Stereo Vision for Obstacle Detection in Autonomous Vehicle Navigation

Sameera Kodagoda

B.Sc(Hons)

A THESIS SUBMITTED

FOR THE DEGREE OF MASTER OF ENGINEERING

DEPARTMENT OF ELECTRICAL AND COMPUTER ENGINEERING

NATIONAL UNIVERSITY OF SINGAPORE

2010

Acknowledgements

I would like to take this opportunity to express my gratitude to all those who offered their time and knowledge to help me complete this thesis. First and foremost, I would like to thank my supervisors, Prof. Ong Sim Heng and Dr. Yan Chye Hwang, for making me a part of this project and also for providing unwavering guidance and constant support during this research.

I would also like to extend my sincere gratitude to Dr. Guo Dong and Lim Boon Wah from DSO National Laboratories for their insightful discussions, useful suggestions and continuous feedback throughout the course of this project. During the first two semesters of my Master's degree, they reduced my workload and made sure that I had sufficient time to prepare for the examinations, and I am deeply thankful to them.

I had the pleasure of working with people in the Vision and Image Processing (VIP) Lab of the National University of Singapore (NUS): Dong Si Tue Cuong, Liu Siying, Hiew Litt Teen, Daniel Lin Wei Yan, Jiang Nianjuan and Per Rosengren. I appreciate the support they provided in developing research ideas and also in expanding my knowledge in the field of computer vision. In particular, I am grateful to Per Rosengren for introducing me to the LyX document processor, which was immensely helpful during my thesis writing. I would also like to thank Mr. Francis Hoon, the Laboratory Technologist of the VIP Lab, for his technical support and assistance.

I wish to mention with gratitude my colleagues at NUS, especially Dr. Suranga Nanayakkara and Yeo Kian Peen, for their immeasurable assistance during my Master's module examinations and thesis writing. A special thanks goes to my friend Asitha Mallawaarachchi for introducing me to my supervisors and the NUS community.

I am indeed grateful to NUS for supporting my graduate studies for the entire duration of three years as part of their employee subsidy program.

Last but not the least, I would like to thank my family: my parents Ranjith and Geetha Kodagoda, my sister Komudi Kodagoda and my wife Iana Wickramarathne for their unconditional love and support in every step of the way. Without them this work would never have come into existence.

Contents

Summary

Autonomous navigation has attracted an unprecedented level of attention within the intelligent vehicles community over the recent years. In this work, we propose a novel approach to a vital sub-problem within this field, obstacle detection. In particular, we are interested in outdoor rural environments consisting of semi-structured roads and diverse obstacles. Our autonomous vehicle perceives its surroundings with a passive vision system: an off-the-shelf, narrow baseline, stereo camera. An on-board computer processes and transforms captured image pairs to a 3D map, indicating the locations and dimensions of positive obstacles residing within 3m to 25m from the vehicle.

The accuracy of stereo correspondence has a direct impact on the ultimate performance of obstacle detection and 3D reconstruction. Therefore, we carefully optimize the stereo matching algorithm to ensure that the produced disparity maps are of expected quality. As a part of this process, we supplement the stereo algorithm by implementing effective procedures to get rid of ambiguities and improve the precision of output disparity. The detection of uncertainties helps the system to be robust against adverse visibility conditions (e.g., dust clouds, water puddles and over exposure), while sub-pixel precision disparity enables more accurate ranging at far distances.

The first and the most important step of the obstacle detection algorithm is to construct a parametric model of the ground plane disparity. A large majority of methods in this category encounter modeling digressions under direct or indirect influence of the non-flat ground geometry, which is intrinsic to semi-structured

terrains. For instance, the planar ground approximation suffers from non-uniform slopes and the v-disparity algorithm is prone to error under vehicle rolling and yawing. The suggested ground plane model on the other hand is designed by taking all such factors into consideration. It is composed of two parameter sets, one each for the lateral and longitudinal directions. The lateral ground profile represents the local geometric structure parallel to the image plane, while the longitudinal parameters capture variations occuring at a global scale, along the depth axis. Subsequently an obstacle map is produced with a single binary comparison between the dense disparity map and the ground plane model. We realize that it is unnecessary to follow any sophisticated procedures, since both inputs to the obstacle detection module are estimated with high reliability.

A comprehensive evaluation of the proposed algorithm is carried out using data simulations as well as field experiments. For a large part, the stereo algorithm performance is quantified with a simulated dense disparity map and a matching pair of random dot images. This analysis reveals that our stereo algorithm is only second to iterative global optimization, out of the compared methods. A similar analysis ascertains best suited procedures and parameters for ground plane modeling. The ultimate obstacle detection performance is assessed using field data accumulated over approximately 35km of navigation. These efforts demonstrate that the proposed method outperforms both planar ground and v-disparity methods.

List of Tables

List of Figures

Chapter 1

Introduction

1.1 Obstacle Detection Problem

The ability to detect and avoid obstacles is a critical functionality deemed necessary for a moving platform, whether it be manual or autonomous. Intuitively, any obstruction lying on the path of the vehicle is considered an obstacle; a more precise definition varies from nature of applications to different environments. Human drivers perform this task by fusing complex sensory perceptions and relating it to an existing knowledge base via cognitive processing. Before attempting any higher level tasks, an unmanned vehicle should also be equipped with a similar infrastructure in order to be able to plan safe paths from one location to another. Although seemingly trivial, it has proved surprisingly difficult to find techniques that work consistently in complex environments with multiple obstacles.

Because of its increasing practical significance, outdoor autonomous navigation has lately received tremendous attention within the intelligent vehicles research community. Outdoor environments are usually spread over much larger regions in contrast to indoor; even a relatively short outdoor mission may consist few kilometers of navigation. Due to this factor, manual rescue of unmanned vehicles

from serious failures can be a tedious task. It imposes a special challenge on the design of the vehicle to ensure that it is able to operate over large time spans without any errors, or, at least, to identify and correct for errors in time to avoid catastrophic failures. The difficulty level of this issue is particularly aggravated by the complexity of the environment, existence of previously unencountered obstacles and unfavorable weather conditions such as rain, fog, variable lighting and dust clouds. While much progress has been made towards solving the said problem in simpler environments, achieving the level of reliability required for true autonomy in completely new operating conditions still remains a challenge.

1.2 Contributions

In this thesis, a stereo vision based obstacle detection algorithm for an unmanned ground vehicle (UGV) is presented. The types of outdoor environments encountered by unmanned vehicles can be broadly considered under three categories: urban, semi-structured and off-road (Figure 1.1). The system we discuss here is particularly intended for detection of obstacles in semi-structured rural roads.

The presence of highly structured components in urban or highway environments typically translate the obstacle detection process to a simpler set of action strategies based on a-priori knowledge. For example, one may assume the ground surface in front of the vehicle to be of a planar nature for an urban road similar to that shown in Figure 1.1(a). On the other hand, approximating large topographic variations of a natural off-road terrain with a simple geometric model might cause the natural rise and fall of the terrain to be construed as obstacles (false positives) or worse, obstacles to go undetected (false negatives) due to overfitting. One possible way to detect obstacles in these complex off-road environments is to build accurate terrain models involving large numbers of parameters. The semi-structured, rural terrains we consider in our work are located somewhere between the two

(a) Structured urban road.

(b) Semi-structured rural road.

(c) Unstructured off-road terrain.

FIGURE 1.1: Different environments encountered in outdoor navigation.

extremes just described. Due to the coexistence of both urban and off-road geometric properties, a clear-cut definition of semi-structured terrains is not straight forward. Therefore, we deem a terrain to be of a semi-structured nature if its geometry cannot be globally represented by a single closed-form function (e.g., a planar equation), but can be approximated as an ensemble of equivalent local functions.

Despite its practical significance, there has been little effort to find a specific solution to this problem. Even though, one might argue that algorithms that work well for complex off-road environments will serve equally well for semi-structured environments, additional flexibility of the ground model would cause adverse effects in

some instances. Apart from that, enforcing a complex geometric model to a relatively simple terrain would result in redundant computations. On a similar note, we observe that non-flat ground modeling techniques designed for urban roads are affected by the vehicle oscillations occurring in semi-structured environments. Taking all these factors into consideration, we propose an obstacle detection algorithm that is ideally balanced between urban and off-road methods, in which assumptions valid under urban conditions are suitably modified in order to cope with vehicle pose and topographic variations. The main contribution of our work is the component that models ground stereo disparity as a piecewise planar surface in a time-efficient manner without compromising terrain modeling accuracy.

1.3 Thesis Organization

This section provides an overview of the thesis content, which will be presented in greater detail throughout the remaining chapters. Chapter 2 presents the background and previous research related to the central topic of this thesis. We review recent developments in the field of autonomous navigation and discuss different methods that have been applied for vision based obstacle detection. Chapter 3 briefly introduces the hardware and software architecture of our system. The next two chapters are devoted to major algorithmic components, stereo vision and obstacle detection. Chapter 4 begins with an introduction to general principles of stereo vision and proceeds to the details of camera calibration, stereo correspondence and 3D reconstruction. This is followed by a comprehensive discussion of the proposed ground plane modeling and obstacle detection algorithms in Chapter 5. Chapter 6 presents the experiments performed to demonstrate the feasibility and effectiveness of our approach and Chapter 7 concludes the thesis with a short discussion on potential future improvements.

Chapter 2

Background and Related Work

2.1 Autonomous Navigation Research

Researchers first pondered the idea of building autonomous mobile robots and unmanned vehicles in the late 1960s. The first major effort of this kind was *Shakey* [1], developed at Stanford Research Institute and funded by the *Defense Advanced Research Projects Agency* (DARPA), the research arm of the Department of Defense of the United States. *Shakey* was a wheeled platform equipped with a steerable TV camera, an ultrasonic range finder, and touch sensors, connected via a radio frequency link to its mainframe computer that performed navigation and exploration tasks. While *Shakey* was considered a failure in its day because it never achieved autonomous operation, the project established functional and performance baselines and identified technological deficiencies in its domain. The first notable success on unmanned ground vehicle (UGV) research was achieved in 1977, when a vehicle built by Tsukuba Mechanical Engineering Lab in Japan was driven autonomously. It managed to reach speeds of up to 30 kmph by tracking white markers on the street. It was programmed on a special hardware system, since commercial computers at that time were unable to match the required throughput.

The 1980s was a revolutionary decade in the field of autonomous navigation. The development efforts that began with *Shakey* re-emerged in the early part of this decade as the DARPA *Autonomous Land Vehicle* (ALV) [2]. The ALV was built on a Standard Manufacturing eight wheel hydrostatically driven all-terrain vehicle capable of speeds of up to 72 kmph on the highway and up to 30 kmph on rough terrain. The initial sensor suite consisted of a color video camera and a laser scanner. Video and range data processing modules produced road edge information that was used to generate a model of the scene ahead. The ALV road-following demonstrations began in 1985 at 3 kmph over a 1 km straight road, then improved in 1986 to 10 kmph over a 4.5 km road with sharp curves and varying pavement types, and in 1987 to an average 14.5 kmph over a 4.5 km course through varying pavement types, road widths, and shadows, while avoiding obstacles. In 1987, HRL Laboratories demonstrated the first off-road map and sensor-based autonomous navigation on the ALV. The vehicle traveled over a 600m stretch at 3 kmph on complex terrain with steep slopes, ravines, large rocks, and vegetation. As another division of this program by DARPA, the CMU navigation laboratory initiated the *Navlab* projects [3]. Since its inception in the late 1980s, the laboratory has produced a series of vehicles, *Navlab 1* through *Navlab 11*. It was also during this period that vision guided Mercedes-Benz robot van, designed by Ernst Dickmanns and his team at the Bundeswehr University of Munich, Germany, achieved 100 kmph on streets without traffic. Subsequent to that, the European Commission started funding the *EUREKA Prometheus Project* on autonomous vehicles [4]. The first culmination point of this project was achieved in 1994, when the twin robot vehicles VaMP and VITA-2 drove more than one thousand kilometers on a Paris multi-lane highway in standard heavy traffic at speeds up to 130 kmph. They demonstrated autonomous driving in free lanes, convoy driving, automatic tracking of other vehicles, and lane changes left and right with autonomous passing of other cars.

From 1991 through 2001, DARPA and the *Joint Robotics Program* collectively sponsored the DEMO I, II and III projects [5]. The major technical thrusts of these projects were the development of technologies for both on and off road autonomous navigation, improvement in automatic target recognition capabilities and enhancement of human supervisory control techniques. In 1995, Dickmanns re-engineered autonomous S-Class Mercedes-Benz took a 1600 km trip from Munich to Copenhagen and back, using saccadic computer vision and transputers to react in real time. The robot achieved speeds not exceeding 175 kmph with a mean time between human interventions of 9 km. Despite being a research system without emphasis on long distance reliability, it drove up to 158 km without human intervention. From 1996 to 2001, Alberto Broggi of the University of Parma launched the *ARGO Project* [6] which programmed a vehicle to follow the painted lane marks in an unmodified highway. The best achievement of the project was a journey of 2000 km over six days on the motorways of northern Italy, with an average speed of 90 kmph. For 94% of the time the car was in fully automatic mode, with the longest automatic stretch being 54 km. The vehicle was only equipped with a stereo vision setup, consisting of a pair of black and white video cameras, to perceive the environment.

In 2002, the DARPA Grand Challenge competitions were announced to further stimulate innovation in autonomous navigation field. The goal of the challenge was to develop UGVs capable of traversing unrehearsed off-road terrains autonomously. The inaugural competition, which took place in March 2004 [7], required UGVs to navigate a 240 km long course through the Mojave desert in no more than 10 hours; 107 teams registered and 15 finalists emerged to attempt the final competition, yet none of the participating vehicles navigated more than 5% of the entire course. The challenge was repeated in October 2005 [8]. This time, out of 195 teams registered, 23 raced and 5 reached the final target. Vehicles in the 2005 race passed through three narrow tunnels and negotiated more than 100 sharp left and

right turns. The race concluded through beer bottle pass, a winding mountain pass with a sheer drop-off on one side and a rock face on the other. All but one of the finalists surpassed the 11.78 km distance completed by the best vehicle in the 2004 race. Stanford's robot *Stanley* [9] finished the course ahead of all other vehicles in 6 hours 53 minutes and 58 seconds and was declared the winner of the DARPA Grand Challenge 2005. The third competition of this kind, known as the *Urban Challenge* [10], took place in November 2007 at the George air force base. The course involved a 96 km urban area course, to be completed in less than 6 hours. Rules included obeying all traffic regulations while negotiating with other traffic and obstacles and merging into traffic. The winner was *Tartan Racing*, a collaborative effort by Carnegie Mellon University and General Motors Corporation. The success of Grand Challenges has led to many advances in the field and other similar events such as the *European Land-Robot Trial* and *VisLab Intercontinental Autonomous Challenge*.

2.2 Vision based Obstacle Detection: Existing Approaches

The sensing mechanism of obstacle detection can be either active or passive. Active sensors, such as ultrasonic sensors, laser rangefinders and radars have often been used since they provide easy-to-use refined information of the surrounding area. But they suffer from intrinsic limitations as discussed by Discant *et al.* in [11]. On the other hand, the more widely used passive counterpart, vision, offers a large amount of perceptual information that requires further processing before obstacles can be detected. The passive nature of the vision sensor is preferred in some application areas, e.g., military industry and multi-agent systems, since it is relatively free of signal interference. Other appealing features of vision in contrast to active range sensors include low cost, rich information content and

higher spatial resolution. We understand that a comprehensive review of different sensing technologies, fusion methods and obstacle detection algorithms can be overwhelming. Therefore, in the remainder of this chapter we limit our interest to vision based obstacle detection. For ease of interpretation, it is divided into three sections: appearance, motion and stereo.

2.2.1 Appearance

In the majority of applications, obstacles will largely vary from one another in terms of intensity, color, shape and texture. Therefore, in reality it is impractical to accurately represent the appearance of obstacles using a finite number of basis functions. On the other hand, enforcing an appearance model (e.g., a color model) to the ground plane is more reasonable in most instances. When the expected appearance of the ground plane is known, obstacles can be detected by comparing the visual cues of the captured scene against the hypothesized ground model. While color is the most popular choice for this purpose, texture has also been occasionally used.

The algorithm presented in [12] uses brightness and color histograms to detect obstacle boundaries in an image. It assumes that the ground plane close to the robot is visible and hence the bottom part of the image corresponds to safe ground. A local window is run over the entire image and, intensity gradient magnitude, normalized RGB color, and normalized HSV color histograms are computed. The non-overlapping area between these histograms and equivalent histograms of safe ground is used to determine obstacle boundaries. In [13], the authors recognize the decomposition between color and intensity in HSI space to be desirable for obstacle detection. A trapezoidal area in front of the robot is used to construct reference histograms of hue and intensity, which are then compared with the same attributes at a pixel level to detect obstacles.

The methods which depend on a single attribute of appearance, work sufficiently well in test environments that satisfy a set of underlying conditions. It is only when they are conducted in more general environments that failures occur due to the violations of stipulated assumptions. This problem is difficult to overcome using monocular vision alone. As a solution, researchers have proposed algorithms that fuse sensing modalities such as color and texture with geometric cues drawn from laser range finders, stereo vision or motion. The system presented in [14] comes under this category. It tracks corner points through an image sequence and group them into coplanar regions using a method called an H-based tracker. The H-based tracker employs planar homographies and is initialized by 5-point planar projective invariants. The color of these ground plane patches are subsequently modeled and a ground plane segmentation is carried out using color classification. During the same period, Batavia and Singh developed a similar algorithm [15] at the CMU robotics institute, in which the main difference is the utilization of stereo vision in place of motion tracking. They estimate the ground plane homography with a stereo calibration procedure and use inverse perspective mapping to warp the left image on to the right image or vice versa. The original and warped images are differenced in the HSI space to detect obstacles. The result is further improved using an automatically trained color segmentation method. In [16], a road segmentation algorithm that integrates information from a registered laser range finder and a monocular color camera is given. In this method laser range information, color, and texture are combined to yield higher performance than individual cues could achieve. In order to differentiate between small patches belonging to the road and obstacles, a multi-dimensional features vector is used. It is composed of six color features, two laser features and six laser range features. The feature vectors are manually labeled for a representative set of images, and a neural network is trained to learn a decision boundary in feature space. A similar sensor fusion system [17] developed at the CMU robotics institute incorporates infrared image intensity in addition to the types of features used in [16]. Their

approach is to use machine learning techniques for automatically deriving effective models of the classes of interest. They have demonstrated that the combination of different classifiers exceeds the performance of any individual classifier in the pool. Recent work in the domain of appearance based obstacle and road detection include [18] and [19]. In [18], Hui *et al.* propose a confidence-weighted Gabor filter to compute the dominant texture orientation at each pixel and a locally adaptive soft voting (LASV) scheme to estimate the vanishing point. Subsequently, the estimated vanishing point is used as a constraint to detect two dominant edges for segmenting the road area. While the emphasis of this work is to accurately segment general roads, it does not guarantee the detected path to be free of obstacles. In [19], authors combine a series of color, contextual and temporal cues to segment the road. Contextual cues utilized include horizon line, vanishing point, 3D scene layout (sky pixels, vertical surface pixels and ground pixels) and 3D road geometry (turns, straight road and junctions). Two different Kalman filters are used to track the locations of horizon and vanishing point and an exponentially weighted moving average (EWMA) model is used to to predict expected road dynamics in the next time frame. Ultimately confidence maps computed based on multiple cues are combined in a Bayesian framework to classify road sequences. The road classification results presented in [19] are limited to urban road sequences.

2.2.2 Motion

With the advent of high-speed and low-cost computers, optical flow has become a practical means of robotic perception. It provides powerful cues for understanding the scene structure. The methods proposed by Ilic [20] and Camus [21] represent some early work in optical flow based obstacle detection. Ilic's algorithm builds a model for the optical flow field of points lying on the ground at a certain robot speed. While in operation, the algorithm compares the optical flow model to the

real optical flow and interprets the anomalies as obstacles. In [21], the fundamental relationship between time-to-collision (TTC) and flow divergence is used to good effect. It describes how the flow field divergence is computed and also how steering, collision detection, and camera gaze control cooperate to avoid obstacles while the robot attempts to reach the specified goal. More recent work in motion based obstacle detection include [22, 23, 24]. The system proposed in [22] performs a motion wavelet analysis of the optical flow equation. Furthermore, the obstacles moving at low speeds are detected by modeling the road velocity with a quadratic model. In [23], the detailed algorithm detects obstacle regions in an image sequence by evaluating the difference between calculated flow and modeled flow. Unlike many other optical flow algorithms, this algorithm allows camera motions containing rotational components, the existence of moving obstacles, and it does not require the focus of expansion (FOE). The algorithm only requires a set of model flows caused by planar surface motions and assumes that the ground plane is a geometrically planar surface. The algorithm proposed in [24] is intended to detect obstacles in outdoor unstructured environments. It firstly calculates the optical flow using the KLT tracker, and then separately evaluates the camera rotation and FOE using robust regression. A Levenberg-Marquardt non-linear optimization technique is adopted to refine the rotation and FOE. Eventually, the inverse TTC is used in tandem with rotation and FOE to detect obstacles in the scene.

2.2.3 Stereo Vision

The real nature of obstacles is better represented by geometric properties rather than attributes such as color, texture or shape. For instance, it makes more intuitive sense for an object protruding above the ground to be regarded as an obstacle, rather than an object that is different in color with reference to the ground plane. The tendency within the intelligent vehicles community to deploy stereo

vision to exploit the powerful interpretive 3D characteristics is a testimony to this claim. It is by far the most popular choice for vision based obstacle detection.

One class of stereo vision algorithms geometrically model the ground surface prior to obstacle detection, and hence is collectively termed ground plane obstacle detection (GPOD) methods. Initial work in this category dates us back to the work of Zheng *et al.* [25] and Ferrari *et al.* [26] in the early 90's. In the context of GPOD, "plane" does not necessarily have to be a geometrically flat plane, but could be a continuous smooth surface. However, in its simplest form, successful obstacle detection has been achieved by approximating the ground surface with a geometric plane [27, 28, 29]. Researchers have investigated flexible modeling mechanisms to extend the role of GPOD beyond indoor mobile robot navigation and adaptive cruise control. The v-disparity method, proposed by Labayrade *et al.* [30], is an important landmark technique in this category. Each row in the v-disparity image is given by the histogram of the corresponding row in the disparity image. Coplanar points in Euclidean space become collinear in v-disparity space, thus enabling a geometric modeling procedure that is robust against vehicle pitching and correspondence errors. Even though originally meant to model road geometry in highway environments as a piecewise planar approximation, it has been successfully applied to a number of cross-country applications [31, 32, 33, 34, 35]. The v-disparity image computation method presented by Broggi *et al.* in [31] does not require a pre-computed disparity map, but directly calculates the v-disparity image with the aid of a voting scheme that measures the similarity between vertical edge phases across the two views. This method has been successfully used in the *TerraMax* robot, one of the five contestants to complete the 2005 DARPA Grand Challenge. In a different algorithm presented in [36], instead of relying on the flatness of the road, the authors model the vertical road profile as a clothoid curve. Structurally, this method is very similar to the v-disparity algorithm since the road profile is modeled by fitting a 2D curve to a set of 2D points corresponding

to the lateral projection of the reconstructed 3D points.

Ground geometry modeling is not an essential requisite of traversability evaluation; the second class of algorithms we discuss falls into this category. A large majority of these algorithms is based on the construction and successive processing of a digital elevation map (DEM), also known as a Cartesian height map. It is a two dimensional grid in which each cell corresponds to a certain portion of the terrain. The terrain elevation in each cell is derived from range data. In principle, one could determine the traversability of a given path by simulating the placement of a 3-D vehicle model over the computed DEM, and verifying that all wheels are in contact with the ground while leaving the bottom of the vehicle clear. Initial stereo vision based work in this category started in the early 90's [37, 38]. More recent developments include [39, 40, 41] in relation to ground vehicles, and [42, 43, 44] in relation to planetary rovers. DEM based approaches, besides being computationally heavy, suffer from non-uniform elevation maps due to nonlinear back-projection from the image domain. Therefore, it is either represented by a multi-resolution structure (which makes the obstacle detection task tedious) or interpolated to an intermediate density uniform grid (which might cause a loss of resolution in some regions). Manduchi *et al.* propose a slightly different approach to the same problem in [45]. They give an axiomatic definition to obstacles using the relative constellation of scene points in 3D space. This rule not only helps distinguish between ground and obstacle points, but also automatically clusters obstacle points into obstacle segments. The algorithms discussed in [46] and [47] are inspired from [45], but modified for better performance, computational speed and robustness against outliers.

Chapter 3

System Overview

3.1 Hardware Platform

In our work a Polaris Ranger XP vehicle (Figure 3.1), which is particularly well designed for semi-structured and off-road conditions, is used as the UGV platform. It is powered by a liquid-cooled Polaris 700 twin-cylinder engine and equipped with electronic fuel injection for fast starts even in extreme temperatures and altitudes. The independent front and rear suspension enables it to maintain high ground clearance and smooth navigation on uneven roads. A complete list of specifications of the Ranger XP can be found in [48].

The stereo vision sensor used in our work is a Bumblebee2 narrow baseline camera manufactured by Point Grey [49]. The expectation is to produce an obstacle map within a range of 3m to 25m from the UGV. To achieve this distance requirement, the Bumblebee2 is mounted on the UGV at about 1.7m from the ground level and tilted downwards by approximately 15 degrees. The Bumblebee2 comprises two high quality Sony ICX204 progressive scan CCD cameras, with 6mm focal length lenses, installed at a stereoscopic baseline of 12 cm. It is able to capture image pairs at a maximum resolution of 1024×768 with accurate time synchronization

FIGURE 3.1: The UGV platform: Polaris Ranger.

and has a DCAM 1.31 compliant high speed IEEE-1394 interface to transfer the images to the host computer. It is factory calibrated for lens distortion and camera misalignments, to ensure consistency of calibration across all cameras and eliminate the need for in-field calibration. During the rectification process, epipolar lines are aligned to within 0.05 pixels RMS error. Calibration results are stored on the camera, allowing the software to retrieve image correction information without requiring camera-specific files on the host computer. The camera case is also specially designed to protect the calibration against mechanical shock and vibration. The run-time camera control parameters can be set to automatic mode to compensate for global intensity fluctuations. More details on Bumblebee2, including a complete list of calibration parameters, can be found in Appendix A.

3.2 Software Architecture

The building blocks of the proposed stereo vision based obstacle detection algorithm are depicted in Figure 3.2. As the initial step, the captured stereo image

pairs are rectified using the calibration parameters together with the Triclops software development kit (SDK) provided by the original equipment manufacturer Point Grey. The images can be rectified to any size, making it easy to change the resolution of stereo results depending on speed and accuracy requirements. After rectification, the images are input to the stereo correspondence module which performs a series of operations to produce a dense disparity map of the same resolution. A binary uncertainty flag is attached to each pixel of the computed disparity map; if the flag is on, it indicates that the disparity calculation is ambiguous and hence is left undetermined. For all unambiguous instances the disparity will have a pixel precision value as well as a sub-pixel correction. During the next stage, the pixel precision disparity map is used by the ground plane modeling algorithm. It adapts a heuristical approach to sample probable ground pixels which are subsequently used to estimate the lateral and longitudinal ground profiles. By comparing the pixel precision disparity map against the computed ground plane model, obstacles can be detected in the image domain, whereas the sub-pixel correction is utilized only during the ultimate 3D representation. The next few chapters are devoted to an in depth discussion of the theoretical aspects, design considerations and empirical performance of the above modules.

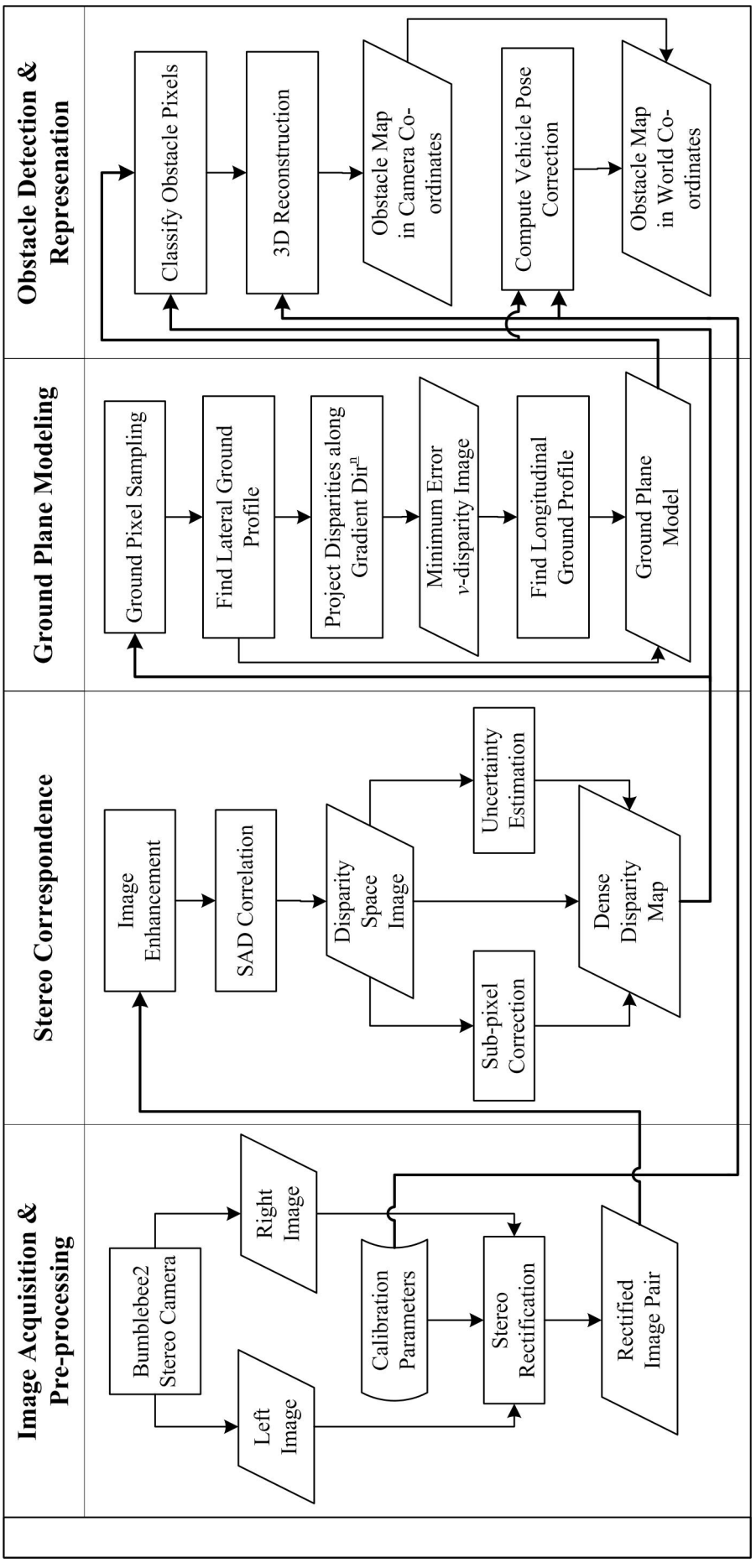


FIGURE 3.2: System architecture.

Chapter 4

Stereo Vision

The perception of depth, which is the intrinsic feel for relative depth of objects in an environment, is an essential requisite for many animals. Among many possibilities, depth perception based on the different points of view of two overlapping optical fields is the most widespread and reliable method. This phenomenon, commonly known as stereopsis, was first formally discussed in 1838 in a paper published by Charles Wheatstone [50]. He pointed out that the positional disparity in the two eyes' images due to their horizontal separation yielded depth information. Similarly, given a pair of two-dimensional digital images, it is possible to extract a significant amount of auxiliary information about the geometric content of the scene being captured. In what follows, we discuss the computational stereo vision subsystem of our work: image formation, theory of stereo correspondence and re-projection of image point pairs back into 3D space.

4.1 General Principles

4.1.1 Pinhole Camera Model

The first photogrammetric methods were developed in the middle of the 19th century by Laussedat and Meydenbauer for mapping purposes and reconstruction of buildings [51]. These photogrammetric methods assumed perspective projection of a three-dimensional scene into a two-dimensional image plane. Image formation by perspective projection corresponds to the pinhole camera model (also called the perspective camera model). There are other kinds of camera models describing optical devices such as fish-eye lenses or omnidirectional lenses. In this work we restrict ourselves to the pinhole model since it represents the most common image acquisition devices, including ours.

The pinhole camera model assumes that all rays coming from a scene pass through one unique point of the camera, the center or focus of projection (O). The distance between the image plane (π), and O is the focal length (f), and the line passing through O perpendicular to π is the optical axis. The principal point or image center (o) is the intersection between π and the optical axis. Figure 4.1 illustrates

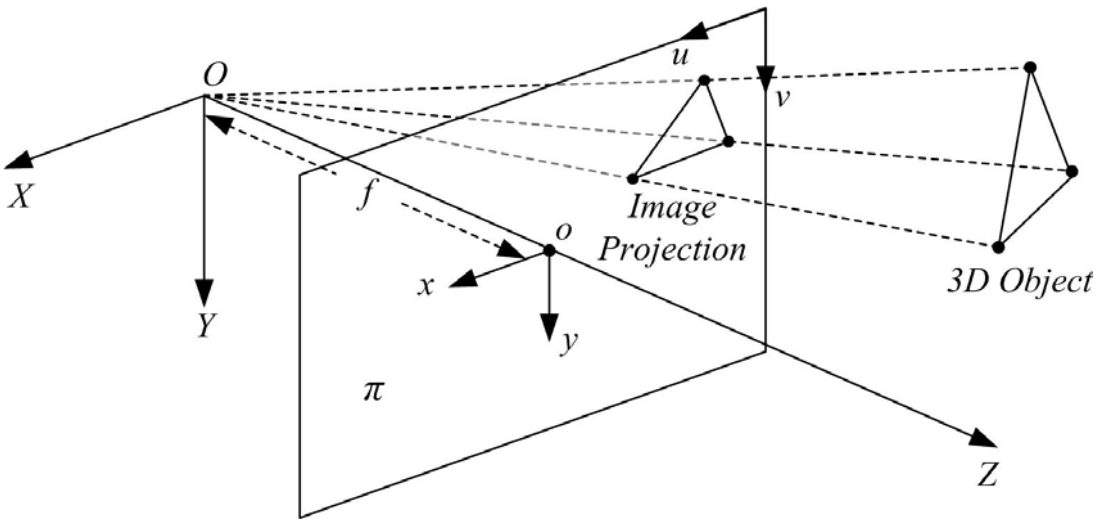


FIGURE 4.1: Pinhole camera model.

the camera model described thus far. Intuitively, the image plane should be placed behind the focus of projection but this will invert the projected image. In order to prevent this the image plane is moved in front of O. The human brain performs a similar correction during its visual cognition process. Furthermore, the origin of the camera coordinate system $\{X, Y, Z\}$ coincides with O and the Z axis is collinear with the optical axis. The origins of image coordinate system $\{x, y\}$ and pixel coordinate system $\{u, v\}$ are placed at o and top left corner of the image plane respectively. The relationship between camera and image coordinates can be obtained using similar triangles:

$$\frac{x}{X} = \frac{y}{Y} = \frac{f}{Z}$$

which can be represented in homogeneous coordinates as

$$\begin{bmatrix} x \\ y \\ 1 \end{bmatrix} = \begin{bmatrix} f & 0 & 0 & 0 \\ 0 & f & 0 & 0 \\ 0 & 0 & 1 & 0 \end{bmatrix} \begin{bmatrix} X \\ Y \\ Z \\ 1 \end{bmatrix} \tag{4.1}$$

Note that the factor $1/Z$ makes these equations nonlinear, hence neither distances between points nor angles between lines are preserved. However, straight lines are mapped into straight lines as demonstrated in Figure 4.1.

4.1.2 Parameters of a Stereo System

Intrinsic Parameters

The intrinsic parameters are the set of parameters necessary to characterize the optical, geometric and digital characteristics of a camera. In a stereo setup, both left and right cameras should be separately calibrated for their intrinsic parameters. They link the pixel coordinates of an image point to the corresponding

coordinates in the camera reference frame. For a pinhole camera, we need three sets of intrinsic parameters, specifying, respectively,

1. the perspective projection, for which the only parameter is the focal length, f;
2. the transformation between image coordinates (x, y) and pixel coordinates (u, v);
3. the optical geometric distortion.

We have already addressed the first in Section 4.1.1. To formulate the second relationship, we neglect any geometric distortions and assume that the CCD array is made of a rectangular grid of photosensitive elements. Then the image coordinates can be represented in terms of the pixel coordinates as

$$\begin{aligned} x &= (u - u_o)\alpha_u \\ y &= (v - v_o)\alpha_v \end{aligned}$$

with (u_o, v_o) the pixel coordinates of the principal point O and (α_u, α_v) the horizontal and vertical dimensions of a rectangular pixel (in millimeters) respectively. The above relationship can be expressed in homogeneous coordinates

$$\begin{bmatrix} u \\ v \\ 1 \end{bmatrix} = \begin{bmatrix} \frac{1}{\alpha_u} & 0 & u_o \\ 0 & \frac{1}{\alpha_v} & v_o \\ 0 & 0 & 1 \end{bmatrix} \begin{bmatrix} x \\ y \\ 1 \end{bmatrix} \tag{4.2}$$

Combining (4.1) and (4.2) we get

$$\begin{bmatrix} u \\ v \\ 1 \end{bmatrix} = \begin{bmatrix} \frac{1}{\alpha_u} & 0 & u_o \\ 0 & \frac{1}{\alpha_v} & v_o \\ 0 & 0 & 1 \end{bmatrix} \begin{bmatrix} f & 0 & 0 & 0 \\ 0 & f & 0 & 0 \\ 0 & 0 & 1 & 0 \end{bmatrix} \begin{bmatrix} X \\ Y \\ Z \\ 1 \end{bmatrix}$$

In homogeneous coordinates this can be further simplified to

$$\begin{bmatrix} u \\ v \\ 1 \end{bmatrix} = \begin{bmatrix} \frac{f}{\alpha_u} & 0 & u_o \\ 0 & \frac{f}{\alpha_v} & v_o \\ 0 & 0 & 1 \end{bmatrix} \begin{bmatrix} X \\ Y \\ Z \end{bmatrix}$$

If we assume square pixels (i.e., $\alpha_u = \alpha_v$) and express the focal length in terms of pixels ($f_p = \frac{f}{\alpha_u} = \frac{f}{\alpha_v}$) we obtain

$$\begin{bmatrix} u \\ v \\ 1 \end{bmatrix} = \underbrace{\begin{bmatrix} f_p & 0 & u_o \\ 0 & f_p & v_o \\ 0 & 0 & 1 \end{bmatrix}}_{M_{int}} \begin{bmatrix} X \\ Y \\ Z \end{bmatrix} \tag{4.3}$$

with M_{int} the intrinsic parameter matrix.

The perspective projection model given in (4.3) is a distortion-free camera model and is useful under special circumstances (discussed in Section 4.2.3). However, due to design and assembly imperfections, the perspective projection model does not always hold true and in reality must be replaced by a model that includes geometric distortion. Geometric distortion mainly consists of three types of distortion: radial distortion, decentering distortion, and thin prism distortion [52]. Among them, radial distortion is the most significant and is considered here. Radial distortion causes inward or outward displacement of image points from their

true positions. An important property of radial distortion is that it is null at the image center, and increases with the distance of the point from the image center. Based on this property, we can model the radial distortion as

$$\begin{aligned} x &= x_d(1 + k_1 r^2 + k_2 r^4) \\ y &= y_d(1 + k_1 r^2 + k_2 r^4) \end{aligned}$$

with (x_d, y_d) the coordinates of the distorted points, $k_1$and k_2 additional intrinsic parameters and $r^2 = x_d^2 + y_d^2$. When geometric distortion is taken into consideration, (4.2) above has to be modified

$$\begin{bmatrix} u \\ v \\ 1 \end{bmatrix} = \begin{bmatrix} \frac{1}{\alpha_u} & 0 & u_o \\ 0 & \frac{1}{\alpha_v} & v_o \\ 0 & 0 & 1 \end{bmatrix} \begin{bmatrix} x_d \\ y_d \\ 1 \end{bmatrix}$$

Extrinsic Parameters

In the context of stereo vision, extrinsic parameters are any set of geometric parameters that uniquely identify the rigid transformation between the left and right camera coordinate frames. A typical choice for describing such a transformation is to use

- a 3D translation vector, T, describing the relative positions of the origins of the two camera frames, and
- a (3×3) rotation matrix, R, an orthogonal matrix ($R^T R = RR^T = I$) that brings the corresponding axes of the two frames on to each other (the orthogonality property reduces the number of degrees of freedom of R to three)

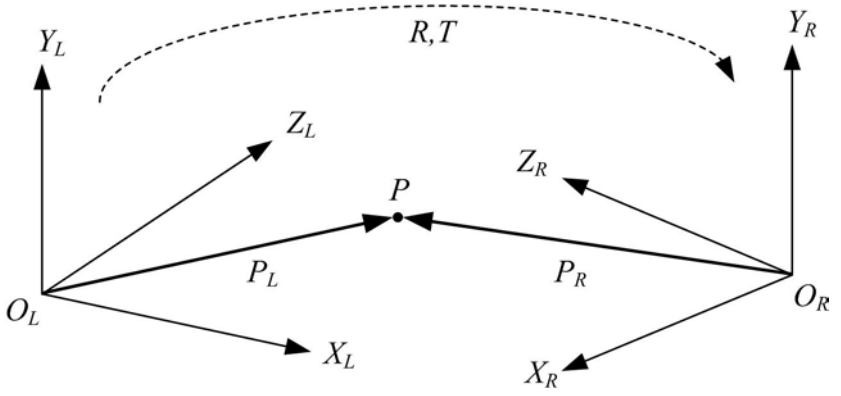


FIGURE 4.2: The transformation between left and right camera frames.

The relationship between the coordinates of a point P in left and right camera frames, P_L and P_R respectively, is

$$P_R = R(P_L - T)$$

This is illustrated in Figure 4.2. For $P_R = [X_R, Y_R, Z_R]^T$ and $P_L = [X_L, Y_L, Z_L]^T$ the above relationship can be expressed in homogeneous coordinates as

$$\begin{bmatrix} X_R \\ Y_R \\ Z_R \\ 1 \end{bmatrix} = \underbrace{\begin{bmatrix} R & -RT \\ 0^T & 1 \end{bmatrix}}_{M_{ext}} \begin{bmatrix} X_L \\ Y_L \\ Z_L \\ 1 \end{bmatrix} \tag{4.4}$$

with M_{ext} the extrinsic parameter matrix.

4.1.3 Epipolar Geometry

When two cameras view a 3D scene from two distinct positions, there are a number of geometric relations between the 3D points and their projections onto the 2D image planes that lead to constraints between the image points. This geometric relation of a stereo setup, known as epipolar geometry, assumes a pinhole camera

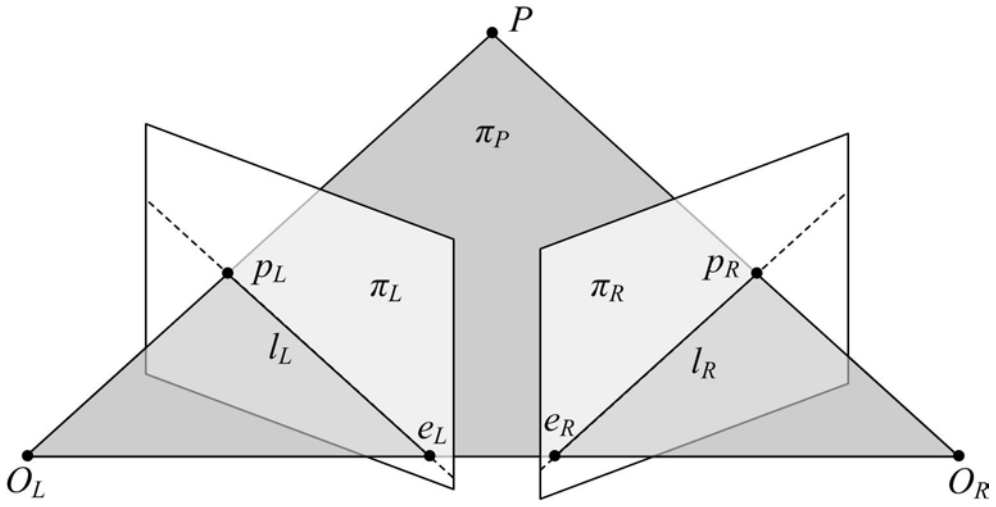


FIGURE 4.3: Epipolar geometry.

model . Epipolar geometry is independent of the scene composition and depends only on the intrinsic and extrinsic parameters.

The notation in Figure 4.3 follow the same convention introduced in Section 4.1.1, with subscripts L and R denoting left and right camera frames respectively. Since the centers of projection of the two cameras are distinct, each of them projects onto a distinct point in the other camera's image plane. These two image points, denoted by e_L and e_R, are called epipoles. In other words the baseline b, that is the line joining O_L and O_R, intersects image planes at respective epipoles. An arbitrary 3D world point P defines a plane with O_L and O_R . The projections of point P on the two image planes, p_L and p_R, also lie on the same plane. This plane is called the epipolar plane (π_P) and its intersection with images planes forms conjugated epipolar lines (l_L and l_R). This geometry discloses the following important facts:

- The epipolar line is the image in one camera of a ray through the optical center and image point in the other camera. Hence, corresponding image points must lie on conjugated epipolar lines (known as the epipolar constraint).

- With the exception of the epipole, only one epipolar line goes through any image point.
- All epipolar lines of one camera intersect at its epipole.

The epipolar constraint is one of the most fundamentally useful pieces of information which can be exploited during stereo correspondence (Section 4.3). Since 3D feature points are constrained to lie along conjugated epipolar lines in each image, knowledge of epipolar geometry reduces the correspondence problem to a 1D search. This constraint is best utilized by a process known as image rectification. However, image rectification generally requires a calibration procedure to be performed beforehand. The following section describes these procedures.

4.2 Calibration and Rectification

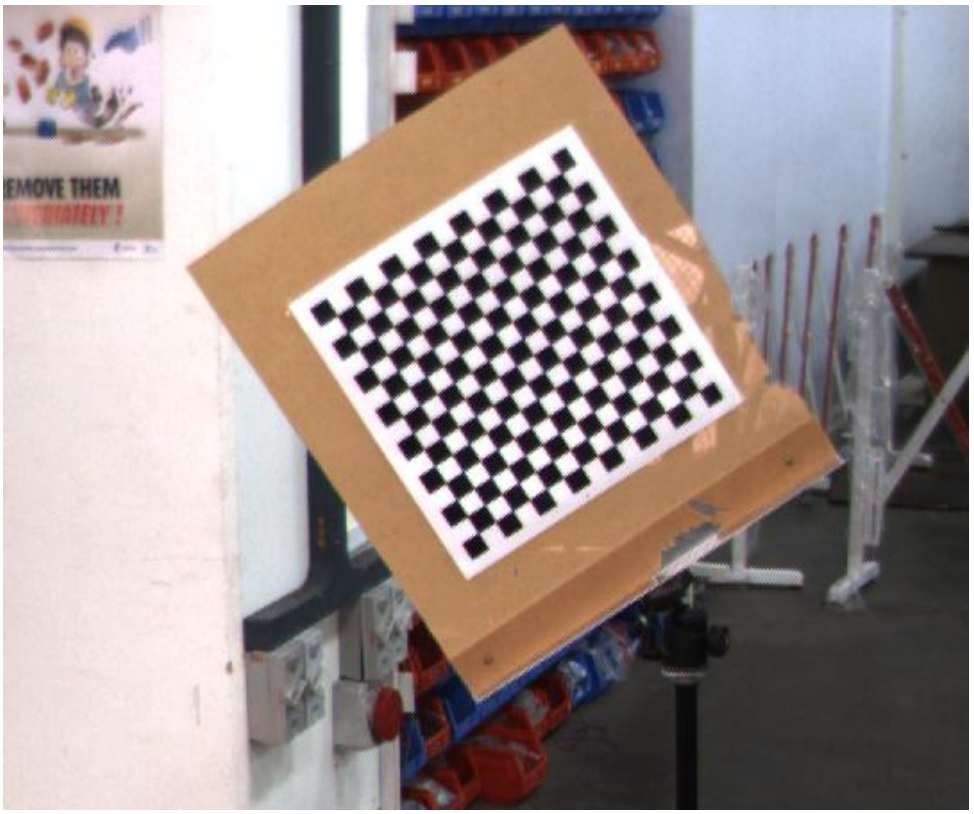
4.2.1 Stereo Camera Calibration

Generally speaking, calibration is the problem of estimating values of unknown parameters in a sensor model in order to determine the exact mapping between sensor input and output. For most computer vision applications, where quantitative information is to be derived from a captured scene, camera calibration is an indispensable task. In the context of stereo vision, the calibration process reveals internal geometric and optical characteristics of each camera (intrinsic parameters) and the relative geometry between the two camera coordinate frames (extrinsic parameters). The parameters associated with this process has already been discussed in Section 4.1.2.

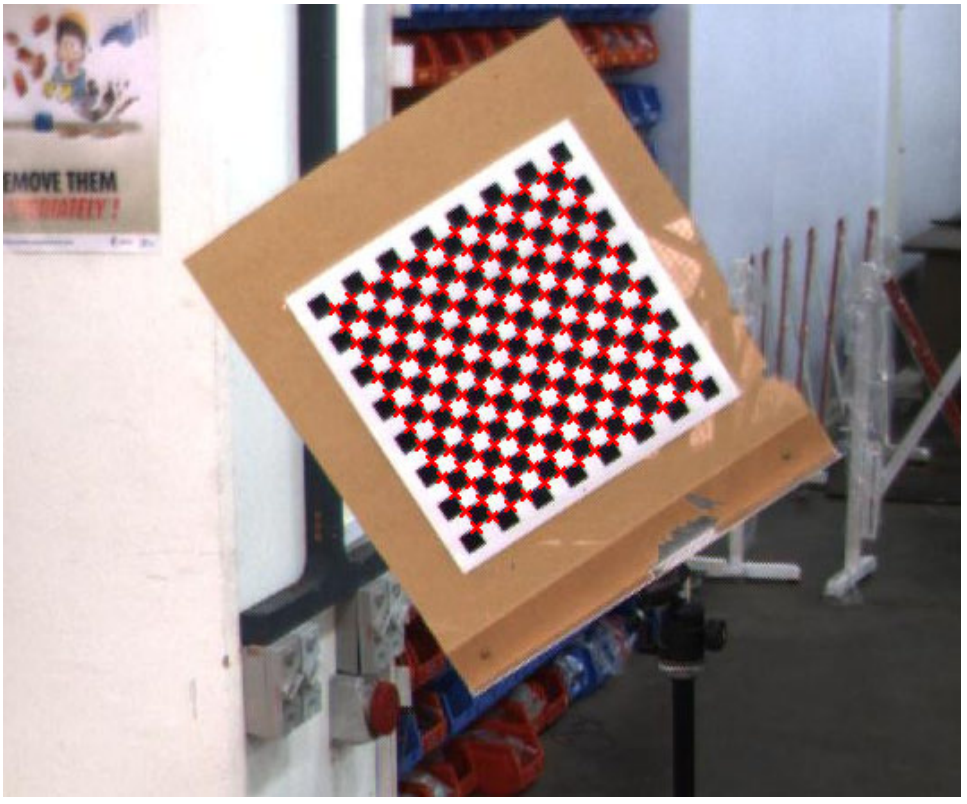
The key idea behind stereo camera calibration is to write a set of equations linking the known coordinates of a set of 3D points and their projections on the left and right image planes. In order to know the coordinates of some 3D points,

calibration methods rely on one or more images of a calibration pattern, that is, a 3D object of known geometry and generating image features that can be located accurately. In most cases, a flat plate with a regular pattern marked on it causing a high contrast between the marks and the background is used. Figure 4.4(a) shows the checkerboard calibration pattern used during the initial test phase of our work. It consists of a black and white grid with known grid size and relative positions. The 3D positions of the vertices of each square, highlighted in Figure 4.4(b), are used as calibration points. As the first step of calibration, multiple images of the calibration pattern are captured by varying its position and orientation (Figure 4.5). After that, the calibration process proceeds to find the projection of detected calibration points in the images and then solves for the camera parameters by minimizing the re-projection error of calibration points. This results in two sets of intrinsic parameters for the two cameras and multiple sets of transformation matrices, one for each calibration grid location and each camera. These transformation matrices are collectively used in the next step to recover the extrinsic parameters of the stereo setup by minimizing the rectification error.

Camera calibration has been studied intensively in the past few decades and continues to be an area of active research within the computer vision community. Two of the most popular techniques for camera calibration are those of Tsai [53] and Zhang [54]. Tsai's calibration model assumes the knowledge of some camera parameters to reduce the initial guess of the estimation. It requires more than eight calibration points per image and solves the calibration problem with a set of linear equations based on the radial alignment constraint. A second order radial distortion model is used while no decentering distortion terms are considered. The two-step method can cope with either a single image or multiple images of a 3D or planar calibration grid, but grid point coordinates must be known. Zhang's calibration method requires a planar checkerboard grid to be placed at more than



(a) Calibration grid.



(b) Calibration points.

FIGURE 4.4: Calibration grid used in the initial experiments.

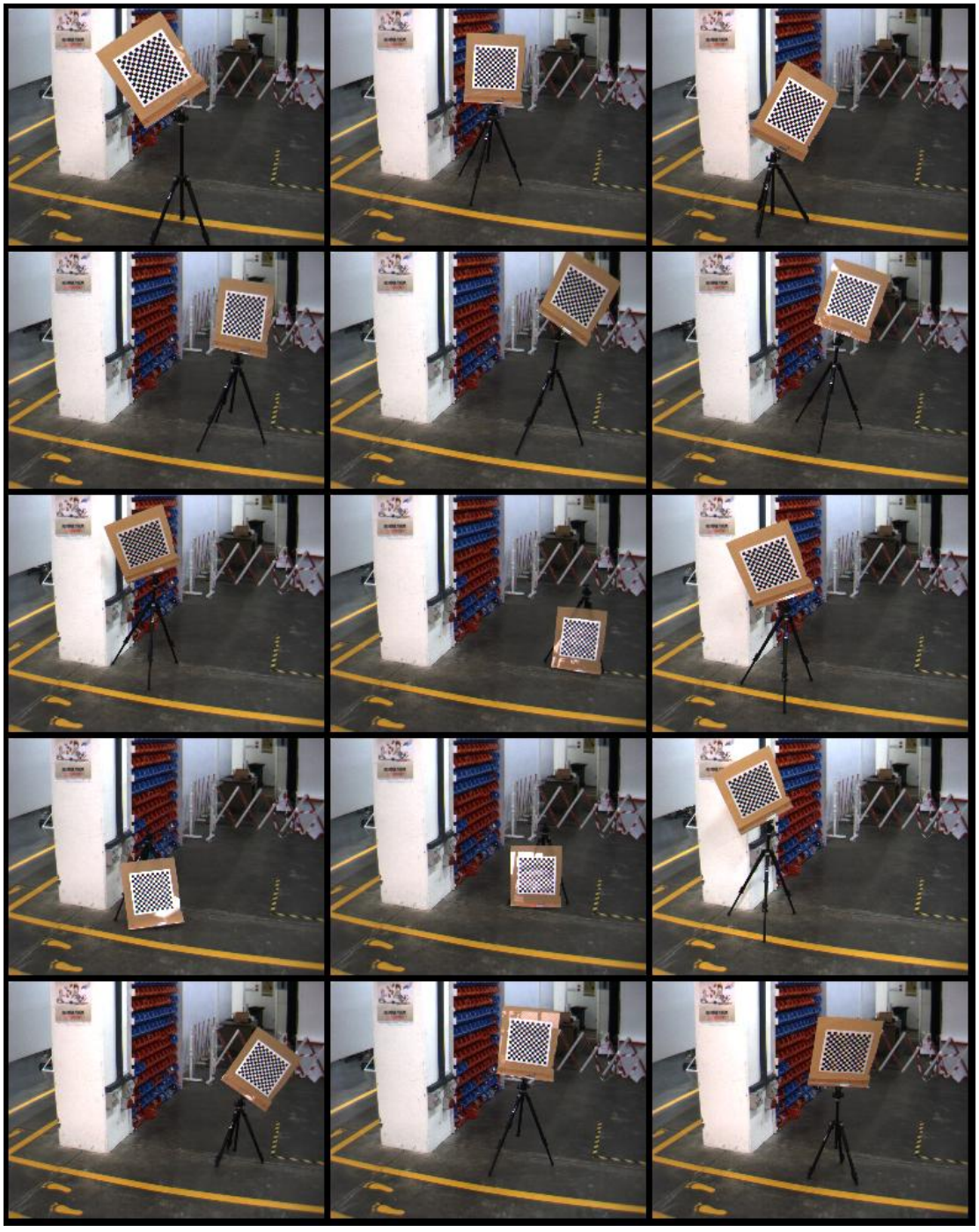


FIGURE 4.5: A set of calibration images.

two different orientations in front of the camera. This algorithm uses the extracted calibration points of the checkerboard pattern to compute a projective transformation between the image points of different images, up to a scale factor. Afterwards, the camera intrinsic and extrinsic parameters are recovered using a closed-form solution, while the third and fifth order radial distortion terms are recovered within a linear least squares solution. A final nonlinear minimization of the re-projection error, solved using a Levenberg-Marquardt method, refines all the recovered parameters. However, apart from the two methods discussed above, other techniques may be used for camera calibration. A comprehensive description of all such methods and their underlying mathematics is beyond the scope of our work. For further reading on this topic we recommend [55].

During the initial experimental stage, to find a suitable baseline for the system, we used our own stereo setup comprising two monocular cameras. This setup was calibrated using a stereo calibration technique developed by Jean-Yves Bouguet at the California Institute of Technology. An open-source Matlab implementation of this method can be found in [56]. However, as mentioned in Section 3.1, the Bumblebee stereo camera used in our final system comes along with a set of precisely calibrated parameters, eliminating the need of a manual calibration.

4.2.2 Stereo Rectification

Given a pair of stereo images, stereo rectification is the process of transforming each image plane such that pairs of conjugate epipolar lines become collinear and parallel to one of the image axes, usually the horizontal one. This process is illustrated in Figure 4.6, which also demonstrates how the points of the rectified images are determined from the points of the original images and their corresponding projection rays. Though the knowledge of stereo calibration parameters is not essential for this task, its availability simplifies the rectification process to a great

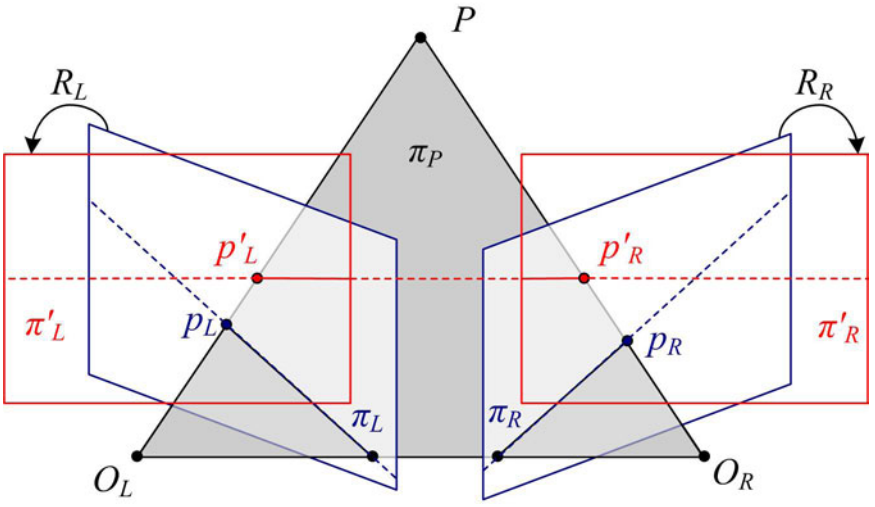


FIGURE 4.6: Rectification of a stereo pair.

extent. In what follows, stereo rectification refers to calibrated rectification; since an accurate set of calibration parameters is available to us, uncalibrated stereo rectification will not be discussed in this work.

The first step of the stereo rectification algorithm is to determine R_L and R_R, the rotation matrices for left and right camera frames respectively. It comprises the following steps:

1. Construct a triple of mutually orthogonal unit vectors e_1, e_2, e_3 from the translation vector T:
   - $e_1 = \frac{T}{\|T\|}$
   - $e_2 = \frac{1}{\sqrt{T_x^2+T_y^2}} \begin{bmatrix} -T_y & T_x & 0 \end{bmatrix}^T$
   - $e_3 = e_1 \times e_2$
2. Define the orthogonal rectification matrix $R_{rect} = \begin{bmatrix} e_1^T \\ e_2^T \\ e_3^T \end{bmatrix}$
3. Set $R_L = R_{rect}$; this transformation takes the epipole of the left camera to infinity along the horizontal axis. In other words the epipolar lines become parallel to the horizontal axis.

4. Set $R_R = R\,R_{rect}$.

In general, the integer coordinates of the rectified and original images will not coincide. Therefore, to avoid round-off errors, rectification is usually performed as an inverse transformation; that is, starting from the rectified image plane, pixels are back-projected to the original image plane. To enable this operation, we need to compute suitable values for the intrinsic parameters of the rectified camera configuration from that of the original configuration:

- Focal lengths are selected such that the rectified images will retain as much information contained in their original counterparts. For simplicity, the focal lengths of both cameras are set to the minimum of the two focal lengths.
- The principal points are chosen to maximize the visible area in the rectified images. For simplicity, principal points for both cameras are set to the average of the two principal points.

Using the above parameters, rectified image pixels are converted to rectified camera coordinates, and subsequently transformed to original camera coordinates using the inverse of R_L and R_R (note that since these rotation matrices are orthogonal, the transpose operation is equivalent to the inverse). After that, geometric distortion is applied and the resulting image coordinates are reconverted into image pixels using original intrinsic parameters. The corresponding gray-scale or color values are computed as a bi-linear interpolation of the original pixel values. In our system, the stereo rectification is performed by the Triclops SDK as previously mentioned in Section 3.2.

4.2.3 Simple Stereo Configuration

From the discussion in the previous section, we can infer that a pair of stereo rectified images is equivalent to a pair of images captured using two coplanar,

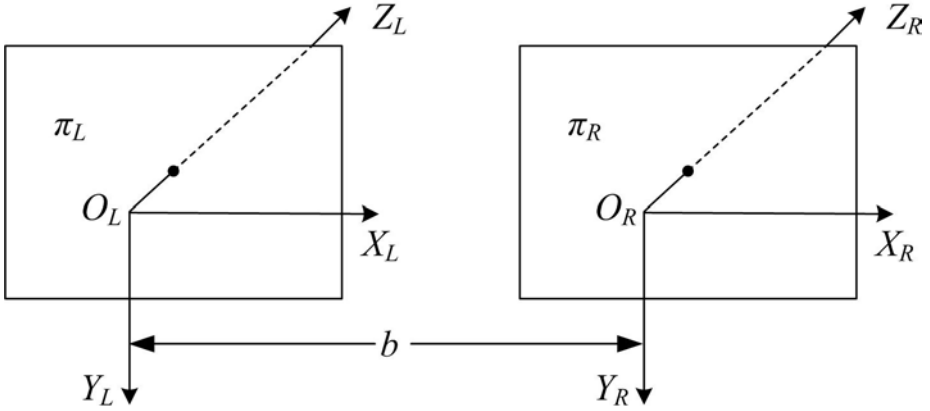


FIGURE 4.7: Simple stereo configuration.

distortion-free cameras with identical intrinsic parameters. This hypothetical configuration, known as the *simple stereo configuration* (Figure 4.7), follows the imaging model given by (4.3). Therefore we have

$$u = \frac{f_p X + u_0 Z}{Z} \tag{4.5}$$

$$v = \frac{f_p Y + v_0 Z}{Z} \tag{4.6}$$

$$u_R = \frac{f_p X_R + u_0 Z_R}{Z_R} \qquad u_L = \frac{f_p X_L + u_0 Z_L}{Z_L} \tag{4.7}$$

The extrinsic parameters of the rectified setup are

$$R = \begin{bmatrix} 1 & 0 & 0 \\ 0 & 1 & 0 \\ 0 & 0 & 1 \end{bmatrix} \qquad T = \begin{bmatrix} 0 \\ 0 \\ b \end{bmatrix}$$

Therefore from (4.4) we have

$$X_R = X_L - b; \qquad Y_R = Y_L; \qquad Z_R = Z_L; \tag{4.8}$$

By substituting (4.8) into (4.7), we may express both u_R and u_L in terms of right camera coordinates

$$u_R = \frac{f_p X_R + u_0 Z_R}{Z_R} \qquad u_L = \frac{f_p (X_R + b) + u_0 Z_R}{Z_R} \tag{4.9}$$

From (4.9) we obtain the stereo disparity, d [1]

$$d = u_L - u_R = \frac{f_p b}{Z_R} \tag{4.10}$$

By treating the right camera coordinate frame as the reference frame, we may omit the subscript indices. By re-arranging (4.10) we obtain

$$Z = \frac{f_p b}{d} \tag{4.11}$$

We can then deduce from (4.5) and (4.11):

$$X = \frac{Z(u - u_0)}{f_p} = \frac{b(u - u_0)}{d} \tag{4.12}$$

Also, from (4.6) and (4.11) we obtain

$$Y = \frac{Z(v - v_0)}{f_p} = \frac{b(v - v_0)}{d} \tag{4.13}$$

Under simple stereo geometry, (4.11), (4.12) and (4.13) govern the unique mapping between the image pixel coordinates and 3D scene points expressed with respect to the camera reference frame.

[1] Disparity is the relative displacement on the two image planes caused by the different perspectives of a scene point. Usually there is a vertical and a horizontal component, but for rectified images only a horizontal disparity exists.

4.3 Stereo Correspondence

The term 'Stereo Correspondence' has been mentioned few times during our preceding discussion. In this section, we give a formal definition of this concept, and elaborate on the components of a dense stereo correspondence algorithm.

Given two different perspectives of the same scene, stereo correspondence is the problem of identifying matching pixel point pairs, across the two views, that are being projected along lines of sight of the same 3D scene element. The automatic establishment of such pixel correspondences of images has traditionally been, and continues to be, one of the most heavily investigated problems in computer vision. The strong interest in this has been spurred by its practical importance, especially in the domain of 3D scene reconstruction. However, due to the ill-posed nature of the correspondence problem, it is virtually impossible to identify correct matches across two images without incorporating additional constraints. In Section 4.1.3, we discussed one such constraint, the epipolar constraint. Even though it helps reduce the search space from 2D to 1D, it is necessary to make use of other assumptions or constraints to deal with the remaining ambiguity. Below is a list of other commonly used constraints.

1. Similarity: the matching pixels must have similar intensity values or in other words the difference should be below a specified threshold (fails under high noise or large distortions).

2. Uniqueness: a given pixel in one image can correspond to no more than one pixel in the other image (fails if transparent objects are present in the scene).

3. Continuity: the disparity of the matches should vary smoothly over the image (fails at depth discontinuities).

4. Ordering: if pixels $\{p_L, p'_L\}$ correspond with pixels $\{p_R, p'_R\}$ on the left and right images respectively, and if p_L is to the left of p'_L, then p_R should also

be to the left of p'_R and vice versa. That is, the ordering of correspondences is preserved across images (fails at forbidden zones).

In contrast to the above, the epipolar constraint has nearly zero probability of failure. As discussed in Section 4.2.2, the rectification process further simplifies the epipolar constraint by bringing corresponding points to a horizontal configuration. In what follows, we assume knowledge of the camera calibration parameters and that all stereo image pairs to have been rectified.

4.3.1 Image Enhancement

In practice, the implementation of the similarity constraint at pixel level leads to unreliable results due to perspective distortions and dissimilar camera parameters. The common practice is to compare a local neighborhood around pixels of interest. Whether to use the color, intensity, high frequency content, non-parametric statistics, or to transform the neighborhood to a feature vector, is determined by the requirements of the system in hand. Using color or intensity values require no additional processing, but generally produces poor disparity maps. In this section we consider three image enhancement methods, that can potentially improve the stereo correspondence accuracy, and at the same time amenable to real time implementation.

Laplacian of Gaussian (LoG) operator

The LoG operator is a 2D isotropic measure of the second spatial derivative of an image [57]. It highlights image regions of rapid intensity change and is therefore often used for feature enhancement in stereo correspondence. The LoG operator is an extension of the Laplacian derivative ($\triangle$) , which in its original form is sensitive to point discontinuities caused by noise. Therefore, prior to the application of $\triangle$ on an image $I(u, v)$, it is filtered by a Gaussian low pass filter $G_\sigma(u, v)$ as given

by

$$\triangle \left[G_\sigma \left(u, v \right) * I \left(u, v \right) \right] = \left[\triangle G_\sigma \left(u, v \right) \right] * I \left(u, v \right) = LoG * I \left(u, v \right) \tag{4.14}$$

Since the convolution operator is associative against linear operations, we may equally apply the Laplacian operator first on the Gaussian smoothing filter, and subsequently convolve the hybrid filter with the image, as shown in (4.14). This hybrid operator is what we term the LoG operator. For an independently and identically distributed bi-variate Gaussian function with zero mean, the LoG operator can be expressed as

$$\text{LoG} = \frac{\partial^2}{\partial u^2} \left(\frac{1}{\sqrt{2\pi\sigma^2}} \exp \left[-\frac{(u^2 + v^2)}{2\sigma^2} \right] \right) + \frac{\partial^2}{\partial v^2} \left(\frac{1}{\sqrt{2\pi\sigma^2}} \exp \left[-\frac{(u^2 + v^2)}{2\sigma^2} \right] \right)$$

$$\text{LoG} = \frac{1}{\sqrt{2\pi\sigma^2}} \left(\frac{u^2 + v^2 - 2\sigma^2}{\sigma^4} \right) \exp \left[-\frac{(u^2 + v^2)}{2\sigma^2} \right] \tag{4.15}$$

Since the input image is represented as a set of discrete pixels, we have to find a discrete convolution kernel of finite size that can approximate (4.15). Ideally the weights should approach zero towards the edge of the kernel even though it never happens in practice for a filter of finite size. A discrete approximation of the LoG operator for a (5×5) kernel is given by

$$\begin{bmatrix} 0 & 0 & 1 & 0 & 0 \\ 0 & 1 & 2 & 1 & 0 \\ 1 & 2 & -16 & 2 & 1 \\ 0 & 1 & 2 & 1 & 0 \\ 0 & 0 & 1 & 0 & 0 \end{bmatrix}$$

The above discrete approximation closely follows the shape of the continuous LoG function shown in Figure 4.8. The mean of all elements in the kernel is forced to

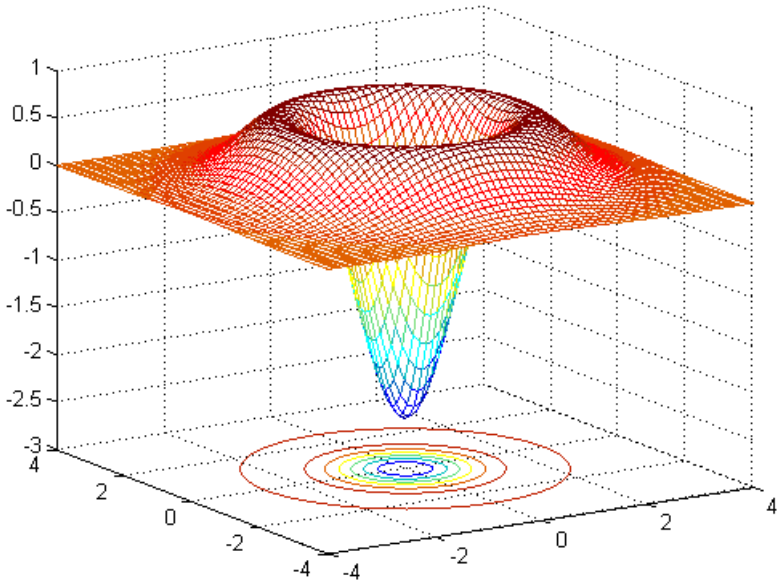


FIGURE 4.8: LoG function.

(a) Original gray scale image. (b) LoG high pass filtered image.

FIGURE 4.9: LoG filtering with a with a 5×5 kernel.

zero (similar to the Laplacian kernel) to ensure that the LoG of a homogeneous region is null at all times. An example of LoG high pass filtering is shown in Figure 4.9.

Rank transform

The rank transform, first used for stereo correspondence by Zabih and Woodfill [58], is a non-parametric measure of the local intensity of an image. It is defined

as the number of pixels in a local region whose intensity is less than the intensity of the center pixel. For an image $I(u, v)$ and a square neighborhood of size $(2n + 1) \times (2n + 1)$ centered around pixel (u_c, v_c), the rank transform R is defined as

$$R(u_c, v_c) = \sum_{i=-n}^{i=n} \sum_{j=-n}^{j=n} U[I(u_c, v_c) - I(u_c - i, v_c - j)]$$

where U is the unit step function. For the above case, the rank transform maps all pixel intensities to integers in the range $[0, (2n + 1)^2 - 1]$. It is important to note that this value does not correspond to any intensity value of the original image. This distinguishes the rank transform from other non-parametric measures such as median filters and mode filters. An illustration and an outcome of the rank transform are shown in Figures 4.10 and 4.11 respectively.

242	255	89
56	214	210
58	42	61

$R(u_c, v_c) = 5$

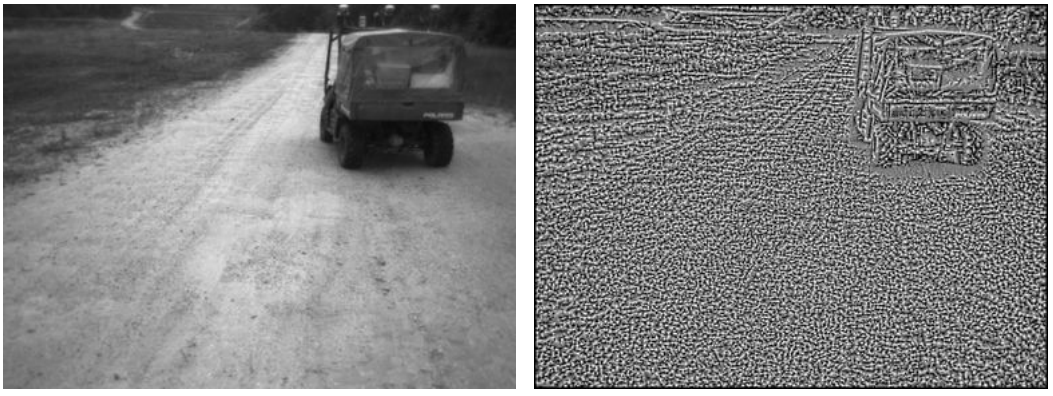
41	43	94
105	43	114
77	42	87

$R(u_c, v_c) = 2$

11	5	5
6	5	8
14	9	7

$R(u_c, v_c) = 0$

FIGURE 4.10: Illustration: rank transform with a 3×3 window.



(a) Original gray-scale image. (b) Rank transformed image.

FIGURE 4.11: Real images: rank transform with a 7×7 window.

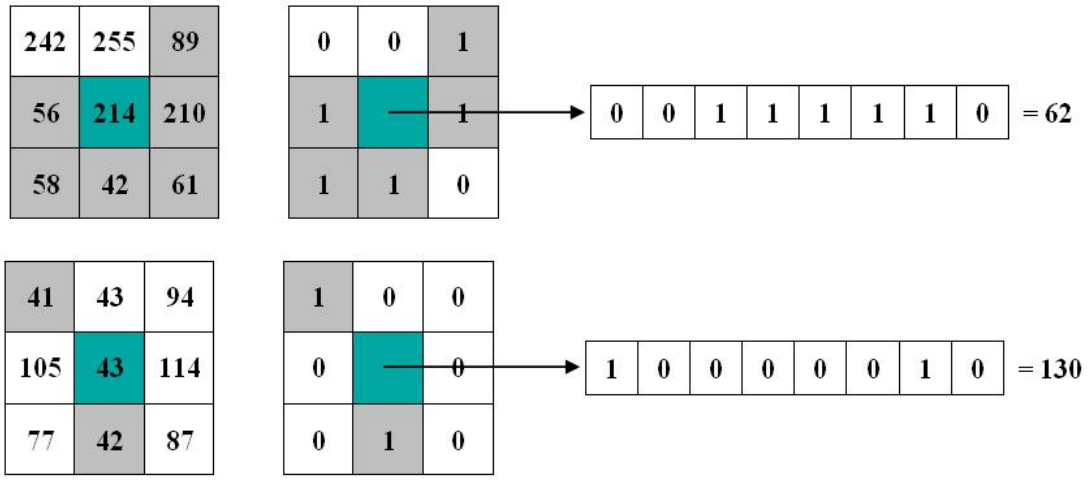


FIGURE 4.12: Illustration: census transform with a 3×3 window.

Census transform

Fundamentally, the census transform is equivalent to the well known texture representation called the local binary pattern (LBP) . However, it was first used in the context of stereo matching in [58]. The census transform encodes a local neighborhood in an image to an ordered bit string by comparing it with the center pixel: pixels that are less than the center pixel are encoded to '1' and otherwise to '0'. For an image $I(u, v)$ and a square neighborhood of size $(2n+1) \times (2n+1)$ centered around pixel (u_c, v_c), the census transform C is obtained by

$$C(u_c, v_c) = \underset{\substack{i=-n \\ \notin (i=j=0)}}{\overset{i=n}{\otimes}} \underset{j=-n}{\overset{j=n}{\otimes}} U[I(u_c, v_c) - I(u_c - i, v_c - j)]$$

where $\otimes$ denotes concatenation. The resulting bit string is stored in the center pixel as a decimal number which has a range $[0, 2^{[(2n+1)^2-1]} - 1]$. This transform is better explained graphically in Figure 4.12 and the achieved texture enhancement is observed in Figure 4.13(b).

4.3.2 Dense Disparity Computation

The input to this process is a pair of stereo rectified and local feature enhanced images I_L and I_R. Disparity computation algorithms can be broadly categorized

(a) Original gray scale image. (b) Census transformed image.

FIGURE 4.13: Real images: census transform with a 3×3 window.

into two classes: feature-based and area-based [59]. Feature-based methods yield sparse correspondence maps in contrast to the dense maps produced by area-based methods. Since we require dense disparity maps as the input to our obstacle detection algorithm, the former category will not be discussed in this writing. A typical area-based stereo matching algorithm finds, for each location in one image, the offset that aligns this location with the best matching location in the other image. For a pair of stereo rectified images, the steps of this process can be summarized as follows:

1. Define a window w_R in the right image with its center at (u, v) .
2. Define a window w_L in the left image that is identical to w_R in size and position.
3. Offset w_L in the positive u direction in unit steps, and compute the matching cost (or correlation) at each pixel.
4. Compute the disparity.

As illustrated in Figure 4.14, the upper bound of the offset or maximum disparity (d_{max}) is determined by the horizontal field of view (FOV) of the two cameras,

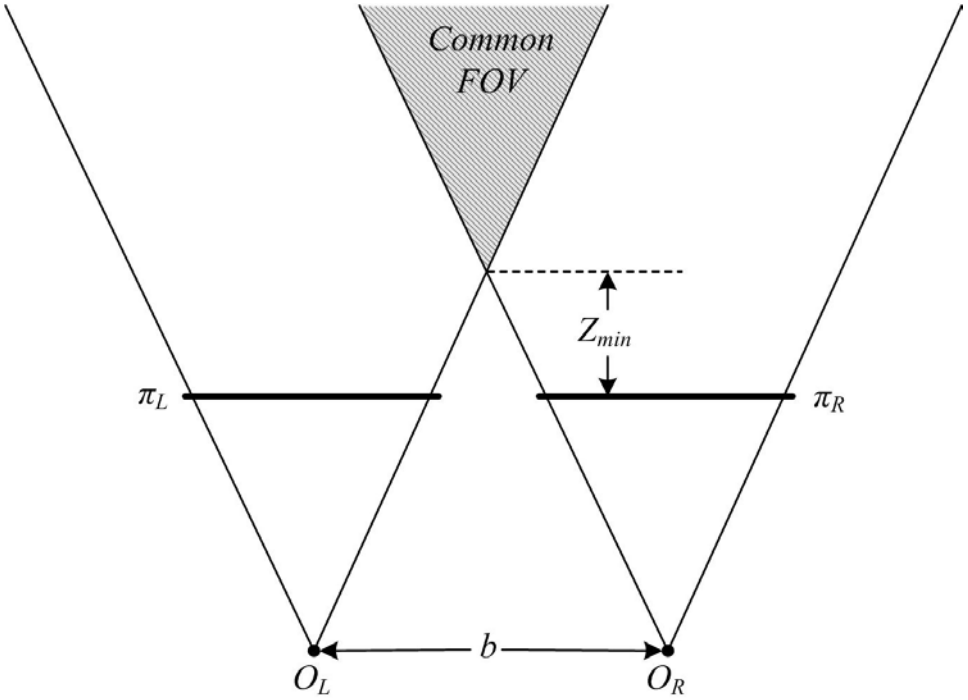


FIGURE 4.14: FOV of a simple stereo configuration.

the baseline width and the image resolution. In this representation, Z_{min} corresponds to the minimum visible distance of both cameras, which is equivalent to d_{max} in disparity space. In our algorithm, the absolute difference between pixel intensities acts as the matching cost. For two overlapping windows the correlation is computed by summing the absolute difference costs within support of the window. Hence this area based correlation method is commonly known as the sum of absolute difference (SAD). For a square window of size $(2n+1) \times (2n+1)$ centered around pixel (u,v), the SAD correlation S is computed as a function of d:

$$S(u,v,d) = \sum_{i=-n}^{i=n} \sum_{j=-n}^{j=n} abs[I_R(u+i, v+j) - I_L(u+i+d, v+j)] \tag{4.16}$$

In SAD, the emphasis is on the matching cost computation and on the cost aggregation steps. Computing the final disparities is trivial; simply choose at each pixel the disparity associated with the minimum SAD, S_{min}:

$$d(u,v) = \arg \ \min[S(u,v,d)]$$

In practice, to perform disparity computation and subsequent disparity refinement in one cycle, a disparity space image (DSI) representation is used. The DSI is a 3D matrix containing SAD correlation values computed at each pixel and each possible offset. The final dense disparity map (Figure 4.15) is formed by the set of indices corresponding to the minimum value along the third dimension of the DSI. This form of disparity computation is usually known as the winner-take-all (WTA) optimization.

The cost aggregation step in (4.16), makes an implicit assumption on smoothness of the support region. In other words, it assumes that all pixels enclosed within a matching window are of equal disparity. Central to this, is the problem of selecting an appropriate window size for SAD correlation. The chosen window size must be large enough to include substantial intensity variation for matching, but small enough to avoid the effects of projective distortion. If the window is too small and does not cover enough intensity variation, it gives a poor disparity estimate, due to low signal-to-noise ratio (SNR). If, on the other hand, the window is too large and covers a region in which the depths of scene points vary substantially, then the position of minimum SAD may not represent correct matching due to different projective distortions in the left and right images. On the other hand, the

(a) Reference image.

(b) Corresponding dense disparity map.

FIGURE 4.15: Dense disparity computation.

WTA optimization fails to enforce a local smoothness condition on the disparity surface. This disparity selection scheme, which disregards the possible geometric correlation between adjacent scene points, might lead to poor disparity estimates under noisy conditions.

To avoid the problem of having to specify a fixed window size, algorithms that can automatically select an appropriate window have been proposed using shiftable windows [60] and adaptive window sizes [61, 62]. We also note that iterative diffusion, an averaging operation that repeatedly adds to each pixel's cost the weighted values of its neighboring pixels' costs, has been used as an alternative method of aggregation [63, 64]. The disparity computation step has been performed by means of global optimization in an energy-minimization framework (e.g., graph cuts method [65] and dynamic programming [66]) and belief propagation [67] to estimate the maximum a posteriori (MAP) inference of disparity. These methods, while being better at reducing uncertainty, handling occlusions and dealing with depth discontinuities, are difficult to implement in real time due to their iterative nature. Even though real time implementations of graph cuts, dynamic programming and belief propagation are available with graphics hardware speedup [68, 69, 70], such acceleration has not been considered in our application. Therefore, we tolerate the errors caused by a fixed size correlation window and WTA optimization. In order to minimize the shortcoming of this approach, we determine an optimum window size for SAD correlation with the aid of simulated ground truth data (discussed in Section 6.2). In addition, we seek to refine the obtained disparity maps by imposing multiple constraints on the correlation profile; this is the focus of the next section.

4.3.3 Elimination of Low-confidence Matches

Spurious mis-matches are an inevitable circumstance faced by any stereo correspondence algorithm. Therefore, most algorithms of this kind are equipped with a supplementary post-processing step to suppress locally anomalous disparities. To implement this, we check for three measures of uncertainty during the process of determining disparity from a SAD correlation function $S(u, v, d)$. For the sake of clarity, we will omit pixel indices (u, v) and denote the correlation function by $S(d)$ in the equations to follow.

1. ***Left-Right consistency check***: If a pixel in the right image that is "matched" to one in the left image, in turn, does not correspond to the same pixel in the right image, then we may safely assume that either one or both disparity estimates are erroneous (Figure 4.16). In other words, when a right image pixel (u, v) has its SAD correlation minimum at index d_0, it is accepted as a valid disparity if and only if, the left image pixel $(u+d_0, v)$ has

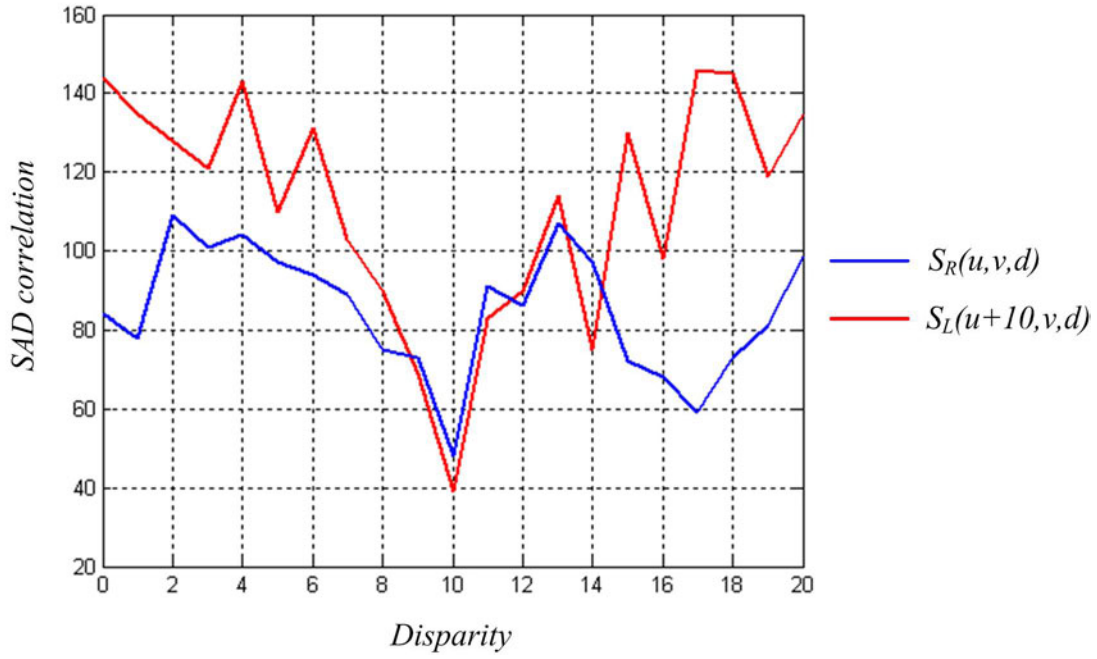


FIGURE 4.16: An example of correlation functions conforming to left-right consistency check.

its correlation minimum at the same index d_0. However, exact enforcement of this cross-checking rule tends to produce holes in the disparity surface close to depth discontinuities. Therefore, during our implementation, this constraint is relaxed as: for a particular pixel, if the left-right disparity error is one or less pixels, then label that disparity estimate as acceptable.

2. ***Entropy***: The entropy of a probability density function (PDF) is a measure of the uncertainty of its information content. To calculate this measure, we first convert $S(d)$ into a PDF by subtracting it from the maximum possible SAD, S_M(which can be calculated for known rank/census and SAD window sizes) and normalize the inverted function.

$$p(d) = \frac{S_M - S(d)}{\sum_{d=0}^{d=d_{max}} [S_M - S(d)]} \tag{4.17}$$

An attractive property of this transformation compared to direct inversion of discrete correlation values is that it preserves the relative differences between correlation values. This is important as our intention is to determine the uncertainty of the existing correlation function without distorting its original content (Figure 4.17). The entropy C_e of a discrete PDF is defined as

$$C_e = -\sum_{d=0}^{d=d_{max}} p(d) \ln[p(d)]$$

Again, to simplify the subsequent thresholding process, we normalize the above expression. Since the maximum entropy corresponds to the maximum uncertainty, normalized entropy $C_{e,N}$ can be obtained by dividing C_e from the entropy of a uniform distribution:

$$C_{e,N} = \frac{-\sum_{d=0}^{d=d_{max}} p(d) \ln[p(d)]}{-\sum_{d=0}^{d=d_{max}} \frac{1}{d_{max}+1} \ln[\frac{1}{d_{max}+1}]} = -\frac{\sum_{d=0}^{d=d_{max}} p(d) \ln[p(d)]}{\ln(d_{max}+1)}$$

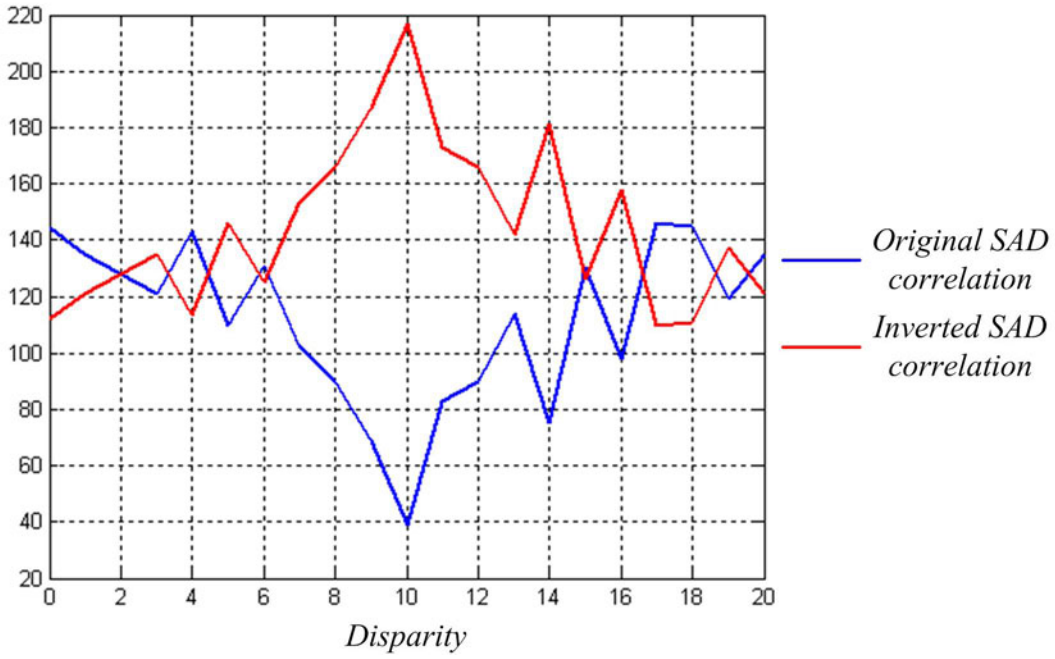


FIGURE 4.17: Conversion of SAD correlation into a PDF.

The normalized entropy lies in a scale from '0' (minimum uncertainty) to '1' (maximum uncertainty). A suitably selected cut-off threshold of the entropy makes the decision regarding the acceptability of a particular disparity.

3. ***Winner margin***: The winner margin C_{wm} is the normalized difference between the minimum, S_{min} and the second minimum, S_{min2} of an SAD correlation function (Figure 4.18). It reflects how clear a minimum exists among the values $S(d)$ for all d. It is calculated by

$$C_{wm} = \frac{S_{min2} - S_{min}}{S_M}$$

Practically the threshold for this measure is chosen well below its ideal value '1'.

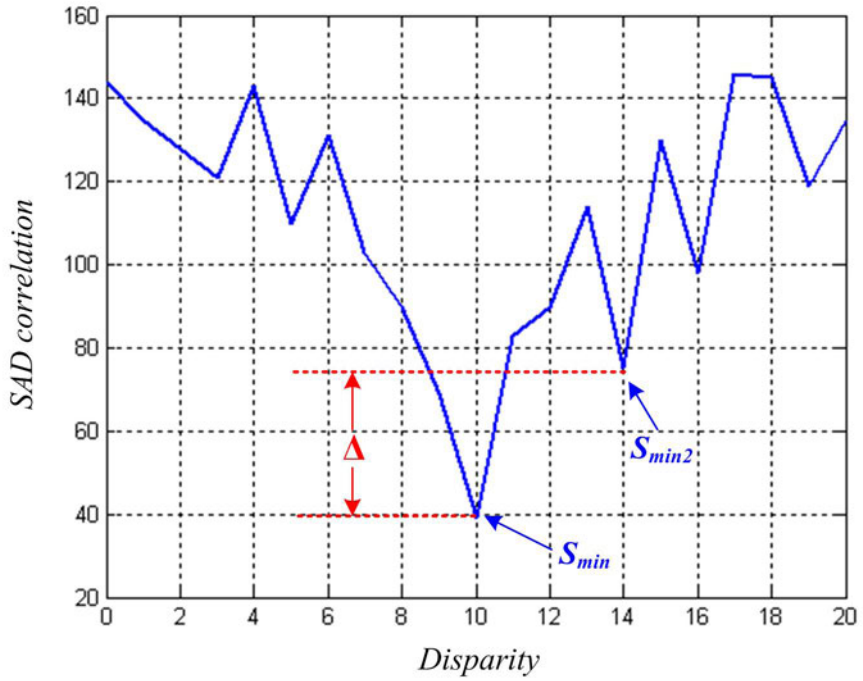


FIGURE 4.18: Winner margin.

4.3.4 Sub-pixel Interpolation

Due to the inverse relationship between stereo disparity and camera coordinates, reconstructed 3D points tend to be sparsely clustered at discrete integer disparities. The reconstruction error caused by this effect increases with the distance, which is especially undesirable for applications of our category, where image information over relatively large distances is utilized. To remedy this situation, algorithms that can establish accurate stereo correspondences at sub-pixel precision have been devised. Such methods can be discussed under three broad categories:

1. Coarse-to-fine search for the true extremum using image pyramids [71]
2. Calculate the correction factor using image intensity gradients [72] or correlation gradient [73]
3. Estimate the true extremum by fitting an analytic function over the indices of the observed extremum and its neighborhood [74, 75].

The first method usually consumes higher memory and computational power, especially when high order interpolation functions are used to reduce the aliasing effect in up-sampled images. Intensity gradient based methods are largely affected by image deformation, while correlation gradient based methods require a high texture content to produce accurate results (in [75] an external projector is used to texture the object being viewed). The last method is computationally least expensive, and hence is a popular choice for real time applications. The associated sub-pixel correction calculation usually consists of multiple additions and a single division, and can be coupled to the existing correlation extremum search function. Two standard functions that are being used for this method are parabolic curves and Gaussian functions. While parabolic curves yield a strong fractional displacement towards integer values, which is known as the *pixel locking effect* [76], Gaussian fitting is able to alleviate this problem [75]. In our work, we verify this claim before choosing one function in favor of the other. Also it is important to note that the extremum of a Gaussian function is a maximum while the extremum of a parabolic curve is a minimum. The extremum of the SAD correlation we use is a minimum, hence it has to be inverted in a suitable manner before fitting to a Gaussian function. Neglecting these properties of fitting functions will lead to meaningless results.

Let's denote an arbitrary correlation function (with a maximum or minimum extremum) with θ. A parabola in $\{\theta, d\}$ space is given by

$$\theta = ad^2 + bd + c \tag{4.18}$$

where b and c are arbitrary coefficients with $a \neq 0$. Differentiating with respect to d we get

$$\frac{d(\theta)}{d(d)} = 2ad + b$$

At the true minimum of the parabola

$$\frac{d(\theta)}{d(d)} = 0 \quad \Longrightarrow \quad d_{min} = \frac{-b}{2a} \tag{4.19}$$

Given the knowledge of three point coordinates on the parabola we may solve for the three coefficients a, b and c. With reference to the diagram shown in Figure 4.19, we substitute point coordinates $(d_0 - 1,\ \theta_{-1})$, $(d_0,\ \theta_0)$ and $(d_0 + 1,\ \theta_{+1})$ to (4.18) and use (4.19) to obtain

$$d_{\theta_{min}} = d_0 + \left[\frac{\theta_{-1} - \theta_{+1}}{2\theta_{-1} - 4\theta_0 + 2\theta_{+1}}\right] \tag{4.20}$$

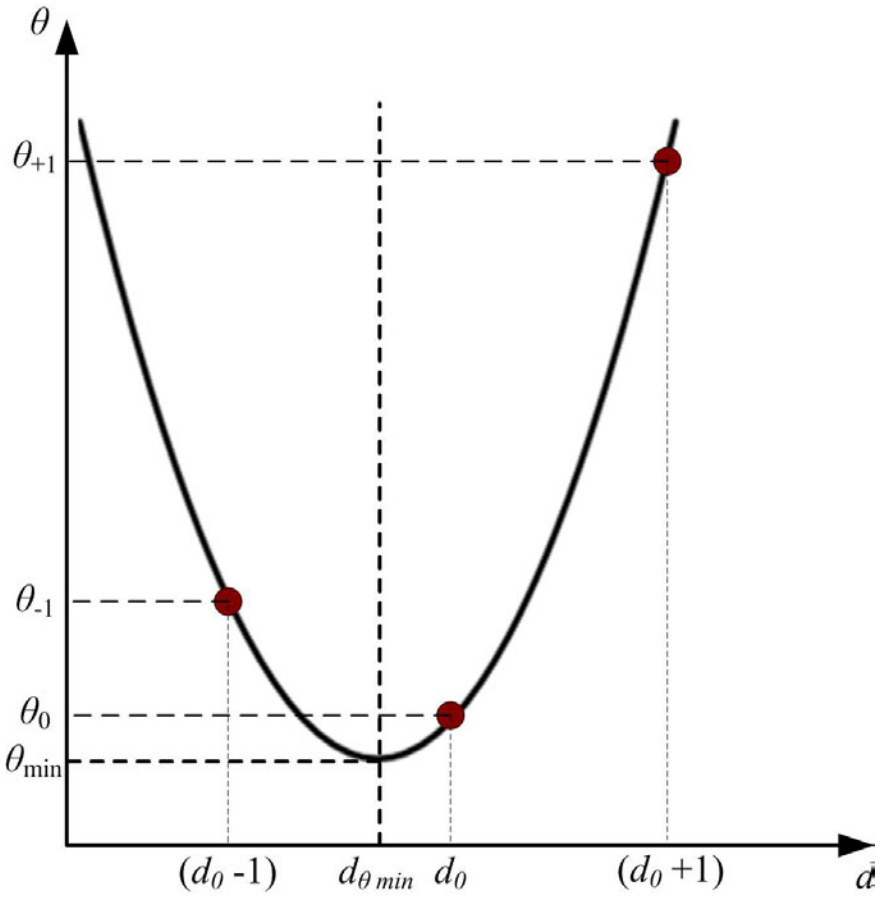


FIGURE 4.19: Parabola fitting for sub-pixel interpolation.

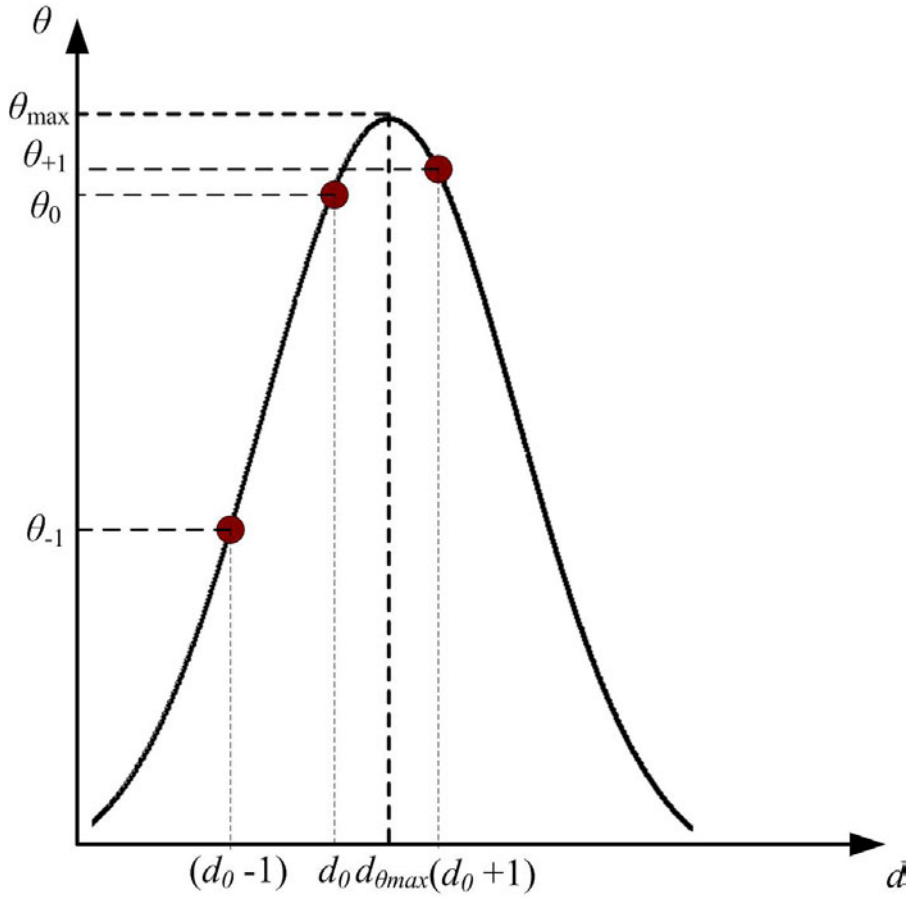


FIGURE 4.20: Gaussian fitting for sub-pixel interpolation.

We now consider a Gaussian function in $\{\theta, d\}$ space:

$$\theta = \exp(ad^2 + bd + c) \tag{4.21}$$

where b and c are arbitrary coefficients with $a \neq 0$. Differentiating with respect to d we get

$$\frac{d(\theta)}{d(d)} = (2ad + b)\exp(ad^2 + bd + c)$$

At the true maximum of the Gaussian function

$$\frac{d(\theta)}{d(d)} = 0 \quad \Longrightarrow \quad d = \frac{-b}{2a} \quad \because \exp(ad^2 + bd + c) \neq 0 \tag{4.22}$$

With reference to the diagram shown in Figure 4.20, we substitute point coordinates $(d_0 - 1,\ \theta_{-1})$, $(d_0,\ \theta_0)$ and $(d_0 + 1,\ \theta_{+1})$ to (4.21) and plug the calculated

coefficients into (4.22) to obtain

$$d_{\theta_{max}} = d_0 + \left[\frac{\ln(\theta_{-1}) - \ln(\theta_{+1})}{2\ln(\theta_{-1}) - 4\ln(\theta_0) + 2\ln(\theta_{+1})} \right] \tag{4.23}$$

For the Gaussian fitting, we use the inverted version of the SAD correlation function given in (4.17).

4.4 Stereo Reconstruction

As discussed earlier, a point P projected to the pair of corresponding points p_L and p_R lies at the intersection of the rays from O_L through p_L and from O_R through p_R, respectively. When both intrinsic and extrinsic camera parameters are given, these rays and their intersection in 3D space can be found. However, in practice, due to imperfections in the calibration and stereo correspondence processes, computed rays will not actually intersect in space as shown in Figure 4.21. Therefore, the intersection is approximated by the point of minimum distance from both rays P'. This reconstruction process is known as triangulation in stereo vision literature. Nevertheless, for the simple stereo configuration we consider in our work, there

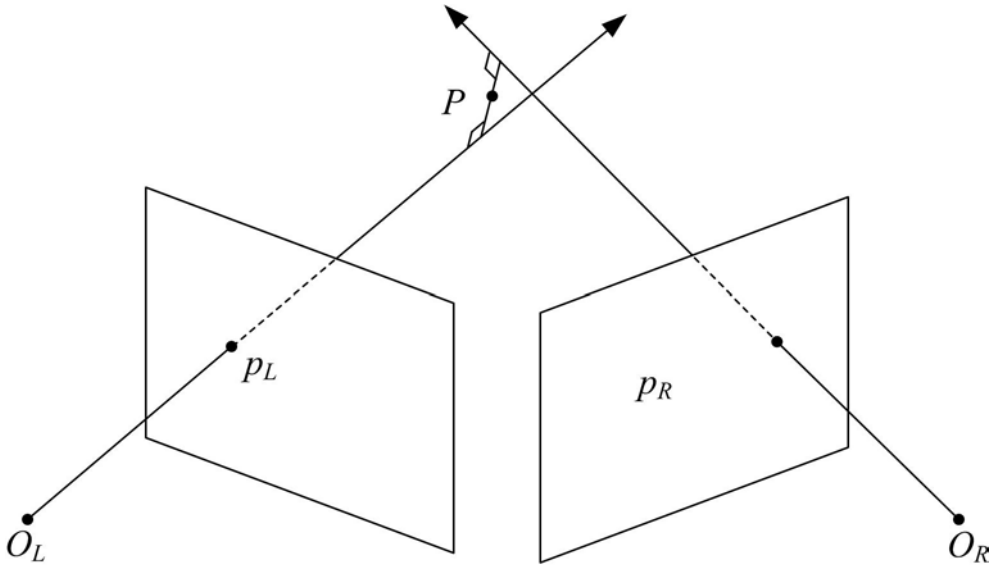


FIGURE 4.21: Stereo triangulation.

exists a much simpler solution in which equations (4.11 - 4.13) can be used to unambiguously solve for the 3D coordinates of a given image point.

So far, in this chapter, we have covered the theoretical and practical aspects of the stereo vision sub-system of our autonomous navigation framework. The camera calibration parameters and refined dense disparity maps produced at this stage are the inputs to the subsequent obstacle detection process. A comprehensive description of this process is given in the next chapter.

Chapter 5

Obstacle Detection

The size and position of the obstacles in 3D space is an essential piece of information for an autonomous vehicle to make correct decisions while maneuvering in complex environments. In this chapter, we propose a computationally inexpensive solution for obstacle detection using dense stereo disparity. The method we propose is specifically customized to produce accurate results for the kind of rural terrains we consider in our work. We begin by defining some of the terms that will be frequently encountered in the rest of the thesis.

- Ground plane: a ground surface that is geometrically smooth and continuous.
- Planar ground: a ground plane that is flat in a geometric sense.
- Vehicle-to-ground clearance: the clearance between the lowest part of the vehicle and the ground when all four wheels are in contact with a planar ground.
- Obstacle: an object that is protruded or depressed with reference to the ground plane to an extent greater than the vehicle-to-ground clearance.
- Traversability: the property of having a lower probability of obstacle occurrence.

- Ground disparity model/map: the disparity map of a ground plane that has approximately equal traversability everywhere.
- Ground pixel: a pixel that is projected from the ground plane.
- Disparity space: $\{u, v, d\}$ coordinate frame.
- Lateral ground profile: the disparity variation of a ground disparity model along the u-axis
- Longitudinal ground profile: the disparity variation of a ground disparity model along the v-axis

5.1 Ground Plane Obstacle Detection

The *ground theory of space perception* (Gibson 1950) states that the foundational surface for terrestrial animals like humans is the ground plane [77]. It also claims that the spatial character of the visual world is given not by the objects in it, but by the ground and the horizon. On a similar note, during locomotion or while steering a vehicle, humans rely on ground signatures to determine a path free of obstructions. Therefore, the notion of ground plane is an inseparable component of any traversability evaluation algorithm. While some methods explicitly model the ground plane geometry, the rest define traversability rules with implicit relations to the ground plane. The former category is what we termed as GPOD, in Section 2.2. Our solution for terrain obstacle detection is derived by analyzing the ground plane modeling component of two such methods, planar ground approximation[1] and the v-disparity method.

[1]While planar approximation is not ideally suited for rural terrain modeling, it can still be helpful in providing useful insights into the overall ground plane modeling problem.

5.1.1 Planar Ground Approximation

Under planar ground approximation, the ground plane can be represented in the camera coordinate frame by

$$a_X X + a_Y Y + a_Z Z + a_0 = 0 \tag{5.1}$$

By substituting (4.11), (4.12) and (4.13) to (5.1) we obtain

$$a_X \frac{b(u-u_0)}{d} + a_Y \frac{b(v-v_0)}{d} + a_Z \frac{f_p b}{d} + a_0 = 0$$

which can be further simplified to

$$a_u u + a_v v + a_d d + \tilde{a_0} = 0 \tag{5.2}$$

The equation above indicates that the geometry of a planar ground is preserved during the projection from 3D space to disparity space. Therefore, as an alternative to estimating planar parameters in 3D metric space, an equivalent operation can be performed in disparity space. It is also important to note that (5.2) can be decomposed into a linear longitudinal ground profile and a fixed lateral ground profile. In order to cope with outliers (i.e., non-ground points), robust regression techniques such as *random sample consensus* (RANSAC) [28] or *iteratively re-weighted least squares* (IRLS) [27] have been used for this task (for more information on robust regression techniques please refer to Appendix B).

5.1.2 The v-disparity Method

The v-disparity method [30], originally designed to model non-flat urban roads, has been implemented in a number of vehicle navigation systems. It is based on the construction and subsequent processing of the v-disparity image, which provides a

robust representation of the geometric content of the ground plane. Essentially, the v-disparity image is a 2D histogram in which the abscissa represents the disparity d, the ordinate represents the image row index v, and the intensity of each pixel represents the number of pixels in the disparity map with respective v and d. In other words, each row in the v-disparity image contains a disparity histogram of the corresponding row. In [30], the authors propose this model for a ground plane that can be approximated by a sequence of oblique planes of the form

$$a_Y Y + a_Z Z + a_0 = 0 \tag{5.3}$$

The equation (5.3) suggests that the ground geometry is independent of X. In turn it implies that the ground plane is parallel to the stereo baseline, since the X axis is collinear with the baseline. By substituting (4.11) and (4.13) to (5.3) we have

$$a_Y \frac{b(v - v_0)}{d} + a_Z \frac{\alpha b}{d} + a_0 = 0$$

which can be further simplified to

$$a_v v + a_d d + \tilde{a_0} = 0 \tag{5.4}$$

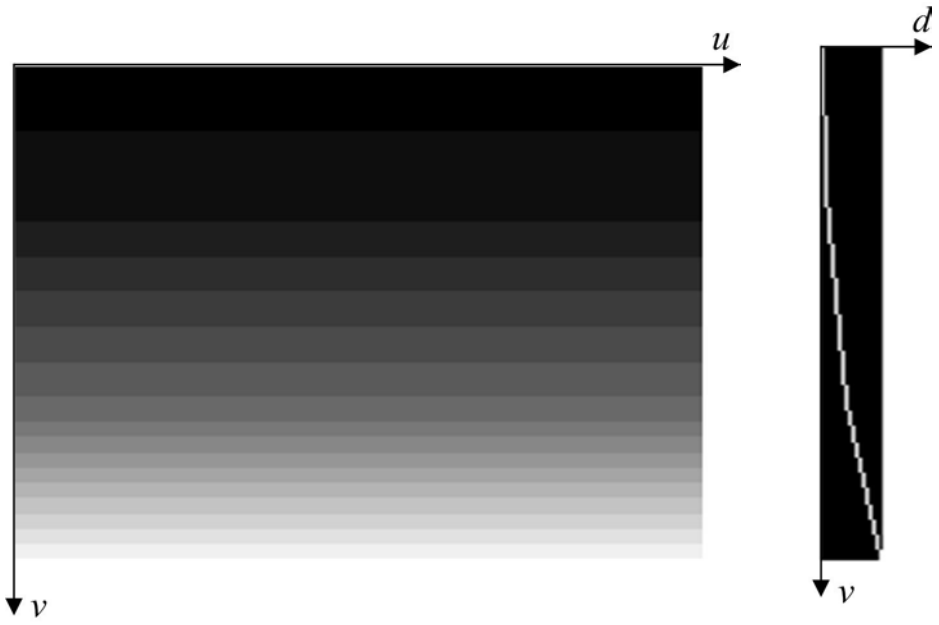
We make two intuitive observations in the planar equation (5.4):

1. Any given row in the ground disparity map will be of uniform disparity (i.e., a lateral ground profile does not exist). This implies that the disparity histogram for a particular v will peak at the corresponding ground disparity bin.

2. Equation (5.4) represents a straight line in $\{v, d\}$ coordinate system.

The first point explains the rationale behind the v-disparity method; when the ground disparity is independent of u, histogramming parallel to the u-axis reduces

the dimensionality of the ground disparity map without any loss of information. Furthermore, the disparity histogram peaks for each row will collectively form a high intensity curve on the v-disparity image (Figure 5.1). This curve, commonly referred to as the *ground correlation line*, can be modeled more accurately than a 3D surface. The second point above reveals shape information of the *ground correlation line*; if the plane governed by (5.3) projects a line on the v-disparity image, a series of such planes result in a piecewise linear curve. A robust line fitting method such as the Hough transform is used to approximate the longitudinal ground profile with a piecewise linear curve.



(a) A ground disparity map governed by (5.4).

(b) Corresponding v-disparity image.

FIGURE 5.1: The v-disparity image generation.

5.2 Vehicle Pose Variation

5.2.1 Effect of Vehicle Pose: Mathematical Analysis

Our aim here is to assess the effect of vehicle pose variation on the two ground plane modeling methods discussed thus far. For both cases we will assume that the ground geometry behaves according to their original assumptions under stationary conditions. If the camera coordinate frame undergoes an arbitrary rotation from $\{X, Y, Z\}$ coordinate frame to $\{X', Y', Z'\}$ during vehicle motion, the resulting transformation is given by

$$\begin{bmatrix} X \\ Y \\ Z \end{bmatrix} = \begin{bmatrix} r_{11} & r_{12} & r_{13} \\ r_{21} & r_{22} & r_{23} \\ r_{31} & r_{32} & r_{33} \end{bmatrix} \begin{bmatrix} X' \\ Y' \\ Z' \end{bmatrix} \tag{5.5}$$

By substituting (5.5) into (5.1) we have

$$(a_X r_{11+} a_Y r_{21} + a_Z r_{31})X' + (a_X r_{12+} a_Y r_{22} + a_Z r_{32})Y' + (a_X r_{13} + a_Y r_{23} + a_Z r_{33})Z' + a_0 = 0 \tag{5.6}$$

which essentially follows the same model as (5.1) for any combination of r_{ij}. Therefore we may conclude that irrespective of the type of pose change (i.e., whether it is rolling, pitching or yawing), the planar ground approximation remains unaffected.

On the other hand, when (5.5) is plugged into (5.3), we have

$$(a_Y r_{21} + a_Z r_{31})X' + (a_Y r_{22} + a_Z r_{32})Y' + (a_Y r_{23} + a_Z r_{33})Z' + a_0 = 0 \tag{5.7}$$

Since a_Y and a_Z are non-zero in general, for (5.7) to be independent of X', both r_{21} and r_{31} should be simultaneously equal to zero. However, this condition is satisfied if and only if the rotation of the camera rig occurs around the X axis

(i.e., if there is only pitching). For any other combination of rolling and yawing, (5.7) will transform to an equation of the form of (5.1). The introduction of an X component (or u component in disparity space) to the piecewise planar equation violates the fundamental assumption made by the v-disparity algorithm. Under these circumstances, the dimensionality reduction of the v-disparity image averages out the lateral disparity variation in an irretrievable manner, eventually leading to an erroneous ground disparity model.

5.2.2 Empirical Evidence

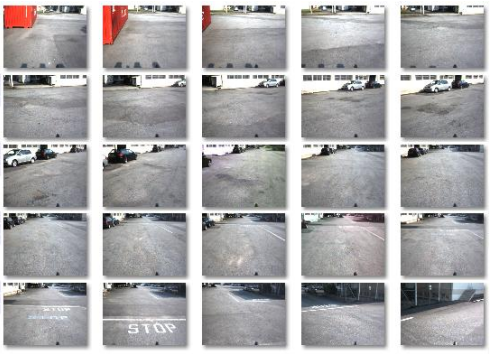
In the absence of rolling and yawing, the v-disparity algorithm has proven to be very effective in modeling the longitudinal ground profile. A judgment on its suitability to our application is hard to make without an explicit analysis of the nature of vehicle oscillations. If we allow the vehicle pose to vary without restrictions, both (5.6) and (5.7) transform to equivalents of (5.2) in disparity space. For a specific disparity $d = d_0$, (5.2) can be expressed as

$$a_u u + a_v v + a_d d_0 + \tilde{a_0} = 0 \Longrightarrow a_u u + a_v v + \hat{a_0} = 0 \tag{5.8}$$

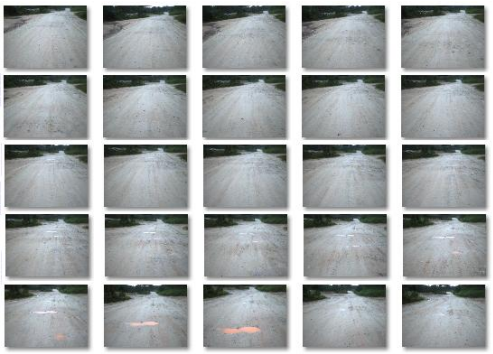
which represents a straight line in image pixel coordinates. When the vehicle undergoes pose variations, we observe a longitudinal shift and an in-plane rotation (or lateral variation) of this line. Alternatively this can be described as a variation in intercept and gradient. By simulating different combinations of rolling, yawing and pitching we observe that (5.8)

1. has a fixed gradient and a variable intercept when only pitching occurs;

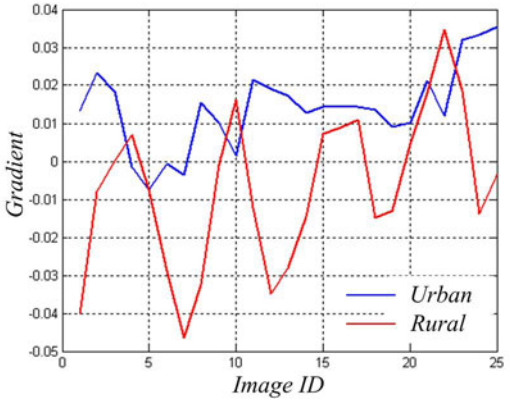
2. has a variable gradient and an intercept for any other form of pose variation.



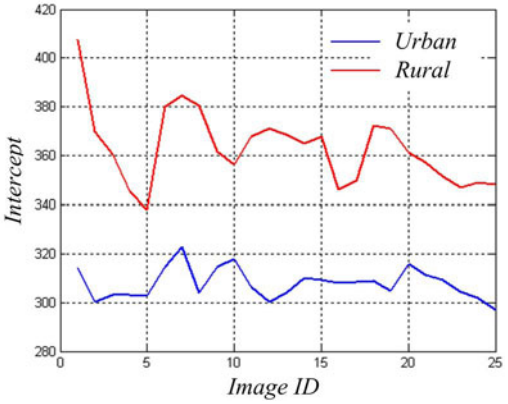
(a) Urban image sequence.



(b) Rural image sequence.



(c) Variation of gradient.



(d) Variation of intercept.

FIGURE 5.2: Effect of vehicle pose variation.

With this information in hand, we gauge the extent of vehicle oscillations occurring in rural environments by analyzing the gradient and intercept of a fixed ground disparity line and comparing it with similar characteristics of an urban road. For this purpose we analyze a short sequence of stereo images from an urban (Figure 5.2(a)) and a rural track (Figure 5.2(b)), which have been captured under identical settings (i.e., same vehicle moving at comparable velocities). Furthermore, in order to rule out the contribution of local topographic changes, we used rural terrain that has a flat ground appearance. The resulting plots of gradient and intercept, for a ground disparity line with $d_0 = 20$ (lying approximately 4m from the vehicle), are shown in Figures 5.2(c) and 5.2(d) respectively. It is clear that the variation of gradient in the urban environment is much lower compared with that of rural

terrain. If vehicle pitching is the only significant pose variation, we would have witnessed similar behaviors for both cases. Therefore, we make the reasonable assumption that rolling and yawing contribute significantly to the overall pose variation in the rural environments under consideration.

5.2.3 Ground Disparity Model

The analysis we have performed thus far suggests that both the planar ground approximation and v-disparity algorithm have their own strengths and weaknesses. While the former is better at modeling the ground profile in the lateral direction, the latter does well in the longitudinal direction. In our work we integrate these positive attributes into one coherent geometric model as follows:

- Allow multiple, non-zero lateral gradients (in contrast to the zero gradient in the v-disparity algorithm and single fixed gradient in planar ground);
- Approximate the longitudinal ground profile with a non-linear model (in contrast to the linear approximation in planar ground).

We propose to implement the above changes in two steps: the lateral ground disparity profile is modeled using a robust gradient estimation method, which is used during the subsequent minimum error v-disparity image construction. The longitudinal ground profile is approximated using a piecewise linear curve as well as a constraint satisfaction vector. An in-depth discussion of this ground plane modeling algorithm is the central topic of the next section.

5.3 Ground Plane Modeling

In this section we will assume the availability of a dense disparity map. Unless otherwise specified, disparity is considered to be of integer precision.

5.3.1 Ground Pixel Sampling

The most prominent advantage of dense disparity is that it avoids the need for an additional obstacle segmentation step. However, in the context of ground plane modeling, it presents a large volume of redundant information. Therefore, usually a subset of image points is used to perform the task of ground plane modeling. In doing so, we seek to maximize the likelihood of sampling ground pixels over pixels that have been projected from non-ground objects. We develop a deterministic sampling method based on the following heuristic:

"*Take a pixel with coordinates* (u, v) *and disparity* (d) *to be a ground pixel if its neighboring pixel with coordinates* $(u, v+1)$ *has a disparity equal to* $(d+1)$"

The underlying rationale of this heuristic can be easily explained using (4.11). According to this equation, a monotonic depth variation, for example the depth profile of a ground surface, generates a staircase signal in v-d space. Hence points belonging to similar scene structures can be located by searching for unit step increments of disparity along the longitudinal direction. In Figure 5.3, d_{G1}, d_{G2}, d_{G3} and d_{O1} represent the disparities of 3D scene points $G1$, $G2$, $G3$ and $O1$ respectively. We consider the following possibilities:

- Case I: $d_{G1} = d_{G2}$; trivial for ground pixel sampling.
- Case II: $d_{G1} = d_{G2} + 1$; a matching event to the heuristic condition. Image of $G1$ will be sampled as a ground pixel.
- Case III: $d_{O1} = d_{G3}$; unlikely to occur unless the obstacle is marginally protruding from ground.
- Case IV: $d_{O1} = d_{G3} + 1$; in general has a lower probability of occurrence for front-parallel obstacles. It is also determined by factors such as the distance from the stereo baseline to $O1$ and height and angle at which the cameras are mounted. Image of $O1$ will be falsely sampled as a ground pixel.

- Case V: $d_{O1} > d_{G3} + 1$; more likely since $O1$ and $G3$ are far apart for a front-parallel obstacle. An abrupt jump in disparity is expected.

Figures 5.4(a) and 5.4(b) show ground pixels sampled at disparity $d = 16$ according to our heuristic. The majority of ground points are accurately sampled, while minor misclassifications involving near-field obstacle pixels are caused by the errors propagated from the disparity calculation phase. Furthermore, Figure 5.4(c) plots the disparity profiles of the two cross-sections highlighted in Figures 5.4(a) and 5.4(b). It demonstrates that fronto-parallel surfaces generate abrupt disparity variations in contrast to unit step disparity increments of the ground surface.

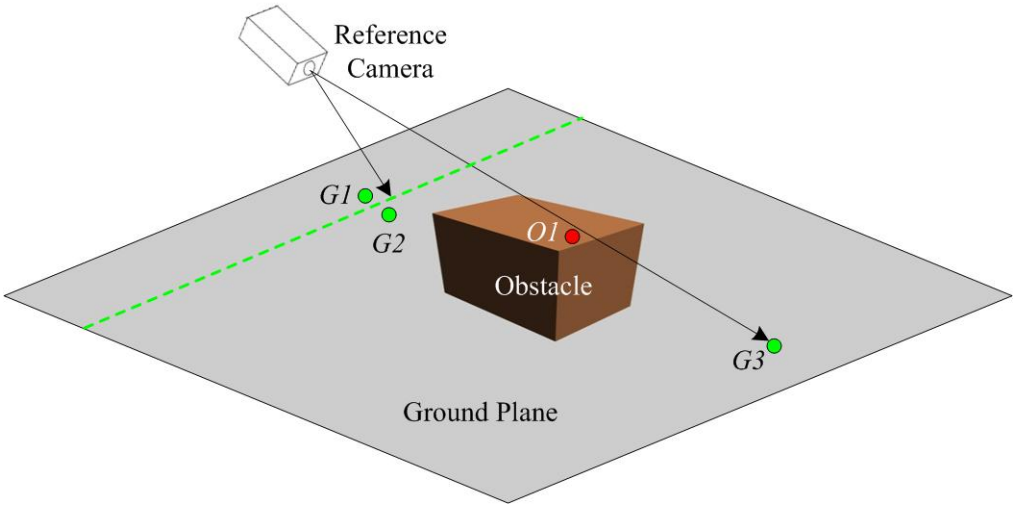
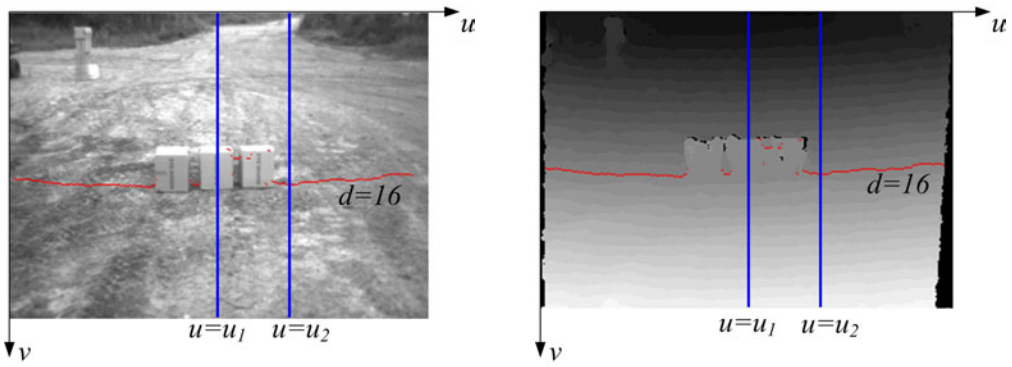


FIGURE 5.3: Illustration of ground pixel sampling heuristic.

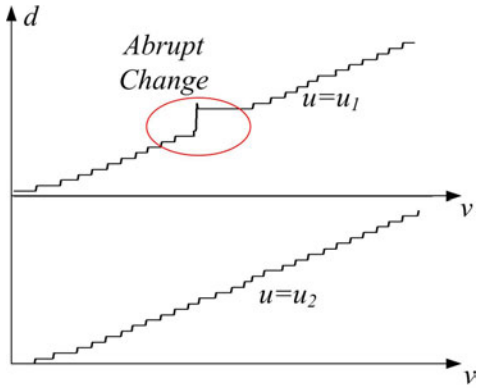
5.3.2 Lateral Ground Profile

According to the analysis performed in Section (5.2), we expect the ground pixels sampled at a particular disparity (S_d) to have a lateral gradient along the u-axis. In our model, lateral gradients (Δ_d) of the entire range of disparities, when considered together, form the lateral ground profile. Furthermore, to factor in possible topographic variations, we allow the lateral gradient to take more than one value for a given scene. As the first step of determining the lateral ground profile, we



(a) Reference image.

(b) Dense disparity map.



(c) Disparity profiles.

FIGURE 5.4: Ground point sampling.

sub-sample ground pixels at regular intervals along the u-axis as shown in Figure 5.5; for each sub-sample, at each disparity, a lateral gradient is calculated. As illustrated in Figure 5.5, gradient samples may also contain non-ground gradients. Therefore, the gradient population has to be further refined to counteract the effect of outliers before reliable estimates for Δ_d can be obtained. We experiment with two approaches to choose Δ_d values from a set of noise degraded gradients. In the following discussion we denote the gradient population for a particular disparity with $\Delta_{s,d}$ and the entire gradient population with Δ_s.

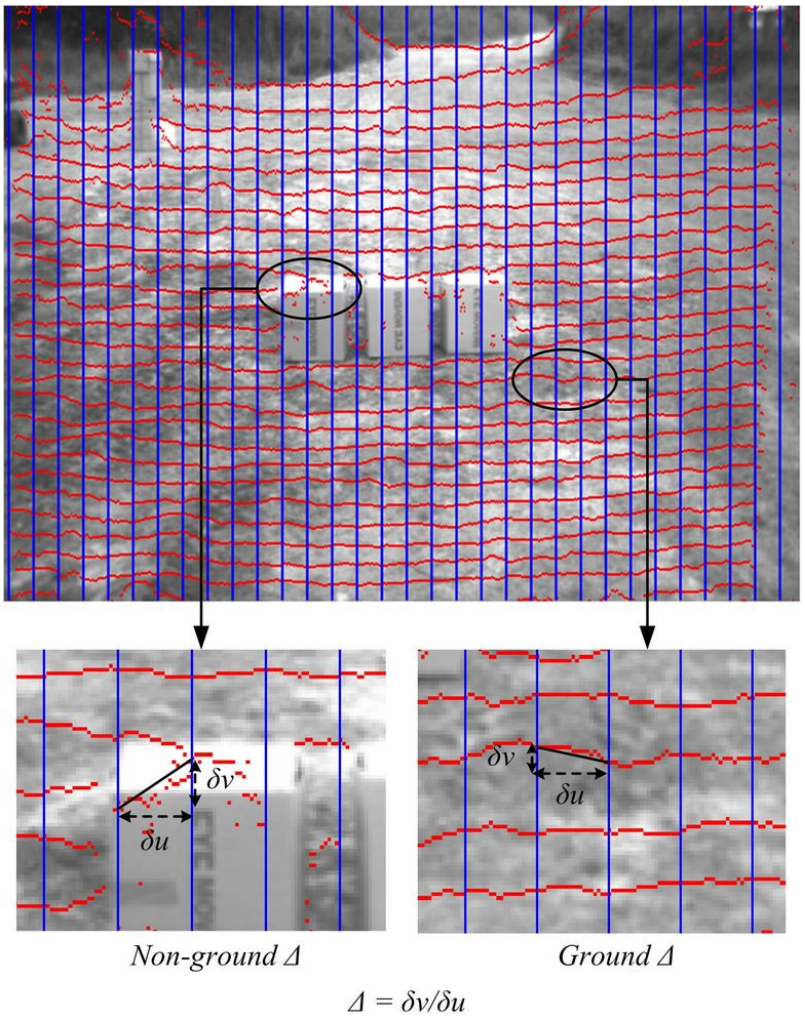


FIGURE 5.5: Lateral gradient sampling

1. Gradient Histogram

1. Construct the cumulative gradient histogram of Δ_s.
2. Discard the tails of the distribution using predefined cut-off values for Δ_d.
3. Find out all Δ_d with a probability greater than 75% of the maximum probability.
4. For each disparity, determine the best possible Δ_d by correlating with S_d.

For a given oblique plane to make a significant contribution to the gradient histogram, it should have a relatively consistent geometry over a large region. More

often than not, a ground plane satisfies this condition better than any other surface in an outdoor scene. Therefore, the histogram analysis above can be viewed as a voting scheme that assigns a fitness value to each different possibility of lateral ground gradient. Any candidate with a vote greater than 75% of the maximum vote will be considered suitable to be a member of the longitudinal ground profile. The correlation procedure associated with the final step is usually implemented as a part of the minimum error v-disparity generation process (discussed in Section 5.3.3).

2. Median Absolute Deviation

The existence of a distinct maximum in a probability distribution is loosely coupled with the extent of its dispersion. In here, we are interested in the probability distribution of $\Delta_{s,d}$. To quantify its dispersion, we compute a robust statistical measure, the median absolute deviation (MAD). The MAD for a sample S_i, drawn from a population S, is given by

$$\text{MAD}(S_i) = med(\,|S_i - med(S)|\,)$$

The relationship between a distinct maximum and dispersion might not always hold true for small populations. Therefore, to avoid the sample size from causing instabilities, we terminate the computation cycle when the sample size of $\Delta_{s,d}$ is smaller than a predefined threshold. The complete procedure of determining the lateral ground profile is as follows:

1. Discard extreme values of $\Delta_{s,d}$ using predefined cut-off values of Δ_d.
2. If the remaining sample size is below a predefined threshold, $\Delta_d = null$.
3. Otherwise calculate the MAD of $\Delta_{s,d}$, and
   (a) if it is less than a predefined threshold, output $\Delta_d = median(\Delta_s)$;
   (b) otherwise $\Delta_d = null$.

4. Repeat steps 1 to 4 for each d.

5. Approximate null values of Δ_d using nearest neighbor interpolation.

The undetermined Δ_d nodes indicate lack of ground-like evidence. As the final step, we approximate these empty nodes with nearest neighbor interpolation, which assumes points that are located close to each other on the ground plane to have similar geometric properties.

5.3.3 Longitudinal Ground Profile

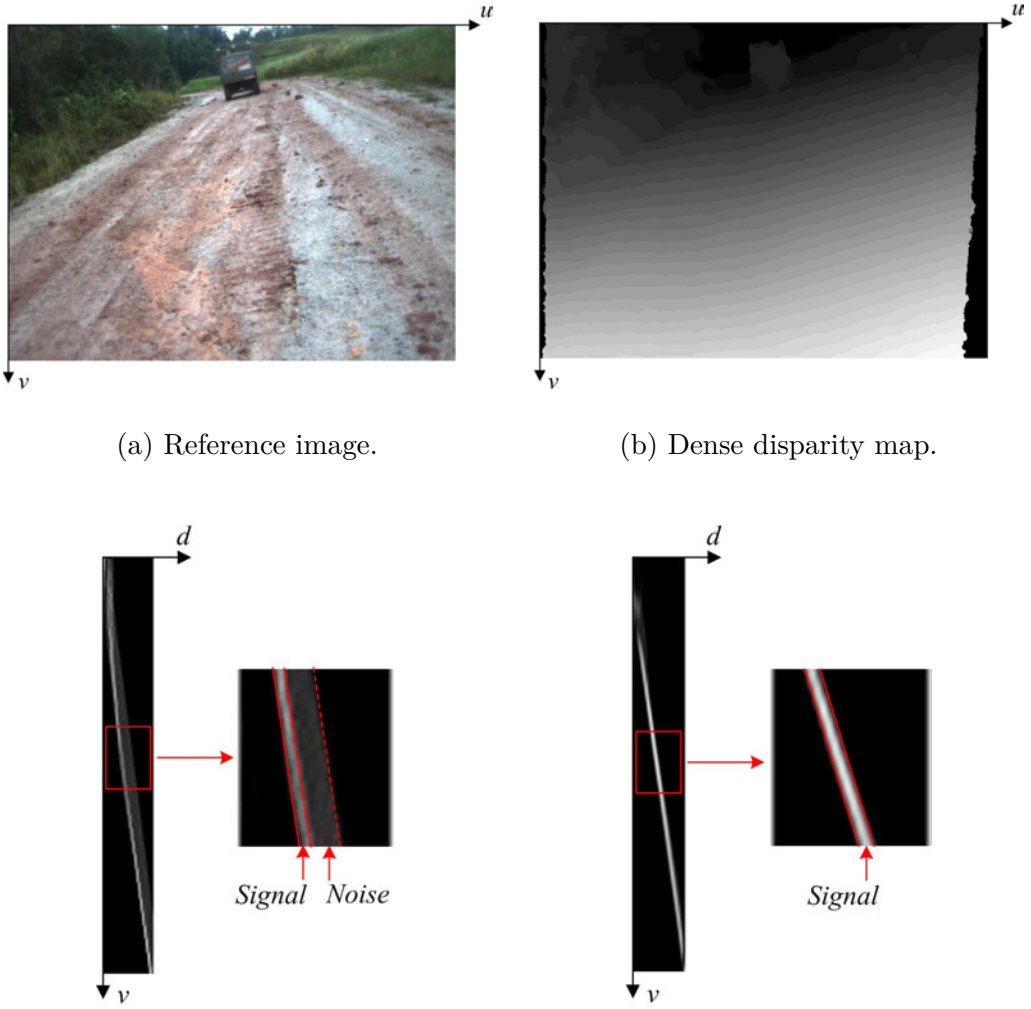
Minimum Error v-disparity Image

The traditional v-disparity algorithm not only causes a loss of lateral ground disparity gradients, but also produces a v-disparity image with poor SNR[2]. However, when lateral gradients are known beforehand, this problem can be alleviated by performing v-disparity projection along the directions of the lateral gradient. The graphical comparison in Figure 5.6 provides additional support to our claim above. Furthermore, in the v-disparity image, by replacing the frequency of disparity occurrence with a correlation function, we managed to achieve a considerable improvement. For any particular disparity d, a correlation function ρ_d can be calculated as:

$$\underset{v=v_{min}}{\overset{v=v_{max}}{\forall}} \left[\rho_d(v) = \frac{\sum G_{0,\sigma}(S_d - l_{d,v})}{N} \right] \tag{5.9}$$

where $G_{0,\sigma}$ denotes a Gaussian function with zero mean and σ standard deviation, $l_{d,v}$ a straight line with gradient Δ_d and intercept v, and N the image width in pixels. The intercept of $l_{d,v}$ is varied over the range of S_d and the correlation ρ_d is calculated at each instance. If S_d does not exist for a particular u, the difference between S_d and $l_{d,v}$ in (5.9) is forced to infinity for that particular u (which in turn is mapped to zero by the Gaussian function). In the case of the gradient histogram

[2]In a v-disparity image the signal is the *ground correlation line*, whereas the rest is considered to be noise



(a) Reference image.

(b) Dense disparity map.

(c) Traditional v-disparity image.

(d) Minimum error v-disparity image.

FIGURE 5.6: Minimum error v-disparity image.

method discussed above, (5.9) is calculated for all likely Δ_d values and only the best is retained. The notation and the described process are illustrated in Figure 5.7. The outcome of this process is what we call the minimum error v-disparity image, in which the dth column contains the correlation function $\rho_d(v)$. Similar to the previous case, we test two methods to model the *ground correlation line*.

1. Piecewise Linear Approximation

If we assume the curvature of the *ground correlation line* to have a constant sign, it can be modeled as a piecewise linear curve. The sequence of steps is as follows:

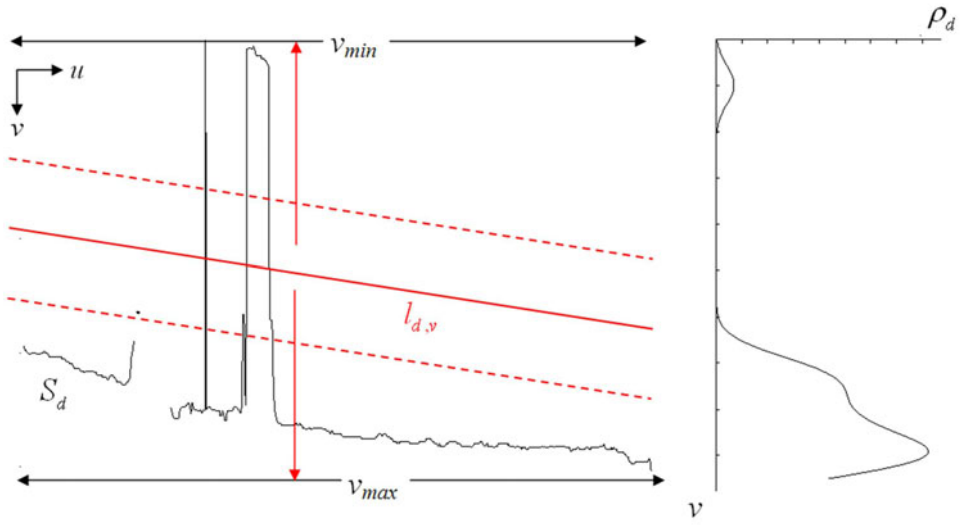


FIGURE 5.7: The v-disparity correlation scheme.

1. Normalize the minimum error v-disparity image by dividing each column with its maximum.

2. Compute the Hough transform (refer to Appendix B for more information) to detect straight lines on the minimum error v-disparity image; bound the Hough space using a-priori knowledge of camera and scene geometry.

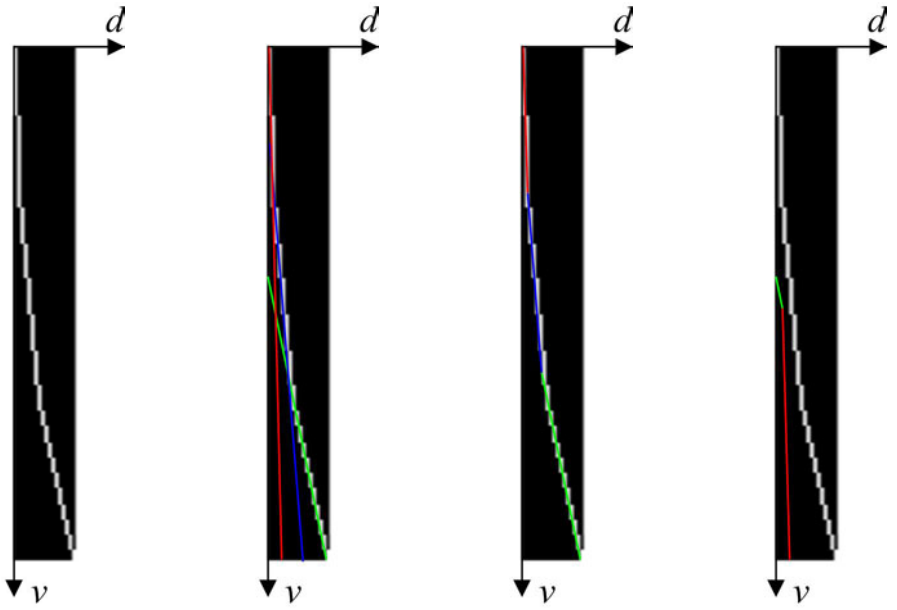
3. Perform *non-maxima suppression* in the Hough space within a $n \times n$ neighborhood; n is suitably selected depending on the precision of the Hough space.

4. Find the family of straight lines corresponding to Hough votes greater than 75% of the maximum Hough vote.

5. Determine the upper and lower envelopes of this family of straight lines (Figure 5.8).

6. Accumulate v-disparity scores along these two envelopes.

7. Return the envelope which is responsible for the larger value in step 6.



(a) The v-disparity image.

(b) Family of straight lines.

(c) Lower envelope.

(d) Upper envelope.

FIGURE 5.8: Detection of v-disparity image envelopes using the Hough transform.

Due to perspective distortion, the projection of the ground surface on the image plane appears progressively narrower with distance (e.g., Figure 5.6(a)). The column-wise normalization carried out in step 1 compensates for this effect and reduces the likelihood of over-fitting to near-field data. Apart from that, the method detailed above closely follows the longitudinal ground profile estimation procedure proposed in [30].

2. Constraint Satisfaction Vector

In this method, the idea is to seek an optimal ground plane geometry based on the available data. In order to do this, we identify two constraints that are necessary and sufficient to define a legitimate longitudinal ground profile:

- Constraint I: the v coordinates should monotonically increase with disparity (preserves the continuity of the ground plane).

- Constraint II: local gradient of the ground profile should remain below a pre-defined upper margin (limits the local slope of the ground plane).

We impose these constraints by defining each potential longitudinal ground profile as a constraint satisfaction vector. The complete procedure is as follows:

1. Threshold the minimum error v-disparity image.
2. Perform *non-maxima suppression* within a $n \times 1$ neighborhood; n is suitably selected according to the resolution of the v-disparity image.
3. Using different combinations of non-zero elements of the output of step 2, create a list of longitudinal ground profile vectors.
4. Delete vector elements which do not conform to either of the two constraints stipulated above; at this stage a vector with empty nodes is considered legitimate.
5. Filter out vectors with highest number of non-empty nodes.
6. If more than one vector is output in step 5, retain the vector corresponding to the maximum accumulated v-disparity score.
7. Interpolate for empty nodes using *piecewise cubic Hermite interpolation* (preserves the monotonicity and shape of data).
8. Return the longitudinal ground profile vector.

Unlike the piecewise linear approximation method, this method relies on local selection in a manner independent of the v-disparity score. Therefore, normalizing v-disparity image columns has no impact on the outcome. Instead, as the first step of this method, we discard unreliable evidence that falls below a pre-defined threshold, and subsequently perform *non-maxima suppression* to reduce the number of different longitudinal ground profiles to a manageable quantity.

Disparity (d)	v coordinates of the longitudinal ground profile
5	{205}
4	{186}
3	{171,157}
2	{161}
1	{169,129}

TABLE 5.1: Intermediate output of the constraint satisfaction vector method.

The constraint satisfaction process in steps 3 and 4 is best explained using an example. We consider a set of intermediate v coordinates of the longitudinal ground profile obtained as the output of step 2. If we assume the ground profile to be unconstrained, we can develop a number of different combinatorial vectors from the data given in Table 5.1. These vectors can then be verified against constraints I and II as shown in Figure 5.9.

5.4 Obstacle Detection

5.4.1 Image Domain Obstacle Detection

In off-road navigation, obstacles are categorized into two main classes, namely, positive obstacles and negative obstacles. A positive obstacle is an object that protrudes beyond the ground plane to an extent greater than the vehicle-to-ground clearance; when the deviation occurs in the reverse direction (i.e., a depression), it is called a negative obstacle. In GPOD, accurate modeling of the ground plane geometry is the most challenging task. When the ground plane model is already known, the obstacle detection process can be summarized by two rules:

1. If a pixel has a disparity greater than the disparity of the corresponding pixel in the ground model, mark it as a positive obstacle.

2. If a pixel has a disparity less than the disparity of the corresponding pixel in the ground model, mark it as a negative obstacle.

Constraint I	$168 > 149$ ✓	$149 > 134$ ✓	$134 > 115$ ✓	$115 > 92$ ✓
Constraint II	$\frac{(168-149)}{1} < 30$ ✓	$\frac{(149-134)}{1} < 30$ ✓	$\frac{(134-115)}{1} < 30$ ✓	$\frac{(115-92)}{1} < 30$ ✓
168	149	134	115	92

Constraint I	$168 > 149$ ✓	$149 > 134$ ✓	$134 > 94$ ✓	$134 > 92$ ✓
Constraint II	$\frac{(168-149)}{1} < 30$ ✓	$\frac{(149-134)}{1} < 30$ ✓	$\frac{(134-94)}{1} < 30$ X	$\frac{(134-92)}{2} < 30$ ✓
168	149	134	~~94~~ NULL	92

Constraint I	$168 > 172$ X	$168 > 134$ ✓	$134 > 115$ ✓	$115 > 92$ ✓
Constraint II	$\frac{(168-172)}{1} < 30$ ✓	$\frac{(168-134)}{2} < 30$ ✓	$\frac{(134-115)}{1} < 30$ ✓	$\frac{(115-92)}{1} < 30$ ✓
168	~~172~~ NULL	134	115	92

Constraint I	$168 > 172$ X	$149 > 134$ ✓	$134 > 94$ ✓	$134 > 92$ ✓
Constraint II	$\frac{(168-172)}{1} < 30$ ✓	$\frac{(149-134)}{1} < 30$ ✓	$\frac{(134-94)}{1} < 30$ X	$\frac{(134-92)}{2} < 30$ ✓
168	~~172~~ NULL	134	~~94~~ NULL	92

FIGURE 5.9: Imposing constraints on the longitudinal ground profile (Note: Gradient threshold was considered to be 30 in the above example).

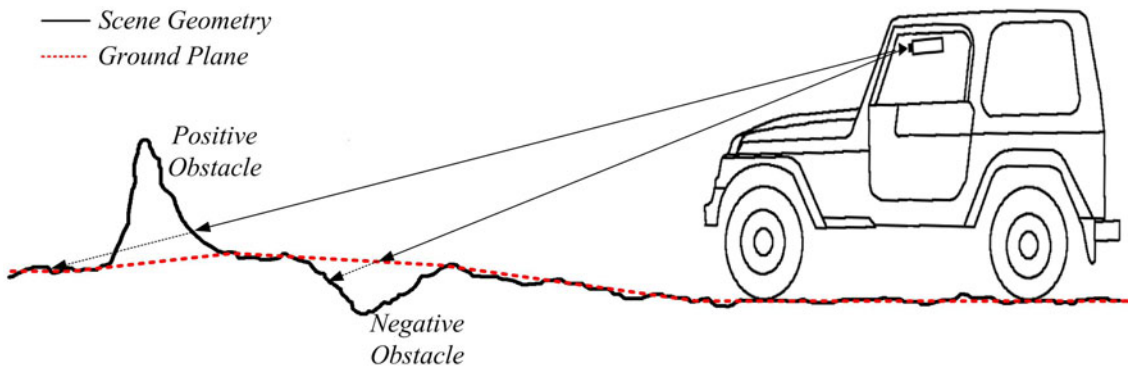


FIGURE 5.10: Projection of positive and negative obstacles.

The rationale behind above rules can be best explained using the illustration in Figure 5.10. As evident, a ray of projection intersects a positive obstacle before it intersects the ground plane. This means that, along a projection ray, a positive obstacle is located closer to the camera than the ground plane. Similarly we can observe that a negative obstacle is located further away along a projection ray when compared to the ground plane. These observations, when coupled with the inverse relationship between distance and disparity, imply the above rules. However, in reality, strictly adhering to these rules will result in a large number of false positives and negatives. Therefore, a suitable error tolerance band is usually determined by trial and error.

In reality, a positive obstacle can be anything that stands out from the ground plane, such as vehicles, animals, trees and vegetation. On the other hand, negative obstacles occur as an intrinsic part of the ground plane irregularity. For this reason, it is uncommon to encounter negative obstacles in semi-structured environments. This remains valid for the type of rural terrains of our concern, and hence only positive obstacle detection is implemented in our algorithm.

5.4.2 3D Representation of an Obstacle Map

Path planning for autonomous vehicles requires that a map of all potential obstacles be produced in real time using the available sensor data. Once obstacles are detected in the image domain, their spatial 3D locations can be expressed with respect to the reference camera coordinate frame with the aid of equations (4.11) - (4.13). However, knowing the obstacle location information in a camera frame that constantly varies its relationship with the ground plane is of little use for navigation. On the other hand, expressing the same information with respect to a world coordinate frame attached to the ground surface, preferably close to the front end of the vehicle, is more useful. In this section, we investigate the mathematical transformation between the reference camera coordinate frame and the world coordinate frame. The transformation we discuss assumes that the ground plane in the vicinity of the vehicle can be accurately approximated using a planar ground. This assumption holds true for a local region since we originally assumed a piecewise planar ground.

We now revert to the planar ground approximation model discussed in Section 5.1.1. The relationship between the coefficients of (5.1) and (5.2) can be alternatively given by

$$\begin{bmatrix} a_u \\ a_v \\ a_d \\ \tilde{a_0} \end{bmatrix} = \begin{bmatrix} b & 0 & 0 & 0 \\ 0 & b & 0 & 0 \\ 0 & 0 & 0 & 1 \\ -bu_0 & -bv_0 & f_p b & 0 \end{bmatrix} \begin{bmatrix} a_X \\ a_Y \\ a_Z \\ a_0 \end{bmatrix}$$

$$A = \begin{bmatrix} a_X \\ a_Y \\ a_Z \\ a_0 \end{bmatrix} = \begin{bmatrix} b & 0 & 0 & 0 \\ 0 & b & 0 & 0 \\ 0 & 0 & 0 & 1 \\ -bu_0 & -bv_0 & f_p b & 0 \end{bmatrix}^{-1} \begin{bmatrix} a_u \\ a_v \\ a_d \\ \tilde{a_0} \end{bmatrix}$$

Define

$$A_{new} = \begin{bmatrix} a_{1,new} \\ a_{2,new} \\ a_{3,new} \\ a_{4,new} \end{bmatrix} = \frac{A}{\|[a_X \quad a_Y \quad a_Z]^T\|}$$

in which the first three components represent the unit normal vector to the ground plane and the fourth component is the normal distance from the camera center to the ground plane. Intuitively we would want the Y axis of the world coordinate frame to be normal to the ground. Hence, we define

$$\overrightarrow{Y_{new}} = \begin{bmatrix} a_{1,new} \\ a_{2,new} \\ a_{3,new} \end{bmatrix} \tag{5.10}$$

The orientations of X and Z should remain unchanged but to be useful for navigation they should coincide with the ground plane. Therefore we define

$$\overrightarrow{X_{new}} = \frac{Y_{new} \times [0 \quad 0 \quad 1]^T}{\|Y_{new} \times [0 \quad 0 \quad 1]^T\|} \tag{5.11}$$

$$\overrightarrow{Z_{new}} = \overrightarrow{X_{new}} \times \overrightarrow{Y_{new}} \tag{5.12}$$

We consider two coordinate frames $\{X, Y, Z\}$ and $\{X', Y', Z'\}$ which are related through an arbitrary rotation. If a vector $\overrightarrow{t}$ in $\{X, Y, Z\}$ transforms to a vector $\overrightarrow{t'}$ in $\{X', Y', Z'\}$, we may write the following relationship:

$$t'_X = t.\overrightarrow{i'} = (t_X\vec{i} + t_Y\overrightarrow{j} + t_Z\overrightarrow{k}).\overrightarrow{i'} = t_X\vec{i}.\overrightarrow{i'} + t_Y\overrightarrow{j}.\overrightarrow{i'} + t_Z\overrightarrow{k}.\overrightarrow{i'}$$
$$t'_Y = t.\overrightarrow{j'} = (t_X\vec{i} + t_Y\overrightarrow{j} + t_Z\overrightarrow{k}).\overrightarrow{j'} = t_X\vec{i}.\overrightarrow{j'} + t_Y\overrightarrow{j}.\overrightarrow{j'} + t_Z\overrightarrow{k}.\overrightarrow{j'}$$
$$t'_Z = t.\overrightarrow{k'} = (t_X\vec{i} + t_Y\overrightarrow{j} + t_Z\overrightarrow{k}).\overrightarrow{k'} = t_X\vec{i}.\overrightarrow{k'} + t_Y\overrightarrow{j}.\overrightarrow{k'} + t_Z\overrightarrow{k}.\overrightarrow{k'}$$

which can be written in matrix form as

$$\begin{bmatrix} t'_X \\ t'_Y \\ t'_Z \end{bmatrix} = \begin{bmatrix} \vec{i}.\overrightarrow{i'} & \overrightarrow{j}.\overrightarrow{i'} & \overrightarrow{k}.\overrightarrow{i'} \\ \vec{i}.\overrightarrow{j'} & \overrightarrow{j}.\overrightarrow{j'} & \overrightarrow{k}.\overrightarrow{j'} \\ \vec{i}.\overrightarrow{k'} & \overrightarrow{j}.\overrightarrow{k'} & \overrightarrow{k}.\overrightarrow{k'} \end{bmatrix} \begin{bmatrix} t_X \\ t_Y \\ t_Z \end{bmatrix} \tag{5.13}$$

Following (5.13), R_{C2W}, the rotation matrix from camera to world coordinate frame can be written in terms of the unit vectors in (5.10), (5.11) and (5.12) as

$$R_{C2W} = \begin{bmatrix} (\overrightarrow{X_{new}})^T \\ (\overrightarrow{Y_{new}})^T \\ (\overrightarrow{Z_{new}})^T \end{bmatrix} \tag{5.14}$$

T_{W2C}, the translation vector from camera to world frames is given by

$$T_{W2C} = \begin{bmatrix} 0 \\ a_{4,new} \\ 0 \end{bmatrix} \tag{5.15}$$

The final 3×4 transformation matrix is constructed by concatenating (5.14) and (5.15).

Once the above process is completed, obstacles can be represented in the form of an occupancy grid. An occupancy grid is a 2D grid made of the X and Z axes of the world coordinate frame. Each grid node contains the average height (or average Y value in world coordinates) of obstacles falling within its boundaries. The occupancy grid is the ultimate output of the stereo vision based obstacle detection system discussed here. It will then be transferred to the path planning module, which combines it with other sensor information to accomplish safe and efficient maneuvering of the unmanned ground vehicle over rural terrains.

Chapter 6

Results and Discussion

The previous two chapters have presented the algorithm design considerations that have gone into our stereo vision based obstacle detection system. In this chapter we will discuss the implementation details and the performance of individual system components under a variety of test conditions.

6.1 Implementation and Analysis

6.1.1 Implementation Details

The performances of both stereo correspondence and obstacle detection algorithms largely depend on appropriate selection of input parameters, threshold values and termination conditions. In reality this is one of the most demanding tasks. The different algorithm parameters used in the final implementation are summarized in Table 6.1. While some of these parameters have been estimated using trial and error, the remainder is determined by analyzing the error statistics of a range of probable values. More information on parameter estimation can be found in the sections to follow.

Algorithm	Parameter	Description	Allowable value(s)	Chosen value
SC → Image Enhancement	W_{LoG}	Square window size: LoG filter	$W_{LoG} > 1$	5 pixels
	W_{Rank}	Square window size: rank transform	$W_{Rank} > 1$	5 pixels
	W_{Census}	Square window size: census transform	$W_{Census} > 1$	3 pixels
SC → Dense Disparity Computation	W_{SAD}	Square window size: SAD	$W_{SAD} > 1$	11 pixels
	d_{max}	Maximum disparity of the scene	Variable	30
SC → Elimination of Low-confidence Matches	$T\_C_{e,N}$	Threshold: normalized entropy	$0 \leq T\_C_{e,N} \leq 1$	0.9995
	T_C_{Win}	Threshold: winner margin	$0 \leq T\_C_{Win} \leq 1$	0.05
GGM → Lateral Ground Profile	δ_u	Sampling interval along u-axis	$\delta_u \geq 1$	15 pixels
	$\Delta_{d,min-max}$	Cutoff values: lateral gradient	$(-\infty, \infty)$	(-0.33,0.33)
	$T\_\Delta_{S,d}$	Threshold: remaining number of samples (MAD)	$0 \leq T\_\Delta_{S,d} \leq 640$	50
	T_MAD	Threshold: MAD	$T\_MAD \geq 0$	2
GGM → Longitudinal Ground Profile	G_{HS}	Gradient resolution: Hough Space	$G_{HS} > 0$	0.1
	I_{HS}	Intercept resolution: Hough Space	$I_{HS} > 0$	1
	W_2D_{NMS}	Square window size: non maxima suppression	$W\_2D_{NMS} > 1$	3 pixels
	T_MEVD	Threshold: minimum error v-disparity image	$T\_MEVD \geq 0$	0.1
	W_1D_{NMS}	1D window size: non maxima suppression	$W\_1D_{NMS} > 0$	5 pixels
	Δ_v	Gradient threshold: Constraint II	$\Delta_v \geq 0$	30 pixels
OD → Image Domain OD	T_POBS	Threshold : positive obstacle to ground deviation	$T\_POBS \geq 0$	0.5m

TABLE 6.1: System parameters. Key to abbreviations: SC - stereo correspondence, GGM - ground geometry modeling, OD - obstacle detection.

In our application, the stereo image pairs are captured, down-sampled to 640×480, subjected to stereo rectification and input to the stereo correspondence and obstacle detection routines. The entire process from image capturing to occupancy grid generation runs at around 3 frames per second on a modern day computer (2.8GHz Intel quad-core processor running on Windows XP). Although the initial prototyping was done in Matlab, to achieve aforementioned computational speed, the final implementation was carried out in C++ using the Intel open source computer vision library (OpenCV) [78]. The program was partially optimized using Intel's Integrated Performance Primitives (IPP)[1] [79] (to accelerate certain OpenCV functions) and OpenMP[2] [80] (to implement parallel processing). The breakdown of approximate computational times for sub-components of our algorithm in one cycle are as follows:

- Stereo rectification - 10ms
- Image enhancement - 20ms
- Disparity map generation - 160ms
- Ground plane model computation - 90ms
- Obstacle detection in image domain - 10ms
- Occupancy grid representation - 20ms

6.1.2 Data Simulation and Collection

An accurate evaluation of any algorithm requires either a theoretical basis or access to some ground truth knowledge of the problem in hand. Similarly, to assess the

[1] Intel IPP is an extensive library of multicore-ready, highly optimized software functions for digital media and data-processing applications. It offers thousands of frequently-used functions that are optimized to deliver performance beyond what optimized compilers alone can deliver.

[2] A multi-platform shared-memory parallel programming API in C/C++.

effectiveness of the stereo vision and obstacle detection algorithms described thus far, we generate a synthetic disparity map by hypothesizing the parameters of a ground plane. The proposed disparity simulation process consists of the following steps:

1. For each disparity, compute a straight line with gradient equal to the assumed lateral gradient and intercept equal to the v coordinate of the assumed longitudinal ground profile at that disparity.

2. Generate an integer precision, dense disparity map using the above lines as level curves.

3. Add random Gaussian noise.

4. Manually insert disparity segments to simulate scene elements lying on the ground plane.

The outputs of steps 2, 3 and 4 are shown in Figures 6.1(a), 6.1(b) and 6.1(c) respectively. In practice, it is almost impossible to encounter an environment with a ground disparity map as consistent as the one depicted in Figure 6.1(a); the addition of Gaussian random noise in the subsequent stage brings it closer to a real world ground disparity map. Then again, it is unlikely for an outdoor environment to be entirely composed of the ground plane, and hence the process is incomplete until we insert disparity segments that simulate objects other than the ground.

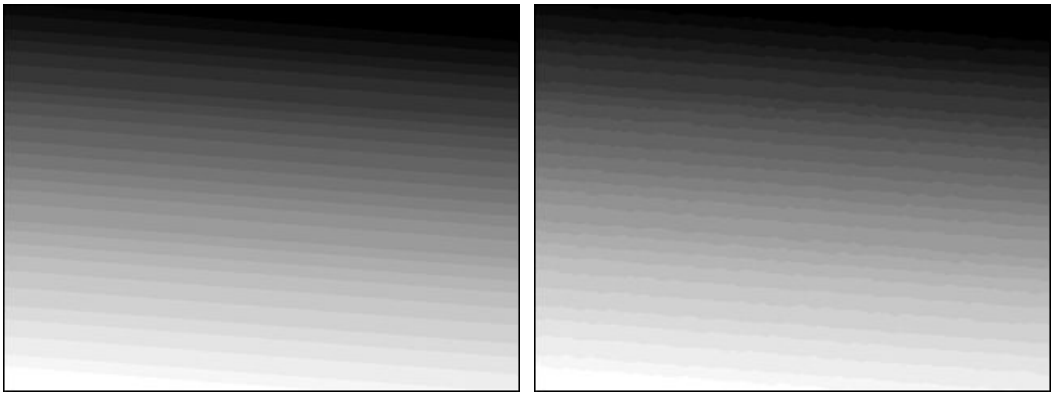
In addition to a disparity map of known ground truth, quantitative performance evaluation of the stereo correspondence algorithm requires a stereo image pair conforming to the computed disparity map. To satisfy this requirement, we adapt the popular random dot stereogram method [81]. As the name suggests, the resulting image pair of a random dot stereogram consists of seemingly random and uncorrelated dots. The complete procedure is as follows:

1. Start with a gray scale image of a rural terrain, and randomly scatter its gray values over the $u - v$ space to generate the right image.

2. Construct the corresponding left view by horizontally shifting gray values of the right image according to a ground truth dense disparity map.

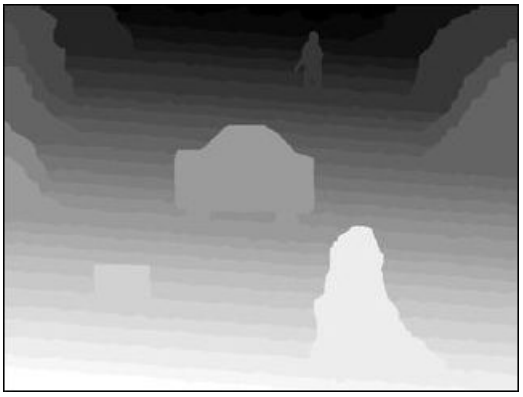
3. Add low pass filtered Gaussian random noise to the left image.

In the first step, using an actual image as the input ensures that the gray value distribution (or image entropy) of the computed random dot image is comparable to that of typical images considered in our work; Figures 6.2(a) and 6.2(b) show,



(a) Ideal ground plane disparity.

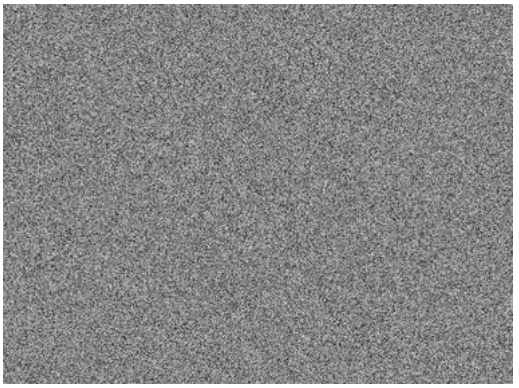
(b) Real world ground plane disparity.



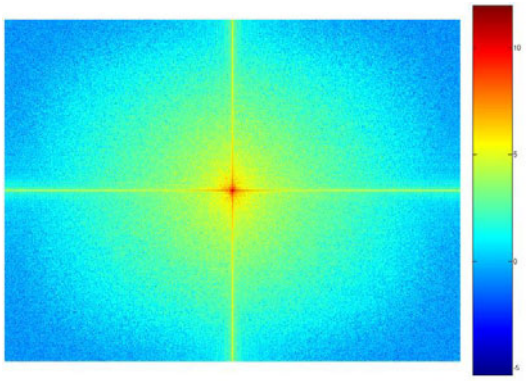
(c) Real world disparity map with obstacles.

FIGURE 6.1: Ground truth disparity simulation.

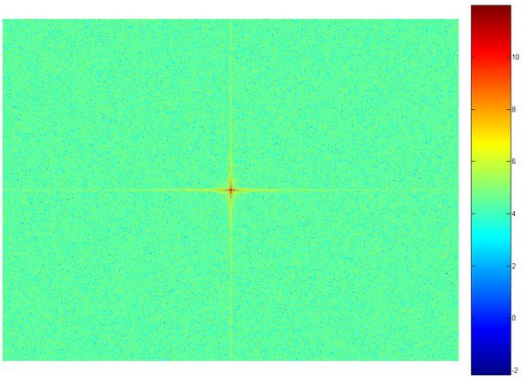
(a) Source image.



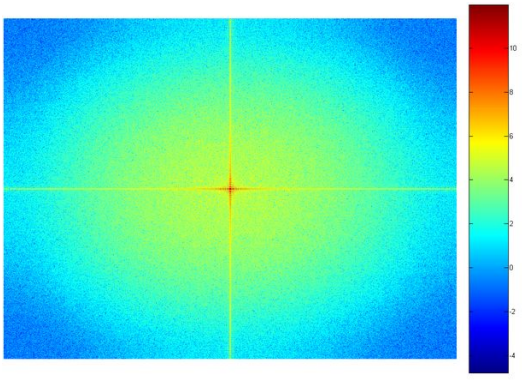
(b) Random dot image.



(c) Fourier spectrum magnitude of the source image.



(d) Fourier spectrum magnitude of the random dot image.



(e) Fourier spectrum magnitude of a low pass filtered random dot image.

FIGURE 6.2: Random dot image generation.

respectively, the input and the output of this step. The frequency spectrum magnitudes of the respective images plotted in Figures 6.2(c) and 6.2(d) demonstrate a relatively larger high frequency content on the part of the random dot image. A high frequency intensity variation is a desirable property for stereo matching, therefore, it enables us to compute a robust disparity map without having to perform an additional image enhancement step. However, when we need to assess the effectiveness of image enhancement, we will subject the random dot images to a Gaussian low pass filtering such that the resulting spectrum will be similar to Figure 6.2(c). An example is shown in Figure 6.2(e). As the final step, we add low frequency Gaussian noise to account for the possible intensity fluctuations caused by the difference in perspectives.

Apart from the simulated data, our algorithms have also been extensively tested with several field image data sequences that were captured by driving the UGV in semi-structured, cross-country roads at speeds not exceeding 40kmph. The data collection was predominantly performed under clear ambient lighting and weather conditions, but both wet environments (consisting of water puddles) and dry environments (consisting of dust clouds) were taken into account. Other than natural obstacles (e.g., vegetation, road-side depressions, water reservoirs and soil barriers), other objects (e.g., vehicles, human beings and cardboard boxes) were purposely placed during data capturing to assess the effectiveness of our obstacle detection algorithm. Table 6.2 provides an overview of the field data that has been tested with our system.

Image sequence ID	No. of image pairs	Navigated distance
R20	15713	~20km
R8	6918	~8km
R4.5	3539	~4.5km
R1.8	1484	~1.8km

TABLE 6.2: Composition of field test data.

6.2 Stereo Algorithm Evaluation

6.2.1 Window Size Selection

Both image enhancement and SAD procedures require the local window size to be specified as an input parameter. As mentioned earlier, since the random dot images demonstrate a high frequency intensity variation, it is reasonable to bypass image enhancement and directly proceed to the disparity computation phase. Therefore, we determine the optimum window size for SAD correlation first and then use the result to obtain a similar estimate for feature enhancement filter size. The two random dot images are matched for a range of SAD correlation window sizes and the root-mean-square (RMS) error between the computed disparity map ($d_C(u, v)$) and the ground truth disparity map ($d_{GT}(u, v)$) is calculated as follows:

$$\text{RMS} = \sqrt{\frac{1}{M \times N}\sum_{u,v}[d_C(u, v) - d_{GT}(u, v)]^2} \tag{6.1}$$

The error curves obtained by varying the square window size from 3×3 to 41×41 is shown in Figure 6.3. This experiment demonstrates that when the correlation window size is gradually increased from 3×3, the disparity error rapidly declines, but when the window size is expanded beyond 11×11, it begins to rise again. The disparity error follows a similar trend for repeated analysis over different intensities of additive noise. Therefore, we select a 11×11 square SAD correlation window for the final implementation of the stereo correspondence algorithm.

As previously discussed in Section 4.3.1, we experiment with three image enhancement techniques. Each of these methods operates within a local neighborhood or a window area of the image. To determine the appropriate enhancement technique and the associated window size giving rise to the minimum disparity error, we fix the SAD correlation window size at 11×11 and perform stereo matching by varying the enhancement filter size (with the exception of the census transform).

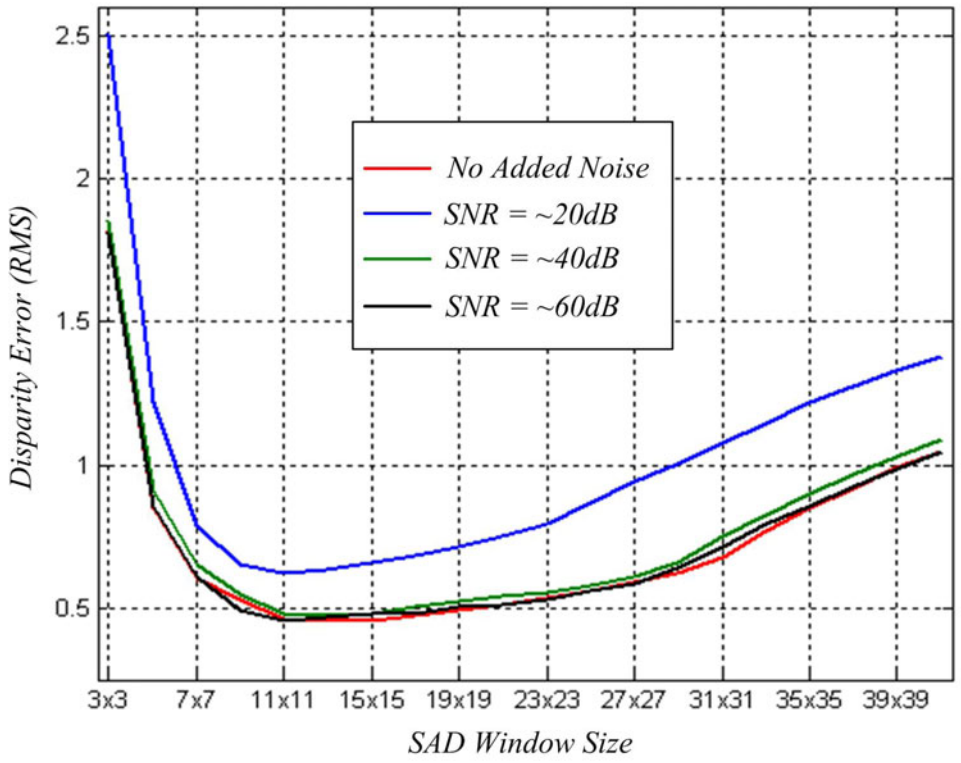
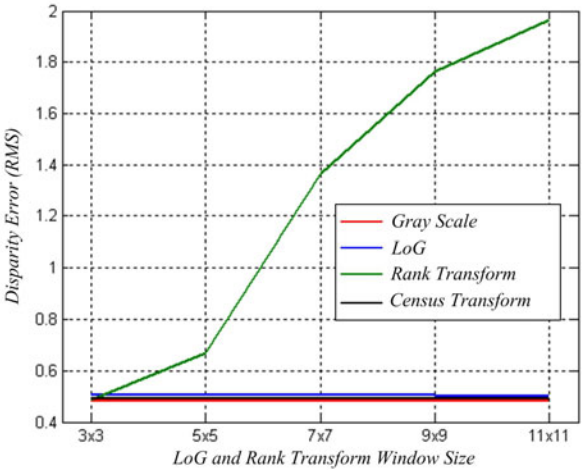
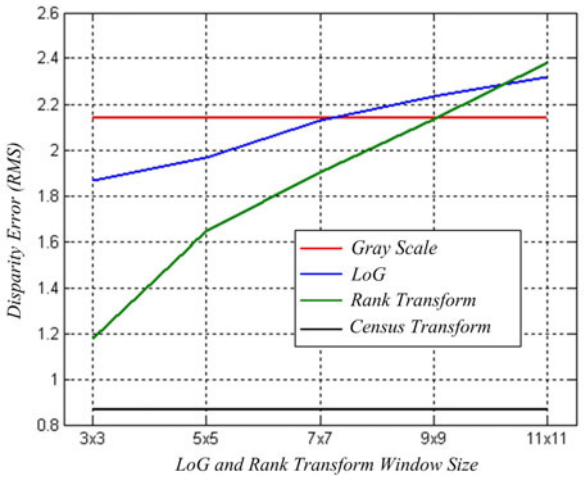


FIGURE 6.3: Variation of RMS disparity error with SAD window size.

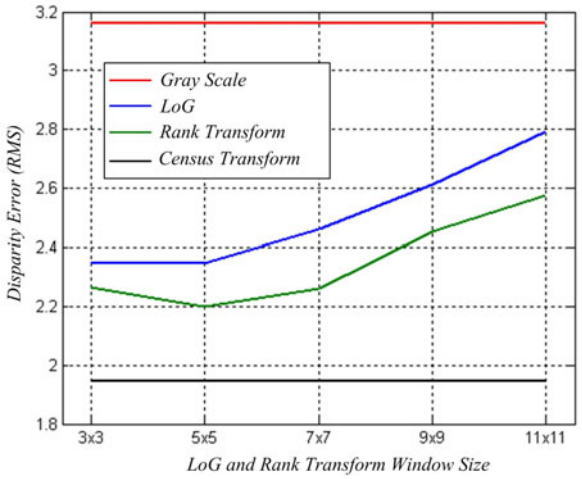
The bit-wise operation of the census transform becomes prohibitively expensive in terms of memory and computational power for large window sizes; therefore we will only test it for a 3×3 window. In order to benchmark the performance of different enhancement filters, we incorporate gray scale SAD to the same analysis. Figures 6.4(a), 6.4(b) and 6.4(c) depict the error profiles of the gray scale SAD and enhancement methods under consideration for a pair of random dot images with 40dB SNR. It is important to note that window size is varied only for the LoG and rank transform. The RMS error is calculated using (6.1) as before. It is clear from this analysis that when the image has a high frequency content, for instance a random dot image, further image enhancement is trivial or can even create undesirable effects. On the other hand, when the image spectrum is dominated by low frequency content, the rank and census non-parametric measures outperform the LoG and gray scale SAD. Due to consistent superior performance shown by the census transform, it is chosen over others for the final design of our stereo algorithm.



(a) Input: random dot image pair.



(b) Input: random dot image pair averaged with a 7×7 window.



(c) Input: random dot image pair averaged with a 13×13 window.

FIGURE 6.4: Comparison of image enhancement techniques.

6.2.2 Dense Disparity: Performance Evaluation

In this section, we compare and contrast the performance of our stereo algorithm against other iterative aggregation and global optimization techniques mentioned in Section 4.3.2. We also include the normalized cross correlation (NCC) matching cost computation [82] in the same analysis. Except for our algorithm and NCC, all other methods are evaluated using the two-frame dense stereo matching platform developed by Scharstein and Szeliski [83]. Additional information on this program, including a definition of its parameters, is provided in [59]. Apart from NCC, all methods use absolute difference as the matching cost and all non-iterative methods aggregate the costs within a square window to reach the final correlation. Table 6.3 presents a performance comparison of each considered method for the same pair of random dot images used during the enhancement filter size selection. The resulting dense disparity maps of the non-iterative and iterative methods are shown in Figures 6.5 and 6.6 respectively. We make the following observations based on the RMS disparity error:

Method	Parameters	Computational Method	RMS Error
Our method	Census transform window size = 3×3, SAD window size = 11×11	Non-iterative	0.5077
NCC	NCC window size = 11×11	Non-iterative	1.1239
Shiftable windows	SAD window size = 11×11, Shiftable area = 7×7	Non-iterative	0.5366
Regular diffusion	Diffusion coefficient $\lambda = 0.15$	Iterative	1.9265
Membrane diffusion	Diffusion coefficient $\lambda = 0.15$, Membrane coefficient $\beta = 0.5$	Iterative	1.2422
Graph cut	Optimization smoothness = 50	Iterative	0.3667
Dynamic programming	Optimization smoothness = 1, Occlusion cost = 10	Iterative	0.4755

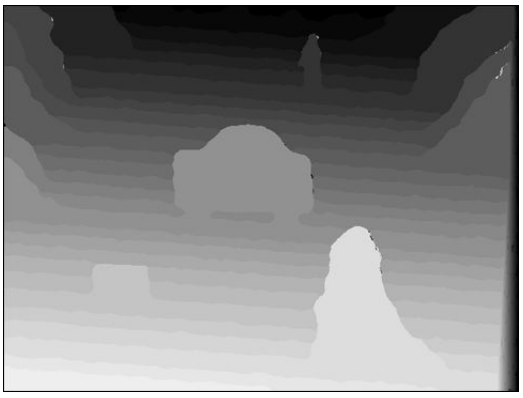
TABLE 6.3: Performance evaluation of dense two-frame stereo correspondence methods.

1. The performance of our algorithm is only second to global optimization methods.

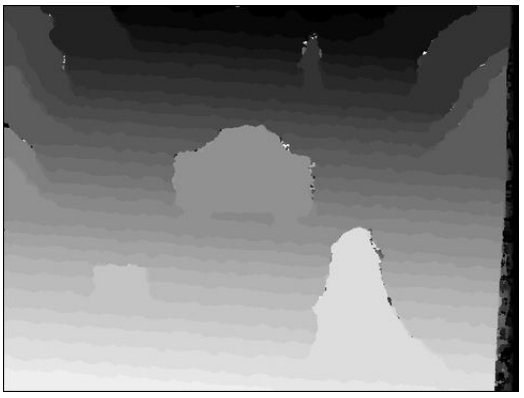
2. Despite being iterative, diffusion methods are inferior to all other methods.

3. Shiftable windows shows marginally comparable accuracy to our algorithm.

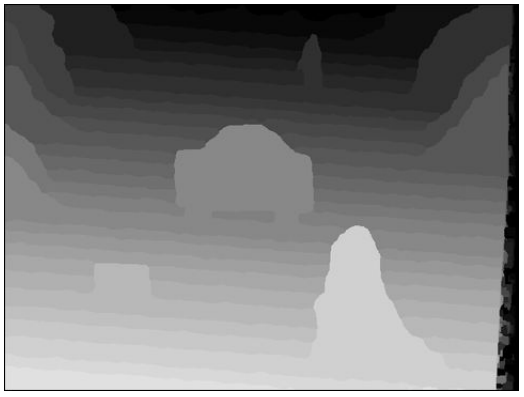
Even though we have considered iterative global optimization methods for the sake of completeness, they are inapplicable to a real time vision based navigation system of our kind. Therefore, in terms of computational complexity and accuracy, the best contender to our algorithm is the shiftable windows method. We acknowledge that with some mathematical manipulation, it can be efficiently implemented using



(a) Our stereo algorithm.

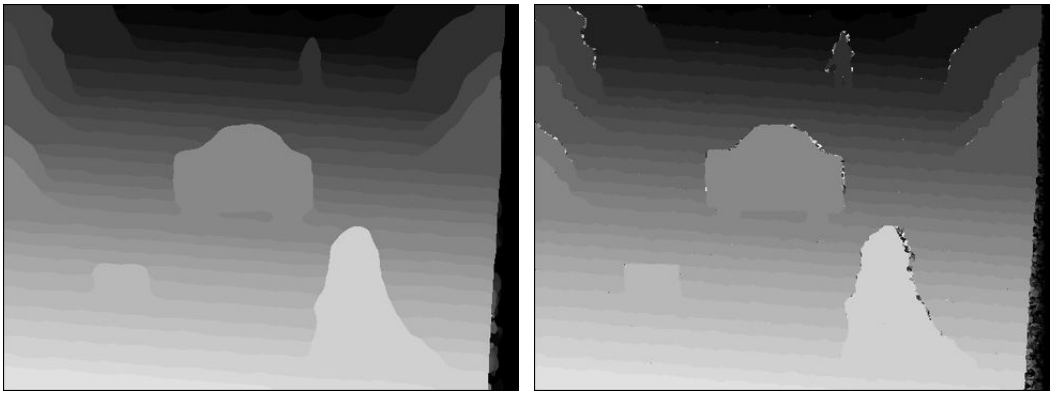


(b) NCC.



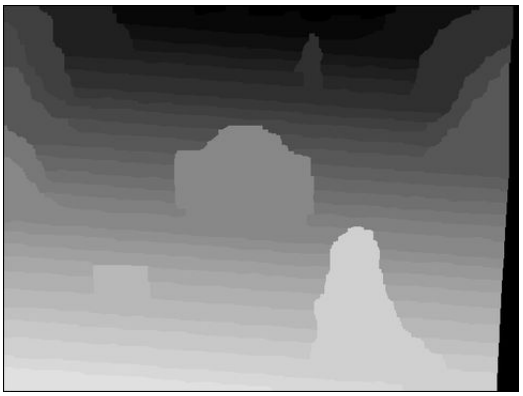
(c) Shiftable windows.

FIGURE 6.5: Results of non-iterative dense disparity computation.

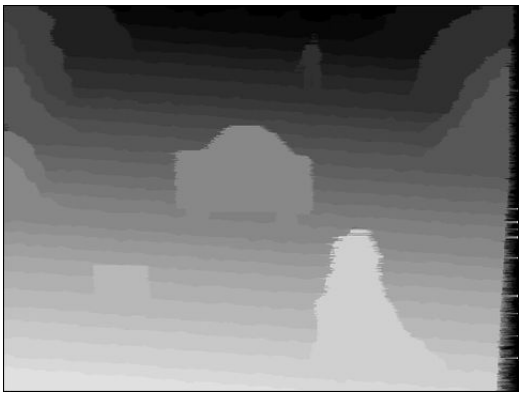


(a) Regular diffusion.

(b) Membrane diffusion.



(c) Graph cuts.

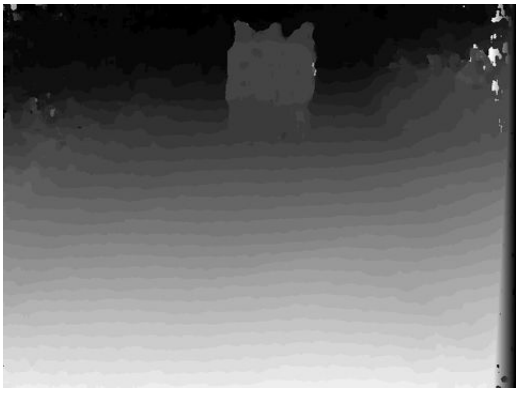


(d) Dynamic programming.

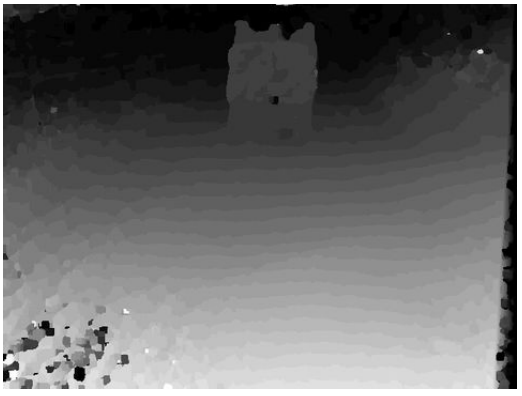
FIGURE 6.6: Results of iterative dense disparity computation.

a separable sliding min-filter and a separable moving average filter. The cascaded effect of these two filters is equivalent to evaluating a complete set of shifted windows since the value of a shifted window is the same as that of a window centered at some neighboring pixel. However, we also experience that the disparity maps produced by the shiftable windows method tend to be noisy for field image data (Figure 6.7). Therefore, in our final implementation, we stick to the census transform method.

(a) Reference image.



(b) Disparity map of the proposed algorithm.



(c) Disparity map of shiftable windows.

FIGURE 6.7: Performance comparison for field data.

6.2.3 Elimination of Low-confidence Matches

In Section 4.3.3, we discussed three possible methods to evaluate the confidence level or uncertainty of a correlation function. These methods examine the existence of a distinct matching offset for a given pair of matching windows. In practice, factors such as perspective distortion, texture content and illumination conditions contribute at different proportions to matching ambiguity, making it extremely difficult to simulate stereo images of measurable uncertainty. For this reason, we perform a qualitative assessment of the proposed uncertainty measures by trial and error on few selected test images.

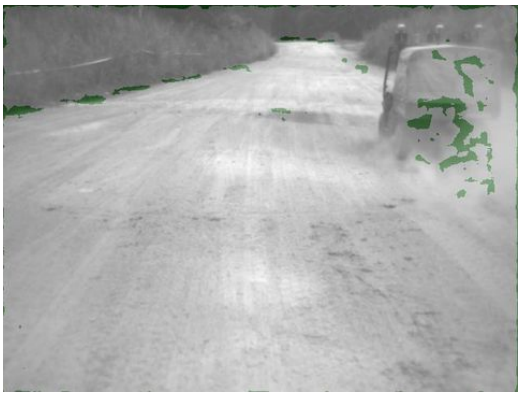
The entropy and winner margin methods require suitably selected threshold values to make a binary decision regarding the uncertainty of a correlation function. For each method, these values are determined by iteratively sampling the decision space of test images at different thresholds; the threshold which produces the least number of false positives while detecting the majority of uncertainties is selected as the optimum threshold. Due to the lack of clear-cut definitions, uncertainties and false positives have to be distinguished in image space using intuitive guesses. To facilitate this process, prior to uncertainty detection, we inspect the input images and identify areas that are likely to have uniform appearance over a sliding window. Examples of these kind of areas in our test images are: dust clouds (Figure 6.8(a)), specular reflections on water puddles (Figure 6.9(a)) and over exposed ground surface (Figure 6.10(a)). We will fine tune the threshold values such that the uncertainties on these regions are maximized without compromising the disparity calculation in the rest of the image. The detected uncertainties using each method are highlighted in Figures 6.8-6.10 (b), (c) and (d). On average, the winner margin method is able to capture about 80% of the uncertainties detected by the left-right consistency check or entropy method. Therefore, by implementing only the winner margin method we achieve a considerable gain in computational speed without compromising the accuracy.

6.2.4 Sub-pixel Interpolation and 3D Reconstruction

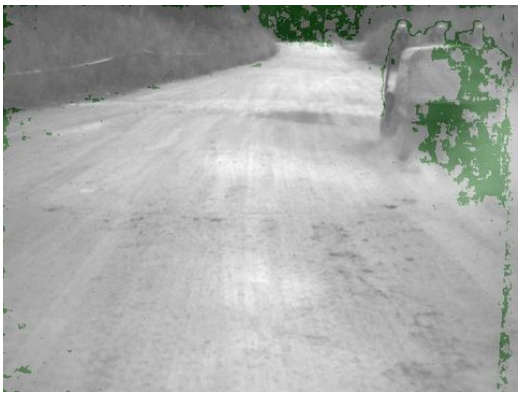
The underlying mathematics of sub-pixel interpolation using parabolic and Gaussian fitting has been derived in Section 4.3.4. Furthermore, based on referred literature, we reported that parabolic fitting is susceptible to the pixel locking effect more than Gaussian fitting. Here, we perform a simple experiment to verify this claim. We consider three instances of a correlation function: ($d_{-1} = 1$, $\theta_{-1} = 2.1$), ($d_0 = 2$, $\theta_0 = 2.1$) and ($d_{+1} = 3$, $\theta_{+1} = 5.4$). As it stands, both methods produce a sub-pixel estimate of 1.5. Next, by varying θ_{-1} from 2.1 to

(a) Reference image.

(b) Left-right consistency check.



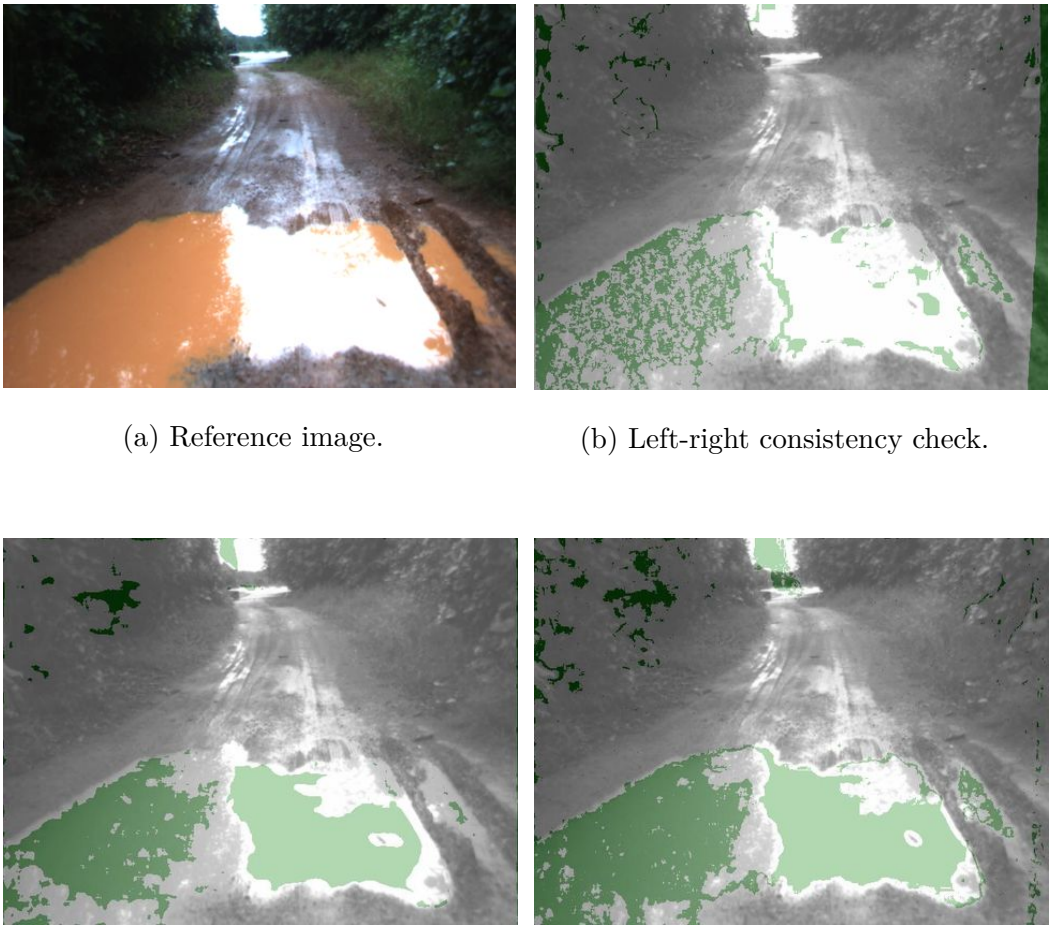
(c) Entropy.



(d) Winner margin.

FIGURE 6.8: Result I: elimination of uncertainty.

5.4 in 0.1 increments and then θ_{+1} from 5.4 to 2.1 in equal decrements, we obtain the curves shown in Figure 6.11. The two plots show that for any given instance, the sub-pixel estimate of parabolic fitting is biased towards the integer disparity, $d = 2$ and thereby indicates that it is prone to the pixel locking effect to a greater extent. However, this experiment alone is insufficient to qualify Gaussian fitting as the best method for our purpose. Therefore, we perform a separate experiment using the DSI of a pair of field images to find out the error characteristics of the two methods. For each correlation function passing the uncertainty test, the location of its actual extremum is estimated by fitting a smooth and continuous cubic spline over the entire correlation function. We assume the outcome of this operation to



(a) Reference image.

(b) Left-right consistency check.

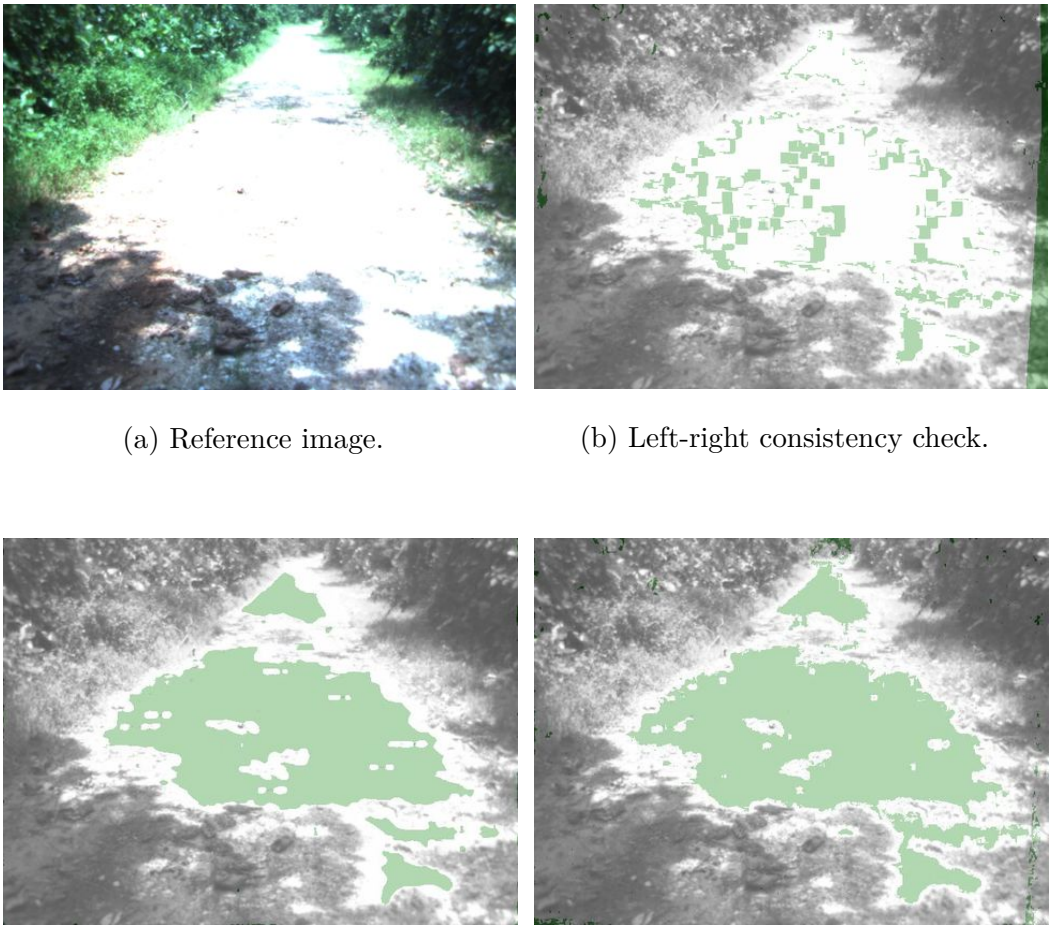
(c) Entropy.

(d) Winner margin.

FIGURE 6.9: Result II: elimination of uncertainty.

be a close approximation to the ground truth sub-pixel estimate. As discussed previously, parabolic and Gaussian fitting are performed over the observed extremum and its neighboring correlation values. Figure 6.12 shows the probability distributions of the absolute errors with reference to the approximated ground truth; the error distribution means are 0.019 and 0.024 for parabolic and Gaussian fitting respectively. This attests that the overall performance of parabolic fitting is better despite being affected by the pixel locking effect. Therefore, it is favored over Gaussian fitting in our final implementation.

The purpose of sub-pixel interpolation is to reduce the resulting stereo reconstruction error during the inverse mapping from a 2D image plane to 3D domain. To



(a) Reference image.

(b) Left-right consistency check.

(c) Entropy.

(d) Winner margin.

FIGURE 6.10: Result III: elimination of uncertainty.

analyze this situation, we capture stereo image pairs of a vehicle stationed in front of the UGV at different distances. A robust measurement on actual distance is obtained using a laser range finder, and the results of stereo reconstruction are compared against it. As expected, the stereo reconstruction error increases with distance for both pixel and parabolic sub-pixel precision estimates. However, the error associated with sub-pixel method is relatively lower, especially at large distances, as observed in Figure 6.13.

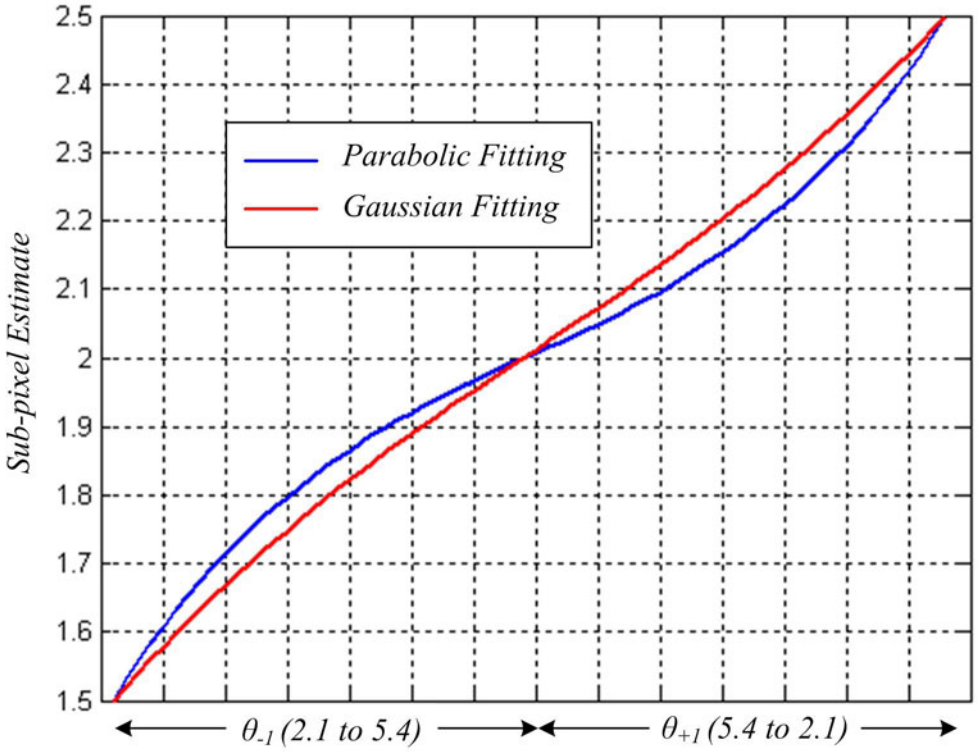


FIGURE 6.11: Pixel locking effect.

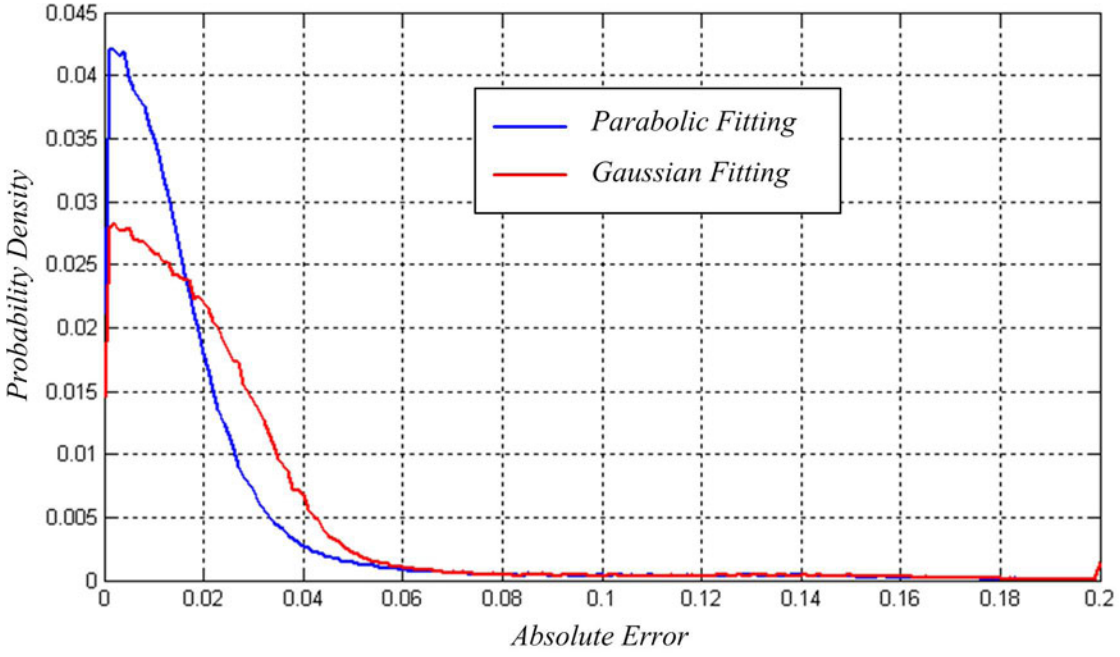


FIGURE 6.12: Sub-pixel estimation error distributions: parabolic vs. Gaussian fitting.

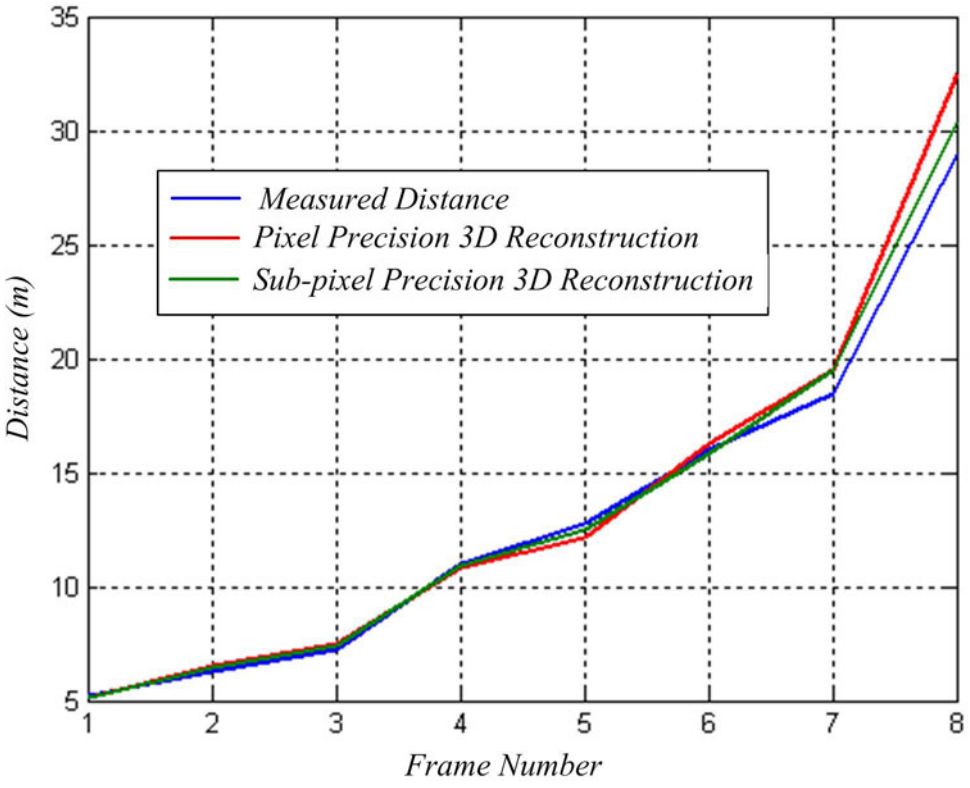


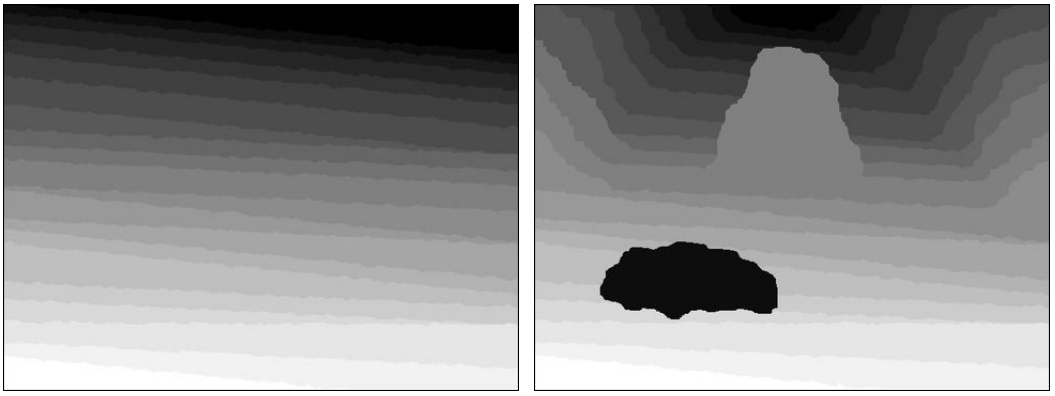
FIGURE 6.13: Accuracy of 3D reconstruction.

6.3 Obstacle Detection Algorithm Evaluation

6.3.1 Ground Plane Modeling

Lateral Ground Profile Estimation

To evaluate the gradient histogram and median absolute deviation methods described in Section 5.3.2, we utilize simulated ground truth disparity maps. It is clear that occlusion of the ground plane has a direct impact on the accuracy of the modeled ground plane. To bring this aspect into play, we repeat our analysis for the two disparity maps shown in Figure 6.14; in the current context we call these empty terrain and populated terrain, respectively. The fundamental difference between these simulations and that shown in Figure 6.1 is the variation of lateral gradients over disparity. For a large part, the simulation uses 0.1 or 0.05 as the lateral gradient while zero is used for a single instance. Even though



(a) Empty terrain. (b) Populated terrain.

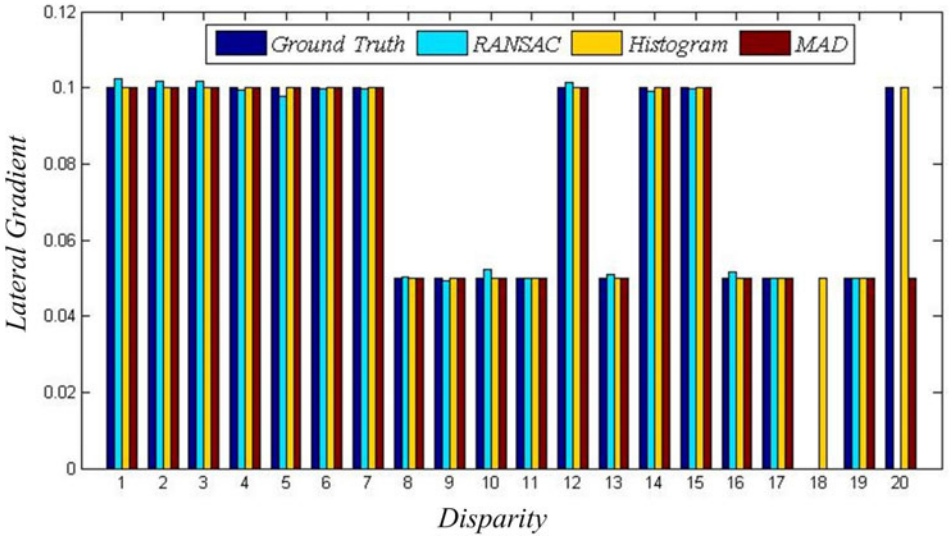
FIGURE 6.14: Input disparity maps to lateral ground profile estimation.

we rarely experience this kind of disparity maps in reality, we also realize that a similar occurrence could lead to unexpected errors. For comparison purposes the lateral gradient of each disparity is also computed using RANSAC line fitting. The outcomes of this analysis are depicted in Figure 6.15. Our observations are as follows:

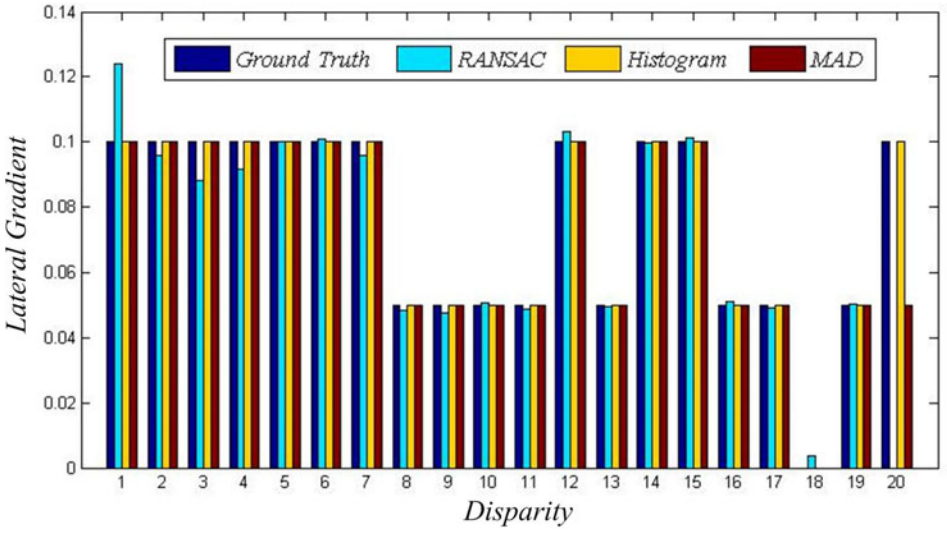
- A large majority of outputs closely follow the ground truth for the simulated empty terrain. This is expected since sampled ground pixels are uncontaminated by non-ground pixels.
- For the populated terrain, gradient histogram and median absolute deviation methods outperform RANSAC line fitting.
- The zero gradient at disparity 18 occurs only once. Therefore it does not make a significant contribution to the gradient histogram and causes a failure in the empty terrain case. (Theoretically, this should remain valid for the populated terrain too. However, in this particular case, zero gradient coincidentally becomes prominent enough when the gradient sample contribution from the ground plane is reduced by occlusion).

- Median absolute deviation method fails at disparity 20 due to the instability resulted by lack of gradient samples.

From this analysis and observations it can be inferred that the gradient histogram method behaves as intended when the histogram is completely characterized by one or few recurring bins; it is unable to detect locally isolated gradient variations. On the other hand, since median absolute deviation operates on each integer disparity



(a) Input: disparity map of an empty terrain.



(b) Input: disparity map of a populated terrain.

FIGURE 6.15: Lateral ground profile estimation.

independently, it is robust against such local variations. However, since it does not incorporate the overall trend of the ground plane, it might produce erroneous results when the confidence level of input data is low. The information available to us at this point is insufficient to choose one method over the other; this decision will be made later on by evaluating the ground reconstruction error of the two methods.

Longitudinal Ground Profile Estimation

The empty and populated terrain simulations in the previous section are used here. Since the actual lateral ground profile is known for these disparity maps, it is possible to construct the corresponding minimum error v-disparity image without bringing gradient calculation into the picture. In Section 5.3.3, we discussed two methods that can be used to estimate the longitudinal ground profile of a v-disparity image. The RMS errors incurred by applying these methods to the minimum error v-disparity image are shown in Figure 6.16. These error plots reflect the following:

- The overall error of the constraint satisfaction method is lower than the piecewise linear method for both empty and populated terrains.
- The piecewise linear method largely deviates from the ground truth at far distances (i.e. small disparity values).
- Both methods demonstrate relatively large error between disparity 4 to 10; this is a result of the occlusion of ground plane caused by the fronto-parallel obstacle at disparity 10.

Due to better overall performance, the constraint satisfaction method is preferred for our ground plane modeling algorithm.

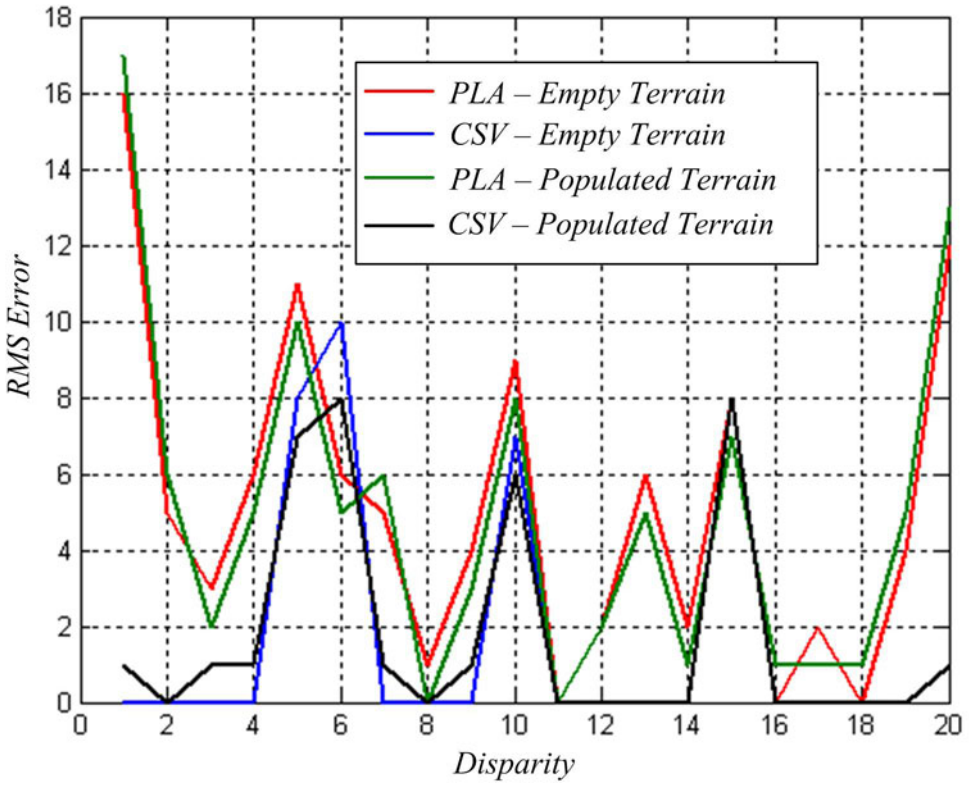


FIGURE 6.16: Longitudinal ground profile estimation error. Key to abbreviations: PLA - piecewise linear approximation, CSV - constraint satisfaction vector.

Overall Reconstruction Error

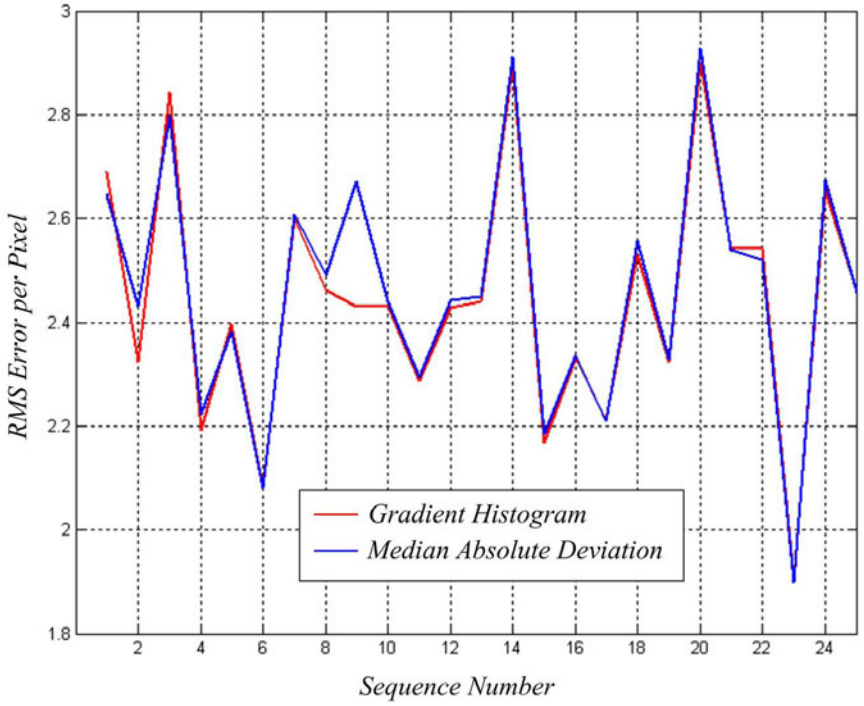
The main purpose of this effort is to finalize the lateral ground profile estimation method, which was left undetermined during our previous analyses. To begin with, we select 25 frames which largely portray the ground surface (Figure 6.17) and manually segment a ground mask for each instance. If we assume the stereo correspondences of this image subset to be of sufficient accuracy, we may in turn consider it a close approximation to the actual ground plane disparity of the masked area. With this information in hand, we proceed to independently reconstruct the ground plane using gradient histogram and median absolute deviation methods; for both cases the longitudinal ground profile is estimated using constraint satisfaction vector. The RMS error variations between actual and reconstructed ground disparities, calculated within the area of the ground mask, are shown in Figure 6.18(a). This evaluation confirms that the gradient histogram

FIGURE 6.17: Ground plane masking.

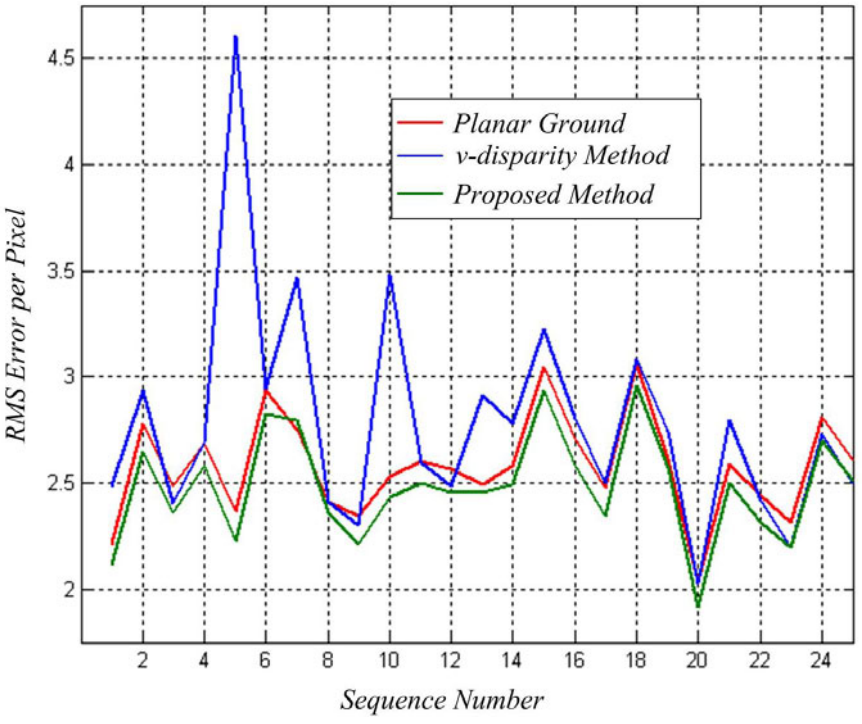
method performs marginally better than its counterpart and hence is the chosen method for our proposed ground plane modeling algorithm. Ultimately, a similar reconstruction error calculation is performed for planar ground approximation and original v-disparity methods. The corresponding error comparison depicted in Figure 6.18(b) demonstrates that the proposed method consistently outperforms the other two methods.

6.3.2 Obstacle Detection

Regardless of its complexity, a computer simulation is unable to comprehensively emulate the subtle dynamics of an actual environment. Therefore, the evaluation of the proposed obstacle detection algorithm is incomplete until it is thoroughly tested and qualified with real world data. In order to enable an unbiased comparison between different ground plane modeling methods, no additional image



(a) Lateral ground profile estimation methods.



(b) Traditional methods vs. proposed method.

FIGURE 6.18: Error comparison: ground geometry reconstruction.

processing operations (e.g., blob filtering) are performed on the resulting image domain obstacle maps. We also realize that it is hard to predefine the size and shape of a blob filter, when the types of obstacles to be detected are unconstrained. The illustrations in this section use the following color scheme:

- Image domain obstacle maps are superimposed on the corresponding gray scale image in red.
- The green line in an obstacle map represents the ground horizon; depending on the algorithm used, this can correspond to zero disparity or the minimum detectable ground disparity.
- The pixel intensity of the world coordinate map is proportional to the average height of an occupancy grid; black represents the ground level.
- The red dot in the world coordinate map marks the location of the vehicle.

In the first analysis, we repeatedly assess the performance of the proposed algorithm for obstacles located at different distances from the vehicle. The outcomes for a relatively large obstacle (vehicle object), a moderate size obstacle (human object) and a small obstacle (cardboard box) are depicted in Figures 6.19, 6.20 and 6.21 respectively. This analysis shows us that the detectability of an obstacle is directly related to its size. An object as large as a vehicle can be easily detected at distances as far as 50m, while a small obstacle such as a box might go undetected even at 15m. However, we believe that the severity of this shortcoming is compensated by the accurate detection of small obstacles when the vehicle moves closer to them.

Figures 6.22 and 6.23 directly compare the obstacle detection performance of the proposed algorithm with that of planar ground approximation and v-disparity methods for few selected instances. When the ground plane is relatively flat and the vehicle is stationary, we would expect all three algorithms to generate obstacle

maps of comparable accuracy. When the flat earth geometry does not hold true, planar ground approximation could yield false obstacle classifications or a ground horizon as demonstrated in Figure 6.22. On the other hand, the most common failure mode of the v-disparity method is a coupled rolling and yawing of the vehicle. The consequent image in-plane rotation introduces a large lateral disparity gradient, which in turn leads to false positives as illustrated in Figure 6.23.

The obstacle detection algorithm we propose here is not without its failure modes. Sometimes, a widely dispersed object, such as vegetation, might possess similar geometric properties to that we seek in a ground plane. In such situations, an erroneous modeling of ground profiles may eventually result in false negatives as shown in Figures 6.24(a) and 6.24(b). Also it is important to note that the algorithm we propose here does not propagate the ground plane model over time and starts from scratch for each pair of stereo images. Therefore, it is absolutely necessary for the ground plane to be at least partially visible for our algorithm function as expected. When this requirement is not met, it can lead to errors as shown in Figures 6.24(c) and 6.24(d).

More obstacle detection results of our algorithm are separately attached in Appendix C.

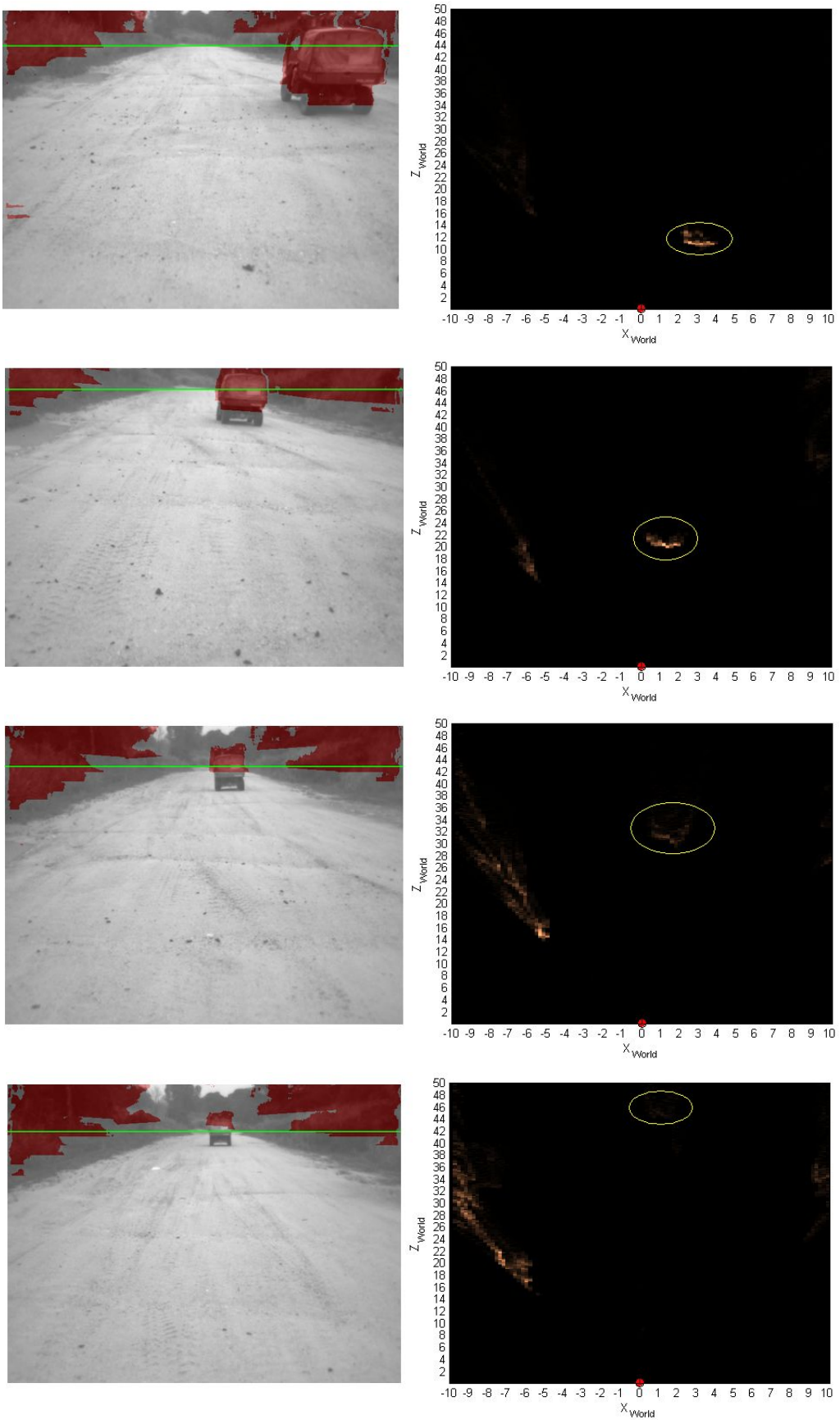


FIGURE 6.19: Detection of a vehicle object at varying distances: left - image domain detection, right - world coordinate frame representation.

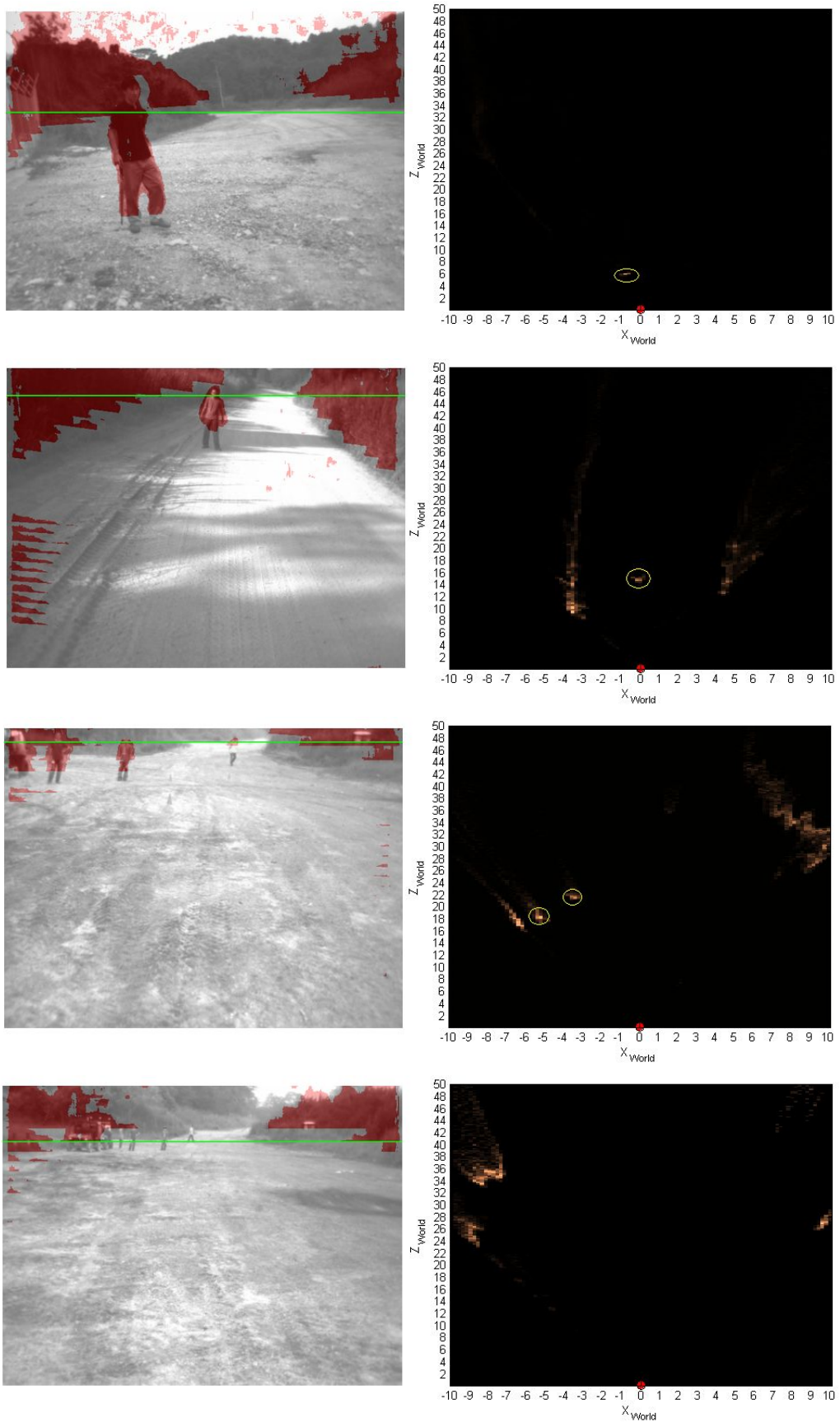


FIGURE 6.20: Detection of a human object at varying distances: left - image domain detection, right - world coordinate frame representation.

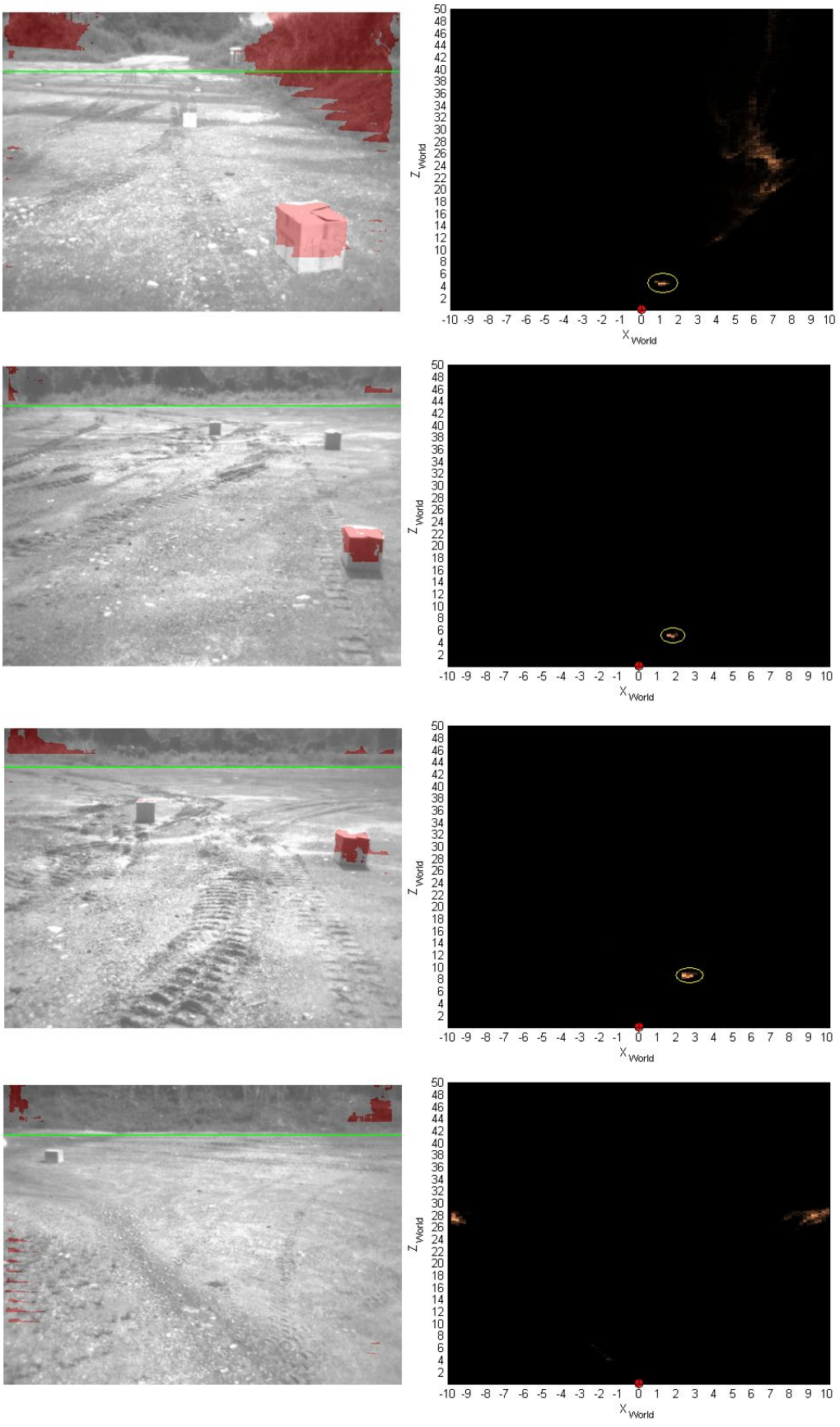


FIGURE 6.21: Detection of a cardboard box at varying distances: left - image domain detection, right - world coordinate frame representation.

FIGURE 6.22: Performance comparison I: left - reference image, center - planar ground approximation, right - proposed algorithm.

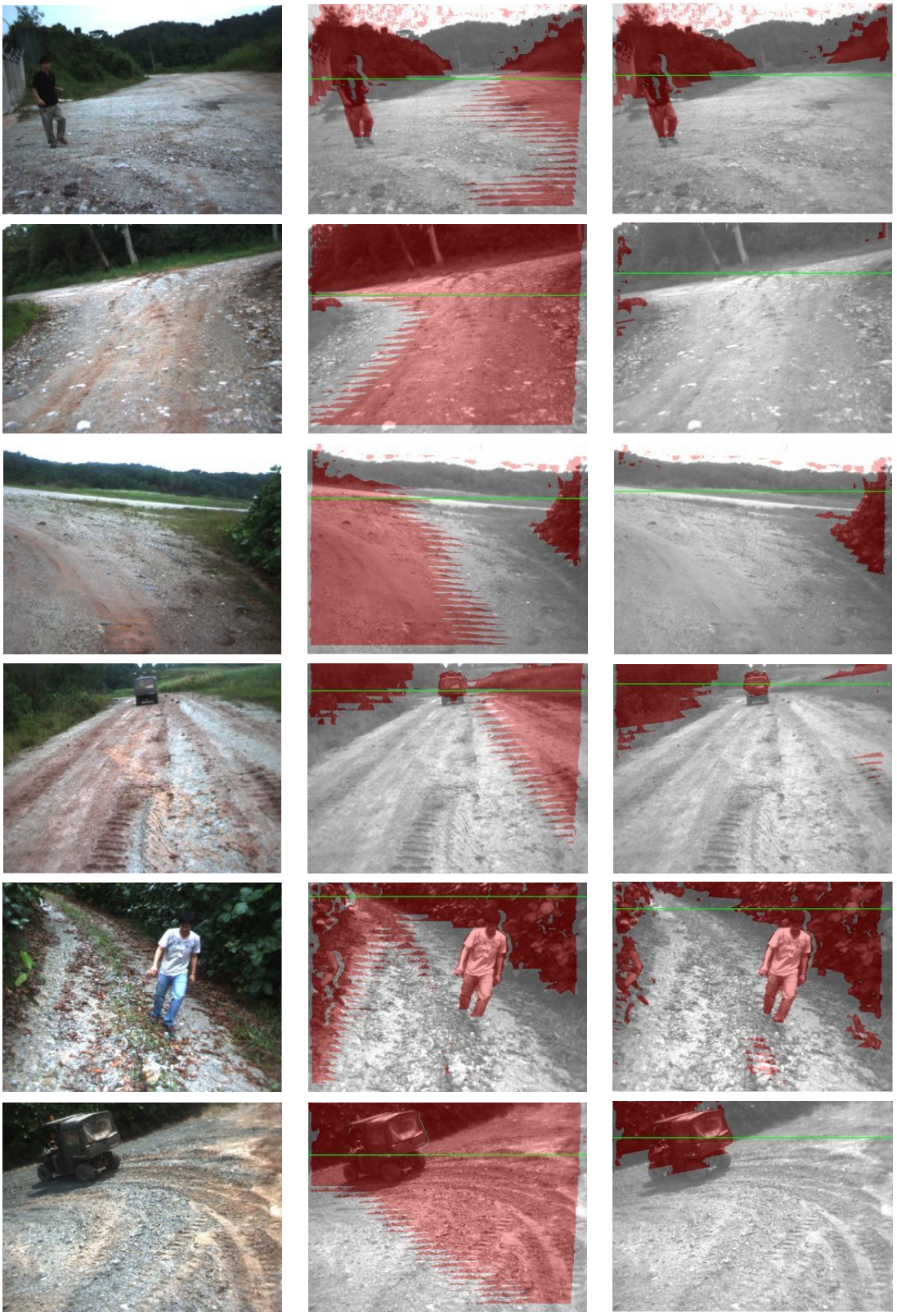
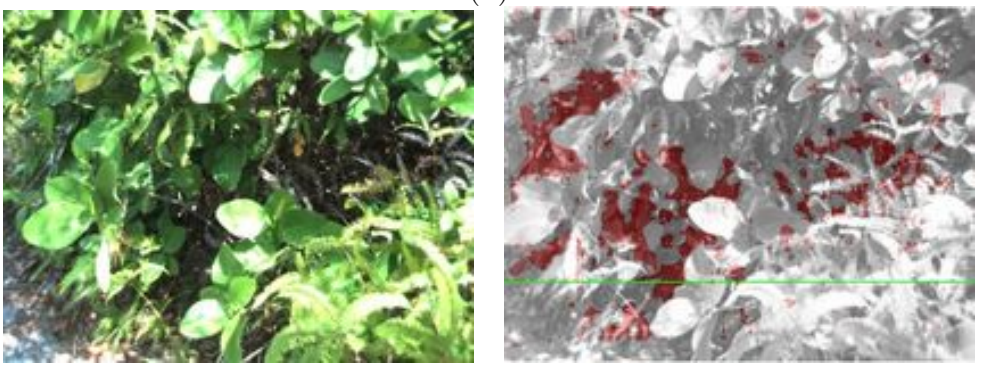


FIGURE 6.23: Performance comparison II: left - reference image, center - v-disparity algorithm, right - proposed algorithm.

(a)

(b)



(c)

(d)

FIGURE 6.24: Obstacle detection errors: left - reference image, right - image domain obstacle detection.

Chapter 7

Conclusion and Future Work

In this thesis, we introduced a stereo vision based obstacle detection and localization method for outdoor autonomous navigation. The presented algorithm is particularly well designed to function robustly under semi-structured rural conditions, where the road geometry is assumed to closely follow a piecewise planar model. Both parametric ground plane model estimation and subsequent obstacle detection are carried out in dense disparity space. The final algorithm is thoroughly tested and successfully deployed in an intelligent unmanned vehicle.

Since the proposed obstacle detection algorithm is entirely dependent on stereo disparity, errors occurring at the stereo matching stage will inevitably propagate to the outcome of obstacle detection. For this reason, establishing accurate stereo correspondences is of utmost importance in our work. Considering the fact that this is not the core concentration of our research, a suitable solution was sought by assessing test image sequences against a number of existing stereo algorithms. Our ultimate choice is an integration of familiar concepts such as the census transform, SAD correlation, parabolic fitting and winner margin to one coherent stereo correspondence algorithm. It was comprehensively evaluated using random dot images of known ground truth disparity and real test images to ensure that the accuracy

and precision are in line with our requirements. These analyses confirmed that our stereo algorithm outperforms a majority of other real time methods in the same category.

The ground plane modeling algorithm, which was discussed in Section 5.3, is the main contribution of our work. It decomposes the piecewise planar approximation into two stages: first the lateral gradient of the ground plane is computed at each disparity using a histogram analysis, and then it is followed by a constrained optimization procedure to unveil the longitudinal ground profile. This modular approach yields greater perseverance of ground details while effectively attenuating the contribution of obstacles. At the same time, it allows easier identification and mitigation of possible sources of error during the reconstruction of the ground plane. Even though an effort has been made to make the algorithm self-adaptive as far as possible, some parameters still have to manually set to reduce the computational complexity. The experimental results testify that the proposed algorithm constantly exceeds the performance of candidate GPOD methods such as planar ground approximation and the v-disparity method.

The empirical evidence demonstrated that scene structures that are similar to the ground plane in a geometric sense, may give rise to false negatives. Also more often than not, water puddles could not be distinguished from the rest of the ground plane due to stereo matching ambiguities. Ideally, we would want to avoid water puddles as they might present occasional hazards. However, we also realize that it is difficult to resolve all these shortcomings using geometric properties alone. One possible remedy would be to incorporate additional visual cues such as color and texture and extend the capability of our algorithm from obstacle detection to an extensive traversability evaluation. For this purpose we intend to use research that have been conducted in relation to the same project at the VIP lab; they include an intrinsic color space road classifier and a water puddle detection algorithm using local binary patterns. The proposed algorithm also requires a fair portion of the

ground plane to be visible in order to build an accurate model. This problem can be alleviated by tracking the ground plane model over time, rather than the current approach of building a new model from scratch for each pair of images. Moving one step further, we may combine successive world coordinate maps to implement a simultaneous localization and mapping (SLAM) algorithm. Accelerating the execution speed by means of parallel processing is amongst other future concerns.

Bibliography

[1] N. Nilsson, "A Mobile Automaton: An Application of Artificial Intelligence Techniques," in *Proceedings of the 1st International Joint Conference on Artificial Intelligence*, 1969, pp. 509–520.

[2] R. Leighty, "DARPA ALV (Autonomous Land Vehicle) Summary," 1986.

[3] C. Thorpe, R. C. Coulter, M. Hebert, T. Jochem, D. Langer, D. Pomerleau, J. Rosenblatt, W. Ross, and A. T. Stentz, "Smart Cars: The CMU Navlab," in *Proceedings of WORLD MED93*, October 1993.

[4] M. Xie, L. Trassoudaine, J. Alizon, M. Thonnat, and J. Gallice, "Active and intelligent sensing of road obstacles: Application to the European Eureka-PROMETHEUS project," in *Fourth International Conference on Computer Vision*, Berlin , Germany, May 1993, pp. 616–623.

[5] C. Shoemaker and J. Bornstein, "The Demo III UGV program: A testbed for autonomous navigation research," in *Proceedings of Intelligent Control (ISIC)*, 1998, pp. 644–651.

[6] A. Broggi, M. Bertozzi, A. Fascioli, C. Bianco, and A. Piazzi, "The ARGO autonomous vehicles vision and control systems," *International Journal of Intelligent Control and Systems*, vol. 3, no. 4, pp. 409–441, 1999.

[7] "The 2004 Grand Challenge," "http://www.darpa.mil/grandchallenge04/".

[8] "The 2005 Grand Challenge," "http://www.darpa.mil/grandchallenge05/".

[9] S. Thrun, M. Montemerlo, H. Dahlkamp, D. Stavens, A. Aron, J. Diebel, P. Fong, J. Gale, M. Halpenny, G. Hoffmann *et al.*, "Stanley: The robot that won the DARPA Grand Challenge," *The 2005 DARPA Grand Challenge*, pp. 1–43, 2007.

[10] "The 2007 Grand Challenge," "http://www.darpa.mil/grandchallenge/".

[11] A. Discant, A. Rogozan, C. Rusu, and A. Bensrhair, "Sensors for Obstacle Detection - A Survey," in *30th International Spring Seminar on Electronics Technology*, Cluj-Napoca, Romania, May 2007, pp. 100–105.

[12] L. Lorigo, R. Brooks, and W. Grimson, "Visually-guided obstacle avoidance in unstructured environments," in *International Conference on Intelligent Robots and Systems*, Grenoble, France, September 1997, pp. 373–379.

[13] I. Ulrich and I. Nourbakhsh, "Appearance-based obstacle detection with monocular color vision," in *Proceedings of the National Conference on Artificial Intelligence*, Austin, Texas, August 2000, pp. 866–871.

[14] N. Pears and B. Liang, "Ground plane segmentation for mobile robot visual navigation," in *International Conference on Intelligent Robots and Systems*, Maui, USA, October 2001, pp. 1513–1518.

[15] P. Batavia and S. Singh, "Obstacle detection using adaptive color segmentation and color stereo homography," in *IEEE International Conference on Robotics and Automation*, Seoul, Korea, May 2001, pp. 705–710.

[16] C. Rasmussen, "Combining laser range, color, and texture cues for autonomous road following," in *IEEE International Conference on Robotics and Automation*, Seoul, Korea, May 2002, pp. 4320–4325.

[17] C. Dima, N. Vandapel, and M. Hebert, "Classifier fusion for outdoor obstacle detection," in *IEEE International Conference on Robotics and Automation*, New Orleans, LA, May 2004, pp. 665–671.

[18] H. Kong, J.-Y. Audibert, and J. Ponce, "Vanishing point detection for road detection," in *IEEE International Conference on Computer Vision and Pattern Recognition*, Miami, FL, June 2009, pp. 96–103.

[19] J. Alvarez, T. Gevers, and A. Lopez, "3D Scene priors for road detection," in *IEEE International Conference on Computer Vision and Pattern Recognition*, San Francisco, CA, June 2010, pp. 57–64.

[20] M. Ilic, S. Masciangelo, and E. Pianigiani, "Ground plane obstacle detection from optical flow anomalies: a robust and efficient implementation," in *Proceedings of IEEE Intelligent Vehicle Symposium*, Paris, France, October 1994, pp. 333–338.

[21] T. Camus, D. Coombs, M. Herman, and T. Hong, "Real-time single-workstation obstacle avoidance using onlywide-field flow divergence," in *Pattern Recognition, 1996., Proceedings of the 13th International Conference on*, Vienna, Austria, August 1996, p. 323.

[22] C. Demonceaux and D. Kachi-Akkouche, "Robust obstacle detection with monocular vision based on motion analysis," in *Proceedings of IEEE Intelligent Vehicles Symposium*, 2004, pp. 527–532.

[23] K. Imiya and R. Hirota, "Motion-Based Template Matching for Obstacle Detection," *Journal of Advanced Computatioanl Intelligence and Intelligent Informatics*, vol. 8, no. 5, 2004.

[24] Y. Shen, X. Du, and J. Liu, "Monocular Vision Based Obstacle Detection for Robot Navigation in Unstructured Environment," in *Proceedings of the 4th*

international symposium on Neural Networks, Nanjing, China, June 2007, pp. 714–722.

[25] Y. Zheng, D. Jones, S. Billings, J. Mayhew, and J. Frisby, "Switcher: A stereo algorithm for ground plane obstacle detection," *Image and Vision Computing*, vol. 8, no. 1, pp. 57–62, February 1990.

[26] F. Ferrari, E. Grosso, G. Sandini, and M. Magrassi, "A stereo vision system for real time obstacle avoidance in unknown environment," in *IEEE International Workshop on Intelligent Robots and Systems*, Ibaraki, Japan, July 1990, pp. 703–708.

[27] N. Chumerin and M. Van Hulle, "Ground plane estimation based on dense stereo disparity," in *The Fifth International Conference on Neural Networks and artificial intelligence*, Minsk, Belarus, May 2008, pp. 209–213.

[28] S. Se and M. Brady, "Ground plane estimation, error analysis and applications," *Robotics and Autonomous Systems*, vol. 39, no. 2, pp. 59–71, May 2002.

[29] Z. Zhang, R. Weiss, and A. Hanson, "Obstacle detection based on qualitative and quantitative 3Dreconstruction," *IEEE Transactions on Pattern Analysis and Machine Intelligence*, vol. 19, no. 1, pp. 15–26, January 1997.

[30] R. Labayrade, D. Aubert, and J. Tarel, "Real time obstacle detection in stereovision on non flat road geometry through "v-disparity" representation," in *Proceedings of IEEE Intelligent Vehicle Symposium*, Versailles, France, June 2002, pp. 646–651.

[31] A. Broggi, C. Caraffi, R. Fedriga, and P. Grisleri, "Obstacle detection with stereo vision for off-road vehicle navigation," in *IEEE Computer Society Conference on Computer Vision and Pattern Recognition*, San Diego, CA, June 2005, pp. 65–65.

[32] B. Hummel, S. Kammel, T. Dang, C. Duchow, and C. Stiller, "Vision-based path-planning in unstructured environments," in *Proceedings of IEEE Intelligent Vehicle Symposium*, Tokyo, Japan, June 2006, pp. 176–181.

[33] W. Abd-Almageed, M. Hussein, and M. Abdelkader, "Real-time human detection and tracking from mobile vehicles," in *IEEE Intelligent Transportation Systems Conference*, Seattle, Washington, September 2007, pp. 149–154.

[34] N. Soquet, D. Aubert, and N. Hautiere, "Road segmentation supervised by an extended v-disparity algorithm for autonomous navigation," in *Proceedings of IEEE Intelligent Vehicle Symposium*, Istanbul, Turkey, June 2007, pp. 160–165.

[35] S. Kodagoda, G. Dong, C. Yan, and S. Ong, "Off-Road Obstacle Detection with Robust Parametric Modeling of the Ground Stereo Geometry," in *Proceedings of the Fourteenth IASTED International Conference on Robotics and Applications*, Cambridge, Massachusetts, November 2009, pp. 343–350.

[36] S. Nedevschi, R. Danescu, D. Frentiu, T. Marita, F. Oniga, C. Pocol, T. Graf, and R. Schmidt, "High accuracy stereovision approach for obstacle detection on non-planar roads," *IEEE Inteligent Engineering Systems*, pp. 211–216, 2004.

[37] G. Giralt and L. Boissier, "The French Planetary Rover Vap: Concept And Current Developments," in *IEEE International Conference on Intelligent Robots and Systems*, Raleigh, NC, July 1992, pp. 1391–1398.

[38] L. Matthies, "Stereo vision for planetary rovers: Stochastic modeling to near real-time implementation," *International Journal of Computer Vision*, vol. 8, no. 1, pp. 71–91, 1992.

[39] W. Van der Mark, F. Groen, and J. van den Heuvel, "Stereo based navigation in unstructured environments," in *IEEE Instrumentation and Measurement Technology Conference*, Budapest, Hungary, May 2001, pp. 2038–2043.

[40] G. Dubbelman, W. van der Mark, J. van den Heuvel, and F. Groen, "Obstacle detection during day and night conditions using stereo vision," in *IEEE International Conference on Intelligent Robots and Systems*, San Diego, California, October 2007, pp. 109–116.

[41] R. Hadsell, J. Bagnell, D. Huber, and M. Hebert, "Accurate rough terrain estimation with space-carving kernels," in *Proceedings of Robotics: Science and Systems Conference*, Seattle, WA, June 2009.

[42] M. Vergauwen, M. Pollefeys, and L. J. V. Gool, "A Stereo Vision System for Support of Planetary Surface Exploration," in *Proceedings of the Second International Workshop on Computer Vision Systems*. London, UK: Springer-Verlag, 2001, pp. 298–312.

[43] C. Olson, L. Matthies, J. Wright, R. Li, and K. Di, "Visual terrain mapping for Mars exploration," *Computer Vision and Image Understanding*, vol. 105, no. 1, pp. 73–85, 2007.

[44] P. Furgale, T. Barfoot, and N. Ghafoor, "Rover-Based Surface and Subsurface Modeling for Planetary Exploration," 2009.

[45] R. Manduchi, A. Castano, A. Talukder, and L. Matthies, "Obstacle detection and terrain classification for autonomous off-road navigation," *Autonomous Robots*, vol. 18, no. 1, pp. 81–102, January 2005.

[46] W. van der Mark, J. van den Heuvel, and F. Groen, "Stereo based obstacle detection with uncertainty in rough terrain," in *2007 IEEE Intelligent Vehicles Symposium*, Istanbul, Turkey, June 2007, pp. 1005–1012.

[47] P. Santana, P. Santos, L. Correia, and J. Barata, "Cross-country obstacle detection: Space-variant resolution and outliers removal," in *IEEE International Conference on Intelligent Robots and Systems*, Nice, France, September 2008, pp. 1836–1841.

[48] "Polaris Ranger," "http://www.polarisindustries.com".

[49] "FireWire CCD Stereo Vision Cameras by Point Grey," "http://www.ptgrey.com/products/stereo.asp".

[50] C. Wheatstone, "On some remarkable, and hitherto unobserved, Phenomena of Binocular Vision," *Philosophical Transactions of the Royal Society of London*, vol. 128, pp. 371–394, June 1838.

[51] P. d'Angelo, "3D scene reconstruction by integration of photometric and geometric methods," Ph.D. dissertation, 2007.

[52] J. Weng, P. Cohen, and M. Herniou, "Calibration of Stereo Cameras Using a Nonlinear Distortion Model," in *Proceedings of the 10th International Conference on Pattern Recognition*, Atlantic Citys, USA, June 1990, pp. 246–253.

[53] R. Tsai, "An efficient and accurate camera calibration technique for 3d machine vision," in *Proceedings of the 3rd International Conference on Computer Vision and Pattern Recognition*, Miami, FL, June 1986, pp. 364–374.

[54] Z. Zhang, "A Flexible New Technique for Camera Calibration," *IEEE Transactions on Pattern Analysis and Machine Intelligence*, vol. 22, pp. 1330–1334, 2000.

[55] A. Gruen and T. S. Huang, *Calibration and Orientation of Cameras in Computer Vision*. Secaucus, NJ: Springer-Verlag New York, Inc., 2001.

[56] J.-Y. Bouguet, "Camera Calibration Toolbox for Matlab," "http://www.vision.caltech.edu/bouguetj/calib_doc/".

[57] R. Gonzalez and R. E. Woods, *Digital Image Processing.* Prentice Hall PTR, 2002.

[58] R. Zabih and J. Woodfill, "Non-parametric Local Transforms for Computing Visual Correspondence," in *Proceedings of the Third European Conference on Computer Vision*, Stockholm, Sweden, 1994, pp. 151–158.

[59] D. Scharstein and R. Szeliski, "A taxonomy and evaluation of dense two-frame stereo correspondence algorithms," *International journal of computer vision*, vol. 47, no. 1, pp. 7–42, April 2002.

[60] A. Bobick and S. Intille, "Large occlusion stereo," *International Journal of Computer Vision*, vol. 33, no. 3, pp. 181–200, 1999.

[61] T. Kanade and M. Okutomi, "A stereo matching algorithm with an adaptive window: Theory andexperiment," in *IEEE International Conference on Robotics and Automation*, Sacramento, California, April 1991, pp. 1088–1095.

[62] O. Veksler, "Stereo matching by compact windows via minimum ratio cycle," in *Eighth International Conference on Computer Vision*, Vancouver, British Columbia, Canada, July 2001, pp. 540–547.

[63] J. Shah, "A nonlinear diffusion model for discontinuous disparity and half-occlusions in stereo," in *IEEE Conference on Computer Vision and Pattern Recognition*, New York, NY, June 1993, pp. 34–34.

[64] D. Scharstein and R. Szeliski, "Stereo matching with nonlinear diffusion," *International Journal of Computer Vision*, vol. 28, no. 2, pp. 155–174, 1998.

[65] V. Kolmogorov and R. Zabih, "Computing visual correspondence with occlusions using graph cuts," in *8th IEEE International Conference on Computer Vision*, Vancouver, BC, July 2001, pp. 508–515.

[66] P. Belhumeur, "A Bayesian approach to binocular steropsis," *International Journal of Computer Vision*, vol. 19, no. 3, pp. 237–260, 1996.

[67] S. Jian, Z. Nan-Ning, and S. Heung-Yeung, "Stereo matching using belief propagation," *IEEE Transactions on Pattern Analysis and Machine Intelligence*, vol. 25, no. 7, pp. 787–800, 2003.

[68] V. Vineet and P. Narayanan, "CUDA cuts: Fast graph cuts on the GPU," in *IEEE Conference on Computer Vision and Pattern Recognition Workshops*, Anchorage, Alaska, June 2008, pp. 1–8.

[69] G. Minglun and Y. Yee-Hong, "Near real-time reliable stereo matching using programmable graphics hardware," in *IEEE Conference on Computer Vision and Pattern Recognition*, San Diego, CA, June 2005, pp. 924–931.

[70] Q. Yang, L. Wang, R. Yang, S. Wang, M. Liao, and D. Nister, "Real-time global stereo matching using hierarchical belief propagation," in *The British Machine Vision Conference*, Edinburgh, UK, September 2006, pp. 989–998.

[71] K. Takita, M. Muquit, T. Aoki, and T. Higuchi, "A sub-pixel correspondence search technique for computer vision applications," *IEICE Transactions on Fundamentals of Electronics Communications and Computer Sciences*, vol. 87, pp. 1913–1923, 2004.

[72] Y. Sugii, S. Nishio, T. Okuno, and K. Okamoto, "Accurate Sub-pixel Analysis on PIV using Gradient Method," *Journal of the Visualization Society of Japan*, vol. 20, 2000.

[73] W. Yu and B. Xu, "A sub-pixel stereo matching algorithm and its applications in fabric imaging," *Machine Vision and Applications*, vol. 20, no. 4, pp. 261–270, 2009.

[74] A. Fusiello, V. Roberto, and E. Trucco, "Symmetric stereo with multiple windowing," *International Journal of Pattern Recognition and Artificial Intelligence*, vol. 14, no. 8, pp. 1053–1066, 2000.

[75] H. Nobach and M. Honkanen, "Two-dimensional Gaussian regression for subpixel displacement estimation in particle image velocimetry or particle position estimation in particle tracking velocimetry," *Experiments in fluids*, vol. 38, no. 4, pp. 511–515, 2005.

[76] M. Shimizu and M. Okutomi, "Sub-pixel estimation error cancellation on area-based matching," *International Journal of Computer Vision*, vol. 63, no. 3, pp. 207–224, 2005.

[77] M. Hershenson, *Visual space perception: A primer*. The MIT Press, 1999.

[78] "OpenCV Wiki," "http://opencv.willowgarage.com/wiki/".

[79] I. Corporation, "Intel IPP - Intel Software Network," "http://software.intel.com/en-us/intel-ipp/".

[80] "The OpenMP API Specification for Parallel Programming," "http://openmp.org/wp/".

[81] R. Szeliski, "Cooperative algorithms for solving random-dot stereograms," 1986.

[82] M. Hannah, "Computer matching of areas in stereo images," Ph.D. dissertation, 1974.

[83] "Middlebury Stereo Vision Page," "http://vision.middlebury.edu/stereo/".

[84] M. Fischler and R. Bolles, "Random sample consensus: A paradigm for model fitting with applications to image analysis and automated cartography," *Communications of the ACM*, vol. 24, no. 6, pp. 381–395, 1981.

[85] W. Dumouchel and F. O'Brien, "Integrating a robust option into a multiple regression computing environment," pp. 41–48, 1991.

[86] P. Hough, "Method and means for recognizing complex patterns," 1962, uS Patent 3,069,654.

[87] A. Goldenshluger and A. Zeevi, "The hough transform estimator," *Annals of statistics*, vol. 32, no. 5, pp. 1908–1932, 2004.

[88] R. O. Duda and P. E. Hart, "Use of the Hough transformation to detect lines and curves in pictures," *Communications of the ACM*, vol. 15, no. 1, pp. 11–15, 1972.

Appendix A

Bumblebee Camera Specifications

Specification	Low-Res (640x480)	High-Res (1024x768)
Imaging Sensor	Two Sony 1/3" progressive scan CCD	
	ICX424 (648x488 max pixels)	ICX204 (1024x768 max pixels)
	7.4µm square pixels	4.65µm square pixels
Baseline	12cm	
Lens Focal Length	3.8mm with 70° HFOV or 6mm with 50° HFOV	
A/D Converter	Analog Devices 12-bit analog-to-digital converter	
Video Data Output	8, 16 and 24-bit digital data (see *Supported Data Formats* below)	
Frame Rates	48, 30, 15, 7.5, 3.75, 1.875 FPS	18, 15, 7.5, 3.75, 1.875 FPS
Interfaces	6-pin IEEE-1394 for camera control and video data transmission 4 general-purpose digital input/output (GPIO) pins.	
Voltage Requirements	8-32V	
Power Consumption	Less than 3W	
Gain	Automatic/Manual/One-Push Gain modes	
	0dB to 24dB	0dB to 24dB
Shutter	Automatic/Manual/One-Push Shutter modes	
	0.01ms to 66.63ms @15 FPS	0.01ms to 66.63ms @15 FPS
	Extended Shutter modes	
	0.01ms to 7900ms @ 15 FPS	0.01ms to 5200ms @ 15 FPS
Gamma	0.50 to 4.00	
Trigger Modes	DCAM v1.31 Trigger Modes 0, 1, 3, and 14	
Signal To Noise Ratio	Greater than 60dB at 0dB gain	
Dimensions	157mm x 36mm x 47.4mm	
Mass	342 grams	
Camera Specification	IIDC 1394-based Digital Camera Specification v1.31	
Emissions Compliance	Complies with CE rules and Part 15 Class A of FCC Rules	
Operating Temperature	Commercial grade electronics rated from 0° to 45°C	
Storage Temperature	-30° to 60°C	

FIGURE A.1: Camera specifications of the Bumblebee2.

Image Acquisition

Feature	Description
Automatic Synchronization	Multiple Bumblebee2's on the same 1394 bus automatically sync
Fast Frame Rates	Faster standard frame rates
Multiple Trigger Modes	Bulb-trigger mode, overlapped trigger/transfer
Color Conversion	On-camera conversion to YUV411, YUV422 and RGB formats
Image Processing	On-camera control of sharpness, hue, saturation, gamma, LUT
Embedded Image Info	Pixels contain frame-specific info (e.g. shutter, 1394 cycle time)

Camera and Device Control

Feature	Description
Frame Rate Control	Fine-tune frame rates for video conversion (e.g. PAL @ 24 FPS)
Strobe Output	Increased drive strength, configurable strobe pattern output
RS-232 Serial Port	Provides serial communication via GPIO TTL digital logic levels
Memory Channels	Non-volatile storage of camera default power-up settings
Temperature Sensor	Reports the temperature near the imaging sensor
Camera Upgrades	Firmware upgradeable in field via IEEE-1394 interface.

Calibration and Mechanics

Feature	Description
Lens System	High quality microlenses protected by removeable glass system
Accurate Pre-Calibration	For lens distortions and camera misalignments
Stereo Pair Alignment	Left and right images aligned to within 0.05[1] pixel RMS error
Calibration Retention	Minimizes loss of calibration due to shock and vibration

FIGURE A.2: Camera features of the Bumblebee2.

Calibration Parameter	Unit	Value
Baseline (b)	cm	12.019
Focal Length (f_p)	Pixels	811.104
Principal point(u_o, v_o)	Pixels	$(323.398, 246.096)$

TABLE A.1: Stereo rectified intrinsic calibration parameters (Note: image resolution = 640×480).

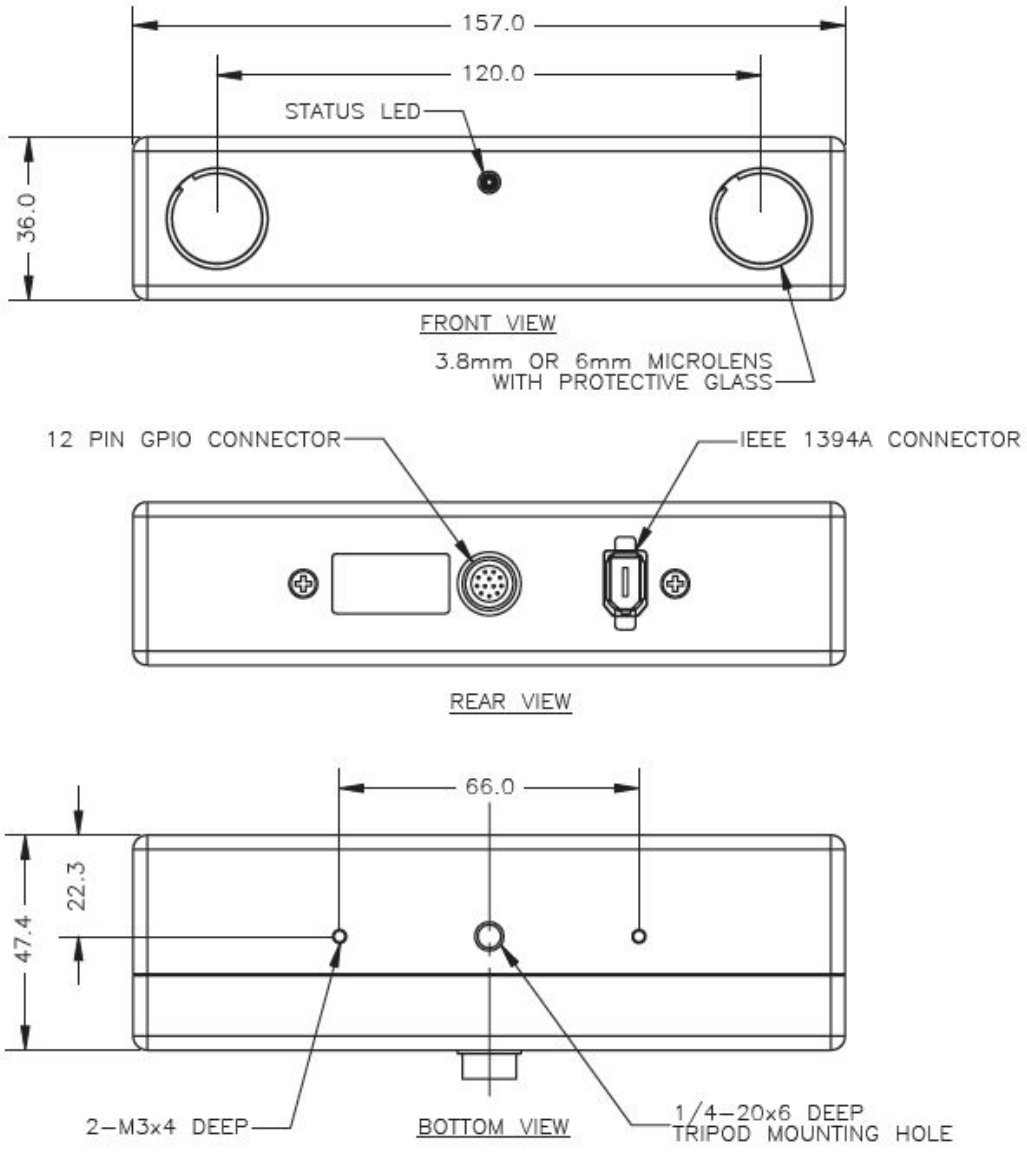


FIGURE A.3: Physical dimensions of the Bumblebee2.

Appendix B

Robust Regression Techniques

Random Sample Consensus

The RANSAC algorithm was first published by Fischler and Bolles in 1981 [84]. It is an iterative algorithm to robustly estimate parameters of a mathematical model from a set of noisy input measurements or data points. An unknown proportion of these input data points are consistent with a model of known parametric form and unknown parameters. These data points are called inliers and the remainder is called outliers. To determine the model parameters, θ, the RANSAC algorithm requires the following inputs:

- parametric form of the model, Θ
- input data points, D_{in}
- a distance threshold Δ to distinguish inliers and outliers
- maximum number of iterations, i_{max}

The sequence of steps of the algorithm are as follows:

1. Select a random subset from D_{in}; this is treated as a set of hypothetical inliers, H_{inlier}.

2. Fit the modelΘ to H_{inlier} in a least square sense and determine corresponding θ.

3. Test the remaining data against the fitted model; if the distance measure is less than Δ, update H_{inlier}.

4. Re-estimate θ using the updated set of inliers.

5. Save θ, H_{inlier} and the total residual error with respect to H_{inlier}.

6. Iterate steps 1 to 5 for i_{max} number of times.

Ultimately, the RANSAC algorithm outputs the θ giving rise to the maximum number of inliers with minimum total residual error.

Iteratively Re-weighted Least Squares Regression

Similar to the RANSAC algorithm, IRLS fits a robust parametric model on a given set of input data. The procedure is as follows:

1. Fit the model using weighted least squares regression; during the first iteration the weight matrix is an identity matrix.

2. Compute the least squares residuals r_i:

$$r_i = y_i - \hat{y}_i$$

where y_i and $\hat{y}_i$ are ith data and fitted value respectively.

3. Calculate adjusted and standardized residuals using r_i:

$$r_{adj} = \frac{r_i}{\sqrt{1-h_i}}$$

$$r_{std} = \frac{r_{adj}}{Ks}$$

where h_i are leverages that adjust the residuals by down-weighting high leverage data points that has a large effect on the least squares fit, K is a tuning constant equal to 4.685, and s is the robust variance given by MAD/0.6745 where MAD is the median absolute deviation of the residuals. A detailed description of h, K, and s is given in [85].

4. Compute the robust bisquare weights as follows:

$$w_i = \begin{cases} (1-r_{std}^2)^2 & |r_{std}| < 1 \\ 0 & |r_{std}| \geqslant 1 \end{cases}$$

5. Iterate steps 1 to 4 until the total residual error converges.

Hough Transform

The Hough transform [86] is a method to detect parameterized geometric curves in images by mapping image pixels into a parameter space; it is closely related to regression methods such as the least median of squares [87]. The target curves (e.g., straight lines, circles, ellipses etc.) in the image can be described by a general implicit equation: $f(u, v, \theta_1, ..., \theta_n) = 0$ where u and v are image pixels and $\{\theta_1, ..., \theta_n\}$ is a set of n parameters specifying the shape of the curve. The parameter space is defined by an n-dimensional histogram called an accumulator, in which each cell corresponds to a specific instance of the shape of interest. Each manifold

in 2D image space votes for accumulator cells that it passes through and only the cells that receive a substantial amount of votes are taken into consideration.

The classical Hough transform was concerned with the identification of lines in a pre-processed image (e.g., a binary edge map of a gray scale image). A straight line in image space can be represented by the equation $v = mu + c$, where m and c denote gradient and intercept respectively. The 2D accumulator space is constructed from quantized values of m and c and its bounding limits can be determined using prior knowledge on the type of lines to be detected. The straight line defined by each accumulator cell is back-projected to image domain. The intensities of the coinciding pixels are accumulated and assigned to the corresponding cell of the accumulator. Typically, the most likely lines can be extracted by seeking local maximas in the accumulator space. A problem with using the equation $v = mu + c$ to represent a line is that the slope approaches infinity as the line approaches the vertical. To get around this difficulty Duda and Hart proposed the generalized Hough transform [88] which represents the equation of a line in polar coordinate space as $u \cos\alpha + v \sin\alpha = \rho$.

Appendix C

Supplementary Results

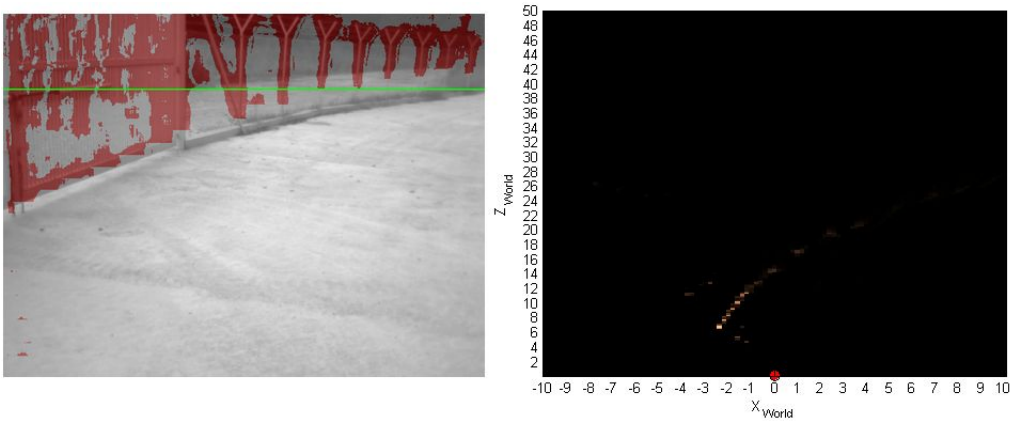


FIGURE C.1: Detection of a fence.

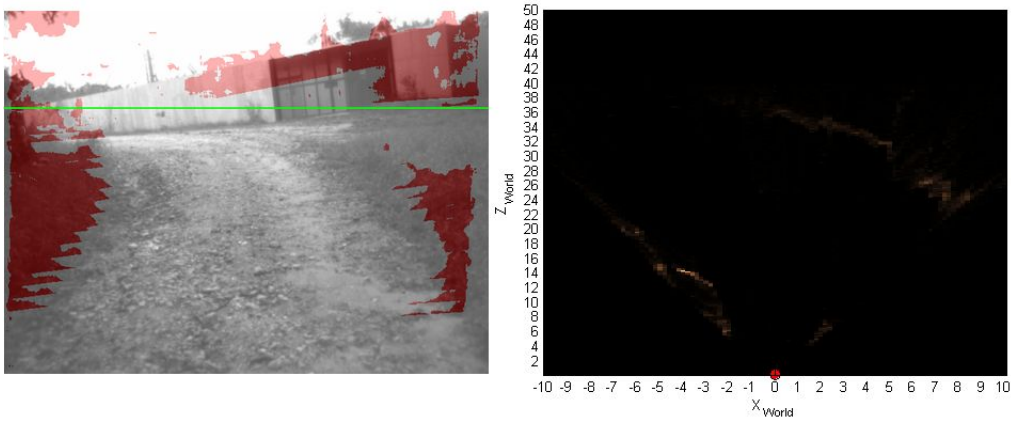


FIGURE C.2: Detection of a wall and a gate.

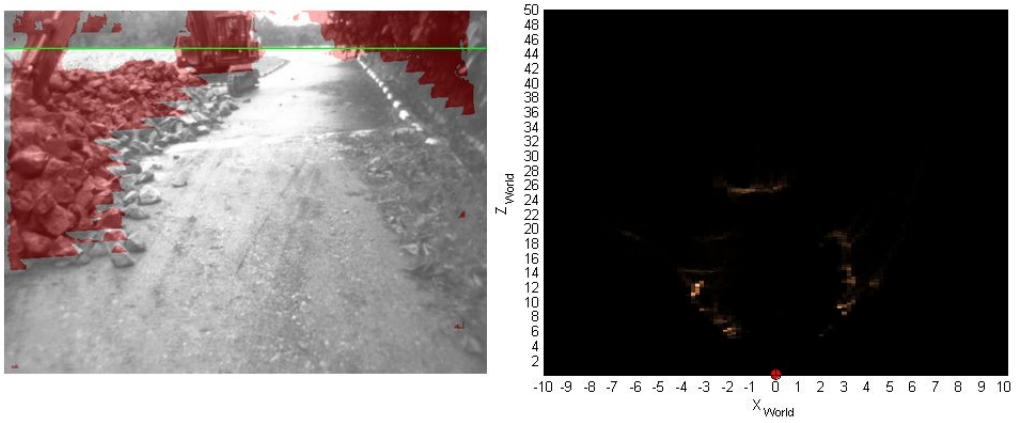


FIGURE C.3: Detection of a heap of stones and a construction vehicle.

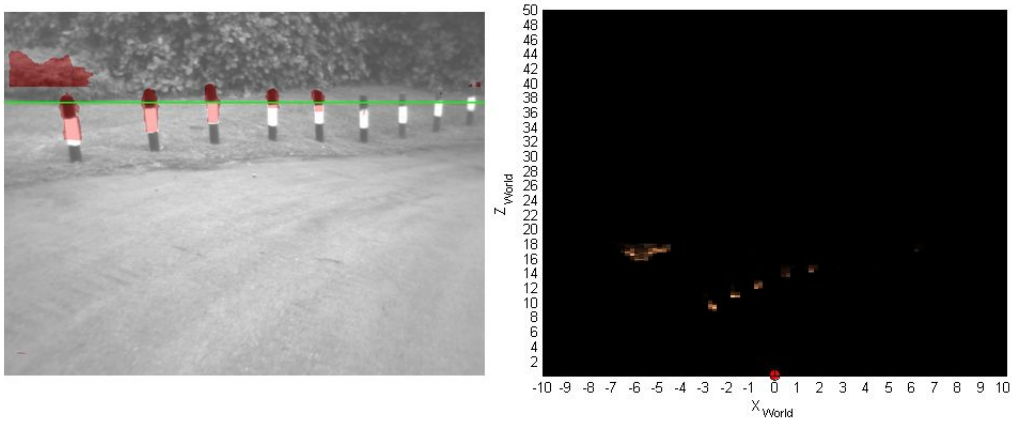


FIGURE C.4: Detection of barrier poles.

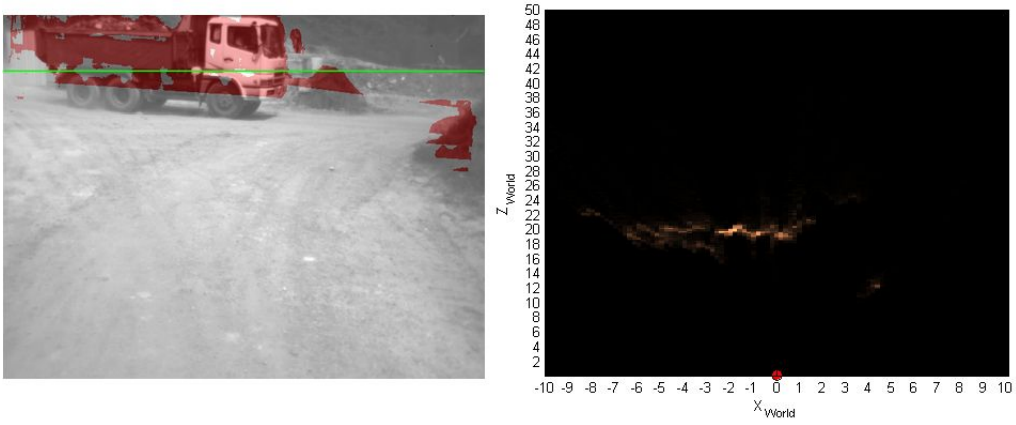


FIGURE C.5: Detection of a truck.

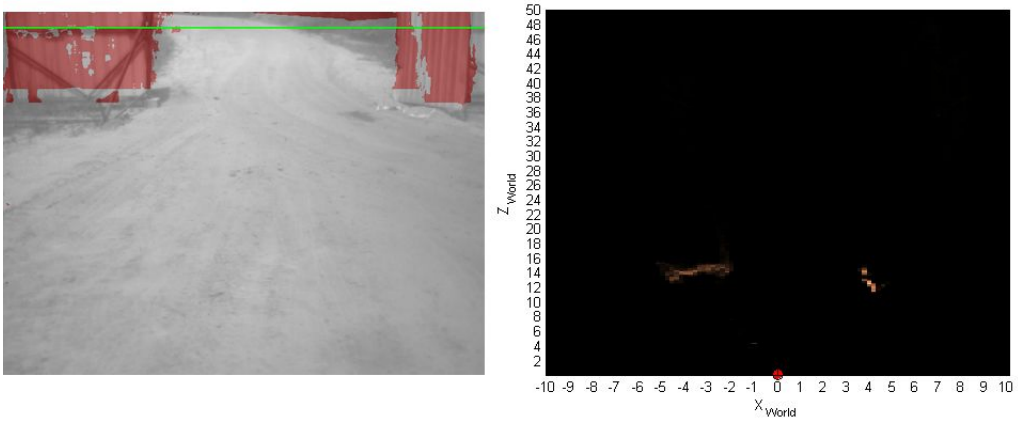


FIGURE C.6: Detection of a gate.

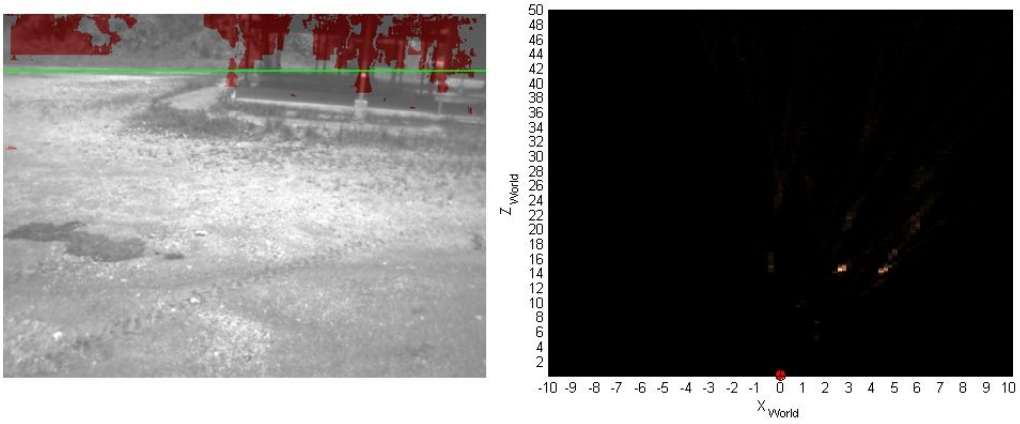


FIGURE C.7: Detection of a hut.

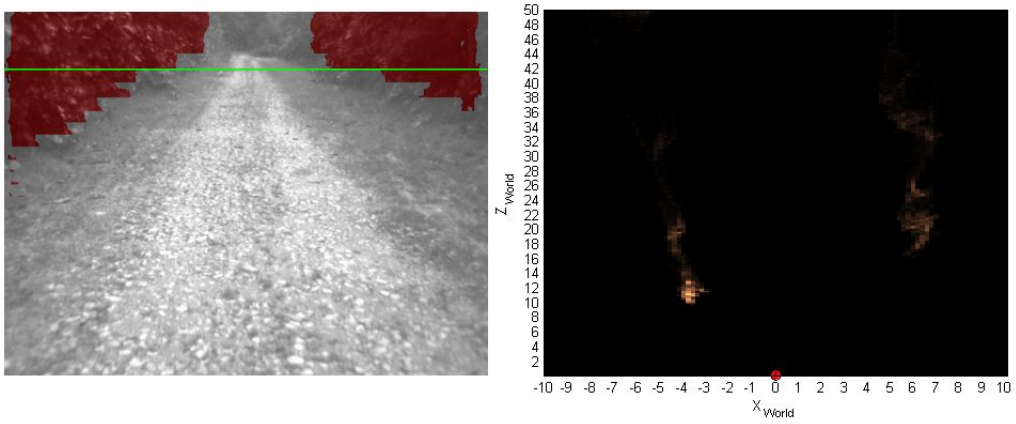


FIGURE C.8: Detection of vegetation.