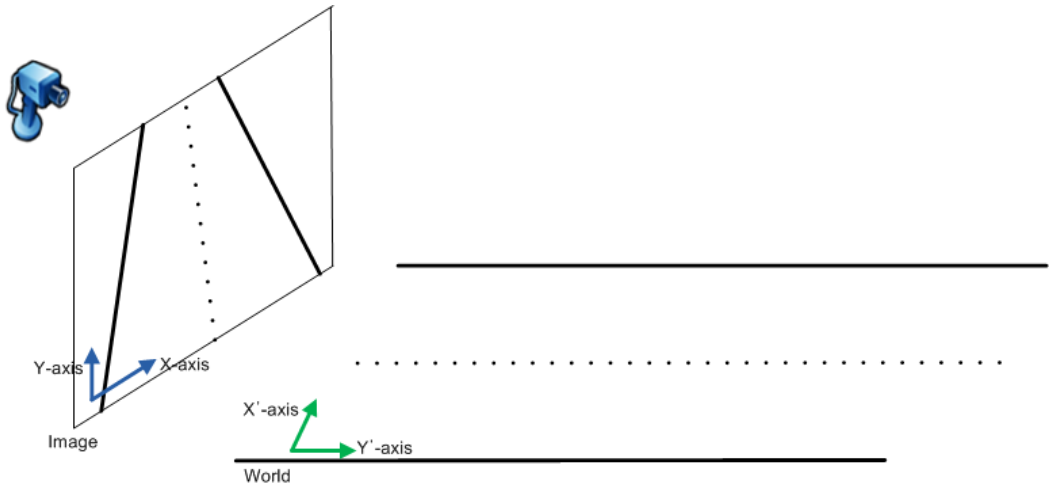
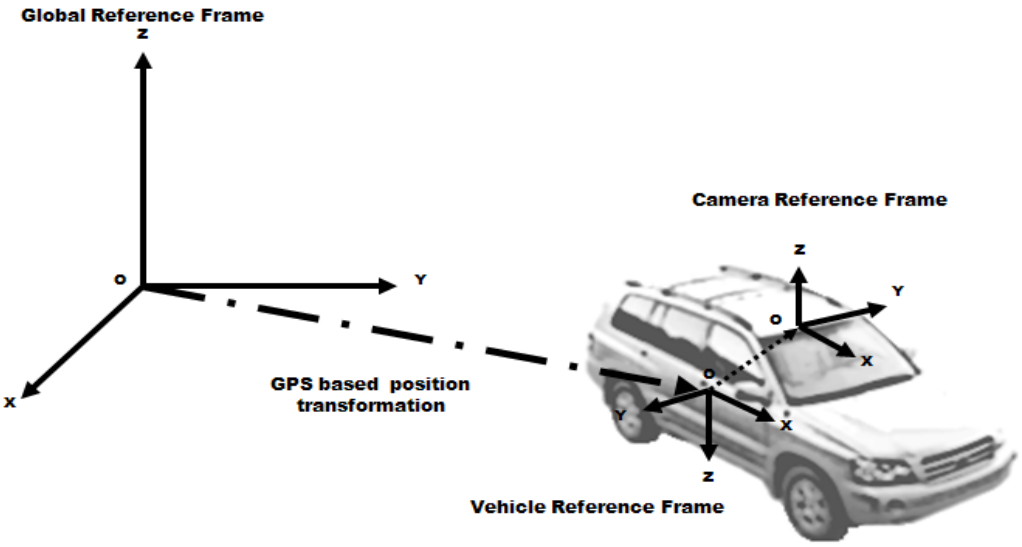


Figure 3-11. Traversability grid map [Solanki 2006]



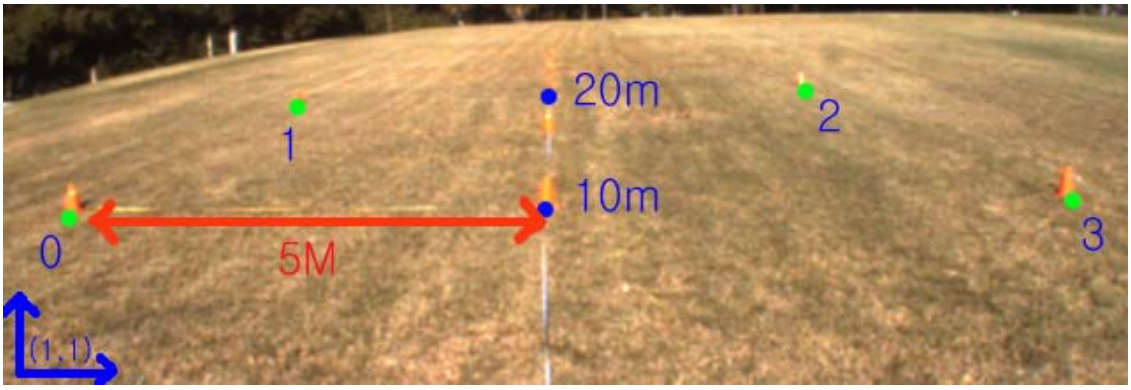
A



B

Figure 3-12. Coordinate systems. A) Relationship between camera view and world view and B) relationship between camera coordinate system, vehicle coordinate system, and earth coordinate system.

A



B

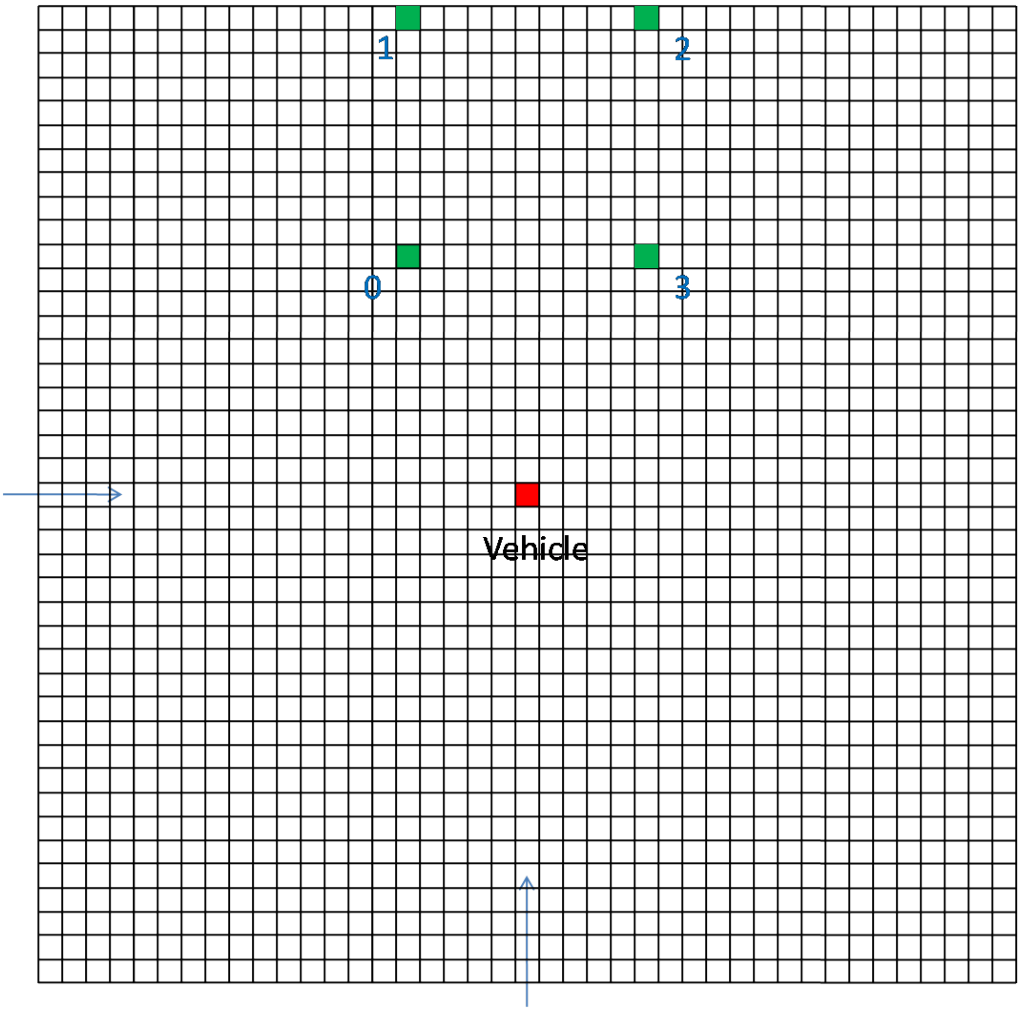


Figure 3-13. Perspective transformation reference points. A) Green dots are reference points in 320 × 108 size image at Flavet Field, University of Florida, and B) Green squares are reference points in a 40 × 40 meter traversability grid map image with 1 meter resolution. Red square is a vehicle.

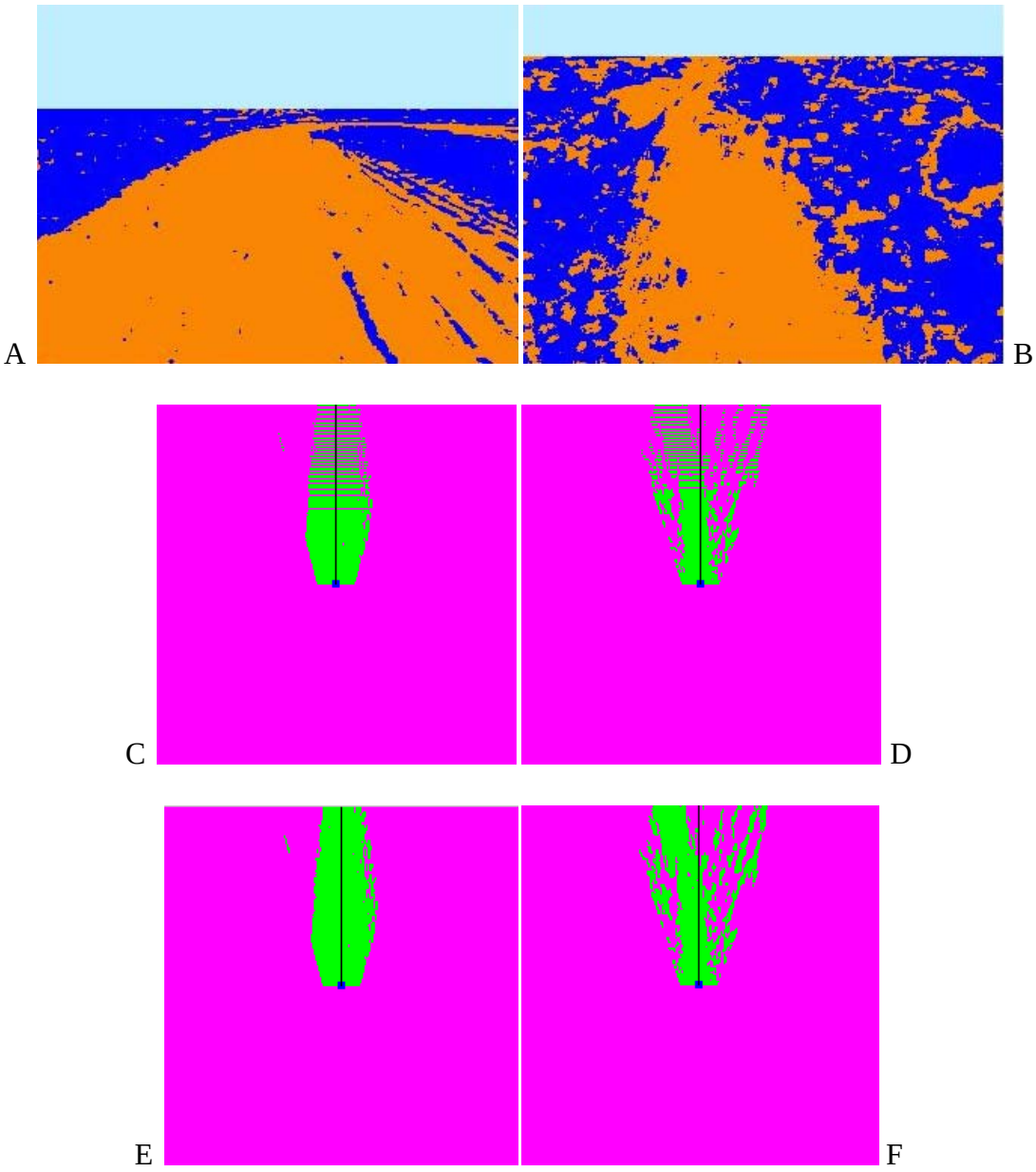


Figure 3-14. Transformed image. A) Classified image of Citra, B) classified image of DARPA Grand Challenge 2005 course, C) traversability grid map image without interpolation of Citra, D) traversability grid map image without interpolation of DARPA Grand Challenge 2005 course, E) traversability grid map image with interpolation, and F) traversability grid map image with interpolation.

CHAPTER 4
LANE FINDER SMART SENSOR

Introduction

The first period of autonomous vehicle development involved operating in an off-road environment where there were no lane demarcations, for example the DARPA Grand Challenge 2005 system and the Mars Explorer robot vehicle. However, at present, most vehicles drive in an urban environment that is generally paved with lanes defined by painted lines. Also, many autonomous vehicles depend on a Global Positioning System (GPS) to compute current location and project routes to desired locations. Unfortunately, GPS provides less accurate positioning solutions in the urban environment than in an open area environment, since urban infrastructure can blocks satellite signals or cause multi-path signals.

An urban traffic area provides more traffic facilities than off-road or highway. These facilities include bike lanes, curbs, sidewalks, crossroads, and various traffic signals. These urban environment facilities help human drivers understand their surroundings. In other words, from the point of view of robot perception, it also increases the complexity of the surroundings. On the highway, the lane lines are usually well-marked, with no sharp curvatures and no oncoming traffic. Therefore, with a driver assistant system, it turns out that highway driving is simpler than inner-city driving [Bräunl 2008].

The outdoor environment also presents an array of difficulties, including dynamic lighting conditions, poor road conditions, and road networks that are not consistent from region to region. Because of these limitations, an autonomous vehicle that is designed for urban driving needs a more adaptive lane tracking system. In this chapter, the lane and its property extraction and tracking system for an autonomous vehicle in an urban environment are described.

Camera Field of View

Two different field of view camera systems are applied to the Lane Finder Smart Sensor. Each vision system uses a different camera field of view and range to improve overall lane tracking system output. Figure 4-1 is a diagram of the camera field of view diagram. Figure 4-1 (A) shows the long-range field of view camera, mounted at the center of the vehicle, and Figure 4-1 (B) shows the short-range, but wide field of view of those two cameras. Two short-range cameras capture the source image from the vehicle front, so it provides enough high resolution and clear road lane source to calculate not only lane tracking, but also lane properties. Also, the two camera-based system provides a clear lane image even if another vehicle stands or travels in front of the robot vehicle. Figure 4-2 (A) and (B) show a two camera-based sample source image.

A long-range camera is a good source for future trajectory estimation. Its resolution is less than a close view camera, but it can see the area further down the road. Figure 4-2 (C) and (D) shows a long-range center camera sample image.

Canny Edge Detector

A road is defined by several characteristics. These may include color, shape, texture, edges, corners, and other features. In particular, the road lane is a human-made artificial boundary line that is marked with color and type information. Consequently, road lane lines contain dominant edge information and this cue is the most important feature for extracting road lane information. The edge of an image is created by several factors, for example, different 3-D object depths on a 2-D image plane, different reflection rates on a surface, various illumination conditions, and sudden object orientation variation.

Edge detection is accomplished by use of the Canny edge filter [Nixon 2008]. The Canny edge filter utilizes a pre-noise removing step and then computes omni-directional edge

information. Finally, it uses two threshold values that requires the detector to utilize much tuning and yields a sufficiently segmented image. Because of this multi-step approach, the Canny edge detector is widely used in many fields.

The following four steps comprise the Canny edge filter algorithm:

- Apply a derivative of a Gaussian noise filter.
- Compute x, y gradients, which are Sobel edge detection and gradient magnitude, respectively.
- Apply non-maximum suppression.
    - Thin multi-pixel wide “ridges” down to a single pixel width.
- Add linking and thresholding.
    - Use low, high edge-strength thresholds,
    - Accept all edges over low threshold that are connected to edges over high threshold.

To further enhance the edge detector’s performance, only the red channel of the source image is processed. This channel is used because it has the greatest content in both yellow and white and thus can provide the greatest contrast between yellow/white regions and asphalt background. Figure 4-3 depicts the results of the Canny edge filter with two sets of threshold values and Figure 4-4 shows Canny filter results in various situations.

First Order Line Decision

The lane finder software has two main functions. One is to establish lane departure warning and tracking, and the other is future trajectory estimation. For the first goal, there is no need for distance to detect a curved line. If the camera sees a local area, curved lines looks like straight lines. Therefore, a first-order line solution is applied for a lane departure warning system. It provides a lane center position with respect to the current driving vehicle position. This

solution is more robust and faster than a high order line solution; therefore, processing update rates can be increased.

For trajectory estimation, the camera has to see as far as it can because farther sight information provides greater environmental understanding for the vehicle. This information is used by the control element and consequently the control element can manage the vehicle at higher speed. However, if the camera has farther sight, just by the nature of the road, it can see many curved lines. Therefore, a higher order line solution is necessary for future trajectory estimation.

Hough Transform

The Hough transform is a technique that locates a certain shape in an image. The Hough transform was first implemented to find lines in images [Duda 1972] and it has been extended to further applications. It is a robust tool for extracting lines, circles, and ellipses. One advantage of the Hough transform in the lane extracting application is that it works well with many noise edges and/or partial line edges. From this point of view, the Hough transform can provide the same result as the template matching technique, but it uses many fewer computational resources [Nixon 2008]. Two disadvantages of the Hough transform is that it requires a large storage space and high processing power, and it produces as many lines as it can detect from the source image. Therefore, searching for road lane lines among all the detected Hough lines is necessary.

The Hough transform algorithm is as follows:

If one considers a line in an image domain, its equation can be written as

$$y = mx + c, \tag{4-1}$$

or

$$x\cos\theta + y\sin\theta = \rho, \qquad (4\text{-}2)$$

where ρ is a distance from the image domain origin to the line and θ is the orientation of ρ (the line from the origin perpendicular to the modeled line) with respect to the X-axis, as illustrated in Figure 4-5. Based on Eq (4-2), one can generate a Hough parameter space that plots possible (ρ, θ) values, which are defined by (x, y) points in the image. Finally, strong lines can be selected by searching maximum values in the Hough parameter space (ρ, θ). Figure 4-6 (B) shows the Hough space diagram when using the Canny filtered edge image in Figure 4-6 (A).

The maximum (ρ, θ) values in the Hough space is the strongest line in the image domain. However, this method cannot guarantee to extract a road lane line since an edge extracted image contains various noise pixels for many reasons. For example, an old tire track can register as a line in the road figure 4-4 (H) and different reflections from the road can appear to be a straight line like the edge in figure 4-4 (B).

Figure 4-7 shows the Hough transform line result in various situations. Figure 4-7 (B) show when a vehicle passes a crossroad area, so stop lines are detected. Also, another artificial line is easily detected and lane lines are blocked by other objects like grass. This case is shown in figure 4-7 (C). Figure 4-7 (D), right image, shows random noise edge pixels become a line object by coincidence. Due to the differing reflection rates of the road surface, a different reflection boundary area can create strong edges and can cause a false line object. Figure 4-7 (E) and (F), left images, illustrate this situation.

Because of this Hough transform property and various real world situations, two steps are required to correct it; a few lane candidate lines are extracted from the Hough space, then lane lines are searched for among the candidate Hough lines.

Lane Line Search

The Hough transform for line extraction finds many lines. These include not just road lane lines, but also other lines, like a crosswalk lane, for example [Hashimoto 2004]. Figure 4-7 shows sample results of this process. Consequently, among the candidate Hough lines, searching lane lines by using the properties of lane lines is a necessary step.

Since road lane lines are parallel to each other and usually at the same angle, two parameters are used for searching the lane lines: angle with respect to vehicle axle and distance from the vehicle center. Figure 4-8 shows this angle and distance. A binary search method is applied for detecting only lane lines among the many Hough lines and those line parameter's threshold values are selected by the heuristic method.

Polynomial Line Decision

The Hough transform-based first order lane line solution is usually enough for lane departure or a lane tracking system. However, if an autonomous vehicle drives at high speed and/or drives on a curved road, an autonomous vehicle control system needs further traversable road information. For example, if a vehicle drives at 40 miles per hour, it means that that vehicle drives around 18 meters per second. Therefore, the perception system has to provide at least an 18 meter traversable area per second from the vehicle location for safe driving. Table 4-1 shows vehicle travel distance per camera frame rate.

The main goal of a perception system is to construct as accurate a representation of the world as possible. Clearly, accurate and high resolution information helps an autonomous vehicle controller control a vehicle properly and safely. Thus, a long-range camera and higher order lane line solution are necessary for future trajectory estimation for high speed driving on a curved

road. Figure 4-9 (B) shows lane extraction and lane center trajectory error in far sight. This case can cause a future path estimation error.

Table 4-1. Vehicle travel distance per camera frame rate

mile/hour	5	10	15	17.5	20	25	30	40	50	60	70
kilometer/hour	8.05	16.09	24.14	28.16	32.19	40.23	48.28	64.37	80.47	96.56	112.65
meter/hour	8046.70	16093.40	24140.10	28163.45	32186.80	40233.50	48280.20	64373.60	80467.00	96560.40	112653.80
meter/sec	2.24	4.47	6.71	7.82	8.94	11.18	13.41	17.88	22.35	26.82	31.29
meter/frame											
35	0.06	0.13	0.19	0.22	0.26	0.32	0.38	0.51	0.64	0.77	0.89
30	0.07	0.15	0.22	0.26	0.30	0.37	0.45	0.60	0.75	0.89	1.04
25	0.09	0.18	0.27	0.31	0.36	0.45	0.54	0.72	0.89	1.07	1.25
20	0.11	0.22	0.34	0.39	0.45	0.56	0.67	0.89	1.12	1.34	1.56
17	0.13	0.26	0.39	0.46	0.53	0.66	0.79	1.05	1.31	1.58	1.84
15	0.15	0.30	0.45	0.52	0.60	0.75	0.89	1.19	1.49	1.79	2.09
10	0.22	0.45	0.67	0.78	0.89	1.12	1.34	1.79	2.24	2.68	3.13
5	0.45	0.89	1.34	1.56	1.79	2.24	2.68	3.58	4.47	5.36	6.26
1	2.24	4.47	6.71	7.82	8.94	11.18	13.41	17.88	22.35	26.82	31.29

Cubic Splines

A spline is a function that describes polynomials for formulating a curve. The spline has been developed and applied in many fields, for example, computer aided design (CAD), computer aided manufacturing (CAM), computer graphics (CG), and computer vision (CV). A number of variants have been developed to control the shape of a curve, including Bezier curves, B-spline, non-uniform rational B-spline (NURBS) and others [Sarfraz 2007].

In this research, the Cubic spline method is applied to the curve lane model. Unlike other spline models, the Cubic spline passes a set of all N control points and it can use different boundary conditions for each application.

The following is the Cubic spline condition:

1. Curve model is a third order polynomial:

$$f_i(x) = a_i + b_i x + c_i x^2 + d_i x^3 \tag{4-3}$$

and spacing between the successive data points is

$$h_i = x_{i+1} - x_i. \tag{4-4}$$

2. Curves pass through all points,

$$f_i(x_i) = f_{i+1}(x_i) = y_i. \tag{4-5}$$

3. The first order derivative, the slope of curve, is equal on either side of a point,

$$f_i'(x_i) = f_{i+1}'(x_i). \tag{4-6}$$

4. The second order derivative is equal on either side of a point,

$$f_i''(x_i) = f_{i+1}''(x_i). \tag{4-7}$$

5. For a natural spline case, the second order derivative of the spline at the end points is zero:

$$f_1''(x_0) = f_n''(x_n) = 0. \tag{4-8}$$

In matrix form, one can write:

$$\begin{bmatrix} 2(h_1 + h_2) & h_2 & \cdots & \\ h_2 & 2(h_1 + h_2) & \ddots & h_{n-2} \\ & & \cdots & 2(h_{n-2} + h_{n-1}) \end{bmatrix} \begin{bmatrix} f_2 \\ f_i \\ \cdots \\ f_{n-1} \end{bmatrix} = 6 \begin{bmatrix} \frac{y_3 - y_2}{h_2} - \frac{y_3 - y_2}{h_2} \\ \cdots \\ \frac{y_n - y_{n-1}}{h_{n-1}} - \frac{y_{n-1} - y_{n-2}}{h_{n-22}} \end{bmatrix}.$$

Finally, the Cubic spline parameters are calculated as follows:

$$\begin{aligned} a_i &= (f_{i+1} - f_i) / 6h_i \\ b_i &= f_i / 2 \\ c_i &= \frac{y_{i+1} - y_i}{h_i} - \frac{2h_i f_i + h_i f_{i+1}}{6} \\ d_i &= y_i. \end{aligned} \tag{4-9}$$

Figure 4-10 is a diagram of a Cubic spline curve.

Even if a vehicle drives a curved road area, it can be assumed that the vehicle drives in a straight lane from the local point of view. In many cases, a lane curve line starts from a straight line and gradually changes its shape to match the curve. Figure 4-11 (C) shows this case. Since the Canny edge + Hough transform-based first order line solution provides a fairly robust solution, the Hough transform-based line geometry is a good initial source for finding higher order line geometry.

Figures 4-9 (B) and 4-11 (B) show the center camera and two camera lane line overlay image using the Hough transform for a curved road, and Figure 4-12 (B) and (C) shows straight and curved line results, respectively.

While the Cubic spline is well behaved for a lane curve model representation [Kim 2006, 2009], it is possible to generate an overshot curve because of one or more false intermediate control points from noise pixels. Therefore, selecting control points is a key step to creating a lane curve model, so it has to be carefully selected.

Control Points

Since the Cubic spline passes through all control points, those points have to be selected precisely. All control points have the same weight; therefore, an incorrectly selected control point or points can create an erroneous lane model. This problem occurs more at the far side of an image. In the real world environment, obtaining a clear lane edge filtered image is almost impossible. Non-lane edge pixels exist randomly by the nature of the world, so the far side of an image is easily washed out compared to the near side of an image. For this reason, a new control point selection method is proposed in this dissertation.

The following describes this method:

- Normally, a curved line's start points match the Hough transform line. Therefore, after computing the Hough transform line, N-distance pixels from the Hough line are selected for the curved line candidate's pixels. Figure 4-12 (A) shows the Canny filtered edge image and Figure 4-12 (B) shows the N-pixel distance area from the Hough transform line. The resulting edge is shown in Figure 4-12 (C).

- The Figure 4-12 (C) image still has many outlier pixels and two lane sides. The greatest size of connected edge pixels is extracted, as shown in Figure 4-12 (D). At this step, only curvature is left to be determined, if a lane is a curved line.

- Next, the normal vectors from the Hough transform line to the curvature pixels and the distance are computed. Based on the image resolution and the real world distance, control points are selected. In Figure 4-12 (F), the blue line shows the normal vectors from the Hough transform line to curvature pixels.

- Finally, the Cubic spline is computed using selected control points.

Lane Model

Because of a property of the Hough transform, many lines, which include not just road lane lines, but also other lines, are detected, but only lane lines need to be classified [Hashimoto 1992]. Therefore a search method is applied to detect only the two lane lines among the many line candidates. While line angle and distance parameters are used for this procedure, sometimes more than two lines meet these line angle and distance conditions. In those cases, the closest line is selected as the lane line.

Those angle and distance parameters are also employed to verify the lane model assumption in which the road is modeled as a plane and the lane lines are parallel to each other in the global view. Also, this lane line model assumption can be applied not only as the vehicle drives a straight road, but also as the vehicle drives along a curved road. Since the camera field of view is local and the update rate is around 20Hz, the far area lane correction error caused by a first order line assumption can be ignored.

A global coordinate view, also called a bird's-eye view, is a good coordinate system for checking the lane model. Figure 4-14 (A) shows detected lines on a straight road and Figure 4-14 (B) shows detected lines on a curved road from a bird's-eye view.

Lane Estimation

In the real world, road conditions may be such that, for a moment, only one or no lane lines are visible. Even on roads in good repair with proper markings, the problem of losing a lane reference may occur. This can happen when there is segmented line painting, intersections, or lane merges. Also, it can happen there is a partial obstruction by another vehicle or there is a strong shadow on the line on a bright day.

For these instances, an estimation technique is employed to estimate the likely location of the missing lane boundary line. This is accomplished by using a previous N number of line parameters that are slope and intersection in the first order line model:

$$y = mx + c. \tag{4-10}$$

Eq (4-10) defines a linear line with slope *m* and intersection *c* in a Cartesian coordinate system and (x,y) is the image pixel location.

The linear least-squares estimation technique is applied to estimate a first order lane line's angle and intersection parameters. Whenever a lane line is detected, N numbers of line angle and intersection parameters are stored in a buffer. Then when the vehicle passes the segmented road line area or crossroad area, those stored parameters are employed for estimating the likely position of the line. Finally, estimated line parameter's quality is checked by the lane line model. If an estimated line meets a lane model, it is used to compute lane correction data. However, even if the estimated line parameters are good enough to computes lane correction, the estimated parameters use old data again and again without considering vehicle behavior. Therefore only N-

number of the estimated data is processed, otherwise the confidence value is set to zero. Figure 4-15 depicts a line parameter estimation flowchart. This method can be applied without an accurate vehicle dynamic model [Apolloni 2005].

Let y be the observation vector and N the observation number. The observation vector can be written as

$$y = [y(1), ..., y(N)]^t \tag{4-11}$$

The least square estimate is the value of *h* that minimizes the square deviation:

$$J(h) = (y - Xh)^T (y - Xh), \tag{4-12}$$

where X is an N × P matrix where P is the order of the polynomial model of the function.

The solution can be written simply as

$$h = (X^T X)^{-1} X^T y. \tag{4-13}$$

Figure 4-16 depicts sample results of the lane estimation result. Figure 4-16 (A) and (C) show two sequential source images from the left camera. Figures 4-16 (B) and (D) show the detected (blue) line and the estimated (orange) line, when a vehicle passes the segmented line area. From Figure 4-16, it is clear that the estimation process can effectively determine the location of the missing boundary and is useful when dealing with segmented lines.

Figure 4-17 shows line angle parameter estimation results. The X axis shows frame number of sequential images and the Y axis shows the Hough transform line's angle parameter. When an autonomous vehicle passes the segmented line area, the lane shows and disappears again and again. Detected line angle parameters displayed in blue points and estimated line angle parameters are displayed in orange points.

Lane Center Correction

Lane Correction by Two Cameras

Information, such as the estimated center of the current lane combined with lane width, is used to determine the vehicle orientation within the lane. After converting data from the image coordinate system to the real world coordinate system, the distances between the detected/estimated lane lines and the vehicle side are computed (Figure 4-18, purple arrows). From these distance values, lane correction data (Figure 4-18, blue arrow) and lane width (Figure 4-18, red arrow) are easily computed. Eq (4-14) shows the definition of the lane correction

$$Correction = d_R - d_L, \tag{4-14}$$

and Eq (4-16) shows how to compute lane width using two distances between the vehicle side and lane boundary,

$$W_{lane} = d_L + d_R. \tag{4-15}$$

where W_{lane} is lane width, ande d_L , d_R is distance between vehicle side and lane boundary. By the camera calibration, the relationship between the real world distance and the image pixel distance is measured. Figure 4-19 (A, B, C, and D) shows two camera-based lane finder calibration images. A resolution at 4, 5, 6, 7, 8, 9, 10, 12 and 14 meters from the vehicle reference point are summarized in Table 4-2. The resolution at the 5 meter position is around 0.66 centimeters per pixel and at the 7 meter position is around 1 centimeter per pixel.

Table 4-2. Two side cameras' horizontal pixel resolution on 640 x 380 image.

Y-Distance	X-pixel location (Center)	X-pixel location (Left)	Pixel distance	Real distance (cm)	Resolution (cm/pxl) on 640 x 380	Resolution (cm/pxl) on 320 x 190
5	470	150	320	212	0.6625	1.3250
6	452	192	260	212	0.815384615	1.6308
7	436	222	214	212	0.990654206	1.9813
8	426	240	186	212	1.139784946	2.2796
9	418	258	160	212	1.325	2.6500
10	412	268	144	212	1.472222222	2.9444
12	404	286	118	212	1.796610169	3.5932
14	398	296	102	212	2.078431373	4.1569

Since two wing cameras can see both vehicle sides, and face the ground, and those cameras' field of view starts from the vehicle's front axis, the accuracy of the lane width and lane center in near view is better than a camera with a far view. Based on the perspective property of a camera or human eye, certain distance points from the vehicle rear axis are selected as a lane center estimation positions. Figure 4-19 (E) and table 4-3 shows that relationship between real world position and camera pixel position from the vehicle reference position. The image resolution from the 5 to 8 meter position is high, but from 9 to 15 meters resolution is low. Therefore, it is unnecessary to select a lane center estimation position every meter. The 5, 6, 7, 8, 9, 10 meter positions and the 12 and 14 meter positions from the vehicle reference points were selected.

Each position's lane correction distance is computed and these values are collected by the LFSS Arbiter component through an experimental JAUS message along with other sensors' lane correction data for estimating the future trajectory. Table 4-4 shows the lane center correction JAUS message data structure in C/C++. This message contains not just lane center correction data, but also lane properties, like lane color, type, and width.

Table 4-3. Side cameras pixel location by real distance on 640 × 380 resolution image.

Distance from Vehicle reference (meters)	Y-location from bottom (pixels)
5	125
6	184
7	220
8	245
9	261
10	276
11	288
12	298
13	305
14	313
15	321

Table 4-4. Lane center correction experimental JAUS message definition

```
typedef struct LaneFinderCorrectionStruct
{
        float rangeM;       // distance ahead of IMU
        float offsetM;      // lane correction
        float offsetConfidence;
        JausUnsignedInteger offsetOrigin; // offset from center
                                          // or offset from curb

        //Road Width
        float roadWidthM;
        float roadWidthConfidence;

        //Lane Width
        float laneWidthM;
        float laneWidthConfidence;

        //Normally, these values are computed by vision sensor
        JausUnsignedInteger boundaryColor;
        float boundaryColorLeftConfidence;
        float boundaryColorRightConfidence;

        //Lane Type
        JausUnsignedInteger boundaryType;
        float boundaryTypeLeftConfidence;
        float boundaryTypeRightConfidence;

        struct LaneFinderCorrectionStruct *nextCorrection;
} LaneFinderCorrectionStruct;
```

Lane Correction by One Camera

The center camera's field of view is larger than the two LFSSWing cameras' fields of view. This is designed for long-range lane correction for farther future estimation. Points at 8, 10, 15, 20, 25, and 30 meters from the vehicle reference are selected to compute long-range lane correction. Table 4-5 summarizes horizontal pixel resolution by center camera and its resolution is lower than two camera-based lane corrections which are shown at Table 4-2.

Table 4-5. The LFSS center camera's horizontal pixel resolution on 640 x 218.

Y-distance	X-pixel location (center)	X-pixel location (left)	Pixel distance	Real distance (cm)	Resolution (cm/pxl) on 640 x 218	Resolution (cm/pxl) on 320 x 109
8	317	15	302	424	1.40397351	2.8079
10	316	72	244	424	1.737704918	3.4754
15	315	149	166	424	2.554216867	5.1084
20	315	191	124	424	3.419354839	6.8387
25	315	215	100	424	4.24	8.4800
30	315	231	84	424	5.047619048	10.0952

Except for resolution and field of view, the lane correction algorithm is the same as the LFSSWing algorithm. Lane correction values are computed by using Eq (4-14). These lane correction values are sent to the LFSS Arbiter component with LFSSWing component correction values.

Lane Property

Every object color consists of two pieces of color information; real object color and lighting color. Because of this property, it is not easy to obtain the exact object color in the outdoor environment in real time. For a moving object, like an autonomous vehicle, lighting source direction changes over time, so it depends on the surroundings and time of day. For this reason, color information is not selected as a primary feature in the lane tracking system.

However people can acquire more information from not just line type, but also line color. For example, a vehicle cannot cross the yellow line.

Although it is hard to indentify real object color in the real world. It is not hard to categorize the line color even if a color source is acquired from an outdoor environment, since normal painted lane lines are only yellow or white. A histogram matching method using color pixel distance is proposed to identify lane color. First, lane line mask images are generated using the bottom part of the image. Because this part includes the most vivid color, high line pixel resolution and partial lines help it to reduce computation power needs. A part of the detected or estimated Hough lines are used to generate the mask images. Figure 3-17 shows two camera field of view mask images.

After generating lane line mask images, the color distance between the lane line color and each color of the lane color look-up table is calculated using Eq. (4-16).

$$C_d = \min \sqrt{(p_r - L_r)^2 + (p_g - L_g)^2 + (p_b - L_b)^2}, \qquad (4\text{-}16)$$

where $P_{r,g,b}$ is the RGB value of lane pixels and $L_{r,g,b}$ is the RGB value of the look-up table color. Table 4-6 shows lane color look-up table.

Table 4-6. Lane color look-up table

Color	Red	Green	Blue
Yellow	255	255	150
White	255	255	255
Black	0	0	0

In this procedure, asphalt color distance is also calculated and then those pixels are ignored for classifying lane color. Finally, color distance values are utilized for creating a histogram and deciding the lane color that has the minimum sum of color distances [Krishnan 2007].

Uncertainty Management

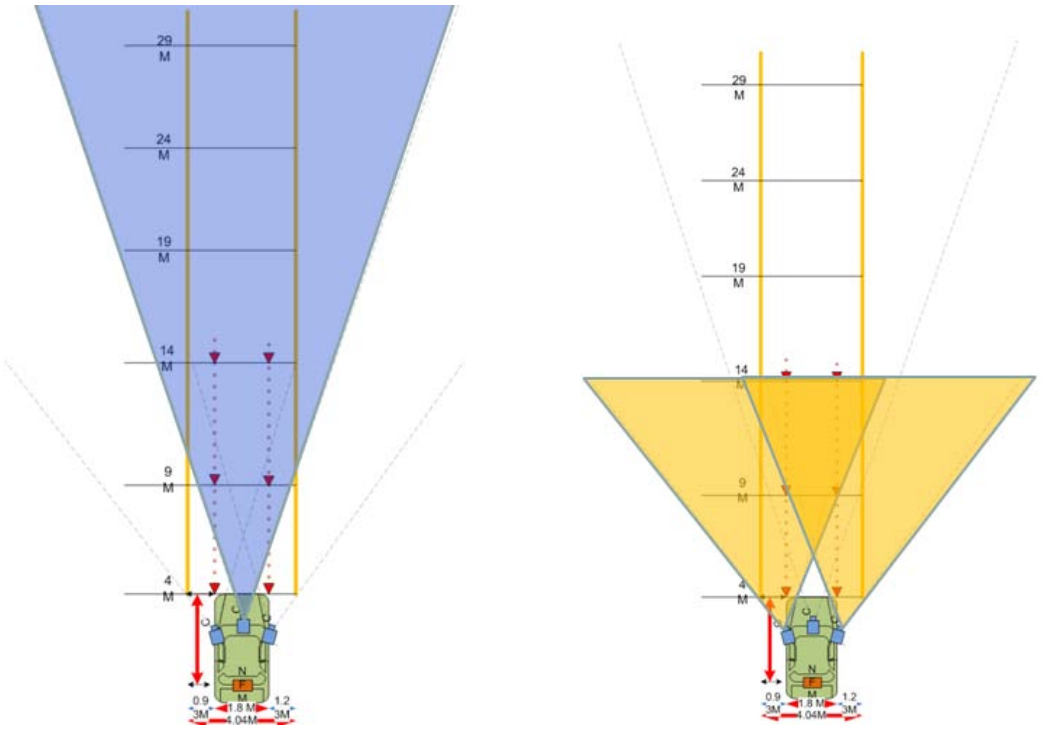
The human perception system consists of more than one sensing element, for example, visual, aural, and tactile senses. Those senses merge together with past experience and using the brain to make judgments for proper action. In a robotics perception system, the same approach is demanded because there is no sensor equipment for capturing all sensing information at one time. Therefore, a sensor fusion process is required in a multiple sensor-based robotic system and each sensor's output data management has an important role in this process.

When different types of sensors are tasked with the same goal, the system has to identify each sensor's output quality. For example, two different field of view camera systems are utilized for the lane tracking system and a LADAR-based lane tracking is also developed. Additionally, even though the vision based lane tracking system gives reliable output in most cases, there will be occasions when there is the risk of poor or even erroneous output from the system because of the environment, machine failure, and so on, and these cases need to be identified.

In addition to the lane tracking outputs, confidence values are provided for uncertainty management. A root mean square deviation (RMSD) value is used to determine the confidence value of the lane tracking system output, such as lane center corrections, lane width, and lane color. The RMSD measures the difference between actual measurement values and predicted values. In this system, two things are assumed; first, the previous data is the measurement value.

Second, the current measurement or estimate data is the predicted value. From this assumption, the RMSD is calculated as the confidence value using Eq (4-17):

$$\mathrm{RMSD}(\hat{\hat{\hat{\theta}}}) = \sqrt{\mathrm{MSE}(\theta)} = \sqrt{\mathrm{E}((\theta - \theta)^2)}. \qquad (4\text{-}17)$$



A B

Figure 4-1. Camera field of view diagram. A) The center camera's view, B) the two side cameras' view.

Figure 4-2. Camera field of view. A) Two camera view in an open area, B) two camera view in a traffic area, C) center camera view at the Gainesville Raceway, and D) center camera view when other vehicle blocks a lane.

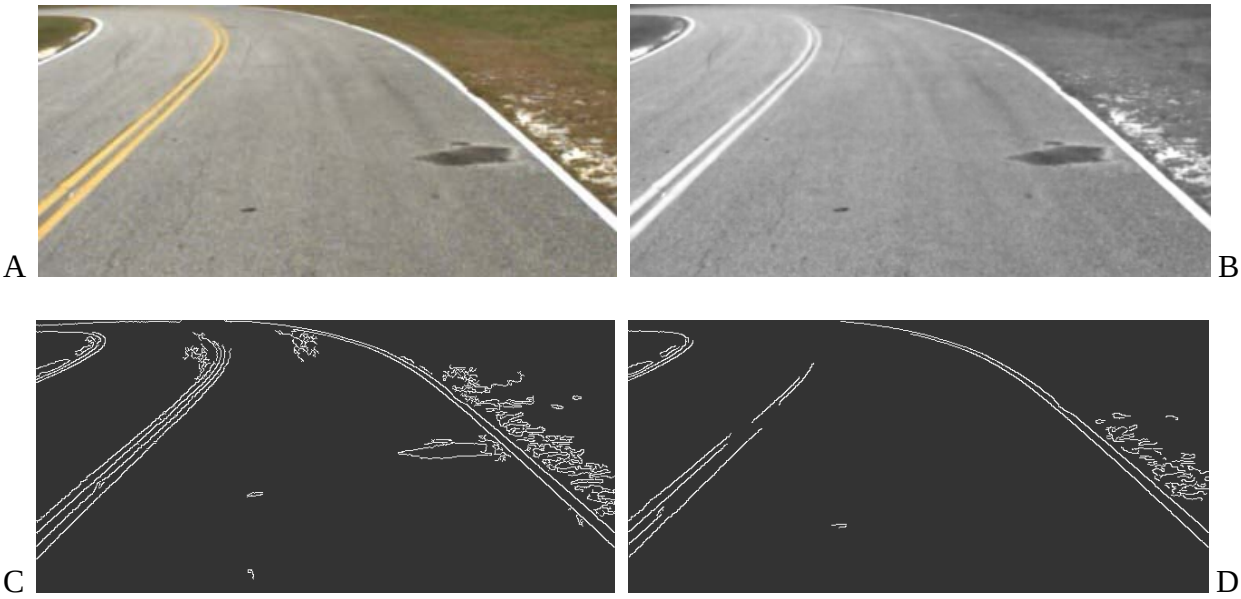
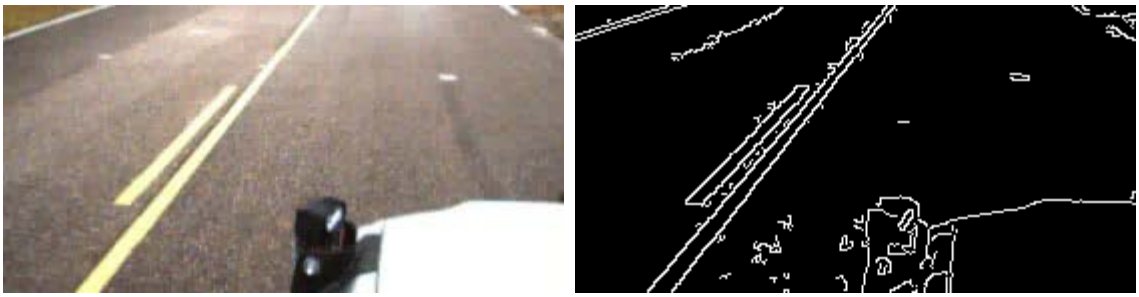
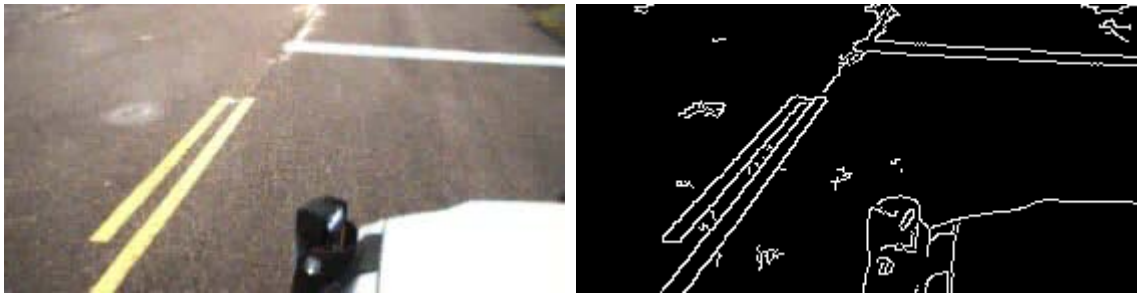


Figure 4-3. Canny filtered image samples. A) Original road Image, B) red channel image, C) Canny filter image with 50/200 threshold value, and D) Canny filter image with 130/200 threshold value.

Figure 4-4. Two-camera Canny filtered image in various situations. A) Solid and segmented line, B) segmented line, C) stop line, D) partial block of line, E) curved line, F) noise on the road, G) wipe out line, and H) old tire track on the road.

A 

B 

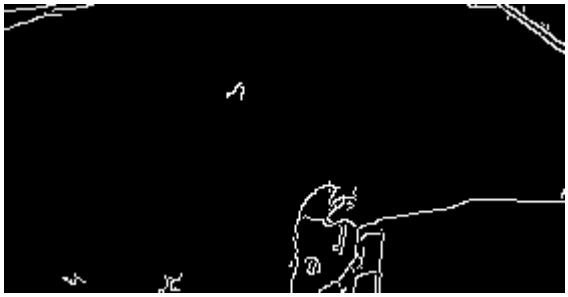
C 

D 

E 

F

G



H

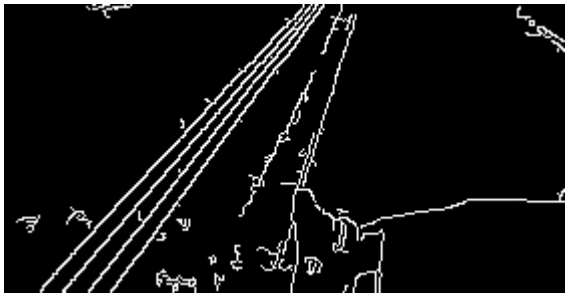


Figure 4-4. Continued.

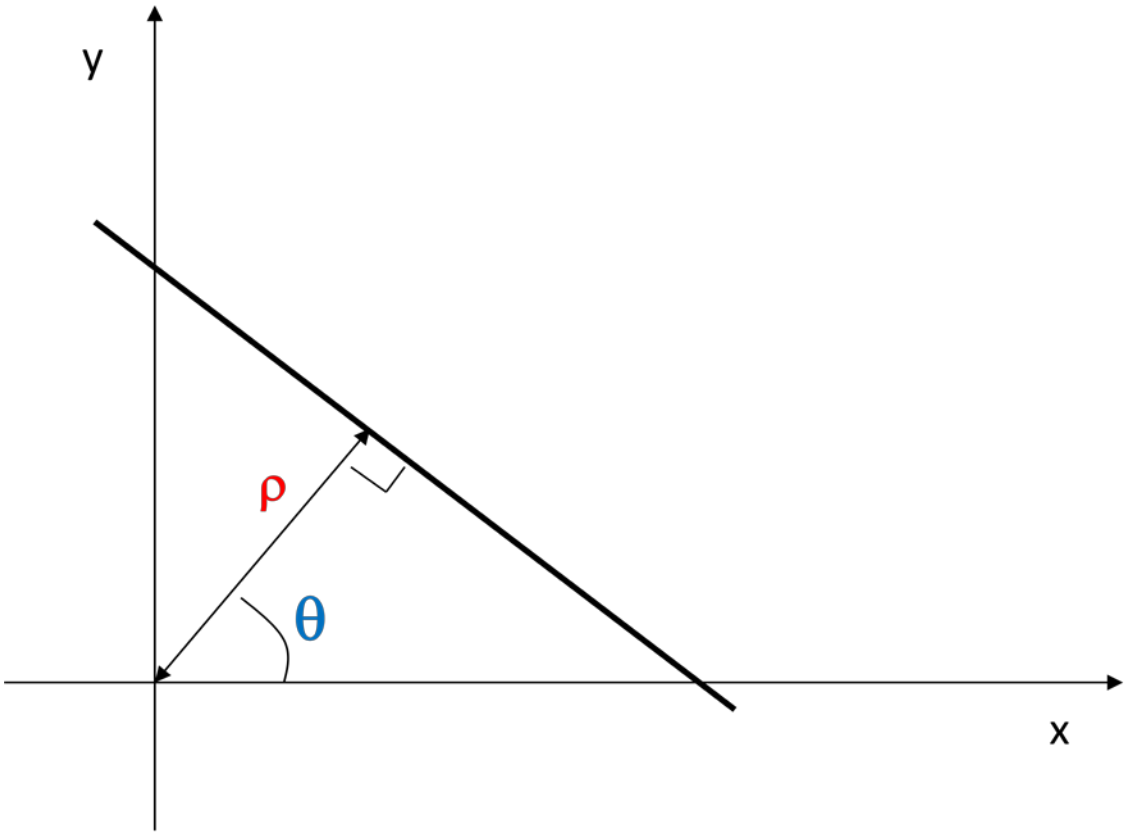
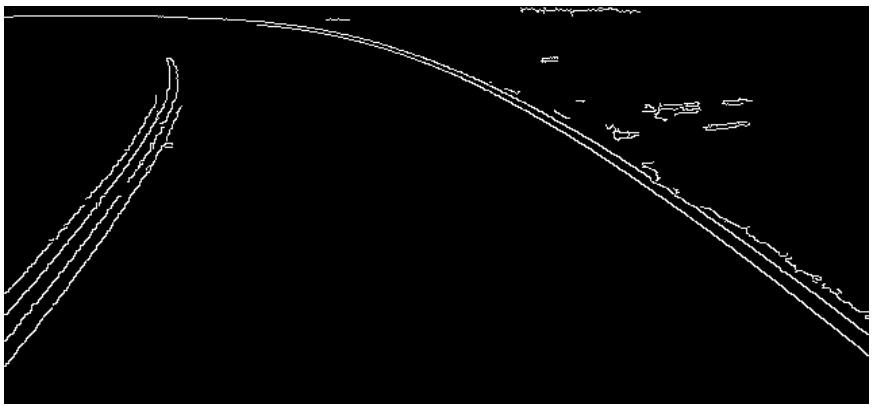
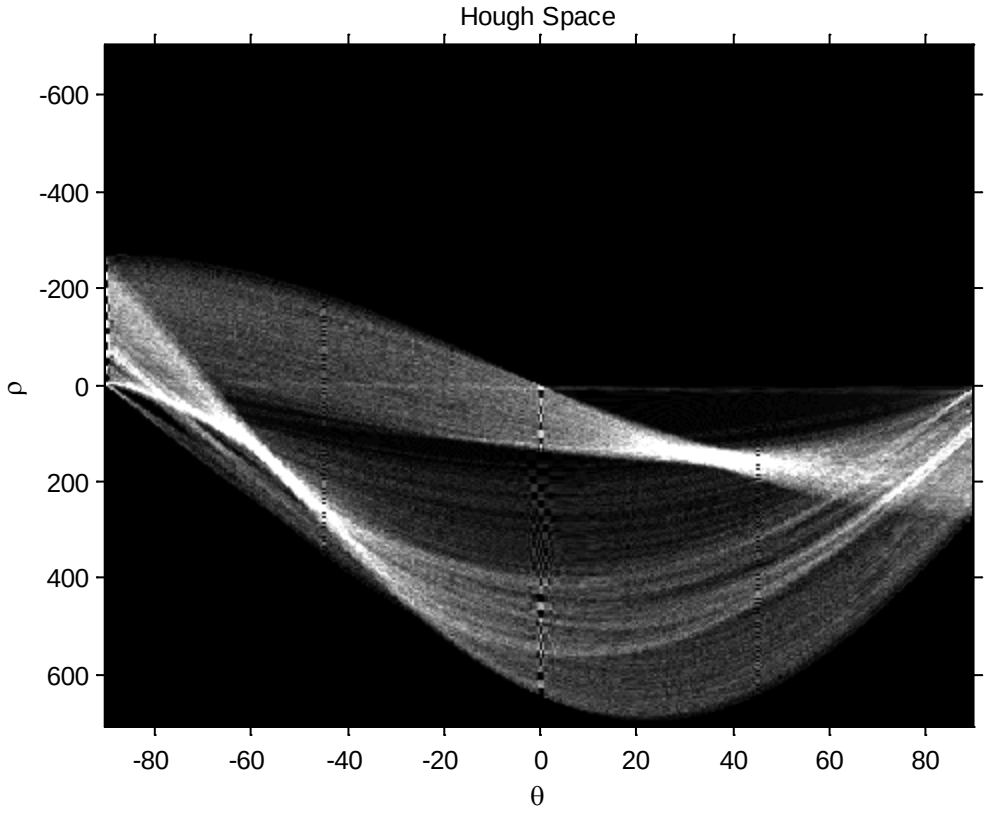


Figure 4-5. Hough space parameters in image coordinate system.



A

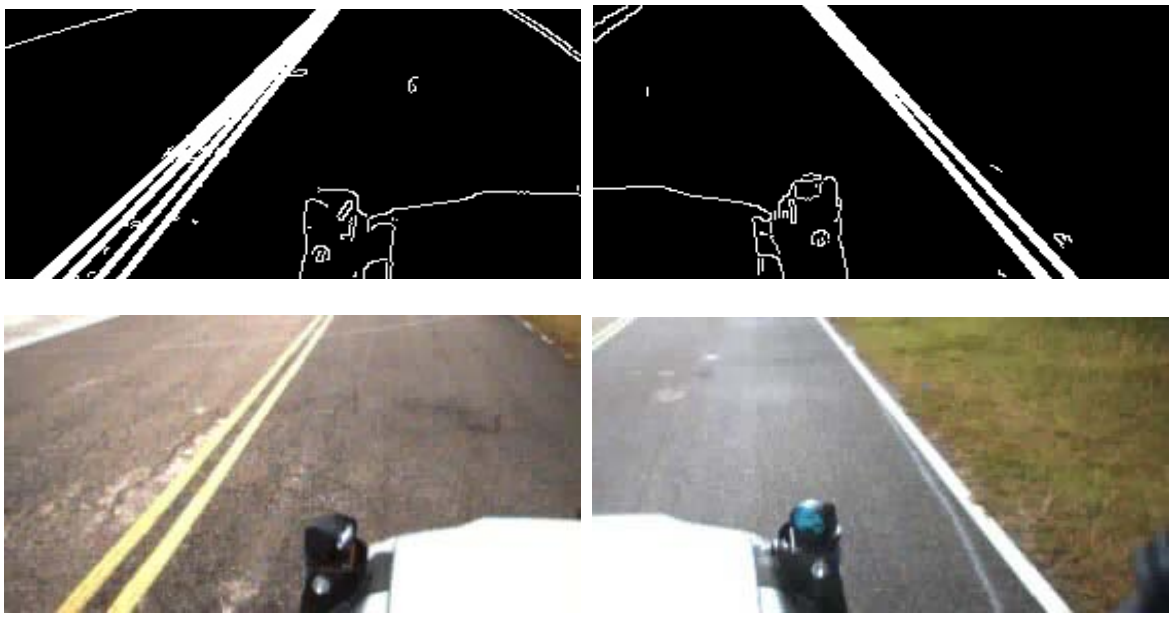


B

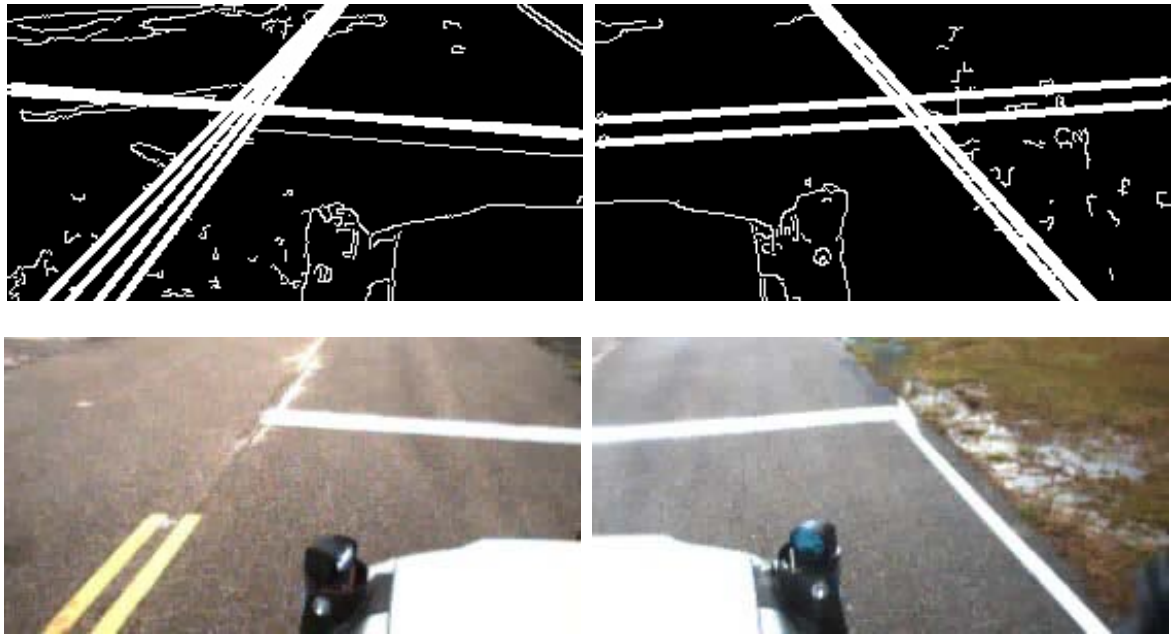
Figure 4-6. Hough space. A) The Canny filtered image at image space, B) A's Hough space.

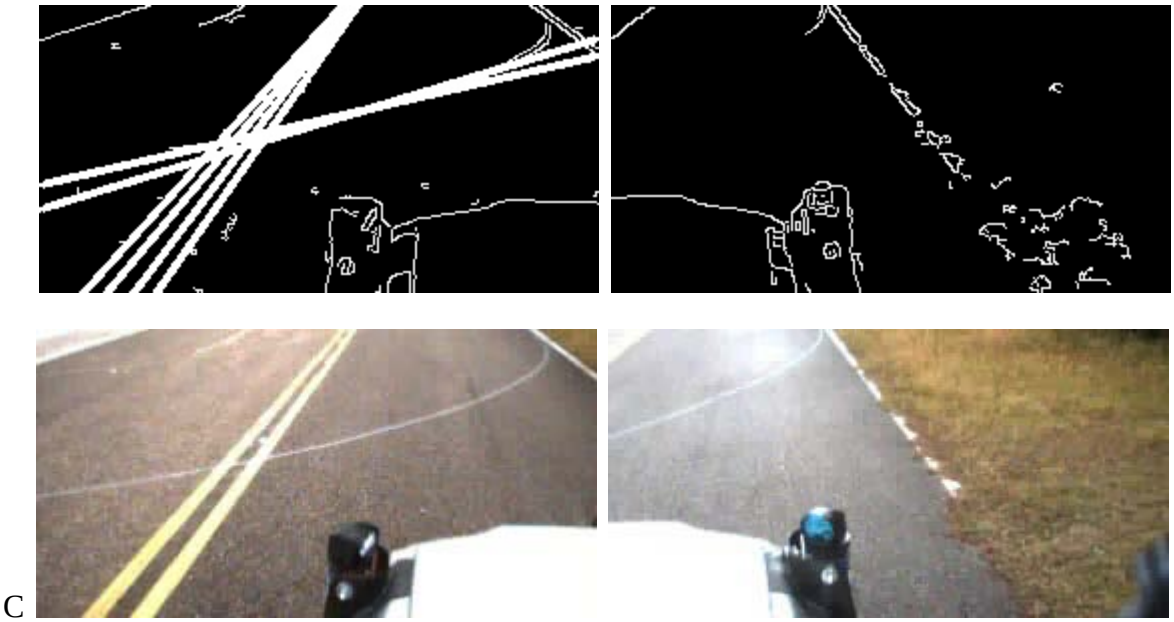
Figure 4-7. Hough line transform results. A) Straight road, B) stop line, C) other lines on the middle of the road, D) other lines by noise edge pixel, E) other lines by illumination difference on the road, case I, and F) other lines by illumination difference on the road, case II.

A

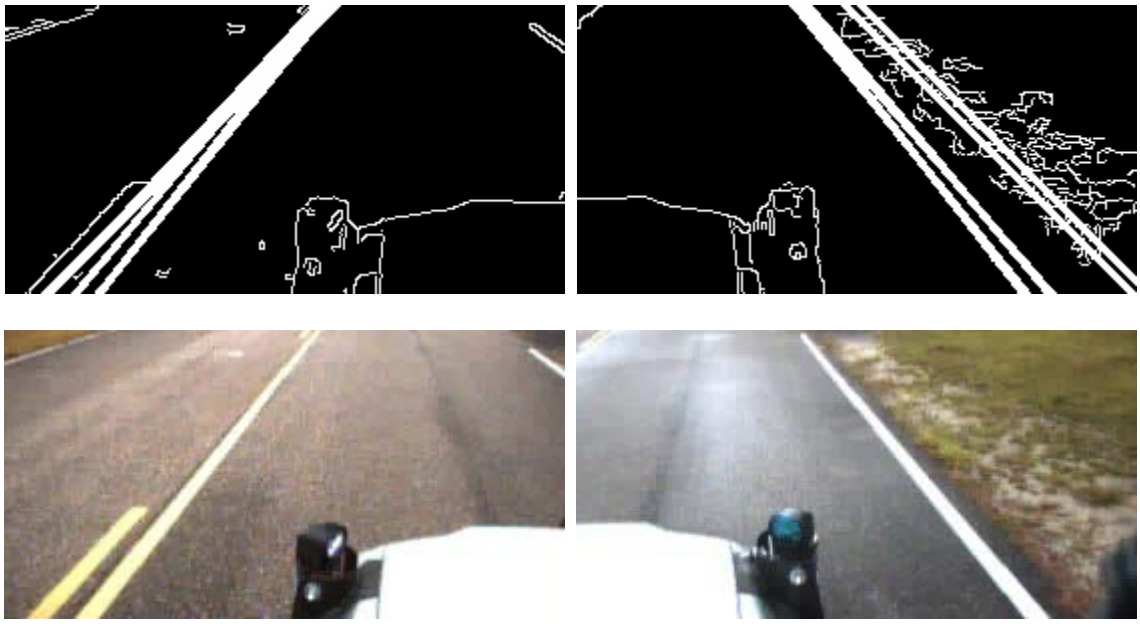


B





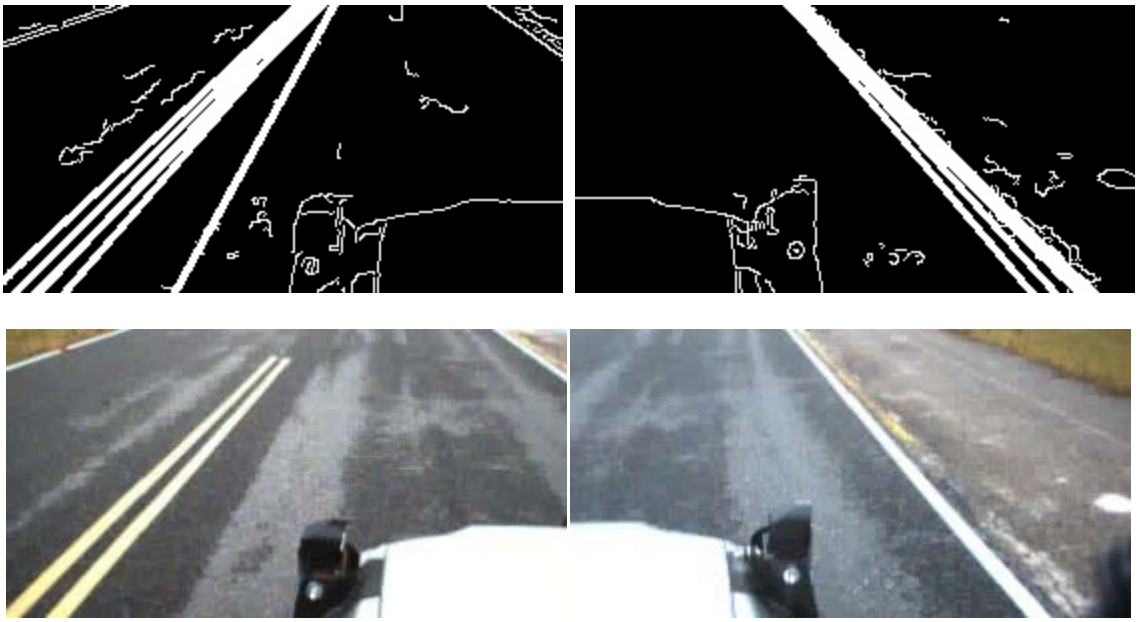
C



D

Figure 4-7. Continued.

E



F

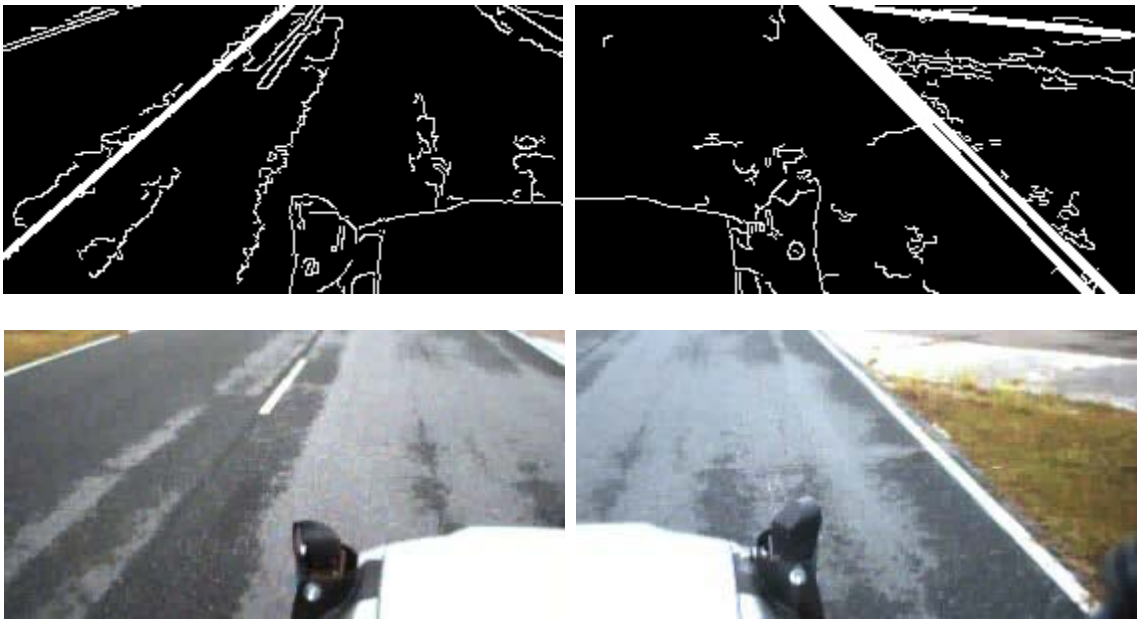


Figure 4-7. Continued.

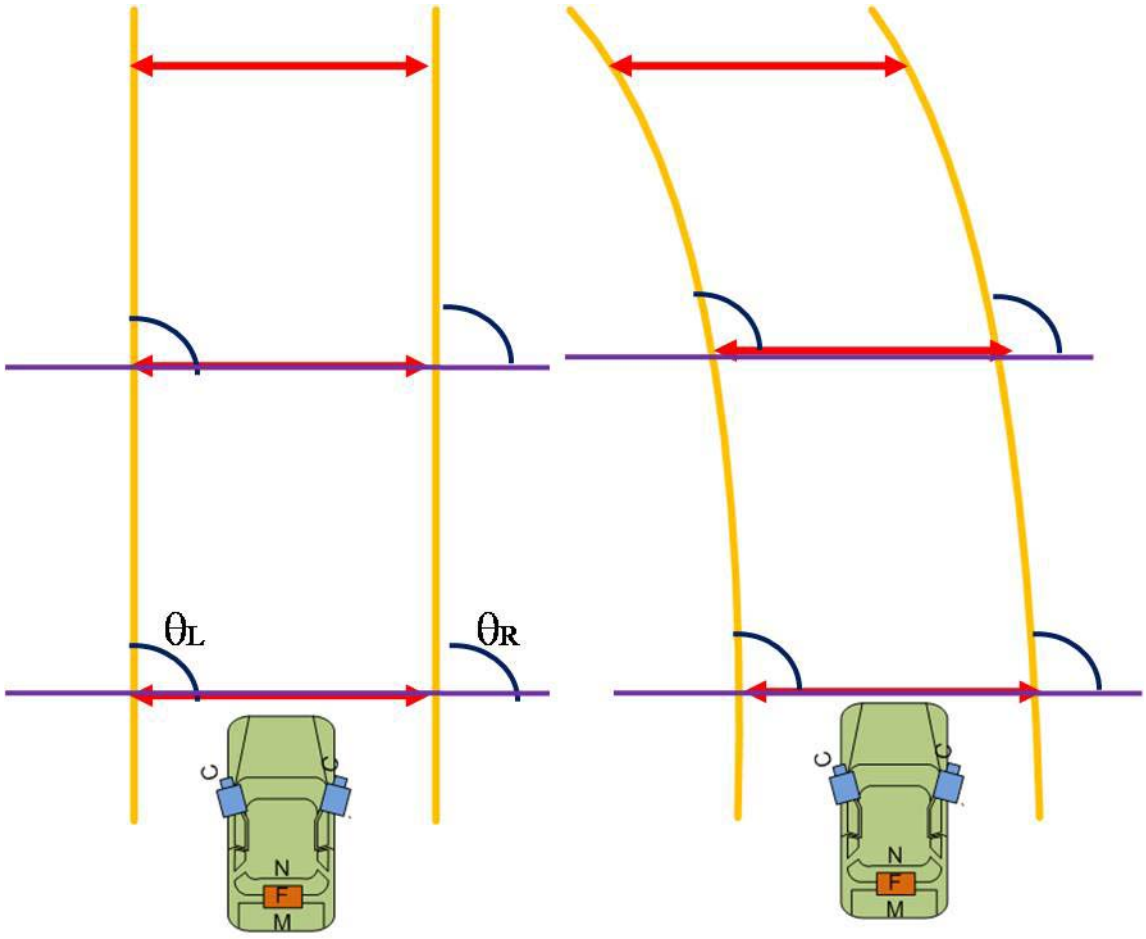
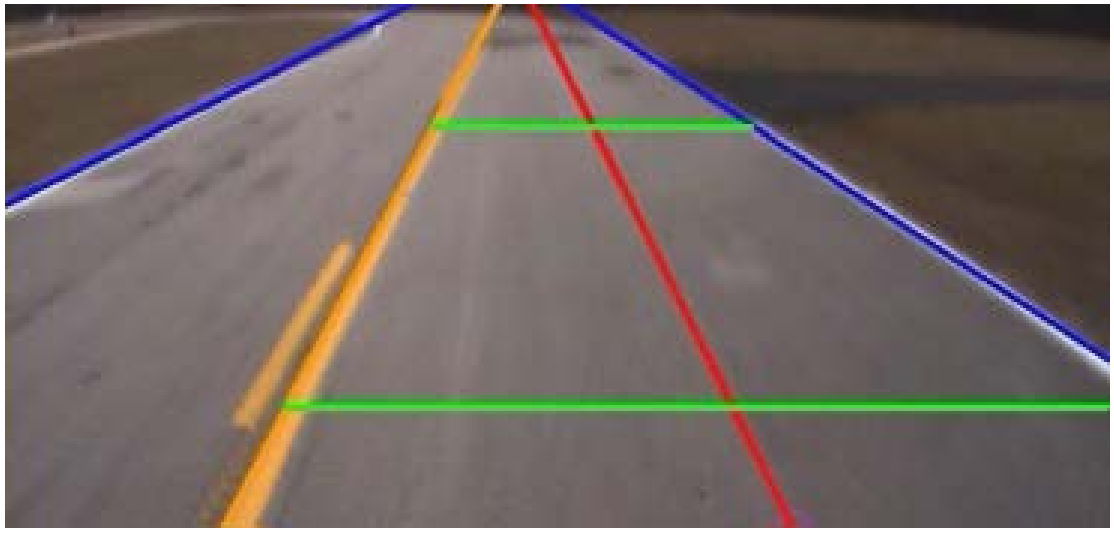


Figure 4-8. Lane line checking parameters, angle (Θ) and distance (d_L , d_R).

A



B

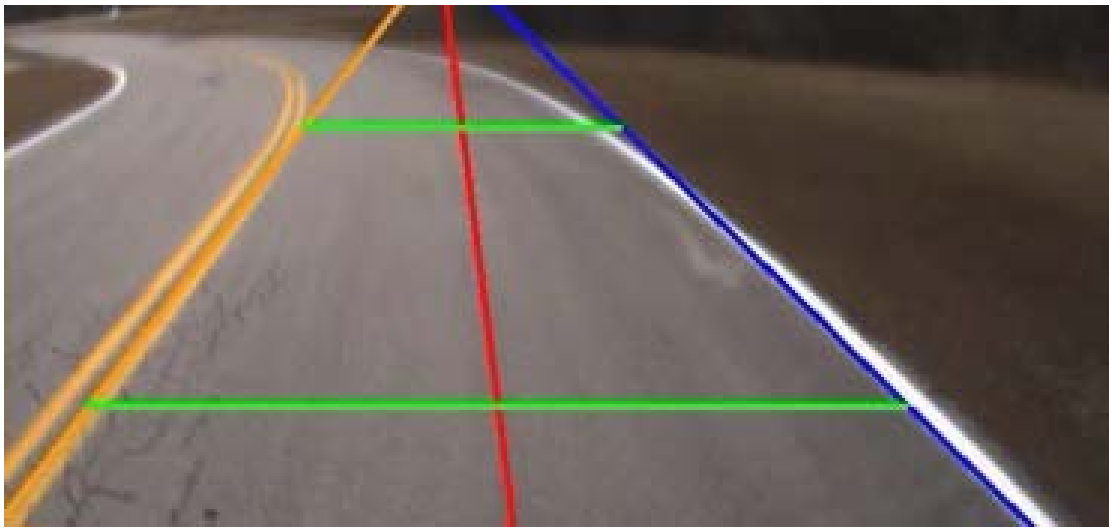


Figure 4-9. Center camera results of lane finding with estimated center line. A) Straight road, B) curved road.

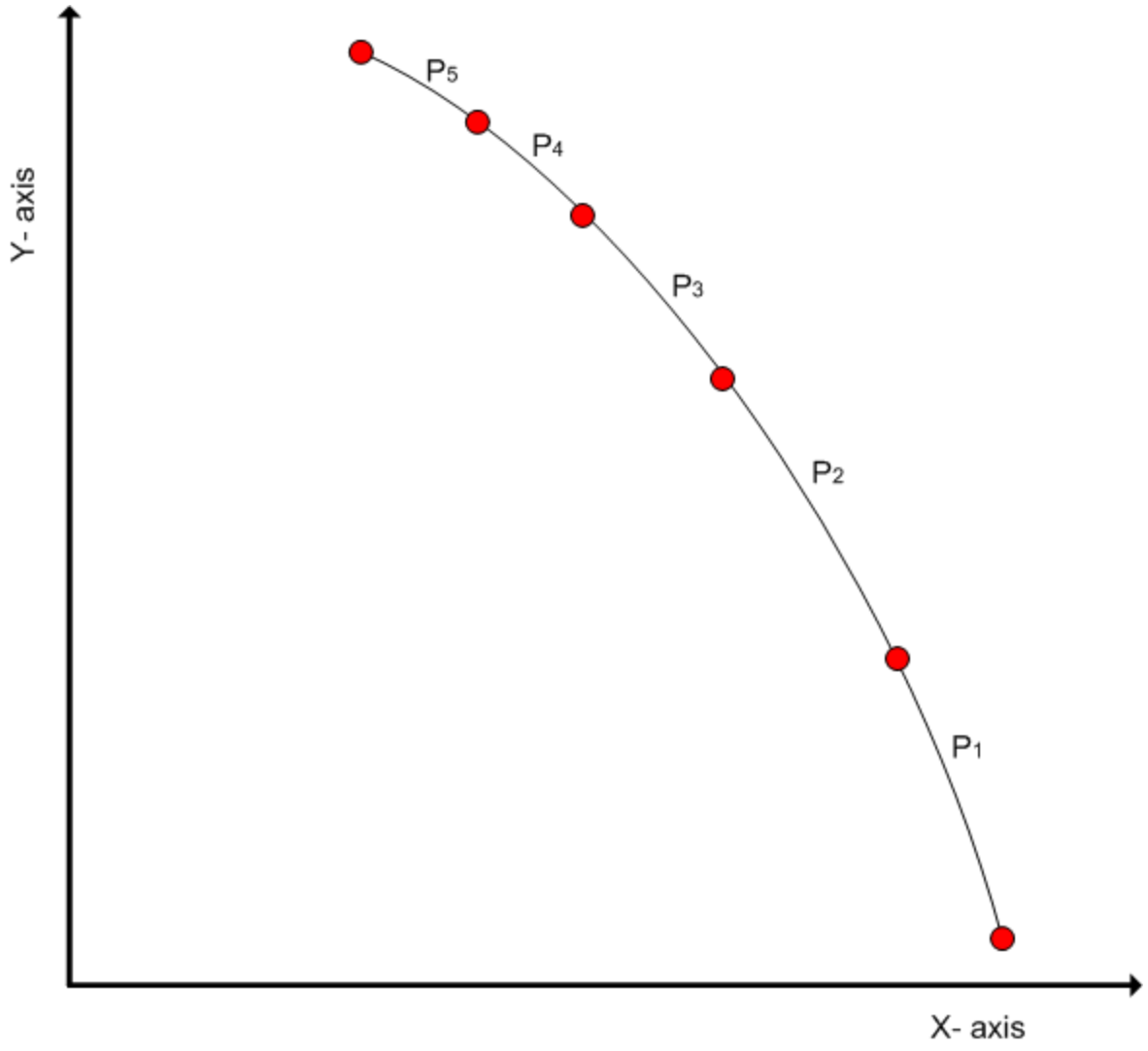


Figure 4-10. Diagram of Cubic spline and control points.

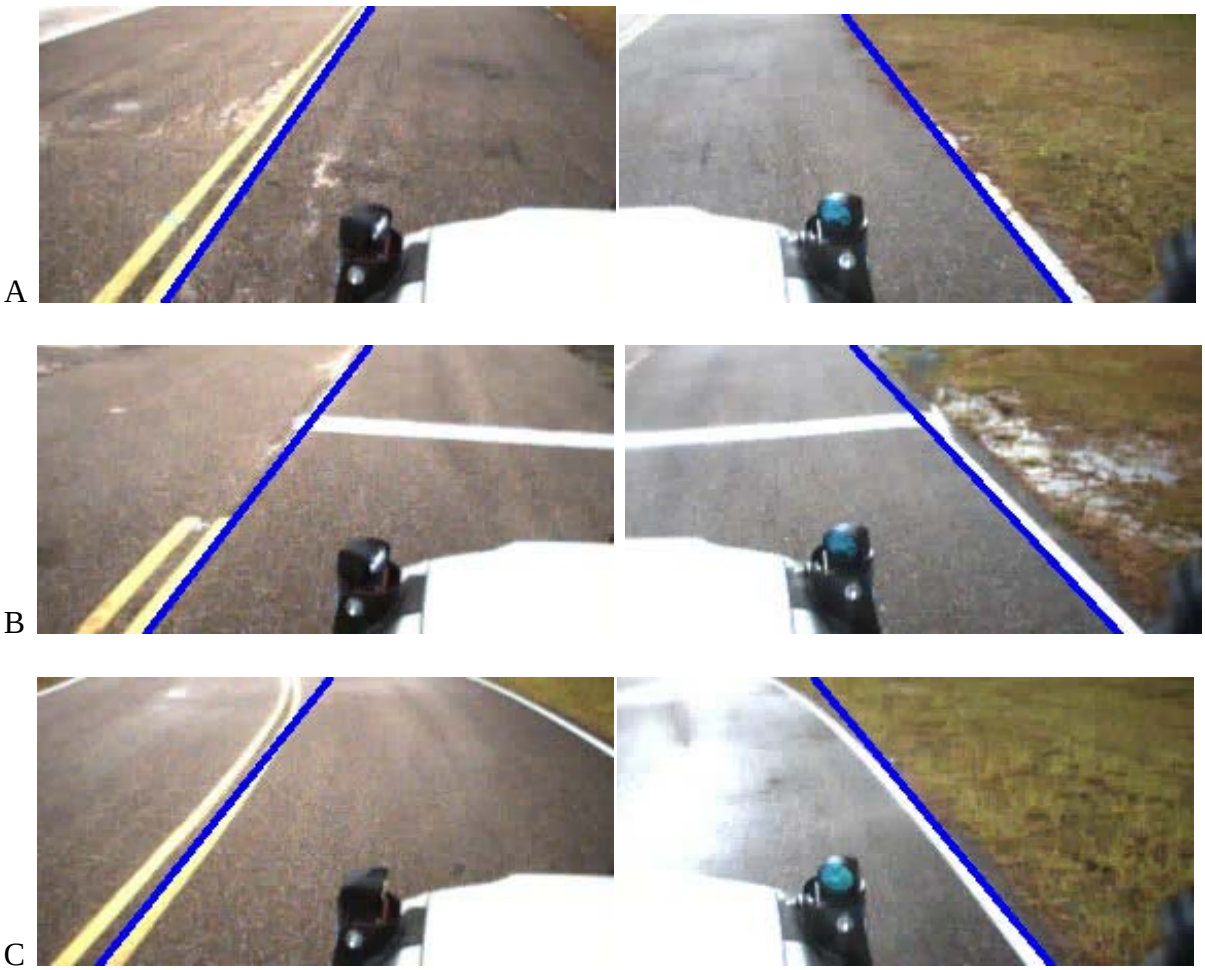


Figure 4-11. Two camera overlay image of lane lines in curved road. A) Straight road, B) crossroad with stop line, and C) curved road.

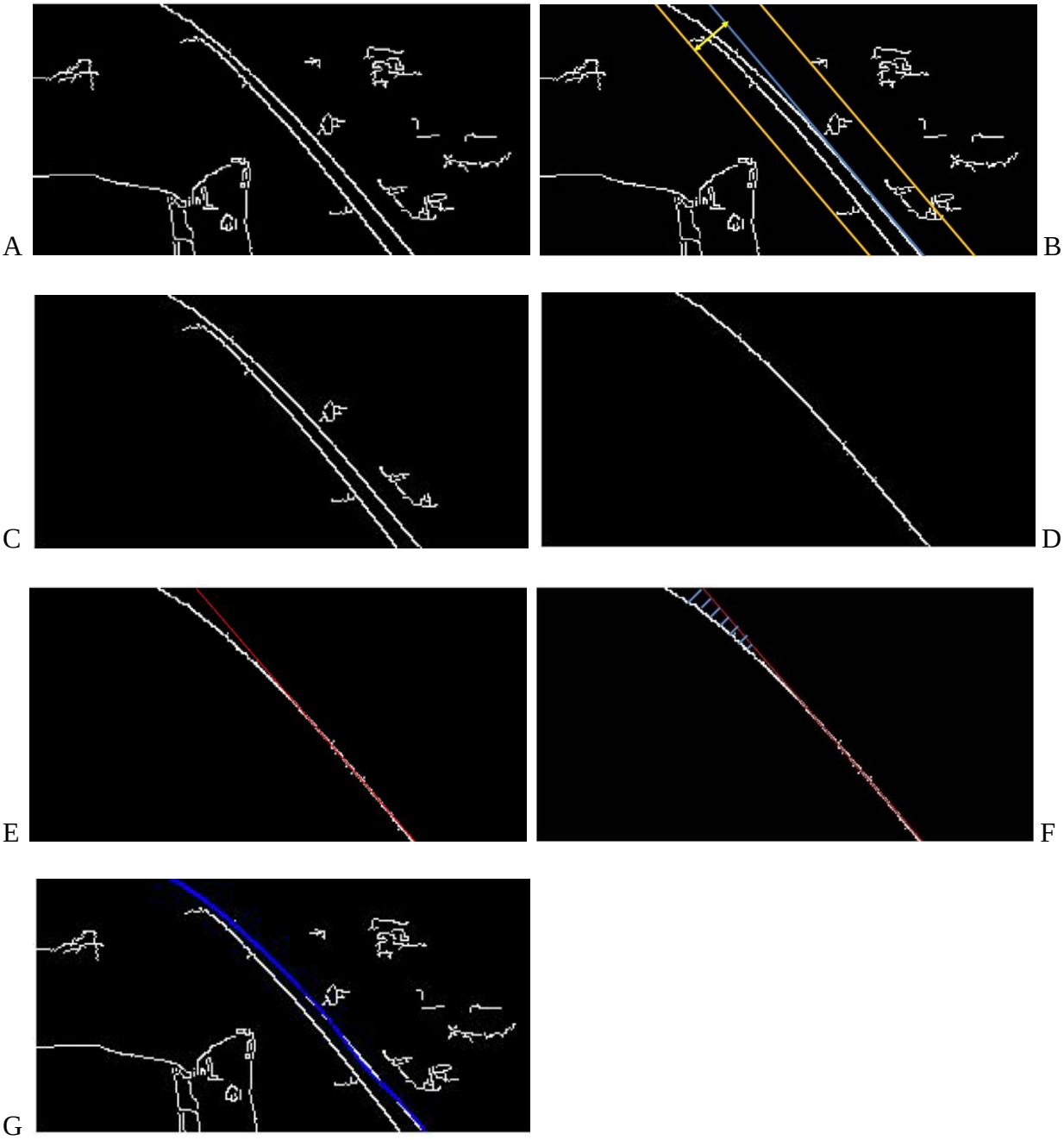
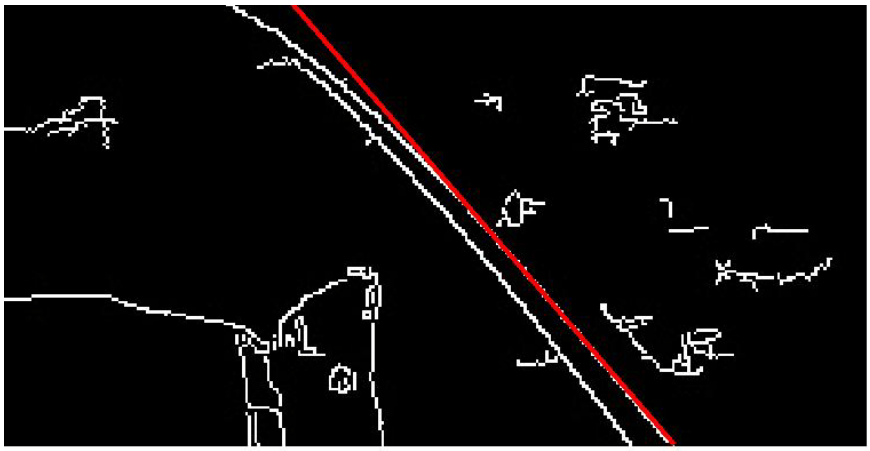


Figure 4-12. Hough line + curve control points for spline model.

A

B



C

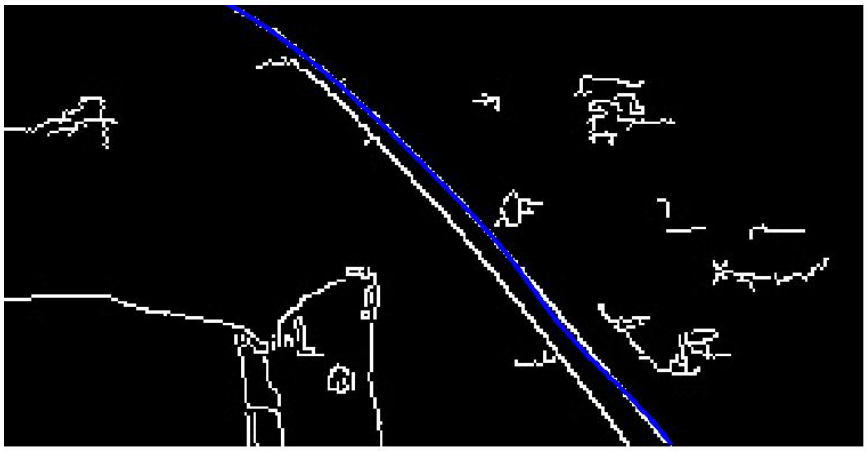


Figure 4-13. Curved lane. A) Source image, B) straight line by the Hough transform, and C) curved line by Cubic spline.

A

B

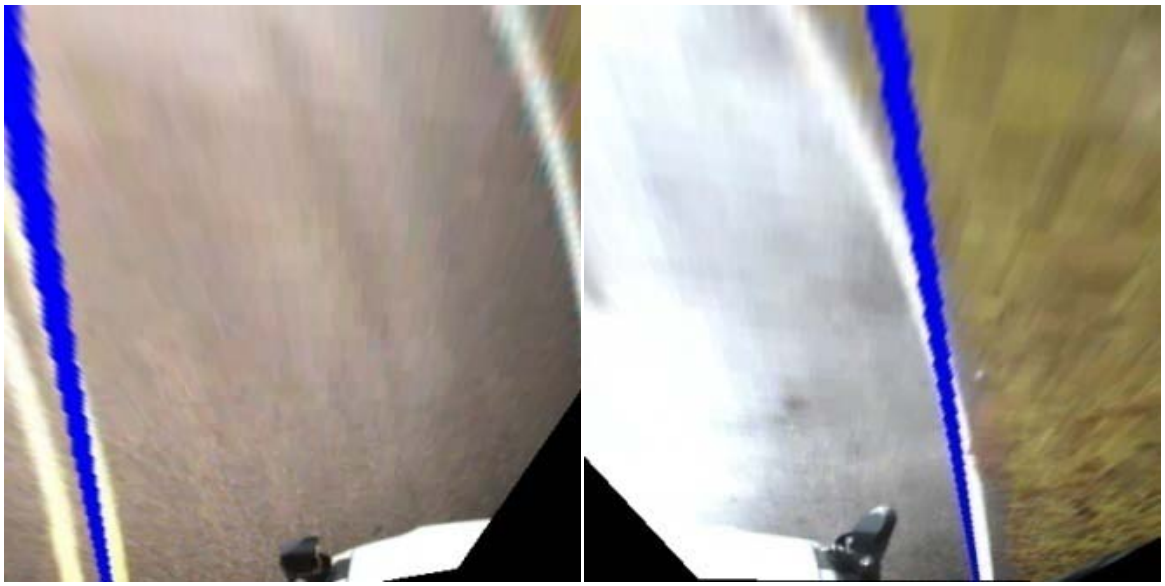


Figure 4-14. Lane model checking view. A) Straight line, B) curved line.

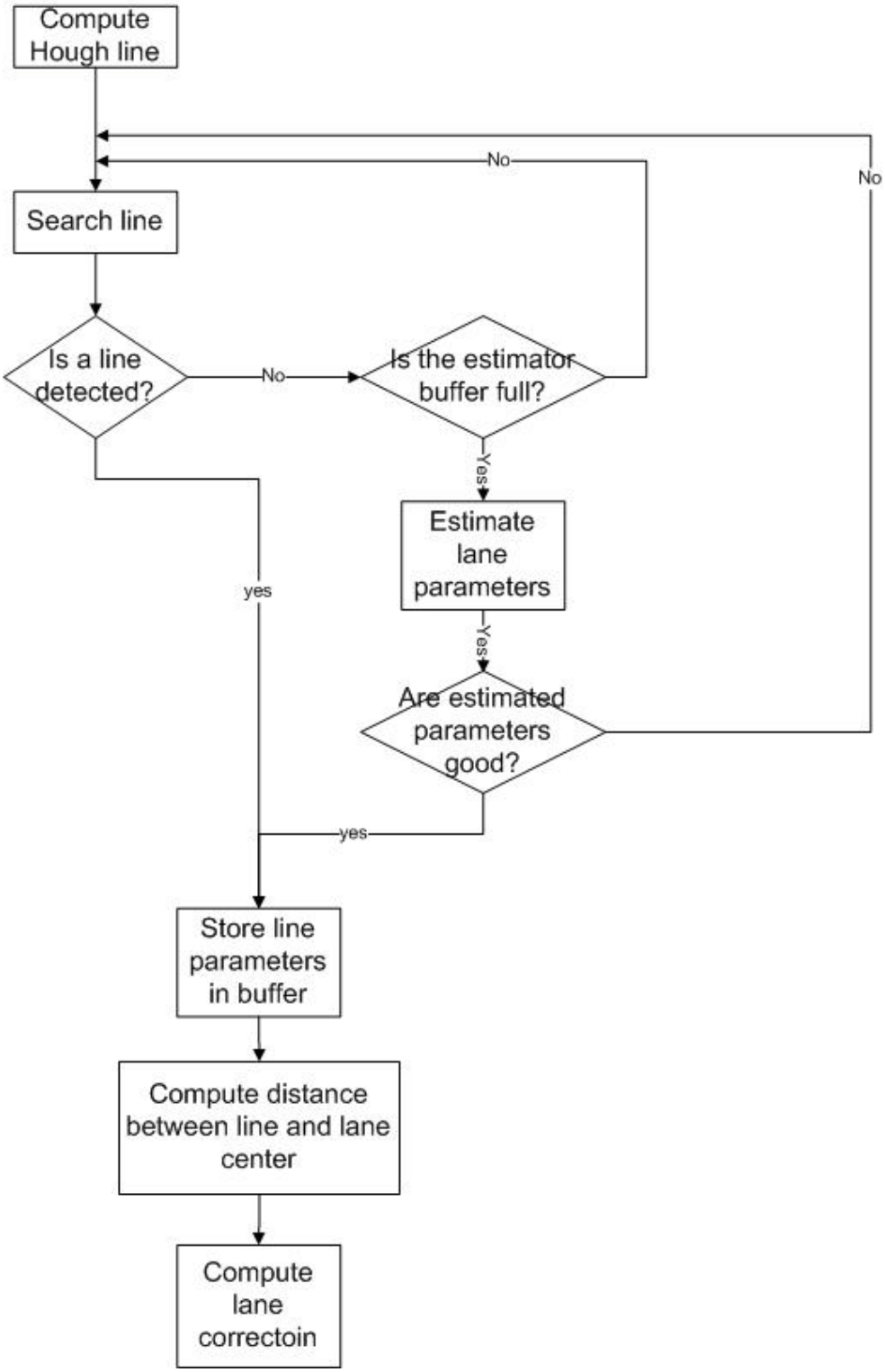


Figure 4-15. Line parameter estimation flowchart.

A   B

C   D

Figure 4-16. Two sequence source image and detected (blue) and estimated (orange) line.

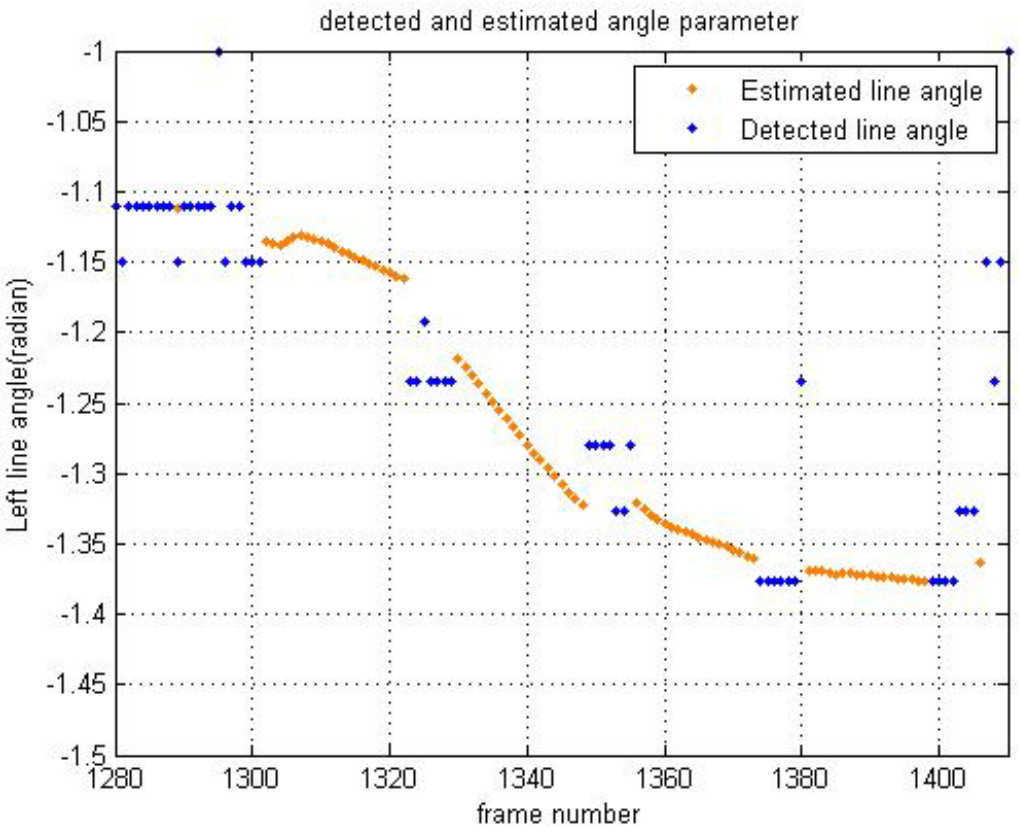


Figure 4-17. Least squares angle parameter estimation result.

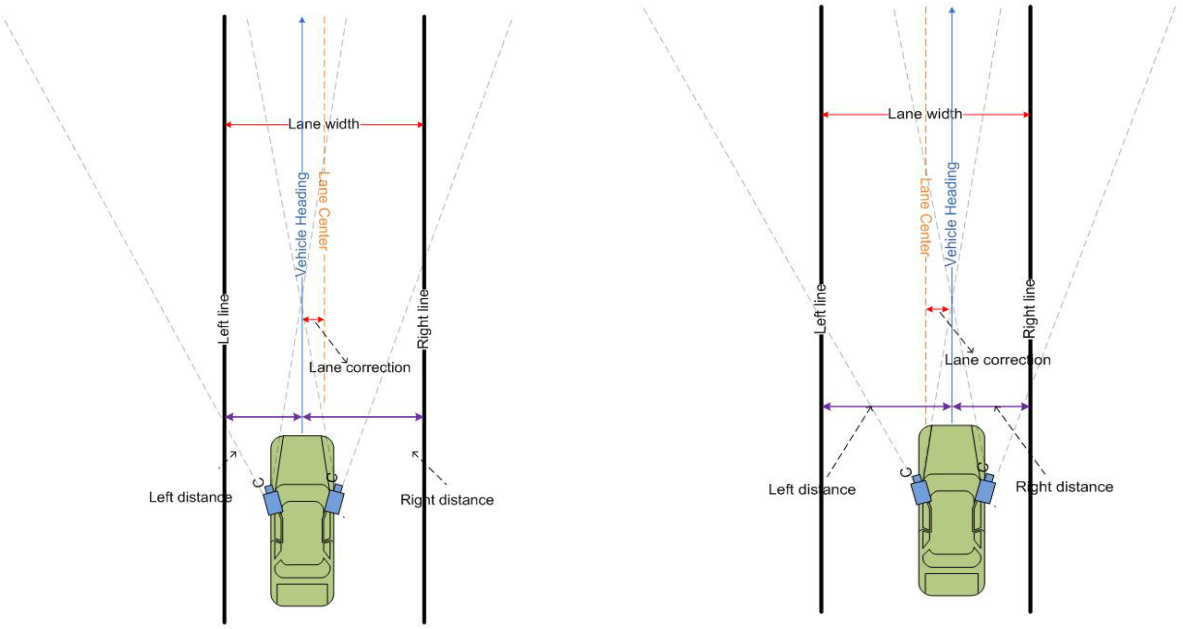
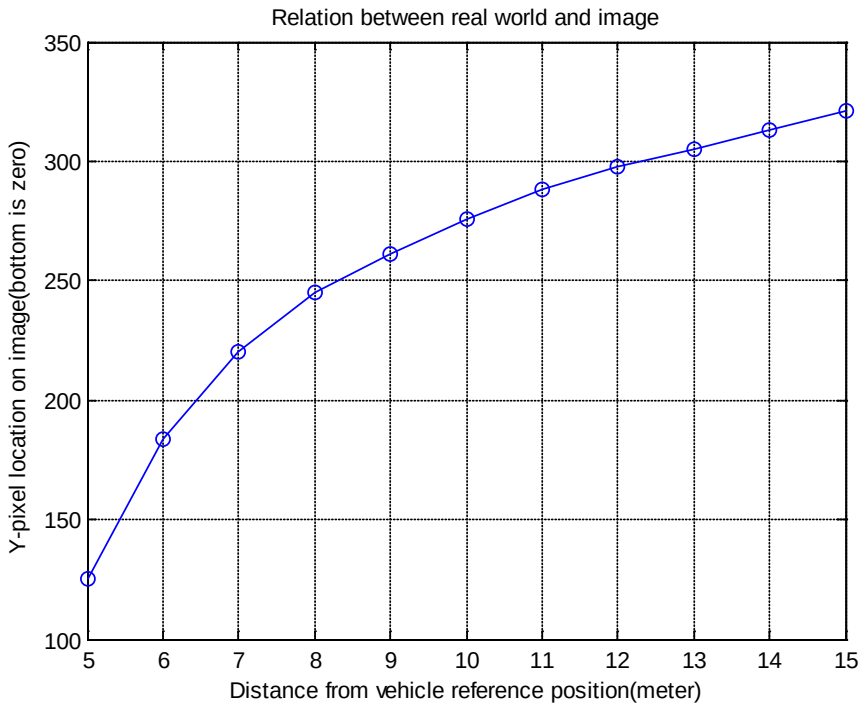


Figure 4-18. Lane correction distance definition. A) When a vehicle drives on the left side of the lane, B) when a vehicle drives on the right side of the lane.

A   B

C   D



E

Figure 4-19. Real world and image distance relationship. A) LFSSWing calibration images, B) Relation between real world position and image pixel on Y axis.

Figure 4-20. The LFSS and PFSS calibration image.

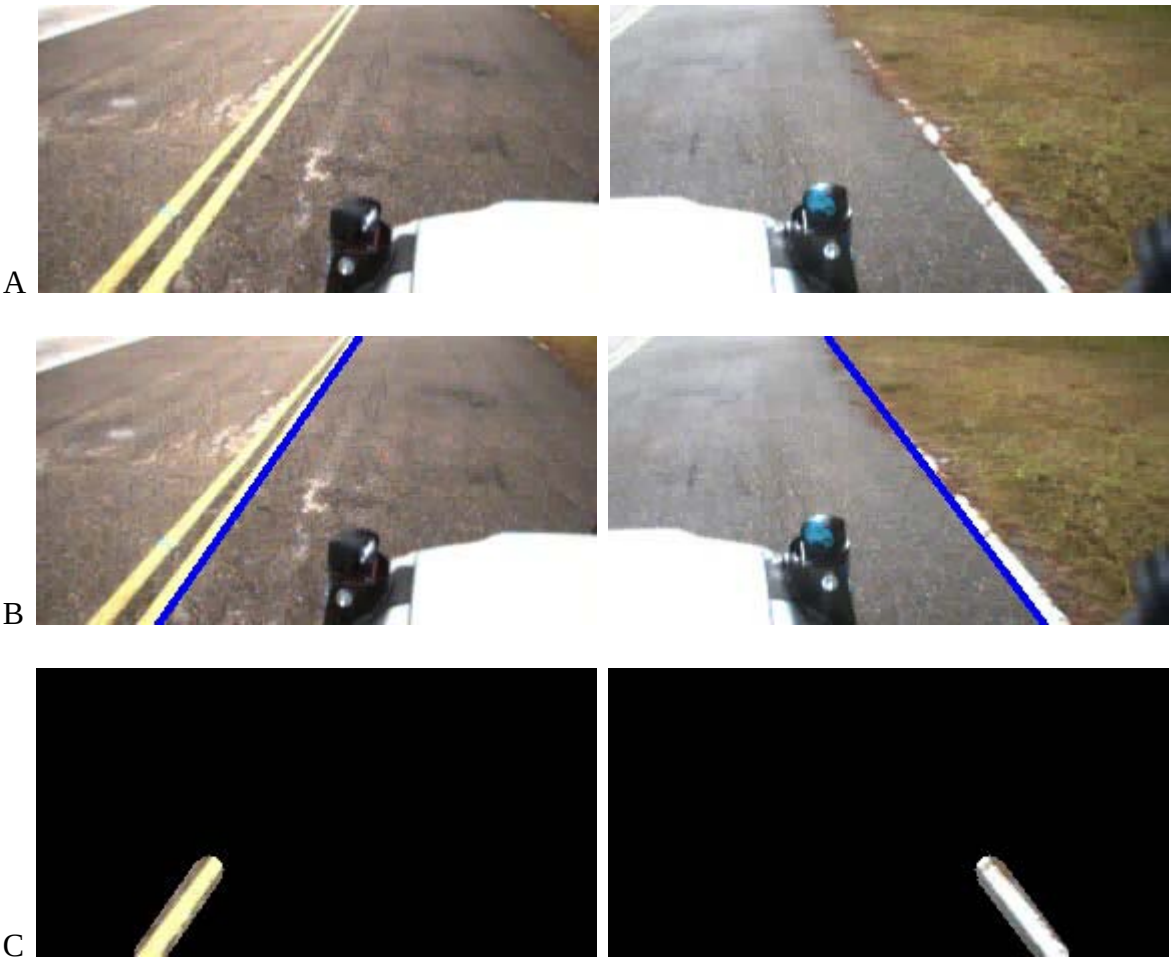


Figure 4-20. Mask images for lane line color. A) Source image, B) Hough transform based line image, and C) mask image.

CHAPTER 5
VECTOR-BASED GROUND AND OBJECT REPRESENTATION

Introduction

One of the biggest issues in computer vision systems is the large sizes of the source and result data. These properties can result in out-sized computation processing, bandwidth requirement problems, communication delays, and storage issues for an autonomous robot vehicle, especially systems consisting of multiple components. A recently developed technique, using computer hardware like general-purpose computing on graphic processing units (GPGPU), helps to reduce processing time remarkably, but problems still remain. For example, if the robot system uses a raster-based world representation and sensor data fusion, it requires great storage space, long computation time, and a large communication bandwidth. It will support only the specific resolution that is initially defined. However, a vector-based world representation permits storing, searching, and analyzing certain types of objects using a small storage space. It can also represent multiple resolution world models that help to improve processing, analyzing, and displaying the data. In addition, a vector-based system can store much property information in addition to object information. With these vector-based sensor data representation advantages, one can store previous traveling information in a database system similar to human memory. Therefore, one can use previous travel data to verify current travel safety.

Approach

With respect to computation and data storage efficiency, the vector-based representation of a ground plane is better suited for real-time robot system components. A vector-based representation can generate both 2-D and 3-D object models with the help of a 3-D sensor like GPS and LADAR. Specifically, a raster-based world model's memory size is highly dependent

on grid map coverage and resolution. If a world model covers a wide area and is utilized with high resolution, it needs large computation power and a very large memory space and it causes bandwidth problem between components. Another issue with a raster-based grid map is that it contains many areas of unknown data, because the map's coverage area is fixed. Figure 5-1 (C, D) shows two different size grid maps from Figure 5-1 (A, B), the source and classified image, respectively. Figure 5-1(C) is a 241 × 241 size, 0.25 meter resolution grid map, with a data size 56k. And figure 5-1 (D) is a 121×121 size, 0.5 meter resolution grid map, with a data size 14k. Therefore, raster-based grid map data size depends mostly on the resolution and coverage area. Table 5-1 summarizes data size of a various format traversability map.

Table 5-1. Raster-based traversability grid map data size

Coverage (meter)	Grid map size (pixel)	Grid map resolution (meter)	Data size
60 × 60	121 × 121	0.5	14K byte
60 × 60	241 × 241	0.25	56 K byte
60 × 60	601 × 601	0.1	352 K byte
300 × 300	601 × 601	0.5	352 K byte
300 × 300	1201 × 1201	0.25	1408 k byte
300 × 300	3001 × 3001	0.1	8794 K byte

Unlike this raster-based world representation, a vector-based method has almost no limitations to coverage and resolution. A vector-based coverage only depends on sensor coverage and sensor resolution. It also needs dramatically less memory for storing, sending, and receiving data, and easily builds a multi-resolution world environment. For example, the PFSS's vector output needs around 4 to 20 points to represent traversable road area in an urban environment. Table 5-2 summarizes a vector-based traversability data size based on the number

of vector points. Compared the raster-based ground representation to table 5-1, the vector-based representation can reduce data size at least 60 times on a 60 × 60 grid map.

Table 5-2. Vector-based traversability data size

Points	2-D map	3-D map
4	2 × 4 x 4 = 32 byte	3 × 4 x 4 = 38 byte
12	2 × 12 x 4 = 96 byte	3 × 12 x 4 = 144 byte
20	2 × 20 x 4 = 160 byte	3 × 20 x 4 = 240 byte

Ground Surface Representation

In the PFSS, the Triangulated Irregular Network (TIN) [Witzgall 2004] is selected to represent a world surface. Fewer points are needed to represent the same sized ground surface than with raster-based representations and it is a digital data structure used for the representation of a ground surface in geographical information systems (GIS). Figure 5-2 shows a sample TIN using LADAR data.

From a road and non-road segmented image as in Figure 3-10 (B, C, D), a high resolution bird's-eye view image is generated, as shown in Figure 5-3 (A), which is applied with a 10 centimeter per pixel resolution. Unlike the grid map image, the sight of the bird's-eye view image covers only the front area of a vehicle to reduce useless information. Because of this property, a bird's-eye view image can be generated with 10 centimeter per pixel resolution. Next, the road boundary and candidate control points are extracted. Figure 5-3 (B) and (C) show boundary image and candidate control point's image, respectively. Finally, fewer candidates' points are selected for consideration of vector resolution, storage efficiency, and accuracy.

For example, if a vehicle drives in a straight line, the road looks like a rectangle. Therefore, fewer points are selected for representing the road. However, if a vehicle drives over a curved

road, the road shape looks like a curvilinear polygon. Consequently, many more points are required to represent a curved road. These selected points are new candidate points for a TIN ground representation. Figure 5-4 shows irregular vector points in various situations, for instance, irregular vector points for a straight road, curved road and T-intersection case, respectively.

Figure 5-5 (A, B, C, D, and E) summarizes how to extract a boundary of traversable area and select candidate points for TIN representation.

- From a bird's-eye view image, noise pixels are eliminated, as shown in Figure 5-5 (B).
- Extracting the road boundary is shown in Figure 5-5 (C).
- Selected TIN control points are shown in Figure 5-5 (D).
- A TIN map is shown in Figure 5-5 (E).

Figure 5-5 (F) shows a 3-D vector-based ground representation with zero height using a TIN algorithm. After selecting the TIN control points of a traversable ground boundary in figure 5-3 (C), those points are used to build a road model and are stored. Since center camera's vertical field of view starts from the 8 meter for the reference position, ranges of stored points are 10 meters to 20 meters from the vehicle reference position. The figure 5-6 illustrates diagram of road boundary polygon area and stored points.

Static Object Representation

There are two main goals for the LFSS and the LFSSWing component software. First is a future trajectory estimation and lane departure correction. Second is building and generating a lane model and lane properties by using GPS data. To solve these two problems, the LFSS and the LFSSWing detect static road objects that are road lane lines, and directly compute the mathematical lane center as an output. However, the LFSS and the LFSSWing cannot detect both

lane lines when a vehicle travels in an area with segmented lines. Therefore, in this case, an estimated line using the parameter estimator is stored using previously detected line information.

After detecting and computing the lane center and its properties, vector points of the road lane are selected to build and store the lane model. Two points from the vehicle reference points are selected at each side lane line. For example, 5 meter and 10 meter lane line points from the vehicle reference, which is located in inertial measurement unit (IMU) are selected, since the point area image resolution is high enough. Figure 5-7 shows a diagram of lane polygon area. These four points create a polygon for notifying and storing traversable area to the World Model Vector Knowledge Store. From these stored lane polygon data, vector-based lane objects can be illustrated.

World Model Vector Knowledge Store

The generated ground surface and lane object data will be stored in a database system that is called the World Model Vector Knowledge Store (WMVKS). The WMVKS can store and retrieve ground surface characteristics, static and moving object, and moving objects images from various type sensors. For the vision-based components, the PFSS sends ground surface polygon points and the LFSSWing sends lane object polygon points to the WMVKS. The WMVKS enables an autonomous vehicle to drive the same place it has already traveled using archived ground surface and ground surface static object information. Tables 5-3 and 5-4 show the LFSS lane object DB table and the PFSS ground surface DB table.

Table 5-3. The LFSS Lane object DB table.

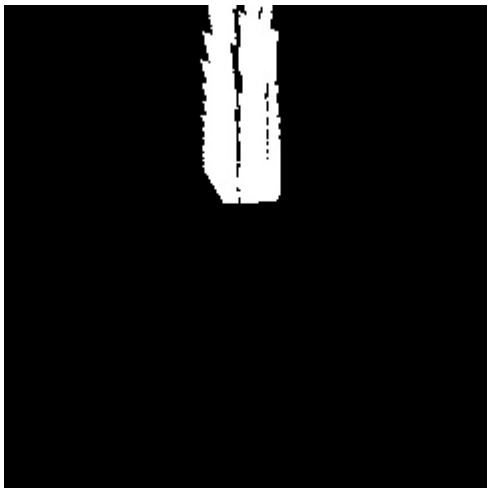
Lane	Lane confidence	Lane property	Lane width	Width confidence	Date/Time	Run time	Component ID
Double	Integer	Integer	double	Integer	YYYY/MM/DD HH/MM/SS	Integer	Integer
{P1,P2,P3,P4} P1= {Latitude, Longitude, Altitude}	0-1	1-White 2-Yellow 10-Solid 20-Segmented	Meter	1 - high confidence 0- low confidence		1,2,…	21-LFSSWing 22-LFSS
Ex)P1= {29.756850064, -82.267883420, 0}	Ex) 1 - high confidence 0- low confidence	Ex) 11-white solid line	Ex) 4.3	Ex) 0.9	2009/10/19 /13/23/01		Ex) 21

Table 5-4. The PFSS ground surface DB table.

Surface ID	Polygon	Surface property	Date/Time	Run time	Component ID
Integer	LLA points (double, double, double)	Integer	YYYY/MM/DD HH/MM/SS	Integer	Integer
1,2,3 …	{Pt1, pt2, … ptN, Pt1}	1-Asphalt 2-Unstructured road 3-Grass 4-Unknown		1,2,…	21-Ladar TSS 22-Vision TSS
Ex)1	Ex) {(29.75692231169602, -82.2675184406604, 0), (29.75691959879387, -82.26765801719262, 0) … (29.75692231169602, -82.2675184406604, 0)}	Ex) 1	2009/10/19 /13/23/01		Ex) 22

A 

B 

C 

D 

Figure 5-1. Various grid map sizes. A) Source image at University of Florida campus, B) classified image, C) 241 × 241with 0.25 meter resolution grid map image, and D) 121 × 121 with 0.5 meter resolution grid map image.

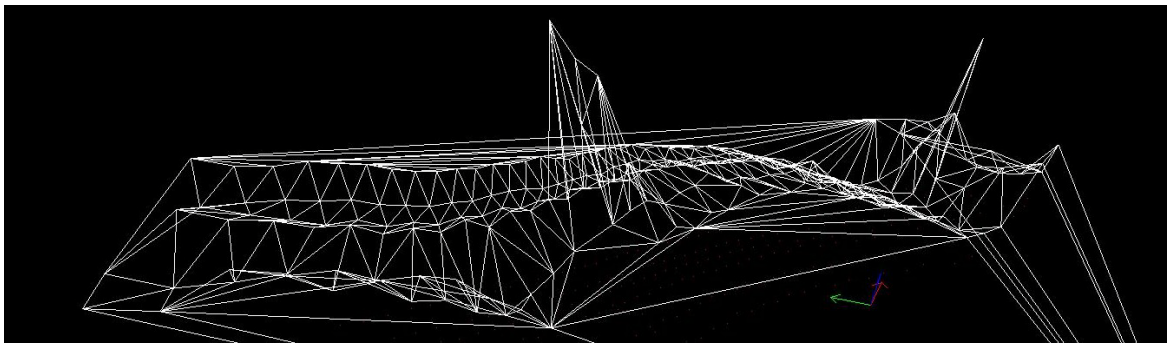
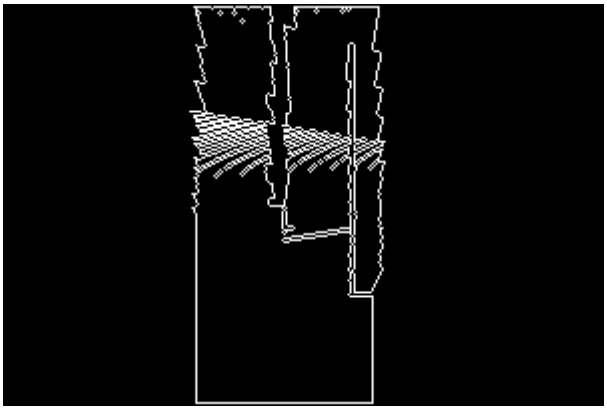


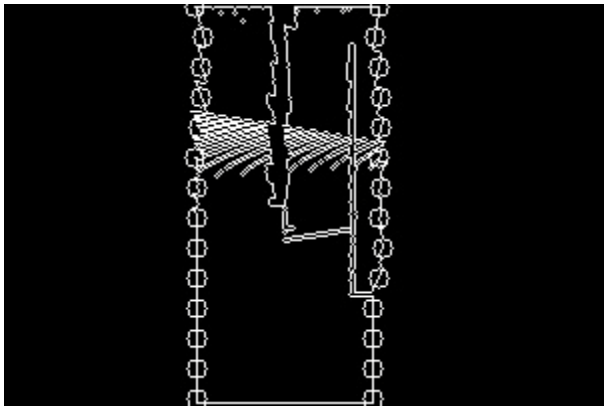
Figure 5-2. The triangulation map[1]

[1] This TIN image is a LADAR–based ground surface image by Jihyun Yoon.

A

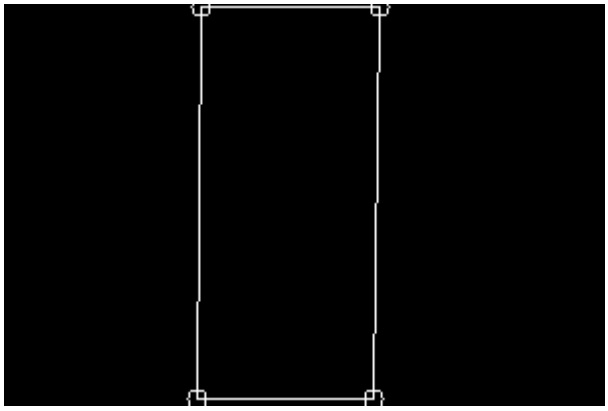


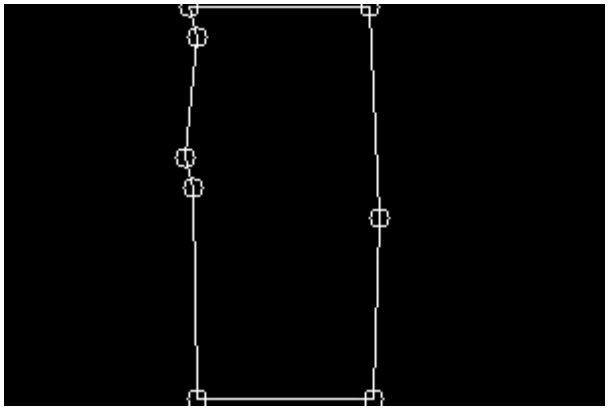
c B

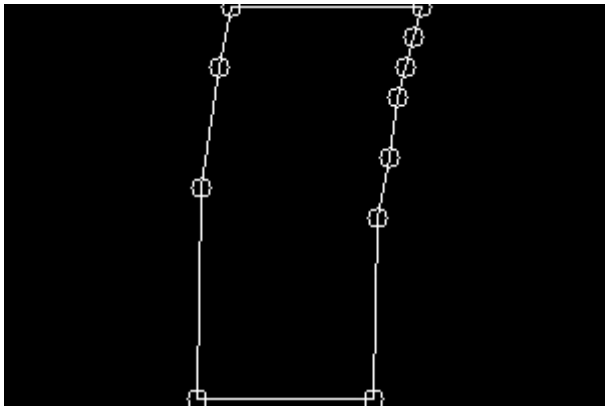


C

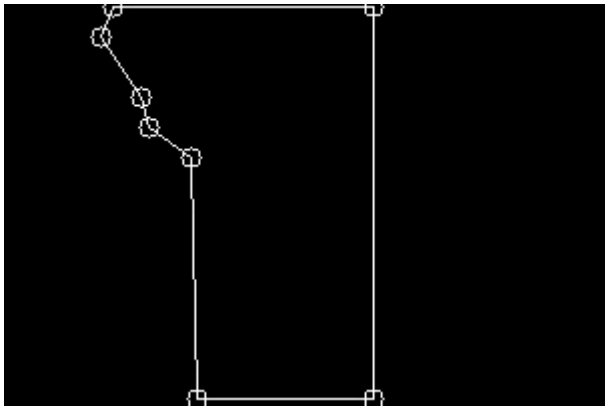
Figure 5-3. Vector representations. A) 10 centimeter resolution bird's-eye view image, B) boundary Image, and C) boundary image with control points.

A  

B  

C  

D



F

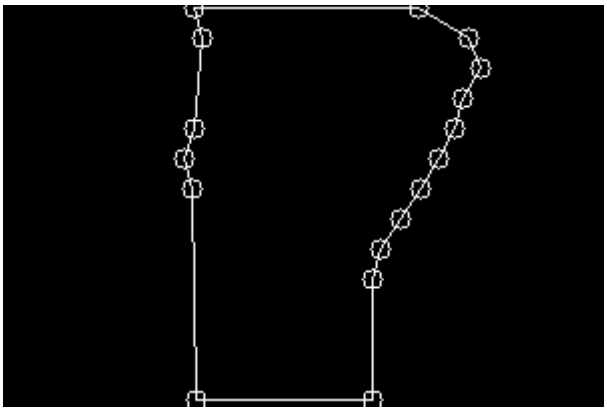


Figure 5-4. Irregular vector points. (A, B), straight road, (C) curved road, and (D, E) T-intersection road.

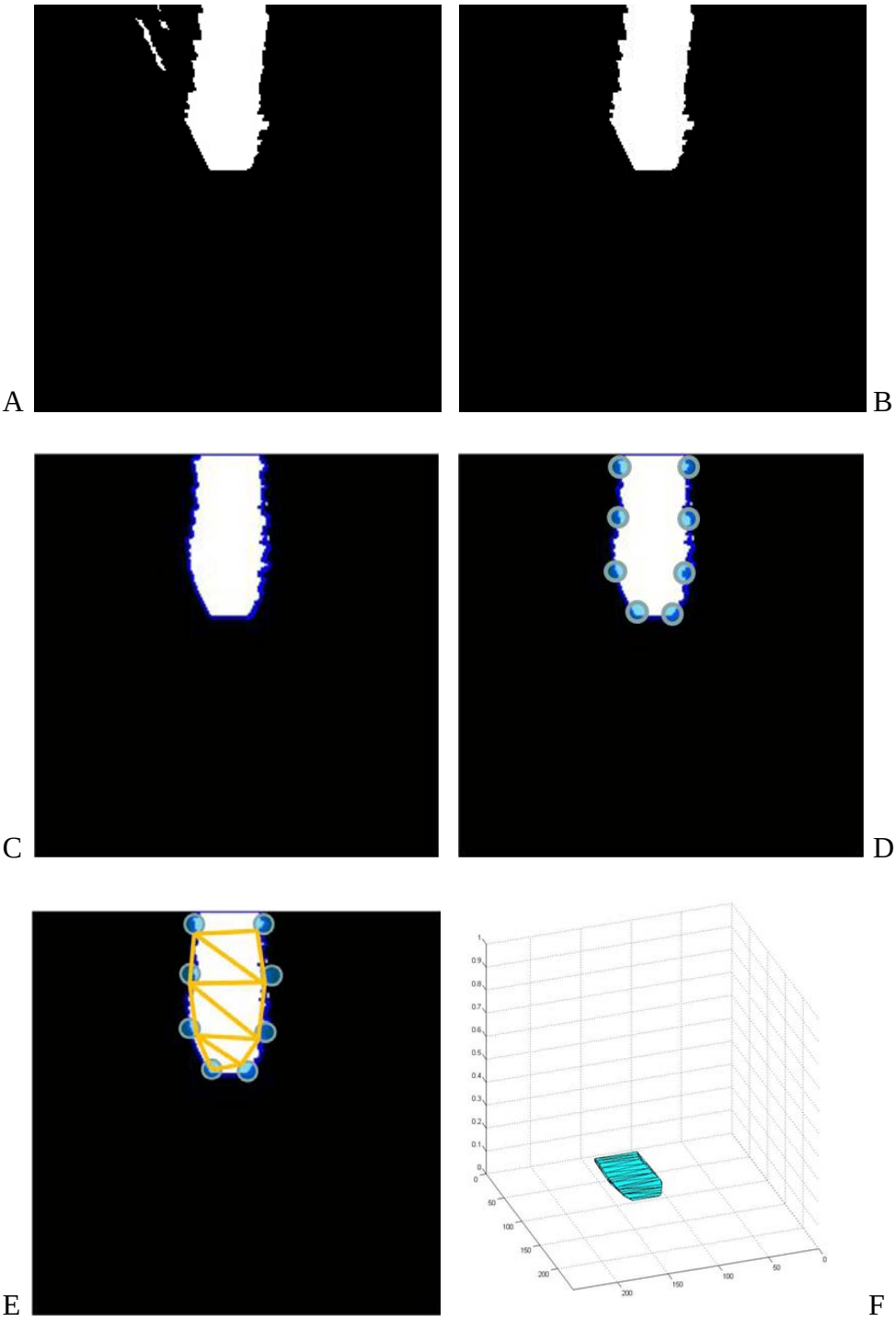


Figure 5-5. The TIN representation of traversability map. A) Bird's-eye view image, B) noise illumination, C) road boundary extraction, D) control point selection, E) 2-D TIN map, and F) 3-D TIN map.

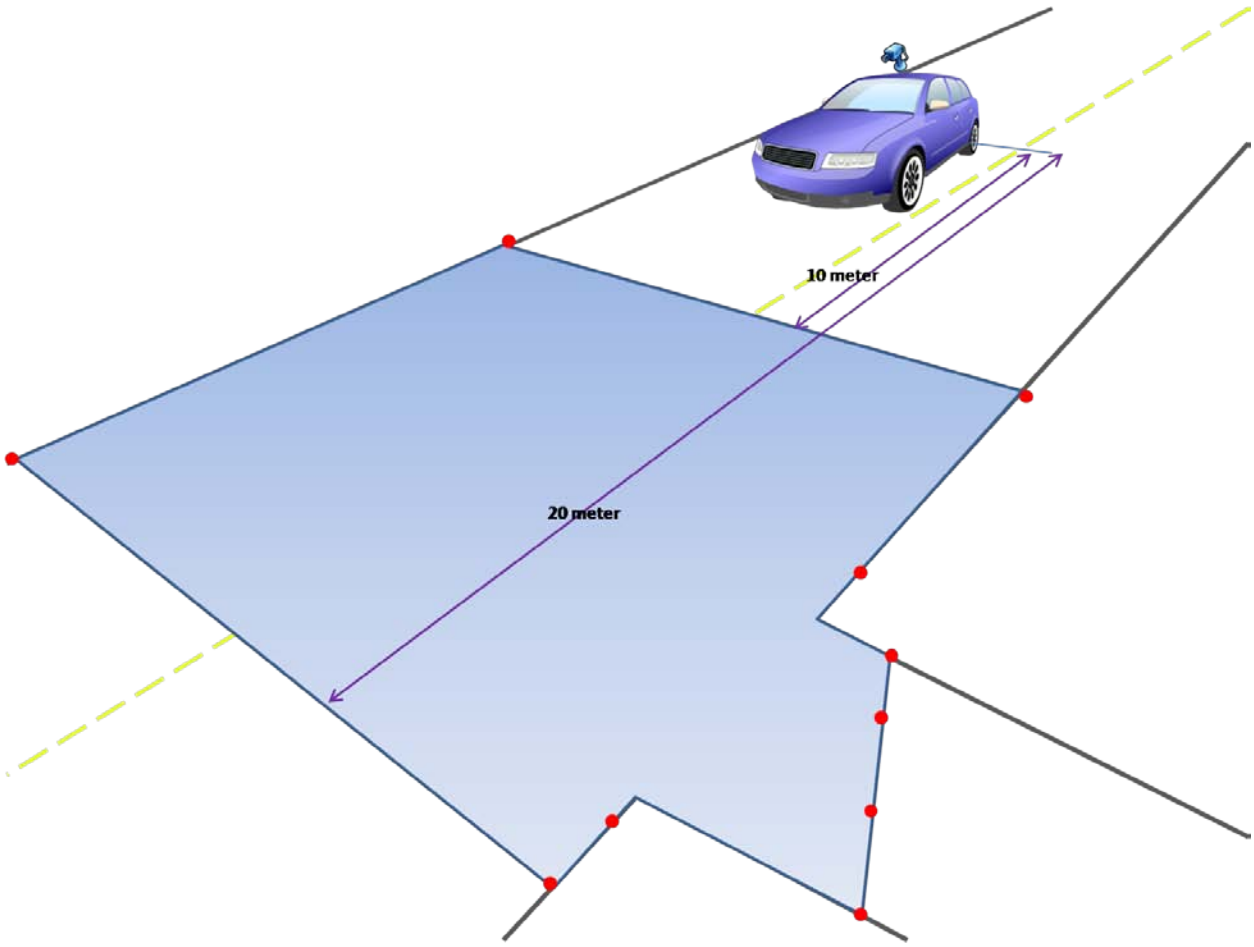


Figure 5-6. Road boundary polygon and stored polygon points (red points).

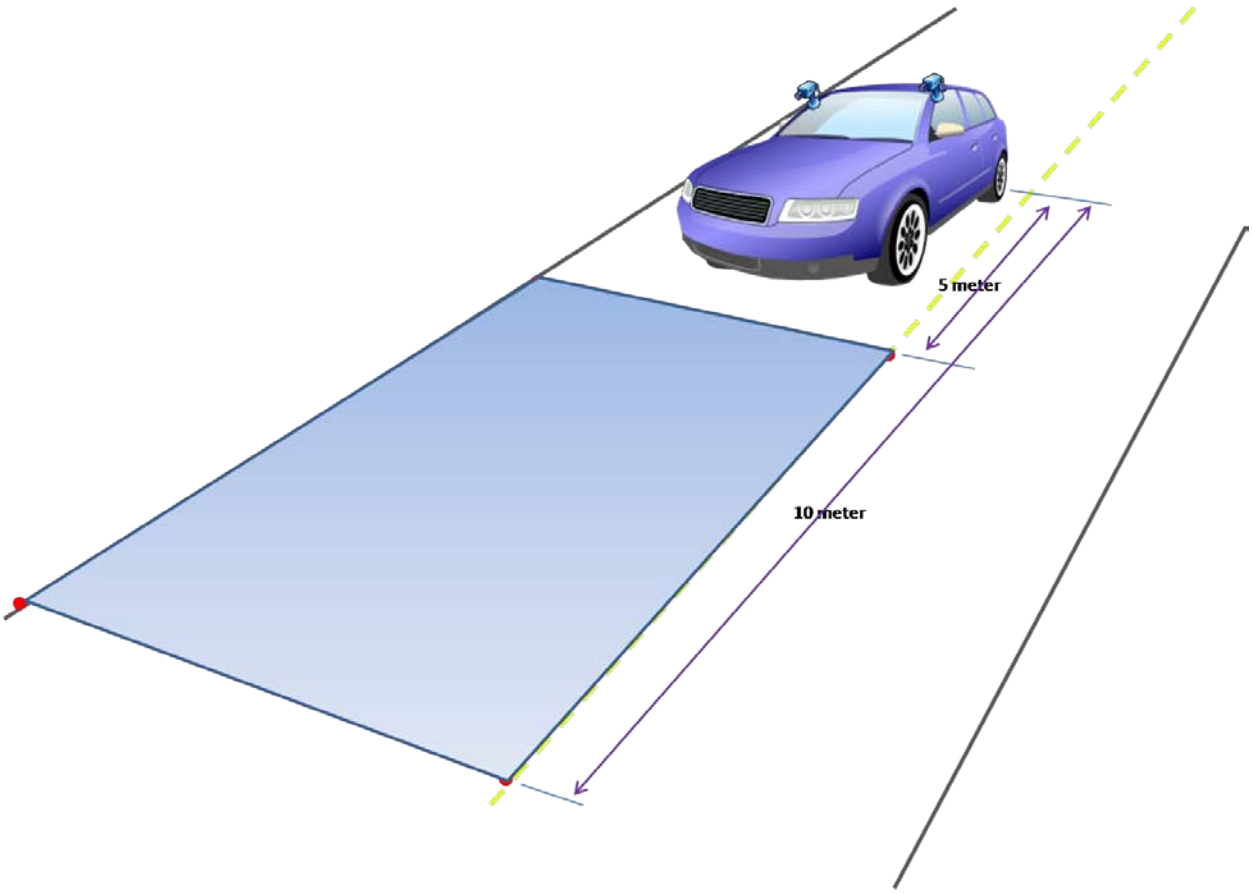


Figure 5-7. Lane objects and stored polygon points (red points).

CHAPTER 6
EXPERIMENTAL RESULTS AND CONCLUSIONS

Platform

The target implementation for this research is the University of Florida DARPA Urban Challenge 2007 team vehicle, called the Urban NaviGator. It is a modified 2006 Toyota Highlander Hybrid SUV equipped with numerous laser measurement systems, cameras, a differential GPS, and an inertial measurement unit (IMU). A total of 8 Linux computers and 3 Windows computers are located in the vehicle to process, compute, and control the vehicle. Figure 6-1 shows the Urban NaviGator vehicle and sensor mount.

Hardware

The Urban NaviGator has one BlueFox high-speed USB 2.0 camera [Matrix Vision 2009] for the PFSS/LFSS and two BlueFox high-speed USB 2.0 cameras for the LFSSWing. The PFSS and LFSS share the source images from the camera mounted in the center and the LFSSWing uses the two side cameras. The wing mounted cameras and the center camera use different focal length lenses and have different fields of view; therefore, they generate lane correction information at different distances and will increase the certainty of the overall prediction of the future trajectory.

The BlueFox USB 2.0 camera provides up to 100 Hz color frame grabbing rates with 640 × 480 resolution. Because the vision components' areas of interest are smaller than the camera's field of view, the source image resolutions are reduced to 640 × 216 pixels for the LFSS, 320 × 108 pixels for the PFSS and 320 × 240 pixels for both cameras used by the LFSSWing. The PFSS/LFSS camera was mounted in the middle of the sensor bridge and faces the ground, and the LFSSWing cameras were mounted on either side of the sensor bridge. Figure 6-2 (A) shows

the PFSS/LFSS and the LFSSWing cameras mounting locations. The center camera uses the TAMRON varifocal lens with 4-12 mm focal length and the two side cameras use 4 mm fixed wide focal lenses. Table 6-1 summarizes the camera and lens specifications and Figure 4-1 shows the PFSS/LFSS and LFSSWing cameras' angles of view, which depends on lens specification and the orientation.

Each vision component uses an AMD dual core computer with 1 GB of RAM. At the DARPA Urban Challenge 2007 competition, the PFSS could run at 15-18 Hz update rates and the LFSSWing could operates with 10-17 Hz update rates with this hardware.

Table 6-1. Camera and lens specification.

	PFSS	LFSS	LFSSWing
Location	Center	Center	Each side of sensor bridge
Source image size	320 × 108	640 × 216	320 × 190
CCD size	1/3"	1/3"	1/3"
Lens	TAMRON varifocal lens	TAMRON varifocal lens	COMPUTAR fixed focal lens
Focal lens	4-12 mm	4-12 mm	4mm
Horizontal angle of view	31.2° – 93.7°	31.2° – 93.7°	63.9°
Vertical angle of view	23.4° - 68.9°	23.4° - 68.9°	49.1°

Software

The PFSS, the LFSSWing, and the LFSS software programs were written in C++ for the Windows environment. Additional functions, algorithms, and GUI are constructed using the Matrix-Vision API library, and the OpenCV library. Also, the Posix thread library was utilized to quickly capture the source images from cameras. Both components support the Joint

Architecture for Unmanned Systems (JAUS) functionality [JAUS 2009]. JAUS is a communication protocol that serves to provide a high level of interoperability between various hardware and software components for unmanned systems. The PFSS, the LFSSWing, and the LFSS outputs are processed to JAUS messages and sent to the other customer components, for example, the LFSS Arbiter [Osteen 2008].

Each component provides various intermediate processing results that can be helpful in running the software parameters or for troubleshooting. For example, the PFSS component program can display two of the following images: the source image, canny-filtered image, source image without lane lines, noise-filtered image, or training area image. For the PFSS output, a raster-based grid map, raster-based grid map without yaw adjustment, road boundary points, or road polygon image can be selected by the user. Figure 6-3 shows various screenshots of the PFSS software.

For the LFSSWing and the LFSS, the source image, edge filtered image, Hough line image, or detected lane line overlay image can be selected. An information window displays each distance lane center corrections, lane color, and lane width values along with their associated confidence values. Figure 6-4 shows screenshots of the LFSSWing implementation (the LFSS software is similar).

The LFSS Arbiter component fuses local roadway data from the TSS, the LFSS, the LFSSWing, and the Curb Finder smart sensor component [Osteen 2008]. The data consist of offsets from the centerline of the vehicle to the center of the road lane estimated at varying distances ahead of the vehicle. Finally, the LFSS Arbiter generates a curve fit from the different sensor data. These data are used to adjust for GPS measurement errors that are supplied to the Roadway Navigation (RN) [Galluzzo 2006] component, which navigates the vehicle within a

lane using an A* search algorithm. Figure 6-5 (A, B) shows the Gainesville Raceway test area from the LFSSWing cameras and from a vantage point off of the vehicle, respectively. Figure 6-5 (C) shows the LFSS Arbiter's curve fit screen when the vehicle drives along a curved road. Figure 6-5 (D) shows the Roadway Navigation component's screenshot of its path searching. Brown represents the A* search candidate branches and white points are the intermediate goal points provided from LFSS Arbiter.

Figure 6-6 shows the Urban NaviGator 2009 system architecture. The PFSS, the LFSSWing, and the World Model Vector Knowledge Store are highlighted. The PFSS stores a vectorized representation of the road in the WMVKS. The LFSSWing also stores its output as a vector area that describe lane.

Results

The following section describes the test results pertaining to this research for both the LFSSWing and the PFSS components. Since the LFSS long range component's output is very similar to the LFSSWing component output, the LFSS output is not described in this section.

Based on chapter 2 assumptions, test areas are flat, and the camera source images are clear enough to see the environment. The auto-exposure control option, which is provided by the Matrix-vision's camera setting software, helps to capture a clear source image from various illumination conditions.

Tests are divided into four categories:

- The Gainesville Raceway,
- The University of Florida campus,
- NW 13th street, and NE 53rd avenue, Gainesville, Florida, and
- Night time setting.

The Gainesville Raceway is the only the place to perform an autonomous drive due to a safety reasons. Since the Gainesville Raceway is a race track, it provides a wide open area and it does not have standard necessary facilities, such as curbs, pedestrian crossing marks, and so on. The University of Florida campus provides various urban environment facilities, such as bike lanes, curbs, pedestrian crossings, sidewalks, more than two lanes, merging or dividing lanes, and shadows from trees. NW 13th street and NE 53rd avenue are selected to test the software with traffic and/or at high speeds. The PFSS component was tested only at the Gainesville Raceway and the University of Florida campus. Finally, the LFSSWing components were tested at night time with illumination provided from the head lights.

For the autonomous driving tests, vehicle speed was approximately 10 mph. For the real world tests, vehicle travel speeds were from 20 mph to 60 mph (driven manually). Vector-based maps were built while traveling approximately 10-20 mph.

LFSSWing test results

The Gainesville Raceway is pictured in the Figure 6-7 (A). The outer loop is approximately 1 km and the smaller half loop is 650 meters. This course sequence includes a straight lane, a curved lane, segmented painted lane line, a narrow lane width area, T-intersection areas, and cross road areas. Since this location is an open area, cameras can receive different direction's light in a short time. Figure 6-7 (B) shows a straight lane at the starting point. Figure 6-7 (C) shows a curved lane, with the top part of the right camera source image washed-out due to the light direction. Figure 6-7 (D) shows that a short length of segmented line can be detected as a lane line. When the Hough candidate lines are extracted from an edge image (Figure 4-4), the line length threshold is decided by the source image height. In this research, 20% of source image height, 48 pixels, is selected as the Hough line minimum length parameter. Figure 6-7 (E)

shows a narrow lane compared to a wide lane area in the Gainesville Raceway. The lane width threshold is defined by a roadway design- plans preparation manual [Florida Department of Transportation 2009]. 12 feet (3.65 meter) is a standard rural lane width. In this research, a 2 foot margin is applied, therefore a 10 foot to 14 foot (3.048 meter to 4.267 meter) lane gap is considered to be a properly detected lane width. The lane width in Figure 6-7 (E) is approximately 3.35-3.5 meters and it is narrower than a standard lane width. Figure 6-7 (F) shows a T-intersection area. In this situation, the right lane line is detected and the left lane line is estimated using previous 20-frames line parameters. Figure 6-7 (G) shows a vehicle traveling on the right side line in an autonomous test run. Since vehicle controller's response is not always fast enough, it is a possibility that the vehicle ventures onto the line momentarily. However, since lane correction values are being updated continuously, the vehicle can drive back to the middle of a lane.

The second test place was the University of Florida campus. This location has many artificial structures, for example bike lanes, curbs, and pedestrian crosswalk and so on. The test course is an approximately 4 km loop. Figure 6-8 (A) shows a satellite photo of the University of Florida campus. In Figure 6-8 (B), the LFSSWing operates with shadows in the image. Figure 6-8 (C) shows a vehicle traveling through a pedestrian crossing area, and figure 6-8 (D) shows the LFSSWing detecting a curb.

The third test place was an urban area with real traffic. In this environment, sample results include high speed conditions, divided lane situations, and roads with more than two lanes. Figure 6-9 shows some urban road test results. Figure 6-9 (A) shows multiple center lane lines, figure 6-9 (B) shows a divided lane area, and figure 6-9 (C) shows the LFSSWing output with real traffic on a four-lane road.

The fourth test place was the University of Florida campus at night with a nighttime camera setting. Since illumination is too weak at nighttime, the camera exposure time was increased and the rest of the LFSSWing setting was the same as daytime. Not just lack of illumination, but also other artificial lighting sources by other traveling car or streetlight are the big difference of this test. Figure 6-10 shows various situations outputs in a night time test.

The PFSS test result

The PFSS was tested at the Gainesville Raceway and the University of Florida campus. Since the PFSS is designed to characterize ground surface area, an urban environment is not suitable to get meaningful output. For example, if another vehicle travels in the camera's field of view, the PFSS possibly considers a vehicle as a non-traversable area. Therefore, the PFSS is designed and tested in an open area only. Figure 6-11 shows the PFSS test results at the Gainesville Raceway (see satellite photo in the figure 6-7 (A)). In each case, the source image, segmented image, and the TIN control points' image are displayed. Figure 6-11(A) shows a straight road and Figure 6-11 (B) shows a T-intersection area on the right hand side. In the TIN control points' image, the right intersection is identified. Figure 6-11 (C) shows a curved road with 10 points being used to describe it. Normally, a curved road area needs more points to represent a ground surface than a straight road area. When a vehicle turns at a T-intersection, a partial part of road is visible at the camera. Figure 6-11 (D) shows such a situation.

Figure 6-12 shows the University of Florida campus test. Straight road and curved road cases are shown in Figure 6-12 (A) and (B), respectively. In Figure 6-12 (C), part of the road is occluded by a bus traveling in the other lane. Therefore representation of the ground surface is incorrect.

Building vector-based map

Both the LFSSWing and the PFSS component's vector-based lane object and ground surface maps are reconstructed. In the section, the Gainesville Raceway and the University of Florida campus are selected as a test environment. Figure 6-7 (A) and Figure 6-8 (A) show each area's satellite image.

To generate a lane object vector representation, the lane object is detected, converted from the local coordinate system to the global coordinate system, and then stored into the WMVKS. Chapter 4 describes the lane finder algorithm. Figure 6-13 (A, B) shows the vector representation of a lane at the Gainesville Raceway and at the University of Florida campus, respectively.

For a ground surface vector representation, the ground surface is classified, and road boundary vector points extracted and converted from the local coordinate system to the global coordinate system, before being stored into the WMVKS. Chapter 3 describes the path finder algorithm. Figure 6-14 (A, B) shows the vector representation of the ground surface at the Gainesville Raceway and at the University of Florida campus, respectively.

Conclusions

The vector-based ground surface and lane objects representation algorithms are development and implementation to extract and simplify the traversable area by using a camera sensor. Unlike in simulation, algorithms and methods are engineered for outdoor real-time applications with continuous and robust output.

This approach allows a robot to have a human-like cognitive system. People feel comfortable when they drive a known area, because the human brain is able to store important features by experience. This vision system's vector output is small enough to be stored and retrieved like the human brain. Therefore, the vector output can be utilized to rebuild road maps.

Also, properties of the road are calculated to better understand the world, such as lane with and lane color. This information assists the vehicle's intelligence element in making proper decisions. All vector data can be stored in a database system in real-time.

Confidence values of output data are also computed. These values play a key role when data are judged and fused with data from different types of sensor output, such as from LADAR. It can be fused with vision-based sensor output since all confidence values are normalized.

The author presents results from various test places, time, and conditions. The autonomous run test verifies that this camera-based lane finder and path finder approach creates robust and accurate lane corrections, road map, and lane map building. With a simple camera calibration, this software can be easily deployed to any JAUS system.

Future Work

There are three main areas which could be improved in this research. First, a 3-D model can be generated using GPS height information and pitch information. Currently, a 2-D model is generated based on a flat road assumption. However, if this system is used on a slope, hill, or mountain area, a 3-D road or lane model would provide more accurate information.

Second, the lane line estimator should consider vehicle dynamics. The current estimator uses previously detected or estimated line parameters to estimate future line parameters without considering the vehicle's movement. If vehicle's yaw information is added to the estimator in addition to the currently used parameters, the system could generate better estimations, especially when the vehicle travels through intersection or cross-road areas.

Third, a real-time vector output verification procedure is suggested. This system can store and build lane and road models. Therefore, if the vehicle re-explores the same area, the system

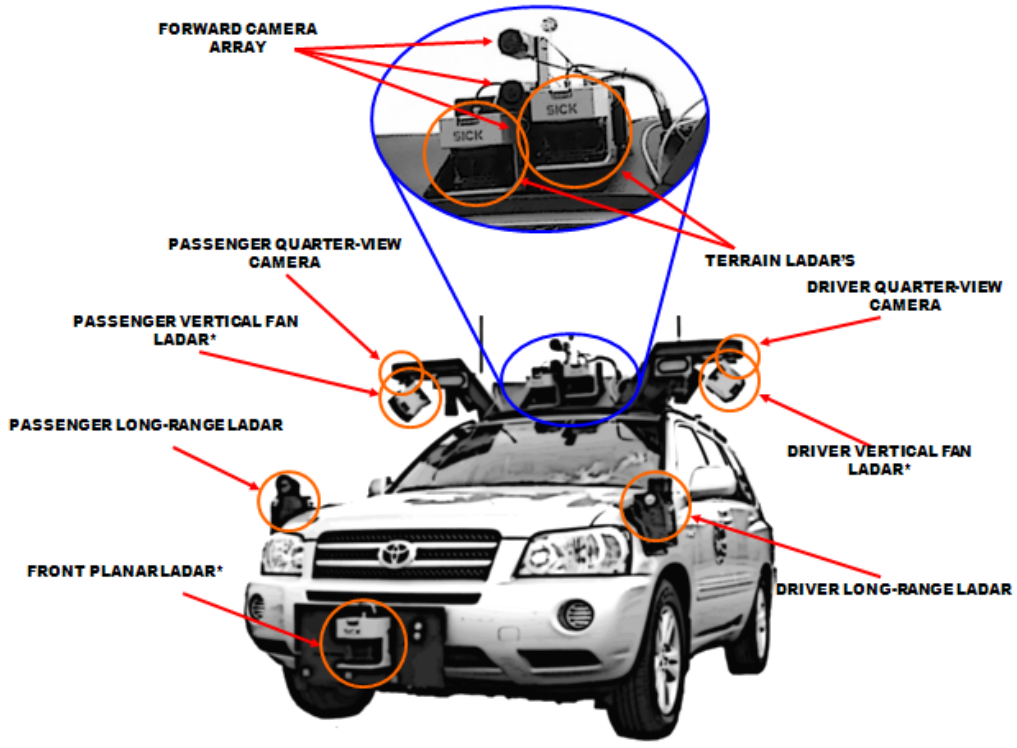
can verify that its current position is within the lane or road by comparing current position with archived lane or road area information.

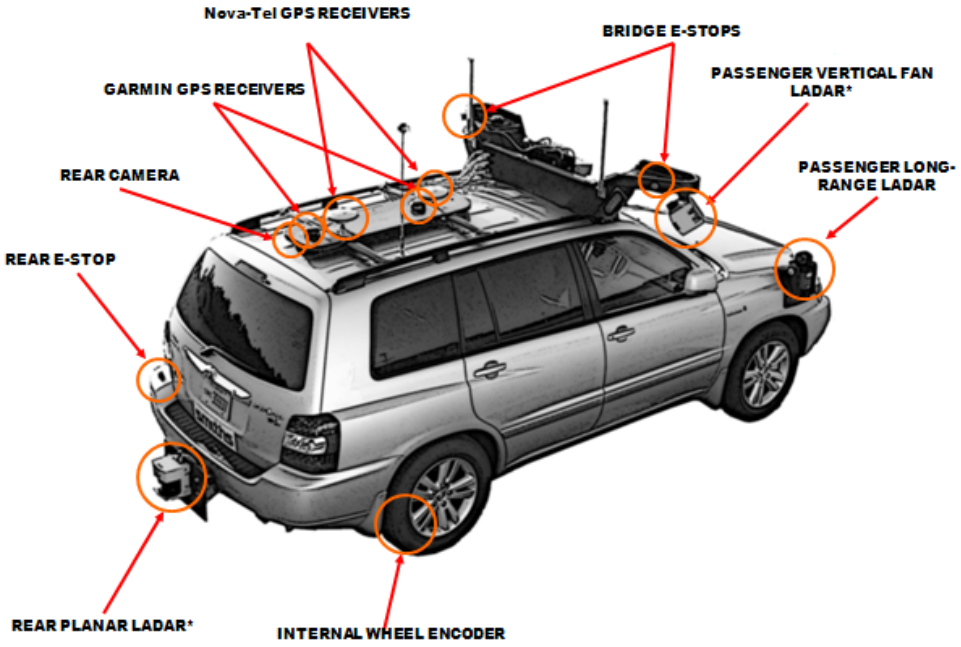
Figure 6-1. CIMAR Navigator III, Urban NaviGator. A) NaviGator III, B) The front view of NaviGator sensor location, and C) The rear view of NaviGator sensor location.

A


NHRA
GAINESVILLE
Raceway
SICK

B


FORWARD CAMERA ARRAY
SICK
SICK
PASSENGER QUARTER-VIEW CAMERA
TERRAIN LADAR'S
DRIVER QUARTER-VIEW CAMERA
PASSENGER VERTICAL FAN LADAR*
PASSENGER LONG-RANGE LADAR
DRIVER VERTICAL FAN LADAR*
FRONT PLANAR LADAR*
DRIVER LONG-RANGE LADAR



C

Figure 6-1. Continued.

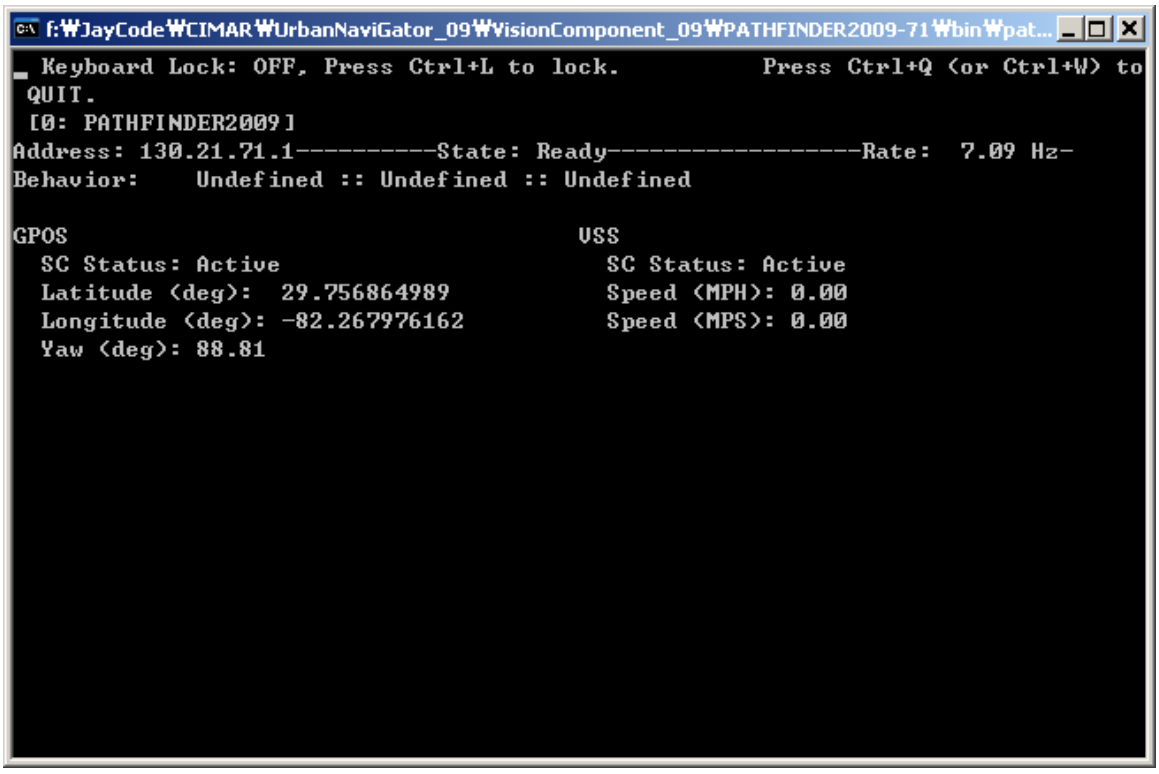
A

B

Figure 6-2. NaviGator III camera sensor systems. A) Cameras location. Center camera is shown in red circle and LFSSWing cameras are shown in blue circles, B) Computer system in truck.

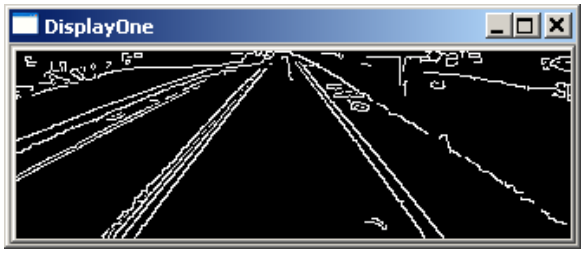
Figure 6-3. The PFSS software. (A) JAUS service connection windows, (B) source image, (C) edge filtered image, (D) lane mask image, (E) source image over lane mask, (F) training area image, (G) classified image, (H) high resolution bird's eye view image, (I) boundary candidates point, (J) TIN control points image, (K) 0.25 meter resolution grid-map image without heading, (L) 0.25 meter resolution grid-map image with heading, and (M) information GUI windows

A


f:₩JayCode₩CIMAR₩UrbanNaviGator_09₩VisionComponent_09₩PATHFINDER2009-71₩bin₩pat...
Keyboard Lock: OFF, Press Ctrl+L to lock. Press Ctrl+Q (or Ctrl+W) to QUIT.
[0: PATHFINDER2009]
Address: 130.21.71.1----------State: Ready----------Rate: 7.09 Hz-
Behavior: Undefined :: Undefined :: Undefined
GPOS
SC Status: Active
Latitude (deg): 29.756864989
Longitude (deg): -82.267976162
Yaw (deg): 88.81
VSS
SC Status: Active
Speed (MPH): 0.00
Speed (MPS): 0.00

B


DisplayOne


DisplayOne

C

D

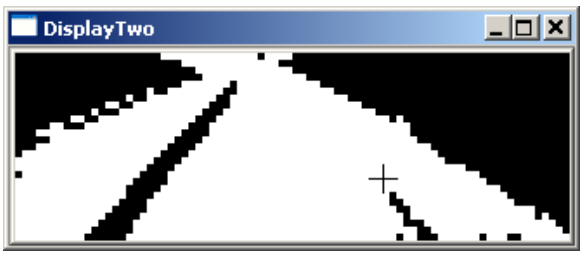

DisplayOne


DisplayOne

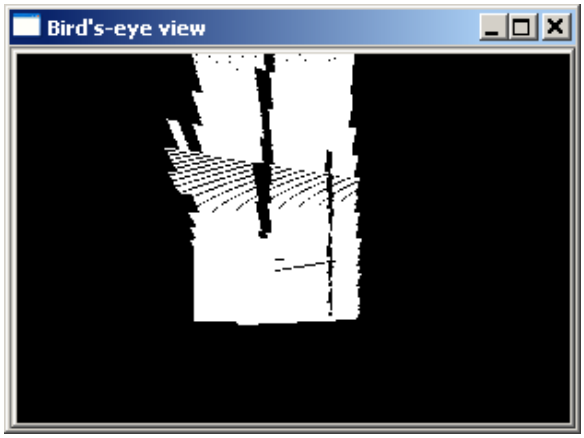
E

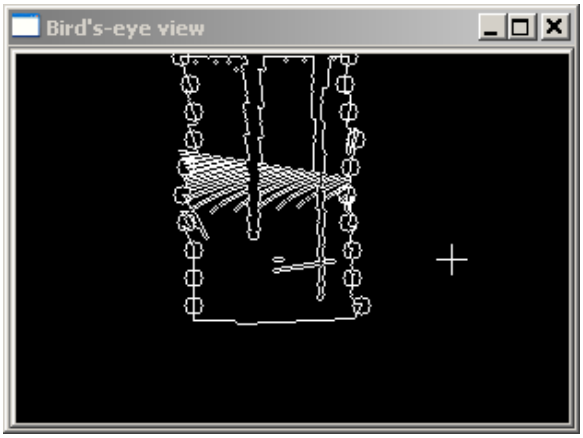
F


DisplayOne

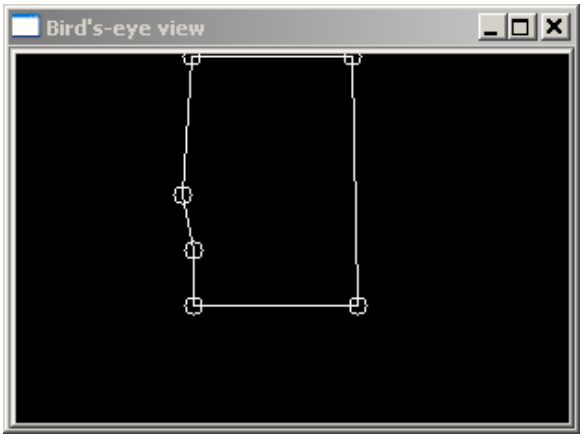

DisplayTwo

G

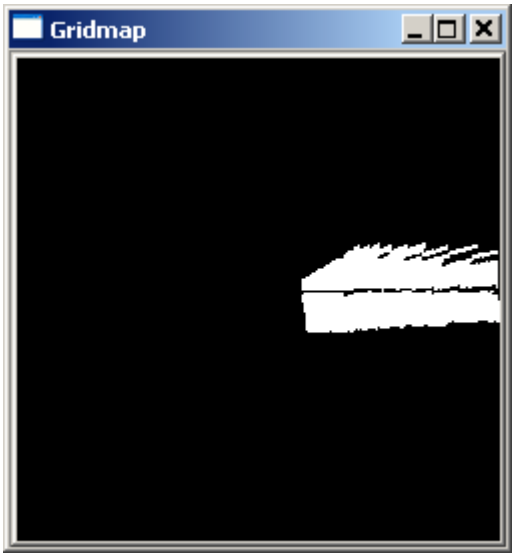
H 



I

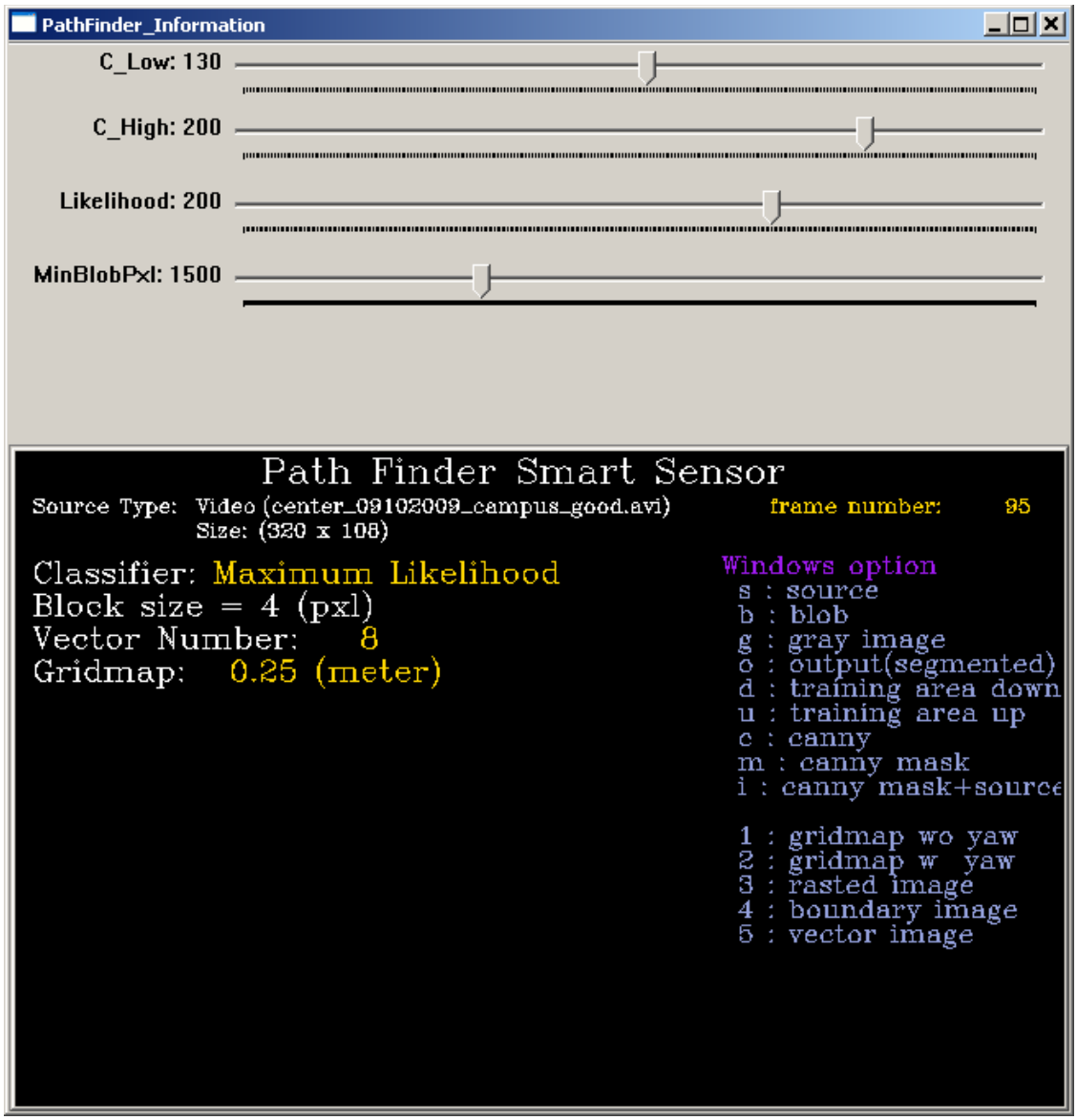
J 

K 



L

Figure 6-3. Continued.

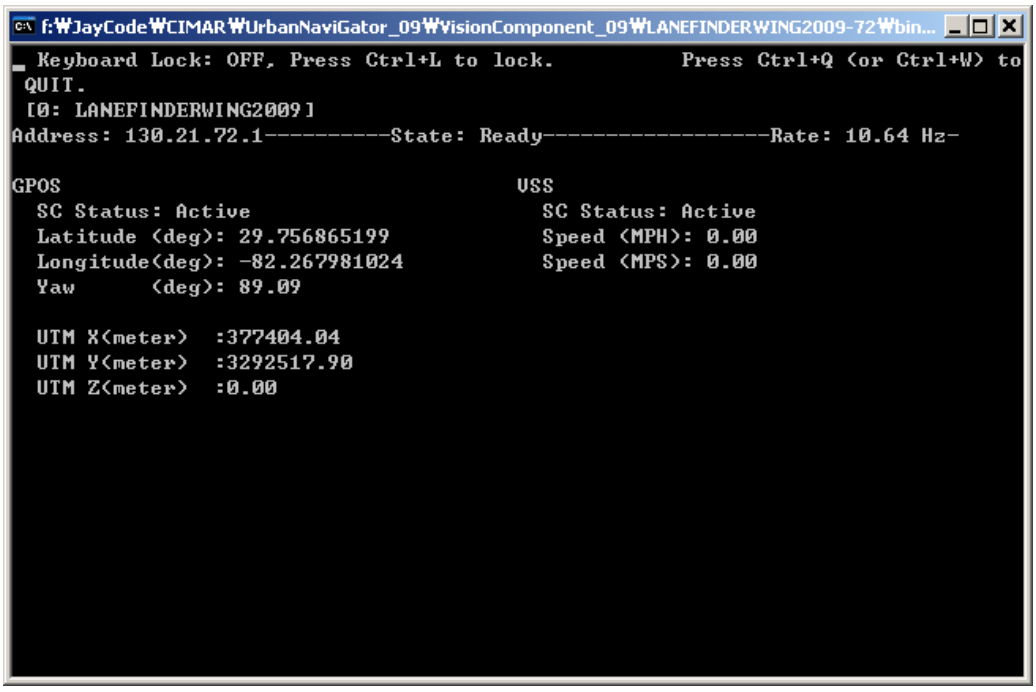


M

Figure 6-3. Continued.

Figure 6-4. The LFSSWing software A) JAUS service connection windows, B) source image, C) edge filtered image, D) Hough line image, E) detected lane line overlay image, and F) the LFSSWing information window

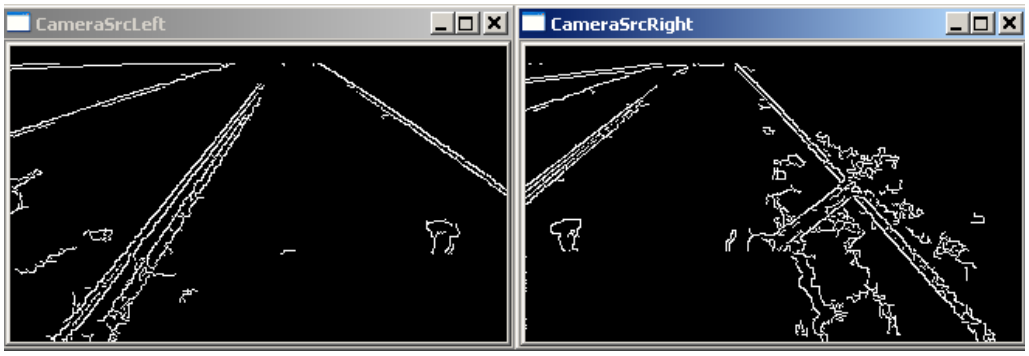
A

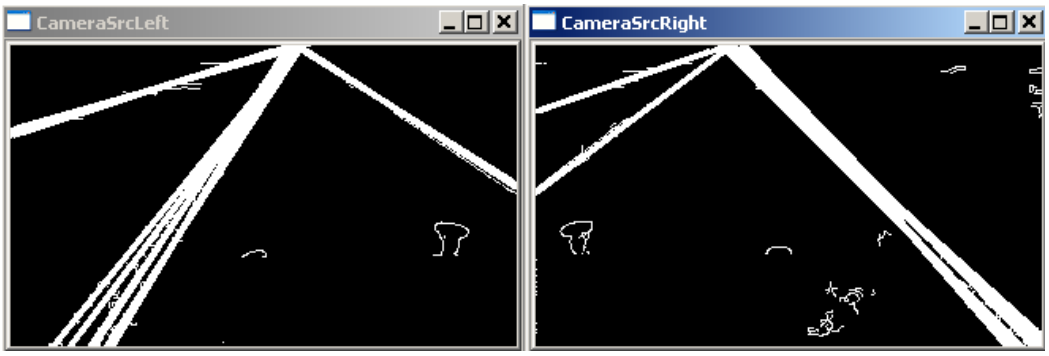

f:₩JayCode₩CIMAR₩UrbanNaviGator_09₩VisionComponent_09₩LANEFINDERWING2009-72₩bin...
Keyboard Lock: OFF, Press Ctrl+L to lock.
Press Ctrl+Q (or Ctrl+W) to
QUIT.
[0: LANEFINDERWING2009]
Address: 130.21.72.1----------State: Ready----------Rate: 10.64 Hz-
GPOS
SC Status: Active
Latitude (deg): 29.756865199
Longitude(deg): -82.267981024
Yaw (deg): 89.09
UTM X(meter) :377404.04
UTM Y(meter) :3292517.90
UTM Z(meter) :0.00
VSS
SC Status: Active
Speed (MPH): 0.00
Speed (MPS): 0.00

B


CameraSrcLeft
CameraSrcRight

C


CameraSrcLeft
CameraSrcRight

D 

E 

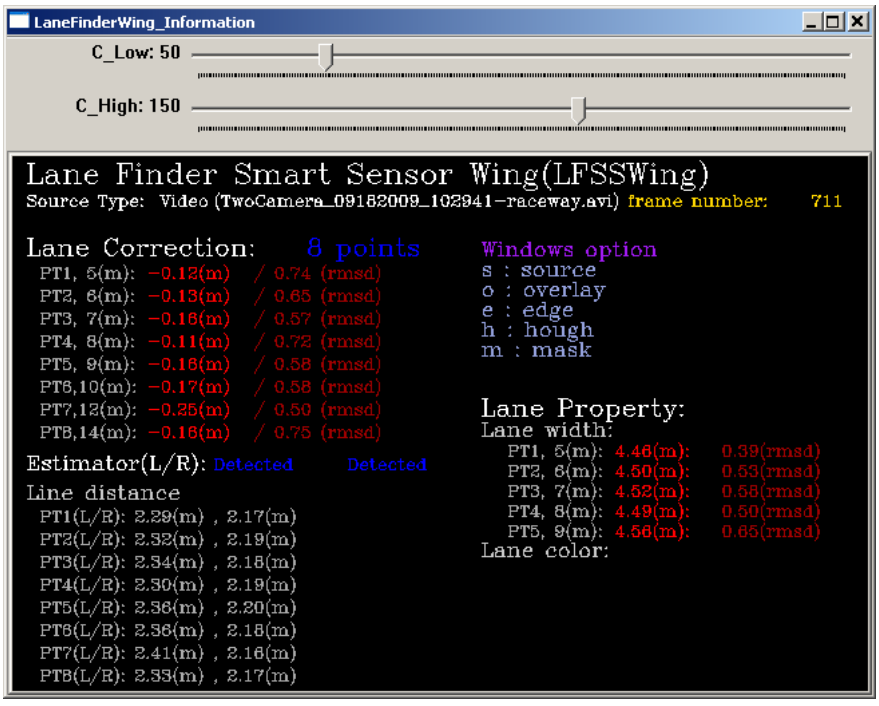
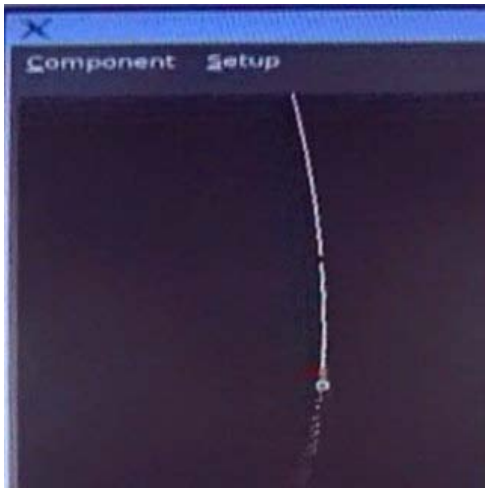
F 

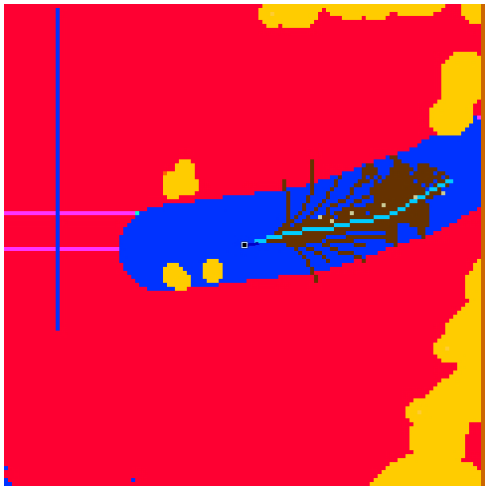
Figure 6-4. Continued.

A

B

C





D

Figure 6-5. The LFSS Arbiter and the RN screen. A) Curved test area, B) different point of view of curved test area, C) the LFSS Arbiter curve fit[2] and D) the RN's A* search result. Brown branches are A* search candidate branch and white points are intermediate travel point by the LFSS Arbiter.

[2] This image is generated by Phil Osteen [Osteen 2008].

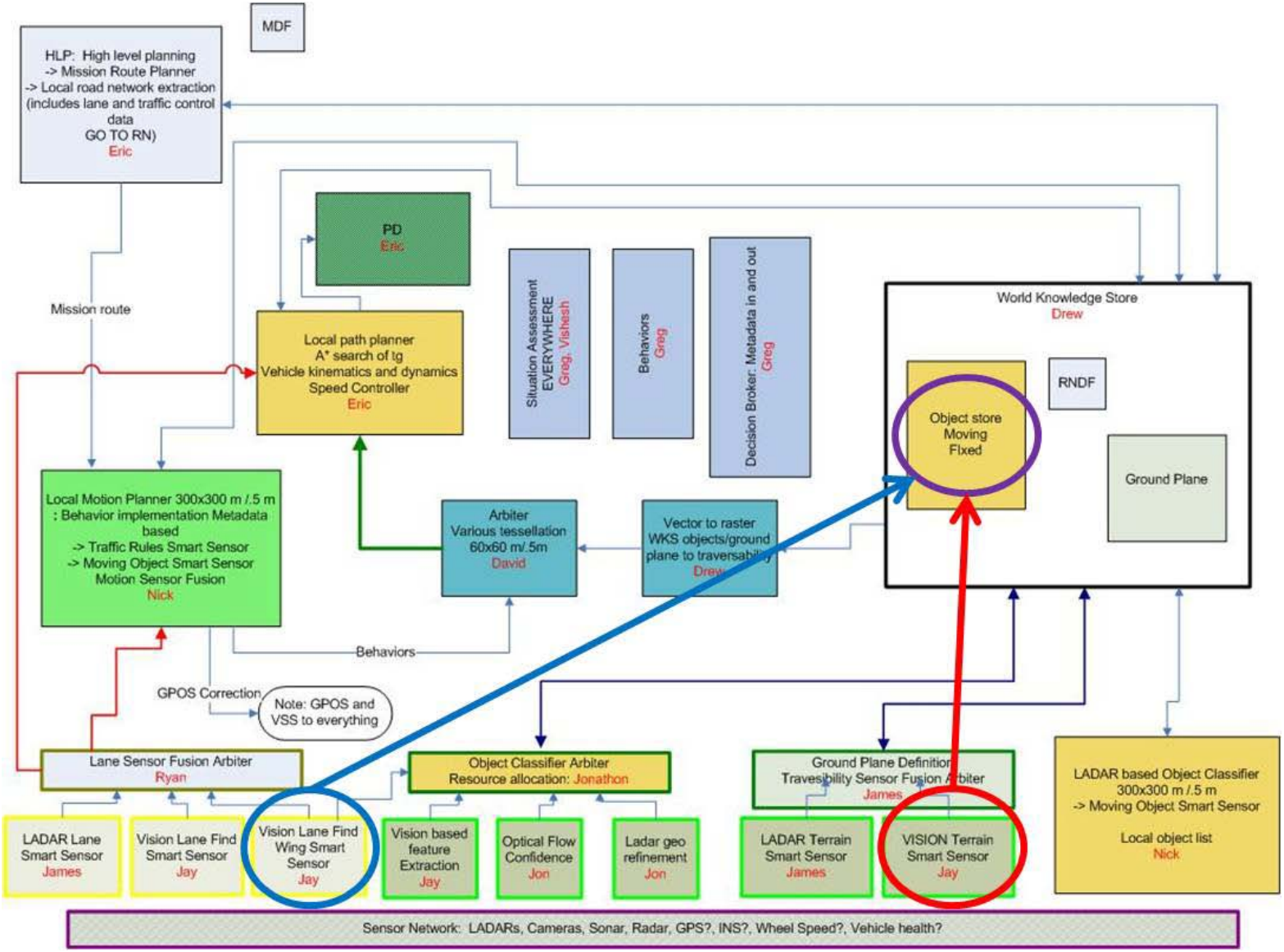


Figure 6-6. Urban NaviGator 2009 system architecture.

Figure 6-7. The LFSSWing test result at the Gainesville Raceway. A) Straight lane, B) curved lane, C) segmented lane, D) narrow lane, E) T-intersection lane, and F) when a vehicle drives on the lane line in autonomous run.

A 

B 

C 

D 

E 

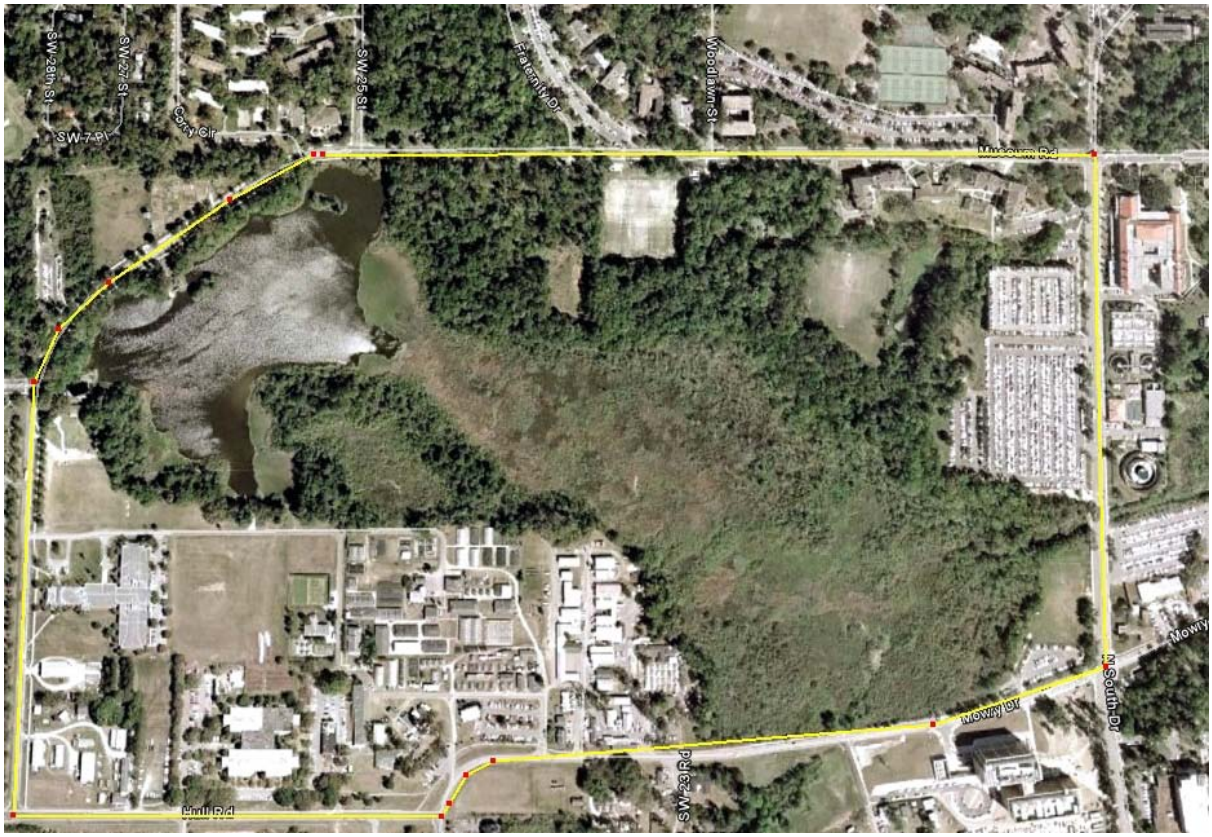
F 

G 

Figure 6-7. Continued.

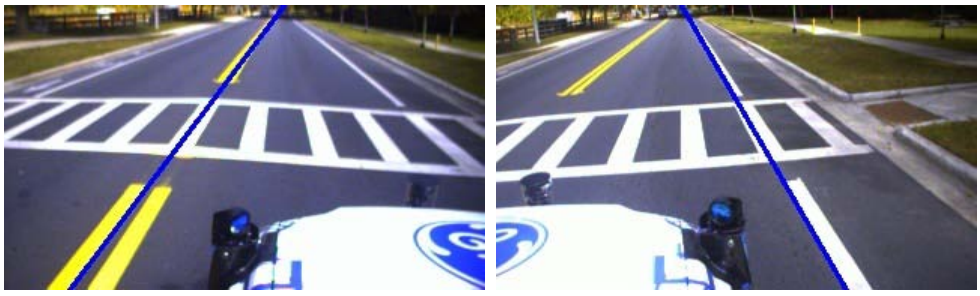
Figure 6-8. The LFSSWing test results at the University of Florida campus. A) Satellite photo, B) lane with bike lane in the shadow, C) pedestrian crossing area, and D) curb area.

A


SW 28th St
SW 27th St
SW 7 Pl
Corry Cir
SW 25 St
Fraternity Dr
Woodlawn St
Museum Rd
Mowry Dr
N South Dr
SW 23rd St
Hull Rd

B

C



D 

Figure 6-8. Continued.

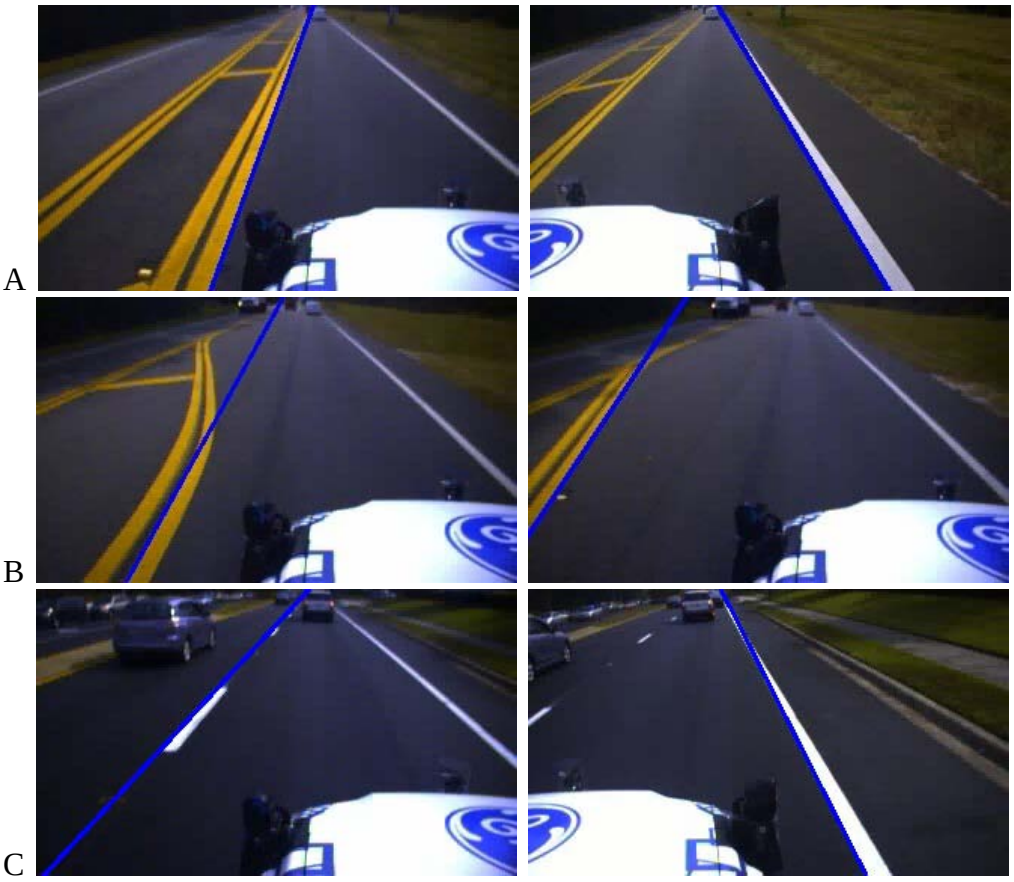


Figure 6-9. The LFSSWing test results in the urban road. A) Multiple center lines, B) divided lane, C) real traffic situation in four lanes road.

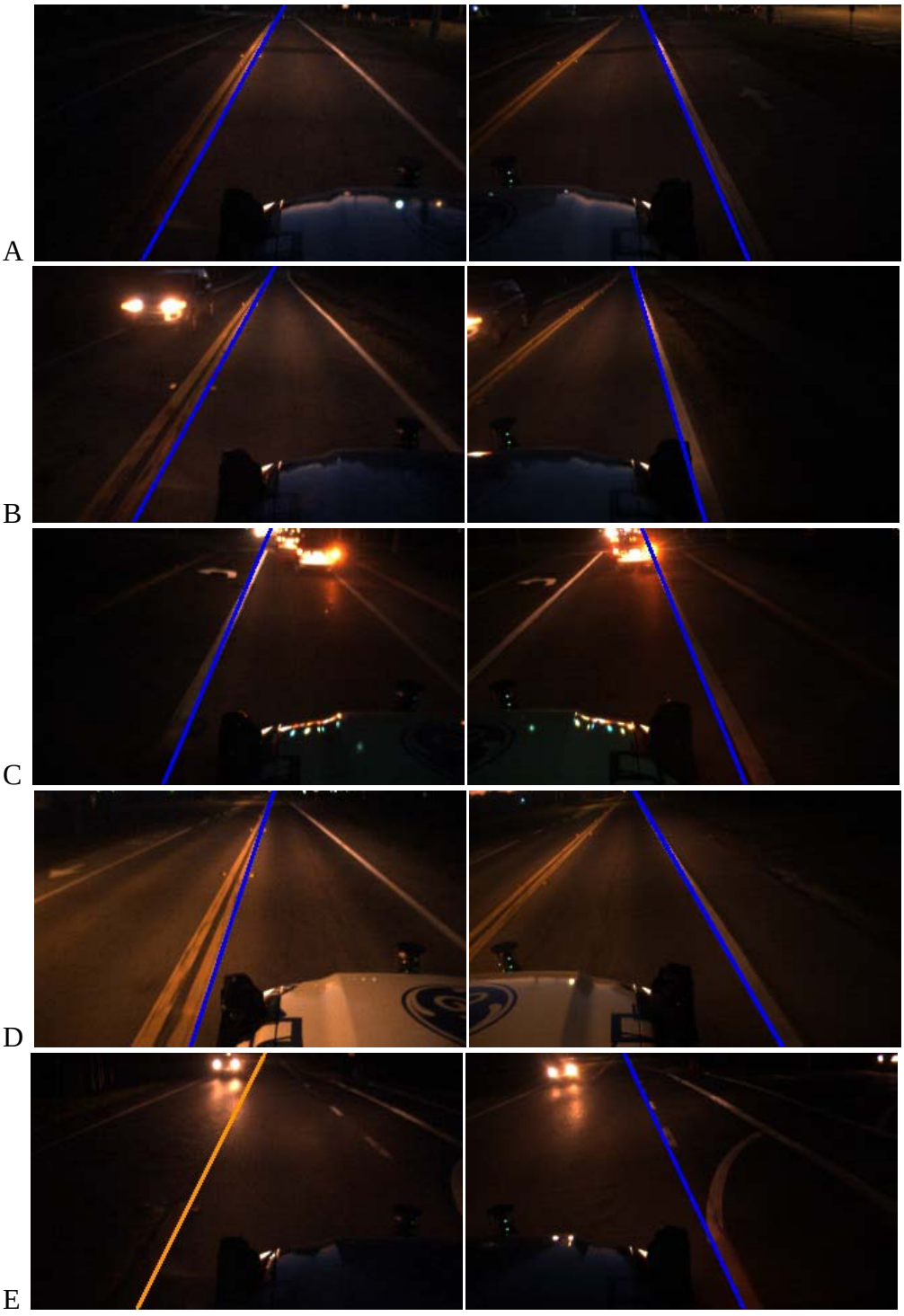
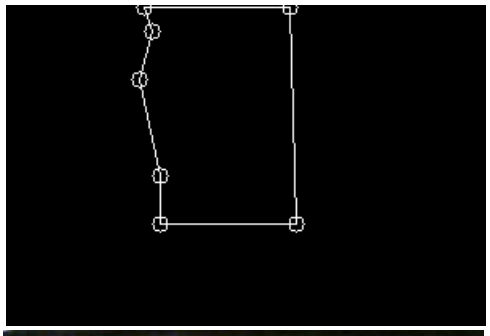


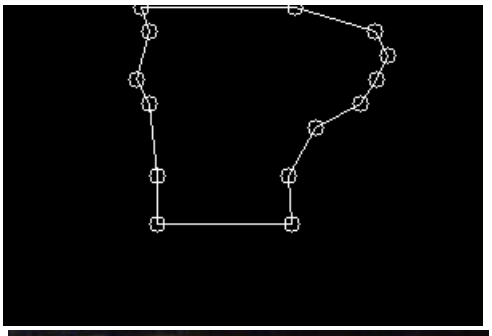
Figure 6-10. The LFSSWing test results at the University of Florida with nighttime setting. A) Straight road, B) other vehicle passed at the other lane, C) other vehicle travel in front of a NiviGator III, D) under a streetlight and E) estimator.

Figure 6-11. The PFSS test results in the Gainesville Raceway. Source, segmented and the TIN control points' images, respectively. A) Straight road, B) T-intersection on right side C) curved road, and D) T-intersection in front.

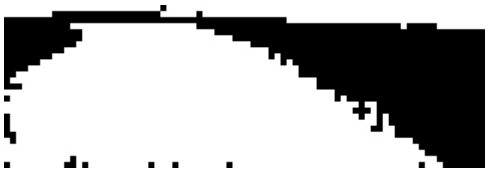
A

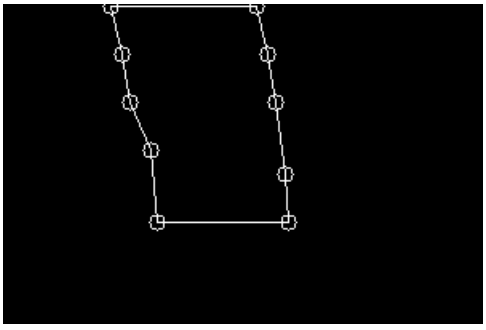


B



C





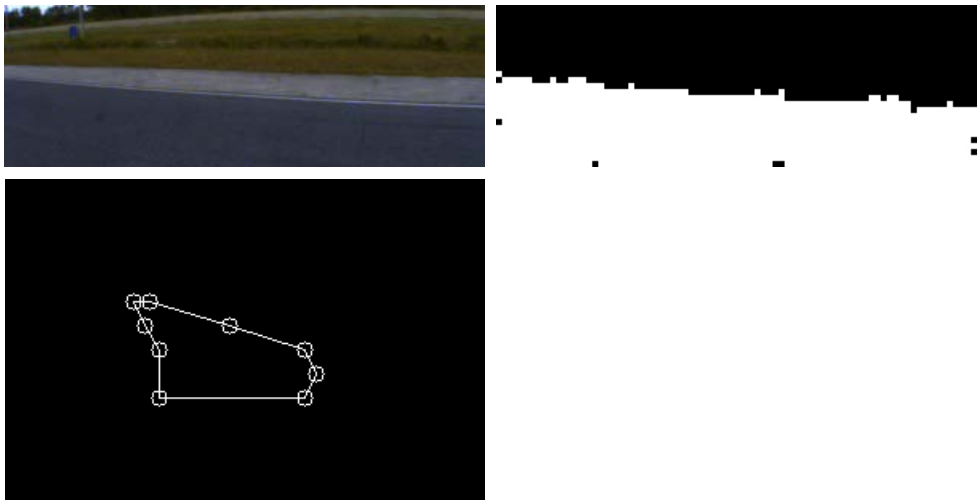
D 

Figure 6-11. Continued.

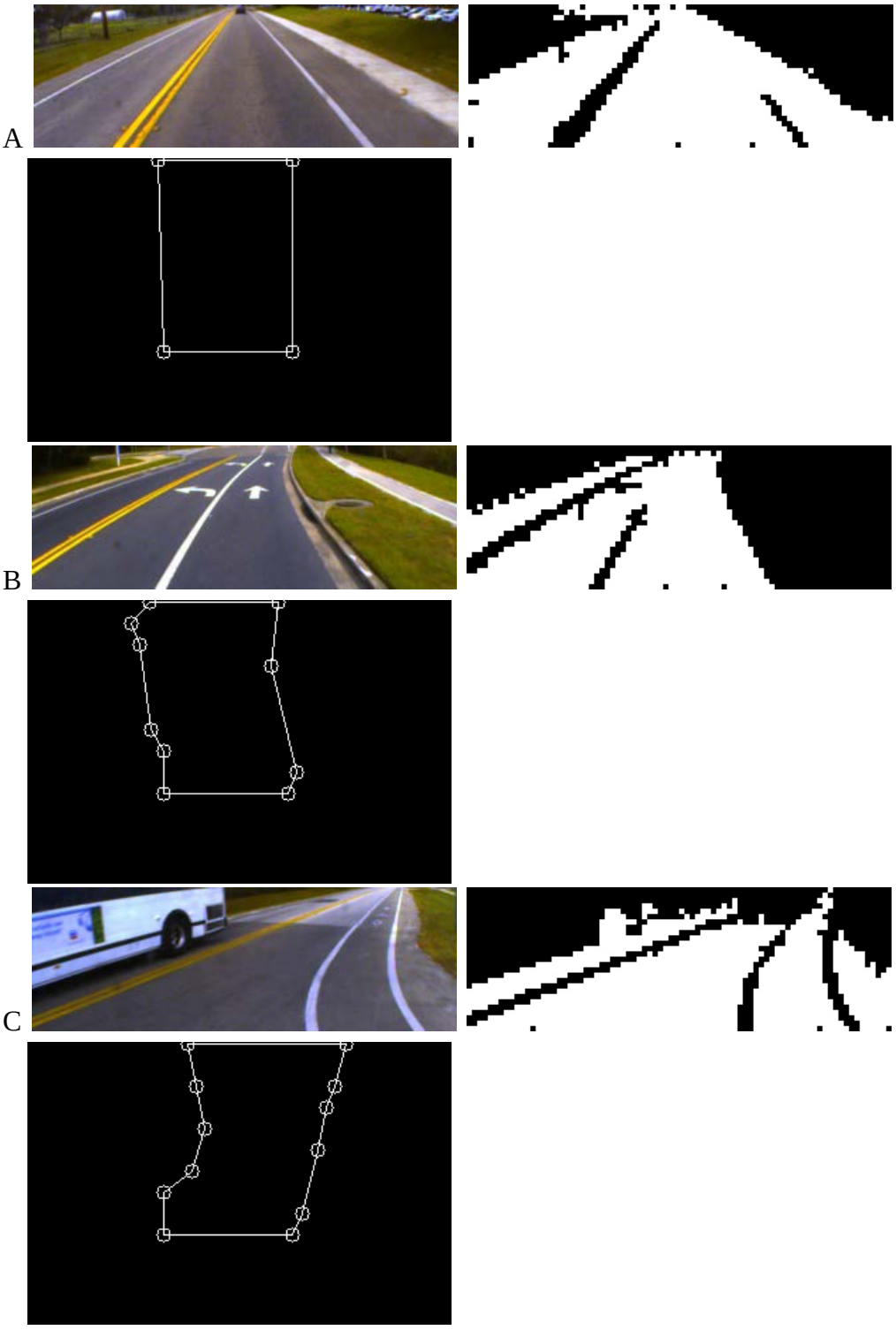
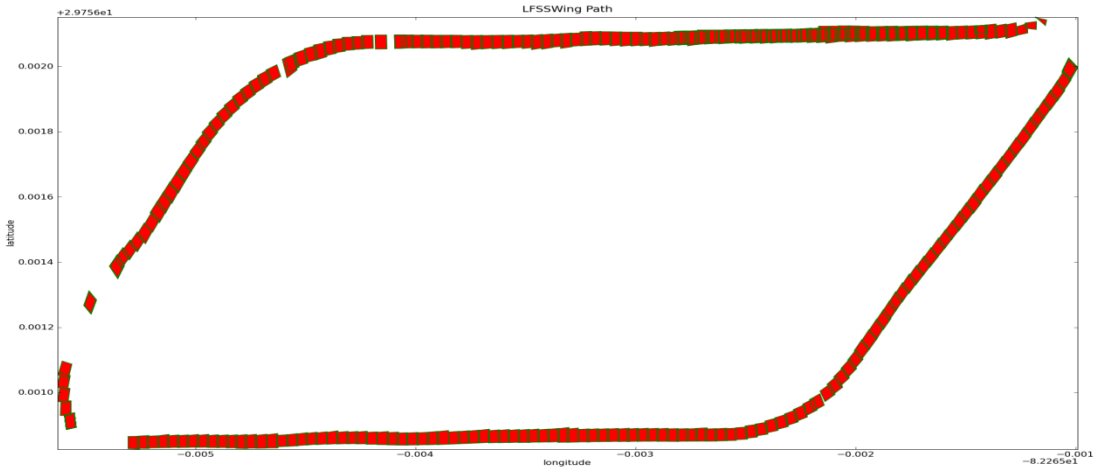
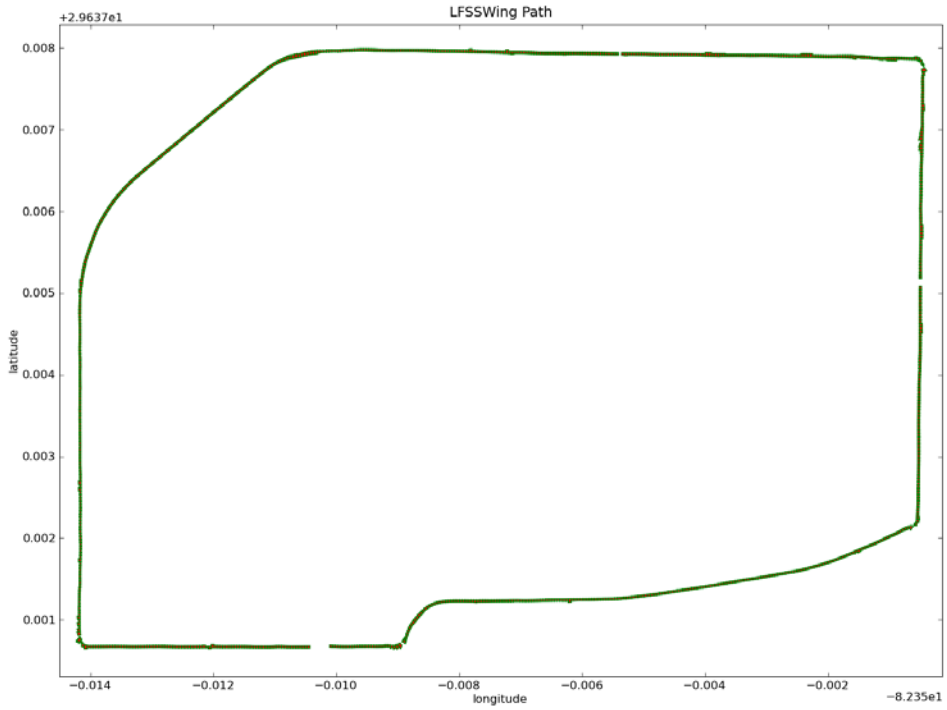


Figure 6-12. The PFSS test results in the University of Florida campus. Source, segmented and the TIN control points' images, respectively. A) Straight road, B) curved road, and C) block by other vehicle.

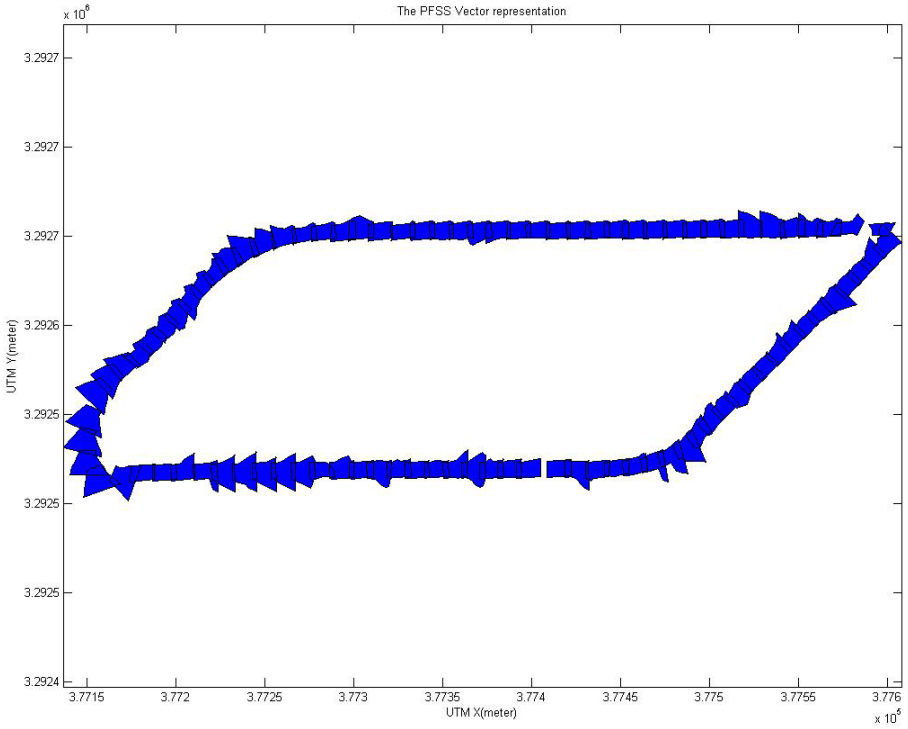


A

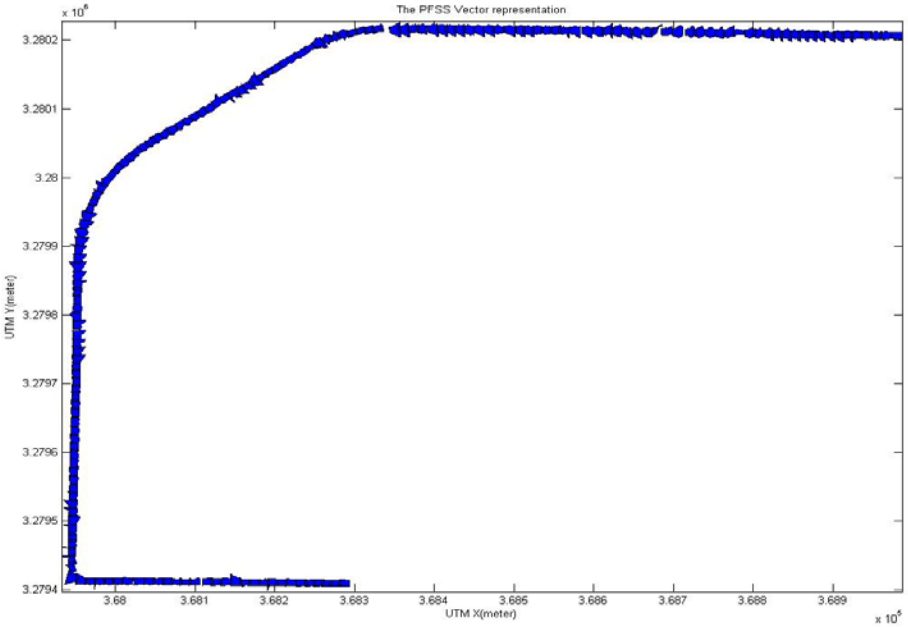


B

Figure 6-13. The LFSSWing vector-based representation. A) The Gainesville Raceway (compare to figure 6-7(A) is satellite image), B) the University of Florida campus (compare to figure 6-8 (A) is satellite image).



A



B

Figure 6-14. The PFSS vector-based representation. A) The Gainesville Raceway (compare to figure 6-7(A) is satellite image), B) the University of Florida campus (compare to figure 6-8 (A) is satellite image).

LIST OF REFERENCES

B. Apolloni, A. Ghosh, F. Alpaslan, L. C. Jain, and S. Patnaik, Machine Learning and Robot Perception (Studies in Computational Intelligence). Berline Heidelberg: Springer, 2005.

R. Behringer, S. Sundareswaran, B. Gregory, R. Elsley, B. Addison, W. Guthmiller, R. Daily, and D. Bevly, "The DARPA grand challenge - development of an autonomous vehicle," in Intelligent Vehicles Symposium, 2004 IEEE, 2004, pp. 226-231.

U. Bergquist, "Colour Vision and Hue for Autonomous Vehicle Guidance," Sweden: Linktping University, 1999.

M. Bertozzi and A. Broggi, "GOLD: a parallel real-time stereo vision system for genericobstacle and lane detection," Image Processing, IEEE Transactions on, vol. 7, pp. 62-81, 1998.

M. Bertozzi and A. Broggi, "Vision-based vehicle guidance," Computer, vol. 30, pp. 49-55, 1997.

S. Beucher and M. Bilodeau, "Road segmentation and obstacle detection by a fast watershed transformation," in Intelligent Vehicles '94 Symposium, Proceedings of the, 1994, pp. 296-301.

T. Bräunl, Embedded Robotics: Mobile Robot Design and Applications with Embedded Systems 3rd ed. Berlin Heidelberg: Springer 2008.

A. Broggi, "Robust real-time lane and road detection in critical shadow conditions," in Computer Vision, 1995. Proceedings., International Symposium on, 1995, pp. 353-358.

A. Broggi and S. Berte, "Vision-based road detection in automotive systems: A real-time expectation-driven approach," Arxiv preprint cs.AI/9512102, 1995.

R. C. Chandler, "Autonomous agent navigation based on textural analysis", Ph.D. dissertation, University of Florida, Gainesville, FL, U.S.A, 2003.

Charles, Martial H. Hebert, Takeo Kanade, 1998, “Vision and Navigation for the Carnegie-Mellon Navlab”, IEEE Transactions on Pattern Analysis and Machine Intelligence. Vol. 10, No. 3, May 1988

J. D. Crisman and C. E. Thorpe, "SCARF: a color vision system that tracks roads and intersections," Robotics and Automation, IEEE Transactions on, vol. 9, pp. 49-58, 1993.

E. R. Davies, Machine Vision : Theory, Algorithms, Practicalities (Signal Processing and its Applications). San Franscisco, CA: Morgan Kaufmann, 2004.

I. L. Davis, A. Kelly, A. Stentz, and L. Matthies, "Terrain typing for real robots," in Intelligent Vehicles '95 Symposium., Proceedings of the, 1995, pp. 400-405.

R. O. Duda, P. E. Hart, and D. G. Stork, Pattern Classification, 2nd ed. New York: Willy-Interscience, 2000.

R. O. Duda and P. E. Hart, "Use of the Hough transformation to detect lines and curves in pictures," Commun. ACM, vol. 15, pp. 11-15, 1972.

Florida Department of Transportation, Roadway design, “Plans Preparation Manual- Jan 2009 update, Volume 1 – Design Criteria and Process”, January, 2009, http://www.dot.state.fl.us/rddesign/PPMManual/2009/Volume1/2009Vol1.shtm,2009

D. W. Gage, "A brief history of unmanned ground vehicle (UGV) development efforts," Unmanned Systems, vol. 13, pp. 9-32, 1995.

R. C. Gonzalez, R. E. Woods, and S. L. Eddins, Digital Image Processing Using MATLAB. New Jersey: Prentice Hall, 2004.

T. Galluzzo, "Simultaneous Planning and Control for Autonomous Ground Vehicles.", Ph.D. dissertation, University of Florida, Gainesville, FL., U.S.A, 2006.

R. Hartley and A. Zisserman, Multiple View Geometry in Computer Vision. Cambridge, UK: CAMBRIDGE UNIVERSITY PRESS, 2004.

K. Hashimoto, S. Nakayama, T. Saito, S. Ishida, K. Unuora, J. Ishii, N. Oono, and Y. Okada, "An image processing architecture and a motion control method for an autonomous vehicle," in Intelligent Vehicles '92 Symposium., Proceedings of the, 1992, pp. 213-218.

M. Isard and A. Blake, "CONDENSATION—Conditional Density Propagation for Visual Tracking," International Journal of Computer Vision, vol. 29, pp. 5-28, 1998.

JAUS Working Group, "Joint Architecture for Unmanned Systems(JAUS)", http://www.jauswg.org/,2009

J. C. McCall and M. M. Trivedi, "Video-based lane estimation and tracking for driver assistance: survey, system, and evaluation," IEEE Transactions on Intelligent Transportation Systems, vol. 7, pp. 20-37, 2006.

A. Kelly and A. Stentz, "An approach to rough terrain autonomous mobility," in International Conference on Mobile Planetary Robots, 1998, pp. 129-198.

Z. Kim, "Realtime lane tracking of curved local road," in Intelligent Transportation Systems Conference, 2006. ITSC '06. IEEE, 2006, pp. 1149-1155.

Z. Kim, "Robust Lane Detection and Tracking in Challenging Scenarios," Intelligent Transportation Systems, IEEE Transactions on, vol. 9, pp. 16-26, 2008.

K. Kluge, "Extracting road curvature and orientation from image edge points without perceptual grouping into features," in Intelligent Vehicles' 94 Symposium, Proceedings of the, 1994, pp. 109-114.

K. Kluge and S. Lakshmanan, "A deformable-template approach to lane detection," in Intelligent Vehicles '95 Symposium., Proceedings of the, 1995, pp. 54-59.

C. Kreucher and S. Lakshmanan, "LANA: a lane extraction algorithm that uses frequency domain features," Robotics and Automation, IEEE Transactions on, vol. 15, pp. 343-350, 1999.

Krishnan, N., Banu, M. S. and Christiyana, C. C. “Content Based Image Retrieval Using Dominant Color Identification Based on Foreground Objects”, City, 2007.

J. Lee and C. D. Crane, "Road Following in an Unstructured Desert Environment Based on the EM(Expectation-Maximization) Algorithm," in SICE-ICASE, 2006. International Joint Conference, 2006, pp. 2969-2974.

J. W. Lee, "A machine vision system for lane-departure detection," Comput. Vis. Image Underst., vol. 86, pp. 52-78, 2002.

Lukac, R. and K.N. Plataniotis, "Color Image Processing: Methods and Applications". 2006, Boca Raton, FL: CRC Press.

Matrix Vision, "The mvBlueFOX USB 2.0 camera", http://www.matrix-vision.com/products/hardware/mvbluefox.php?lang=en,2009

L. Matthies, M. Maimone, A. Johnson, Y. Cheng, R. Willson, C. Villalpando, S. Goldberg, A. Huertas, A. Stein, and A. Angelova, "Computer Vision on Mars," International Journal of Computer Vision, vol. 75, pp. 67-92, 2007.

J. C. McCall and M. M. Trivedi, "Video-based lane estimation and tracking for driver assistance: survey, system, and evaluation," IEEE Transactions on Intelligent Transportation Systems, vol. 7, pp. 20-37, 2006.

H. P. Moravec, "The Stanford Cart and the CMU Rover," Proceedings of the IEEE, vol. 71, pp. 872-884, 1983.

M. Nixon and A. S. A. A. (Author), Feature Extraction in Computer Vision and Image Processing, 2nd ed. Oxford, UK: Elsevier, 2008.

P. Osteen, "Local Sensor Data Fusion Applied to Autonomous Navigation", M.S. thesis, University of Florida, Gainesville, Fl., U.S.A, 2008.

D. Pomerleau, "RALPH: Rapidly adapting lateral position handler," in IEEE symposium on intelligent vehicles, 1995, pp. 506-511.

D. Pomerleau and T. Jochem, "Rapidly adapting machine vision for automated vehicle steering," IEEE Expert, vol. 11, pp. 19-27, 1996.

C. Rasmussen, "Combining laser range, color, and texture cues for autonomous road following," in Robotics and Automation, 2002. Proceedings. ICRA'02. IEEE International Conference on, 2002.

M. Sarfraz, Interactive Curve Modeling: with Applications to Computer Graphics, Vision and Image Processing London, UK: Springer-Verlag, 2007.

H. Schneiderman and M. Nashman, "A discriminating feature tracker for vision-based autonomous driving," Robotics and Automation, IEEE Transactions on, vol. 10, pp. 769-775, 1994.

S. Solanki, "Development of Sensor Component for Terrain Evaluation and Obstacle Detection for an Unmanned Autonomous Vehicle", Ph.D. dissertation, University of Florida, Gainesville, Fl., U.S.A, 2006.

B. Southall and C. J. Taylor, "Stochastic road shape estimation," in International Conference on Computer Vision, 2001, pp. 205-212.

R. Sukthankar, D. Pomerleau, C. Thorpe, and I. Carnegie-Mellon Univ Pittsburgh Pa Robotics, Panacea: An Active Sensor Controller for the ALVINN Autonomous Driving System: Carnegie Mellon University, The Robotics Institute, 1993.

T. Suttorp and T. Bucher, "Learning of Kalman Filter Parameters for Lane Detection," in Intelligent Vehicles Symposium, 2006 IEEE, 2006, pp. 552-557.

C. J. Taylor, J. Kosecka, R. Blasi, and J. Malik, "A comparative study of vision-based lateral control strategies for autonomous highway driving," The International Journal of Robotics Research, vol. 18, p. 442, 1999.

C. Thorpe, M. Hebert, T. Kanade, and S. Shafer, "Vision and Navigation for the Carnegie-Mellon Navlab," Annual Reviews in Computer Science, vol. 2, pp. 521-556, 1987.

S. Thrun, M. Montemerlo, H. Dahlkamp, D. Stavens, A. Aron, J. Diebel, P. Fong, J. Gale, M. Halpenny, and G. Hoffmann, "Stanley: The robot that won the DARPA Grand Challenge," JOURNAL OF FIELD ROBOTICS, vol. 23, p. 661, 2006.

R. Touchton, "An Adaptive Planning Framework for Situation Assessment and Decision-Making of an Autonomous Ground Vehicle.", Ph.D. dissertation, University of Florida, Gainesville, FL., U.S.A, 2006.

S. J. Velat, J. Lee, N. Johnson, and C. D. Crane, "Vision Based Vehicle Localization for Autonomous Navigation," in Computational Intelligence in Robotics and Automation, 2007. CIRA 2007. International Symposium on Jacksonville, FI 2007, pp. 528-533

V. Vezhnevets, V. Sazonov, and A. Andreeva, "A Survey on Pixel-Based Skin Color Detection Techniques," in Proc. Graphicon, 2003.

Y. Wang, D. Shen, and E. K. Teoh, "Lane detection using catmull-rom spline," in IEEE International Conference on Intelligent Vehicles, 1998.

Y. Wang, D. G. Shen, and E. K. Teoh, "Lane detection using spline model," Pattern Recognition Letters, vol. 21, pp. 677-689, Jul 2000.

Y. Wang, E. K. Teoh, and D. G. Shen, "Lane detection and tracking using B-Snake," Image and Vision Computing, vol. 22, pp. 269-280, Apr 1 2004.

C. Witzgall, J. Bernal, and G. S. Cheok, "TIN Techniques for Data Analysis and Surface Construction," National Institute of Standards and Technology Internal Report 7078, 2004, 2004.

B. F. Wu, C. T. Lin, and Y. L. Chen, "Dynamic Calibration and Occlusion Handling Algorithms for Lane Tracking," Ieee Transactions on Industrial Electronics, vol. 56, pp. 1757-1773, May 2009.

Y. U. Yim and S. Y. Oh, "Three-feature based automatic lane detection algorithm (TFALDA) for autonomous driving," IEEE Transactions on Intelligent Transportation Systems, vol. 4, pp. 219-225, 2003.

Yu, X., S. Beucher, and M. Bilodeau. "Road tracking, lane segmentation and obstacle recognition by mathematical morphology". in Intelligent Vehicles '92 Symposium., Proceedings of the. 1992.

J. Zhang and H. H. Nagel, "Texture-based segmentation of road images," in Intelligent Vehicles' 94 Symposium, Proceedings of the, 1994, pp. 260-265.

BIOGRAPHICAL SKETCH

Jaesang Lee was born and raised in Seoul, Korea. He completed his Bachelor of Science degree in an Electrical Engineering Department at the Inha University in Inchon, Korea, and began his Master of Science degree at Inha University. After finishing military duty at the Air Force, Jaesang decided to go back to school to finish his master's degree in the United States. He finished his master's program in Electrical and Computer Engineering Department at the University of Florida and then joined his Ph.D. program at the Center for Intelligent Machines and Robotics (CIMAR) in the Mechanical and Aerospace Engineering Department. Jaesang has worked on various robotics projects as a graduate research assistant under the guidance of his advisor, Dr. Carl D. Crane III. Among the projects, Jaesang was actively involved in the autonomous vehicle development for the Defense Advanced Research Projects Agency (DARPA) Grand Challenge 2005 and DARPA Urban Challenge 2007. His current research goal is computer vision system development for autonomous vehicle and drive assistant system. After graduation, Jaesang will continue his career as an engineer working in the robotics industry. His field of interest includes computer vision, image processing, and pattern recognition system development for real time and real world.