Understanding the Visual Appearance of Road Scenes Using a Monocular Camera

Submitted in partial fulfillment of the requirements for

the degree of

Doctor of Philosophy

in

Department of Mechanical Engineering

Jong Ho Lee

B.S., Mechanical Engineering, Hanyang University
M.S., Mechanical Engineering, Carnegie Mellon University

Carnegie Mellon University
Pittsburgh, PA

February, 2017

Abstract

Over the past several decades, research efforts in the development of self-driving vehicles have drastically improved accompanying technologies. Since the challenges held by Defense Advanced Research Projects Agency, the autonomous driving industry has increased significantly, and almost all the automotive companies have started to develop the technologies to deploy autonomous driving vehicles in the real world. Even though a lot of companies have been making efforts to achieve fully automated vehicles, the current technologies are not mature enough to be deployed in the real world yet, because self-driving vehicles need to respond to uncontrolled environments, such as moving objects, pedestrians, traffic lights, and unexpected work-zones. Among these uncontrolled environments, this thesis focuses on understanding road information and estimating states of traffic lights.

Given that all of the traffic control devices are regularized in colors, color is one of the most significant features to be recognized. In order to accomplish such necessary a vision task, self-driving vehicles must incorporate cameras. Despite the fact that traffic control devices have their own regularized color and cameras can *see* those devices, they are still difficult to detect and recognize by autonomous vehicles. One of the biggest problems is that the color of those devices can be captured differently based on illumination.

In this thesis, we investigate the problem of recognizing static objects using a monocular camera to assist self-driving vehicles in perceiving traffic control devices. The perception system, specifically a camera, should recognize the objects robustly regardless of the environment. Throughout this thesis, we exploit different color spaces and apply machine learning to reduce color variance. Also, we develop algorithms which compensate for illumination changes by considering the Sun position, to further improve the road sign recognition. Furthermore, we improve a traffic light state estimation which performs robustly under various illumination conditions. We deploy and demonstrate all of the algorithms in an autonomous vehicle.

Acknowledgments

The completion of this doctoral thesis could not have been possible without the support and assistance of the people around me.

First, I would like to deeply thank my advisor as well as the chair of the committee, David Wettergreen, for his greatest support, gudiance, encouragement, and patience over the last several years. Throughout my PhD journey, he always provided me invaluable advices and opportunities to grow as a better researcher. Also, he helped me become a more creative and proactive person. Without him, I would not have been able to finish this journey.

I would like to extend my thanks to my committee member, John Dolan. He encouraged me to think and solve problems in both theoretical and practical ways. He also provided me a great opportunity to lead a team, which was a valuable experience for me.

I would like to thank Fernando De la Torre and Levent Burak Kara, my committee members, for agreeing to be a part of this work. Their knowledge and expertise in this field have enriched my research.

I also want to show my appreciation to GM for the financial support.

It has been a great pleasure working with my colleagues. In particular, I thank the members of CMU AD-CRL, Jarrod Snider, Junsung Kim, Gaurav Bhatia, Junqing Wei, Wenda Xu, and Hyoseung Kim for providing invaluable supports and discussion. I enjoyed endless discussion with them and learned a lot in many different ways.

Last but not least, I thank God for the grace he has given me. I would like to also thank my wife, In-Soo Jung, for her endless support and encouragement to finish my PhD journey. It would not have been possible without her. And of course, I must express my gratitude to my parents, Ku-Taek Lee and Jung-Ran Lee, and my sisters, Kyung-Eun Lee and Jooha Lee for always being there for me.

Contents

List of Figures

List of Tables

Chapter 1

Introduction

1.1 Motivation

Over the past several decades, research efforts in the development of self-driving cars have drastically improved accompanying technologies in perception, localization, planning, behavior, and

Figure 1.1: CMU autonomous vehicle

control. In the 1980s, the Navlab of Carnegie Mellon University (CMU) first successfully demonstrated a self-driving vehicle [Kanade et al., 1986]. Since then, many researchers have been conducting experiments to show the capability of autonomous driving vehicles in the real world. In 2004, the Defense Advanced Research Projects Agency (DARPA) started to hold a competition for autonomous driving vehicles [DARPA, 2004] [DARPA, 2005] [DARPA, 2007]. The challenge given in the first and second competitions in 2004 and 2005 was to drive through the desert without human intervention, while the challenge given in the third year was to drive through an artificial city while interacting with other vehicles. Since these competitions, the autonomous driving industry has increased significantly, and almost all the automotive companies have started to develop the technologies to deploy autonomous driving vehicles in the real world. CMU also has been developing an autonomous driving vehicle as depicted in Figure 1.1, and showed its capabilities to drive in urban areas and on highways.

Some automotive companies have now started to produce self-driving vehicles, and they have been improving their technologies by taking different approaches and levels of vehicle automation. There are five levels of vehicle automation defined by the U.S. Department of Transportation's National Highway Traffic Safety Administration (NHTSA): no-automation (level 0), function-specific automation (level 1), combined function automation (level 2), limited self-driving automation (level 3), and full self-driving automation (level 4) [National Highway Traffic Safety Administration, 2013]. Volvo, Mercedez, BMW, Hyundai, Ford, and Tesla are developing self-driving vehicles, and most of these companies already implemented the level 2 of autonomous driving, which is usually a combination of lane centering and adaptive cruise control on highways [Thompson, 2016b] [Howard, 2016] [Marshall, 2016] [Thompson, 2016a] [Weber, 2016] [Kang, 2015]. These companies are targeting the level 4 of vehicle automation by 2025. Other companies, such as Google and Uber, are also developing autonomous driving vehicles by putting LiDAR sensors on top of vehicles [Google, 2016] [Anthony, 2016]. Furthermore, automotive component suppliers, such as Delphi, Bosch, and Continental, are developing not only sensors which can be integrated into vehicles, but also 'fully autonomous' driving technologies as well [Bigelow, 2016] [Gordon-Bloomfield, 2016] [Continental, 2016].

Even though a lot of companies have been making efforts to achieve the level 4 of vehicle automation, the current technologies are not mature enough to be deployed in the real world yet, because self-driving vehicles need to respond to unexpected situations, such as moving objects, pedestrians, traffic lights, and unexpected work-zones. Among these unexpected situations, self-driving vehicles first need to understand road information by reading road signs. More importantly, self-driving vehicles should also respond to traffic lights as well as unexpected signs and channelizers in work-zones. Given that all of the traffic control devices are regularized in colors as shown in Figure 1.2, color is one of the most significant features to be recognized. In order to accomplish such a necessary vision task, self-driving vehicles must incorporate cameras.

Despite the fact that traffic control devices have their own regularized color and cameras can *see* those devices, they are still difficult to detect and recogniz by autonomous vehicles. One

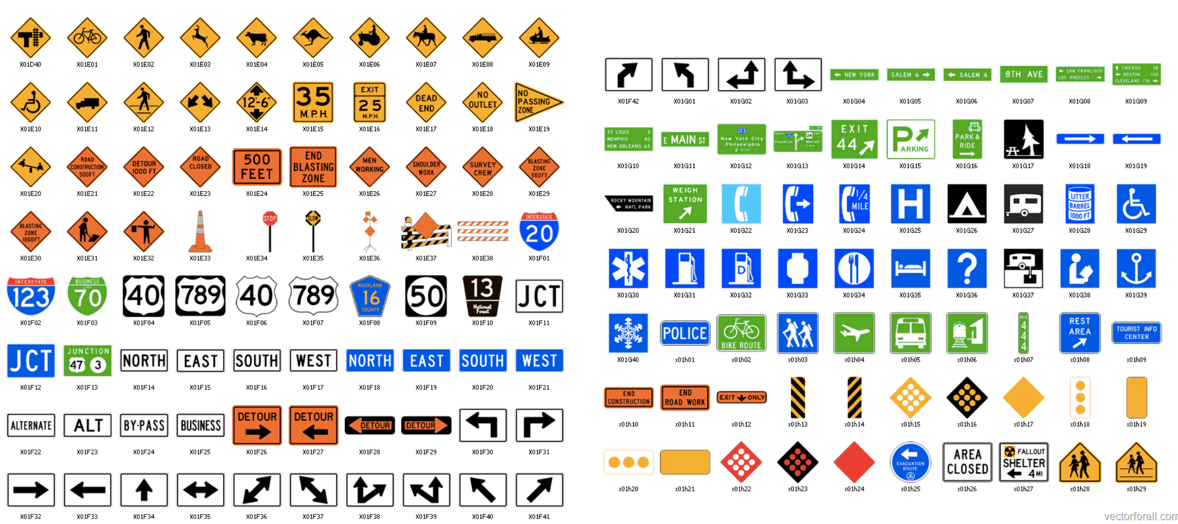


Figure 1.2: Examples of traffic signs. All the color and shape of signs are regularized.

of the biggest problems is that the color of those devices can be captured differently based on illumination, as depicted in Figure 1.3. Cameras capture the amount of reflected and emitted light from objects, and this amount of light can differ based on various factors, such as having different light sources, various object materials, different geometrical relationships between light sources and objects, etc.

Thus, this thesis focuses on the estimation of the color that is especially applicable to recognizing traffic control devices, so that self-driving vehicles can detect and respond to those devices robustly.

Figure 1.3: The same object is seen as having different colors due to different illumination

1.2 Thesis Statement

Modeling illumination improves real-time understanding of road signs and traffic signals by self-driving vehicles under real lighting conditions.

In this thesis, we investigate the problem of recognizing static objects using a monocular camera to assist self-driving vehicles in perceiving traffic control devices. The perception system, specifically a camera, should recognize the objects robustly regardless of the environment. In order to overcome these challenges, we develop algorithms which compensate for illumination changes. Throughout this thesis, we describe several algorithms developed to effectively exploit the color information by reducing the variance of the color or focusing on estimating the target color.

1.3 Outline

Chapter 2 of this thesis explains how color is represented in an image.

In Chapter 3, we explore and evaluate detecting an object which does not emit light, without compensating for illumination. In order to reduce the variance of color, we use the HSV color space, which is less sensitive than the RGB color space. Also, kernel-based tracking is applied so that the object can be tracked in a sequence of images irrespective of slight object color changes. We also classify construction zone signs so that autonomous vehicles can understand road conditions and respond accordingly. We demonstrate these algorithms in a vehicle.

Chapter 4 extends the work from Chapter 3 in order to compensate for light sources when estimating colors. We understand how the color is changed based on the geometrical relationship between a light source and a surface of an object, and represent it as a probabilistic approach to estimate the color. We validate the improvement of the performance after compensating for a light source from different positions.

Chapter 5 describes detecting an object which has its own emitted light. In contrast to an object that reflects a light source, an object that emits its own light is not affected critically by the geometrical relationship between a light source and an object. Thus, we propose how to achieve consistent performance in traffic light state estimation and apply these techniques to a self-driving vehicle. Lastly, this thesis concludes with a summary and future directions.

Chapter 2

Background

Color is a very important cue in recognizing objects. Most people are capable of understanding what the color of an object is. Among many sensors, a camera has the capability of getting the color information. A camera has a lens and sensor array that capture the amount of light and produce an image. However, the images sometimes look different even though the same camera

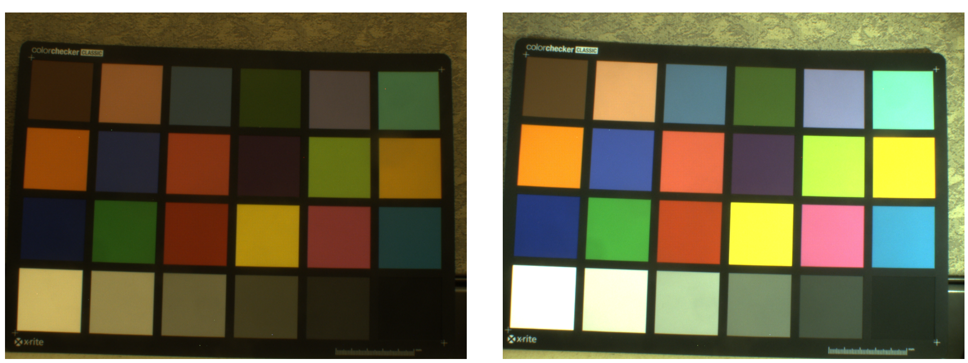


Figure 2.1: Images of a color calibration target using the same camera but different illumination conditions show dramatic difference in color.

is capturing the same object, as depicted in Figure 2.1. Why do the colors look different? We will examine how a camera produces the color. The following sections provide the basic concepts that are necessary to understand the research problems presented in this thesis. They are mostly based on Ebner [Ebner, 2007].

2.1 Color Theory

2.1.1 Radiance and Irradiance

In radiometry, *radiance*, L, is defined as the amount of light emitted or reflected by an object while *irradiance*, E, is defined as the amount of light falling on a surface.

$$L\left[\frac{W}{m^2 sr}\right] = \frac{d^2\Phi}{d\Omega dA} = \frac{d^2\Phi}{d\Omega dA_0 \cos\theta} \tag{2.1}$$

$$E\left[\frac{W}{m^2}\right] = \frac{d\Phi}{dA} \tag{2.2}$$

where Φ is the luminous flux, or the perceived power of light, and Ω is the solid angle, which is the area of the projection of a planar area A onto the unit hemisphere, as depicted in Figure 2.2.

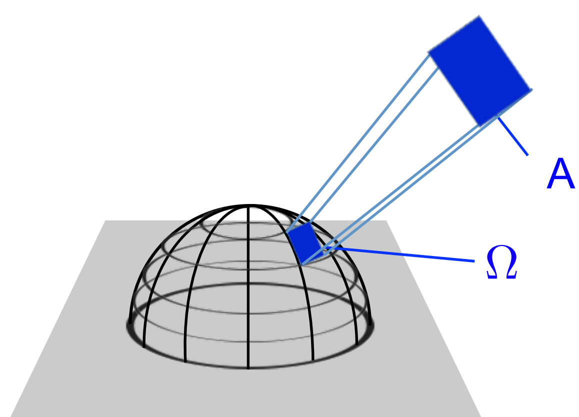


Figure 2.2: The solid angle is defined as the area of the projection of a planar area on the unit hemisphere.

Thus, the irradiance is related to the radiance, but there are various factors that affect this relationship, such as properties and locations of objects and light sources, and characteristics of sensors used.

Figure 2.3 demonstrates the relationship between the radiance from an object and the irradiance onto a camera. When the radiance from a small patch of the object passes the camera lens, the irradiance captured by the camera sensor can be described as follows:

$$E = \frac{dP_{cam}}{dA_{cam}} \tag{2.3}$$

where P_{cam} is the power of the light captured by a camera lens, and dA_{cam} is a small area of the camera sensor. The dP_{cam} can be further described by the solid angle of a camera, Ω_{cam}, the area of the object patch, dA_{obj}, and the angle between the object and the camera.

$$dP_{cam} = L\Omega_{cam} dA_{obj} \cos\alpha \tag{2.4}$$

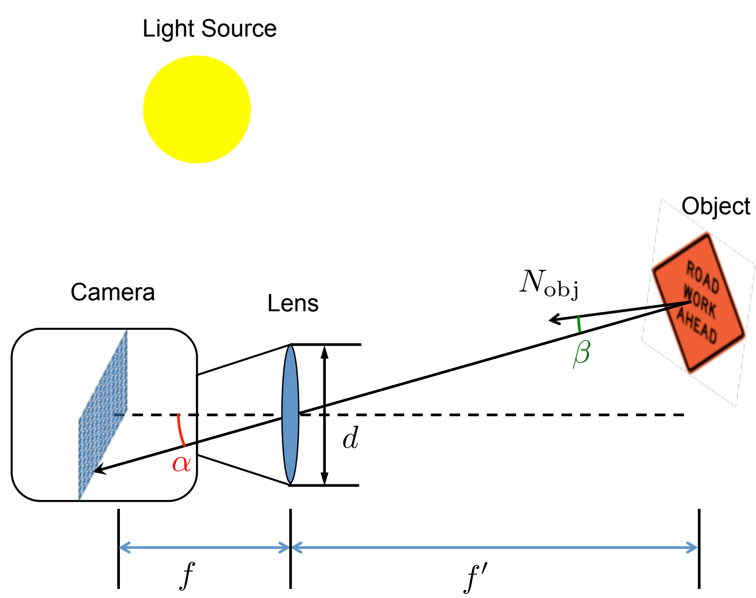


Figure 2.3: A sensor array measures an irradiance to produce a color, and the irradiance is proportional to the radiance of an object.

Given that a solid angle is defined as $\frac{A\cos\theta}{r^2}$ and the area of a camera lens as $\frac{\pi}{4}d^2$, the solid angle of the camera can be expressed as

$$\Omega_{cam} = \pi\frac{d^2}{4}\frac{\cos\theta}{r_{obj}^2} \tag{2.5}$$

Since the irradiance captured by the camera and the radiance from the object generate the same solid angle, we can equate the two solid angles from the camera and the object.

$$\frac{dA_{cam}\cos\alpha}{r_{cam}^2} = \frac{dA_{obj}\cos\beta}{r_{obj}^2} \tag{2.6}$$

r_{cam} and r_{obj} denote the distance from the sensor to the lens and that from the object to the lens respectively. By substituting equations 2.3 to 2.6 into equation 2.3, we finally come up with equation 2.7.

$$E = \pi L\frac{d^2}{4f^2}\cos^4\alpha \tag{2.7}$$

This equation shows that the irradiance, E, is proportional to the radiance, L, by the degree of an angle between the patch of the camera sensor and the center of the lens, the diameter of the lens, and the focal length of the lens.

2.1.2 Reflectance Models

In the previous section, we showed that the irradiance falling on the camera sensor is proportional to the radiance from the object. This radiance from the object varies based on the surface properties of the object, and this section describes how different surfaces of objects and different directions of light sources can change the radiance from an object.

The Bidirectional Reflectance Distribution Function (BRDF) is the most common model to express the relationship between the incoming light source (irradiance) and the reflected light (radiance). Figure 2.4 illustrates the concept of BRDF, and it is defined as

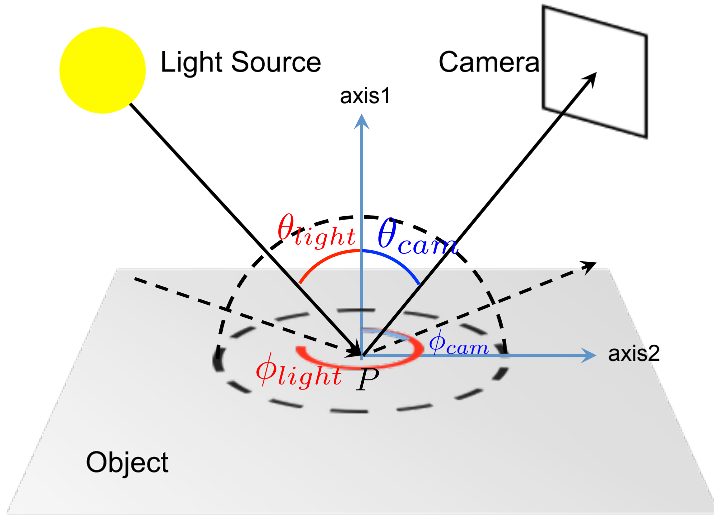


Figure 2.4: An incoming light source falling at a point *P* of an object is reflected in a reflected direction. The incoming light and the reflected light are parameterized by azimuth angles, ϕ, and elevation angles, θ.

$$f(\theta_{light}, \phi_{light}, \theta_{cam}, \phi_{cam}) = \frac{L_p(\theta_{cam}, \phi_{cam})}{E_p(\theta_{light}, \phi_{light})} \tag{2.8}$$

The BRDF varies based on different surface properties. Table 2.1 presents the overview of reflectance models based on different types of object surfaces.

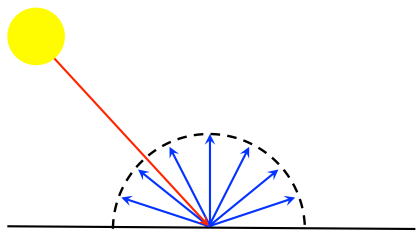
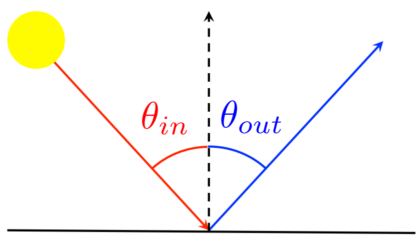
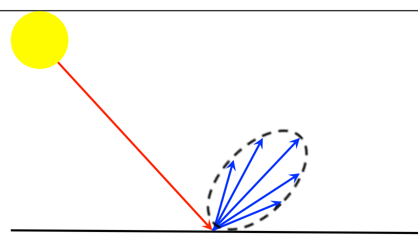
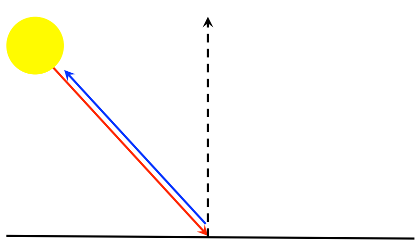
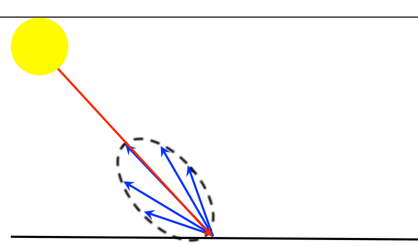
2.1.3 Radiance from Object

If we know the BRDF of the object, then we can calculate the radiance from the object. In order to calculate the radiance at a point P from the object, we need to integrate all the irradiance falling onto the hemisphere of that point, and it can be calculated as follows:

$$L_p = \int_{\Omega} f(\theta_{light}, \phi_{light}, \theta_{cam}, \phi_{cam}) E_p(\theta_{light}, \phi_{light})\, d\Omega \tag{2.9}$$

For some objects, such as traffic signals and turning signals of vehicles, which emit their own lights, it is important to consider additional radiance as shown in equation 2.10. This is sometimes called the *rendering equation*:

Table 2.1: Overview of reflectance models

Model name	Illustration	Description
Lambertian reflection		A Lambertian surface perfectly diffuses the incident light equally in all directions regardless of the angle of the incoming light. So, if a surface is Lambertian, the BRDF becomes a constant
Mirror reflection		When an object has a perfect mirror-like surface, the angle of incoming light is the same as the angle of reflected light
Glossy reflection		A glossy surface reflects the incident light like a mirror, but the reflected light spreads
Perfect retro reflection (corner cube)		When the direction of incoming light and the direction of reflected light are exactly opposite, we call this reflectance model retro-reflectance
Retro reflection		A retro-reflectant surface reflects the incident light like a corner cube, but the reflected light spreads.

$$L_p = L_e(x, w_0) + \int_{\Omega} f(\theta_{light}, \phi_{light}, \theta_{cam}, \phi_{cam}) E(\theta_{light}, \phi_{light}) \, d\Omega \tag{2.10}$$

where $L_e(x, w_0)$ is the intensity of the emitted light.

2.1.4 Sensor Response

The camera integrates the irradiance to produce an image. Specifically, the camera captures certain wavelengths of the irradiance. Its response is integrated over all possible wavelengths in order to obtain the output of a sensor:

$$I_c(p_i) = \int E(\lambda, p_i) S_c(\lambda) \, d\lambda \tag{2.11}$$

where $E(\lambda, p_i)$ is the irradiance of wavelength λ falling onto a patch on the sensor array located at p_i, and $S_c(\lambda)$ is a response function of a sensor with $c \in \{\text{R,G,B}\}$. As shown in equation 2.7, the irradiance at a camera is proportional to the radiance from an object. Because all the parameters, such as diameter of lens and focal length in equation 2.7, can be fixed, we can substitute equation 2.7 into equation 2.11 and simplify as follows:

$$I_c(p_i) = \int L(\lambda, p_{obj}) S_c(\lambda) \, d\lambda \tag{2.12}$$

As seen in the previous sections, the radiance model, $L(\lambda, p_{obj})$, is a function of a reflectance model and the irradiance, so we can use this radiance model to calculate an intensity value. If we assume that the object has a Lambertian surface, BRDF is a constant. Also, its irradiance is proportional to the radiance from the light source and the angle between the normal vector of the object and the light source, i.e., $E_{obj} = L \cos \alpha$.

$$I_c(p_i) = k \cos \alpha \int L(\lambda) S_c(\lambda) \, d\lambda \tag{2.13}$$

Assuming that the response function of a sensor can be approximated by a delta function, we have $S_c(\lambda) = \delta(\lambda - \lambda_c)$. Finally, we obtain the following equation, which measures the intensity for each channel:

$$I_c(p_i) = k \cos \alpha L(\lambda_c) S_c(\lambda_c) \tag{2.14}$$

2.2 Related Work

We examined how a camera produces color by measuring an amount of light. As explained before, an amount of light from an object varies based on various factors, such as the amount of light source, the material of the object, and the geometrical relationship between the light source, the object, and the camera. Thus, researchers have been conducting experiments in order to remove the effect of these factors. The following sections provide literature reviews on: 1) an illumination-invariant image, and 2) a color constancy.

2.2.1 Illumination-invariant Image

An illumination-invariant image has a single channel similar to a grayscale image. However, an illumination-invariant image reduces illumination variation, while a grayscale image does not. Table 2.2 summarizes the existing works on illumination-invariant features or images. Even though researchers have been developing different applications, the basic concept of the illumination-invariant image comes from [Finlayson et al., 2004] and [Finlayson et al., 2006].

Finlayson *et al.* proved that an illumination image could be generated under the assumption that the illumination can be approximated by Planck's law and objects in images have a Lambertian reflectance model. They showed that the value of each channel (red, green, and blue) is a

Table 2.2: Overview of illumination-invariant features or images

Authors	Application
[Marchant and Onyango, 2000]	Classify vegetation and background
[Finlayson et al., 2004]	Remove shadows
[Finlayson et al., 2006]	Remove shadows
[Alvarez et al., 2008]	Road segmentation
[Corke et al., 2013]	Dealing with shadows for localization
[Maddern et al., 2014]	Localization by combining illumination-invariant images and raw RGB images

function of the wavelength and other constant parameters. By dividing each of the red and blue channel by the green channel, they could derive chromaticities. In these chromaticities, the colors of an object that vary due to illumination formed a straight line. Then, a perpendicular axis to this line was calculated in order to generate an illumination-invariant image by projecting all the color variation onto this perpendicular axis. They found this perpendicular axis by minimizing entropy and used this algorithm to remove shadow effects in the color image.

Marchant *et al.* classified vegetation and background by using the illumination-invariant image. They first calculated the histogram of the illumination-invariant image, and found two different peaks which represented two different classes.

Alvarez *et al.* used the illumination-invariant image to segment the road for ADAS. After generating the illumination-inavriant image, they applied a seeded region-growing algorithm to segment the road. They chose seeds randomly near the bottom of the image. Then, they calculated histograms using those seeds and their surrounding regions to classify whether those seeds belonged to the road or not. Once those seeds were classified as road, the same procedure was repeated until pixels were classified as background.

Corke *et al.* localized the autonomous vehicle by using the illumination-invariant image. They drove the same route several times under different weather conditions and times of day. Then, they created illumination-invariant images along the route using one dataset as a reference and tried to match the other datasets with a reference to estimate where the vehicle was.

Maddern *et al.* also used the illumination-invariant image to localize the autonomous vehicle. They collected the same route for every hour of a 24-hour period. They showed that the invariant images from daytime and nighttime were different due to light sources. The dominant light source during daytime was the Sun, while the dominant light source at nighttime was a street lamp and a headlight. So, they used illumination-invariant images during daytime and raw RGB images during nighttime to improve the performance.

2.2.2 Color Constancy

Buluswar *et al.* tried to solve the challenge of the color constancy algorithm for autonomous vehicles outdoors [Buluswar and Draper, 1998]. They learned about the variations in the apparent color of objects with respect to existing models of daylight and surface reflectance by using a Multivariate Decision Tree from the training samples. The Multivariate Decision Tree recursively divided the feature space with hyperplanes so that it performed a piecewise-linear approximation of the region in color-space consisting of the target color.

Finlayson *et al.* calibrated a camera by using 4 sensors to represent an invariant image with chromaticities to remove effects from shadows and lighting [Finlayson and Drew, 2001]. They considered band-ratio chromaticities from 4 different sensor responses to make an invariant im-

Table 2.3: Overview of color constancy algorithms

Authors	Light source	Surface assumption	Method
[Buluswar and Draper, 1998]	outdoor	none	Learning-based
[Finlayson and Hordley, 2001]	indoor	Lambertian & Specularity	Physics-based
[Tsin and Collins, 2001]	outdoor	Lambertian	Learning-based
[Sridharan and Stone, 2007]	indoor	none (Floor and Wall)	Learning-based
[Gehler et al., 2008]	indoor / outdoor	Lambertian	Learning-based
[Ratnasingam and Collins, 2010]	outdoor	Lambertian & Specularity	Physics-based
[Gijsenij and Gevers, 2011]	outdoor	none	Physics-based
[Ratnasingam et al., 2013]	outdoor	Lambertian & Specularity	Physics-based

age.

Tsin *et al.* developed a Bayesian color constancy algorithm [Tsin and Collins, 2001]. They modeled factors of the image formation process, the sensor noise distribution, geometry, material types, and illuminant spectral distributions. Then, these prior distributions were formed through a training process so that they could successfully classify and segment an image by light source and material types.

Sridharan *et al.* learned a different color model by detecting how the illumination is changed [Sridharan and Stone, 2007]. They separated their algorithms into 3 different phases: when, how, and what to learn. They first determined the change in the illumination, and compared the color of the known object, and learned a new color map for a mobile robot to localize.

Gehler *et al.* improved a Bayesian color constancy algorithm by learning more precise priors for illumination and reflectance [Gehler et al., 2008]. They collected their own data with accurate illumination labels to increase the performance of a Bayesian color constancy algorithm. By using labeled illumination, they were able to remove any bias because the unknown illumination might have a bias due to the scale factor. They also chose the illumination prior as the empirical distribution of the training illuminations rather than the uniform illumination.

Gijseniji *et al.* improved the performance of the color constancy algorithms by fusing various existing color constancy models [Gijsenij and Gevers, 2011]. They used natural image statistics to select the best color constancy model automatically. Given the training samples, they computed all the image statistics, and applied a mixture-of-Gaussian classifier and EM algorithm to the training samples to find the weights (parameters) of the model. Using these parameters, they applied the mixture-of-Gaussian classifier to the testing images to get the maximum posterior probability.

Ratnasingam *et al.* improved their algorithms by using the knowledge of correlated color temperature of the Sun [Ratnasingam et al., 2013]. By including the solar elevation information, they successfully improved the algorithms from [Finlayson and Drew, 2001] and [Ratnasingam and Collins, 2010]. They figured out that the correlated color temperature of the Sun didn't change much when the solar elevation was higher than 20 degrees. So, they divided the daylight into 3 different phases: after sunrise (morning), in the middle of the day (midday), and before sunset (evening). Then, they used different parameters for each phase to improve the previous algorithms.

We reviewed how other researchers have been conducting research to remove variations caused by illumination. An illumination-invariant image is capable of reducing the variation of intensity and shadow effect, but it loses color information. Most color constancy algorithms estimate illumination and reflectance to achieve color constancy. Most of the researchers, however, made assumptions that the surface is Lambertian [Buluswar and Draper, 1998], [Tsin and Collins, 2001], [Sridharan and Stone, 2007], [Gehler et al., 2008], [Gijsenij and Gevers, 2011], [Bianco and Schettini, 2012], [Ratnasingam et al., 2013]. This assumption simplifies the sensor equation so that the geometric relationship between a light source and a surface is linear. However, human-made objects, specifically those on roads, such as vehicles, road signs, and traffic signals, are not Lambertian, thus this assumption is not suitable. Furthermore, road signs are retro-reflectant, and none of the previous researchers, to the best of our knowledge, has accounted for the appearance of a retro-reflectant surface. Also, the previous research tried to estimate illumination indoors and outdoors to transfer color accordingly. However, in our case, the application domain is an autonomous driving assistance system or a self-driving vehicle, so the dominant light source is the Sun. Thus, in this thesis, we investigate how to estimate the illumination with a better understanding of the Sun by considering its elevation and azimuth to predict the color of an object based on this Sun model.

Chapter 3

Traffic Sign Recognition

3.1 Related Research

Many researchers have proposed road sign recognition systems. The critical part of road sign recognition is detection. Work in road sign detection can be categorized into color-based and shape-based methods, Table 3.1, because road signs have a standard color and shape. Once detected there are many classification techniques that can be applied [Ciresan et al., 2012], [Zeng et al., 2015].

Most of the color-based sign detection methods use color to segment candidate sign regions as an initial step. Several researchers investigated color values based on the data that they collected, while others, including us, used machine learning techniques to obtain the optimal thresholds of target color [Lopez and Fuentes, 2007]. Most researchers who used color-based sign detection methods mentioned the sensitivity of the color based on the illumination, and used hue-saturation-intensity (HSI) or hue-saturation-value (HSV) color spaces. HSI and HSV are less sensitive to lighting or brightness variation than RGB. Hue and saturation represent color information while intensity or value represents illumination information. By using only hue and saturation, the illumination variation can be reduced. Gao *et al.* investigated color based on the temperature of the light source [Gao et al., 2006], [Gao et al., 2008], and Broggi *et al.* analyzed the color of traffic signs in the YUV space in order to reduce color variation [Broggi et al., 2007]. After segmentation, they post-processed candidate regions in order to detect signs. Even though researchers tried to reduce the variation of the color by transforming RGB to other color spaces, they didn't specifically analyze the relationship between the surface model of a road sign and the illumination.

Shape-based sign detection methods, on the other hand, usually do not find a candidate region. Instead, they use a sliding-window technique, where a window moves along the image to look for a predefined shape. The researchers who use shape-based sign detection methods usually utilize the gradient of the image in gray scale. However, most of them only focused on speed-limit signs or specific shapes of the signs (e.g. circular or triangular signs in Europe).

It is important for autonomous vehicles to smoothly adjust driving behavior in unexpected situations. When driving, unexpected roads are mostly indicated by workzone signs. Thus, we investigated how to improve detecting workzone signs by developing color-based sign detection given that all of the workzone signs have an orange color.

3.2 Sign Detection

The shape and color of road signs are strictly regulated by local and national traffic authorities. For example, the background color of all workzone signs is orange. However, it is still challenging to identify an orange-colored pixel in an image because aging, wearing, weather, and lighting often cause color variations. To reduce the variation of orange colors, we first transformed the RGB color space to the HSV color space [Wandell, 1993]. As depicted in Figure 3.2, orange colors in the HSV color space have less variation than in the RGB color space, and Table 3.2

Table 3.1: Overview of road sign detection methods.

	Paper	Color	Shape	Geometry
Color	[Miura et al., 2000]	manual	edges	X
	[Bahlmann and Zhu, 2005]	O	Haar wavelets	O
	[Lafuente-Arroyo et al., 2005]	manual	distance to border	X
	[Vazquez-Reina et al., 2005]	manual	distance transform	X
	[Siogkas and Dermatas, 2006]	Otsu's threshold	symmetry detection	X
	[Gao et al., 2006]	manual (hue, chroma)	extract edge histogram	X
	[Ruta et al., 2007]	manual	radial symmetry transform	X
	[Broggi et al., 2007]	manual with chromatic equalization	simple shape matching	X
	[Maldonado-bascón et al., 2007]	manual on HSI	distance to border	X
	[Gao et al., 2008]	CIECAM	X	X
	[Gil Jiménez et al., 2008]	manual	shape classification using FFT	X
	[Ruta et al., 2010]	manual	radial symmetry transform	X
	[Lafuente-Arroyo et al., 2010]	histogram & achromatic	distance to border	X
	[Timofte and Prisacariu, 2011]	manual	MSER & hough	O
	[Greenhalgh and Mirmehdi, 2012]	manual	MSER	X
Shape	[Barnes and Zelinsky, 2004]	X	radial symmetry transform	X
	[Moutarde et al., 2007]	X	edges	X
	[Keller et al., 2008]	X	radial symmetry transform	X
	[Meuter, 2008]	X	radial symmetry transform	O
	[Nunn et al., 2008]	X	O	O
	[Pettersson et al., 2008]	X	edge histogram	X
	[Muller-Schneiders et al., 2008]	X	radial symmetry transform	X
	[Loy and Barnes, 2009]	X	radial symmetry transform	X
	[Overett and Petersson, 2011]	X	HOG	X

explains the variation of orange colors quantitatively. We scaled RGB and HSV values into the range of 0 to 1 and calculated the variations. The variations of hue and saturation are less than 0.014 while those of red, green, and blue are twice as big as that. After converting to the HSV color space, we additionally developed a Bayesian inference framework to detect orange color more robustly as in Equation 3.1:

$$P(\mathtt{sign}|\mathbf{X}) = \eta P(\mathbf{X}|\mathtt{sign}) P(\mathtt{sign}) \tag{3.1}$$

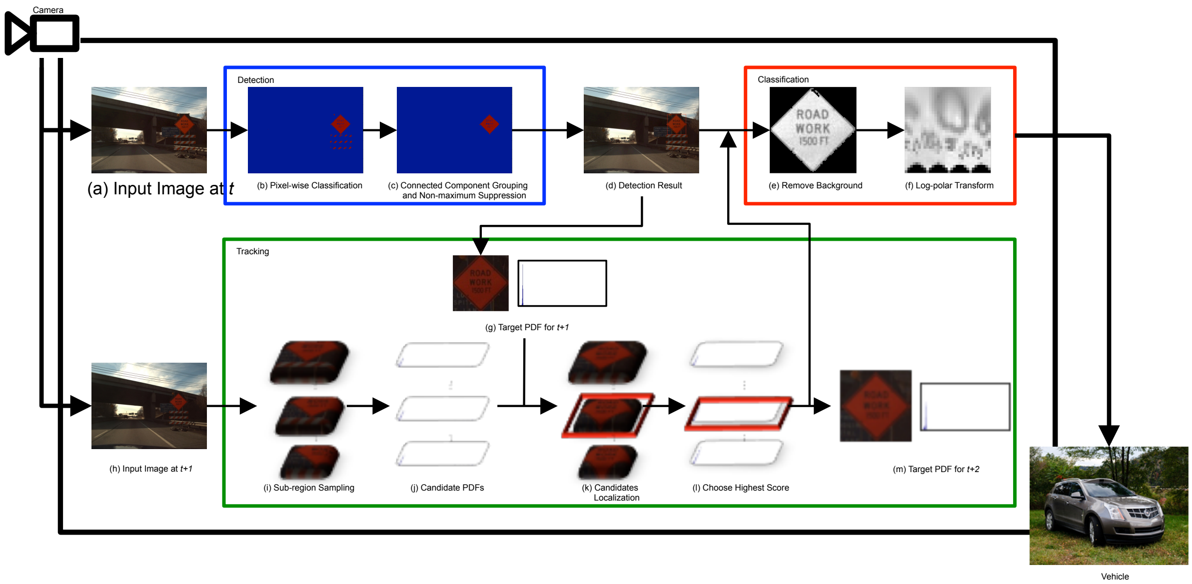


Figure 3.1: Schematic overview

where $\mathbf{X}$ is the total number of m-dimensional pixels and η is a normalizer for the posterior distribution. We obtained the prior probability distribution, $P(\mathtt{sign})$, from the ground truth of our training data, and used AdaBoost [Freund and Schapire, 1996] to learn the likelihood function, $P(\mathbf{X}|\mathtt{sign})$, of a given pixel as part of the target sign. The training data contained 29 images of positive data and 37 images of negative data. 19 images of those positive data were chosen from the web while 10 of them were from our collected data. Also, 22 images of those negative data were from the web while 15 of them were from our collected data. Those images were collected in various weather conditions, such as sunny, overcast, and rainy. A set of

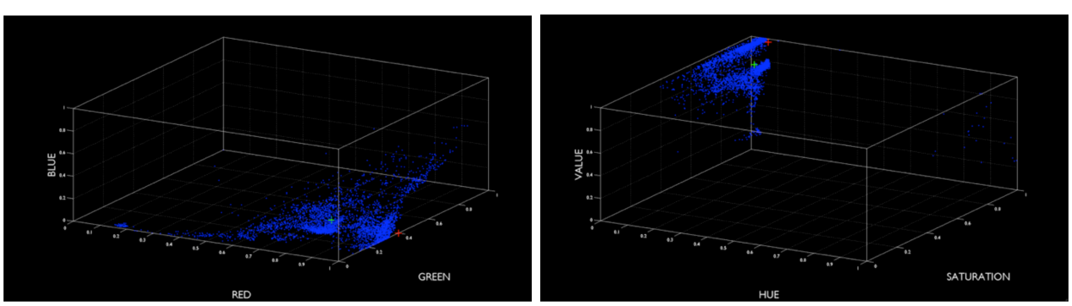


Figure 3.2: Orange colors in RGB space and HSV space

Table 3.2: Variations of orange color in RGB color space are bigger than those in HSV color space. RGB and HSV are scaled into the range of 0 to 1 to calculate these variations.

(a) RGB

	Variation
Red	0.052
Green	0.0311
Blue	0.0112

(b) HSV

	Variation
Hue	0.0076
Saturation	0.0137
Value	0.0224

weak-classifiers and their weights were trained by these data:

$$P(\mathbf{X}|\texttt{sign}) = \texttt{mode}\left(\cup_j g\left(f\left(\mathbf{x}_j|\texttt{sign}\right)\right)\right)$$

$$f(\mathbf{x}_j|\texttt{sign}) = \sum_{i=1}^{H} \alpha_i h_i(\mathbf{x}_j)$$

where $\mathbf{x}_j$ is a 2-dimensional color vector which consists of hue and saturation from pixels in an image, $\mathbf{x}_j \in \mathbf{X}$, g is a function to convert the binary output of AdaBoost into a probabilistic output [Friedman et al., 2000], $g(f(y)) = \frac{exp(f(y))}{exp(f(y))+exp(-f(y))}$, H is the number of weak classifiers, h_i represents a weak classifier and α_i is its weight. We obtained 97% pixel-wise detection accuracy using this color classifier.

This color classifier was executed on every pixel to assign a probability of whether the pixel is part of an orange workzone sign or not. Our sign detector runs a connected-component grouping algorithm to identify orange blobs and uses non-maximum suppression to select the largest bounding box when the width or height is bigger than 30 pixels. To detect a regulatory, rectangular workzone sign which includes two colors (orange at the top and white at the bottom), we heuristically increased the bounding box based on the aspect ratio.

3.3 Sign Tracking

In this section, we use similar notation to [Comaniciu et al., 2003] with appropriate modifications when necessary. The bounding box of a potential workzone sign from our sign detector is given as an input to our sign tracking component. Since the color variation between two consecutive frames is small, we represented the potential workzone sign, the target, as a probability density distribution by computing a histogram in hue-saturation-value (HSV) color space. We chose HSV color space since it is less sensitive to illumination variation when compared to red-green-blue (RGB) color space. In particular, only hue and saturation values were quantized into $n_h \times n_s$ bins, where n_h and n_s are the bin numbers of hue and saturation, respectively. To avoid underfitting or overfitting, we chose both n_h and n_s as 20. To represent the target model consistently with various sizes of signs, we normalized the pixel coordinates. Furthermore, to focus on the appearance of a sign rather than the background, we utilized a kernel, K, which combines an Epanechinikov profile [Comaniciu et al., 2003] and a *mask* of the detected sign. In this kernel, the pixel coordinates closer to the center get larger weights and those within *mask* get weights of $K(\mathbf{p}) = c_k\,(1 - \mathbf{p}) \cdot$ *mask*. Fig. 3.3 illustrates this kernel. Finally, we were able to calculate the probability density distribution of the target model, $\mathbb{T} = \{t_b\}_{b=1\cdots n_h \times n_s}$, on normalized pixel coordinates, $\mathbf{p}$, as

$$\begin{aligned} t_b &= t_{n_s \cdot (j-1)+k} \\ &= C \sum_{i=1}^{n} K\left(\|\mathbf{p}_i\|^2\right) H(\mathbf{p}_i, h_j, s_k) \end{aligned} \tag{3.2}$$

where

$$H(\mathbf{p}_i, h_j, s_k) = \begin{cases} 1 & \mathtt{hue}(\mathbf{p}_i) \in h_j \cap \mathtt{sat}(\mathbf{p}_i) \in s_k \\ 0 & otherwise \end{cases}$$

and C is a normalization constant.

We also calculated candidates, $\mathbf{c}(\mathbf{z})$, to localize in the subsequent frame, where $\mathbf{z}$ is the new center. We could calculate candidates same as the target except the new normalized pixel coor-

(a) (b) (c)

Figure 3.3: Kernel functions. (a) the output of the detected sign (b) a mask of shape. In order to reduce the background effect and concentrate on the target color, an Epanechinikov kernel and the *mask* of shape from the detection output are combined. (c) illustrates this combined kernel.

dinate, $\mathbf{p}_i^{\mathtt{new}}$ based on $\mathbf{z}$, because the location of the sign in the subsequent frame shifts. Also, the size of the bounding box increases as the vehicle gets closer to the sign. Thus, to increase our performance, we applied five different sizes of kernels (0%, 2%, 5%, 7%, and 10% increased kernels) and chose one as the tracking result. $\mathbf{c}(\mathbf{z})$ is calculated by Equation 3.2 with $\mathbf{p}_i^{\mathtt{new}}$.

The distance between these two probability density distributions is minimized when the candidate model is matched to the target model. Minimizing the distance can be interpreted as maximizing the similarity, and we chose the Bhattacharyya coefficient [Comaniciu et al., 2003], $B(\mathbf{t}, \mathbf{c}(\mathbf{z}))$, to measure the similarity, where $B(\mathbf{t}, \mathbf{c}) = \sum_{b=1}^{n_h \times n_s} \sqrt{t_b \cdot c_b}$. Using linear approximation around $\mathbf{z}$, we were able to use the mean-shift algorithm [Comaniciu et al., 2003] to find the mode of

$$\frac{C_s}{2} \sum_{i=1}^{n} w_i K \left(\left\| \frac{\mathbf{z} - \mathbf{p}_i^{\mathtt{new}}}{\mathbf{s}} \right\|^2 \right) \tag{3.3}$$

where w_i is

$$w_i = \sum_{b=1}^{n_h \times n_s} \sqrt{\frac{t_b}{c_b(\mathbf{z})}} H(\mathbf{p}_i^{\mathtt{new}}, h_j, v_k)$$

We found the maximum value of 3.3 by the gradient. If the Epanechnikov kernel is used, the gradient of Equation 3.3 will be a weighted summation as the standard mean-shift algorithm. In our case, though, the mean-shift algorithm is represented as

$$\mathbf{z}^{\texttt{new}} = \frac{\sum_{i=1}^{n} \mathbf{p}_i^{\texttt{new}} w_i \cdot \mathit{mask}}{\sum_{i=1}^{n} w_i \cdot \mathit{mask}} \tag{3.4}$$

which is still a dot product that has low computational cost. The mean-shift algorithm was repeatedly executed until the error was less than the predefined threshold or the number of iterations was less than the maximum iteration number, which was less common. The center position was updated in every iteration by 3.4. Once all of the candidates converged, we chose the final output which had the highest the Bhattacharyya coefficient and provided it to our sign classification component.

3.4 Sign Classification

A bounding box from both our sign detection and tracking as a potential workzone sign was given as an input to our sign classification component. The goal of this research is to recognize temporary changes in workzones, so it is important to recognize the signs by classifying them. Thus, we chose nine different workzone signs as target classes which indicate either workzone bounds or temporary changes The remaining workzone signs were assigned to another class. Table 3.3 explains the target classes in detail.

There are three major issues when classifying the detected signs. First, the intensities of the detected signs can vary due to the environment. Second, the dimensions of the detected signs vary; the dimension of a sign looks bigger when the sign is closer to the camera. Third, it is almost impossible to see the canonical shape of a workzone sign because of the perspective projection into an image. Also, signs can be slightly tilted or rotated over time or even when installed. In order to effectively handle these issues, our sign classifier first normalized the intensity of the detected sign, and then transformed it into a log-polar image. This log-polar transform

Table 3.3: Nine different workzone signs are chosen as target classes. Each cell contains an example image, a content, and the number of samples for the training.

		
Workzone begins	Workzone ends	Road work warning
61	42	86
		
Speed limit change	Lane shift leftward	Lane shift
75	26	36
		
Left lane closed	Left lane closed	Lane shift rightward
19	17	43

extracts more intensity values near the center of a sign image where the distortions are relatively small, and then sparsely collects values from sign image boundaries where the geometric distortions are large. Also, this log-polar transformed image always have ρ by θ resolution, because the detected sign image was converted into a log-polar image based on these two parameters: ρ is the distance between sampling bins and the centroid, and θ is the rotation angle of sampling bins in counterclockwise. Finally, our sign classifier produced the same length of a column vector, $|\rho \times \theta| \times 1$ (e.g., a combination of the parameters, $\rho = 32$ and $\theta = 32$, resulting in 1024×1), for every sign.

Given that the column vector was still high-dimensional compared to the training data, we reduced the original dimension using principal component analysis (PCA). We then built an eigenspace from the training data and projected a testing sign image in the log-polar coordinate space onto this eigenspace. The 10 eigenbases represented more than 93% of the total variance of perfectly labeled input data. We used a support vector machine (SVM) to classify the signs among those eigenbases.

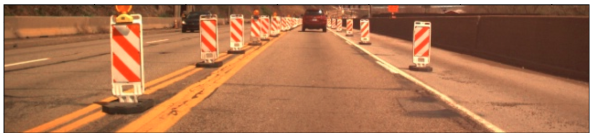
3.5 Workzone Channelizer Detection

Not only the workzone signs, but also channelizers are important indicators of workzones. There are several different channelizers, but we only considered the vertical panels, because the vertical panels are more commonly used channelizers in highway workzones for temporary road geometry changes. In our work, the term ”channelizers” means ”the vertical panel”.

To verify temporary road geometry changes, channelizers should be detected. Channelizers are comprised of diagonal strips in alternating white and orange color. To utilize these diagonal patterns, our channelizer detection first finds diagonal edges. In order to speed up the algorithm, we utilized simple geometry. The channelizers are always standing on the ground and they are 24 inches tall. Due to this geometry, the region-of-interest (ROI) for channelizers was assumed to be located under the vanishing point.

Two different filters were applied to this ROI of the image to distinguish these diagonal patterns. After removing noise based on predefined thresholds, potential channelizer regions were determined by searching a channelizer pattern from left to right, and then from top to bottom. In the end, our channelizer detector allowed the potential bounding boxes as the output only if there were orange blobs inside the bounding boxes.

We took three steps to detect channelizers within the ROI by considering their characteristics



(a) Input

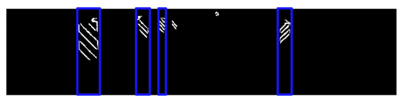
(b) Diagonal Edge Detection

(c) Orange Color Detection

(d) Output

Figure 3.4: Channelizer Detection procedure

(a) Diagonal Edge



(b) Window searching from the left to the right

(c) Window searching from the top to the bottom

Figure 3.5: Diagonal Edge Detection procedure

of the diagonal strips and the orange color. First, as shown below, we applied simple filters that enhanced the diagonal edges. Because the diagonals could be represented in two different directions (i.e., upper left to lower right vs. upper right to lower left), we applied two different filters to an image:

$$\begin{aligned} I_{diag1} &= I * g \\ I_{diag2} &= I * g' \\ I_{diag} &= \sqrt{I_{diag1}^2 + I_{diag2}^2} \end{aligned} \qquad where \quad g = \begin{bmatrix} -1 & 0 \\ 0 & 1 \end{bmatrix}$$

where I is an image, g is a diagonal filter and g' is a horizontally flipped filter of g.

Since the output, which is the total magnitude of two different diagonal edge images, contains noise, we further removed the noise by setting predefined thresholds. Secondly, after threshold-

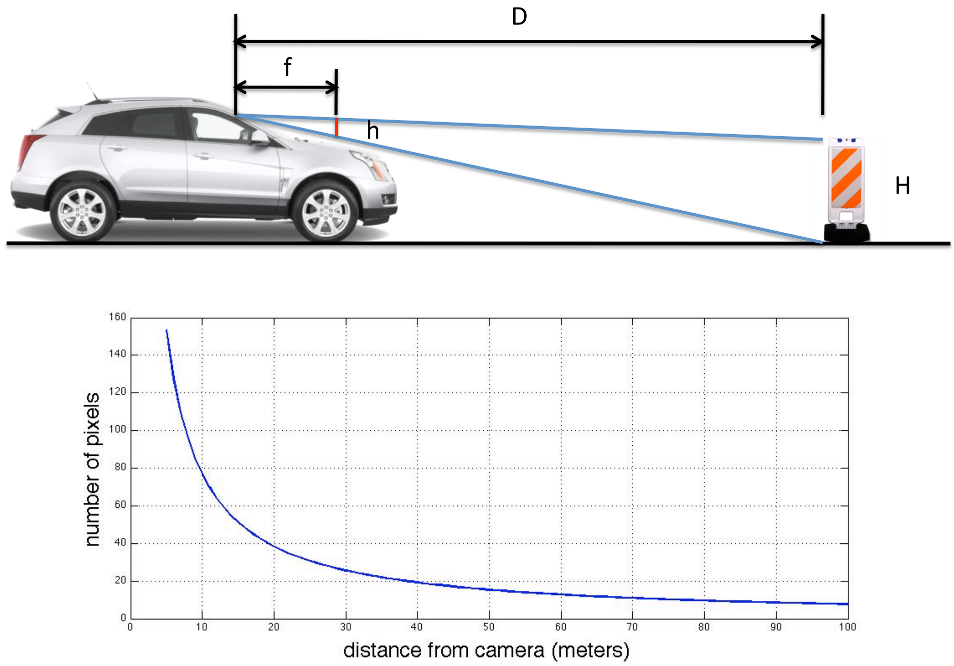


Figure 3.6: The graph of the distance from the camera and the number of pixels. Viewed best in color.

ing (Figure 3.5(a)), we used a virtual vertical line to scan the image from left to right to detect the patterns of the channelizers (Figure 3.5(b)). After localizing possible channelizers, we lastly used the virtual horizontal line to scan them from top to bottom to provide bounding boxes (Figure 3.5(c)). Not only the diagonal edge, but also the orange color was an important feature for the channelizer. To reduce false positives, we considered if the bounding box contained the orange color.

After detecting channelizers, it is important to know where the channelizers are located relative to a vehicle. We can estimate the depth information when we know the actual size of the detected object. Since the actual height of a channelizer is 24 inches, which is a government standard, the depth information (D) of a channelizer can be estimated by

$$D = \frac{H \cdot f_y}{h} \tag{3.5}$$

where H is an actual height, f_y is a focal length of $y - axis$, and h is a number of pixels. Figure 3.6 shows the relationship between the height and the distance from the camera.

After estimating the depth information, we calculated the lateral and longitudinal distances. Assuming that the skew of the camera is zero, the lateral distance (x) and the longitudinal distance (y) are

$$x = \frac{u - u_0 \cdot D}{f_x} \tag{3.6}$$

$$y = \frac{v - v_0 \cdot D}{f_y} \tag{3.7}$$

where u and v are the position in image coordinates, and u_0, v_0, and f_x are intrinsic parameters of camera which represent the center of the camera and the focal length of $x - axis$, respectively.

3.6 Experiments

3.6.1 Data Collection

We collected video sequences in 640 by 480 resolution at 15 fps under various environmental conditions. We labeled 5 video sequences with ground truth of position and target class. These testing data contained a sequence of a nominal highway, a workzone, and another nominal highway under various weather conditions. The top row of Table 3.4 explains the total number of frames, the number of images in which workzone signs appear, and the weather conditions.

3.6.2 Evaluation

Our experiments measured the performance of detection and classification by executing two different detection types, 'detection only' and 'detection and tracking,' to show the improvement of our workzone recognition system when applying the tracking component under various weather conditions. 'Detection only' executed the sign detector in every frame, while 'detection and tracking' executed the sign detector until it satisfied the predefined conditions. Then, the sign tracking was executed in the remaining frames until a sign disappeared. The classifier predicted a target class of a sign once the sub-region image was generated by either the detection or tracking.

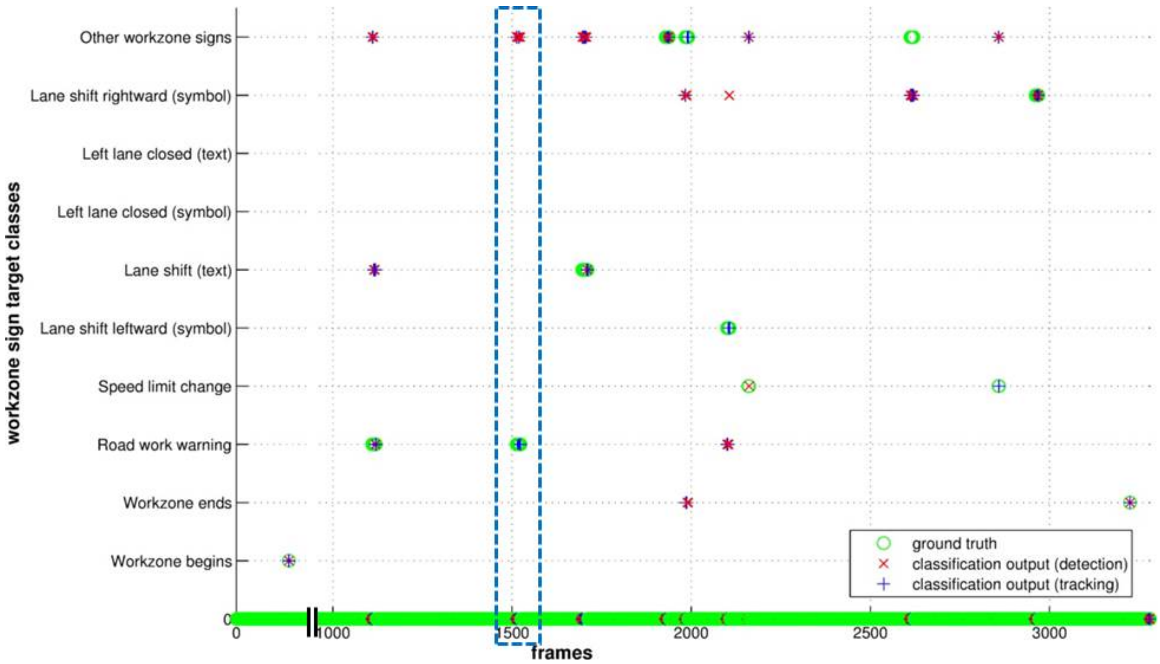
To evaluate the performance of our localization, we used the metrics used for PASCAL object detection challenges [Ponce et al., 2006]. An output bounding box, o_i, was considered as a potential match to the ground truth bounding box, g_i, in a given image frame, i, if an overlapping area was greater than a predefined value, $\tau < \frac{Area(o_i \cap g_i)}{Area(o_i \cup g_i)}$. Once a potential match was found in a given image, sign localization performance could be further analyzed by measuring the following: *precision*= $\frac{Area(o_i \cap g_i)}{Area(o_i)}$ and *recall*= $\frac{Area(o_i \cap g_i)}{Area(g_i)}$.

Figure 3.7 shows one of the results (i.e., video dataset, E) in detail. The x-axis represents the number of image frames ordered in time, and the y-axis represents the workzone sign target classes. Whenever a classification output was matched to the ground truth labeling, we counted it as a correct classification. We further analyzed the classification result with respect to the performance of our localization. Figure 3.7 (b) magnifies the dashed rectangle in Figure 3.7 (a). "Road work warning" signs appeared 12 times before they disappeared from the camera's field of view. As shown in Figure 3.7, the recall of detection fluctuated while that of tracking was stable. We treated our localization method as missing a sign when its PASCAL measurement was less than 0.65. Our detector missed at frame 1515 and 1517. On the other hand, our tracking module did not miss any signs during these frames and provided bounding boxes with high precision and recall, which led to a better classification performance.

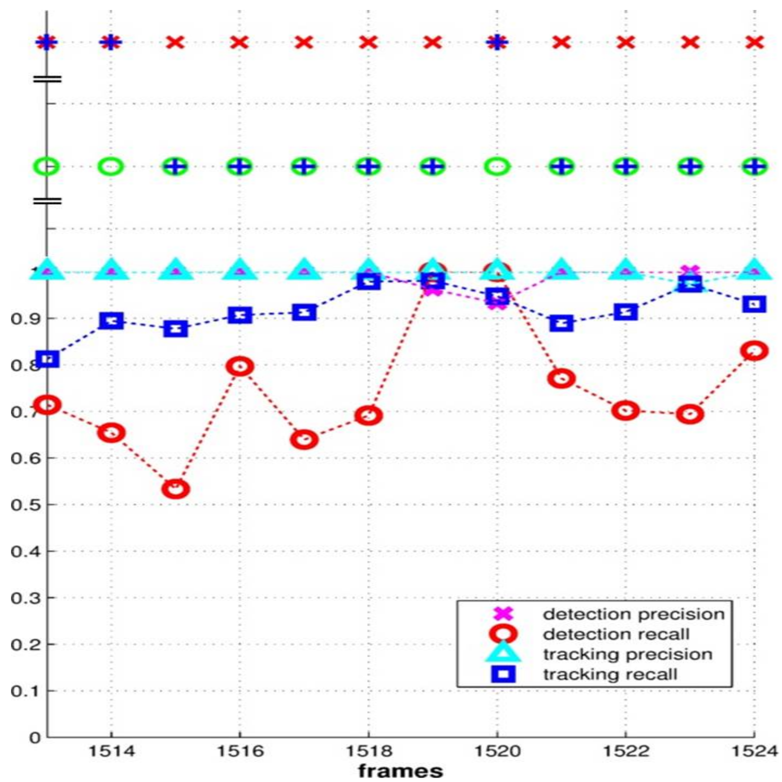
We evaluated the performance of both localization and classification of our workzone sign recognition system. For the localization performance, we counted the number of signs that were missed by our sign detection module but detected by our sign tracking module. We calculated its

Table 3.4: Results of our system performance tests under different driving conditions. Cells which contain two rows represent the results from the tracking (top row) and those from the detection only (bottom row).

		A	B	C	D	E
Test video	Number of images	3305/445	4232/646	874/255	3148/401	3280/445
	Season	Winter	Winter	Spring	Spring	Spring
	Weather	Overcast	Overcast	Sunny	Sunny	Rainy
Detection	Number of tracked signs	97	117	49	152	98
	Number of covered signs	4	11	3	20	13
	Precision / Recall	**0.999** / 0.879 0.957 / **0.905**	**0.988** / **0.870** 0.967 / 0.823	**0.989** / **0.885** 0.967 / 0.874	**0.962** / 0.903 0.854 / **0.925**	**0.983** / **0.882** 0.956 / 0.832
Classification	Workzone Begins	o/1.0/0.235 o/1.0/0.235	o/**0.667**/1.0 o/0.333/1.0	o/0.125/1.0 o/**0.250**/1.0	o/0.714/0.556 o/**1.0**/**1.0**	o/1.0/1.0 o/1.0/1.0
	Workzone Ends	o/1.0/1.0 o/1.0/1.0	o/1.0/1.0 o/1.0/1.0	o/1.0/1.0 o/1.0/1.0	o/1.0/1.0 o/1.0/1.0	o/1.0/0.2 o/1.0/**0.25**
	Road work	o/**0.867**/1.0 o/0.533/1.0	o/**0.585**/0.774 o/0.366/0.714	o/**0.688**/1.0 o/0.0625/1.0	o/**0.667**/0.857 o/0.5/**1.0**	o/**0.478**/**0.786** o/0.0870/0.25
	Speed limit change	N/A	N/A	N/A	x/0.0/0.0 **o/0.4/1.0**	o/0.2/1.0 o/0.2/1.0
	Lane shift leftward symbol	**o/0.5/1.0** x/0.0/0.0	**o/0.231/1.0** x/0.0/0.0	N/A	N/A	**o/0.625/1.0** x/0.0/0.0
	Lane shift text	o/**0.677**/1.0 o/0.647/1.0	o/**0.444**/1.0 o/0.222/1.0	N/A	N/A	o/**0.25**/**0.4** o/0.125/0.333
	Left lane closed symbol	N/A	N/A	N/A	o/0.625/1.0 o/**0.813**/1.0	N/A
	Left lane closed text	N/A	N/A	N/A	o/0.143/1.0 o/0.143/1.0	N/A
	Lane shift rightward symbol	N/A	x/0.0/0.0 x/0.0/0.0	N/A	o/1.0/0.7 o/1.0/**0.778**	o/1.0/**0.45** o/1.0/0.356



(a) Summary of the test results.



(b) Magnifies the dashed rectangle in (a).

(c) Example images of localization results.

Figure 3.7: Testing results of video data, E. (a) A graph showing the summary of the test results. The X-axis represents a sequence of frames, and the y-axis represents sign target classes. Green circles represent a groundtruth of the target class, red crosses represent a prediction of detection, and blue plus signs represent a prediction of the tracking system. (b) This subfigure magnifies the dashed rectangle in (a). The precision and recall of detection and tracking are presented, and each classification result is presented. (c) The yellow dotted rectangles are ground truth, the blue solid rectangles are tracking results, and the red dashed rectangles are detection results.

precision and recall on each sign by averaging the results from each image frame. We evaluated the classification performance on the nine target classes. For comparison, two different types of localization algorithms were executed: 'detection and tracking' and 'detection only'. Table 3.4 further provides details of the experimental results. The precision and recall values were calculated only from the frames that were executed on the tracking (e.g., all of the detection performance data before the tracking start were ignored.). The first row in each cell represents the performance when the tracking module was included while the second row represents the performance when the detection module was executed only. The results show that our tracker could successfully capture signs that were missed by our sign detector. For example, our tracking module detected 13 image frames that were missed by our sign detector. Furthermore, our tracker provided higher precision and recall than our detector only, which led to a better performance of the classification.

The performance of our workzone sign recognition system improved with the tracking module except on one test dataset, D. Our tracking system crops the potential sign region using the mode and standard deviation of the intensity within the bounding box. However, this sometimes over-crops the sign, which causes misclassification. The relatively poor classification performance on dataset D was also caused by this over-cropping problem. Also, as explained in Section 3.2, we heuristically increased the bounding box for a rectangular workzone sign ("workzone begins" and "speed limit change" signs), which contains two colors. Our tracking system only tried to track the orange color at the top and increased the bounding box based on the predefined ratio, which is prone to error when increasing the bounding box. We demonstrated that our system including the tracking component not only identifies the workzone bounds but also recognizes the details of temporary changes.

In order to examine if our workzone sign recognition system also works well in classifying other road signs, we further tested three different US road signs to detect and track: "stop", "pedestrian crossing" in school areas, and "workzone" signs. We chose them because of their unique characteristics. All of these signs have distinctive colors with standardized shapes as shown in table 3.5. Since there are no other signs that have the same characteristics as these signs, we only had to detect them, not requiring further classification. Also, these signs play important roles in improving the relationship among drivers, pedestrians, and road construction workers, particularly in suburban areas. Three different target color models were trained separately for the detection algorithm and applied appropriately based on the testing data.

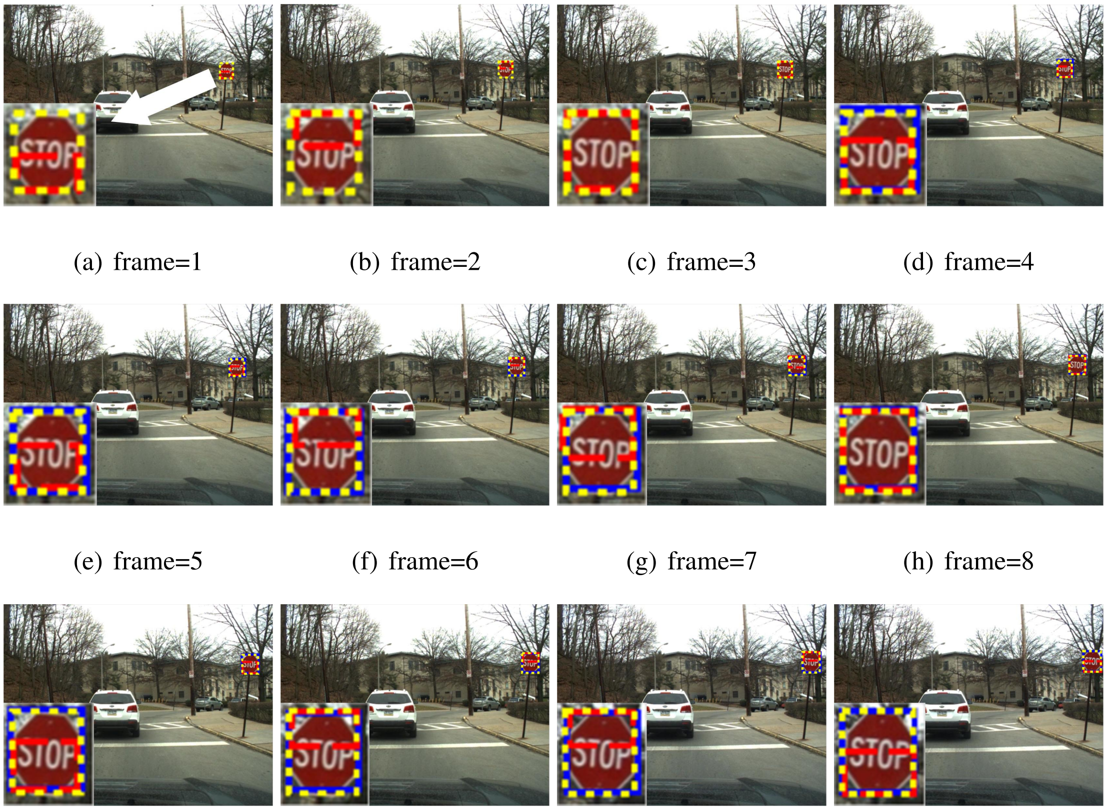
Video sequences were collected in 480 by 640 resolution at 15 fps under various environmental conditions. Each of these videos was decomposed into sequences of images. Seven different image sequences of each sign ("stop", "pedestrian crossing", and "workzone") were prepared for the testing data, and the remaining images were used to train the color classifier. For each of the testing data, sequential images were given as an input.

For comparison, we also executed two different types of localization algorithms as before: 'detection only' and 'detection and tracking'. 'Detection only' executed the algorithm as described in section 3.2 in every input image. 'Detection and tracking' executed the detection algorithm at first, and once the output from the detection satisfied the predefined conditions, the tracking algorithm was executed on the remaining frames. We empirically found that the tracking algorithm performed best when the size of the bounding box was bigger than 32 pixels. To represent the target and candidates, we set both n_h and n_s to 20, which resulted in a 400-bin histogram.

To depict our experimental results, we first demonstrated one of the results qualitatively and quantitatively. Figure 3.8 illustrates one of the experimental results. The detection process was executed until the size and aspect ratio of the bounding box satisfied the predefined thresholds ((a) to (c)). Then, the target model was calculated within the bounding box for the subsequent frame. As described in Section 3.3, the candidate was chosen as one with the maximum similarity to the target, and the target model was updated for the subsequent frame. Once our detector successfully detected a sign, the tracking system provided consistent bounding boxes through the rest of the sequence even when our detector partially detected the sign. Our sign detector applied a pixel-wise binary classification to every pixel in the ROI and identified blobs using connected-

Table 3.5: Three US road signs are chosen as target signs for our experiment. Each sign has its unique appearance [DOT, 2009].

Type	Shape	Color
stop	octagon	red
pedestrian crossing	diamond	fluorescent yellow-green
workzone	diamond and rectangle	fluorescent orange



(a) frame=1 (b) frame=2 (c) frame=3 (d) frame=4

(e) frame=5 (f) frame=6 (g) frame=7 (h) frame=8

(i) frame=9 (j) frame=10 (k) frame=11 (l) frame=12

Figure 3.8: Localization of detection and tracking in a sequence. The yellow dotted rectangle is a groundtruth bounding box labeled manually. The blue solid rectangle represents the output bounding box from tracking (starting at (d)), and the red dashed rectangle from detection alone. The inset at the bottom left in each frame is the magnified image of the stop sign.

component grouping. While doing this, our sign detector sometimes treated these blobs as two different parts (top and bottom individually) and picked the bigger part as a potential sign. However, we could improve this weakness by adding the tracking module, which provided a fitted bounding box by using the kernel. Once the sign was detected, the kernel contained a mask of the sign and found the most similar appearance of the target in the subsequent frame. This feature provided stable tracking.

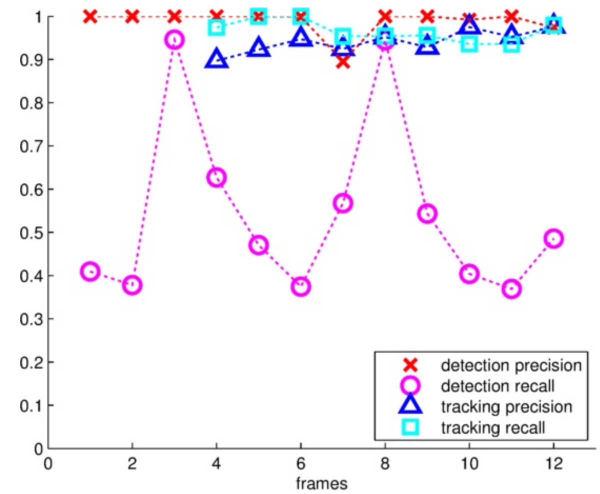


Figure 3.9: Localization performance result of a "stop" sign sequence in Figure 3.8 is illustrated. The tracking starts at frame 4. The precision and recall of the tracking system are better than those of the detection system.

Figure 3.9 shows the performance measurement of this sequence, where the x-axis represents the number of image frames ordered by time, and y-axis represents the values of precision and recall. A red cross is a precision, and a magenta circle is a recall. A cyan rectangle and blue triangle represent a precision and recall of tracking, respectively. As shown in Figure 3.9, the precision of detection is equal to or greater than that of tracking in the entire sequence except in frame 7. Our sign detector, as mentioned above, sometimes provides the bounding box from either the top half or the bottom half of the stop sign. Since the bounding boxes locate the inner-portion of the "stop" sign, they have high precision. However, they result in low recall because they only detect a portion of the sign. In contrast, our sign tracking system consistently detects the sign similarly in every frame.

We calculated the performance of each testing datum separately and averaged individual measurements over testing sequences to summarize the overall performance on each road sign. The results are provided in table 3.6. The first row of each subtable represents the precision

Table 3.6: Localization performance of 3 different signs from 7 image sequences of each sign. Each cell in the table shows the mean and standard deviation.

(a) workzone sign

	Precision	Recall
Detection and tracking	**0.979** / 0.013	**0.910** / 0.033
Detection only	0.950 / 0.034	0.775 / 0.105

(b) Pedestrian crossing sign

	Precision	Recall
Detection and tracking	**0.959** / 0.037	**0.916** / 0.051
Detection only	0.911 / 0.069	0.896 / 0.070

(c) Stop sign

	Precision	Recall
Detection and tracking	**0.954** / 0.018	**0.963** / 0.013
Detection only	0.947 / 0.055	0.774 / 0.221

and recall of ‘detection and tracking’ while the second row represents those of ‘detection only’. The results show that our tracking system improved both the precision and recall of the three road signs. Especially, the recall of ‘stop’ sign was higher than the recall of ‘workzone’ and ‘pedestrian crossing’ signs due to their shapes and kernel. We used a combination of a *mask* of shape of the detected sign and Epanechnikov as our kernel. A *mask*, however, sometimes could not perfectly be matched to its actual sign. Also, the kernel helps to create candidate PDFs by weighting more from the center. This results in a smaller bounding box than its actual size. Since diamond-shaped signs have more background than octagon-shaped signs, the recall values of the diamond-shaped signs were lower than those of the octagon-shaped signs. Not only do the precision and recall show better performance when tracking was included, but also the standard deviations showed less variation. This result implies that including a kernel-based tracking sys-

Total	Missed	Detect Rate
346	22	93.6%

Table 3.7: Performance of the channelizer detector.

tem provides more robust and stable sign localization performance.

We initially applied our approach to the workzone signs and the precision and recall were 97.9% and 91% respectively. Then, we extended our approach to other types of signs, i.e., pedestrian crossing sign and stop sign. The precision and recall on the pedestrian crossing signs were 95.9% and 91.6% respectively, and those on the stop signs were 95.4% and 96.3%, respectively.

In order to evaluate the performance of our channelizer detector, we counted the number

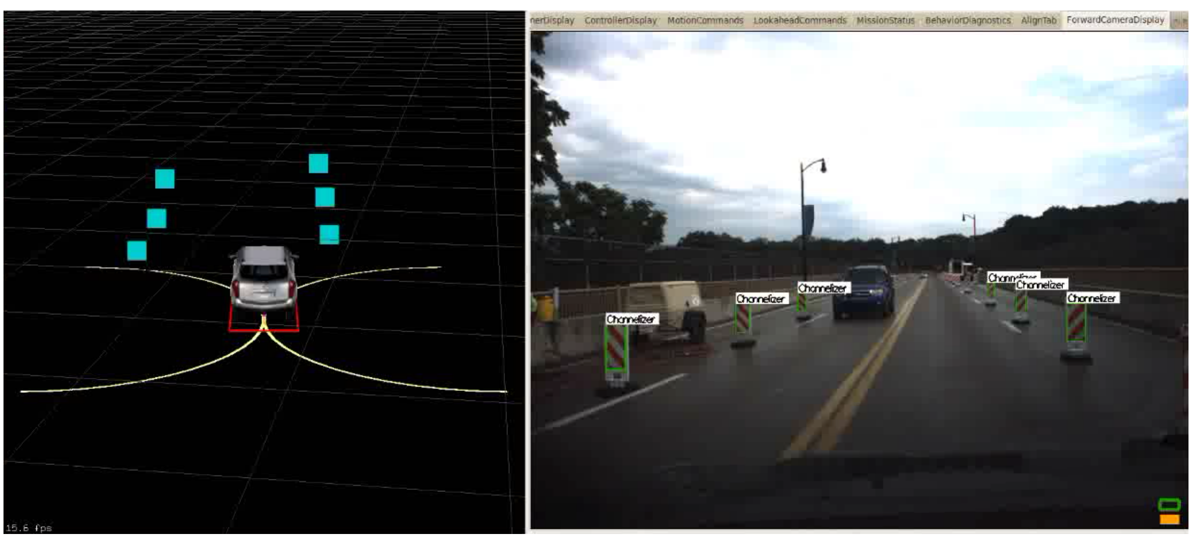


Figure 3.10: Blue rectangles in the left hand graphic show the position of detected channelizers. There are bounding boxes on top of channelizers in the right image and a small orange color indicator at the bottom right which indicates the vehicle is inside a workzone.

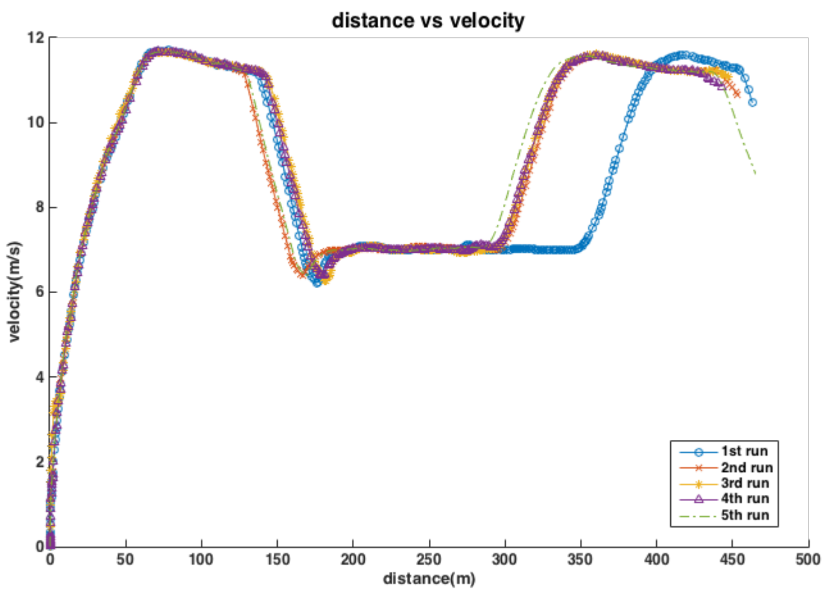
of missed channelizers. We treated the channelizer as detected when our algorithm detected the channelizer at least once among multiple appearances before disappearing. There were 346 channelizers in 5 videos, and only 22 of them were missed. They were missed because the channelizers were tilted, or too far from the upright position.

3.6.3 Demonstration

In the previous sections, we showed how well our workzone recognition system identified the bounds of workzones and the changes of highway geometry and traffic rules. However, such a perceptual capability alone would be insufficient to guarantee reliable autonomous driving. To prove the effectiveness of our workzone recognition system, we integrate our workzone recognition system into a self-driving vehicle to test how coherently our workzone recognition system works as a part of a self-driving car and helps the vehicle to adjust its driving behaviors on time. Our workzone recognition system ran on a 2.53GHz quad core (Intel Core2 Extreme processors QX9300s) with 1GB RAM. Since our camera was grabbing images at 5Hz, we also set the operating cycle of the workzone recognition system to 5Hz. In the next paragraphs, we demonstrate that our self-driving car can adjust its speed when our workzone recognition system observes a "Workzone Begins" sign and a "End Road Work" sign.

In order to generate a closed-loop interaction between our workzone recognition system and a self-driving car, we set up a mock workzone at our testing site. Figure 3.11 (a) shows our setup for a workzone. The longitudinal length of this workzone setup is about 100 m in a 450 m long straight road. It begins with the "Workzone" (R22-1) and ends with the "End Road Work" (G20-2) sign. In the middle of the site, there are two "Road Work Ahead" signs and multiple channelizers for simulating road geometry change. In this setup, we aimed to demonstrate that our autonomous driving car could 1) recognize the workzone, 2) lower its speed based on the changed speed limit, 3) avoid channelizers and 4) drive at a normal speed after passing through

(a) Mock-up workzone at Robot City near Carnegie Mellon University campus.



(b) Figure of distance vs. velocity of a vehicle movement.

Figure 3.11: Mock-up Workzone at Robot City near Carnegie Mellon University campus and the speed of the autonomous vehicle. The autonomous vehicle responds to the mock-up workzone by slowing down and going back to the normal speed when exiting the workzone.

the workzone.

We drove the vehicle five times on different dates and at different times. Figure 3.11 (b) shows the experimental results. From the starting location, the workzone begins at 185 m and ends at 290 m. In all of the five test runs, our workzone recognition system could successfully recognize the bounds of the workzone and the channelizers. After perceiving the presence of workzone, our vehicle was able to lower its speed as it approached the workzone and drove through the workzone. After passing through the workzone, the speed of the vehicle went back to normal. In the second trial as depicted by the blue line in Figure 3.11 (b), however, we found that the vehicle control logic had a communication issue, preventing the vehicle from accelerating on time.

In this chapter, we demonstrated that our workzone recognition system successfully detected and classified the nine different workzone signs. Furthermore, it could detect channelizers and additional road signs (i.e., pedestrian crossing signs and stop signs) with high recall and precision.

When evaluating the performance of our localization and classification, we first measured the performance of the detection only. Then, we compared its results to the performance of applying both detection and tracking to see how the tracking further improved the performance. The results showed that including tracking, in most cases, had higher than 90% precision and recall, and this good performance was due to more robust and stable sign localization.

In the end, we also demonstrated the capability of the workzone recognition system in the self-driving vehicle. In the next chapter, we further improve our workzone recognition system by considering the effect of illumination. Since color is very sensitive to different illumination, we researched how colors vary based on illumination, specifically based on different azimuth angles of the Sun, to predict colors of road signs more accurately.

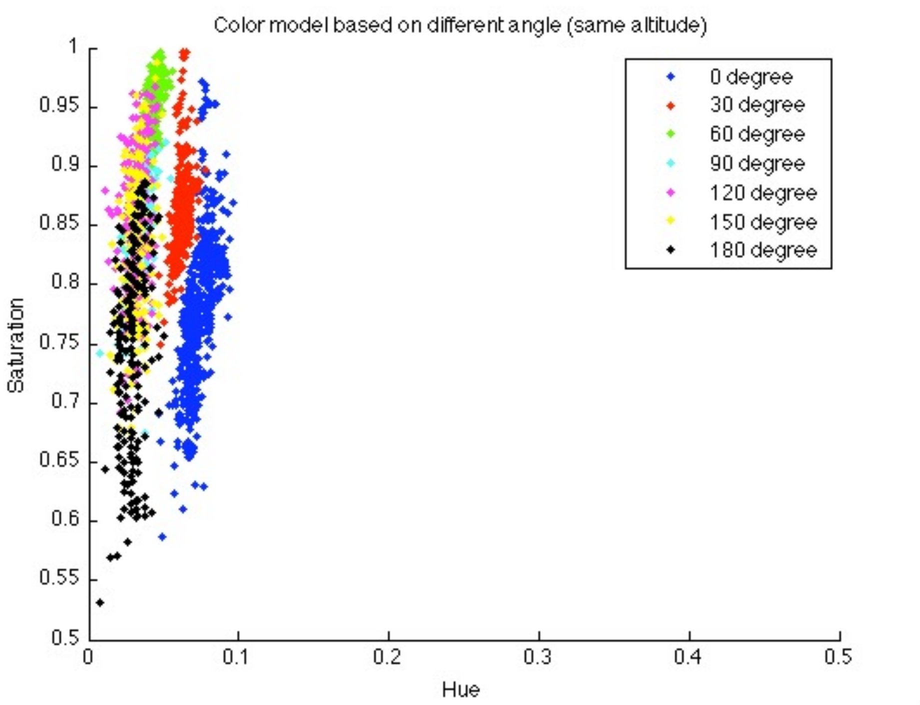
Chapter 4

Improved Traffic Sign Recognition with Sun Position

The sign recognition presented from the previous chapter was based on the pixel-wise color classifier to detect true orange pixels on the sequences of images. However, this technique sometimes produced false positives or false negatives depending on how the classifier was trained. As depicted in Figure 4.1 (a), the same object, which is captured consequently within 5 minutes, has different colors due to the different incident angles of the Sun. Even though the HSV color space was exploited to reduce the color variations caused by different illumination conditions, there still remained the variance of changed color, as depicted in Figure 4.1 (b). When the classifier set a tight boundary around orange pixels to reduce false positives, the number of false negatives increased. On the other hand, if the classifier set a loose boundary to reduce false negatives, the number of false positives increased. The performance of the pixel-wise color classifier was directly related to the performance of the sign detector. Thus, in order to improve the performance of the sign detector, we further investigated how to predict the orange color more accurately.

In this chapter, we present research on how color is changed, and how we improved the performance of construction zone sign detection. In section 1, we explore Solar illumination.

(a) The same sign under different illuminations



(b) Hue and saturation values of signs under different illuminations

Figure 4.1: The same sign is seen as having different colors due to the different angle of the light source.

Section 2 explains how we collected the data in order to tackle this problem, and section 3 describes how we applied these data. Section 4 shows the performance of the improved sign detection algorithm and compares to the performance of the algorithm shown in Chapter 3 in detail.

4.1 Solar Illumination

4.1.1 Solar Geometry

The Sun is the star at the center of the solar system, and the Earth travels around the Sun through a predictable orbit. A complete orbit occurs every 365.256 days, and this allows us to calculate the position of the Sun relative to the Earth. The position of the Sun in the sky can be specified by spherical coordinates θ_s and ϕ_s, where θ_s is the elevation of the Sun and ϕ_s is the azimuth of the Sun. These angles represent the location of the Sun relative to a horizontal direction and the north direction respectively at an hour angle (HRA) as depicted in Figure 4.2. The HRA, ϵ, is a function of the local solar time, γ, and it can be calculated by

$$\epsilon = 15^\circ(\gamma - 12) \tag{4.1}$$

where the local standard time (LST) relies on the location longitude and the equation of time to adjust local time, η.

$$\gamma = \eta + \frac{TC}{60} \tag{4.2}$$

The declination angle, δ, can be calculated by

$$\delta = \sin^{-1}\left(\sin(23.45^\circ)\sin\left(\frac{360}{365.24}(d-81)\right)\right) \tag{4.3}$$

where d is the day of the year with January 1 as $d = 1$.

The elevation angle, θ_s, can be calculated by

$$\theta_s = \sin^{-1}[\sin\delta\sin\varphi + \cos\delta\cos\varphi\cos(HRA)] \tag{4.4}$$

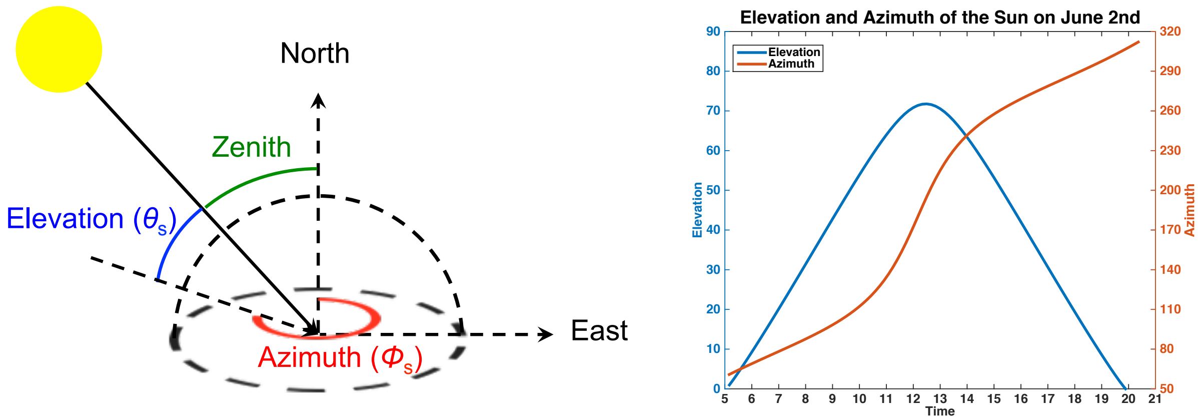


Figure 4.2: Azimuth and elevation of the Sun

where φ is the latitude of the location of the interest and HRA is the hour angle.

- TC = time correction factor, $TC = 4(Longitude - LSTM) + EoT$
- LSTM = local standard time median, $LSTM = 15° \cdot \Delta T_{GMT}$
- EoT = Equation of Time, $EoT = 9.87\sin(2B) - 7.53\cos(B) - 1.5\sin(B)$

where $B = \frac{360}{365}(d - 81)$.

The azimuth angle is defined as the angle between the vector of the north direction and the projected vector of the Sun onto the horizontal plane.

$$\phi_s = \cos^{-1}\left[\frac{\sin\delta\cos\phi - \cos\delta\sin\phi\cos(HRA)}{\cos\theta_s}\right] \tag{4.5}$$

The zenith angle is the angle between the Sun and the vertical.

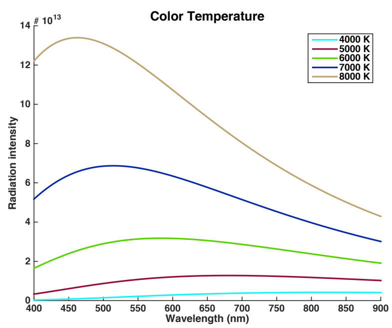
$$\zeta = 90° - \theta_s \tag{4.6}$$

4.1.2 Solar Radiance

The Sun emits radiation over a wide range of wavelengths, such as X-rays, ultraviolet, visible light, infrared, and radio waves. However, the important wavelength to produce the color is the radiation between 400 nm and 700 nm, which is the spectrum visible to the human eye. The spectrum of the Sun is close to that of an ideal black-body radiator, and the solar radiation $L(\lambda, T)$ can be calculated by Planck's law.

$$L(\lambda, T) = \frac{2hc^2}{\lambda^5} \frac{1}{e^{\frac{hc}{\lambda k_B T}} - 1} \tag{4.7}$$

where h is the Planck constant, c is the speed of the light, and k_B is the Boltzmann constant. Figure 4.3 shows the spectra of various temperatures of black-body radiators.



Daylight Sources	Color Temperature(K)
Skylight (Blue Sky)	12000 to 18000
Overcast Sky	7000
Noon Sun/Clear Summer Sky	5000 to 7000
Noon Sun/Clear Winter Sky	5500 to 6000
Noon Sunlight (Date Dependent)	4900 to 5800
Average Noon Sunlight (Northern Hemisphere)	5400
Sunlight at 30-Degree Altitude	4500
Sunlight at 20-Degree Altitude	4000
Sunlight at 10-Degree Altitude	3500
Sunrise and Sunset	3000

Figure 4.3: Spectra based on different color temperature and approximated color temperature of common light sources

4.2 Data Collection with Different Sun Position

Traffic control devices provide the primary means of communicating important information to drivers, and road signs are one of the three basic types of traffic control devices. Road signs have significant roles including providing regulatory, warning, and guidance information to drivers, and they should be visible at daytime and nighttime. For the nighttime visibility, road signs have retro-reflective surfaces, so the direction of the incoming light and that of the reflective light is opposite, as explained in Chapter 2.

The two important geometries of the retro-reflection model are the *incident angle* and the *observation angle*, as depicted in Figure 4.4. The *incident angle* is an angle between the line from the light source and the line which is perpendicular to the sign, and the *observation angle* is an angle between the line from the light source and the line from the sign to the camera. The retro-reflected light from a road sign spreads in a very narrow cone shape.

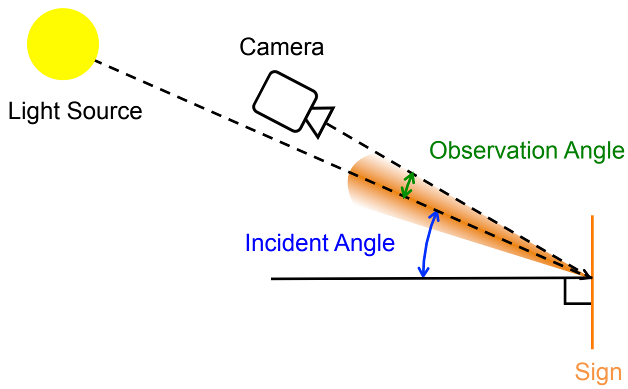


Figure 4.4: Incident and observation angles of retro-reflectance model

Given that all of the objects that control traffic are facing vehicles, we placed the camera to face the sign when collecting the data. Also, we fixed all the parameters of the camera in order to remove any other effects, such as shutter speed, gain, white balance, etc. We collected the data using the following steps:

1. Align the Sun, the sign, and the camera in order
2. Turn the sign 30 degrees clockwise, and move the camera accordingly
3. Follow the same procedure until the sign rotates 180 degrees
4. Repeat Steps 1 to 3 every hour (different elevation of the Sun)
5. Repeat Steps 1 to 4 every two weeks until July 4th

Figure 4.5 shows 7 different images, which have relative azimuth angles of 0°, 30°, 60°, 90°, 120°, 150°, and 180° respectively. All these images were taken within 10 minutes, and the color of the same object changed based on the different relative azimuth angles.

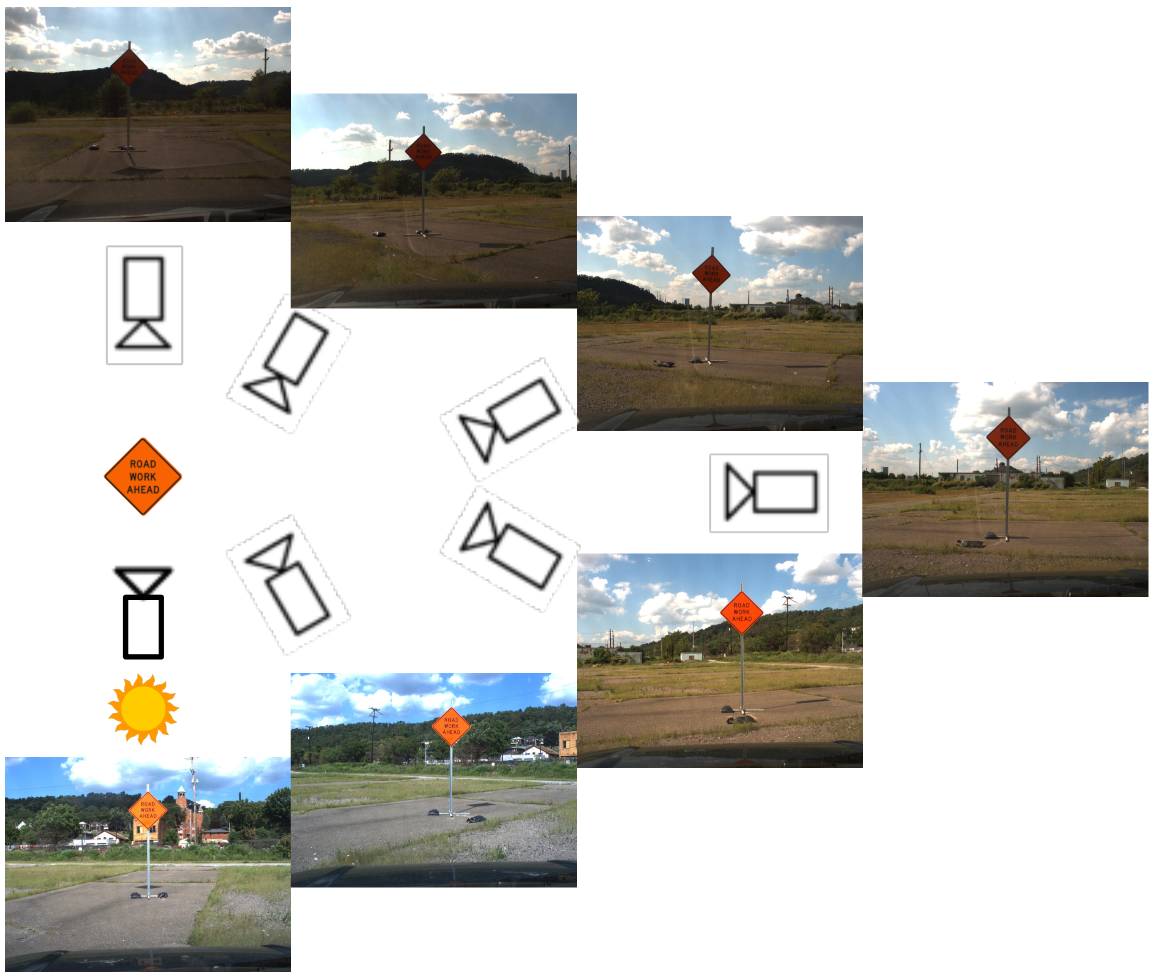


Figure 4.5: The Sun is right behind the camera, and the camera is facing the sign. Then, the sign is rotated every 30 degrees, and the camera is moved and rotated respectively to face the sign.

4.3 Sign Detection by Compensating Sun Position

As discussed in Section 2.1, the irradiance, E, onto the camera is proportional to the scene radiance, L. The illumination model relies on the BRDF of an object, and it can be represented as follows:

$$L = L_e(x, w_0) + \int_0^{2\pi} \int_0^{\frac{pi}{2}} f(\theta_{light}, \phi_{light}, \theta_{cam}, \phi_{cam}) E(\theta_{light}, \phi_{light}) \sin\theta \cos\theta \, d\theta d\phi \quad (4.8)$$

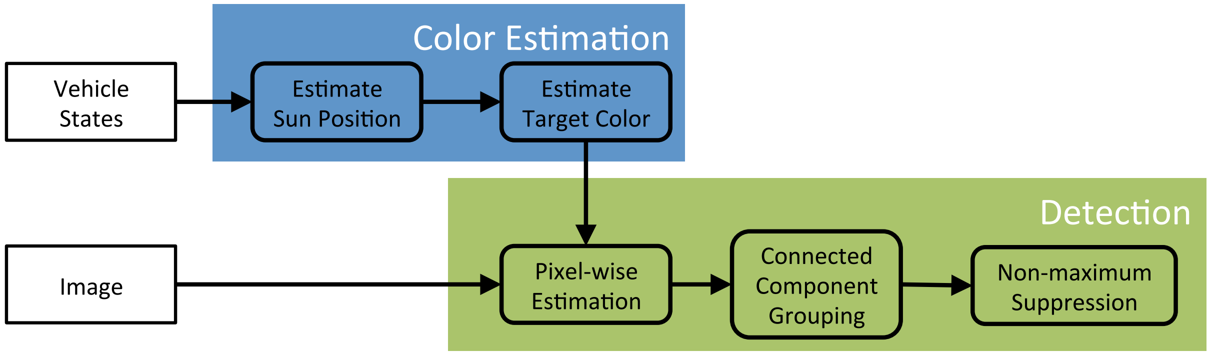


Figure 4.6: Schematic overview of sign detection by compensating the Sun position

We can simplify this equation given the fact that traffic signs do not have any emitting lights by themselves. Thus, we can ignore the first term of Equation 4.8, and this equation is simplified as follows

$$L = \int_0^{2\pi} \int_0^{\frac{pi}{2}} f(\theta_{light}, \phi_{light}, \theta_{cam}, \phi_{cam}) E(\theta_{light}, \phi_{light}) \sin\theta \cos\theta \, d\theta d\phi \tag{4.9}$$

From Equation 4.9, we can see that the amount of the illumination is proportional to the cosine of the relative angle between the light source and the sign and the sine of the relative angle between the sign and a camera. Under the assumption that the camera is facing the sign so that the sine of the angle between the camera and the sign is negligible, the main factor of an illumination change is the relative angle between the light source and the sign. When it is overcast, the light source is global, and the appearance of the sign is not changed based on the different directions. However, when the Sun is shining in a clear sky, we can assume that the Sun is a point light source, and the appearance of the sign is not changed based on the relative position between the Sun and the camera.

In order to compensate for the Sun position when predicting the appearance of a sign, we collected data by varying relative azimuth angles between the Sun and the sign as discussed in Section 4.2, and used a linear regression model to find the relationship between the relative angle

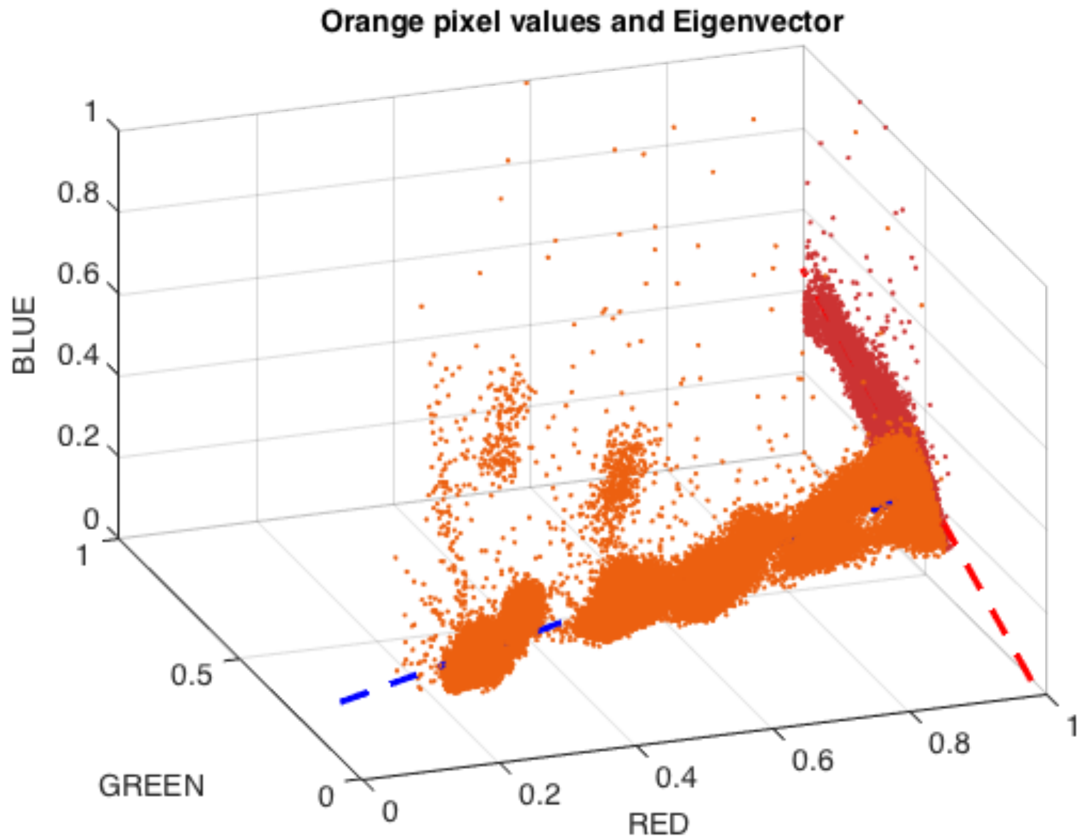


Figure 4.7: Orange dots represent the change of the orange color before the red channel is saturated. Red dots represent the change of the orange color after the red channel is saturated.

and the radiance.

First, we manually labeled the signs in images, extracted their colors based on the different relative azimuth angles, and plotted in RGB space, as shown in Figure 4.7. We only plotted data when the relative azimuth angle between the sign and the Sun was less than 90°, because the color of the sign is directly affected by the Sun when the relative azimuth angle is less than 90°. The orange dots represent the change of orange color before the red channel is saturated, and the red dots represent the change of orange color after the red channel is saturated. Since the orange and the red dots vary along the lines, we applied Principal Component Analysis (PCA) to find the eigenvectors of those dots. The blue and red dashed lines represent the eigenvectors of the orange and the red dots, respectively. As shown in Figure 4.7, the slope of the red dashed line is steeper than that of the blue dashed line due to the surface reflection. Figure 4.8 shows examples of body reflection and surface reflection. The red polygons inside the signs depict the surface reflection while the green polygons depict the body reflection.

Figure 4.8: Example of labeled data. Green polygons are for the body reflection, and red polygons are for the surface reflection.

After extracting the eigenvectors, we studied how the target color is changed along the eigenvectors. Equation 4.9 shows that the amount of the illumination varies with the cosine function, meaning that each magnitude of the red, green, and blue values varies with the cosine function. Even though we found how target pixels were changed, it was also important to consider the variations of the target pixels in order to estimate the target color correctly. Thus, we first calculated two variations; one is the variation along the eigenvector, v_a, and the other one is the variation of the distance to the eigenvector, v_d. Then, by exploiting the function of the magnitude along the eigenvector and these variations, we could estimate the target color.

$$e_d = d(\mathbf{c}_i, \bar{\mathbf{c}}_i)$$
$$e_a = d(\mathbf{c}_t, \bar{\mathbf{c}}_i)$$

where $\mathbf{c}_t$ is the target color at the specific relative azimuth angle between the Sun and a sign, $\bar{\mathbf{c}}_i$ is the estimated color which is projected onto the eigenvector, and $d(a, b)$ is the distance between a and b. e_a and e_d are distances from an original color to a projected color onto the eigenvector, and from the projected color to a target, respectively. Then, we calculated the probability of each pixel using these errors and chose the pixels with the probability higher than a predefined threshold.

Once it was determined whether all the pixels were the same color as the target color or not, we ran a connected-component grouping algorithm to find segmentations. Then, we checked the characteristics of these segments to remove false positives. First, we calculated the probability of each segment so that the segment could be removed from candidates. Then, we applied morphological operations to remove any noise and have better shape information. Finally, some of the segments were removed based on the shape information such as size, ratio, and orientation. All the remaining segments were the final output of our sign detector.

4.4 Experiments

4.4.1 Data Collection

We tested our framework in the real work-zone area. To show the capability of detecting signs in several illumination conditions (various Sun positions), we chose one on-going workzone area near the CMU campus, which is depicted in Figure 4.9, and drove the same route every hour on one day to collect a full sequence of images under various sun positions in a clear sky. We used a Flea3 camera with resolution 1280 by 960 at 8 fps. We labeled 14 video sequences with manually annotated ground truth of positions and target classes.

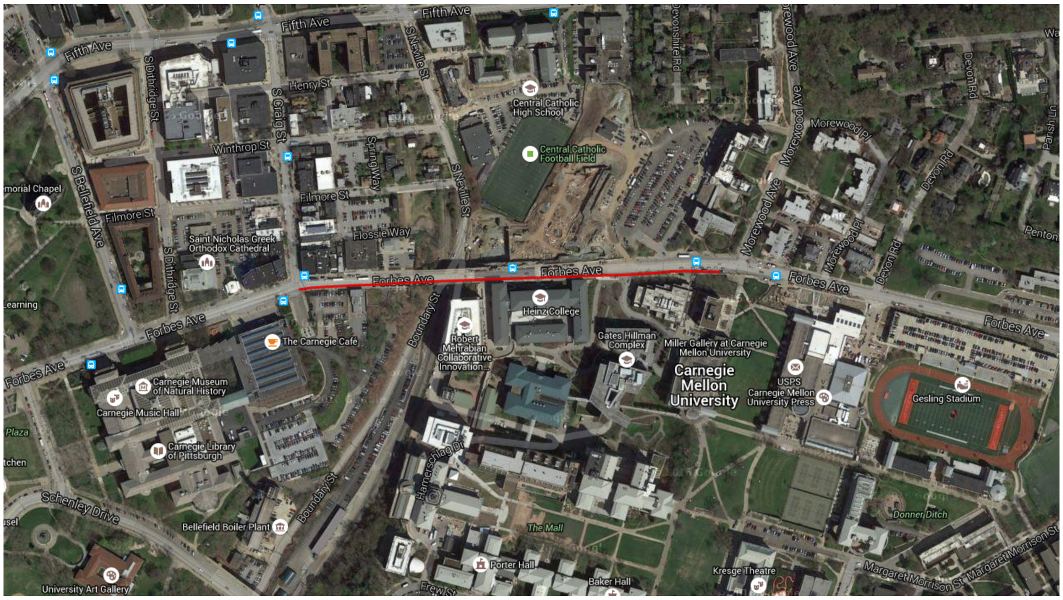
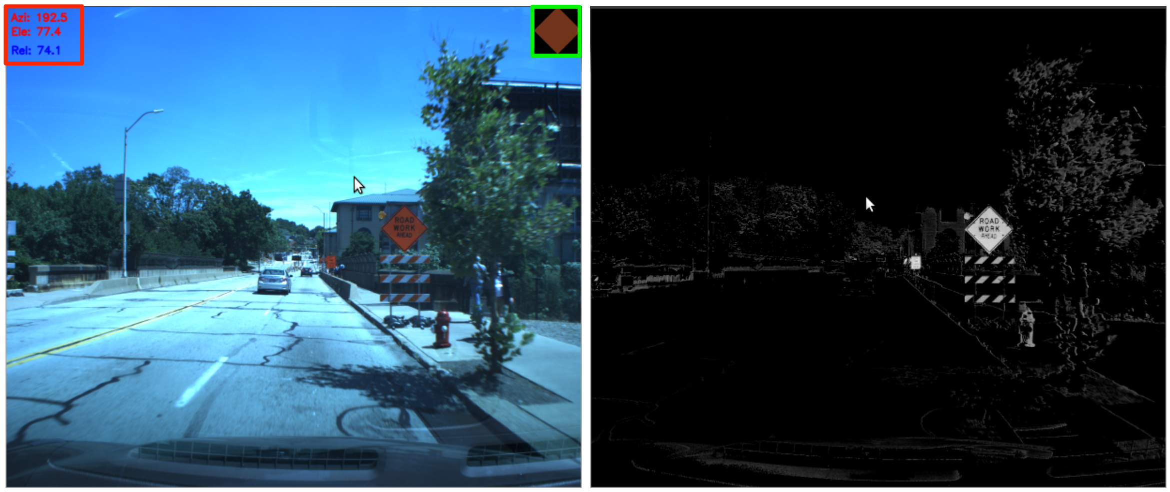


Figure 4.9: Testing area near Carnegie Mellon University

4.4.2 Evaluation

Our framework ran on a 2.90GHz quad core (Intel Core i7-3920XM), and could run faster than 10Hz. Since our camera was grabbing images at 8Hz, however, we also set the operating cycle of our system to 8Hz to run in real-time. In order to evaluate the performance, we used the metrics used for PASCAL object detection challenges [Ponce et al., 2006]. An output bounding box, o_i, was considered a potential match to the ground truth bounding box, g_i, in a given image frame, i, if their overlapping area was greater than a predefined value, $\tau\frac{Area(o_i \cap g_i)}{Area(o_i \cup g_i)}$. When a potential match was found in a given image, sign detection performance could be further analyzed by measuring the precision, $\frac{Area(o_i \cap g_i)}{Area(o_i)}$, and recall, $\frac{Area(o_i \cap g_i)}{Area(g_i)}$.

In order to clearly illustrate our experimental results, we first provide one of the sequences



(a) Input image (b) Distance image to the target color

Figure 4.10: Output of a single image. The text at the left top corner represents the azimuth and elevation angles of the Sun (red), and the relative azimuth angle between the Sun and the camera (blue).

of the images in detail. Figure 4.10 is an example from the output of the single image. The texts inside the red rectangle at the top left corner indicate the current position of the Sun and the relative angle between the Sun and the camera. The red texts 'Azi' and 'Ele' represent the azimuth and elevation angles of the Sun, respectively, and the blue text 'Rel' represents the relative azimuth angle between the camera and the Sun. The diamond shape at the top right corner shows the target color, which is the predicted color of the construction zone sign based on the current relative angle.

The performance was calculated based on each sequence, and the overall performance was summarized by summing all the sequences, as shown in Table 4.1 and Table 4.2. Each row represents a dataset, a number of the construction zone signs, true positives, false positives, false negatives, precision, recall, and the relative azimuth angle in the dataset. Table 4.1 includes datasets that were collected when the Sun was mostly behind the camera so that the illumination

Table 4.1: West To East

Dataset	Number of signs	TP	FP	FN	Precision	Recall	Relative Azimuth Angle
A	64	60	1	4	0.983	0.938	73.8-75.9
B	99	90	6	9	0.938	0.909	60.6-62.8
C	110	110	0	0	1.000	1.000	32.6-34.9
D	73	72	5	1	0.935	0.986	18.3-20.2
E	120	110	4	10	0.965	0.917	0.2-1.8
F	78	77	2	1	0.975	0.987	7.4-9.0
G	94	91	0	3	1.000	0.968	12.7-14.3
Overall	638	610	18	28	0.971	0.956	0.2-75.9

was relatively gentle, and when the target color varied more apparently because the light sources were the combination of the Sun and the sky. On the other hand, Table 4.2 contains the datasets that were collected when the Sun was in front of the camera. The target color was not changed much because there was only a global light source, but the images were saturated because of the direct Sun to the camera.

Our approach achieved 97.1% precision and 95.6% recall, as shown in Table 4.1. All these data were recorded when the relative azimuth angle was less than 75.9°. As shown in the table, there were 27 false positives from 7 different datasets, and we learned that they were mostly caused by the shadows of the other structures. As shown in Figure 4.11 (a), the sign is right next to the building, and the direct sunlight is blocked by the building. Due to the surface reflection, the predicted color was different. Figure 4.11 (b) shows another example. Our model could predict the color correctly on the left and right part of the sign. However, there was a shadow from a pole, thus it was difficult to predict this part. In this case, the connected component grouping algorithm split the sign into two parts and missed the sign.

(a) Sign under the shadow (b) A shadow of a pole splits the sign into two part

Figure 4.11: Examples of false negatives in Table 4.1. Signs are under the shadows.

On the other hand, we achieved 85.6% precision and 77.5% recall when the angle was bigger than 85.6 °. The performance was degraded especially when the Sun was in front of the camera, and the glare from the Sun saturated the images. Most of the false negatives arose from this case. However, when the vehicle came close to the signs, the signs were located at the edges of the images, which would be less affected by the saturation, thus our system was able to recognize the signs correctly. Most false positives were due to other workzone objects on the route, which

Table 4.2: East To West

Dataset	Number of signs	TP	FP	FN	Precision	Recall	Relative Azimuth Angle
H	70	70	15	0	0.824	1.000	85.6-106.4
I	61	61	9	0	0.871	1.000	98.4-124.7
J	66	43	4	23	0.915	0.652	143.7-164.8
K	56	22	5	34	0.815	0.393	163.5-180
Overall	253	196	33	57	0.856	0.775	85.6 - 180

also contained orange components.

In order to compare the performance with and without the Sun position, we also ran the algorithm explained in Chapter3 with the same dataset. In order to avoid any overfitting problem in AdaBoost, we used a leave-one-out cross validation technique. For example, we used datasets from B to G in order to train a pixel-wise orange color classifier and applied them to classify dataset A. As shown in Table 4.3, we included additional columns, which are precision and recall without the Sun position. The better performance in precision and recall between the two different algorithms is highlighted in bold. In overall performance, we obtained higher precision and recall from the algorithm that compensates the Sun position. It achieved 97.5% precision and 94.0% recall, while the algorithm without the Sun position only achieved 93.5% and 85.1% respectively. The recall could be increased by almost 10% because the sign detector can clearly distinguish a true orange color from red or yellow by compensating the Sun position. However, there were certain cases where the previous algorithm performed better. We explain these cases in detail.

In general, the algorithm with the Sun position outperformed the algorithm without the Sun

Table 4.3: Comparison Table of Two Different Algorithms

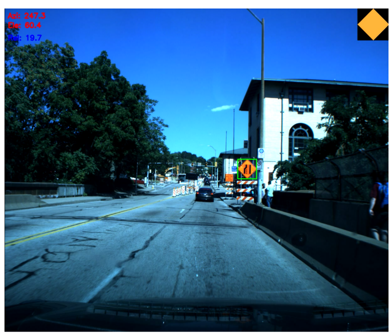
		with Sun Position					without Sun Position					
Datset	Number of signs	TP	FP	FN	Precision	Recall	TP	FP	FN	Precision	Recall	Relative Azimuth Angle
A	64	60	1	4	**0.983**	**0.938**	52	1	12	0.981	0.813	73.8-75.9
B	99	90	6	9	0.938	0.909	92	3	7	**0.968**	**0.929**	60.6-62.8
C	110	110	0	0	**1.000**	**1.000**	103	23	7	0.817	0.936	32.6-34.9
D	73	73	4	0	**0.947**	**1.000**	66	8	7	0.892	0.904	18.3-20.2
E	120	110	4	10	0.965	**0.917**	78	2	42	**0.975**	0.65	0.2-1.8
F	78	77	2	1	**0.975**	**0.987**	58	10	20	0.853	0.744	7.4-9.0
G	94	91	0	3	**1.000**	**0.968**	79	6	15	0.929	0.840	12.7-14.3
Overall	638	610	17	27	**0.973**	**0.955**	528	53	110	0.909	0.828	0.2-75.9

position. Whenever there were false positives, they were mostly caused by other construction objects having the orange color. On the other hand, the algorithm without the Sun position confused red with orange because of the similar hue values of red and orange. Without the Sun

(a) Dataset A

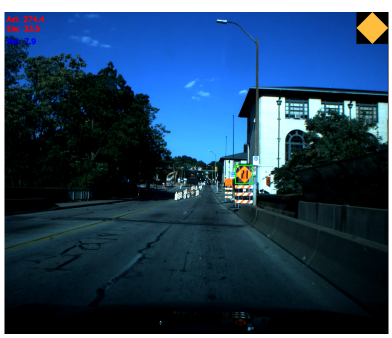
(b) Dataset B

(c) Dataset C

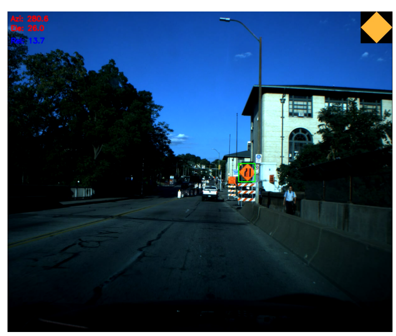


(d) Dataset D

(e) Dataset E



(f) Dataset F



(g) Dataset G

Figure 4.12: Example images in Table 4.1. The sign at the top right corner of each image shows the predicted color based on the relative azimuth angle.

(a) Dataset D (b) Dataset E

(c) Dataset F

Figure 4.13: Examples of false positives in Table 4.1 when the algorithm that compensates for the Sun position is applied. Sign detector falsely detects an orange trailer related to the construction. (highlighted by green box)

position, this algorithm will have either high false positives and low false negatives or low false positives and high false negatives. Thus, in order to calculate the performance in Table 4.3, no fine-tuning to achieve the best performance was executed.

4.4.2.1 Dataset C

The relative azimuth angle between the Sun and the camera is between 32.6° and 34.9° in this dataset. There are 110 workzone signs labeled in this dataset. As you can see in Table 4.3, there

(a) Dataset C (b) Dataset C

(c) Dataset D (d) Dataset D

(e) Dataset D (f) Dataset F

Figure 4.14: Examples of false positives in Table 4.1 without compensating the Sun position. Most false positives occur because the color pixel classifier conflicts red and orange pixels.

were neither any false positives nor false negatives when the Sun position was compensated by the algorithm.

However, when the old algorithm was applied, there were 23 false positives and false negatives. As depicted in Figure 4.14, the color classifier confused orange with red pixels, so the false positives occurred with red vehicles, as shown in Figure 4.14 (a) and (b). They are the most common failure cases, because hue values are the input of the color classifier, and hue values of red and orange are very close to each other, making the pixel-wise color classifier misclassify red and orange pixels.

4.4.2.2 Dataset D

The relative azimuth angle between the Sun and the camera is between 18.3° and 20.2° in this dataset. There are 73 workzone signs labeled in this dataset. Both of the two algorithms had false positives, but they were raised by different objects.

The algorithm with the Sun position did not have any false negatives, but it had false positives from the construction objects, as depicted in Figure 4.13. Those false positives were caused by the construction object in 4 consecutive images. Since the construction objects also have orange color, our color prediction gave a false positive.

On the other hand, the algorithm without the Sun position had 8 false positives and 8 false negatives. As depicted in Figure 4.14 (c) - (e), all of the false positives occurred with red objects, such as buses and flowers, meaning that the color classifier confused red with orange pixels.

4.4.2.3 Dataset E

The relative azimuth angle between the Sun and the camera is between 0.2° and 1.8° in this dataset, where the Sun is almost right behind the vehicle. In this case, the surface reflection from a sign also changes the appearance, so both body-reflected and surface-reflected colors should be considered as a target color. Again, the algorithm with the Sun position had false positives from construction objects. Also, in this dataset, 10 false negatives occurred. As depicted in Figure 4.12 (e), there was a shadow in the middle of the sign, and the connected-component grouping divided the sign into two different segments. The same happened when the algorithm without the Sun position was applied. Moreover, the algorithm without the Sun position failed in a couple more cases because it did not consider the surface reflection part.

4.4.2.4 Dataset F

The relative azimuth angle between the Sun and the camera is between 7.4° and 9.0° in this dataset, where the appearance is still affected by the surface reflection. In this case, the surface reflection from a sign also changes the appearance, so both body reflected and surface reflected colors should be considered as a target color. Again, the algorithm with the Sun position had false positives from construction objects. Also, there were still false negatives because of this surface reflection.

4.4.2.5 Dataset G

The relative azimuth angle between the Sun and the camera is between 12.7° and 14.3° in this dataset, where the appearance is still affected by the surface reflection. All the false negatives came from the beginning of the workzone sign, and this happened to most of the other datasets. The image of the workzone sign from the beginning of the route contains only a small portion of orange pixels at the top and the black texts in the orange background. Thus, our detector sometimes missed these signs due to the lack of orange pixels. Even though the sign detector first failed to detect this part, our algorithm could successfully pick up estimating those orange pixels in the following images.

In this chapter, we first described the importance of illumination in sign detection and developed an algorithm that compensates for the Sun position in order to further improve estimating colors of signs.

We collected data with various relative angles between the Sun and the camera, and compared the performance of the algorithm that does not compensate for the Sun position to the performance of the algorithm that compensates for the Sun position.

The performance of sign detection is much improved with an algorithm that compensates for the Sun position. The recall from every dataset was higher than 90%, as was the precision in most cases. Because we considered the relationship between the relative angles and the corresponding radiance, the algorithm more robustly distinguishes the true orange color from red or yellow. We examined that this algorithm was not sensitive to the slope road because we calculated the relative angle between the Sun and the camera.

There were very few false positives, and they were mostly caused by other construction objects that are also orange. Most false negatives were caused by either shadow or surface reflection, but the number of false negatives and false positives is significantly reduced when we considered the Sun position because we could predict true orange color in images.

We believe this algorithm can be generalized to the other applications such as road markings and judge barriers because this framework understands how the predicted color would be changed based on different materials and diffrerent positions of illumination.

Chapter 5

Traffic Light State Estimation

Advanced driver assistance systems (ADAS) provide various functions, such as lane marking detection, speed limit sign detection, stop sign detection, vehicle detection, etc. Recently, ADAS started to provide traffic light detection, which is one of the most demanding functions given that traffic lights are required during driving. However, they only have capabilities of detecting traffic lights within 50 meters or so, which is a relatively short range given that most drivers need a minimum of 59.27 meters to stop under unexpected situations when driving at 40 MPH [Administration, 2014]. Also, ADAS produce many false positives, which can be a major problem in autonomous driving.

There are two ways to read traffic lights. One is to use cameras and the other is to use communication technologies, such as vehicle-to-infrastructure (V2I). V2I technology can provide the traffic light information, e.g., the current state and the timing of traffic lights so that autonomous vehicles can drive safely and efficiently. However, V2I technology needs transmitters from traffic light control boxes and receivers in vehicles, which are very expensive to deploy in the real world.

This chapter explores traffic light detection by using data collected from cameras. We first review previous work on traffic light detection in Section 5.1. Section 5.2 provides how to set camera parameters to get illumination-invariant performance. Section 5.3 explains the algorithms of our traffic light state estimation system, and we conclude with the performance evaluation of our traffic light state estimation system.

5.1 Related Research

Many researchers have been conducting experiments to advance traffic signal detection systems for assisting human drivers and intelligent vehicles. Most of them attempt to improve the signal detection by exploring color information. There are also those who utilize shape information to detect traffic signals. The next paragraphs first summarize the experiments performed based on color information and continue to introduce color and/or shape-based experiments. Table 5.1 provides an overview of previous work on traffic light detection.

Table 5.1: Overview of Traffic Light Detection.

Paper	Color	Shape	Night Time	Camera Setting	Distance	Performance
[Chung et al., 2002]	○[a]	○	×[c]	-	-	
[Lindner et al., 2004]	○	△[b]	×	16 mm lens	100 meters	
[Hwang et al., 2006]	○	×	×	720 x 480	85 meters	
[Kim et al., 2007]	○	×	○	-	100 meters	90 %
[Charette and Nashashibi, 2009]	△	○	×	AVT Marlin F-046C	-	P:95.38 %, R:98.41 %
[Shen and Ozguner, 2009]	△	○	×	Sony DFW-V500 (640 x 480)	50 meters	99 %
[Gong et al., 2010]	○	○	×	780 x 580 with 15 mm lens	-	-
[Nienhuser, 2010]	○	○	×	512 x 384	-	89.6 %
[Fairfield and Urmson, 2011]	○	×	○	5 MP image with 30 degree FOV	150 meters	P:0.99, R:0.62
[Levinson et al., 2011]	○	×	○	1.3 MP image	140 meters	91.7 %
[Cai et al., 2012]	○	○	×	1392 x 1040 with 20.4 degree FOV	-	92.61 %
[Diaz-Cabrera, 2012]	○	×	○	752 x 480	80 meters	P:90.32 %

[a] The authors mainly researched on this feature

[b] The authors partially researched on this feature

[c] The authors did not explore this feature

Hwang *et al.* [Hwang et al., 2006] applied color thresholding in the HSI color space and found blobs using a morphology operation. They showed the performance with respect to the distance from the vehicle to the traffic signal, but the performance was less than 90% when the distance was more than 85 meters, and they did not test their algorithm at night. Kim *et al.* [Kim et al., 2007] applied intensity thresholding, color segmentation, and an RGB histogram to detect traffic signals. They tested under three different weather conditions and achieved the best performance (90%) during daytime within 100 meters. However, their maximum distance was not long enough to comfortably stop while driving at 45 MPH. Fairfield *et al.* [Fairfield and Urmson, 2011] first mapped all the traffic signals using the pose of the vehicle and sequences of images. They used geo-spatial queries to automatically select the images at intersections, detected traffic signals, and calculated their locations by least square triangulation. When they executed the traffic signal detection in the vehicle, they predicted the region of interest using the locations of the traffic signals, and estimated its current state by color. They showed a good performance at a distance of 150 meters, but they had difficulty in detecting arrow traffic lights that were dim. Levinson *et al.* [Levinson et al., 2011] used a similar approach. They also mapped the traffic signal off-line and collected the color information of traffic signals while mapping. Then, they made a histogram as lookup-tables to estimate the current state. They performed a back-projection of these histograms to calculate the probability of each signal and chose the highest probability among them. However, the performance was not good enough with an accuracy of 91.7%. Diaz-Cabrera *et al.* [Diaz-Cabrera, 2012] also used the color thresholds in the normalized RGB channel to find the back panel of the traffic signal. Then they computed intensity values of the traffic signal to remove false positives. However, the maximum distance was only 80 meters.

There are research groups who mainly examined shape information. Charette *et al.* [Charette and Nashashibi, 2009] recognized traffic signals by detecting spot lights and matching adaptive traffic signal templates. They found bright areas by the White Top-Hat transform and adaptively matched templates of back panels of the traffic signals and the poles. However, they did not test their algorithm at night. Shen *et al.* [Shen and Ozguner, 2009] constructed traffic signal feature models for red, yellow, and green signals and used these models to find candidates. They used the morphological operation to find traffic signals among those candidates. The performance was above 99% in 5 different scenarios, and 92.7% in the worst case, but they only measured the performance when the distance between the vehicle and the traffic signal was less than 50 meters. Also, it executed at 1.4 FPS in MATLAB only.

The work of Chung *et al.* [Chung et al., 2002] explored both color and shape features of traffic signals. They detected traffic signals with a video sequence that is recorded by a stationary camera at the intersection. They generated background images and estimated the mean time illumination parameters and detected traffic signals in the HSI color space with spatial and temporal information. However, there was no quantitative evaluation. Next, Gong *et al.* [Gong et al., 2010] first extracted the candidate region of the traffic signals based on color segmentation. They set thresholds on hue and intensity channels to find the initial candidates and removed noise by the morphological operation. Then, they used the Continuously Adaptive Mean Shift (CAMSHIFT) algorithm to track those candidates. However, they also did not evaluate their algorithms quantitatively. Nienhüser *et al.* [Nienhuser, 2010] detected traffic signals using the morphology operation and rule-based validation on a HSV color space. Then, they applied a Hidden Markov Model to reduce the false positives. However, the performance was less than 90%. Cai *et al.* [Cai et al., 2012] focused on detecting arrow traffic signals only. They first found the black back panel of traffic signals by performing color thresholding and a morphology operation, and used another color threshold to detect traffic signals. Then, they used the Gabor wavelet transform to pick up important features to classify the arrows.

Most of the traffic signal detection systems mentioned above achieved more than 90% detection performance. However, the distance of the detection range was less than 100 meters in most cases, which is not enough range for the self-driving vehicle to stop smoothly when driving at 45 MPH. In addition, most of them did not mention their performance at nighttime, and the execution time was not fast enough to be deployed in autonomous vehicles.

In order to overcome these limitations, we worked on improving performance at nighttime. Moreover, we chose to map the traffic signals as a priori information to increase the detection range and reduce the computational cost. We also improved the problem of detecting dimmed traffic lights by first examining the frequencies of traffic signals and then setting the parameters of a camera properly. This approach allowed us to execute in real-time and test in an autonomous vehicle.

5.2 Camera Setting

The parameters of the camera for the traffic light detection need to be defined specifically in order to execute the system robustly under various illuminations. Before setting the camera parameters, we first describe a general illumination model and define an illumination model of traffic signals.

$$L = L_e(x, w_0) + \int_0^{2\pi} \int_0^{\frac{pi}{2}} f(\theta_{light}, \phi_{light}, \theta_{cam}, \phi_{cam}) E(\theta_{light}, \phi_{light}) \sin\theta \cos\theta \; d\theta d\phi \tag{5.1}$$

As seen in equation 5.1, the illumination model of an object is comprised of two parts; the first term is an emission and the second term is a reflection caused by other light sources.

The regularized design of a traffic light has a cover on top of each light, and a back panel to help drivers see the lights more clearly. Because of these covers and back panel, we can assume

Figure 5.1: Examples of traffic light images with parameters fixed to a shutter speed of 10ms and a gain of 0db. The white balance is turned off, and the neutral density filter is installed.

that there is no reflection on the light and simplify equation 5.1 as follows:

$$L = L_e(x, w_0) \tag{5.2}$$

Once the camera has proper fixed parameters, we can expect to have a consistent color from each traffic light no matter where the traffic light is facing because the camera responds only to the emitted light from a traffic light.

To set the camera parameters, we collected data at different times on a sunny day. We focused on two main features when setting the parameters. The first one was to get the high saturation values from the traffic light, but not totally saturated values. The second one was not to have the high saturation values from the sky. We fixed the shutter speed as 0.5 ms and the gain as 0. Also, we disabled all the other parameters, such as the white balance, in order to get consistent values from each traffic light.

Even though a fixed shutter speed and gain produce a relatively consistent value from traffic lights, it is also important to consider the frequency of the traffic light emission. Traffic lights usually emit their own lights at 50-60Hz. If the shutter speed is set as 0.5 ms, which is relatively

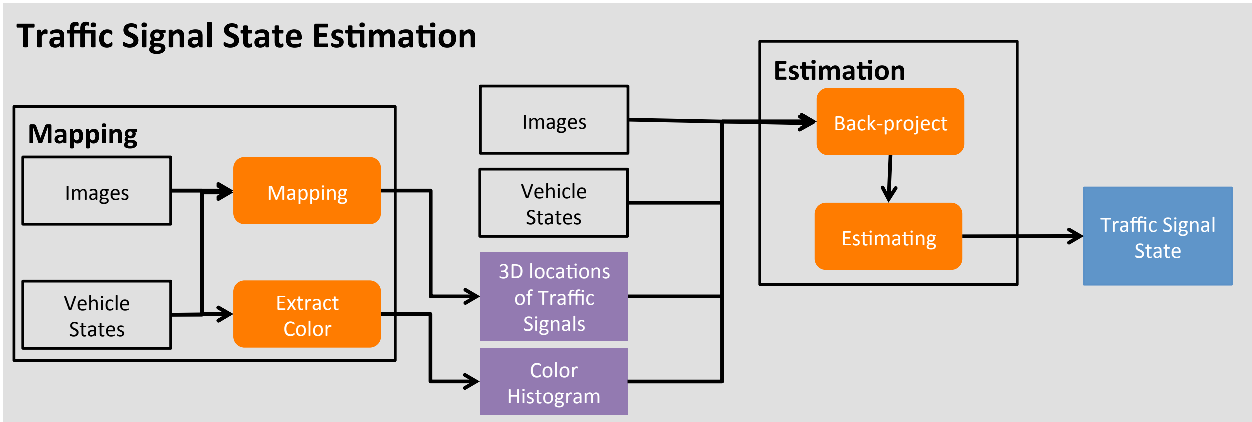


Figure 5.2: Schematic overview of traffic light state estimation.

too fast given the emission frequency, traffic lights on images in a sequence will repeatedly dim and brighten. In order to show consistently bright traffic lights in images, the shutter speed should fully cover half of one cycle of the emission, which is 8 - 10 ms. Thus, we fixed the shutter speed as 10 ms and installed a neutral density filter on the image. We specifically choose 5% emission of the neutral density filter to achieve these constraints.

5.3 Traffic Light State Estimation

Our traffic light detection algorithm is comprised of two parts. Figure 5.2 shows the schematic overview of our traffic light detection algorithm. The first part is Mapping, which is an offline process to understand where traffic lights are located and how they look, and the second part is Estimation, which is an online process to estimate the current states of traffic lights while driving. In the next sections, we explain each part in detail.

5.3.1 Mapping

In order to map the positions of traffic signals in world coordinates (or Universal Transverse Mercator (UTM) coordinates), we first drove a route with a car equipped with the camera and

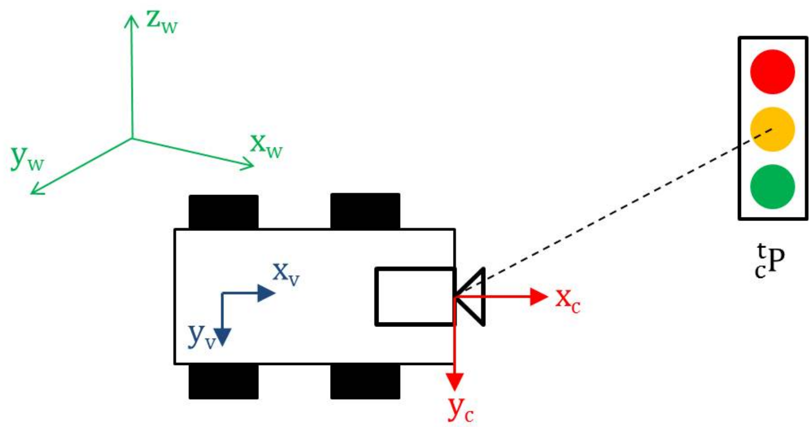


Figure 5.3: Transform of the traffic signal location from the camera coordinate to world coordinate.

navigation sensors. After logging the vehicle state and images, we manually selected the images that contain traffic signals. We roughly set the range of hue, saturation, and intensity values and used those ranges for thresholding values to find a candidate region of the traffic signal. Then, we manually selected the region as an input and executed the morphology operation to remove noise if it exists. After finding a traffic signal in an image, we tracked the traffic signal in the consecutive images until it disappeared from the image. While tracking, we also kept the vehicle state, $\mathbf{P}_v$, which contains the position in UTM coordinates and the orientation angle.

When we know the intrinsic parameters of the camera and the image coordinates, it is possible to calculate the ray equation of a certain pixel in the image. For each image, we calculated the ray equation, $\mathbf{l}$, by using the coordinates of the center of the traffic signal, $\mathbf{p}$, and intrinsic parameters of the camera, K, as follows:

$$\mathbf{l} = K^{-1}\mathbf{p} \tag{5.3}$$

Since we find the same object in a sequence of images, all the rays from a sequence of images should converge to a single point if the extrinsic and intrinsic parameters of the camera are

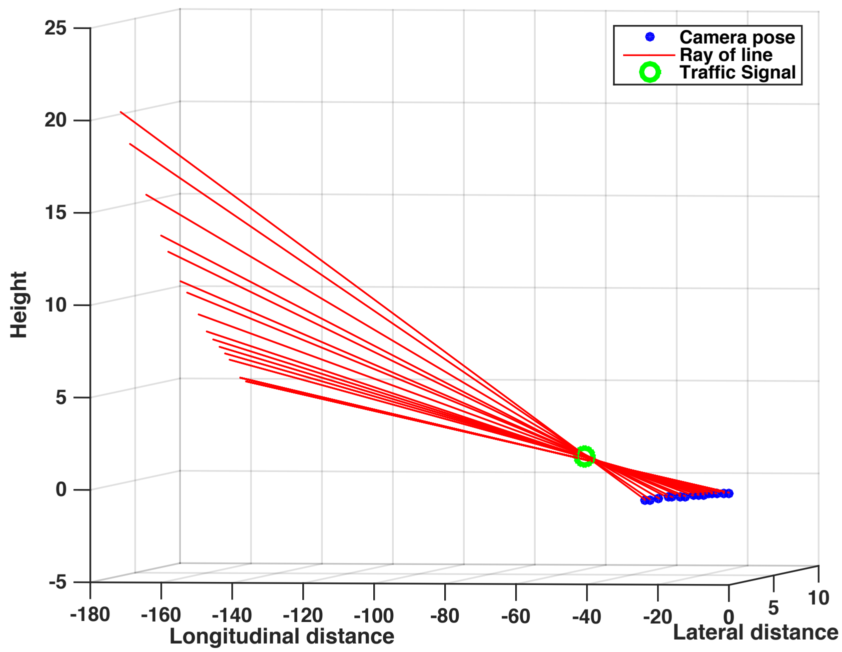


Figure 5.4: Sequence of rays from each vehicle pose and its optimal intersecting point.

calibrated, and the vehicle state has no errors. However, because the parameters and the sequence of the vehicle state contained errors, we used least squares triangulation to find an optimal point of intersection of these rays. Figure 5.3 shows the coordinate system from the camera to UTM. First, we treated the first vehicle pose, $\mathbf{P}_0^v$, in a sequence as an origin and transformed all the remaining vehicle poses, $\mathbf{P}_t^v$, relative to the first vehicle pose. Then, we transformed all the ray equations of every sequence from the camera coordinates to the vehicle coordinates by using equation 5.4:

$$\mathbf{l}_t^{v_0} = \mathrm{T}_w^{v_0}\, \mathrm{T}_v^w\, \mathrm{T}_c^v\, \mathbf{l}_t^c, \qquad t = 1, ..., n \tag{5.4}$$

where T_a^b is a transformation from a coordinates to b coordinates, v_0 is a vehicle coordinate at $t = 0$, v is a vehicle coordinate, w is an UTM coordinate, and $\mathbf{l}_t^c$ is a t-th ray equation in camera coordinates. After calculating all the rays from every sequence, we calculated the optimal intersecting point of these rays by using a least square method. Then, we calculated the traffic light position relative to v_0 and obtained $\mathbf{P}^w$ by applying one transformation matrix. Once all the positions of traffic lights were calculated, they were integrated into our detailed map with lane level association so that current states of traffic lights could be estimated online.

5.3.2 Camera calibration

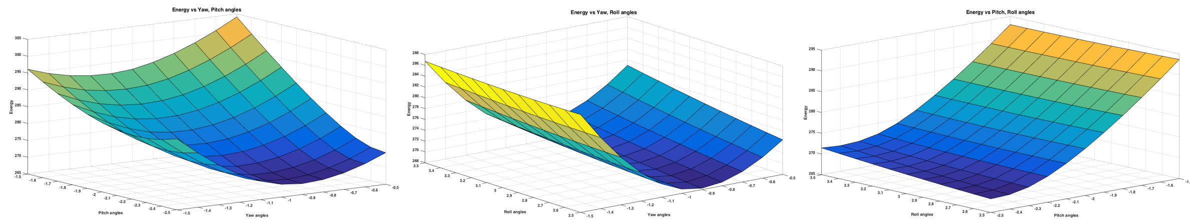
As discussed in 5.3.1, the traffic light position can be integrated into the custom map by calculating the optimal intersecting point of rays. In order to achieve the optimal point, both a sequence of vehicle poses and extrinsic and intrinsic parameters of a camera are critical. We were able to obtain very accurate vehicle poses (better than 3cm accuracy) by post-processing GPS and IMU data. The next paragraph illustrates how we calculated optimal camera parameters, which minimized the error between the actual traffic light position in an image and the projected traffic light position to the image.

We used the 'Camera Calibration Toolbox' from Caltech [Bouguet, 2015] to calculate the intrinsic parameters and distortion coefficients so that we could undistort images and use the pin hole camera assumption to calculate rays.

A more critical issue related to a camera parameter is extrinsic parameters. The extrinsic parameters contain two parts: the translation and rotation. By using self-leveling laser equipment, we could measure the translation from the center of the vehicle to the camera accurately, but there was no specific way to measure the rotation angles. So, in order to calculate the optimal rotation angles, we minimized the distance between actual pixel coordinates of traffic lights and projected pixel coordinates from a mapped traffic light by using the following equation:

$$e = \sum_{j} \sum_{i} \left\| \mathbf{p}_{i,j} - \hat{\mathbf{p}}_{j} \right\| \tag{5.5}$$

where $\mathbf{p}_{i,j}$ is an image pixel coordinate of the j-th traffic light in image i, and $\hat{\mathbf{p}}_j$ is a projected point of the j-th traffic light.



(a) Error vs Yaw and Pitch angles when roll angle is 2.5 degree (b) Error vs Yaw and Roll angles when Pitch angle is -2.5 degree (c) Error vs Pitch and Roll angles when yaw angle is -0.9 degree

Figure 5.5: Error based on different combination of angles.

We first roughly measured the initial camera angles of roll, pitch, and yaw respectively. Then, we calculated the error by changing roll, pitch, and yaw angles every 1 degree. Once we found the minimum error, we refined our approach by testing smaller angles, e.g., 0.5, 0.1 degree until this error was further minimized. Figure 5.5 shows the optimal combination of the angles which minimized the error, and we used these angles as extrinsic parameters of the camera.

5.3.3 Color Extraction

While mapping traffic signals, we extracted hue, saturation, and intensity values, which can be used to create histograms for each channel as a look-up table.

The number of traffic light pixels in an image increases as a vehicle approaches to the traffic lights. If we select all the traffic light pixels from every image from a sequence to create the histograms, it would be biased to the closer traffic lights. In order to avoid this, we selected the same number of traffic light pixels among a sequence of images when generating the histograms. For example, if the number of traffic light pixels from the furthest distance was 25, we only selected 25 pixels from every image sequence even though there were more pixels from the closer images.

By taking the approach mentioned above, we generated five different histograms: three histograms of traffic lights (red, yellow, and green) for a hue channel, one histogram for a saturation channel, and one histogram for an intensity channel. We used the OpenCV library to convert RGB to HSI color space, where the range of a hue channel is set between 0 and 180 and the ranges of saturation and intensity channels are set between 0 and 255. Thus, we used 180 bins for the hue channel while we used 255 bins for each of the saturation and intensity channels, and all of the histograms were scaled from 0 to 1.

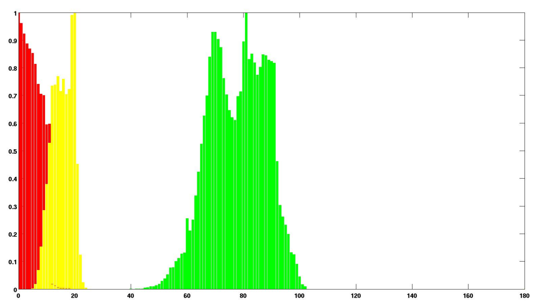


Figure 5.6: Histograms for each traffic signal in hue channel.

5.3.4 Detection

As mentioned in Section 5.3.1, traffic lights need to be associated with each lane. In order to know which traffic signal belongs to the current lane, we first looked up the current lane information in a map and found associated traffic lights that exist within 150 meters. Once we found the corresponding traffic signals, we created a region of interest (ROI) around the position of the traffic signal and projected it onto the image. Since the position of the traffic signals is in UTM coordinates, we transformed the position to the camera coordinates. Finally, we could form a bounding box, which contains the traffic signal. The image was cropped by this bounding box, and the cropped image was converted from RGB to HSI color space.

Algorithm 1 Traffic Signal State Estimator

procedure ESTIMATESINGLETRAFFICLIGHTSTATE
    - Get the current vehicle state and the current lane
    if *approaching to the intersection* **then**
        - Find a corresponding traffic signal to the current lane
        if *exist* & *distance*<150 meters **then**
            - Create a cube which center is the position of the traffic signal
            - Project the cube onto the image and set the region of interest (ROI)
            - Convert the image inside ROI from RGB to HSV color space
            - Back-project to calculate the probabilities of hue, saturation, and intensity
            - Calculate the probability of *red, yellow, green*
            - Select the state with the highest probability

Using the histograms of hue, saturation, and intensity we calculated before, we performed a back projection to calculate the probability of each pixel.

$$\begin{aligned} H_{b,red}(x,y) &= H_{red}(I(x,y)) \\ S_b(x,y) &= S_{one}(I(x,y)) \\ I_b(x,y) &= I_{one}(I(x,y)) \end{aligned} \tag{5.6}$$

where $H_{b,red}(x,y)$ is a probability density image of *red* state, and $I(x,y)$ is the value at the pixel (x,y). In order to compare three different states (*red, yellow, green*), we used three different histograms for the hue channel. After calculating these probability density images, we combined these images by

$$P_{state} = H_{b,state} \cdot S_b \cdot I_b \text{ where } state \in \textit{red,yellow,green} \tag{5.7}$$

to get the probability of each state. Based on the manual [DOT, 2009], we know that the actual diameter of the traffic signal is either 200 or 300 millimeter. So, we calculated the pixel size of the diameter of the traffic signal in an image, d, by

$$d = \frac{2Df}{L} \tag{5.8}$$

where D is the actual diameter of the traffic signal, f is the focal length of the lens, and L is the distance from the camera to the traffic signal. We created a circular filter based on the expected size of the traffic light in the image and convolved the filter with the image. Then we chose the maximum value among the three different probabilities (*red, yellow, green*) as a current state.

Even though we used the same camera to create the histogram and applied back projection, there were still a few false positives due to the orientation of the traffic signals. If traffic signals were slanted or rotated slightly, the probability of every signal was low. Furthermore, if traffic signals were hanging from a rope, the traffic signal kept moving so the probability of the traffic signals fluctuated. In order to make this smooth, we applied a low-pass filter.

There exists usually more than one dedicated traffic light to a lane. Whenever the lane has more than one dedicated traffic light, we applied the same algorithm to multiple traffic lights and merged the outputs of all the traffic lights to publish the current state. First, we checked the outputs of the traffic lights to see if they had a consensus for a specific state. If not, we found the traffic light with the maximum probability and treated it as the final state of the current intersection. However, the maximum probability was sometimes still very low. Thus, we set a threshold, t, to publish a fourth state, *unknown*, whenever the maximum probability was lower than this threshold.

5.3.5 System Integration

The CMU autonomous vehicle has several different tasks, and most tasks are coupled to one or multiple tasks. Figure 5.7 shows a few tasks that are parts of CMU's autonomous driving system

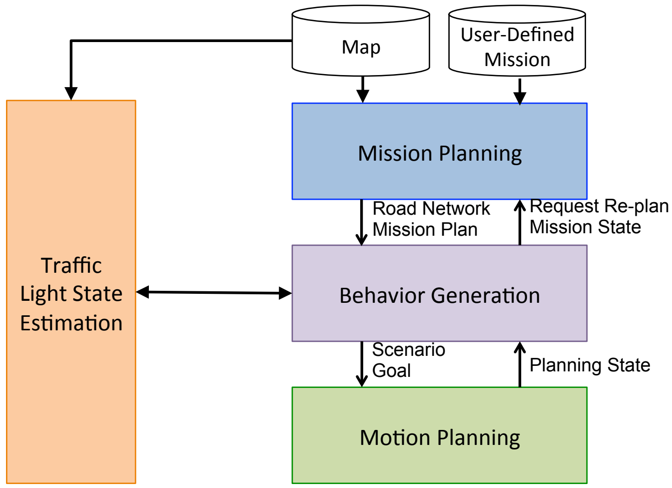


Figure 5.7: System integration

architecture. For example, the mission planning task generates a road network mission plan by utilizing a map and a user-defined mission. Then, the behavior task selects a goal based on the road network mission plan, and the motion-planning task finally generates a trajectory. Likewise, our traffic light state estimation algorithm keeps communicating with the behavior task in order to understand the current motion goal of the vehicle. Also, our traffic light state estimation algorithm reads a lane-level detailed map which contains positions of traffic lights.

When the current motion goal, a certain lane in an intersection, contains traffic lights, our traffic light state estimation algorithm publishes the current state of the traffic lights. When the distance between the traffic lights and the vehicle is greater than 150 meters, our traffic light state estimation publishes the state as 'UNKNOWN'. When the distance is less than 150 meters, our algorithm estimates the current states of traffic lights that are associated with the current motion goal and publishes the state. The information of this published state is directly fed into the behavior task and the behavior task decides what to do based on the current state of the traffic light. For example, the behavior task lets the vehicle pass the intersection if it does not have enough time to stop based on the current speed.

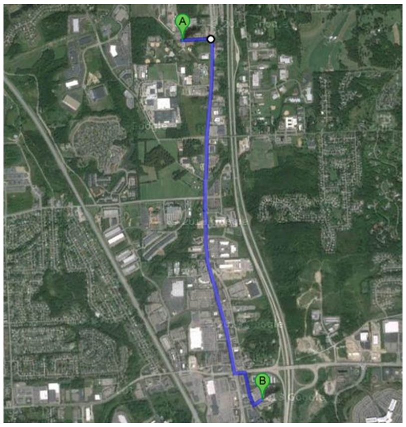
5.4 Experiments

5.4.1 Data Collection

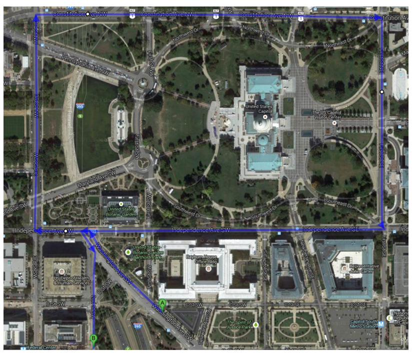
We used our autonomous vehicle to collect data and evaluate the performance. We drove three different routes as depicted in Figure 5.8 with our autonomous vehicle, which has a camera with the resolution 2448 by 2048, and we used a 6-millimeter fixed focal length lens, which provides a 54-degree field of view, with our camera. As we mentioned in the previous section, we fixed a shutter speed, a gain, and turned off all the other parameters, so that we could obtain a robust performance in various lightning conditions at different times. In order to smoothly stop when the vehicle drives at 50 MPH, our traffic signal state estimation started to execute when the vehicle was within 150 meters of the corresponding traffic signal. Our traffic signal state estimation system ran on a 2.53GHz quad core (Intel Core2 Extreme processors QX9300s) with 1GB RAM. Since our camera was grabbing images at 5Hz, we also set the operating cycle of the traffic signal state estimation to 5Hz.

5.4.2 Evaluation

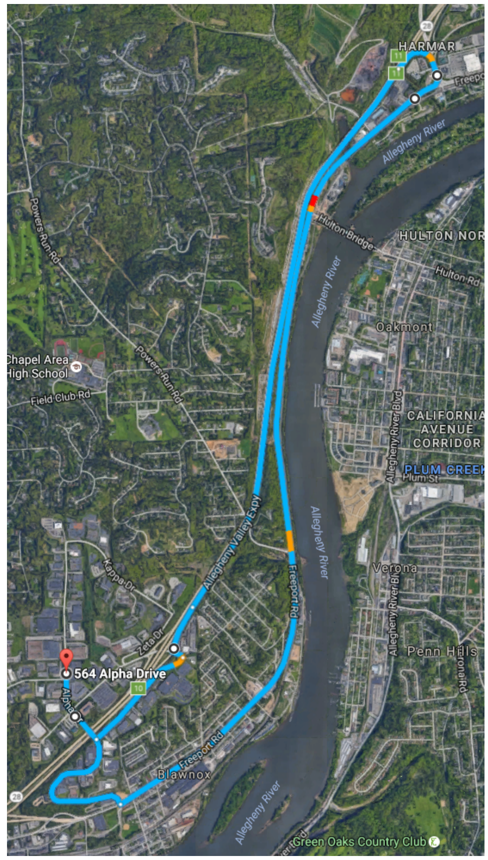
We mapped traffic signals at 46 different intersections on three routes, as depicted in Figure 5.8. The first route, shown in Figure 5.8 (a), is Route 19 in Cranberry, a suburban area inorth of Pittsburgh, Pennsylvania. This route has 3 lanes each way, and the speed limit is 45 MPH. The traffic lights on this route are usually hanging above the intersections, and each lane has 2-3 associated traffic lights. The second route, shown in Figure 5.8 (b), is located around the United States Capitol in Washington D.C, an urban area with a 25 MPH speed limit. There are 4-6 lanes (2-3 each way) in this area, and traffic lights in this urban area usually are located at the sides of the intersection. The third route, shown in Figure 5.8 (c), is Freeport Rd in Fox Chapel near Pittsburgh, Pennsylvania, a combination of rural and suburban areas with a 25-45 MPH speed



(a) Driving route in Cranberry, PA



(b) Driving route in Washington D.C.



(c) Driving route in Fox Chapel, PA

Figure 5.8: Three different routes for testing traffic signal estimation.

limit. The traffic lights are usually located either above or at the sides of the intersections. The total distance of these three routes is 15 miles, and we tested our traffic signal state estimation system under various weather conditions in order to show the robustness of our system.

We chose one specific intersection in order to describe our experimental results qualitatively as well as quantitatively, and summarized the whole performance.

As shown in Figure 5.9, the traffic light in a fixed global reference was projected onto the image coordinates by transforming the fixed global reference to the camera reference, and the

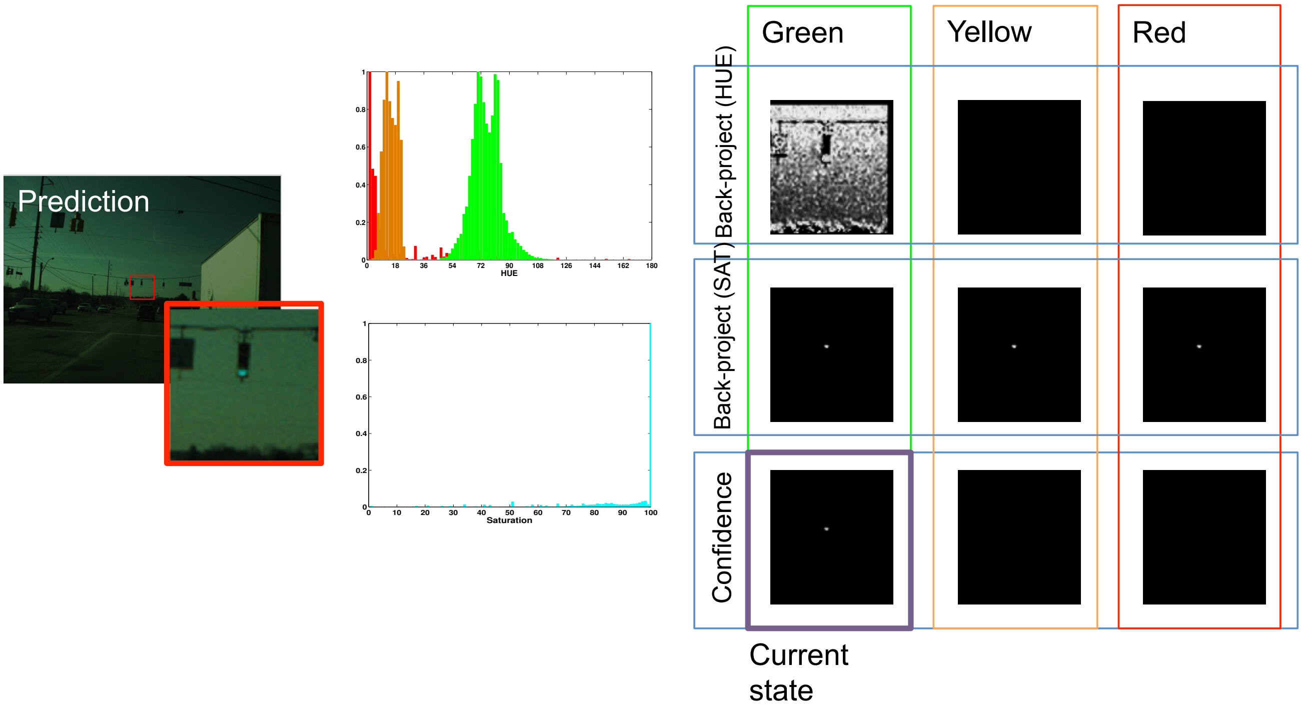
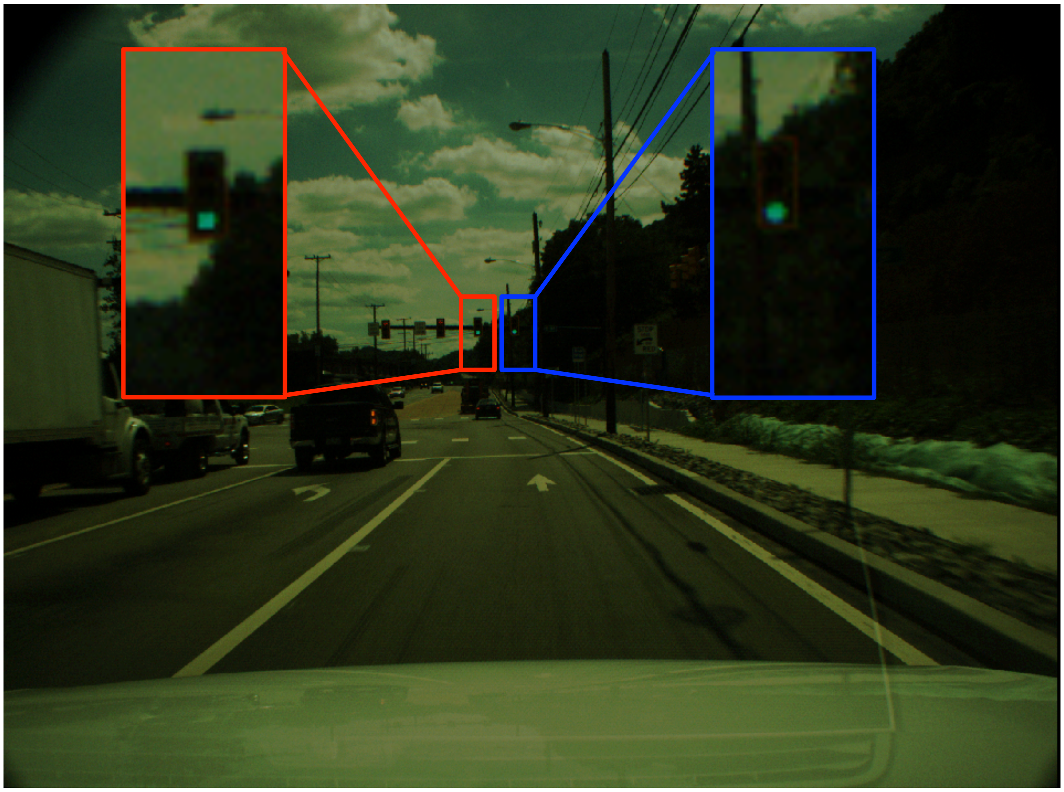


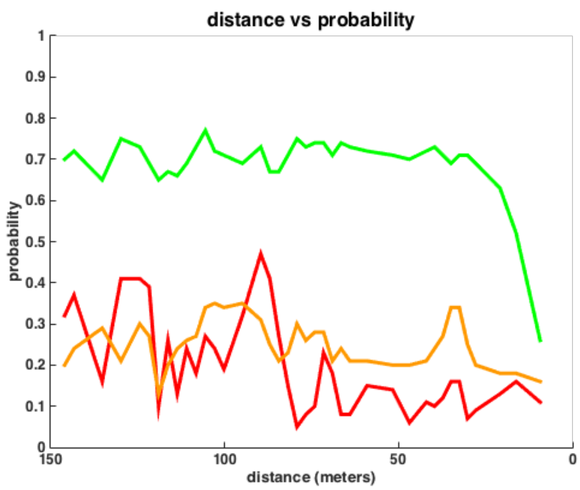
Figure 5.9: Output of a single traffic light

ROI was set. We extracted the image to back-project histograms of hue, saturation, and intensity.

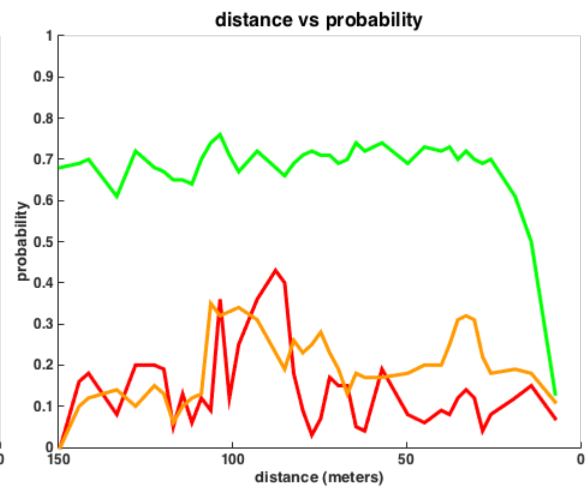
We illustrated one sequence of the outputs of the traffic lights at the intersection in Figure 5.10. Two traffic lights were green until they disappeared from the image. As explained before, a vehicle knows where it drives at the lane level, and each lane contains the positions of dedicated traffic lights. In this case, the vehicle only estimated the current states of the two traffic lights (those inside the red and blue rectangles). When the distance between a traffic light and the vehicle was less than 150 meters, the vehicle started to estimate the current states of each traffic light separately. Figure 5.10 (b) shows the probability of each state of the traffic light inside the red rectangle, and Figure 5.10 (c) shows the probabilities inside the blue rectangle. As shown in Figure 5.10 (b) and (c), the vehicle started to estimate the current state of the traffic light as green when the distance from the vehicle to the traffic light was 146 meters. The probability of green light was about 70% while the probabilities of red and yellow were 32% and 20%, respectively.



(a) Two traffic lights at an intersection



(b) Output of traffic light in the red rectangle



(c) Output of a traffic light in the blue rectangle

Figure 5.10: One example of traffic lights output at an intersection. From the map, a vehicle knows which traffic lights are dedicated to the current lane. Red and yellow lights have higher probability than an ideal case due to the presence of a brake light from a truck while estimating the state of the traffic light.

In the ideal case, the probabilities of red and yellow should be less than 20%, but some of the reflection of the Sun and a brake light from a truck increased those probabilities. However, since the pixel size of the traffic lights can be estimated based on the distance from a vehicle to the traffic lights, and the probabilities of getting the reflection of the Sun and a brake light are usually less than the probability from the traffic light, we could correctly estimate the traffic light states at most times.

As the vehicle approached the traffic lights, the probability of green states became low. As shown in Figure 5.10, the probability of the green light was less than 50 % when the distance was about 15 meter. This is because when traffic lights emit their light forward, the angle between the traffic light and a camera increases as the vehicle approches the traffic light, so the camera captures less light from the traffic light. However, 15 meters is usually less than the length of an intersection, so a vehicle can still respond to a traffic light. Even for traffic lights at a small intersection, we can still estimate their states correctly because they are usually hanging relatively lower than at bigger intersections.

In contrast to the previous result, we also illustrate a failure case in Figure 5.11. The traffic light in the yellow rectangle is closer than those in the red and blue, so our traffic light detector started to estimate the current state of the intersection using the traffic light inside the yellow rectangle. It estimated the current state correctly at first, but it gave false positives when the distance was about 120 meters. In Figure 5.12, we can see that there is a reflection of the Sun on a vehicle, and this resulted in a wrong estimation. Also, as we see Figure 5.11 (b), our traffic light detector was confused until the distance was less than 80 meters. Since the two traffic lights inside the red and blue rectangles were slanted, the camera could not capture the emitted light from the traffic lights even when the distance was greater than 80 meters, because their lights were facing downwards. However, when the distance was getting smaller, our traffic light detector could estimate the traffic lights as green, because the traffic lights started to face the camera.

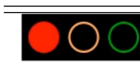
To evaluate the overall performance, we used the images captured when the traffic signals were within 150 meters from the vehicle. If there was more than one traffic light in an image, we counted each traffic light separately. Also, when traffic signals were not visible due to obstacles, we did not count those traffic signals as ground-truth. There were 3,993 traffic lights in total, and 3,897 traffic lights were estimated correctly. Table 5.2 (a) shows a confusion matrix of our traffic light detection, and Table 5.2 (b) shows precision and recall of each traffic signal. The traffic signal state estimation achieved 97.7% precision and 98.9% recall on the red traffic light. For the green traffic light, the precision and recall were 97.6% and 98.5% respectively. However, the precision and recall of the yellow traffic light were 89.7% and 42.7% respectively. As shown in the confusion matrix, all of the falsely predicted yellow traffic lights were estimated as red traffic lights (false negatives), thus with respect to the safety issue, it does not cause any problem in autonomous driving. Also, there were false positives when the state of a traffic light was green. This usually happens when there are trucks in front of a vehicle. When a traffic signal changes from red to green, trucks usually release their brakes slowly. In this case, our traffic light detector picked up their brake lights as traffic lights, and estimated the state as red.

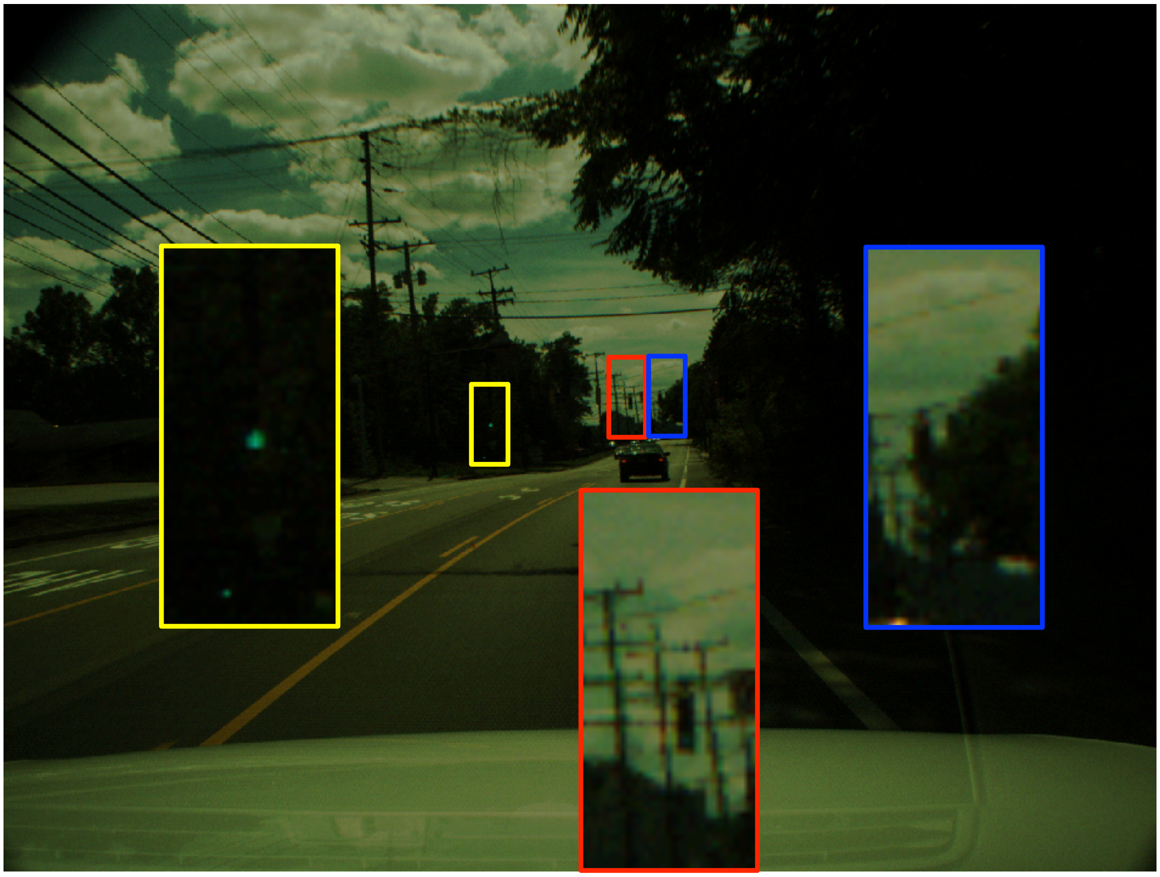
Table 5.2: The confusion matrix and precision/recall of the traffic signal state estimation system.

(a) Confusion matrix

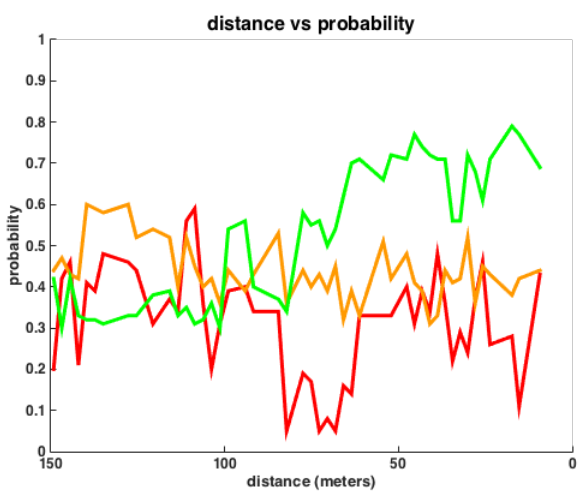
			
	8833	48	21
	45	221	7
	276	144	5872

(b) Precision and recall

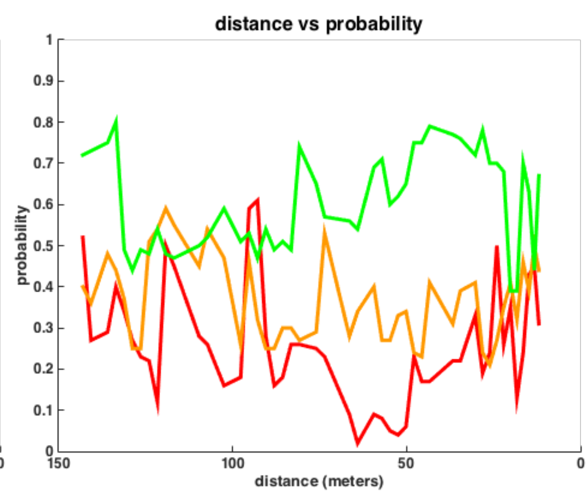
	Precision	Recall
	0.9649	0.9982
	0.5351	0.8095
	0.9953	0.9332



(a) Three traffic lights at an intersection



(b) Output of traffic light in the red rectangle



(c) Output of a traffic light in the yellow rectangle

Figure 5.11: Another example of traffic lights output at an intersection. Our traffic light state estimator failed to estimate correctly because the traffic lights are slanted.

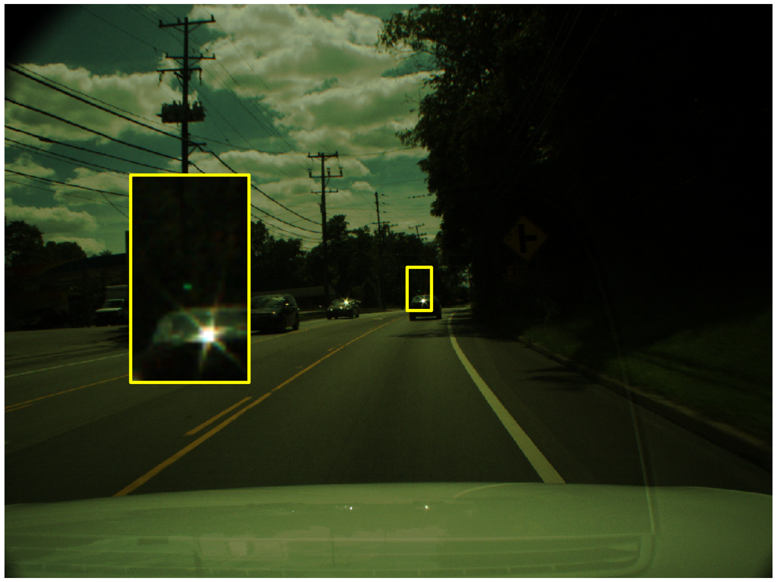
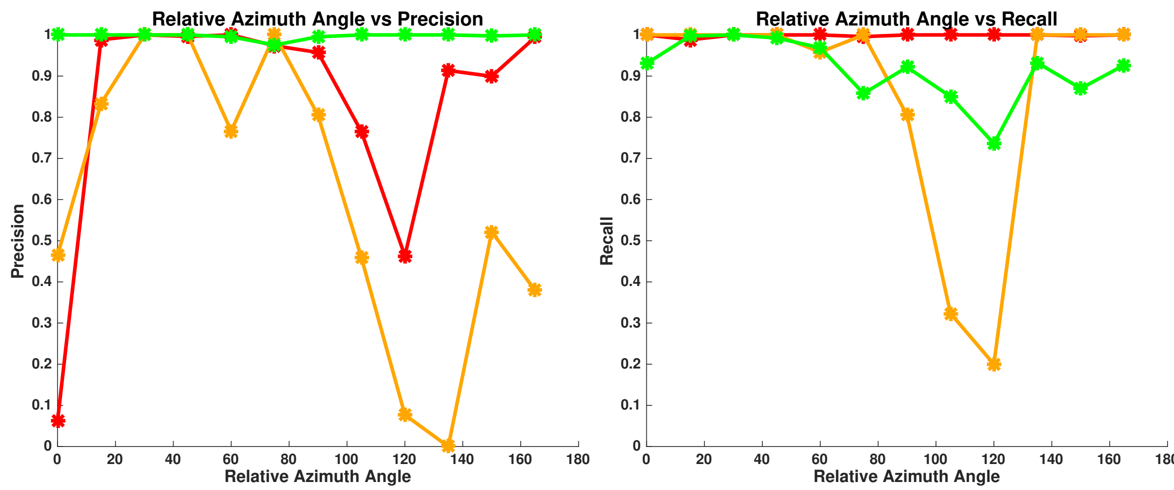


Figure 5.12: Failure case of our traffic light state estimation. Our algorithm estimated the current state as yellow due to the reflection of the Sun from a vehicle.

We also measured the performance based on the different relative azimuth angles between the Sun and the camera. When the Sun was behind the camera, we defined the relative azimuth angle as $0°$, and when the Sun was in front of the camera, we defined it as $180°$. We only considered the relative azimuth angles up to $180°$ because it did not matter whether the Sun was on the right side of the camera or on the left side.

In order to measure the performance, we drove route (c) in Figure 5.8 several times under the Sun. Table 5.3 and Figure 5.13 show the results. The minimum recall of the red lights was 0.988 when the relative azimuth angle between the vehicle and the Sun was in the range of 15-30 degrees. The precision of the red lights, however, was less than 90% when the relative azimuth angle was in the range of 0-15, 105-120, or 120-135 degrees. In Table 5.3 (a) and (i), the low precisions were mostly caused by relatively small numbers of available data for the red lights. The precision and recall of the yellow lights were lower than those of the red lights. However, it is also hard to conclude that this was a bad result, because the number of available data for these



(a) Relative azimuth angle vs Precision

(b) Relative azimuth angle vs Recall

Figure 5.13: Relative azimuth angle vs Precision and Recall

angles was relatively smaller than the others.

Most of the false positives from the green lights were identified as red. We observed the recall of the green lights was not as good as that of the red lights. As explained earlier, most false positives from the green lights were caused by brake lights from buses or trucks in front of the host vehicle. Even after traffic lights changed from red to green, because most drivers braked for several seconds longer, our traffic light detector responded to their brake lights and estimated the current state of the traffic lights as red. Once the brakes were released, our traffic light detector could correctly estimate the current state as green.

There were certain cases where our traffic light detector caused some issues. For example, when our vehicle stopped at an intersection because of a red light, a truck that was traveling in perpendicular traffic instantaneously blocked the traffic light. At this instant, the system gave false positives by estimating the state as green, and the vehicle started to move. In order to avoid these false positives, it is important to consider such occluded situations.

In this chapter, we presented our traffic light state estimation algorithm, which consists of mapping and estimation. Our algorithm first detects a traffic signal in an image sequence and

Table 5.3: The confusion matrix of the traffic signal state estimation system based on different relative angles.

(a) Between 0° and 15°

	1	0	0
	0	26	0
	15	30	603

(b) Between 15° and 30°

	84	1	0
	0	5	0
	1	0	391

(c) Between 30° and 45°

	685	0	0
	0	0	0
	0	0	286

(d) Between 45° and 60°

	875	0	0
	0	3	0
	3	0	399

(e) Between 60° and 75°

	1213	0	0
	0	23	1
	0	7	217

(f) Between 75° and 90°

	1219	0	5
	0	16	0
	33	0	199

(g) Between 90° and 105°

	811	0	0
	14	66	2
	22	16	448

(h) Between 105° and 120°

	312	0	0
	23	11	0
	73	13	488

(i) Between 120° and 135°

	19	0	0
	8	2	0
	14	24	106

(j) Between 135° and 150°

	349	0	0
	0	0	0
	33	7	553

(k) Between 150° and 165°

	456	0	1
	0	12	0
	51	11	415

(l) Between 165° and 180°

	176	0	0
	0	22	0
	1	36	468

tracks the detected traffic signal in the subsequent images. It also calculates the positions of traffic lights by using the sequence of ray equations and finding the optimal intersection points of those rays, which are integrated into our map with lane-level association. Next, the state estimation algorithm extracts hue, saturation and intensity values from the collected images to create corresponding histograms. These histograms are used to calculate the probability of each traffic state (red, yellow and green), and the final state is estimated based on the maximum probability.

In order to enhance detection of dimmed traffic lights, we first examined the frequencies of the traffic signals and set the parameters of the camera properly. Also, given that our algorithm generated the mapping information in advance, it can better detect and estimate the traffic light states at a greater distance (150 meters) and reduced computational cost so it can be integrated into the autonomous driving system.

The precision and recall of estimating red and green traffic lights were above 97%. The few false positives raised from estimating the green lights were mostly caused by the brake lights from the trucks that were in front of our vehicle. We observed relatively more false negatives when estimating the yellow traffic lights, as our algorithm sometimes estimated the yellow lights as red. However, this result does not raise any safety concern for autonomous driving.

We also evaluated the performance of our algorithm under various relative azimuth angles between the Sun and the camera. We could successfully estimate the green and red lights with high precision, but because there were relatively fewer data available for the yellow lights, their precision was not as high as that of the other states. Similar to our previous result, most false positives were caused by the delayed brake lights of the trucks or buses that were in front of our vehicle.

Chapter 6

Conclusion

6.1 Summary of the work

In this thesis, various computer vision and machine learning techniques were explored and evaluated in an effort to improve color-based object detection and classification. We focused on recognizing road signs, especially temporary traffic control signs in construction zones, and estimating traffic lights to be deployed in autonomous driving vehicles in the real world.

In order to see the potential of estimating the color of road signs correctly, we first examined different color spaces and several machine learning techniques using the real data collected under various weather conditions. We observed that hue represented pure color, and saturation represented the perceived intensity of a specific color. By converting color space and exploiting hue and saturation channels, the orange colors from construction zone signs could be detected more reliably. Those detected signs were transformed to log-polar coordinates to extract more information from the less distorted region, which is near the center of the sign, and get more robust features. Given these features, we used a multi-class classifier to predict the types of workzone signs. By integrating all these algorithms into the system of the autonomous driving vehicle, we could demonstrate that the vehicle successfully responded to a mock construction zone; slowed down while approaching the workzone, avoided channelizers in the workzone, and sped up after passing the workzone.

Even though we used different computer vision and several machine learning techniques to reduce the variation of the orange colors and classify them correctly, we found these techniques were not able to fully handle every situation. We learned that the Sun position and illumination condition are key factors in the discrimination of road signs. Thus, we formed an appearance model based on the geometrical relationship between the Sun and road signs by understanding how colors appear in images. Then, we could correctly estimate the target color (the orange color) based on the Sun position and the current heading angle of the camera. This approach especially increased the distinction among red, orange, and yellow colors so that the performance of sign detection could be improved by 7% in precision and 12% in recall.

Furthermore, we identified the illumination characteristics of an object which emits its own light. Because there was no reflectance from other light sources, the color of emitting objects did not vary much even when the geometrical relationship between light sources and objects was changed. By fixing all the parameters of the camera, we could generate consistent look-up tables of red, yellow, and green colors from traffic signals. Given all the positions of traffic signals on a route, we could successfully estimate states of those traffic signals by back-projecting these look-up tables onto the region of interest in images while driving. Furthermore, we could robustly estimate the current state of the intersection by combining outputs from multiple traffic signals.

6.2 Contributions

The research in this thesis has five main contributions:

1. Exploited different color space, machine learning techniques, and appearance-based tracking methods that are applicable to images to reduce color variance for workzone sign recognition
2. Conducted experiments in the vehicle, and showed the capability that the vehicle responds to the workzone
3. Explored the appearance model based on the geometrical relationship between the Sun and road signs and improved the road sign recognition given the Sun position and illumination conditions
4. Designed a traffic light state estimation to work robustly under various illumination conditions
5. Implemented the traffic light state estimation system and demonstrated its capability to respond correctly to the current state of the traffic light using an autonomous vehicle

1. **Exploited different color space, machine learning techniques, and appearance-based tracking**

   Given that there are no other sensors to understand the pose of a camera (location and orientation), we investigated various data-driven color detection techniques that are applicable to images for sign recognition. We experimented with different color spaces to reduce color variations and applied machine learning techniques to achieve the best performance of the color classifier. Also, we exploited kernel-based tracking in a sequence of images so that even in the presence of slight color variations, objects can be robustly tracked in consecutive images. Furthermore, we detected not only workzone signs, but also workzone channelizers in images and localized them relative to a vehicle so that the vehicle was able to follow new construction road patterns.

2. **Showed the capability to respond to the workzone in an autonomous vehicle**
   We evaluated not only the performance relative to the object detection, but also the performance relative to the autonomous driving safety. All the algorithms developed were integrated into an autonomous driving vehicle and demonstrated in a mock workzone site and other real word workzones so that an autonomous driving vehicle is able to respond to temporary traffic controls. The vehicle successfully reduced its speed after recognizing that it is approaching to the workzone, avoided channelizers and followed the new pattern of the road inside the workzone, and sped up to the normal driving speed after passing through the workzone. We believe that this research would greatly benefit not only the driver's safety, but also improve the capability of an autonomous driving vehicle so that it can be operated in unexpected situations. This unique application shows the capability of the self-driving vehicle to be deployed in the real-world.

3. **Explored the appearance model based on the geometrical relationship between the Sun and road signs**
   We showed that Sun position and illumination conditions are key factors in understanding colors projected onto an image, and in the discrimination of road signs. In order to understand the variation of colors, thus correctly estimate color, we investigated a function of color, and the geometrical relationship of the Sun, a camera, and an object. After compensating the position of the Sun, the performance of the workzone sign recognition was increased, and specifically the number of false positives was reduced. All the devel-

oped algorithms were integrated into an autonomous driving vehicle, which estimated the current Sun position, and calculated the geometrical relationship among the Sun position, a camera, and an object, so that the color of an object can be estimated. We showed the improved performance of compensating the position of the Sun's light, which had not been done yet to the best of our knowledge.

4. **Designed a traffic light state estimation**

   We first identified the characteristics of an object which emits its own light, and investigated how to reduce the variation of its color. We realized that a traffic light is a special traffic control object, which has a cover above each signal. Because the cover blocks most of the other light sources, we assumed that there was no reflectance from other light sources. Also, given that a traffic signal emits its own light at a specific frequency, the camera should have slower shutter speed than the frequency in order to remove the out-of-frequency effect.

   Once the camera was set correctly, we mapped all the locations of traffic lights in a route semi-automatically, and extracted the color information from data. Then, we applied a data-driven statistical approach to estimate the current state of a traffic light. We showed that the performance is consistent based on the different Sun positions. By using these algorithms, our self-driving vehicle is able to operate in several different areas.

5. **Demonstrate the capability to respond correctly by using an autonomous vehicle**

   We successfully integrated the traffic light state estimation system into a car. Given the current lane information, this system detects multiple traffic lights that are associated with this lane, and fuses all the information from each of the traffic lights to increase the robustness of this system. This system can estimate the states of traffic lights as far away as 170 meters in order to stop the vehicle smoothly while driving at speeds as high as 72KMPH (45MPH).

6.3 Future Work

This thesis focused on predicting the target color of temporary traffic control devices, especially construction zone signs, because they indicate a deviation from the expected road configuration and this means that the prior road model held by the autonomous vehicle may not match the current situation. We believe our work can be extended to detect various objects on the road such as stop signs, pedestrian crossing signs, information signs, etc.

In this thesis, we described various applications of machine learning. Recently, new research in convolutional neural networks has been applied to sign classification. Deep learning can estimate the unknown model, and choose the important features. Some researchers applied this deep learning algorithm to recognize traffic signs, especially classify traffic signs [Zeng et al., 2015], [Ciresan et al., 2012], [Zhu and Liang, 2016]. Deep learning requires a lot of labeled data to train, and its training computational cost is very expensive. Moreover, deep learning needs to tune various parameters, so it is hard to find a network with the best performance. However, deep learning is known to show impressive performance gains in many areas, such as computer vision, speech recognition, and some text analysis. We would like to expand our classification by creating different datasets based on the illumination. Since we can estimate the color of a traffic sign based on the illumination, we can create a lot of different training dataset by changing the illumination rather than collecting all the data and labeling them. We believe it is possible that the types and performance of target sign classification can be improved if the deep learning technique is applied.

Additionally, the research presented in this thesis analyzed the target color of traffic control devices only under the Sun. In particular, the basic assumption of this thesis was that objects should have direct Sun light so that the objects have surface and body reflections to generate a target color. If we can understand 3-D structures and shadows on the objects, we can further improve the performance of estimating the target color by excluding the surface reflection and using body reflection to estimate the target color.

Furthermore, we only estimated how the color is changed under the direct Sun in daytime. However, given that the illumination usually changes dramatically at twilight, the time between dawn and sunrise or between sunset and dusk, we can enhance our appearance estimation of objects if we understand the illumination at twilight. There are a couple of different phases in twilight, such as blue hour and golden hour. In these phases, it is hard to model the color of the sky, which is the global illumination. For instance, the color of the sky is the combination of red, orange, and yellow at the golden hour while the color of the sky is the gradient of colors from blue to orange during the blue hour. This challenge requires more data from these specific phases as well as further investigation on the dominant illumination.

Lastly, the Sun is not the only light source on the road, especially when the natural illumination is low. The dominant light source during this period is from the vehicle headlights, but there are street lights as well. It is easy to assume that the headlights' illumination is a point light source, and that of the street lights is global illumination. However, this assumption is often invalid if there are direct reflections of the street lights on objects. If we understand the locations of the street lights and their reflections on the object, there is the potential that the target color can be estimated even when the natural illumination is low.

We believe these future works can further extend our algorithms, which were demonstrated in an autonomous vehicle with a great performance, to many other applications.

Bibliography

[Administration, 2014] Administration, F. H. (2014). Guidelines for the Use of Variable Speed Limit Systems in Wet Weather - Safety — Federal Highway Administration. 5

[Alvarez et al., 2008] Alvarez, J., Lopez, A., and Baldrich, R. (2008). Illuminant-invariant model-based road segmentation. In *Intelligent Vehicles Symposium*. 2.2

[Anthony, 2016] Anthony (2016). Pittsburgh, your Self-Driving Uber is arriving now. 1.1

[Bahlmann and Zhu, 2005] Bahlmann, C. and Zhu, Y. (2005). A system for traffic sign detection, tracking, and recognition using color, shape, and motion information. *IEEE Intelligent Vehicles Symposium*, pages 255–260. 3.1

[Barnes and Zelinsky, 2004] Barnes, N. and Zelinsky, A. (2004). Real-time radial symmetry for speed sign detection. *2004 IEEE Intelligent Vehicles Symposium*. 3.1

[Bianco and Schettini, 2012] Bianco, S. and Schettini, R. (2012). Color Constancy Using Faces. In *Computer Vision and Pattern Recognition*, number 1, pages 1–12. 2.2.2

[Bigelow, 2016] Bigelow, P. (2016). Delphi Intends To Deliver Fully Autonomous Cars By 2022. 1.1

[Bouguet, 2015] Bouguet, J.-Y. (2015). Camera Calibration Toolbox for Matlab. 5.3.2

[Broggi et al., 2007] Broggi, A., Cerri, P., Medici, P., Porta, P. P., and Ghisio, G. (2007). Real time road signs recognition. In *IEEE Intelligent Vehicles Symposium*, number section III, pages 981–986. 3.1, 3.1

[Buluswar and Draper, 1998] Buluswar, S. D. and Draper, B. a. (1998). Color machine vision for autonomous vehicles. *Engineering Applications of Artificial Intelligence*, 11:245–256. 2.2.2, 2.3, 2.2.2

[Cai et al., 2012] Cai, Z., Gu, M., and Li, Y. (2012). Real-time arrow traffic light recognition system for intelligent vehicle. *International Conference on Image Processing, Computer Vision, and Pattern Recognition*. 5.1

[Charette and Nashashibi, 2009] Charette, R. D. and Nashashibi, F. (2009). Real time visual traffic lights recognition based on spot light detection and adaptive traffic lights templates. *IEEE Intelligent Vehicles Symposium*, pages 358–363. 5.1

[Chung et al., 2002] Chung, Y., Wang, J., and Chen, S. (2002). A vision-based traffic light detection system at intersections. *Journal of Taiwan Normal ...*, 47(1):67–86. 5.1

[Ciresan et al., 2012] Ciresan, D., Meier, U., Masci, J., and Schmidhuber, J. (2012). Multi-column deep neural network for traffic sign classification. *Neural Networks*, 32:333–338. 3.1, 6.3

[Comaniciu et al., 2003] Comaniciu, D., Ramesh, V., and Meer, P. (2003). Kernel-Based Object Tracking. *Pattern Recognition and Machine Intelligence*, 25(5):564–577. 3.3, 3.3

[Continental, 2016] Continental (2016). Pioneering Automated and Autonomous Driving: Continental Accepts the Challenges. 1.1

[Corke et al., 2013] Corke, P., Paul, R., Churchill, W., and Newman, P. (2013). Dealing with shadows: Capturing intrinsic scene appearance for image-based outdoor localisation. *IEEE International Conference on Intelligent Robots and Systems*, pages 2085–2092. 2.2

[DARPA, 2004] DARPA (2004). Grand Challenge. 1.1

[DARPA, 2005] DARPA (2005). Grand Challenge. 1.1

[DARPA, 2007] DARPA (2007). Urban Challenge. 1.1

[Diaz-Cabrera, 2012] Diaz-Cabrera, M. (2012). Suspended traffic lights detection and distance estimation using color features. ... *Systems (ITSC), 2012 ...*, pages 1315–1320. 5.1

[DOT, 2009] DOT, U. (2009). *Manual on Uniform Traffic Control Devices*. Number May. (document), 3.5, 5.3.4

[Ebner, 2007] Ebner, M. (2007). *Color Constancy*. 2

[Fairfield and Urmson, 2011] Fairfield, N. and Urmson, C. (2011). Traffic light mapping and detection. *IEEE International Conference on Robotics and Automation*, pages 5421–5426. 5.1

[Finlayson and Drew, 2001] Finlayson, G. and Drew, M. (2001). 4-Sensor Camera Calibration for Image Representation Invariant To Shading, Shadows, Lighting, and Specularities. *Proceedings Eighth IEEE International Conference on Computer Vision. ICCV 2001*, 2. 2.2.2, 2.2.2

[Finlayson and Hordley, 2001] Finlayson, G. and Hordley, S. (2001). Color constancy at a pixel. *Journal of Optical Society of America A*, pages 1–28. 2.3

[Finlayson et al., 2004] Finlayson, G. D., Drew, M. S., and Lu, C. (2004). Intrinsic Images by Entropy Minimization. In *European Conference on Computer Vision*, pages 582–595. 2.2.1, 2.2

[Finlayson et al., 2006] Finlayson, G. D., Hordley, S. D., Lu, C., and Drew, M. S. (2006). On the removal of shadows from images. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 28:59–68. 2.2.1, 2.2

[Freund and Schapire, 1996] Freund, Y. and Schapire, R. R. E. (1996). Experiments with a New Boosting Algorithm. *International Conference on Machine Learning*, pages 148–156. 3.2

[Friedman et al., 2000] Friedman, J., Hastie, T., and Tibshirani, R. (2000). Additive logistic regression: A statistical view of boosting. 3.2

[Gao et al., 2008] Gao, X., Hong, K., Passmore, P., Podladchikova, L., and Shaposhnikov, D. (2008). Colour Vision Model-Based Approach for Segmentation of Traffic Signs. *Journal on*

Image and Video Processing, 2008:1–7. 3.1, 3.1

[Gao et al., 2006] Gao, X. W., Podladchikova, L., Shaposhnikov, D., Hong, K., and Shevtsova, N. (2006). Recognition of traffic signs based on their colour and shape features extracted using human vision models. *Journal of Visual Communication and Image Representation*, 17(4):675–685. 3.1, 3.1

[Gehler et al., 2008] Gehler, P., Rother, C., and Blake, A. (2008). Bayesian color constancy revisited. *Computer Vision and ...*, pages 1–8. 2.3, 2.2.2

[Gijsenij and Gevers, 2011] Gijsenij, A. and Gevers, T. (2011). Color constancy using natural image statistics and scene semantics. *IEEE transactions on pattern analysis and machine intelligence*, 33(4):687–98. 2.3, 2.2.2

[Gil Jiménez et al., 2008] Gil Jiménez, P., Bascón, S. M., Moreno, H. G., Arroyo, S. L., and Ferreras, F. L. (2008). Traffic sign shape classification and localization based on the normalized FFT of the signature of blobs and 2D homographies. *Signal Processing*, 88(12):2943–2955. 3.1

[Gong et al., 2010] Gong, J., Jiang, Y., Xiong, G., Guan, C., Tao, G., and Chen, H. (2010). The recognition and tracking of traffic lights based on color segmentation and CAMSHIFT for intelligent vehicles. *2010 IEEE Intelligent Vehicles Symposium*, pages 431–435. 5.1

[Google, 2016] Google (2016). Google Self-Driving Car Project. 1.1

[Gordon-Bloomfield, 2016] Gordon-Bloomfield, N. (2016). Bosch: Well Have Fully Autonomous, Connected Vehicles In Four Years Time. 1.1

[Greenhalgh and Mirmehdi, 2012] Greenhalgh, J. and Mirmehdi, M. (2012). Real-Time Detection and Recognition of Road Traffic Signs. *IEEE Transactions on Intelligent Transportation Systems*, 13(4):1498–1506. 3.1

[Howard, 2016] Howard, B. (2016). 2017 Mercedes-Benz E-Class extends the boundaries of self-driving at Detroit auto show. 1.1

[Hwang et al., 2006] Hwang, T.-H., Joo, I.-H., and Cho, S.-I. (2006). Detection of traffic lights for vision-based car navigation system. *Advances in Image and Video …*, pages 682–691. 5.1

[Kanade et al., 1986] Kanade, T., Thorpe, C., and Whittaker, W. (1986). Autonomous land vehicle project at CMU. *CSC '86 Proceedings of the 1986 ACM fourteenth annual conference on Computer science*. 1.1

[Kang, 2015] Kang, Y. (2015). Semiautonomous Hyundai car will hit the streets this year. 1.1

[Keller et al., 2008] Keller, C. G., Sprunk, C., Bahlmann, C., Giebel, J., and Baratoff, G. (2008). Real-time Recognition of U.S. Speed Signs. *IEEE Intelligent Vehicles Symposium*. 3.1

[Kim et al., 2007] Kim, Y. K., Kim, K. W., and Yang, X. (2007). Real Time Traffic Light Recognition System for Color Vision Deficiencies. *2007 International Conference on Mechatronics and Automation*, pages 76–81. 5.1

[Lafuente-Arroyo et al., 2005] Lafuente-Arroyo, S., Gil-Jimenez, P., Maldonado-Bascon, R., Lopez-Ferreras, F., and Maldonado-Bascon, S. (2005). Traffic sign shape classification evaluation I: SVM using distance to borders. *IEEE Intelligent Vehicles Symposium*, pages 557–562. 3.1

[Lafuente-Arroyo et al., 2010] Lafuente-Arroyo, S., Salcedo-Sanz, S., Maldonado-Bascón, S., Portilla-Figueras, J., and López-Sastre, R. J. (2010). A decision support system for the automatic management of keep-clear signs based on support vector machines and geographic information systems. *Expert Systems with Applications*, 37(1):767–773. 3.1

[Levinson et al., 2011] Levinson, J., Askeland, J., Dolson, J., and Thrun, S. (2011). Traffic light mapping, localization, and state detection for autonomous vehicles. *2011 IEEE International Conference on Robotics and Automation*, pages 5784–5791. 5.1

[Lindner et al., 2004] Lindner, F., Kressel, U., and Kaelberer, S. (2004). Robust recognition of traffic signals. *IEEE Intelligent Vehicles Symposium, 2004*, pages 49–53. 5.1

[Lopez and Fuentes, 2007] Lopez, L. and Fuentes, O. (2007). Color-based road sign detection and tracking. In *International Conference On Image Analysis and Recognition*, pages 1138–1147. 3.1

[Loy and Barnes, 2009] Loy, G. and Barnes, N. (2009). Fast Shape-based Road Sign Detection for a Driver Assistance System. *IEEE Intelligent Vehicles Symposium*. 3.1

[Maddern et al., 2014] Maddern, W., Stewart, A., McManus, C., Upcroft, B., Churcill, W., and Newman, P. (2014). Illumination invariant imaging: Applications in robust vision-based localisation, mapping and classification for autonomous vehicles. In *IEEE International Conference on Robotics and Automation*. 2.2

[Maldonado-bascón et al., 2007] Maldonado-bascón, S., Lafuente-arroyo, S., Gil-jiménez, P., Gómez-moreno, H., and López-ferreras, F. (2007). Road-sign detection and recognition based on support vector machines. *IEEE Transactions on Intelligent Transportation Systems*, 8(2):264–278. 3.1

[Marchant and Onyango, 2000] Marchant, J. a. and Onyango, C. M. (2000). Shadow-invariant classification for scenes illuminated by daylight. *Journal of the Optical Society of America. A, Optics, image science, and vision*, 17(11):1952–1961. 2.2

[Marshall, 2016] Marshall, A. (2016). BMWs Bold Plan to Make a Fully Self-Driving Car by 2021. 1.1

[Meuter, 2008] Meuter, M. (2008). 3d traffic sign tracking using a particle filter. In *IEEE Conference on Intelligent Transportation Systems*, number 1, pages 168–173. Ieee. 3.1

[Miura et al., 2000] Miura, J., Kanda, T., and Shirai, Y. (2000). An Active Vision System for Real-Time Traffic Sign Recognition. In *IEEE Intelligent Transportation Systems*, pages 52–57. 3.1

[Moutarde et al., 2007] Moutarde, F., Bargeton, A., Herbin, A., and Chanussot, L. (2007). Robust on-vehicle real-time visual detection of American and European speed limit signs, with a modular Traffic Signs Recognition system. In *IEEE Intelligent Vehicles Symposium*, volume 51, pages 1122–1126. 3.1

[Muller-Schneiders et al., 2008] Muller-Schneiders, S., Nunn, C., and Meuter, M. (2008). Performance Evaluation of a Real Time Traffic Sign Recognition System. In *IEEE Intelligent Vehicles Symposium*, pages 79–84. 3.1

[National Highway Traffic Safety Administration, 2013] National Highway Traffic Safety Administration (2013). National Highway Traffic Safety Administration Preliminary Statement of Policy Concerning Automated Vehicles. *National Highway Traffic Safety Administration*, page 14. 1.1

[Nienhuser, 2010] Nienhuser, D. (2010). Visual state estimation of traffic lights using hidden Markov models. . . . *Systems (ITSC), 2010* . . . , pages 0–5. 5.1

[Nunn et al., 2008] Nunn, C., Kummert, A., and Muller-Schneiders, S. (2008). A novel region of interest selection approach for traffic sign recognition based on 3D modelling. In *IEEE Intelligent Vehicles Symposium*, pages 654–659. Ieee. 3.1

[Overett and Petersson, 2011] Overett, G. and Petersson, L. (2011). Large Scale Sign Detection using HOG Feature Variants. *IEEE Intelligent Vehicles Symposium*. 3.1

[Pettersson et al., 2008] Pettersson, N., Petersson, L., and Andersson, L. (2008). The Histogram Feature A Resource-Efficient Weak Classifier. *IEEE Intelligent Vehicles Symposium*, pages 678–683. 3.1

[Ponce et al., 2006] Ponce, J., Berg, T., Everingham, M., Forsyth, D., Hebert, M., Lazebnik, S., Marszalek, M., Schmid, C., Russell, B., Torralba, a., Williams, C., Zhang, J., and Zisserman, A. (2006). Dataset issues in object recognition. *Toward Category-Level Object Recognition*, (i):29–48. 3.6.2, 4.4.2

[Ratnasingam and Collins, 2010] Ratnasingam, S. and Collins, S. (2010). Study of the photodetector characteristics of a camera for color constancy in natural scenes. *Journal of Optical Society of America A*, 27(2):286–294. 2.3, 2.2.2

[Ratnasingam et al., 2013] Ratnasingam, S., Collins, S., and Hernández-Andrés, J. (2013). Analysis of colour constancy algorithms using the knowledge of variation of correlated colour temperature of daylight with solar elevation. *EURASIP Journal on Image and Video Processing*, 2013:14. 2.3, 2.2.2

[Ruta et al., 2007] Ruta, A., Li, Y., and Liu, X. (2007). Towards Real-Time Traffic Sign Recognition by Class-Specific Discriminative Features. In *British Machine Vision Conference*, pages 24.1–24.10. British Machine Vision Association. 3.1

[Ruta et al., 2010] Ruta, A., Li, Y., and Liu, X. (2010). Real-time traffic sign recognition from video by class-specific discriminative features. *Pattern Recognition*, 43(1):416–430. 3.1

[Shen and Ozguner, 2009] Shen, Y. and Ozguner, U. (2009). A robust video based traffic light detection algorithm for intelligent vehicles. *Intelligent Vehicles ...*, pages 521–526. 5.1

[Siogkas and Dermatas, 2006] Siogkas, G. and Dermatas, E. (2006). Detection, Tracking and Classification of Road Signs in Adverse Conditions. *IEEE Mediterranean Electrotechnical Conference (MELECON)*, pages 537–540. 3.1

[Sridharan and Stone, 2007] Sridharan, M. and Stone, P. (2007). Color Learning on a Mobile Robot: Towards Full Autonomy under Changing Illumination. In *The International Joint Conference on Artificial Intelligence*, number January. 2.3, 2.2.2, 2.2.2

[Thompson, 2016a] Thompson, C. (2016a). Elon Musk: Tesla's fully autonomous cars will 'blow people's minds'. 1.1

[Thompson, 2016b] Thompson, C. (2016b). Volvo's first self-driving car has a big edge over the competition and it's coming sooner than you think. 1.1

[Timofte and Prisacariu, 2011] Timofte, R. and Prisacariu, V. (2011). Combining Traffic Sign Detection with 3D Tracking Towards Better Driver Assistance. *Emerging topics in ...*. 3.1

[Tsin and Collins, 2001] Tsin, Y. and Collins, R. (2001). Bayesian Color Constancy for Outdoor Object Recognition. *Computer Vision and Pattern Recognition*. 2.3, 2.2.2, 2.2.2

[Vazquez-Reina et al., 2005] Vazquez-Reina, A., Lafuente-Arroyo, S., Siegmann, P., Maldonado-Bascon, S., and Acevedo-Rodriguez, F. J. (2005). Traffic sign shape classification based on correlation techniques. *International Conference on Signal Processing, Computational Geometry & Artificial Vision*, 2005:149–154. 3.1

[Wandell, 1993] Wandell, B. (1993). Color appearance: the effects of illumination and spatial pattern. *Proceedings of the National Academy of Sciences of the United States of America*, 90(November):9778–84. 3.2

[Weber, 2016] Weber, H. (2016). Fords self-driving car is totally normal. 1.1

[Zeng et al., 2015] Zeng, Y., Xu, X., Fang, Y., and Zhao, K. (2015). Traffic sign recognition using extreme learning classifier with deep convolutional features. *The 2015 international conference on intelligence science and big data engineering (IScIDE 2015), Suzhou, China*. 3.1, 6.3

[Zhu and Liang, 2016] Zhu, Z. and Liang, D. (2016). Traffic-Sign Detection and Classification in the Wild. *Cvpr*, pages 2110–2118. 6.3