Thesis for the Degree of Doctor of Philosophy

Dynamic Obstacle Detection of Road Scenes using Equi-Height Mosaicking Image

Min Woo Park

School of Computer Science and Engineering
The Graduate School

December 2013

The Graduate School
Kyungpook National University

Dynamic Obstacle Detection of Road Scenes using Equi-Height Mosaicking Image

Min Woo Park

School of Computer Science and Engineering
The Graduate School

Supervised by Professor Soon Ki Jung

Approved as a qualified thesis of Min Woo Park
for the degree of Doctor of Philosophy
by the Evaluation Committee

December 2013

Chairman [illegible]

[illegible]

Soon Ki Jung

Woo Jin Lee

Dongkyun Kim

The Graduate School Council
Kyungpook National University

Contents

List of Figures

List of Tables

Chapter 1

Introduction

1.1 Background

Today, the automobile is widely used as the most popular form of public transportation, which it has been since shortly after it was first developed by Nicolas Joseph Cugnot [1]. As the performance of the automobile has improved, the area that each person can be active within has widened by vehicle usage. Thereby, the quality of people's lives has also been improved by the automobile. However, many people are also killed or injured by traffic accidents. Therefore, automobile companies and researchers have spent a lot of resources developing many different safety systems to reduce traffic accident fatalities.

As computer and electronics technologies have advanced in recent years, many researchers have developed intelligent safety systems. Especially, recent research has focused on developing advanced driver assistance systems (ADASs) such as blind spot warning (BSW) systems [2] or forward collision warning systems (FCWS) [3]. These safety systems, which use sensors to protect both the driver and pedestrians from a driver's mistakes, are called active safety systems. In active safety systems, vehicle detection or traffic information recognition technologies are used. Therefore, many automobile companies have produced intelligent vehicles by installing active safety systems to prevent traffic accidents.

In recent years, significant research has been focused on a vision-based active safety system that prevents collisions with pedestrians and vehicles in advance. Vision-based sensors are preferable because they are low cost and can be used for a variety of purposes depending on the software algorithm. In automobiles, a vision-based system is mainly used as a general-purpose device performing multiple functions such as collision warning and blind spot visualization. For example, a night vision system using an infrared camera [4] and a 360 degree view monitoring system [5] are already used for safe driving. Most of all, recent studies have focused on dynamic obstacle detection that can detect and identify other vehicles or pedestrians using one or two cameras as sensors [3] to prevent forward collisions by driver error.

Despite all the research effort put into dynamic obstacle detection, however, it has not yet been commercialized in the automobile industry.

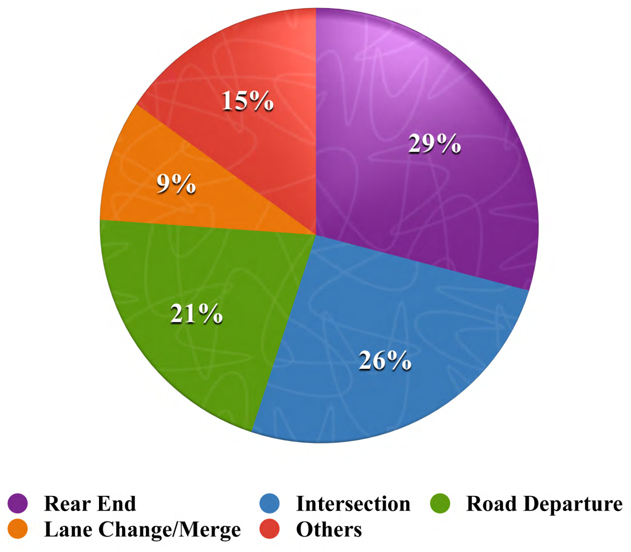


Figure 1.1: Police-Reported Light Vehicle Crashes [7].

A dynamic obstacle detection system demands fast processing and high accuracy for the safety of the driver and pedestrians. In order to increase the processing speed, many hardware acceleration algorithms have been designed, for example, graphic processing unit (GPU) or field programmable gate array (FPGA) algorithms [6].

Therefore, we propose a robust vehicle and pedestrian detection system that works in real time on any road scene. The proposed system uses efficient new image representation to improve the processing speed of the GPU-accelerated vehicle and pedestrian detector [6]. We also present partial side detection of approaching dangerous vehicles.

1.2 Motivation and Contribution

1.2.1 Motivation

In recent years, many traffic accidents that occur are due to driver error. According to a study released by the National Highway Traffic Safety Administration (NHTSA) and the Virginia Tech Transportation Institute (VTTI), 80 percent of automobile accidents and 65 percent of near-accidents involve at least some form of driver distraction within three seconds of the crash or near-miss [8]. And, according to police-reported light vehicle crashes [7], the leading cause of light vehicle crashes is a rear-end collision by distracted driving. Furthermore, according to Traffic Accident Statics by MoDOT Safety [9], the second most common cause of total crashes is the inattention of the driver. Inattention is also the major cause of fatal crashes.

Many researchers are focused on reducing distracted driving by creating active safety systems that include features such as adaptive cruise control, collision warnings, or 360 degree view monitors. These active safety systems prevent traffic accidents by preventing distracted driving in advance. In particular, recent automobile companies and research institutes are developing the Augmented Reality-based Full-Windshield Head Mounted Display (AR-based FW-HUD) system to reduce distracted driving and visualize the driving information, as illustrated in Fig. 1.4.

In visualized driving information, information about the vehicle and

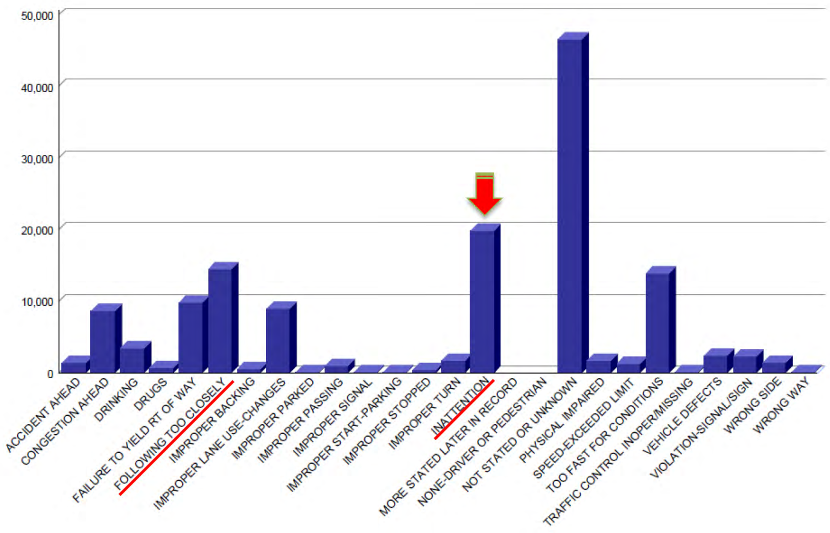


Figure 1.2: Total crash (2009 Traffic Accident Statics by MoDOT Safety).

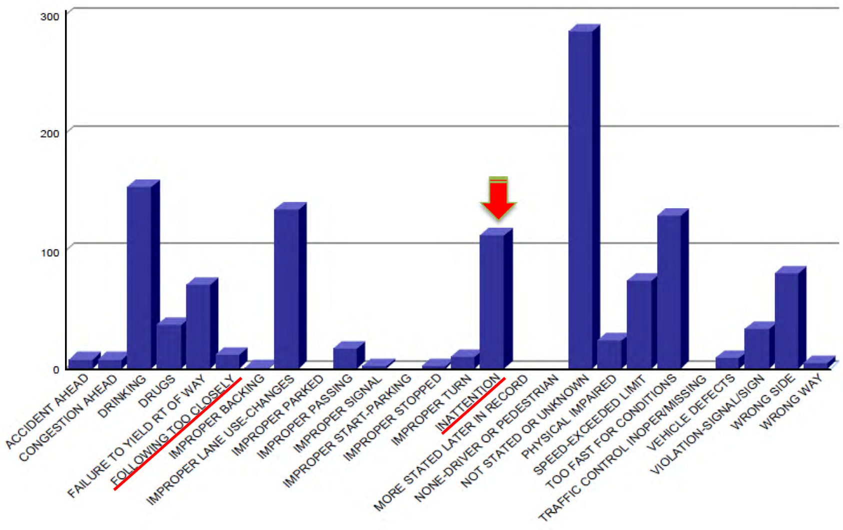


Figure 1.3: Fatal crash (2009 Traffic Accident Statics by MoDOT Safety).

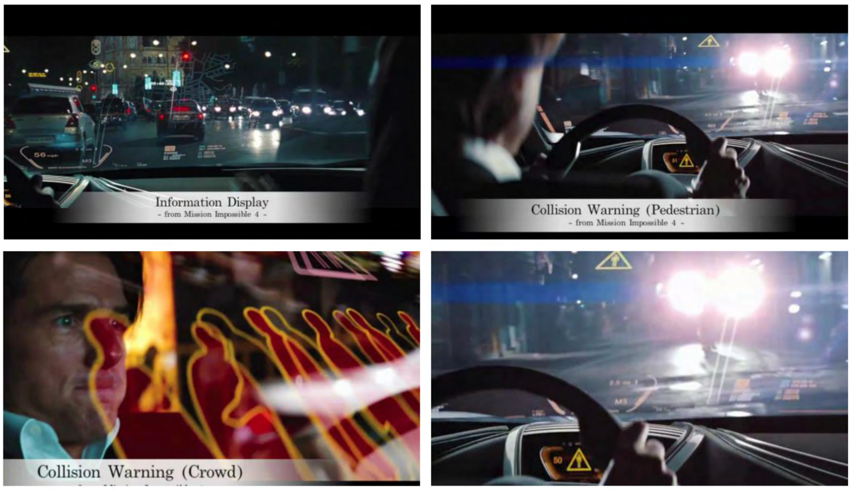

Information Display
- from Mission Impossible 4 -
Collision Warning (Pedestrian)
- from Mission Impossible 4 -
Collision Warning (Crowd)

Figure 1.4: The example of AR based FW HUD [10].

pedestrians is relatively more important than other information, because that information is related to saving people's lives. Therefore, the proposed system focuses on vehicle and pedestrian detection. In this thesis, we present a real-time dynamic obstacle detection system that implements the AR-based FW-HUD system to reduce distracted driving. Also, we make an AR-based FW-HUD simulator to effectively visualize the information of the detected vehicles and pedestrians.

1.2.2 Contribution

In this thesis, the proposed system contributes three important factors as follows:

- Reduce the computational region using equi-height image under on-

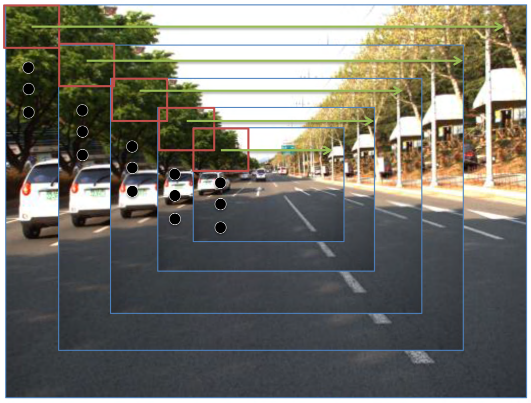


Figure 1.5: Traditional ”sliding window” based approach.

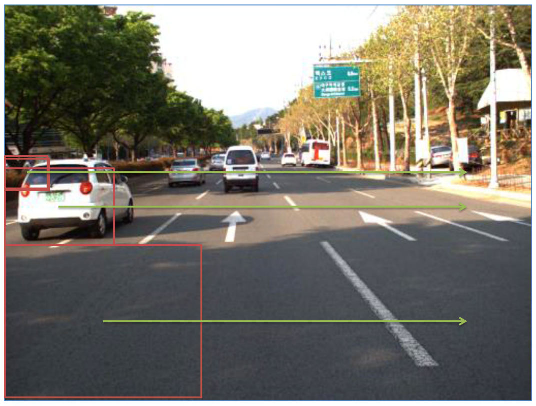


Figure 1.6: ”Equi-height image” based approach.

road geometry

- Improve the processing speed of GPU-based dynamic obstacle detection using equi-height mosaicking image

- Detect approaching dangerous vehicles using part-based side detection of the approaching vehicle

First of all, the proposed system contributes to reduce false positives using the equi-height images. In general, many researchers have suggested various detection approaches such as symmetry or edege [11, 12]. All approaches are classified into two types based on the method used to search: knowledge-based searching approaches and sliding window-based searching approaches. The knowledge-based searching approach performs detection of the selected region by information acquired from the scene. This approach dramatically reduces the computational region, but a lot of false negatives can be generated by excluding regions of the image. And false negatives are an unacceptable risk in vehicle and pedestrian detection. In addition, the approach is too sensitive in an external environment. Because obstacle detection must be performed while driving, this approach is difficult to use in a real vehicle. On the other hand, the sliding window-based searching approach is a simple and exact approach because it searches in all candidate regions. In general, this approach generates a low amount of false negatives, but it is very slow because it has many computational regions.

In the first contribution, therefore, the proposed system suggests using the improved sliding window-based approach. The proposed system already assumes that all obstacles exist on the road. Thus, the proposed system searches for regions with equal height on the road. Also, on-road obstacles with equal heights seem larger in perspective if the distance between the camera and obstacle is close. Therefore, the proposed system uses an image strip with equal heights on a 3D space, called an equi-height image. The equi-height image-based approach improves the sliding window-based approach. Because this approach removes unnecessary searching regions by assuming uniform road geometry, our approach reduces computational region. The proposed system also reduces false positives and increases precision by using equi-height images generated by road-geometry information. Fig. 1.5 and Fig. 1.6 illustrate the difference of search regions between the traditional sliding window-based approach and equi-height image-based approach.

Next, the proposed system suggests a new image representation called an equi-height mosaicking image for faster detection. The equi-height mosaicking image is a set of equi-height images. In equi-height images, a GPU-based detection algorithm performs multiple function calls for each equi-height image. This approach is inefficient to perform on a GPU. However, the proposed system performs a single function call using an equi-height mosaicking image. This image representation reduces the number of function calls to improve the processing speed on the GPU. Fig. 1.7 and

Fig. 1.8 illustrate the difference of the function calls between the equi-height image and equi-height mosaicking image. Fig. 1.8 also shows the search region of the equi-height mosaicking image-based approach.

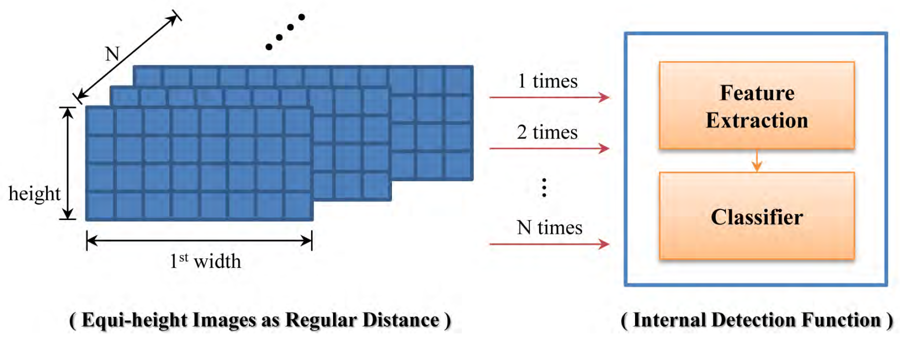

N
height
1st width
1 times
2 times
N times
Feature Extraction
Classifier
(Equi-height Images as Regular Distance)
(Internal Detection Function)

Figure 1.7: Equi-height image based multiple function call.

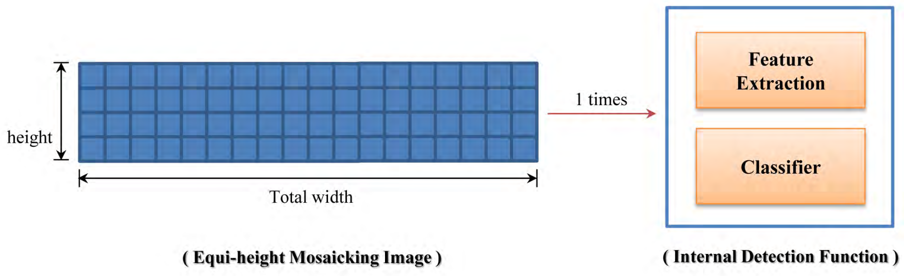

height
Total width
1 times
Feature Extraction
Classifier
(Equi-height Mosaicking Image)
(Internal Detection Function)

Figure 1.8: Equi-height mosaicking image based a single function call.

Finally, the proposed system presents a part-based side detection algorithm for approaching vehicles. Many drivers feel danger when a vehicle is passed by on the side. The passing vehicle is called the approaching vehicle. Because the vehicle has an incomplete shape when passing, general

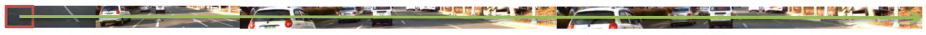


Figure 1.9: Search regions of equi-height mosaicking images based approach.

vehicle detection algorithms cannot detect approaching vehicles. Therefore, the proposed system suggests a part-based side detection algorithm for approaching vehicles. In this algorithm, the proposed system generates an equi-height peripheral mosaicking image using the equi-height mosaicking image. Then the proposed system detects the front or rear parts using a part-based side detection algorithm, and decides the region of the approaching vehicle.

1.3 Previous Researches

1.3.1 Vehicle Detection

For several years now, situation awareness systems for intelligent vehicles have been available. They detect obstacles around the vehicle, including other vehicles and pedestrians. In particular, many researchers have investigated learning-based approaches.

Zhang Sun *et al.* [13, 14] implement a single camera-based vehicle detection system using a Gabor filter-based SVM classifier. Its detection results showed a low rate of false positives using a two-step hypothesis and a Gabor filter-based learning algorithm. Navneet Dalal *et al.* [15] and Feng

Han *et al.* [16] presented a human and vehicle detection method using Histograms of Oriented Gradients (HOG). These learning-based studies show a high accuracy for object detection. Because it guarantees high accuracy, HOG is the feature used most frequently for learning in the vehicle or pedestrian detection field. It does, however, consume substantial processing time. Consequently, Victor Adrian Prisacariu *et al.* [6] and XinXin Wang *et al.* [17] implement an acceleration approach using a hardware accelerator. It was faster than CPU based non-acceleration methods. Fanjing Kong *et. al* [18] suggested Histogram of Multi-Scale Orientations (H-MSO) based vehicle detection approach. H-MSO is a modified HOG feature set for vehicle detection. It also used the SVM classifier to detect the vehicle. The H-MSO feature is more powerful to detect the vehicle. However, its detection speed is low because of using gabor filter.

1.3.2 Pedestrian Detection

Dalal and Triggs [15] and Han *et al.* [16] have invented a pedestrian detection method that uses HOG, as it shows a high accuracy for object detection. However, it also consumes a lot of processing time. Therefore, recent studies have focused on improving processing time. Prisacariu and Reid [6] have implemented an acceleration approach using a hardware accelerator. It is faster than a CPU-based non-acceleration approach. Rodrigo Beneson *et al.* [19] have created a fast GPU-based pedestrian detection method using the stixel computed by a stereo camera. Junjie Yan *et. al.* [20] suggested multiple pedestrian detection using multiple resolution training set. In general, many pedestrian detection classifier cannot almost classify a pedestrian image of low resolution when it is performed the training using trainin data of high resolution. Therefore, it makes the classifier using multiple resolution training data. It is good idea but training data are increased exponentially.

1.4 Outlines

This thesis is organized as follows. Chapter 2 describes our dynamic obstacle detection system. In this Chapter, we first explain the obstacle detection scenario and give an overview of the proposed system with online and offline processing. Chapter 3 describes the process to generate an equi-height image representation. In this chapter, we explain the calibration method and road scene analysis for equi-height images. In addition, we explain the generation method for an equi-height mosaicking image. Chapter 4 describes forward dynamic obstacle detection using an equi-height mosaicking image. We define the dynamic obstacle on the road scene and explain the learning method and detection method on equi-height mosaicking images. Chapter 5 describes part-based side detection for approaching vehicles. In this Chapter, we define the approaching vehicle from each side and explain the generation of an equi-height peripheral mosaicking image. After that, a learning and detection method is described. Chapter 6 describes its experimental setup and results for the proposed system. And Chapter 7 includes our conclusion and future work.

Appendix A describes the AR-based FW-HUD simulator applied our detected results. Additionally, we explain how to make the AR-based FW-HUD simulator. And we show the visualized result of the detected obstacles.

Chapter 2

Dynamic Obstacle Detection System

In this chapter, we illustrate the detection scenario of the proposed system as detection targets. And we also describe the detailed structure of the dynamic obstacle detection system.

2.1 Obstacle Detection Scenario

In this section, the proposed system illustrates the detection scenario for the vehicle and pedestrians. The proposed algorithm is implemented to detect vehicles and pedestrians in various road scene scenarios. However, the proposed system assumes several conditions for the road scene. The assumptions are as follows:

- The road is a flat
- Our vehicle is running or stopping on the road
- All dynamic obstacles exist on the road

And then, the proposed system determines targets for detection. The detection targets are as follows:

- The rear-end section of vehicles
- The part-based side detection for approaching vehicles
- Walking or standing pedestrians

Under the assumption, the proposed system sets up several scenarios in order to detect the above-mentioned targets. The detection scenarios are as follows:

- Several vehicles are driving in front of our vehicle on a highway
- A lot of vehicles are driving in front of our vehicle on a downtown road
- Several vehicles overtake our vehicle on a highway
- A lot of vehicles overtake our vehicle on a downtown road
- Several pedestrians cross the crosswalk in front of our vehicle

- A lot of pedestrians are walking or stopping in front of our vehicle on a narrow city street

Fig. 2.1 shows the scenario of vehicles ahead and overtaking vehicles. The dark blue vehicle is driving in front of our vehicle. The light blue vehicle is overtaking our vehicle. Under the scenario, the proposed system detects the vehicles using both forward vehicle detection and part-based side detection of the approaching vehicle.

Fig. 2.2 shows the scenario of pedestrians crossing a crosswalk. The black shapes are the walking pedestrians. The pedestrians can also be detected using the proposed system.

2.2 System Overview

In this section, we describe the real-time vehicle and pedestrian detection system using an equi-height mosaicking image. The proposed system improves the execution time of the existing GPU-based acceleration approach and reduces false positives triggered using geometry-based equi-height mosaicking images. In addition, the proposed system suggests part-based side detection of approaching dangerous vehicles. Fig. 2.3 illustrates the flowchart of the proposed system.

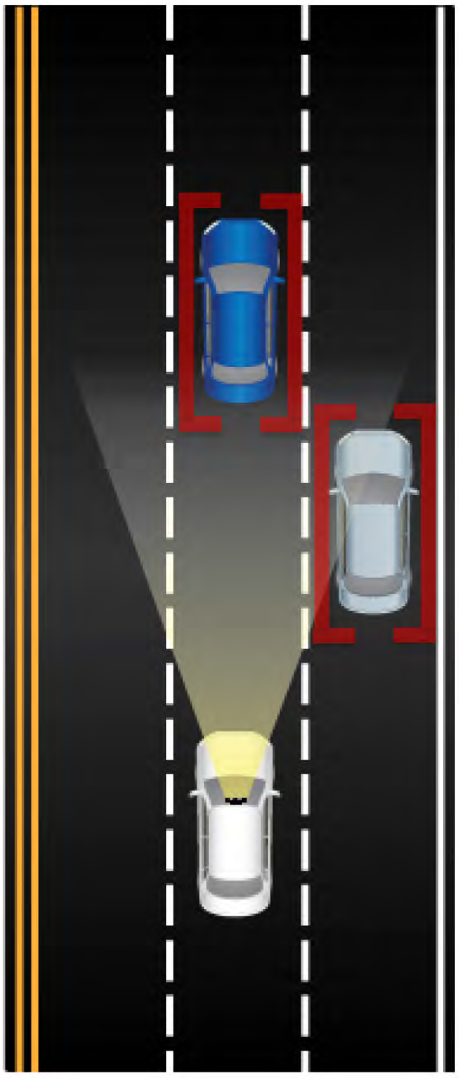


Figure 2.1: Scenario of vehicle detection.

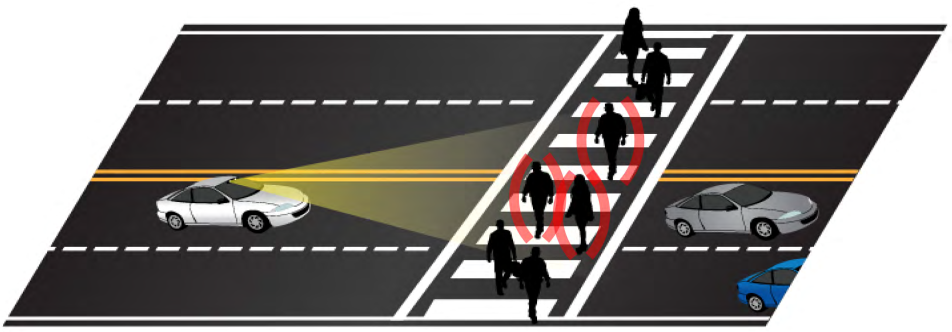


Figure 2.2: Scenario of pedestrian detection.

2.2.1 Offline Processing

The proposed system is segmented into two processing sections: online processing and offline processing. In order to perform the detection in real time, the proposed system pre-computes the initial processes offline. In offline processing, the proposed system first performs a system calibration. In this step, the proposed system generates an undistortion map using the lens distortion and skew rotation parameters. Each parameter is computed by camera calibration and skew rotation measurement. Another process is initial horizon detection, which consists of line extraction and vanishing point detection. First of all, the proposed system extracts a line on the road using Probabilistic Hough Transform [21] and computes the vanishing point from the extracted inliers using M-estimator SAmpling and Consensus (MSAC) [22, 23]. Then the proposed system estimates the horizon using the y-coordinate of the vanishing point. The other process is

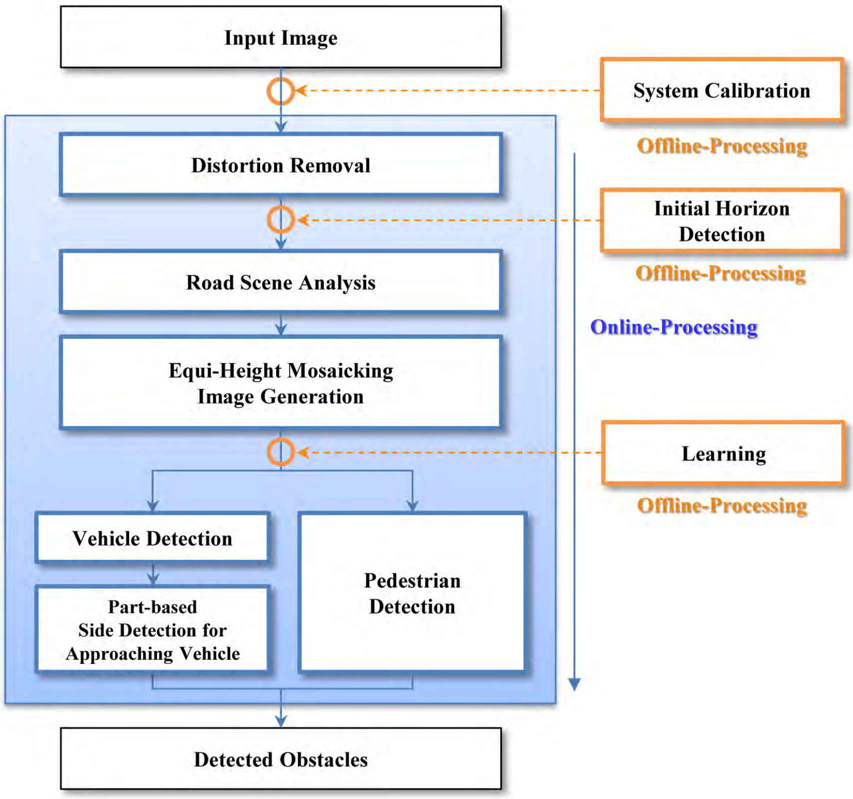


Figure 2.3: Flowchart of obstacle detection system.

learning. Because learning consumes a lot of resources for processing, the proposed system performs learning offline using training data. After the proposed system performs as many computations as possible offline, the proposed system processes an input image to detect vehicles and pedestrian online. Fig. 2.4 shows the pre-computed offline process.

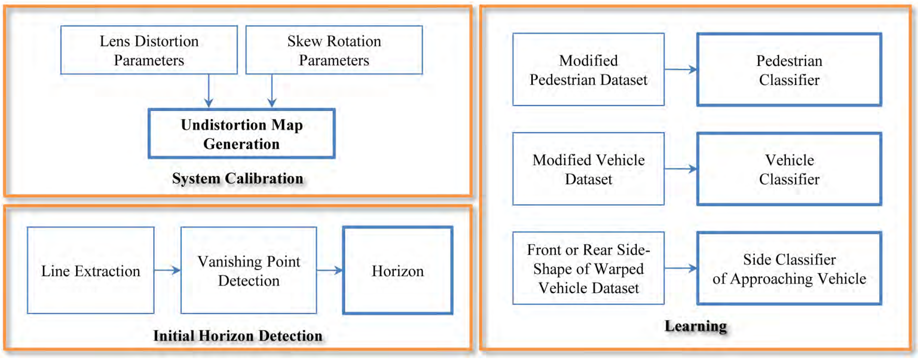

Lens Distortion Parameters
Skew Rotation Parameters
Undistortion Map Generation
System Calibration
Line Extraction
Vanishing Point Detection
Horizon
Initial Horizon Detection
Modified Pedestrian Dataset
Pedestrian Classifier
Modified Vehicle Dataset
Vehicle Classifier
Front or Rear Side-Shape of Warped Vehicle Dataset
Side Classifier of Approaching Vehicle
Learning

Figure 2.4: Precomputed off-line processes.

2.2.2 Online Processing

In online processing, the proposed system consists of the distortion removal step and three primary parts: equi-height mosaicking image generation, vehicle/pedestrian detection, and side-part detection of an approaching vehicle. First of all, the proposed system calibrates for lens distortion and removes any image skew caused by camera yawing. Before the proposed system removes the distortion, it pre-computes the distortion parameter and skew rotation parameter offline. Then it generates the undistortion map offline, which is essentially a mapping table between the distorted and undistorted images. In online processing, the proposed system just applies the undistortion map to an input image to remove distortions. Fig. 2.5 illustrates the process of distortion removal.

After the distortion removal is completed, the proposed system performs the three primary steps. The first step is equi-height mosaicking

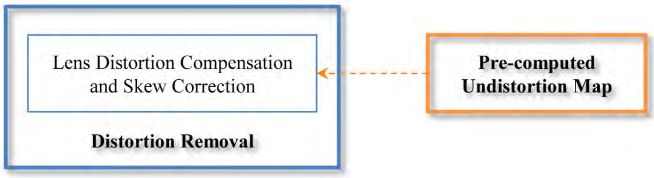

Lens Distortion Compensation and Skew Correction
Distortion Removal
Pre-computed Undistortion Map

Figure 2.5: Diagram of the distortion removal.

image generation. In this step, the proposed system first analyzes the road scene to make the equi-height images. The road scene analysis consists of hypothesis position sampling and height estimation. So the proposed system extracts the sampling position to generate the equi-height images. The sampling position means y coordinates on the image. The sampling position is extracted in 3D space as regular distance from the camera position of the vehicle. Then the proposed system estimates the height of the equi-height images on a sampled position using the horizon and reference object's height. In this case, the horizon is pre-computed using the vanishing point calculated during offline processing. After road scene analysis is completed, the proposed system generates the equi-height image using sampled positions and estimated height. In detail, the proposed system crops the image in order to make the equi-height images on sampled positions using the estimated height. Next, the proposed system concatenates the equi-height images to generate the equi-height mosaicking image. Fig. 2.6 illustrates the process of equi-height mosaicking image generation.

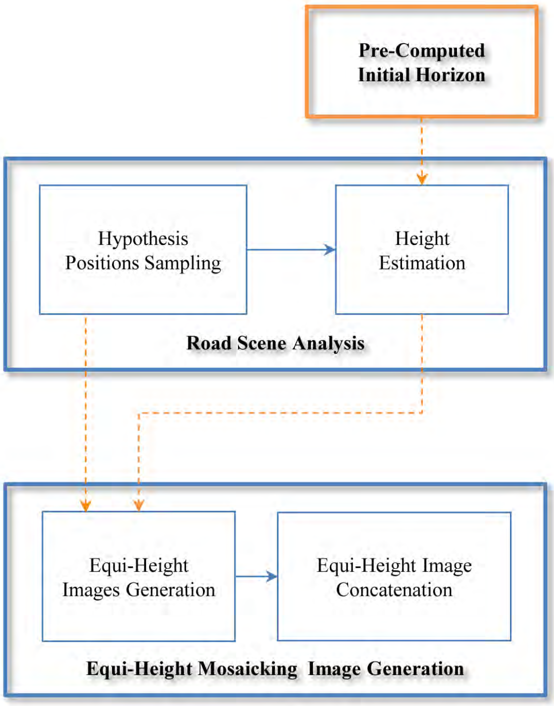

Pre-Computed Initial Horizon
Hypothesis Positions Sampling
Height Estimation
Road Scene Analysis
Equi-Height Images Generation
Equi-Height Image Concatenation
Equi-Height Mosaicking Image Generation

Figure 2.6: Process of the equi-height mosaicking image generation.

Once the equi-height mosaicking image is generated, the proposed system performs the second part. The second part is vehicle and pedestrian detection. In this step, the proposed system performs a 1D search to detect any vehicles or pedestrians in the equi-height mosaicking image. In the search process, the proposed system uses a HOG-based SVM classifier trained by the rear image of each vehicle and modified images of pedestrians in offline processing. When vehicles or pedestrians are detected, the proposed system converts the coordinates of the detected vehicles or pedes-

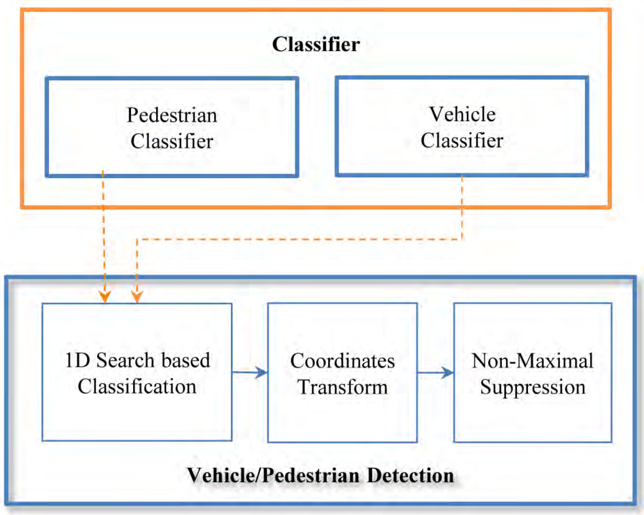


Figure 2.7: Process of the vehicle and pedestrian detection.

trians from equi-height mosaicking image space to original image space. Then the system performs grouping for multiple detected vehicles using non-maximal suppression [24]. Fig. 2.7 illustrates this process of vehicle and pedestrian detection.

The third step is part-based side detection for approaching vehicles. In this step, the proposed system performs partial detection for the side of approaching vehicles. Before the partial sides of approaching vehicles are detected, the proposed system first generates an equi-height peripheral mosaicking image. The equi-height peripheral mosaicking image is a concatenated image that is made by image warping and image concatenation for each side region of the road, similar to the process of creating an equi-height mosaicking image. After the equi-height peripheral mosaicking

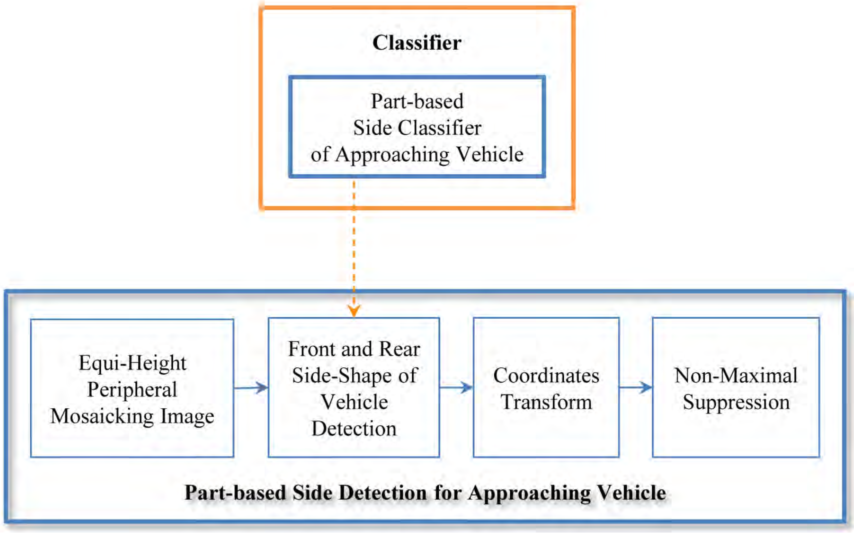

Classifier
Part-based Side Classifier of Approaching Vehicle
Equi-Height Peripheral Mosaicking Image
Front and Rear Side-Shape of Vehicle Detection
Coordinates Transform
Non-Maximal Suppression
Part-based Side Detection for Approaching Vehicle

Figure 2.8: Process of part-based side detection for approaching vehicle.

image is generated, the proposed system performs a 1D search to detect the partial side-shape of the vehicles. When the detection is finished, the proposed system performs a coordinate transform. The coordinate transform performs two transforms. The first transform is the coordinate transform between an equi-height peripheral mosaicking image and equi-height mosaicking image. The second transform is the other coordinate transform between the equi-height mosaicking image and the original image. After the coordinates are transformed, the proposed system also performs grouping using non-maximal suppression [24]. Fig. 2.8 illustrates the process of the part-based side detection algorithm for approaching vehicles.

Chapter 3

Equi-Height Mosaicking Image Generation

In this chapter, the proposed system generates the equi-height mosaicking image for GPU-based vehicle and pedestrian detection in real time [25,26]. It will be used as input during the vehicle and pedestrian detection phase. Before real-time processing begins, the proposed system preprocesses the captured image by performing a calibration for lens distortion and skew rotation. The proposed system then computes the obstacle's height on the image by analyzing the road scene. Next, it generates the equi-height image using the computed obstacle's height at all positions, which are sampled at regular distance intervals. Finally, the proposed system concatenates the equi-height images to create the equi-height mosaicking image. It then becomes the input for the real-time detection process.

3.1 Image Representation for Fast Detection

Before the equi-height mosaicking image is generated, the proposed system first describes the equi-height image and equi-height mosaicking image. In this thesis, the proposed system uses these image representations to perform vehicle and pedestrian detection in real time.

3.1.1 Equi-Height Image

In general, the proposed system can simply define that the road scene consists of the road, background, and obstacle. In this instance, almost all obstacles are running or walking on the road. If the road is flat, all obstacles existed apart from the camera at a regular distance. Therefore, it can define image strips as a depth in 3D space. Also, the image strips are perpendicular on the road. The image strip is sampled at the hypothetical position on the road geometry. This image strip is designated as the "equi-height image."

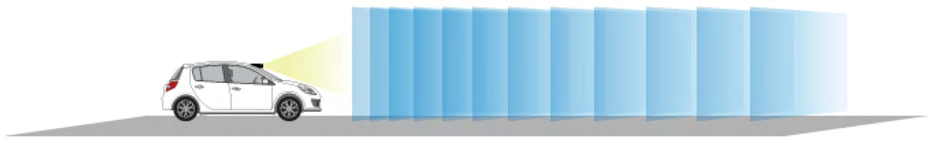


Figure 3.1: Road geometry-based equi-height image.

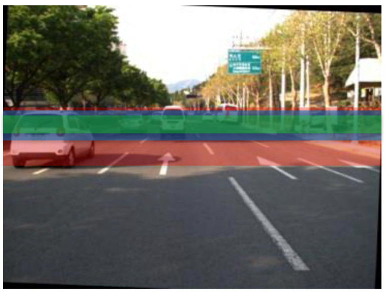
The equi-height image can be summarized as a road geometry-based image strip with equal heights. In order to generate this equi-height image, the proposed system assumes that all obstacles exist on the road and obstacles of the same type have similar heights. When the assumption is satisfied, the proposed system generates equi-height images at a regular distance.

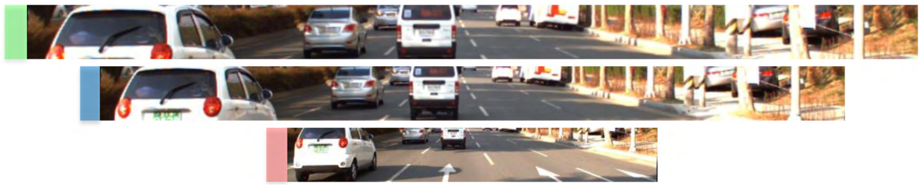
The image can be used to reduce the computational region and false positives in the obstacle detection step. The equi-height image is generated in the road scene analysis step. Fig. 3.1 illustrates the road geometry-based equi-height image generation process. The blue rectangles mean equi-height images sampled by hypothetical positions.

3.1.2 Equi-Height Mosaicking Image

As mentioned above, the equi-height image is a road geometry-based image strip with equal heights. This image representation can be used to reduce false positives when the proposed system detects an obstacle. However, the image representation is not suitable to high-speed processing for obstacle detection. Therefore, the proposed system suggests a novel image representation called an equi-height mosaicking image.

In order to present the real-time obstacle detection system in this thesis, the proposed system has to perform effective and fast processing on a GPU. However, the equi-height image-based approach is not efficient because a repetitive function call must be executed to process the many


(a) Sampling the image strips.


(b) Generated equi-height images from sampling positions.

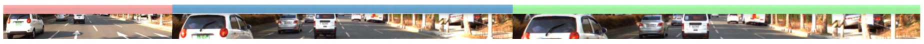

(c) Generated equi-height mosaicking image.

Figure 3.2: Process of equi-height mosaicking image generation.

equi-height images. In general, the repetitive function call is a serious bottleneck on a CPU and also a GPU. Therefore, the proposed system uses an equi-height mosaicking image as the streaming input of equi-height images. The novel image representation can be used to remove the repetitive function call. So the proposed system calls the feature extraction and detection function just once, using the equi-height mosaicking image.

Actually, the equi-height mosaicking image is a concatenated long image stream made by equi-height images. The image representation is simple, but it is powerful in the GPU-accelerated processing step. Fig. 3.2

simply illustrates the process of equi-height mosaicking image generation.

3.2 Image Distortion Removal

In this section, the proposed system removes the distortion of an input image before equi-height images are generated. The distortion removal of the input image is performed to generate the correct equi-height images. In offline processing, the proposed system pre-computes the undistortion map for lens distortion and skew rotation. In online processing, it performs the calibration of the image distortion using an undistortion map.

3.2.1 System Calibration

Before dynamic obstacle detection is performed, the proposed system must calibrate the input images in a preprocessing step. Generally, captured images from a camera have radial and tangential lens distortion. And captured images also have skew rotation because of the established camera pose. So in this step, the proposed system performs offline calibration for the lens distortion and image skew.

Calibration of Lens Distortion

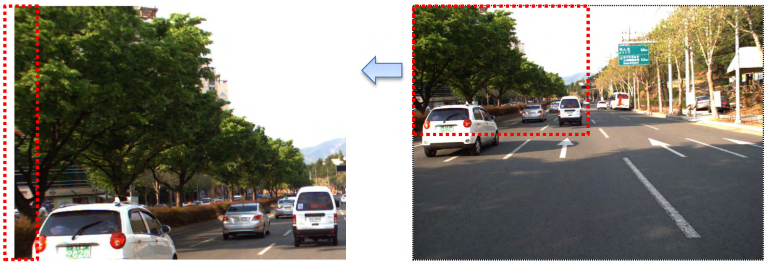
In general, the lens of a camera has both radial and tangential distortion. Even though these distortion coefficients have tiny numerical values, they still influence accurate sampling to generate equi-height images. Therefore, the proposed system must first compensate for lens distortion using intrinsic camera parameters, including lens distortion coefficients.

The intrinsic camera parameters and lens distortion coefficients can be computed by the camera calibration method [27]. The lens distortion is formulated by the following equation:

$$\begin{aligned} u = f_x * \{x(1 &+ k_1 r^2 + k_2 r^4 + k_3 r^6) \\ &+ 2p_1 xy + p_2(r^2 + 2x^2)\} + c_x, \\ v = f_y * \{y(1 &+ k_1 r^2 + k_2 r^4 + k_3 r^6) \\ &+ p_1(r^2 + 2y^2) + 2p_2 xy\} + c_y, \end{aligned} \tag{3.1}$$

where (x, y) is the coordinates of the distorted image and (u, v) is its corresponding undistorted coordinates. k_1, k_2 and k_3 are the radial distortion coefficients. p_1 and p_2 are the tangential distortion coefficients. And finally, $r^2 = x^2 + y^2$. (f_x, f_y) and (c_x, c_y) are the focal length and the optical center, respectively.

Fig. 3.3 illustrates the calibrated result of lens distortion. The top row is the original input image and the bottom row is the calibrated image. The left column is the zoomed-in image. In the zoomed-in image, the proposed system can has a calibrated result in which barrel distortion has been eliminated.


(a) Original input image.

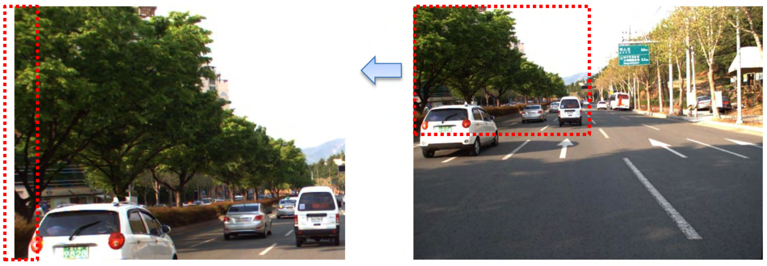

(b) Calibrated image.

Figure 3.3: Calibrated result of lens distortion.

Calibration of Skew Roatation

After the lens distortion is eliminated, the proposed system computes the angle for image skew correction. The image skew occurs when the camera does not maintain a horizontal position with respect to the road plane. In order to correct the image skew, the proposed system estimates an angle of image skew from the skewed lines and corrects the skew on the image with a two-dimensional rotation [28].

Once the angle of image skew is measured, the proposed system generates the skew rotation matrix based on the z-axis. Then it performs a

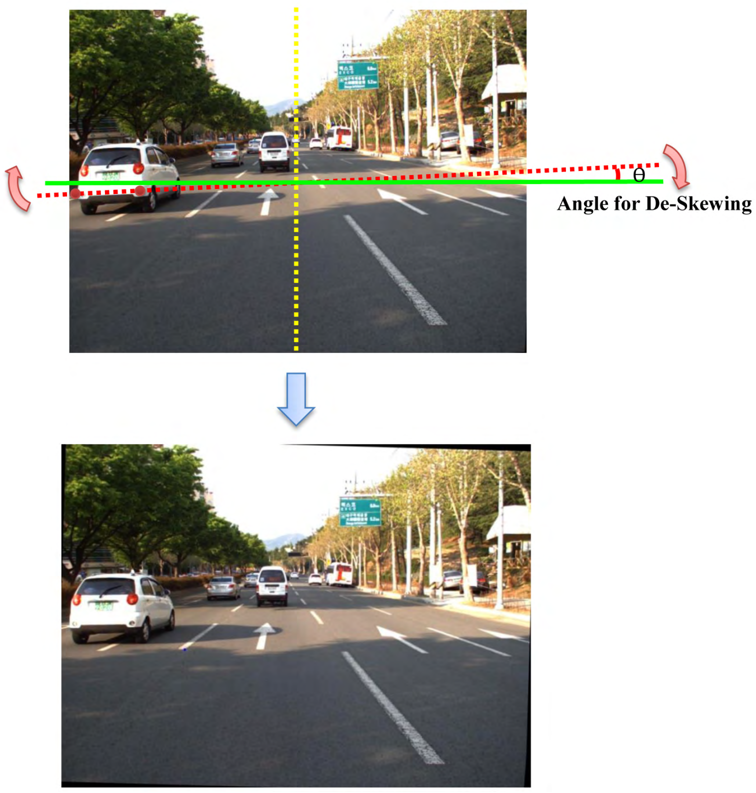


Figure 3.4: Result of the skew correction.

rotation in the camera coordinates. The skew correction is represented by the following equation:

$$R_{skew} = \begin{bmatrix} cos(\theta) & sin(\theta) & 0 \\ -sin(\theta) & cos(\theta) & 0 \\ 0 & 0 & 1 \end{bmatrix}, \qquad (3.2)$$

$$H_R = KRK^{-1}, \qquad (3.3)$$

$$I_{corr} = H_R I_{ud}, \qquad (3.4)$$

where *theta* is the measured angle of image skew, and R_{skew} means the skew rotation matrix. K is the intrinsic matrix, I_{ud} is the calibrated image for lens distortion, and I_{corr} is the corrected image for image skew. Fig. 3.4 shows the result of the skew correction. The figure of the top row shows how to measure the skew rotation from the input image. The figure on the bottom row shows the image after image skew is corrected. θ indicates the angle of de-skewing for the correction.

3.2.2 Distortion Removal using Undistortion Map

After the lens distortion and image skew calibrations are computed, the proposed system generates the undistortion map for mapping between an input image and an undistorted image. Distortion removal consumes a lot of processing time. Thus, the proposed system makes a look-up table between the input image and undistorted image. In order to generate the look-up table, it uses the pre-computed results for lens distortion and image skew on offline processing. The look-up table for distortion removal is an undistortion map.

Since converting the original image to the undistorted one is time-consuming, the undistortion map is applied in real time. Therefore, the proposed system quickly performs the calibrations for lens distortion and image skew using the undistortion map. Fig. 3.5 illustrates the undistortion map between coordinates of every pixel.

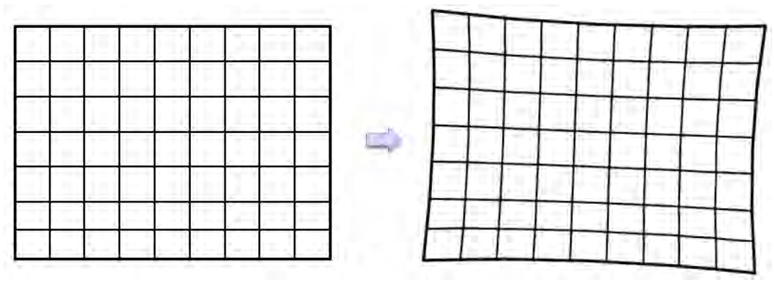


Figure 3.5: Undistortion mapping in image pixel level.

3.3 Scene Analysis for Equi-Height Image

After the distortion is removed, the proposed system acquires the information to generate equi-height images from the road scene. Therefore, it describes the definition of an equi-height image and the analysis of the road scene in this section.

In order to generate the equi-height image, the proposed system analyzes the road scene. First of all, it performs the position sampling of equi-height images on the road. Then it estimates the height of the obstacle on the sampled position. The estimated height of the obstacle is used as the height of the equi-height images on the sampled position. In the height estimation step, it first estimates the horizon by vanishing point detection during offline processing. Then it computes the height of the obstacle using the horizon during online processing.

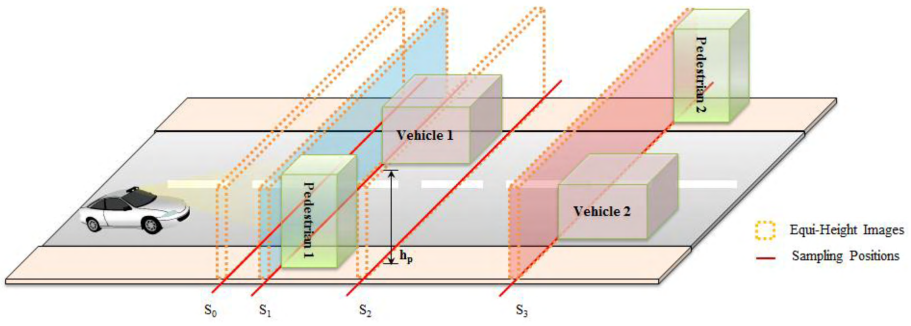


Figure 3.6: Equi-height image generation in 3D space.

3.3.1 Hypothesis Position Sampling

The first step to generate the equi-height images is hypothesis position sampling. This process extracts the y coordinates of the image at a regular distance interval from the camera position. In this case, the regular distance interval is the sampling positions $S_0 \sim S_3$ shown in Fig. 3.6.

The hypothesis positions are sampled with a regular interval based on a distance from the camera. In order to perform the hypothesis position sampling as a regular distance interval, the proposed system extracts the hypothesis positions as a bird's-eye-view image that is the rectified image of the road plane using image warping. The proposed system generates the bird's-eye-view using a reference lane mark on the road [2]. Once the bird's-eye-view is generated, the proposed system performs a sampling for the y coordinates with a regular distance interval. Fig. 3.7 shows the result of the hypothetical position sampling on the bird's-eye-view image.

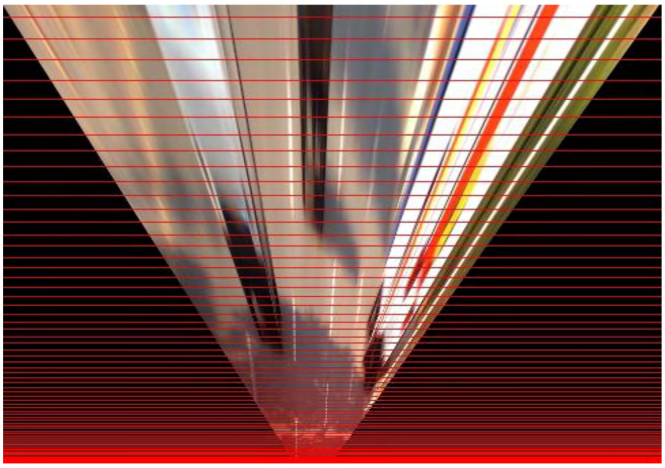


Figure 3.7: Sampled position on bird's-eye-view.

When the position sampling finishes, the sampling coordinates are inversely warped from the bird's-eye-view image to the original image. In the next step, the proposed system generates an equi-height image using the inverse warped sampling coordinates, because any obstacle may be located at the sampling positions. Fig. 3.6 illustrates the equi-height image generation of sampled positions in 3D space. S_0, S_1, S_2 and S_3 indicate sampling positions. And Fig. 3.8 illustrates the visualized result of the sampled position on a perspective image.

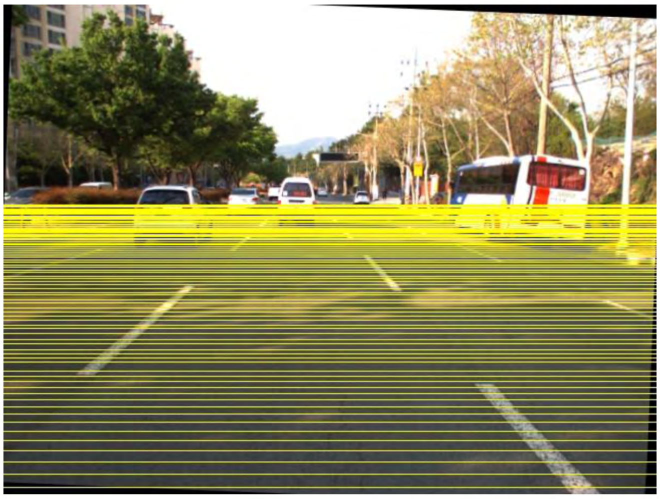


Figure 3.8: Sampled position on perspective image.

In the actual implementation, the proposed system cannot generate the bird's-eye-view image. It is only used to explain hypothesis position sampling. Actually, the proposed system just establishes the sampling positions and only computes the warping matrix. Then it performs warping for the coordinates of the sampling positions.

3.3.2 Horizon Detection

Before the height of the interested obstacle is performed, the proposed system has to detect the horizon during offline processing. In the initial step it detects the vanishing point on the image to determine the horizon. The vanishing point detection consists of line detection, clustering, and estimation. First, the proposed system extracts the lines on the image using a Probabilistic Hough Transform [21]. It then clusters them to classify inliers of the extracted lines using MSAC [22,23]. Next, the proposed system performs an estimation of the initial vanishing point using a least-square solution from the clustered lines.

When the initial vanishing point is estimated, the proposed system sets up the initial horizon using the y coordinates of the vanishing point. However, the coordinates of the initial vanishing point may be incorrect. Therefore, the system performs line filtering for the extracted lines by removing the existent lines above the initial horizon. It then performs the clustering and estimation again for a more robust estimation of the vanishing point. When the robust vanishing point is estimated, the proposed system can set up a more accurate horizon. The estimated vanishing point and horizon are used to compute the height of a vehicle. Fig. 3.9 shows the result of the vanishing point and horizon detection. In the figure, the red lines are detected using a Probabilistic Hough Transform. The red circle and green line indicate the vanishing point and horizon, respectively.

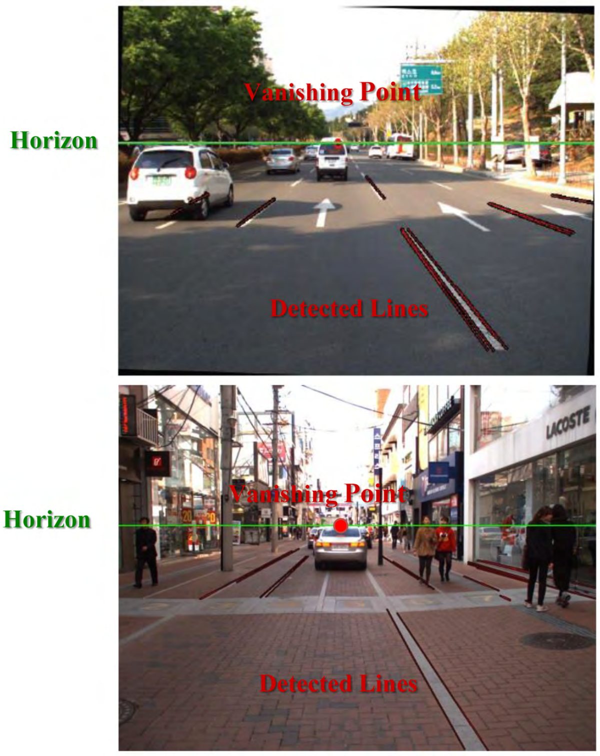


Figure 3.9: Result for estimated vanishing point and the horizon.

3.3.3 Height Estimation on Perspective Image

When the sample position and horizon have been estimated, the proposed system performs height estimation on the perspective image to generate the equi-height images. In the proposed system, it assumes that all obstacles are on the road plane and the road is flat. Using this assumption, the proposed system applies a simple height estimation method to compute the height of the vehicle candidates on the perspective image [30]. First, the proposed system measures the height of a reference obstacle and the distance between the horizon and the bottom of the reference vehicle on the image. Then it computes the height of the image at any position on the road. The height of the image is computed by the following equation:

$$R_h/(R_p - l_h) = O_h/(O_p - l_h), \tag{3.5}$$

where R_h and R_p are the reference height and position, respectively. O_h and O_p are the height and the position of the vehicle at the sampling positions, respectively. l_h represents the horizon. The estimated height is used to the height of the equi-height image generated in the sampled position. Fig. 3.10 shows the result of height estimation on a perspective image. H_1 and H_2 mean the estimated height at the sampling positions, respectively.

3.4 Equi-Height Mosaicking Image Generation

In section 3.2, the proposed system performed the sampling of the y coordinates of the image and estimated the height on the sampled position. In this section, the proposed system generates the equi-height images that are then concatenated to be used for fast detection in the GPU process.

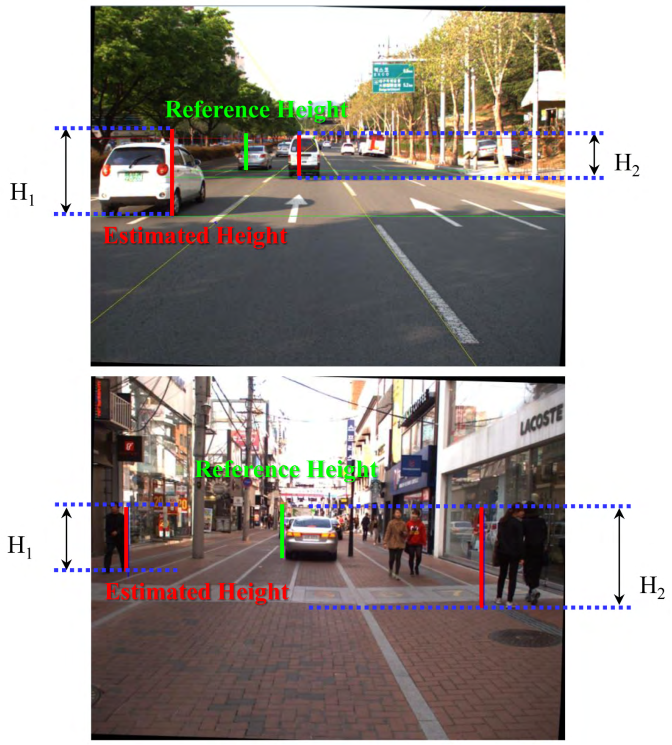


Figure 3.10: Result of height estimation on perspective image.

3.4.1 Generation of Equi-Height Images

When the road scene analysis is finished, the proposed system generates the equi-height images. Each equi-height image is generated by the projected height of each sampling position. Therefore, it uses the estimated height of the obstacle and the sampling position to generate the equi-height image. Based on each sampling position, the proposed system crops the image strip to the projected height of the obstacle in the image. Fig. 3.11 shows the generated equi-height images on the perspective image. The rectangles mean equi-height images on the image. Each red and blue rectangle is an equi-height image sampled at S_1 and S_3 positions in Fig. 3.6.

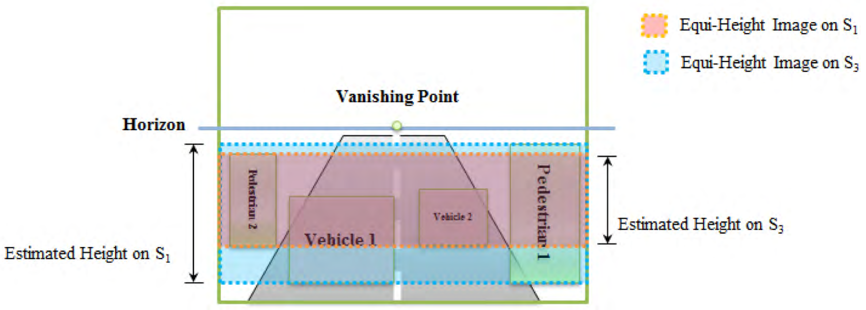
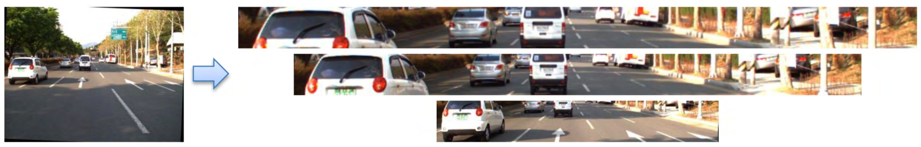
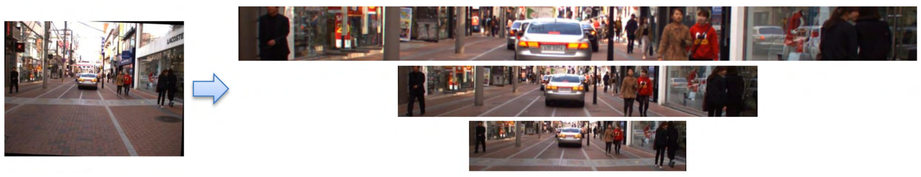


Figure 3.11: Generation of equi-height images on an image.

Fig. 3.12 also shows the generated equi-height images for three sample positions from the input images of two types. The left images are the calibrated input images and the right images are several equi-height images.



(a) Case of the vehicle.



(b) Case of the pedestrian.

Figure 3.12: Generated equi-height images.

After the equi-height image is generated, the proposed system visualizes the generated equi-height images in 3D space as illustrated in Fig. 3.1. The visualized equi-height images show how to detect the obstacle in the proposed system. Fig. 3.13 and Fig. 3.14 show the visualized equi-height images as scenes of two types of 3D space.

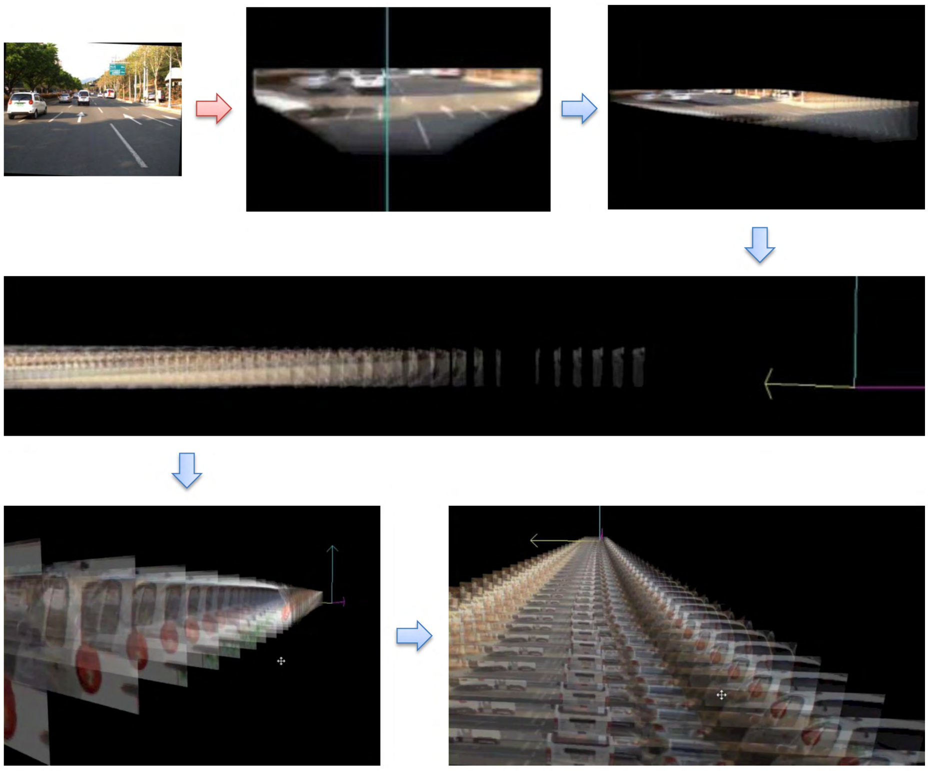


Figure 3.13: 3D visualization of equi-height images on road scene.

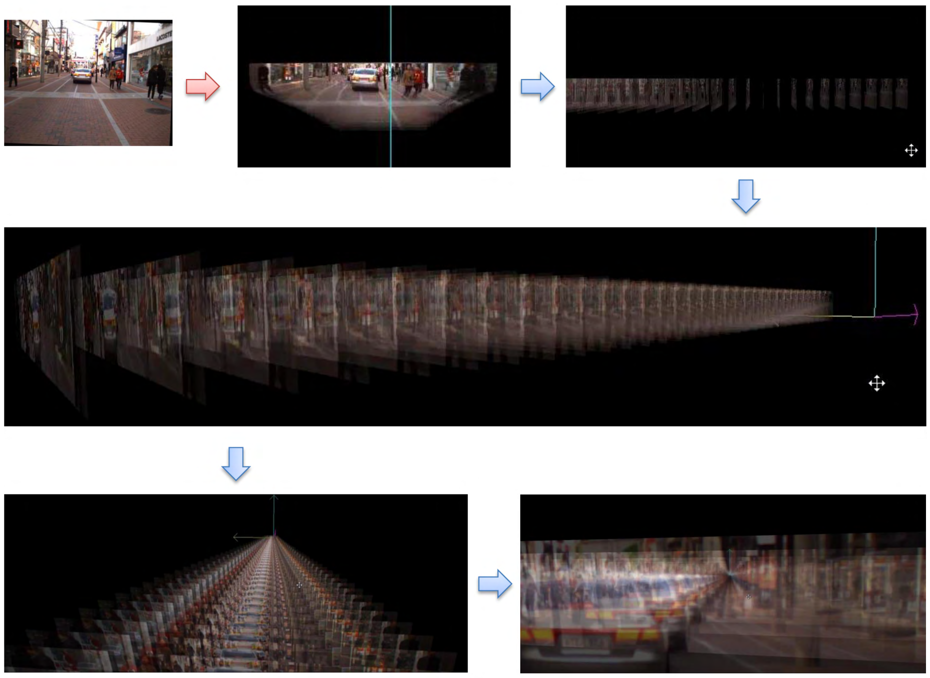
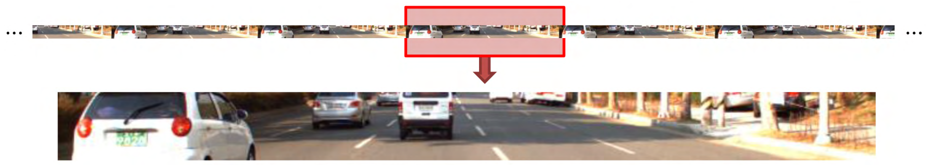


Figure 3.14: 3D visualization of equi-height images on downtown.

3.4.2 Mosaicking of Equi-Height Images

Finally, the proposed system concatenates equi-height images into a panorama-like image to quickly detect the obstacle. The concatenated image is called the Equi-height Mosaicking Image. In the previous section, the proposed system generates an image strip cropped on the basis of the height of a normal obstacle called the equi-height image. Then it concatenates these images in a panorama-like manner. In order to concatenate the equi-height images, the proposed system resizes the equi-height images to a fixed image height that is equal to the size of the training data. After the equi-height images are resized, it performs the concatenation. The generated equi-height mosaicking image is used as the input for the obstacle detection process. Fig. 3.15 shows the generated equi-height mosaicking image for two types of scenes. The generated equi-height mosaicking image is used as effective input in order to quickly detect the obstacle by GPU.


(a) Case of the vehicle.

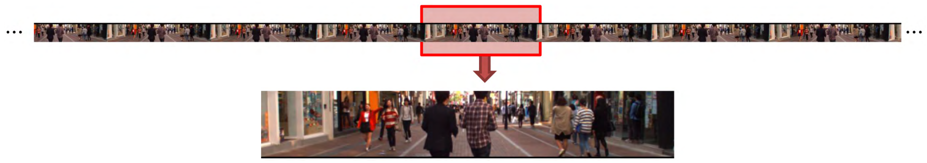

(b) Case of the pedestrian.

Figure 3.15: Generated equi-height mosaicking images.

3.4.3 Equi-Height Mosaicking Image for GPU Acceleration

In previous section, the proposed system made an equi-height mosaicking image. This image is used as effective input in the proposed system. In general, the function call in the program consumes the processing time. When the functions are performed, the program internally performs memory allocation and deallocation repetitively. If the program must repetitively call the function, this repetition becomes a serious bottleneck. This problem of repetitive function calls also exists in GPU processing. A GPU program has a widely-known data transfer problem as well as a bottleneck caused by a repetitive function call. Therefore, the proposed system uses the equi-height mosaicking image as the input to reduce the function calls to one. The proposed system calls the function for obstacle detection just once, because the equi-height mosaicking image is made by the concatenation of equi-height images. Fig. 3.16 and Fig. 3.17 show each function for obstacle detection by equi-height images and the equi-height mosaicking image. In this figure, the equi-height image performs the function call Ntimes. However, the equi-height mosaicking image performs the function call only one time.

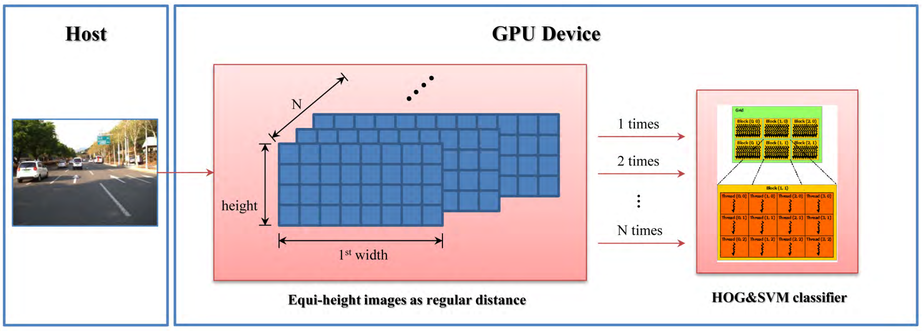


Figure 3.16: Detection process using equi-height images.

Table 3.1 shows the experimental comparison of using an equi-height image (EHI) with the equi-height mosaicking image (EHMI). The proposed system measures the processing time for obstacle detection using both types of images. As shown the Table, the equi-height mosaicking

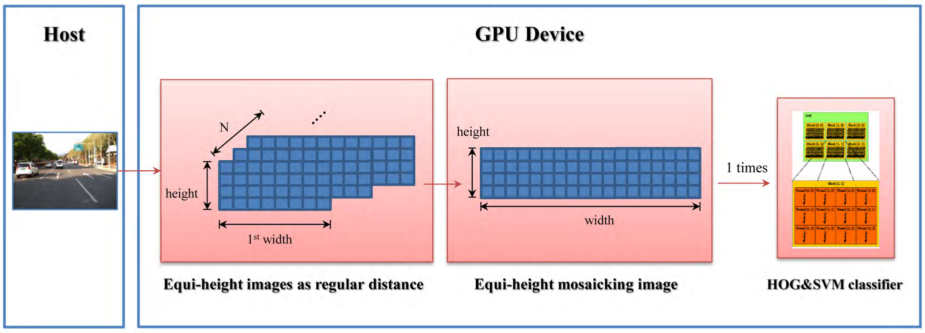

Host
GPU Device
N
height
1st width
Equi-height images as regular distance
height
width
Equi-height mosaicking image
1 times
HOG&SVM classifier

Figure 3.17: Detection process using equi-height mosaicking image.

Image(640x480)	EHI Generation	EHMI Generation	SVM Classification	Total Processing Time
EHIs	12ms	N/A	40ms (N times)	52ms
EHMI	12ms	2ms	11ms	25ms

Table 3.1: Result of processing speed between EHIs and EHMI.

image consumes extra processing time to concatenate the panorama-like image. However, the equi-height mosaicking image saves more processing time during the function for obstacle detection.

Chapter 4

Forward Dynamic Obstacle Detection

In Chapter 3, the proposed system suggested the equi-height mosaicking image as a new representation and generates the equi-height mosaicking image as effective input for fast GPU-based detection. In this chapter, we describe the detection approach for dynamic obstacles such as forward vehicles and pedestrians. The proposed system uses a HOG as feature data to detect vehicles and pedestrians at the same time. And the proposed system mainly uses the linear support vector machines (SVM) classifier for detection. In order to detect dynamic obstacles, however, the proposed system can also apply other classifiers such as AdaBoost or random forest.

4.1 Dynamic Obstacle of Road Scenes

Before the detection of dynamic obstacles is performed, the proposed system defines the detection target in the proposed system. In this chapter, we suggest the system to detect dynamic obstacles such as vehicles or pedestrians in front of the driver's vehicle on the road. Thus, the proposed system assumes the condition to define dynamic obstacles. The conditions are as follows:

- The obstacle exists on the road
- The obstacle exists in front of the driver
- The obstacle is able to move by itself
- The obstacle has a complete shape

Therefore, the proposed system finally defines an On-Road Dynamic Obstacle as follows:

- The complete rear shape of the forward vehicle on the road
- The walking or standing pedestrian of the complete shape in front of the driver

The proposed system performs the detection for these obstacles in real time. In this next section, the proposed system performs learning to generate the linear classifier for each vehicle and pedestrian. The generated classifiers are used to detect obstacles on equi-height mosaicking image. After the classifiers are generated, the proposed system performs a 1D search based on vehicles and pedestrians on the equi-height mosaicking image at the same time. Then it converts the coordinates between equi-height mosaicking image and the original image. Finally, the proposed system decides the detected obstacles using region grouping.

4.2 Learning for Equi-Height Mosaicking Image

In order to detect dynamic obstacles, the proposed system generates linear classifiers suitable to equi-height mosaicking image in offline processing. To resolve a problem caused by using an equi-height image, it generates the modified classifiers using modified positive data. After that, the proposed system extracts feature data from the training data using a HOG and generates the modified linear classifier suitable to equi-height mosaicking image for dynamic obstacle detection. The generated linear classifier, derived from modified positive data, ensures competent precision in spite of using the equi-height mosaicking image.

4.2.1 Vehicle Classifier Generation

In this subsection, we create the training data set to generate the modified classifiers for vehicle detection. It uses a training image size of 40x32 because the aspect ratio for the rear of a vehicle is about 1.25:1. We not

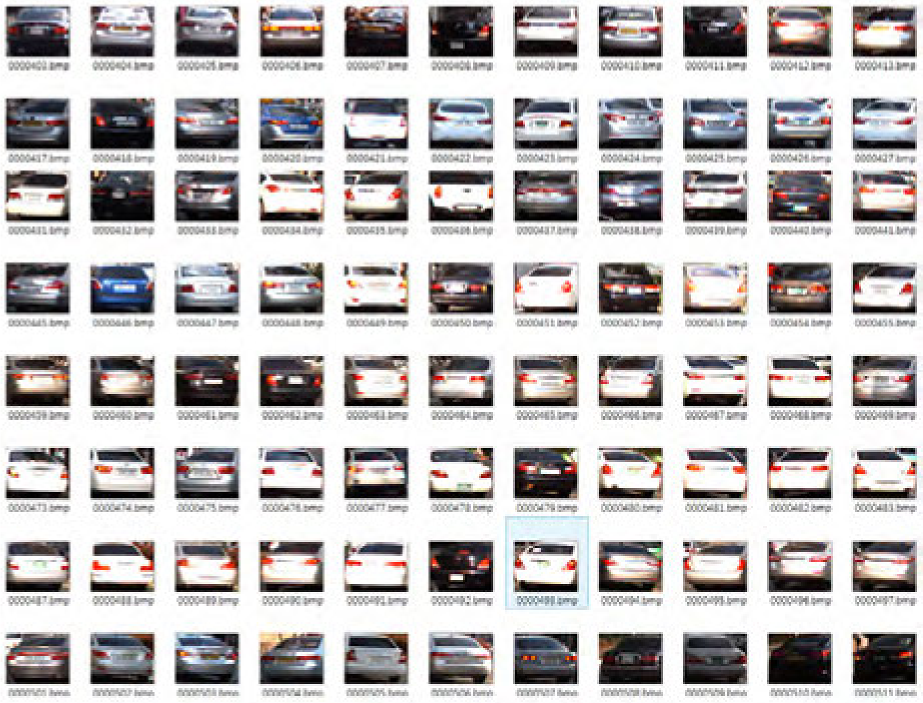


Figure 4.1: Postive data of normal-sized vehicle.

only use the normal-sized rear-shape image of the vehicle, but also use a reflective image of a normal-sized image to generate a better classifier. This step improves detection performance.

Furthermore, we use two types of positive data to resolve the problem caused by using the equi-height image. In the vehicle classification step, the proposed system uses the equi-height mosaicking image as a modified input to improve processing speed. It is a very useful method but it has a minor weak point. The weak point is that the height of the object in the equi-height mosaicking image is limited. Therefore, the size of a normal

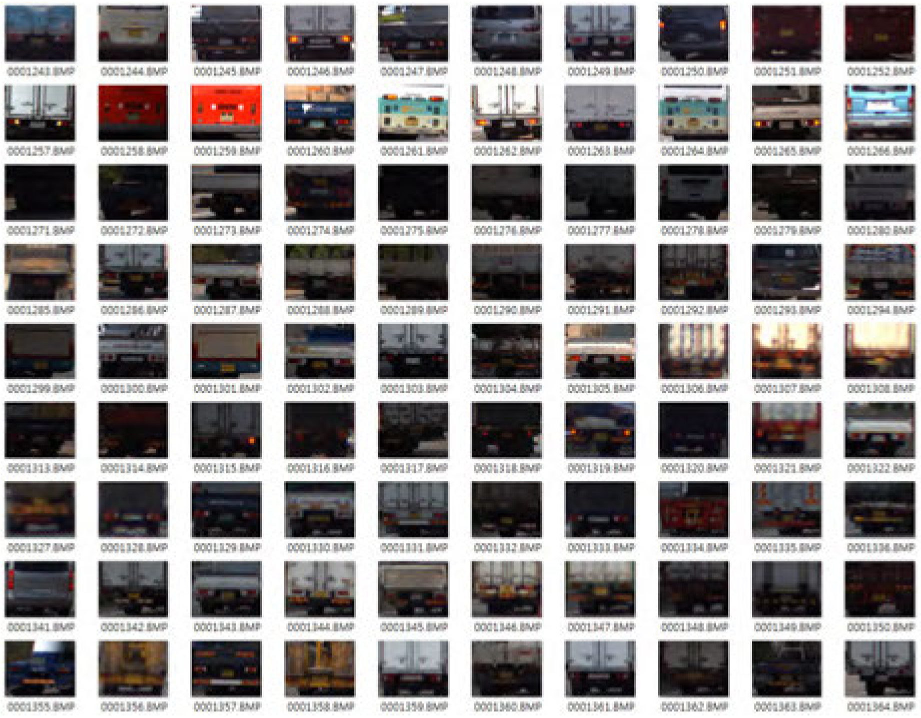
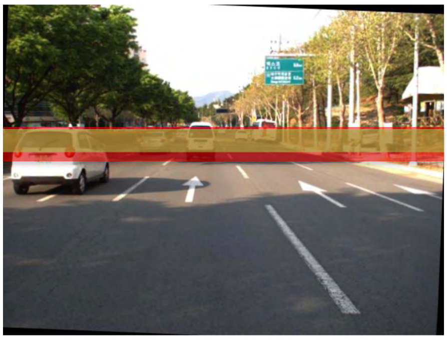
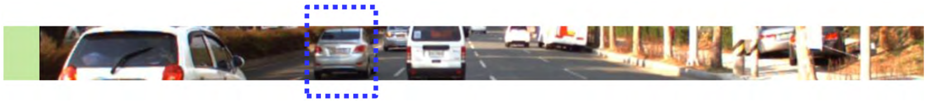


Figure 4.2: Top-cropped positivie data of larger vehicle.

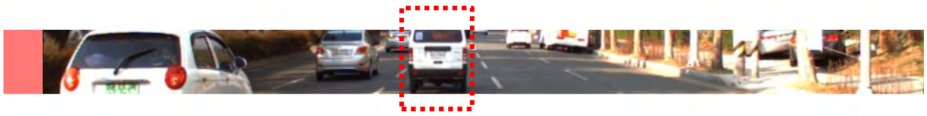
vehicle such as a sedan is able to be captured, but a larger vehicle such as truck or bus cannot be captured on the equi-height image. Therefore, we extract two types of positive data. The first piece of positive data is the rear images of normal-sized vehicles, and the second piece of positive data is the top-cropped images for larger vehicles. Since the equi-height images have the same height image for various-sized vehicles, we crop the upper part of the large size vehicle image and perform the training using the lower part of the image. Fig. 4.1 and Fig. 4.2 show both normal-sized



(a) Input image of the road scene.



(b) Normal-sized vehicle.



(c) Top-cropped vehicle.

Figure 4.3: Normal-sized and top-cropped vehicle on the equi-height image.

vehicles and top-cropped vehicles as positive data points to generate the vehicle classifier. And Fig. 4.3 illustrates normal-sized and top-cropped vehicles on the equi-height image.

After the training data modification is finished, the proposed system extracts the feature data. In this thesis, the proposed system uses HOG to detect the vehicle. In general, many another researchers use a Gabor filter-based feature or Haar-like features, rather than HOG. However, we use HOG because the proposed system performs detection for the vehicle and pedestrian using a single type of feature at the same time. The extracted HOG features are used to detect both the vehicle and pedestrian on equi-height mosaicking image.

Once the feature extraction for training data is finished, the proposed system performs the learning using supervised learning algorithm. The proposed system can use various learning algorithms such as SVM, AdaBoost, or random forest. In this thesis, however, the proposed system mainly uses the linear SVM classifier generated by svm-light tools [], because the SVM generally shows good performance in spite of insufficient data. Therefore, it generates a linear SVM classifier to detect vehicles.

4.2.2 Pedestrian Classifier Generation

Like the vehicle classifier generation, the proposed system generates a modified classifier for pedestrians on equi-height mosaicking image. In order to generate a linear classifier for pedestrian detection, we make a modified dataset suitable for the equi-height mosaicking image, shown in this section. To generate this modified dataset, we gather pedestrian data, and also use the reflective image for better classifier generation. Next, we generate tall pedestrian data by cropping and resizing the normal data. We also generate small pedestrian data by padding and resizing the normal-sized pedestrian's image. Because some pedestrian heights on the equi-height image can be taller or smaller than the average height of a pedestrian, we also generate both tall and small pedestrian datasets. Therefore, the proposed system generates a classifier suitable to the equi-height mosaicking image by using the modified positive data.

Fig. 4.4 shows an example of modified datasets created as an INRIA person dataset [29]. Fig. 4.5 shows the modified positive data as different heights of the pedestrian. And, Fig. 4.6 shows a pedestrian of various heights on the equi-height image.

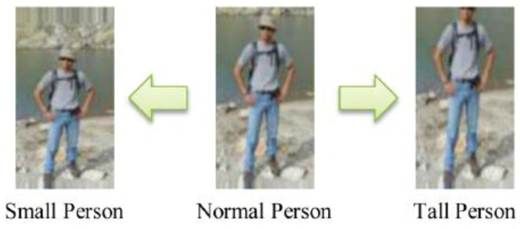
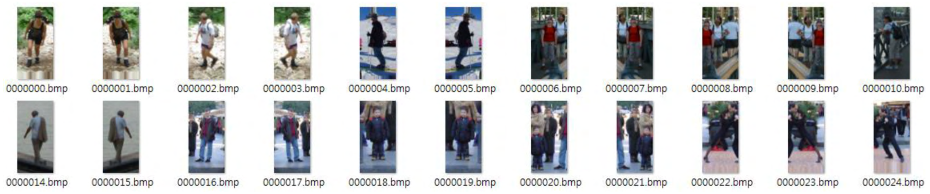


Figure 4.4: Modified training dataset as pedestrian's height.

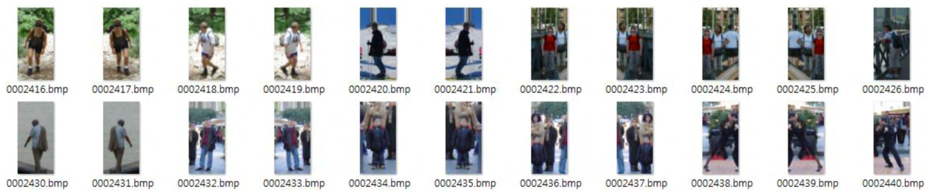
When the modified training data is generated, the proposed system extracts the features for training data using the HOG. After the feature extraction for training data is finished, the proposed system performs the learning using svm-light. Thus we generate the linear SVM classifier to detect the pedestrian.

4.3 Obstacle Detection on Equi-Height Mosaicking Image

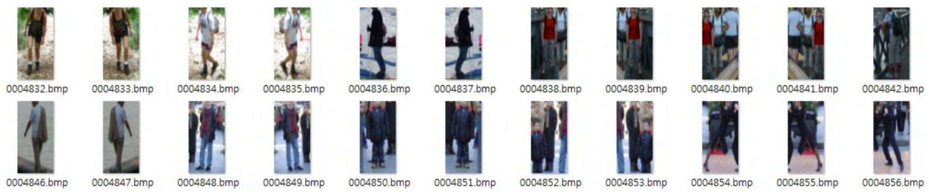
When the classifier for the vehicle and pedestrian detection is generated, the proposed system performs the detection for vehicles and pedestrians.



(a) Normal person.



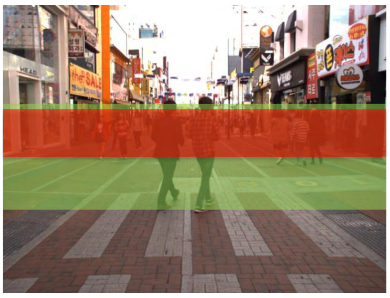
(b) Small person.



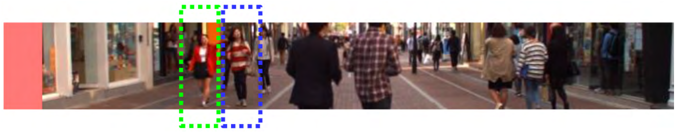
(c) Tall person.

Figure 4.5: Modified positive data for pedestrian classifier.

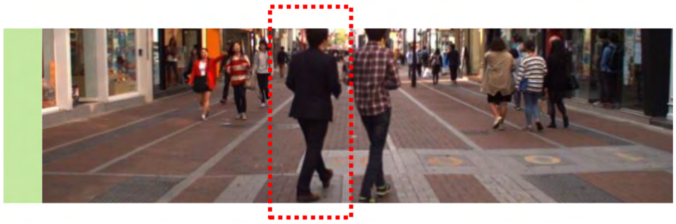
In order to detect forward vehicles and pedestrians on the equi-height mosaicking image, the proposed system performs 1D search-based obstacle detection using generated linear classifiers. In this section, the proposed system performs GPU-accelerated searching on equi-height mosaicking image. Next, it performs a coordinate transform from the detected coordinates of the equi-height mosaicking image to coordinates of the original image. Finally, the proposed system performs grouping for the detected rectangles to finally decide the region of the detected obstacle.



(a) Input image of the road scene.



(b) Small and normal person.



(c) Tall person than the average height.

Figure 4.6: Pedestrian of various heights on the equi-height image.

4.3.1 1D search-based Obstacle Detection

In order to search based on 1D space, the proposed system uses equi-height mosaicking image. The height of the equi-height mosaicking image is equal to the height of training data used for classifier generation. Therefore, the proposed system is able to perform GPU-accelerated 1D search-based obstacle detection on equi-height mosaicking image. After that, this detection

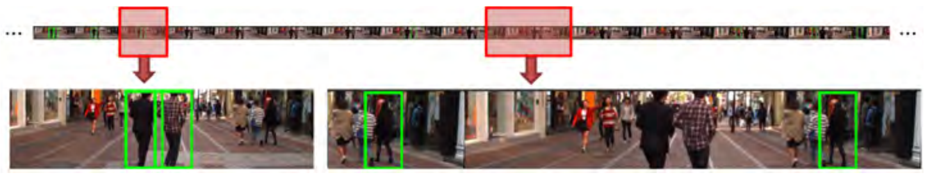


Figure 4.7: Detected results on an equi-height mosaicking image.

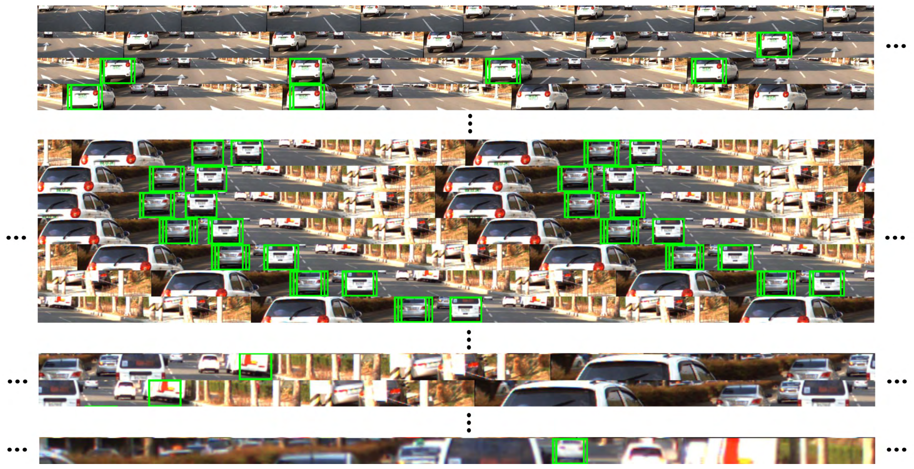


Figure 4.8: Result of 1D Search on the equi-height mosaicking image.

approach reduces the processing time by using only one function call.

Before the detection is performed, the proposed system first performs feature extraction for the equi-height mosaicking image. At this time, the proposed system uses the HOG [15] feature. After feature extraction is finished, it performs the 1D search on extracted the feature set of the equi-height mosaicking image using a linear SVM classifier.

In order to implement a fast real-time system, the proposed system applies a modified method based on the existing GPU-based HOG detec-

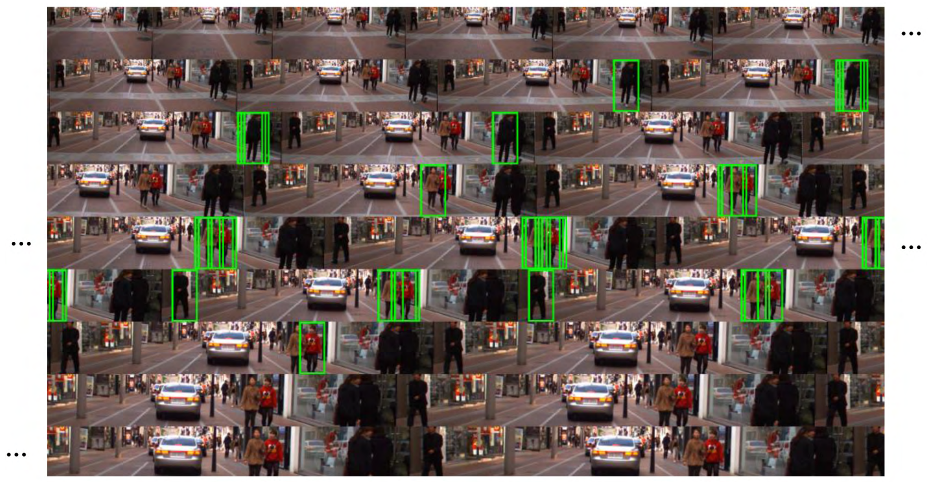


Figure 4.9: Result of 1D Search on the equi-height mosaicking image.

tor [6,30]. It is fast enough to be used in real time, but it is not enough for various scales. If the existing HOG detector would be modified to detect objects of various scales, it would become very slow due to the processing of the image scaling. The modified method is focused on reducing the computational complexity of the sliding window-based existing HOG detector, which requires a lot of time for image scaling and linear SVM classification [6].

In order to reduce computational complexity, we use the equi-height mosaicking image that is resized to a uniform scale. It changes the search process of the sliding window to a 1D search without scaling. Additionally, the proposed system also reduces the image transfer between the GPU processor and the host by reducing the computational region on the captured image. Fig. 4.7 shows the detected obstacle candidates on the equi-height mosaicking image.

For another road scene, Fig. 4.8 and Fig. 4.9 show several parts of the result of performing the 1D search on the equi-height mosaicking image. In these figures, these equi-height mosaicking images are visualized by stacked images on 2D space. The green rectangles indicate detected obstacle candidates as various sampling positions.

4.3.2 Coordinate Transform and Grouping

After several regions of obstacle candidates are detected in the equi-height mosaicking image, the proposed system maps them into the original image. This step is called coordinate transform. Actually, it transforms the coordinates from equi-height mosaicking image space to original image space.

In order to transform the coordinates, the proposed system needs the scale between the sub-regions of the equi-height mosaicking image including the detected obstacle and the original equi-height image. It also needs x-coordinates in the sub-region of equi-height mosaicking image. Finally, the proposed system needs a number of sub-regions, which is the number of the concatenated equi-height images. The coordinates are represented by the following equation:

$$Coord_{org} = X_i^{EHI} + (X_{org}, Y_{org}, W_{org}, H_{org})$$

$$\begin{aligned} X_{org} &= (X_{EHMI} - \sum_{n=1}^{i-1} W_n^{EHI})/\delta_i^{EHI} \\ Y_{org} &= (SP_i^{EHI}) - H_i^{EHI} \\ W_{org} &= W_{Train}/\delta_i^{EHI} \\ H_{org} &= H_{Train}/\delta_i^{EHI} \end{aligned} \tag{4.1}$$

where $Coord_{org}$ means the coordinates in the original image. X_{org} and Y_{org} are x and y coordinates in the original image, respectively. W_{org} and H_{org} are the width and height in the original image, respectively. And X_{EHMI} means the x-coordinate of the detected obstacle on the equi-height mosaicking image. i is the number of the concatenated equi-height images for equi-height mosaicking image generation. And δ^{EHI}, SP^{EHI}, X_i^{EHI}, W^{EHI}, and H^{EHI} mean scale, sampling position, x-coordinate of i-th equi-height image, width, and height on each equi-height image, respectively. W_{Train} and H_{Train} is the width and height of the training data, respectively.

Finally, the proposed system performs a non-maximal suppression [24] for grouping multiple overlapping obstacles. Fig. 4.10 shows the results of mapping to the original image with non-maximal suppression. The green

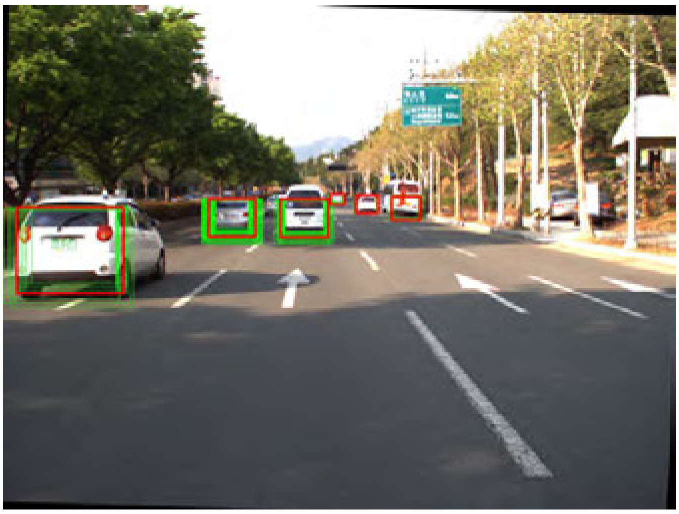
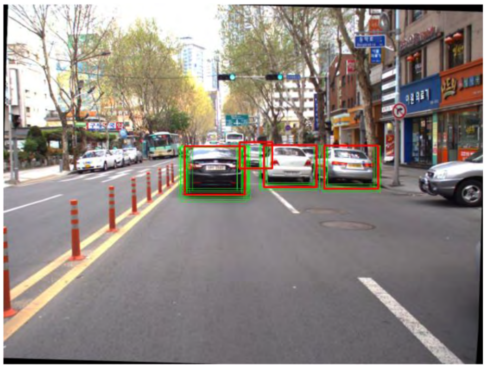
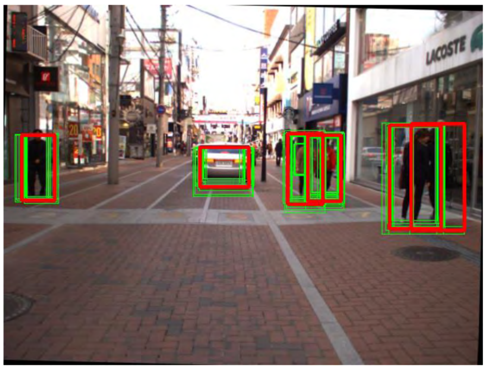


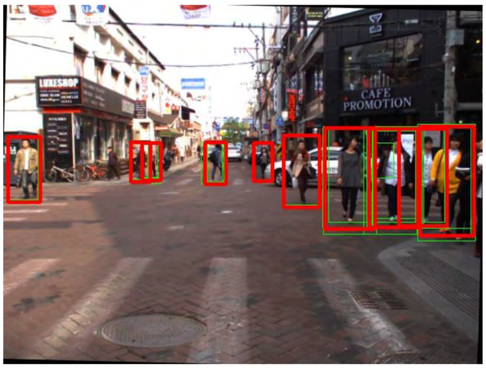
Figure 4.10: Results of mapping to original image and the non-maximal suppression.

rectangles are obstacle candidates that are converted to the original image. The red rectangles indicate definitive obstacles using non-maximal suppression. And Fig. 4.10 also shows the detected results for another road scene.


(a) 5446 frame.



(b) 7839 frame.



(c) 9105 frame.

Figure 4.11: Detected results for another road scene.

Chapter 5

Part-based Side Detection of Approaching Vehicle

In Chapter 4, we suggested an equi-height mosaicking image-based detection approach for complete-shape obstacles. However, many dangerous situations are made by approaching vehicles from the left or right sides. In this case, our obstacle detection system cannot detect the approaching vehicles, because the approaching vehicles do not present a complete rear shape. Therefore, we suggest a new approach for part-based side detection of approaching vehicles.

In this chapter, we first define the approaching vehicle, including the incomplete side profile, before we describe the detection approach for the partial side of the approaching vehicle. Once the approaching vehicle is de-

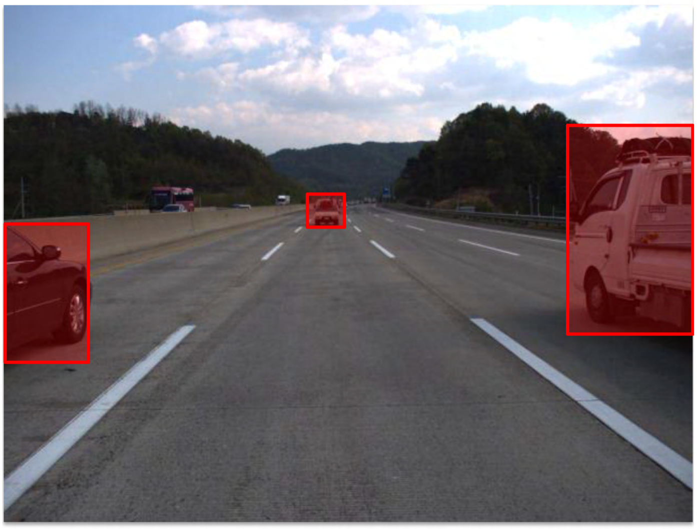


Figure 5.1: Dynamic obstacles in the road scene.

fined, the proposed system generates an equi-height peripheral mosaicking image from the equi-height mosaicking image. Then it performs learning for classifier generation suitable to the equi-height peripheral mosaicking image. Next, the proposed system performs a GPU-accelerated 1D search for part-based side detection using the equi-height peripheral mosaicking image. Finally, it transforms the coordinates in two stages and performs the grouping as normal.

5.1 Approaching Vehicle from Each Side

In this section, we define approaching vehicles from each side of the car. Fig. 5.1 illustrates vehicles around the car in a road scene. In this figure, we can see three vehicles. What is more dangerous vehicle? In general, the vehicle approaching on the left or right is more dangerous, because these vehicles exist in the driver's blind spot. However, many studies as well as our obstacle detection described in Chapter 3 are focused on vehicle detection via complete rear shapes. Therefore, the proposed system additionally detects approaching vehicle without complete rear shapes.

Above all, we specifically define the approaching vehicle as a dangerous obstacle before we perform part-based side detection. The dangerous approaching vehicle is defined as follows:

- The vehicle suddenly appears from either side region of the road
- The vehicle shows a partial side in driver's blind spot

When the approaching vehicle suddenly appears, the driver feels the danger while driving. Therefore, the proposed system additionally performs detection for the approaching vehicle to warn it about dangerous obstacles. In order to detect the approaching vehicle, the proposed system uses part-based side detection.

The part-based side detection approach detects the front and rear parts of the approaching vehicle. Thus, the proposed system can easily detect the approaching vehicle including ones showing incomplete shapes in the driver's blind spot.

5.2 Equi-Height Peripheral Mosaicking Image Generation

In Chapter 4, the proposed system detected vehicles and pedestrians that have complete shapes using the equi-height mosaicking image. However, this method cannot detect the obstacle of an incomplete shape. In particular, vehicles often approach a car quickly. In this case, the proposed system needs another detection method to warn of an approaching vehicle with an incomplete shape. Therefore, we suggest a part-based side detection method using equi-height peripheral mosaicking image.

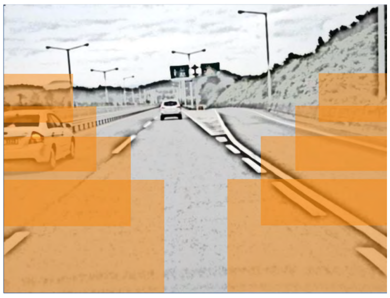
In order to detect the partial side of the approaching vehicle, we suggest another new image representation, called equi-height peripheral mosaicking image. The equi-height peripheral mosaicking image is the concatenated image for each side region of the road scene. This image is used to detect the partial sides of any approaching vehicles.

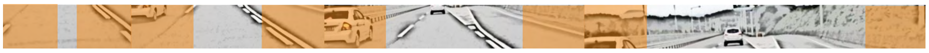
To generate the equi-height peripheral mosaicking image, the proposed system performs three steps: candidate region extraction, peripheral-region warping, and concatenation of the warped images. In candidate region extraction, we extract each side-region of the driver on the road scene, because the approaching vehicle is almost passed in each side-region. In peripheral-region warping, the proposed system performs image warping for extracted regions in order to visualize the detected partial side of the approaching vehicle, if the approaching vehicle exists in the side-region. In the concatenation of the warped images, the proposed system performs the concatenation for warped images to generate a panorama-like image. The generated image is used as effective input for GPU acceleration, similar to the equi-height mosaicking image.

5.2.1 Candidate Region Extraction

First of all, in this subsection, the proposed system extracts the candidate region in which the approaching vehicle exists. In general, the approaching vehicle suddenly appears on each side on the road scene. Thus we only extract the side-region of the road scene for effective detection. Fig. 5.1 (a) shows the search region on the input image. Like this figure, the proposed system only performs detection in the orange rectangles.

The size of each search region is set as a distance from the driver's vehicle. Thus the proposed system also uses the equi-height images, because the image can reduce false positives. In Chapter 4, however, the proposed system generates the equi-height mosaicking image that is made by equi-height images. Therefore, it extracts the search region using the equi-height mosaicking image. Because the subset of the equi-height mosaicking image is each equi-height image, we crop each side of the equi-height image, depending on sampled position of the equi-height image. Fig. 5.2 (b) shows the search region of the equi-height mosaicking image. Fig. 5.2 (c) shows the extracted candidate regions depending on the sampled position of equi-height image. After the extracted search regions in Fig. 5.2 (b) are


(a) Search regions on the image.


(b) Search regions of the equi-height mosaicking image.

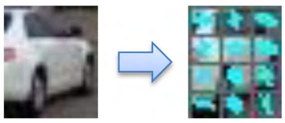
(c) The real candidate regions for the part-based side detection of approaching vehicle.

Figure 5.2: Process of the candidate region extraction.

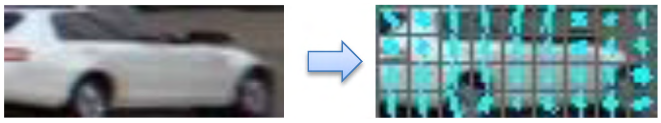
cropped, the cropped search region is called the candidate region.

5.2.2 Peripheral Region Warping

When the candidate regions are extracted, the proposed system performs image warping for the candidate regions. The image warping for the candidate region is called peripheral region warping. Peripheral region warping is performed to visualize the partial side of the approaching vehicle if the approaching vehicle exists in the side region. At this time, the warped image to visualize the partial side is called the peripheral warped image. In the next subsection, the proposed system will concatenate the peripheral warped images.



(a) The side of original vehicle.



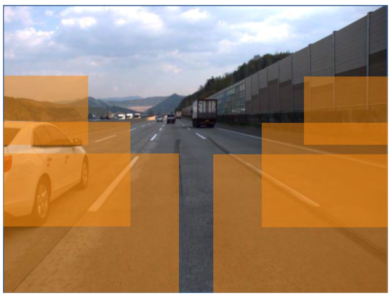
(b) The side of the warped vehicle.

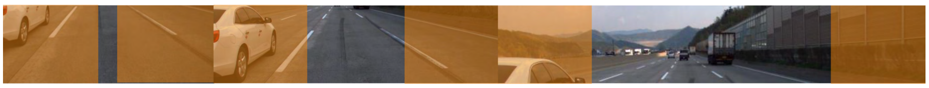
Figure 5.3: Difference of HOG feature between warped and original image.

Despite consuming processing time, peripheral region warping is performed. Because the partial side image of the approaching vehicle is smaller than a complete rear shape, the proposed system cannot extract sufficient HOG features. Therefore, it performs peripheral region warping. Once peripheral region warping is performed for candidate regions, the proposed system is able to get a wide side image of the approaching vehicle. Thus, it is able to extract sufficient HOG features. Fig. 5.3 shows the difference between HOG feature extraction between a warped side image and the original side image. When peripheral region warping is performed, the proposed system has to sequentially perform the image warping for sub-regions of the candidate region using a rectangle of fixed size.

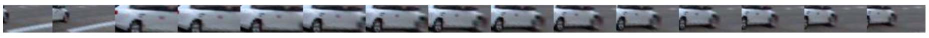
5.2.3 Mosaicking of Warped images

After peripheral region warping is completed, the proposed system concatenates the warped images like the equi-height mosaicking image. The


(a) Input image.


(b) Equi-height mosaicking image.

(c) Real candidate regions for the part-based side detection of approaching vehicle.


(d) Equi-height peripheral mosaicking image for the left-side of the driver.

(e) Equi-height peripheral mosaicking image for the left-side of the driver.

Figure 5.4: Process of equi-height peripheral mosaicking image generation.

concatenated image becomes the equi-height peripheral mosaicking image. This image also has an advantage similar to the equi-height mosaicking image. In other words, the equi-height peripheral mosaicking image is used as effective input for GPU acceleration when the part-based side detection of the approaching vehicle is performed.

Fig. 5.4 shows the total process done on the equi-height peripheral mosaicking image. As seen in the figure, the proposed system has to generate two types of equi-height peripheral mosaicking image for each side-region. Fig. 5.4 (d) is the equi-height peripheral mosaicking image for the left side and Fig. 5.4 (e) is the equi-height peripheral mosaicking image for the right side.

5.3 Learning for Equi-Heigth Peripheral Mosaicking Image

In this section, the proposed system performs learning to create classifiers for part-based side detection of the approaching vehicle after the equi-height peripheral mosaicking image is generated. First of all, the proposed system extracts the normal side region of the approaching vehicle. Then it performs image warping for the side of the vehicle. In this case, we cannot use the whole side-shape of the vehicle for approaching vehicle detection because the whole side-shape of the vehicle does not always appear. Thus we use the front and rear side sections of the vehicle as training data. Then, the proposed system extracts the HOG features and performs learning to create classifiers for front and rear part of the side.

5.3.1 Training Data Extraction

In this subsection, we create the training data from gathered vehicle data. In general, the approaching vehicle does not present its complete shape. Most approaching vehicles show only a partial side. Thus, the proposed

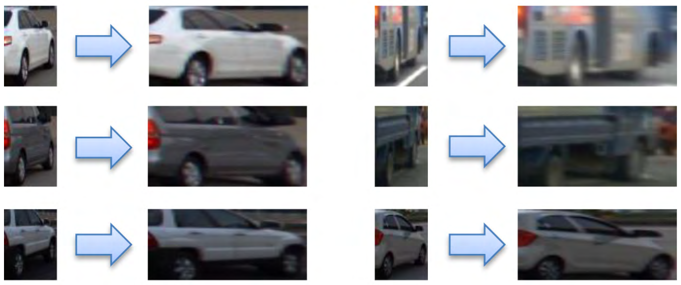


Figure 5.5: Warped side-shape for the learning.

system focuses on side-shape detection. However, the proposed system cannot extract sufficient HOG features because the side-shape region is too narrow. Therefore, it performs image warping for the side region of the vehicle. Fig. 5.5 shows the warped side shapes used for the learning.

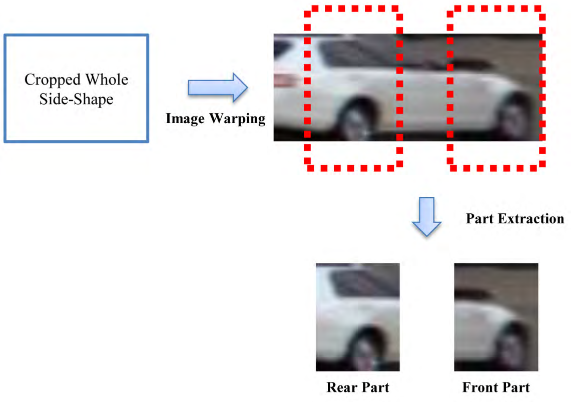


Figure 5.6: Process of training data extraction.

In the proposed system, however, the warped complete side shape of the vehicle cannot be used for vehicle detection because the whole side-

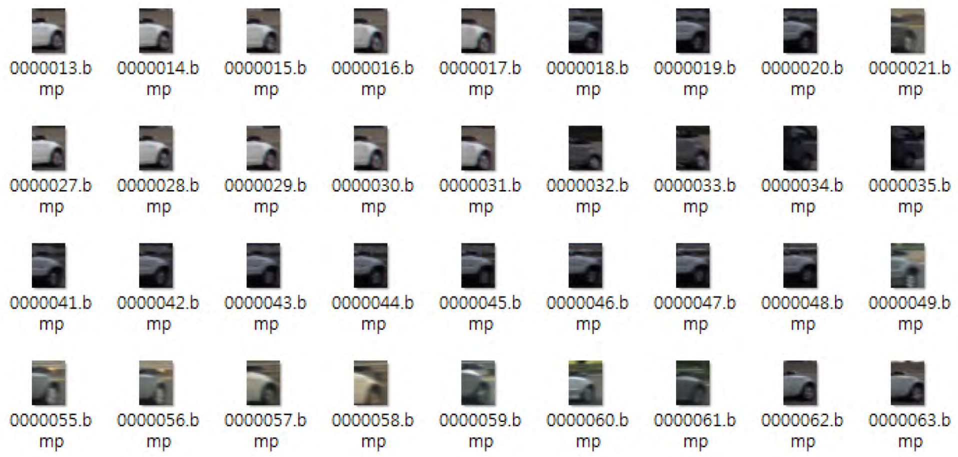


Figure 5.7: Positive data for front side-shape of the approaching vehicle.

shape of vehicle does not always appear. Thus we mainly use the front part of the side-shape for training data. The front part of the side-shape almost always appears if the approaching vehicle exists in the side road scene. Also, the proposed system uses the rear side-shape to prepare for missing the front side-shape if the approaching vehicle moves away from the driver. Therefore, we extract the front and rear side shapes as training data to create a linear classifier. Fig. 5.6 shows the process of training data extraction. The red rectangles of the dotted line show the front side-shape and rear side-shape, extracted from the warped image.

Fig. 5.7 and Fig. 5.8 show the extracted positive data for the front and rear side-shapes of warped approaching vehicles. The negative data is made in a similar manner with positive data. First, we extract a negative region from background images and other images. Then we perform image

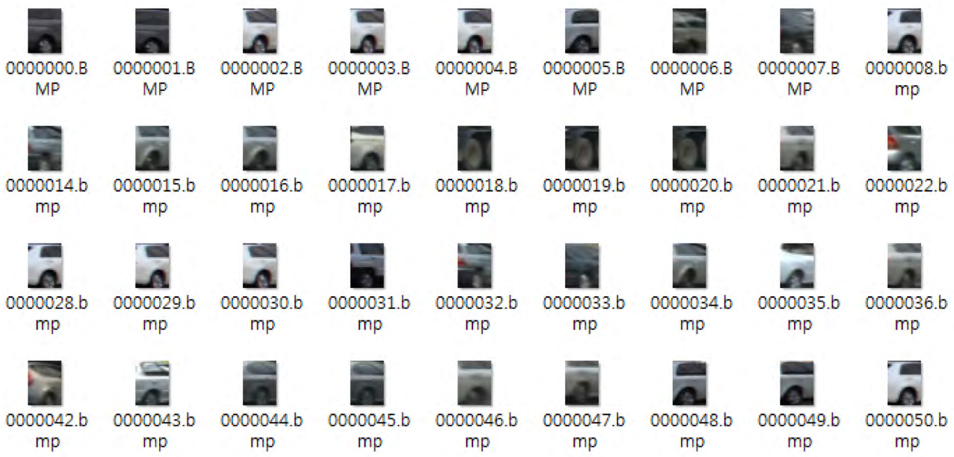


Figure 5.8: Positive data for rear side-shape of the approaching vehicle.

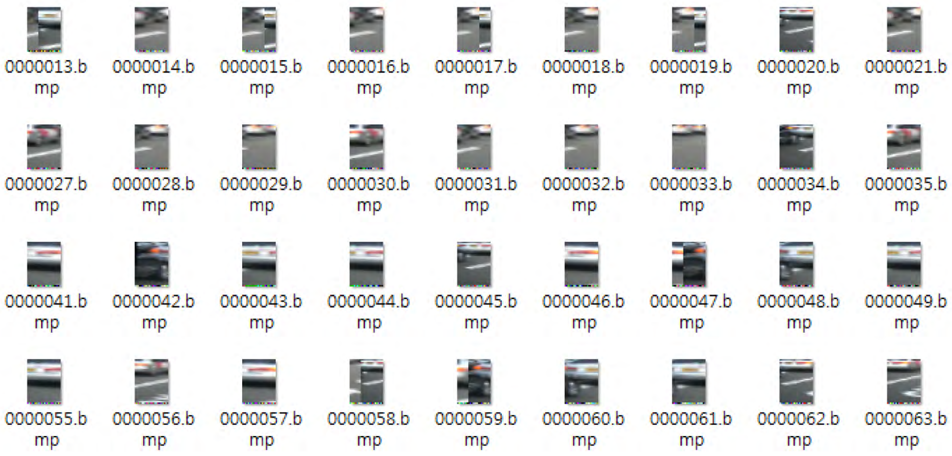


Figure 5.9: Negative data for the learning.

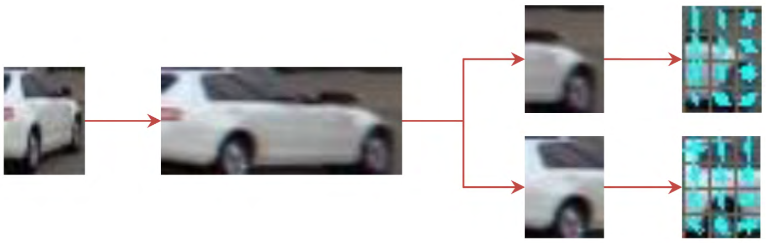


Figure 5.10: Process of the training data extraction and feature extraction for learning.

warping and extract the sub-region of equal sizes with the front or rear side-shape. Fig. 5.9 shows the negative data used for learning.

5.3.2 Classifier Generation

After training data extraction is completed, the proposed system generates the classifier for front and rear side-shape detection. First of all, the proposed system extracts the HOG features for extracted side-shape partial images as training data. Fig. 5.10 illustrates the process of the training data extraction and HOG feature extraction.

When feature extraction is finished, the proposed system performs learning to generate a classifier. In the proposed system, we can use various classifiers such as SVM, AdaBoost, or random forest. But we mainly use AdaBoost as the classifier for part-based side detection. The process of the classifier generation is performed on offline processing. In the next section, the proposed system performs GPU-accelerated part-based side detection using the generated classifier on online processing.

5.4 1D Search based Detection

Next, the proposed system performs 1D search-based detection on equi-height peripheral mosaicking image. In this case, the height of the equi-height peripheral mosaicking image is equal to the height of the training data used for classifier generation. Thus it is able to perform GPU-accelerated 1D search-based detection.

Before the 1D search-based detection is performed, the proposed system first extracts the HOG features for the equi-height peripheral mosaicking image. Then, the proposed system classifies on the extracted feature set for the equi-height peripheral mosaicking image using AdaBoost classifier. This search approach uses the equi-height peripheral mosaicking image that is resized to a uniform scale to reduce the computational complexity. It changes the search process of the traditional sliding window to a 1D-based search without scaling. And then the equi-height peripheral mosaicking image is used as an effective input for GPU-based fast-side detection.

5.5 Coordinate Transform and Grouping

Once the side detection on equi-height peripheral mosaicking image is completed, the proposed system performs coordinate transform and grouping.

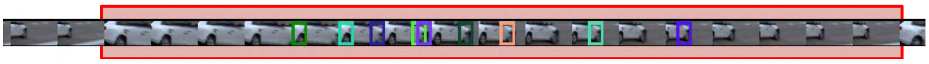
If the several side parts are detected, the proposed system first performs coordinate transforms between equi-height peripheral mosaicking image and the original image.

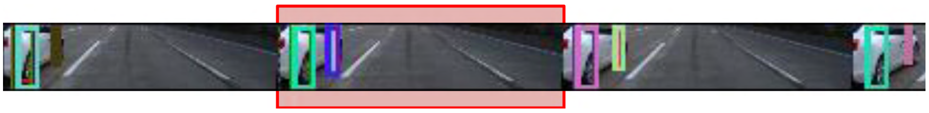
The coordinate transform is performed in two stages. The first stage is a transform between the equi-height peripheral mosaicking image and equi-height mosaicking image. In this stage, the proposed system performs inverse warping and coordinate conversion from equi-height peripheral mosaicking image to equi-height mosaicking image. The second stage is the transform between equi-height mosaicking image and the original image, like in Chapter 4.

For the second stage, the proposed system needs a scale between the sub-region of the equi-height mosaicking image that includes the detected obstacle and the original equi-height image. And it also needs x-coordinates in the sub-region of equi-height mosaicking image. Then we need the number of sub-regions of equi-height mosaicking image. The basic coordinates are represented as the following equation:

$$\begin{aligned} Coord_{EHPMI} =& \quad (X_{EHPMI}, Y_{EHPMI}, W_{EHPMI}, H_{EHPMI}), \\ Coord_{EHMI} =& \quad (X_{EHMI}, Y_{EHMI}, W_{EHMI}, H_{EHMI}), \\ Coord_{org} =& \quad (X_{org}, Y_{org}, W_{org}, H_{org}), \end{aligned} \tag{5.1}$$

where $Coord_{EHPMI}$, $Coord_{EHMI}$, and $Coord_{org}$ mean the detected coor-


(a) The detected result on the equi-height peripheral mosaicking image.


(b) The result of the 1st stage of coordinate transform to the equi-height mosaicking image space.

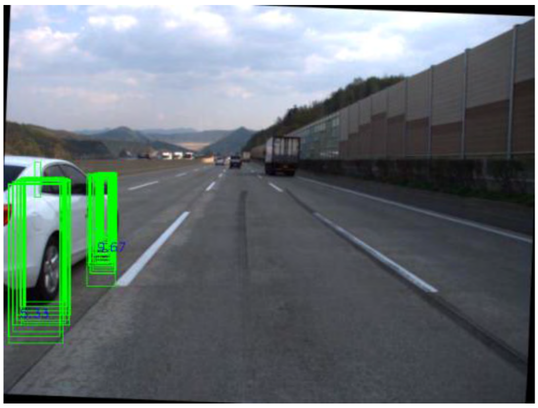

(c) The result of the 2nd stage of coordinate transform to original image space.

Figure 5.11: Process of the coordinate transform in two stages.

dinate in equi-height peripheral mosaicking image, the transformed coordinate to equi-height mosaicking image, and the final coordinate of the original image. X, Y, W, and H mean the x-coordinate, y-coordinate, width, and height, respectively. First of all, the coordinate transform between the equi-height peripheral mosaicking image and equi-height mosaicking image is represented as the following equation:

$$Coord_{INV\_EHPMI} = \quad IH_{warp} * Coord_{EHPMI}, \tag{5.2}$$

where IH_{warp} means the inverse warped matrix for peripheral region warping. Therefore, $Coord_{INV\_EHPMI}$ means the inverse warped coordinate from $Coord_{EHPMI}$. In Eq. (5.2), the proposed system first performs inverse warping for the detected coordinate in equi-height peripheral mosaicking image.

$$\begin{aligned}
X_{EHMI} = &\quad X_j^{PWI} + (X_{INV\_EHPMI} - \sum_{n=1}^{j-1} W_n^{PWI}), \\
Y_{EHMI} = &\quad Y_{INV\_EHPMI}, \\
W_{EHMI} = &\quad W_{INV\_EHPMI}, \\
H_{EHMI} = &\quad H_{INV\_EHPMI},
\end{aligned} \tag{5.3}$$

where j is the index included x-coordinate of the detected vehicle in the equi-height peripheral mosaicking image. X_j^{PWI} means x-coordinate of j-th peripheral warped image in equi-height peripheral mosaicking image. And W^{PWI} means the width of the peripheral warped image. After the first coordinate transform is calculated, the proposed system performs the other coordinate transform between the equi-height mosaicking image and the original image.

$$\begin{aligned} X_{org} = &\quad X_i^{EHI} + (X_{EHMI} - \sum_{n=1}^{i-1} W_n^{EHI})/\delta_i^{EHI}, \\ Y_{org} = &\quad (SP_i^{EHI}) - Y_{EHMI}, \\ W_{org} = &\quad W_{EHMI}/\delta_i^{EHI}, \\ H_{org} = &\quad H_{EHMI}/\delta_i^{EHI}, \end{aligned} \tag{5.4}$$

where $Coord_{org}$ means the final transformed coordinate in the original image. X_{org} and Y_{org} is x and y coordinates in the original image, respectively. W_{org} and H_{org} is the width and height of the original image, respectively. And X_{EHMI} means the x-coordinate of the detected obstacle on the equi-height mosaicking image. i is the number of the concatenated equi-height images for equi-height mosaicking image generation. And δ^{EHI}, SP^{EHI}, X_i^{EHI}, W^{EHI}, and H^{EHI} mean the scale, sampling position, x-coordinate of i-th equi-height image, width, and height on each equi-height image, respectively. W_{Train} and H_{Train} is the width and height of the training data, respectively.

Fig. 5.11 illustrates the process of the coordinate transform in two stages. Fig. 5.11 (a) shows the detected results on the equi-height peripheral mosaicking image. The colored rectangles mean the detected side-part of the approaching vehicle. Fig. 5.11 (b) sshows the coordinate transformed results on the equi-height mosaicking image. The color rectangles mean

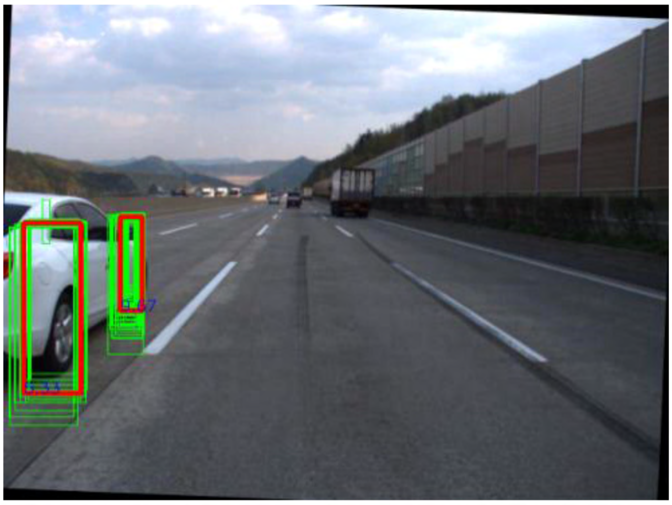


Figure 5.12: Result of the part-based side detection for approaching vehicle.

the transformed regions of the partial side using Eq. (5.2) and Eq. (5.3). Fig. 5.11 (c) shows the final transformed coordinates of the original image. The green rectangles mean the final transformed results using Eq. (5.4).

After the coordinate transform is finished, the proposed system finally performs a non-maximal suppression [24] for grouping multiple overlapping obstacles like Chapter 4. Fig. 4.10 shows the final results of the part-based side detection for the approaching vehicle.

Chapter 6

Experiments

In this thesis, we describe the experimental setup and results for three types of data. In order to perform the experiments, we set up a camera and processing system. In these experiments, we use a high-end camera system but can use another low-end camera system. Actually, we perform the experiments using the ETH dataset [31] captured by another camera system in pedestrian detection. In the experimental results, the detection result is measured with an $F1$ measure that includes recall and precision. And we visualize the detection results for various cases and settings.

6.1 Experiment Setup

The proposed system presents real-time dynamic obstacle detection for an AR-based FW-HUD system. In order to measure the processing time, we used Visual C++ on an i5 750(2.66GHz) CPU with 16GB of RAM and an nVidia Geforce GTX 680 graphics card. The proposed system also used a Grasshopper2 camera by Point Grey Research connected by IEEE 1394b. Table 6.1 shows the specifications of the camera and lens used in the proposed system.

Table 6.1: Specification of our camera system.

Type	Name	Spec.	Interface	Company
Camera	GS2-FW-14S5	1384x1036 at 30fps	1394b	Point Grey Research
Lens	LM8JMC	f=8mm	C-Mount	KOWA

6.2 Experimental Results

In this section, we perform the experiments with the three types of obstacle data. The first and second types of data points are the vehicles and pedestrians existing on the road scene. Thus we perform detection for the complete shape of the vehicles and pedestrians. The third type of data point is partial vehicles that exist in the road scene. In this case, we detect the vehicle included in the incomplete shape. Then we effectively visualize the various analyses of the experimental results.

When complete shapes are available in vehicle and pedestrian detection, the proposed system is executed faster than the existing HOG-based SVM detector under the same experimental environment [26]. In the experiments, we compared the performance of the proposed system to the existing HOG-based SVM detector that uses a GPU-based OpenCV HOG detector [30].

In approaching vehicle detection, we perform experiments for the detection rate. And we analyze what is appropriate learning to detect the approaching vehicle detection. In this analysis, we select the appropriate parts of the vehicle to generate a high-performance classifier. In this experimental result, the accuracy was measured with an $F1$ measure [32,33] and computed with the following equation:

$$\begin{aligned} \text{F1 Measure} &= 2 \times (R \times P)/(R + P), \\ where, R &= TP/(TP + FN), \\ P &= TP/(TP + FP). \end{aligned} \tag{6.1}$$

where R and P denote recall and precision, respectively. TP, FP and FN represent true positive, false positive, and false negative, respectively.

6.2.1 Hypothesis Sampling Level for Balanced Detection

Before the experiments are performed, we set the hypothesis sampling level by the experiment. In general, the performance of the proposed system is up to the hypothesis sampling level for equi-height mosaicking image generation. When the hypothesis sampling is denser, the accuracy of the proposed system is reduced. However, the execution time is also increased, because the width of the equi-height mosaicking image is longer because of

Table 6.2: Detection rate and execution time of each sampling level.

Samp. Lev.	EHMI Size	Detection Rate (%)			Execution Time (ms)		
		Rec.	Prec.	F1	EHMI	HOG	Total
Level 8	2687×32	0.22	100.00	0.44	5	1	10
Level 18	5882×32	5.41	93.59	10.23	7	1	12
Level 28	9426×32	51.45	93.15	66.28	8	2	14
Level 38	13583×32	66.42	88.02	75.71	12	3	18
Level 48	16790×32	75.24	87.88	81.07	13	4	21
Level 58	20784×32	79.76	88.12	83.74	14	4	23
Level 68	24049×32	84.14	87.78	85.92	15	4	25
Level 78	27363×32	84.58	87.77	86.15	18	5	29
Level 88	29536×32	85.47	87.68	86.56	21	5	32
Level 98	33031×32	86.66	88.03	87.34	24	5	36
Level 108	35447×32	87.47	88.92	88.19	25	6	39

the increased equi-height images. When the hypothesis sampling is sparser, the accuracy is increased and execution time is reduced. Therefore, we set the hypothesis sampling level to have the desired performance for balanced detection.

The experiments are performed with a hypothesis sampling level of eleven stages. Table 6.2 illustrates the detection rate and execution time as changing the sampling level. Fig 6.1 and Fig 6.2 show a graph for the detection rate and execution time. The light gray and light red denote the balanced sampling level. Therefore, we set that balanced sampling level to 68 by experimental result. However, the sampling level changes as desired performance and input data.

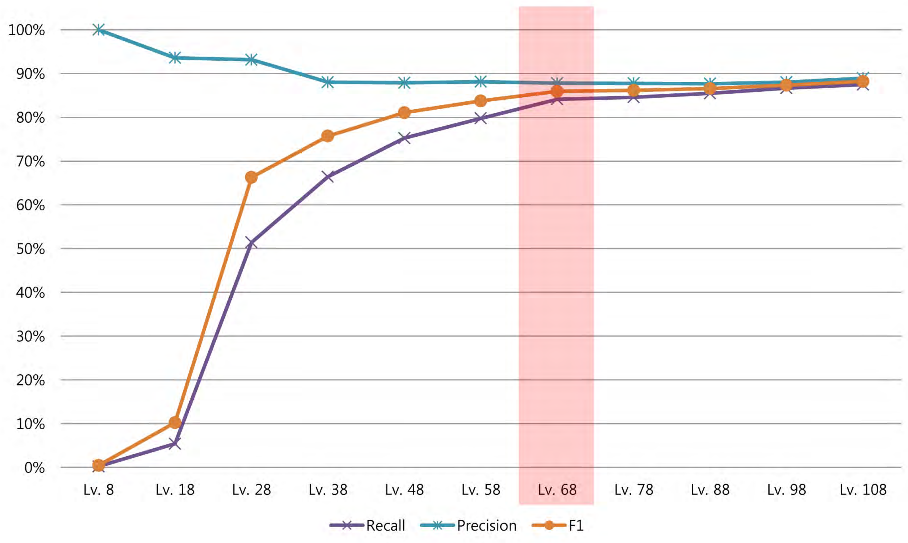


Figure 6.1: Detection rate of each hypothesis sampling level.

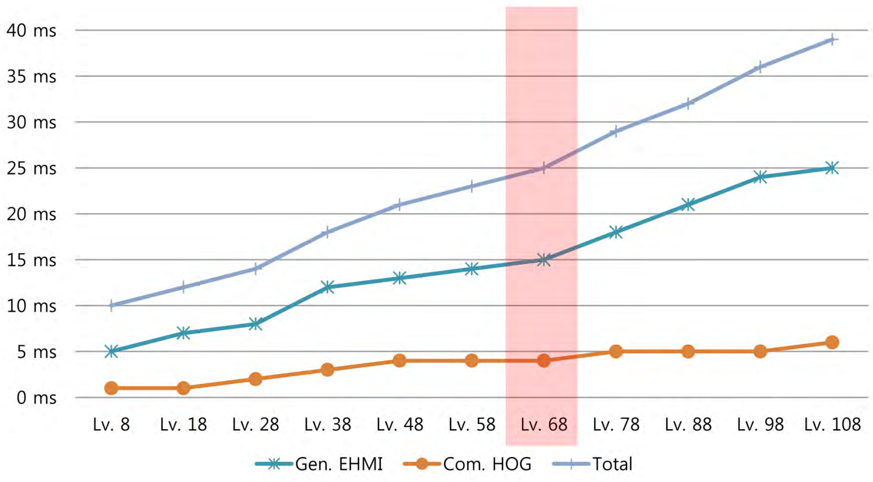


Figure 6.2: Execution time of each hypothesis sampling level.

6.2.2 Vehicle Detection of Road Scenes

In this subsection, we perform the experiments for vehicle detection. As shown in Table 6.3, our approach enhances the execution time in whole steps. In particular, image resizing and block histogram computation are improved by restricting the detecting area. Precision has improved but only marginally so. Because both systems used a linear SVM classifier with the same performance level, their accuracy is also equal. Therefore, the precision of the proposed system seems to have improved because the equi-height mosaicking image eliminates unnecessary image areas using geometric information of the road scene. However, the proposed system performs faster than the GPU-based OpenCV HOG+SVM detector [30] under the equal sampling level. Therefore, the proposed system can be added to other hypothesized methods of improving detection accuracy.

Table 6.3: Experimental result of vehicle detection.

Type	Appr.	Size	Scale	Execution Time (ms)				Detection rate(%)		
				Resize	HOG	Tran.	Total	Prec.	Recall	F1
Highway	ocv	640 x 480	70	37	371	1	472	82.14	80.20	81.16
	ours			15	5	1	27	88.34	85.21	86.75
Downtown	ocv		100	53	416	1	497	78.92	78.23	78.57
	ours			19	6	2	38	86.25	83.67	84.95

The experiments were executed on downtown roads and a highway during the day. We used 640x480 resolution images to clearly show the difference in the execution time. For training the SVM classifier, we used a training data set that contains positive images of 3,070 vehicles and negative images of 42,253 non-vehicles.

6.2.3 Pedestrian Detection of Road Scenes

Next, we perform the experiments for pedestrian detection on road scene data. Table 6.4 shows the experimental environment and results. As shown in Table 6.4, the proposed system considerably improved execution time. Image resizing time and block histogram computation were improved by reducing the computational image area. The data transfer and classification time was also reduced by using 1D search on the equi-height mosaicking image using the same height as the training data. The precision of the proposed system showed a small improvement, but this was just a side effect. Because the two approaches both use a linear SVM classifier with the same performance, the accuracy of the two approaches were also equal. Therefore, the precision of the proposed system showed improvement because it used the equi-height image to eliminate unnecessary image areas using the geometry information of the road scene. The proposed system performed faster than ocv-gpusvm [30] on an equal sampling level. The proposed system, therefore, can be added to other hypothesis methods to improve the detection accuracy.

The experiments were executed in a downtown alley during the day. We used 640x480 resolution images to clearly show the difference in the execution time. For training the SVM classifier, we used a training data set that modified positive images of 7,248 pedestrians and negative images

Table 6.4: Experimental result of pedestrian detection.

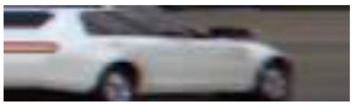
Approach	Res.	Samp. Lv.	Execution Time(ms)			Detection rate(%)		
			Resize	HOG	Total	Prec.	Rec.	F1
gpu-ocv	640	50	16	245	348	77.48	72.26	74.78
ours	x 480		26	22	63	88.17	80.52	84.17

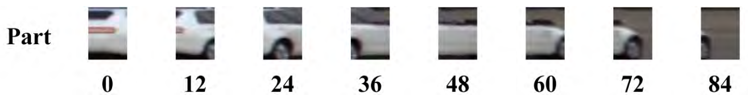
of 19,940 non-pedestrians.

6.2.4 Part-based Side Detection for Approaching Vehicle

Finally, we perform the experiments for side detection of the approaching vehicle. Before we perform the side detection of the approaching vehicle, we select the part of the side for high accuracy detection. Actually, the side of the approaching vehicle cannot be accurately detected because the side face is narrow. The narrow face of the side is difficult on which to perform feature extraction. Therefore, we use the warped image for side in order to extract better features for detection. However, the proposed system cannot use the whole side of the approaching vehicle because we often see the part of the approaching vehicle in the road scene. In Chapter 5, therefore, we suggested a part-based side detection algorithm. In this approach, we mainly use the front part for side detection. Then we use the rear part for verification and preparation for missed detection by the front part.

However, these parts for detection are not selected without basis. In order to select the part, we perform the experiments for detection rate with each part. For the experiment, we separate the side into eight parts. Fig 6.3


(a) Side of the warped vehicle.



(b) Side-parts of the warped vehicle for learning.

Figure 6.3: Each part for learning.

Table 6.5: Detection rate for each part of side.

	Recall (%)	Precision (%)	F1 Measure (%)
Part 0	4.0	66.7	7.5
Part 12	46.0	92.0	61.3
Part 24	78.0	97.5	86.7
Part 36	94.0	78.3	85.5
Part 48	92.0	61.3	73.6
Part 60	90.0	72.6	80.4
Part 72	88.0	93.6	90.7
Part 84	2.0	50.0	3.8

shows separated parts of eight for learning. After the part is separated, we perform the learning and detection for on 300 images of warped vehicles and non-vehicle objects. Table 6.5 illustrates the results of detection for each part. In Table 6.5, we can see that Parts 24 and 72, including a front and rear tire, are better than other parts in precision and recall. Fig 6.4 shows the result of detection by each section. Therefore, we select Part 72 as the main part for side detection. Then, Part 24 is used for verification or preparation for missed detection by Part 72.

After the part for side detection is selected, we perform experiments

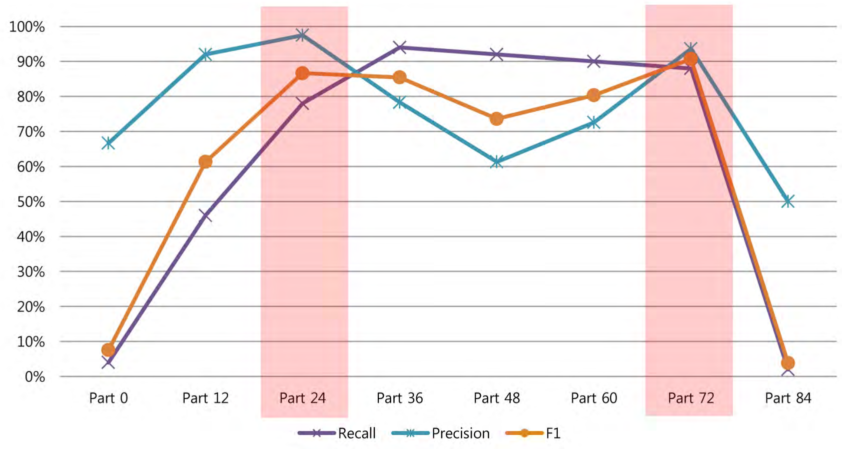


Figure 6.4: Detection rate of each part.

for side detection of the approaching vehicle. As shown in Table 6.6, the experimental result shows the detection rate and execution time. The proposed system shows a highly-accurate detection rate, because this approach is performed on each side of road scene. However, the proposed approach cannot dramatically improve the execution time of part-based side detection compared with our forward vehicle detection. Because we use the segmented regions of equi-height mosaicking image and perform the warping for segmented regions, the proposed system consumes a relatively large amount of execution time. As equi-height mosaicking image generation, however, we also control the sampling level for equi-height peripheral mosaicking image generation.

The experiments were executed on a downtown street and highway during the day. We used 640x480 resolution images to clearly show the

Table 6.6: Experimental result of part-based side detection

Road Type	Res.	Samp. Lv.	Execution Time(ms)			Detection rate(%)		
			EHPMI	HOG	Total	Prec.	Rec.	F1
Highway	640 x 480	40	71	7	82	93.12	91.39	92.25
Downtown			72	8	85	89.01	85.74	87.34

difference in the execution time. For training the AdaBoost classifier and HOG feature, we used a training data set that modified positive images of 1,712 front/rear parts and negative images of 8,964 non-parts.

Chapter 7

Conclusion

In this thesis, the proposed system presents a real-time dynamic obstacle detection algorithm using equi-height mosaicking image on the road scene. We describe the two types of obstacle detection algorithm in detail. The first algorithm is the forward dynamic obstacle detection method using equi-height mosaicking image. The algorithm performs detection for the complete shapes of forward vehicles and pedestrians. The second algorithm is the part-based side detection method of an approaching vehicle using equi-height peripheral mosaicking image. The algorithm performs detection for the approaching vehicles displaying incomplete shapes. In order to detect the various types of dynamic obstacles in real time, we suggest two types of new image representations, called equi-height mosaicking image and equi-height peripheral mosaicking image.

The proposed system presents three contributions. The first contribution is that the proposed system reduces the processing region for the obstacle detection. The contribution has two advantages: high precision and fast processing. On the other hand, the reduction of processing regions means low computation and low false positives.

The second contribution is that the proposed system presents the equi-height mosaicking image for fast detection. The equi-height mosaicking image is used as effective input on GPU-based acceleration. The image representation has dramatically reduced repetitive function calls using equi-height images. The third contribution is that the proposed system detects the approaching vehicles that exhibit only incomplete shapes. In the road scene, the approaching vehicle is a dangerous obstacle, but many detection systems cannot detect the approaching vehicle. Thus the proposed system presents the part-based side detection method for approaching vehicles. The method has detected the approaching vehicle and can warn about the dangerous situation.

Actually, the proposed system used four types of image representations for detection. The first representation is the equi-height image, which means a road geometry-based image strip with equal heights. The simple image representation presents high precision of detection. The second representation is the peripheral warped image that means warped side-region image-by-image warping. The image representation presents a wide region for detailed feature extraction by side-region warping. The third representation is equi-height mosaicking image, which is a long image concatenated by equi-height images. The fourth representation is equi-height peripheral mosaicking image, which means a long image concatenated by the peripheral warped image. The two kinds of image representations present high GPU processing speed in detection.

In this thesis, we have assumed that the road is flat and that all obstacles exist on the road. With these assumptions, we suggest two types of obstacle detection algorithms: forward dynamic obstacle detection and part-based side detection for approaching vehicles. In order to detect the forward dynamic obstacle, the proposed system has first generated the equi-height images on the road scene. Next, the proposed system has also made the equi-height mosaicking image as an effective input image for GPU acceleration. Then, the proposed system has detected vehicles and pedestrians, including the complete shape using the equi-height mosaicking image in real-time. Finally, the proposed system performs the coordinate transform and grouping to decide on the final detected region. However, the previous approach cannot detect obstacles with incomplete shapes. Therefore, we suggest part-based side detection for approaching vehicles using equi-height peripheral mosaicking image. This approach detects the front side-shape and rear side-shape of the vehicle. In order to detect the approaching vehicle, the proposed system has generated the equi-height peripheral mosaicking image from extracted side-regions of the equi-height mosaicking image. Next, the proposed system has performed the GPU accelerated 1D search on equi-height peripheral mosaicking image. Then, the proposed system has performed the coordinate transform in two stages and has performed the grouping for multiple detected regions. Finally, the proposed system has detected dangerously-approaching vehicles included those with incomplete shapes.

In the future, we will improve the accuracy using a new feature on the equi-height mosaicking image. The proposed system used the HOG feature to quickly detect vehicles and pedestrians at the same time. However, various features are used for accurate vehicle detection in many studies. Therefore, we will develop the improved accurate obstacle detection method using a new feature without reducing processing speed. And we will develop more effective tracking on equi-height mosaicking image. In order to perform reliable detection, we need a more effective tracking algorithm suitable to equi-height mosaicking image. Therefore, we will develop an effective and fast tracking algorithm on equi-height mosaicking image.

Appendix A

AR-based Full-Windshield HUD Simulator

Today, most smart vehicles developed by automobile companies are established Head-Up Display (HUD) system to effectively visualize various information for driving. Especially, many automobile companies have developed HUD system using large See-through display to offer more various information. However, see-through HUD system using the see-through display is not immersive than Full windshield HUD system [34] using the windshield display. Therefore, we suggest Augmented Reality based Full-Windshield HUD (AR-based FW HUD) simulator as indoor test-bed.

The AR-based FW HUD system can effectively visualize the various information for driving. Also it prevents the distracted driving to warn

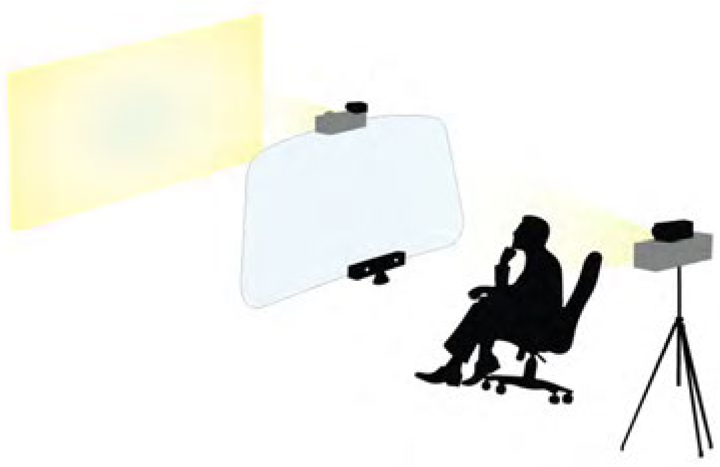


Figure A.1: Concept of AR based full windshield HUD simulator.

the dangerous obstacles while driving. In this thesis, we use the AR-base FW HUD simulator to effectively visualize the result of the vehicle and pedestrian detection. Therefore, we also present an indoor simulator to visualize the warning for the dangerous obstacles.

A.1 System Setup

In this section, we describe the setup of AR-based FW HUD simulator. The AR-based FW HUD simulator is made to be similar to real vehicle. The prototype simulator consists of the real windshield, projector, camera, and transparent reflection film. Fig. A.1 illustrates the concept of AR-based full windshield HUD simulator.

In Fig. A.1, one projector is used to display the road scene while driving. The other projector is used to visualize the detected obstacles on the windshield. And camera is used to calibrate augmented information based on driver's eye position.

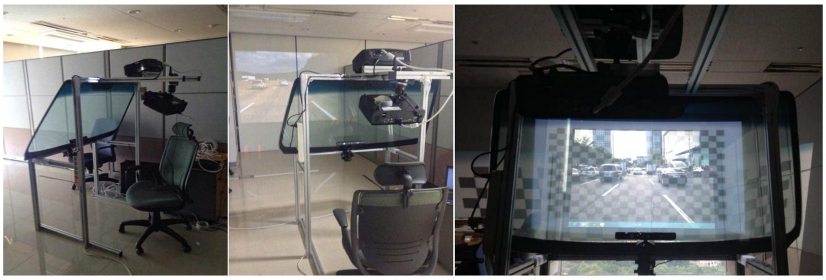


Figure A.2: AR-based full windshield HUD simulator.

Fig. A.2 shows the real AR-based FW HUD simulator created as the concept. In this simulator, we use the wide-angle projector to offer the widely field of view for driver.

A.2 Calibration

In order to visualize the augmented information, the proposed system performs the calibration based on fixed driver's eye position [34, 35]. In this case, we apply the simple calibration method to align between the detection region projected on windshield and the obstacle of road scene. The calibration method performs two stages. In the first stage, we remove the distortion of the windshield. In this case, the proposed system computes the homography between the original image and projected image on the windshield. The unwarped image to remove the distortion is computed by the following equation:

$$I_{uw} = H_{IW}^{-1} * I_{in} \tag{A.1}$$

where I_{uw} is the unwarped image, I_{in} is the original input image, and H_{IW}^{-1} means inverse homography between the original image and projected image on the windshield. When the unwapred image is projected on windshield, the image is displayed without distortion.

After the distortion of the windshield is removed, we perform the alignment between the detection region projected on the windshield and the detected region projected on the wall in the second stage. In this case, we also compute the homography between the projected image on the wall and the projected unwarped image on the windshield. The final aligned image is represented by the following equation:

$$H_{total} = H_{WW} * H_{IW}^{-1} \tag{A.2}$$

$$I_{new} = H_{total} * Iin \tag{A.3}$$

where H_{WW} means the homography between the detection region projected on the windshield and the detected region projected on the wall.

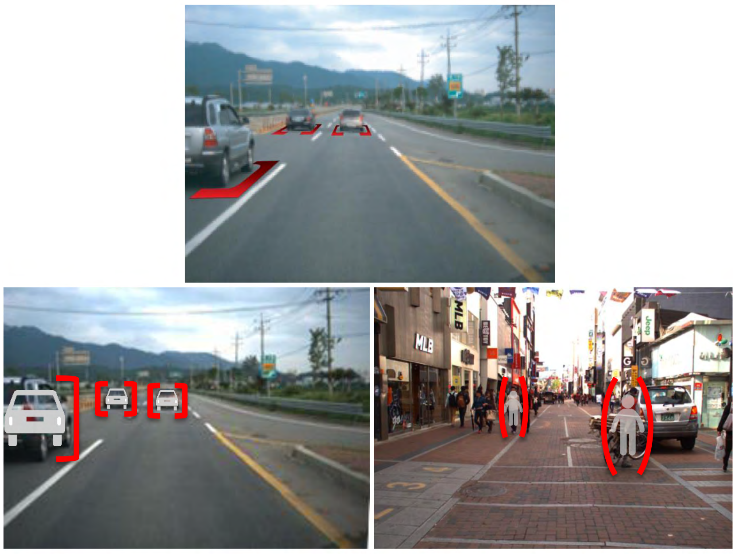


Figure A.3: Various concepts for AR-based visualization.

And, I_{new} is finally the transformed image to correctly align in the detected region of road scene. The proposed system can visualize the detected obstacle by projecting the I_{new} on windshield.

A.3 Dynamic Obstacle Visualization

In this thesis, we suggest the special visualization. According to Gestalt Laws of Perceptual Organization, objects grouped together are seen as a whole by Law of closure []. Therefore, we designed the shape of square bracket to visualize the detected obstacle. Fig. A.3 shows various shape of designed square bracket.

After the shape for visualization is designed, we augment the detected obstacle on AR-based FW HUD Simulator. Fig. A.4 and Fig. A.5 show the result of augmented vehicle on windshield.

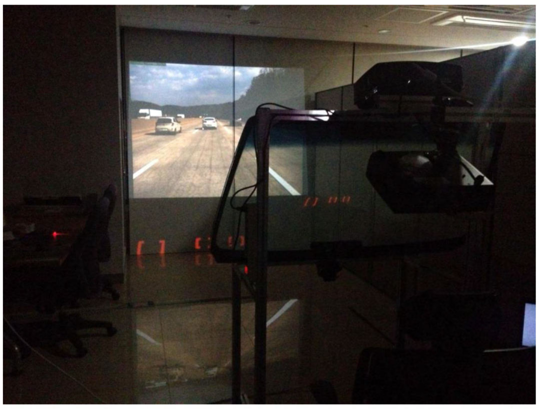


Figure A.4: The AR-based FW HUD Simulator.

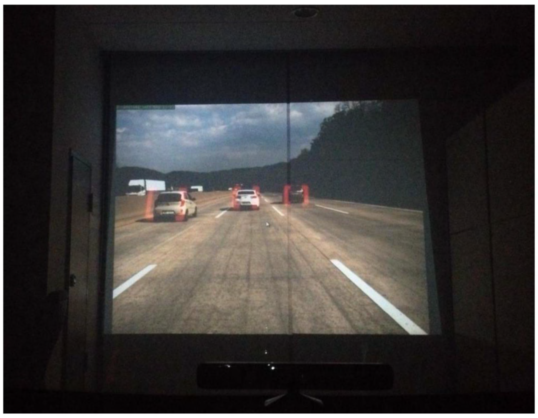


Figure A.5: The result of augmented vehicle on full windshield.

Bibliography

[1] “The history of the automobile,” http://inventors.about.com/library/weekly/aacarssteama.htm/.

[2] M. W. Park, K. H. Jang, and S. K. Jung, “Panoramic vision system to eliminate driverś blind spots using a laser sensor and cameras,” *International Journal of Intelligent Transportation Systems Research*, vol. 10, no. 3, pp. 101–114, 2012.

[3] G. P. Stein, O. Mano, and A. Shashua, “Vision-based acc with a single camera: bounds on range and range rate accuracy,” in *Intelligent vehicles symposium, 2003. Proceedings. IEEE.* IEEE, 2003, pp. 120–125.

[4] “Short wave infrared night vision camera systems for vehicle navigation,” http://www.sensorsinc.com/nightvision.html.

[5] “Nissan around view monitor,” http://www.nissan-global.com/EN/TECHNOLOGY/OVERVIEW/avm.html/.

[6] V. Prisacariu and I. Reid, “fastHOG - a real-time GPU implementation of HOG,” Department of Engineering Science, Oxford University, Tech. Rep. 2310/09, 2009.

[7] W. Najm, B. Sen, J. Smith, and B. Campbell, “Analysis of light vehicle crashes and pre-crash scenarios based on the 2000 general estimates system,” Tech. Rep., 2003.

[8] “Top 25 causes of car accidents,” http://seriousaccidents.com/legal-advice/top-causes-of-car-accidents/.

[9] “Missouri department of transportation - 2009 missouri state highway system: Traffic crash statistics,” http://www.modot.org/safety/trafficaccidentstatistics.htm.

[10] “Mission impossible - ghost protocol,” http://en.wikipedia.org/wiki/Mission:_Impossible_%E2%80%93_Ghost_Protocol.

[11] B. D. B. Dai, Y. F. Y. Fu, and T. W. T. Wu, “A Vehicle Detection Method via Symmetry in Multi-Scale Windows,” *Audio, Transactions of the IRE Professional Group on*, pp. 1827–1831, May 2007.

[12] L.-W. Tsai, J.-W. Hsieh, and K.-C. Fan, “Vehicle detection using normalized color and edge map.” *IEEE Transactions on Image Processing*, vol. 16, no. 3, pp. 850–864, Mar. 2007.

[13] Z. Sun, G. Bebis, and R. Miller, "On-road vehicle detection using evolutionary gabor filter optimization," *Intelligent Transportation Systems, IEEE Transactions on*, vol. 6, no. 2, pp. 125–137, 2005.

[14] ——, "On-road vehicle detection: A review," *Pattern Analysis and Machine Intelligence, IEEE Transactions on*, vol. 28, no. 5, pp. 694–711, 2006.

[15] N. Dalal and B. Triggs, "Histograms of oriented gradients for human detection," in *Computer Vision and Pattern Recognition, 2005. CVPR 2005. IEEE Computer Society Conference on*, vol. 1. IEEE, 2005, pp. 886–893.

[16] F. Han, Y. Shan, R. Cekander, H. S. Sawhney, and R. Kumar, "A two-stage approach to people and vehicle detection with hog-based svm," in *Performance Metrics for Intelligent Systems 2006 Workshop*, 2006, pp. 133–140.

[17] X. Wang and B. E. Shi, "Gpu implemention of fast gabor filters," in *Circuits and Systems (ISCAS), Proceedings of 2010 IEEE International Symposium on.* IEEE, 2010, pp. 373–376.

[18] F. K. F. Kong, Q. Y. Q. Ye, N. Z. N. Zhang, K. L. K. Lu, and J. J. J. Jiao, "On-road vehicle detectioin using histograms of multi-scale orientations," *Audio, Transactions of the IRE Professional Group on*, pp. 212–215, Sep. 2009.

[19] R. Benenson, M. Mathias, R. Timofte, and L. Van Gool, “Pedestrian detection at 100 frames per second,” in *Computer Vision and Pattern Recognition, IEEE Computer Society Conference on.* IEEE, 2012, pp. 2903–2910.

[20] J. Yan, X. Zhang, Z. Lei, S. Liao, and S. Z. Li, “Robust Multi-resolution Pedestrian Detection in Traffic Scenes,” in *Computer Vision and Pattern Recognition (CVPR), 2013 IEEE Conference on*, 2013, pp. 3033–3040.

[21] W. Burger and M. J. Burge, *Digital image processing.* Springer, 2008.

[22] M. Nieto and L. Salgado, “Real-time robust estimation of vanishing points through nonlinear optimization,” pp. 772 402–772 402–14, 2010.

[23] P. H. S. Torr and A. Zisserman, “Mlesac: A new robust estimator with application to estimating image geometry,” *Computer Vision and Image Understanding*, vol. 78, p. 2000, 2000.

[24] F. Devernay, “A Non-Maxima Suppression Method for Edge Detection with Sub-Pixel Accuracy,” INRIA, Tech. Rep. RR-2724, Nov. 1995. [Online]. Available: http://hal.inria.fr/inria-00073970

[25] M. W. Park and S. K. Jung, “Gpu-based real-time pedestrian detection and tracking using equi-height mosaicking image.” in *ICONIP*

(3), ser. Lecture Notes in Computer Science, M. Lee, A. Hirose, Z.-G. Hou, and R. M. Kil, Eds., vol. 8228. Springer, 2013, pp. 409–416.

[26] ——, "Real-time vehicle detection using equi-height mosaicking image," in *ACM Reliable And Convergent Systems, 2013 International Conference on.* ACM, 2013, p. (Accepted).

[27] Z. Zhang, "A flexible new technique for camera calibration," *Pattern Analysis and Machine Intelligence, IEEE Transactions on*, vol. 22, no. 11, pp. 1330–1334, 2000.

[28] I. Safonova, H. Leeb, S. Kimb, and D. Choib, "Intellectual two-sided card copy."

[29] "Inria person dataset," http://pascal.inrialpes.fr/data/human/.

[30] "Opencv gpu hog detector," http://docs.opencv.org/modules/gpu/doc/object_detection.html/.

[31] A. Ess, B. Leibe, and L. V. Gool, "Depth and appearance for mobile scene analysis," in *International Conference on Computer Vision (ICCV'07)*, October 2007.

[32] L. M. Manevitz and M. Yousef, "One-class svms for document classification," *J. Mach. Learn. Res.*, vol. 2, pp. 139–154, Mar. 2002.

[33] Y. Yang, "An evaluation of statistical approaches to text categorization," *Information retrieval*, vol. 1, no. 1-2, pp. 69–90, 1999.

[34] A. Sato, I. Kitahara, and Y. Kameda, "Visual navigation system on windshield head-up display," in *Proceedings of 13th . . .*, 2006.

[35] W. Wu, F. Blaicher, J. Yang, T. Seder, and D. Cui, "A prototype of landmark-based car navigation using a full-windshield head-up display system," in *AMC '09: Proceedings of the 2009 workshop on Ambient media computing.* ACM Request Permissions, Oct. 2009.

동등 높이 모자이킹 영상을 이용한 도로 영상의 움직이는 장애물 검출

박 민 우

경북대학교 대학원 컴퓨터학부
(지도교수 정순기)

(초 록)

오늘날 자동차 산업은 차량 성능 위주의 발전에서 벗어나, 교통사고에 의한 인명 손실을 막기 위한 지능형 안전 장치를 개발하기 위한 방향으로 발전하고 있다. 특히, 최근에는 운전자 주위의 차량과 보행자를 검출하여 운전자의 인지능력을 향상시킴으로서 교통사고를 미연에 방지하기 위한 능동형 안전장치(active safety system) 개발에 관한 연구가 활발히 이루어지고 있다.

본 논문에서는 동등 높이 모자이킹 영상(equi-height mosaicking image)을 이용하여, 도로 영상에서 차량 혹은 보행자와 같은 움직이는 장애물을 검출하는 효과적인 방법을 제안한다. 운전 중 장애물 검출은 상황에 따라 두 가지의 검출 방법을 제안한다. 첫번째 방법은 운전 중 전방에 나타나는 완전한 모습의 차량 혹은 보행자를 검출하는 방법이다. 이 방법은 운전자 전방의 차량 혹은 보행자를 실시간에 검출함으로서 효과적인 가시화를 통해 운전자의 인지능력을 향상시킬 수 있다. 제안하는 두번째 방법은 운전자의 차량을 추월 혹은 끼어들기 위한 목적으로 갑작스럽게 운전자 좌측 혹은 우측에 나타나는 차량의 부분적인 모습을 검출하는 방법이다. 이 방법은 운전자의 사각지대에서 나타나는 인지하기 어려운 차량을 검출하여 운전자에게 알려줌으로서 운전자의 부주의한 운전을 사전에 방지한다.

본 논문에서는 실시간 검출을 위해서 동등 높이 모자이킹 영상을 제안한다. 동등 높이 모자이킹 영상이란, 거리에 따라 추출된 지면으로부터 동일한 높이를 가지는 영상들을 하나의 긴 영상으로 이어붙인 영상표현 방법을 의미한다. 이 영상을 사용함으로서 GPU

내부 검출함수의 반복적인 호출을 줄임으로서 보다 빠른 검출을 수행한다. 또한, 동등 높이 모자이킹 영상을 생성하기 위한 영상의 추출 및 크기조절 모두 GPU 내부에서 수행함으로서 속도 저하를 최소화시킨다. 동등 높이 정합영상을 생성하기 위해서, 첫 단계로 입력영상을 버즈 아이 뷰(bird's-eye-view) 영상으로 와핑(warping)한 후 운전자로부터 일정한 거리에 따라 도로상의 위치들을 샘플링(sampling)한다. 샘플링된 위치를 원영상의 위치로 다시 변환한 뒤, 해당 위치에서 계산된 차량 혹은 보행자의 일정한 높이를 이용하여 동등 높이 영상을 생성한다. 샘플링된 위치의 수만큼 생성된 동등 높이영상을 크기조절하여 하나의 영상으로 이어붙여서(concatenate) 동등 높이 모자이킹 영상을 생성한다.

동등 높이 모자이킹 영상을 생성한 뒤, 이 영상으로부터 GPU로 가속화된 서포트벡터머신(SVM) 분류기를 사용하여 전방의 차량 혹은 보행자를 실시간에 검출한다. 동등 높이 모자이킹 영상에서 차량 혹은 보행자 검출이 완료되면 이를 원영상으로 좌표변환(coordinate transform)을 수행하고 중복검출되는 영역을 그룹화(grouping)함으로서 검출된 차량 혹은 보행자를 결정한다.

이와 동시에, 동등 높이 모자이킹 영상은 운전자 좌측 혹은 우측의 사각지대에서 접근하는 차량을 검출하기 위해서 다시 한번 사용된다. 앞서 획득한 동등 높이 모자이킹 영상 중 운전자 좌측 그리고, 우측에 해당하는 영상을 샘플링하여 관심영역을 지정한다. 그리고, 관심영역들을 차량의 옆면이 보일수 있도록 와핑(warping)한다. 와핑된 결과를 영역들을 이어붙여서 동등 높이 주변 모자이킹 영상(equi-height peripheral mosaicking)을 생성한다. 이 영상을 GPU로 가속화된 차량 옆면의 앞부분과 뒷부분에 대한 학습 분류기를 사용하여 차량의 옆면 중 일부분이 나타나는지 검출을 수행한다. 동등 높이 주변 모자이킹 영상에서 차량의 앞부분 혹은 뒷부분이 검출되었다면, 검출된 영역을 두 단계에 걸쳐서 원영상으로 좌표변환을 수행하고, 중복된 영역에 대해서 그룹화를 하여 운전자의 사각지대에서 접근하는 차량을 검출한다.

본 논문에서 제안된 방법은 운전자 전방에 설치된 단일 카메라를 이용하여 실시간에 입력되는 영상으로부터 두 가지의 특징이 다른 움직이는 차량과 보행자를 검출하여 경고해 줌으로서, 운전자의 부주의한 운전으로 인한 사고를 미연에 방지하기 위한 능동형 안전장치의 기반기술로 활용될 수 있다.

Thanks to loving parents and my wife