VISION-BASED NAVIGATION FOR MOBILE ROBOTS ON ILL-STRUCTURED ROADS

A Dissertation

by

HYUN NAM LEE

Submitted to the Office of Graduate Studies of
Texas A&M University
in partial fulfillment of the requirements for the degree of

DOCTOR OF PHILOSOPHY

August 2008

Major Subject: Electrical Engineering

VISION-BASED NAVIGATION FOR MOBILE ROBOTS ON ILL-STRUCTURED ROADS

A Dissertation

by

HYUN NAM LEE

Submitted to the Office of Graduate Studies of
Texas A&M University
in partial fulfillment of the requirements for the degree of

DOCTOR OF PHILOSOPHY

Approved by:

Co-Chairs of Committee,	Dezhen Song
	Deepa Kundur
Committee Members,	Eun Jung Kim
	Narasimha Reddy
Head of Department,	Costas Georghiades

August 2008

Major Subject: Electrical Engineering

ABSTRACT

Vision-based Navigation for Mobile Robots on Ill-structured Roads. (August 2008)

Hyun Nam Lee, B.S., Hanyang University;

M.S., Hanyang University

Co–Chairs of Advisory Committee: Dr. Dezhen Song
Dr. Deepa Kundur

Autonomous robots can replace humans to explore hostile areas, such as Mars and other inhospitable regions. A fundamental task for the autonomous robot is navigation. Due to the inherent difficulties in understanding natural objects and changing environments, navigation for unstructured environments, such as natural environments, has largely unsolved problems. However, navigation for ill-structured environments [1], where roads do not disappear completely, increases the understanding of these difficulties.

We develop algorithms for robot navigation on ill-structured roads with monocular vision based on two elements: the appearance information and the geometric information. The fundamental problem of the appearance information-based navigation is road presentation. We propose a new type of road description, a vision vector space (V^2-Space), which is a set of local collision-free directions in image space. We report how the V^2-Space is constructed and how the V^2-Space can be used to incorporate vehicle kinematic, dynamic, and time-delay constraints in motion planning. Failures occur due to the limitations of the appearance information-based navigation, such as a lack of geometric information. We expand the research to include consideration of geometric information.

We present the vision-based navigation system using the geometric information. To compute depth with monocular vision, we use images obtained from different camera perspectives during robot navigation. For any given image pair, the depth error in regions close to the camera baseline can be excessively large. This degenerated region is named untrusted area, which could lead to collisions. We analyze how the untrusted areas are distributed on the road plane and predict them accordingly before the robot makes its move. We propose an algorithm to assist the robot in avoiding the untrusted area by selecting optimal locations to take frames while navigating. Experiments show that the algorithm can significantly reduce the depth error and hence reduce the risk of collisions. Although this approach is developed for monocular vision, it can be applied to multiple cameras to control the depth error. The concept of an untrusted area can be applied to 3D reconstruction with a two-view approach.

To my parents

ACKNOWLEDGMENTS

I would like to thank my advisor Dr. Dezhen Song for his support and guidance. I extend my gratitude to Dr. Deepa Kundur, Dr. A.L. Narasimha Reddy, and Dr. Eun Jung Kim for their invaluable advice. I also appreciate Dr. Scott Miller for his effort and time. Thanks to Dr. Jingang Yi for his insightful discussions and feedback.

I am grateful to my colleagues Dr. Qin, Y. Xu, C. Kim, Z. Goodwin, and Q. Hu. They gave me a great time in Netbot Laboratory. Thanks, also, to the Blue team for their effort and time for the DARPA Grand Challenge.

I would like to thank to all my friends who stay with me.

Finally, I would like to thank my family and Jeawoo. Without their support, love, and encouragement, I could not finish this long journey.

TABLE OF CONTENTS

CHAPTER Page

I INTRODUCTION 1

II RELATED WORK 5

III VISION-BASED MOTION PLANNING FOR AN AUTONOMOUS MOTORCYCLE ON ILL-STRUCTURED ROADS 8

- A. Introduction 8
- B. Related Work 10
- C. V^2-Space and Problem Description 12
  - 1. Nomenclature 12
  - 2. Assumptions 13
  - 3. Inputs 14
  - 4. V^2-Space 15
  - 5. Problem Statement 16
- D. Algorithms 17
  - 1. V^2-Space Construction 17
    - a. Color Correction 18
    - b. Surface Verification 19
    - c. Direction Extraction 22
  - 2. Motion Planning in V^2-Space 23
    - a. A Point Robot with No Time Delay 24
    - b. Incorporating GPS Information 27
    - c. Vehicle Size and Image Processing Delay 28
  - 3. Algorithm 30
- E. Experiments and Results 32
  - 1. Experiments with Video Clip 32
  - 2. Field Tests 35
- F. Summary 36

IV MONOCULAR VISION-BASED DEPTH-ERROR-AWARE ROBOT NAVIGATION 38

- A. Introduction 38
- B. Related Work 40

CHAPTER Page

C. Problem Description 43
  1. Coordinate Systems 43
  2. Nomenclature 44
  3. Assumptions 45
  4. Problem Context 46
    a. Frames and Frame Parameters 46
    b. Obstacle-free Region 47
    c. Region of Interest 47
  5. Untrusted Area 48
  6. Problem Definition 48
D. Analysis of Depth Error 50
  1. Computing Depth from Two Views 50
  2. Estimating the Depth Error Range 52
  3. Predicting Untrusted Area 56
    a. Partition R_i According to the Sign of Δe 56
    b. Computing A_u for $\Delta e > 0$ 57
    c. Computing A_u for $\Delta e < 0$ 60
    d. Computing the Overall A_u 62
E. Algorithm 63
  1. Computing $R_\varnothing$ 65
  2. Minimizing the Risk of Collisions Caused by $\overline{R}_\varnothing$ 70
  3. Depth-Error-Aware Navigation (DEAN) Algorithm 71
F. Experiments 73
  1. Software and Hardware 73
  2. Two Representative Cases 74
  3. Depth Error Reduction Effectiveness 75
G. Summary 78

V DEPTH-ERROR-AWARE 3D SCENE RECONSTRUCTION WITH A TWO-VIEW APPROACH 79

A. Introduction 79
B. Problem Description 80
  1. Coordinate Systems 80
  2. Nomenclature 81
  3. Assumptions 82
  4. Untrusted Area for a 360 Degree View 82
  5. Region of Interest 83
  6. Problem Definition 83

CHAPTER Page

C. Compute Untrusted Area for Two Cameras 84
  1. Untrusted Areas When the Cameras Face z_w-axis, A_u^1 . . . 85
  2. Untrusted Area When the Cameras Face x_w-axis, A_u^2 . . . 86
  3. Untrusted Area When the Cameras Face between the x_w-axis and z_w-axis, A_u^3 88
  4. $A_u^c(C_l, C_r)$ 89
D. Algorithm 93
E. Experiments 101
  1. Software and Hardware 101
  2. The Representative Case 103
  3. Depth Error Reduction Effectiveness 103
F. Summary 105

VI CONCLUSIONS AND FUTURE WORK 106

A. Conclusions 106
B. Future Work 107

REFERENCES 109

VITA 122

LIST OF TABLES

TABLE Page

I The evaluation of candidate arcs with respect to the RFQ functions in Eq. (3.18) for the examples in Figs. 10 and 11. 34

LIST OF FIGURES

FIGURE Page

1 (a) Autonomous motorcycle and (b) unstructured road in desert. 9

2 Camera configuration on the motorcycle. 14

3 An example of the collision-free directions $\mathbf{\Theta}$. 15

4 V^2-Space construction block diagram. 18

5 An example of the V^2-Space construction. (a) An original video frame with shadow. (b) The classification of the road using c_3 signature in the shadow invariant color model (c_1, c_2, c_3). (c) Output of surface verification. (d) Collision-free direction information $\mathbf{\Theta}$ over surface pixels. We omit the regions with $\Theta = \emptyset$. 19

6 Appearance-based obstacle detection. (a) The reference region in $\{W\}$. (b) The trapezoid is the reference region in $\{I\}$. 21

7 Directions extracted for a pixel. (a) A pixel at (u, v). (b) The similarity comparison along eight neighboring directions. (c) The extracted direction information. 22

8 Sample candidate arc trajectories and vehicle boundaries in (a) the world coordinate system $\{W\}$ and (b) the image coordinate system $\{I\}$. The solid arcs are candidate trajectories while dashed arcs are augmented boundaries of the vehicle that characterize the size of the vehicle and the size/location of the reference region defined in Fig. 6. Each solid arc has two corresponding dashed arcs. 26

9 A schematic of delay compensation. 30

10 An illustration of V^2-Space algorithm using a snapshot of the video clip captured in the Mojave desert. (a) $\mathcal{V}_s$, (b) $\mathcal{V}$, (c) result of the arc evaluation using Eq. (3.18), and (d) final choice of the arc with GPS inputs. 33

11 Two examples using the video data from the Mojave desert. 34

FIGURE Page

12 Robots used for field tests. 36

13 A 2D conceptual illustration of depth error distribution for two types of vision-based navigation systems. The dash line in each figure is the robot moving direction. The short line segments I and I' represent the imaging planes for camera C and C', respectively. The shaded regions in the figures represent the conic regions generated by back-projecting corresponding pixels from the camera centers to 3D space. The regions determine the depth error ranges. 39

14 Definition of coordinate systems and their relationship. The WCS is a fixed coordinate system while a CCS is attached to the moving camera. 43

15 A robot moves in region R_f to take frames to construct the depth information for region R_i to plan its navigation in R_i. The accuracy of the depth is determined by camera positions when the robot moves in R_f. 46

16 An illustration of depth error caused by the image correspondence error in $\overline{F}$. The intersection zone between the ray from $\underline{q}$, and the pyramid from $\overline{q}$ is the error range. If the error range projects onto the z axis, it is always bound between $z_w(\overline{u}+r, \overline{v}+r)$ and $z_w(\overline{u}-r, \overline{v}-r)$. (a) and (b) illustrate two subcases. 53

17 An illustration of Δe. Robot positions are set to be $x_w(0)$ =10 cm, $z_w(0)$ =-50 cm. The data and parameters are based on a Canon VCC4 camera. 55

18 A_us with different robot positions $(x_w(0), z_w(0))$. We set the threshold $e_t = 0.2z_w$. 63

19 $A_u(\underline{F}, \overline{F})$ and $A_u(F, \overline{F})$ are partially overlapped when F is taken in $\overline{R}_{\varnothing}$. 64

20 The relationship between A_us and the triangles generated by connecting three points, $(x_w^l(z_M, x_w(0), z_w(0)), z_M)$, $(x_w^h(z_M, x_w(0), z_w(0)), z_M)$, and $(0, 0)$. 66

21 The robot and the camera used in our experiments. 73

FIGURE Page

22 An illustration of robot positions and the relationship between A_us when $R_\varnothing \neq \emptyset$. 74

23 An illustration of robot positions and the relationship between A_us when $R_\varnothing = \emptyset$. 76

24 The effectiveness of depth error reduction. The height of the bar is the mean value of e_r and the vertical interval represents the variance of e_r. The number in the parenthesis is the trial number. 77

25 An illustration of the system configuration. The vision system consists of cameras C_l and C_r reconstructs a 360 degree view within region of interest R_i. To control depth error in the degenerated region, we use an additional camera C_n. C_n should be placed within the predefined available area R_c. 80

26 Definition of coordinate systems and their relationship. 81

27 An illustration of A_u^2. 86

28 An illustration of A_u^3. Two cameras are rotated by θ with respect to the y_w-axis. 88

29 An illustration of the relationship between $A_u^c(C_l, C_r)$ and $A_u^c(C_l, C_n)$ when $0° < \phi < 90°$. 96

30 An illustration of the relationship between $A_u^c(C_l, C_r)$ and $A_u^c(C_l, C_n)$ when $90° < \phi < 180°$. 100

31 The shaded area presents available (d', θ) set to satisfy the non overlapping condition between $A^c(C_l, C_r)$ and $A^c(C_l, C_n)$. 102

32 The camera used in experiments. 102

33 The shaded area illustrates the available set (d', ϕ) on the x-z plane when $d =$-50 cm. 103

34 The relationship between $A_u^c(C_l, C_r)$ and $A_u^c(C_l, C_n)$ when d=-50 cm. . . 104

FIGURE Page

35 The effectiveness of depth error reduction with different depth of obstacles. The height of the bar is the mean value of e_r and the vertical interval represents the variance of e_r. The number in the parenthesis is the trial number. 104

CHAPTER I

INTRODUCTION

Autonomous robots can explore hostile environments and perform tedious and dangerous work for humans. For example, a pair of autonomous robots have roamed on the surface of Mars since early 2004. As another example, research and rescue robots entered ground zero to look for survivors after the 9/11 tragedy. In transportation systems, autonomous navigation capability of vehicles can warn drivers before colliding with obstacles.

Let us consider an autonomous robot in the desert. The autonomous robot aims to reach a certain point. The mobile robot knows its position from on-board GPS readings. In order to arrive at its destination safely, it should avoid obstacles using on-board sensors such as vision sensors and range sensors. This navigational ability is a fundamental component for autonomous robots.

Due to the inherent difficulties in understanding natural objects and changing environments, navigation for unstructured environments, such as natural areas and the surface of Mars, has largely unsolved problems. However, navigation for ill-structured environments [1], where roads do not disappear completely, increases the understanding of these difficulties. In addition, DARPA Grand challenge[1] inspired us to develop algorithms for robot navigation on ill-structured roads.

Navigation for autonomous robots has been a very popular research field in the past decades [2,3]. It follows a *Sense-Plan-Act* model [4]. To implement sense and plan abilities in the navigation system, range sensors, such as ultrasonic sensors, laser range finders and radar, and/or passive vision sensors, such as cameras, are required. In the case of range

---

The journal model is *IEEE Transactions on Automatic Control.*

[1] http://www.darpa.mil/grandchallenge/

sensors, ultrasonic sensors are cheap but have specular reflection as well as poor angular resolution problems. Laser range finders provide better resolutions, but they are not eye safe and are also complex and expensive. Furthermore, range sensors cannot distinguish between different types of surfaces. This causes problems especially in outdoor navigation. For this reason, passive vision sensors are preferred for autonomous robot navigation. This dissertation focuses on vision-based navigation.

Vision-based navigation systems can be classified into three groups based on road conditions: structured environments, unstructured environments, and ill-structured roads. Vision-based navigation in structured environments, such as highways or roads in urban areas, is a solved problem because structured environments have clear boundaries and uniform surfaces. Examples of navigation systems in this category are the *No Hands Across America* project [5] and the ARGO project [6]. In the case of unstructured environments such as natural environments or the surface of Mars, robot navigation has many problems to be overcome since it is difficult to find navigable areas. The research for navigation system in unstructured environments is still in its infancy. Our research focuses on ill-structured environments where roads do not disappear completely. Although the particular properties of road areas are not clear, there is appearance information, such as color or texture, and geometric information, such as road surfaces. The primary challenges to consider for ill-structured roads are shadow and illumination changes, the lack of clear road features, drastic changes of road conditions and little or no prior knowledge of the roads.

Another classification for vision-based navigation systems is based on sensing methods: monocular vision, stereo vision, and vision combined with active sensors. Monocular vision uses a single camera, while stereo vision employs multiple cameras. Vision sensors can be combined with active sensors, such as LADAR. In the case of stereo vision, the baseline distance between two cameras would be very limited due to the width limitations of the robot platform. Stereo vision system with a limited baseline distance cannot provide accurate and timely depth information [7]. LADARs are not eye safe and cannot be used in populated areas. Therefore, our system focuses on monocular vision.

We develop a navigation system based on two elements: the appearance information of roads, such as color and texture, and the geometric information of roads. We begin with the navigation system based on appearance information. The fundamental problem of the appearance information based navigation system is how to represent roads. We propose a new type of road description, a *vision vector space(V^2-Space)* for ill-structured roads, which is a unitary vector set that represents local collision-free directions in image space. The V^2-Space is constructed by extracting the vectors based on the similarity of adjacent pixels, which captures both the color information and the directional information from prior vehicle tire tracks and pedestrian footsteps. We report how the V^2-Space is constructed to reduce the impact of varying lighting conditions in outdoor environments. We also show how the V^2-Space can be used to incorporate vehicle kinematic, dynamic, and time-delay constraints in motion planning to fit the highly dynamic requirements of a motorcycle. We have implemented the proposed algorithms. The algorithm is tested with both the video data from actual road data and a three-wheel mobile robot. Experimental results show that the algorithm can make correct decisions at a rate of more than 90%. During the development of the appearance information-based navigation system, we noticed that the appearance information-based navigation system cannot deal with obstacle avoidance problems effectively without depth information. This inspired us to develop the geometric information-based navigation system.

Geometric information, especially depth, is required for robot navigation to overcome the limitations of appearance information-based navigation. We present the vision-based navigation system based on geometric information. This demonstrates how to compute depth information using image frames taken from different camera perspectives while the robot is traveling. Given a calibrated camera, the accuracy of the depth largely depends on the locations where images have been taken. For any given image pair, the depth error in regions close to the camera baseline can be excessively large or even infinite. This is due to the degeneracy introduced by the triangulation in depth computation. Unfortunately, this region often overlaps with the robot moving direction, which can lead to collisions. We name this region an untrusted area. Notice that the robot's move determines the location of the future untrusted area. We analyze how the untrusted areas are distributed on the road plane and predict them accordingly before the robot makes its move. We propose an algorithm to assist the robot in avoiding the untrusted area by selecting optimal locations to take frames while navigating. Therefore, the overall depth error can be controlled below a predefined threshold. We have implemented the algorithm and tested tested using a three-wheel mobile robot. Experimental results show that the proposed algorithm can significantly reduce the depth error and hence reduce the risk of collisions. Although our depth error model is developed for a monocular vision system, it can be applied to a robot with multiple cameras or pan-tilt-zoom camera network to control the depth error. We apply the depth error model for a vision system with multiple cameras.

The concept of an untrusted area is applied to 3D scene reconstruction with a two-view approach. For surveillance purpose, a 360 degree view is reconstructed using a two-view approach to recognize objects on the ground surface. During the stereo reconstruction, the untrusted area is unavoidable due to the degeneracy introduced by triangulation. This degenerated region might cause failures in object detection. We found that the location of the untrusted area for a 360 degree view depends on camera positions. Hence, we use an additional camera to avoid the untrusted area. In this research, we compute the untrusted area for a 360 degree view and the location of an additional camera to produce the least depth error. Experimental results confirm depth error analysis.

CHAPTER II

RELATED WORK

Using vision to assist mobile robots and vehicles in navigation has been a popular research field in the past decade [2, 3]. With applications ranging from intelligent vehicles to autonomous mobile robots, research can be classified based on road conditions and sensing methods.

If a robot is running on a *well-structured road*, such as freeways or the roads in an urban area [8, 9], the primary focus of research is lane detection [10] using surface and boundary features, and road following [11], which detects road trends. Since the road has a relatively uniform surface and clear lane markings, techniques such as road segmentation, road edge detection [12], and curve-fitting [13] are often used to generate vehicle control inputs. The vision-based navigation systems in this category have proved their performance. In the *No Hands Across America* project [5] of Carnegie Mellon University, the RALPH algorithm [14] navigated Navlab5 from Pittsburgh, PA to San Diego, CA in 1995. The Navlab5 drove a total of 2849 miles, and it ran 2797 (98.2%) without human assistance. Another example is the ARGO proejct [6] of the University of Parma, Italy. In 1998, the ARGO autonomous vehicle drove about 2000km throughout Italy using the GOLD algorithm [11].

When a robot is running in an *unstructured environment* such as a natural environment [15, 16] or the surface of Mars [17], terrain classification and obstacle avoidance become the primary challenges [18]. In such cases, advanced sensors such as stereo cameras, Laser RADAR (LADAR), and appropriate sensor fusion techniques are necessary to deal with the complex environment [19–21]. Due to the inherent difficulties in understanding natural objects and changing environments, autonomous driving is still in its infancy. However, existing results such as motion planning with 3D vision and the use of multiple classifiers [18, 22] shed light on a different class of problems, which are referred to as *ill-structured environments* [1], where roads do not disappear completely.

We discuss the motion planning problem for an autonomous motorcycle on an *ill-structured road* that does not have clear lane markings or pavement but might contain the color information and the directional information from prior vehicle tire tracks and pedestrian footsteps. Color information is used to distinguish obstacles and navigable areas [23, 24]. Color information is combined with other features, such as texture and road hight, to extract travelable regions accurately [25]. The mobile robot is guided by extracting vanishing points using directional information [1]. Recent developments in this area are largely driven by the DARPA Grand Challenge in 2005. Although the 2005 DARPA Grand Challenge indicates a big success in robot navigation development, the vision-based navigation has almost no significant contributions to this success. All winning teams rely on LADARs as primary sensing inputs during the race. This also illustrates the reality of the vision-based navigation development. It cannot be used as a reliable sensing mechanism for robot navigation. The vision-based navigation on ill-structured roads remains a challenging problem.

Although a LADAR is a great sensor to obtain the shape of a road, it has the limitation of the inability of distinguishing surface types (i.e. to tell the difference between a water surface from a road surface). It is a great idea to combine LADAR inputs with vision data [15, 26]. However, this does not address the problem that long range LADARs are not eye safe and hence cannot be used in a populated area. The vision-based navigation that is based on passive sensing has its irreplaceable advantages.

Stereo vision [27] can provide 3D information about the environment. However, considering the width limitations of the motorcycle platform, the “baseline” distance between cameras would be very limited if binocular stereo vision is used. As pointed out by Michels et al. [28], a fast moving vehicle needs to observe obstacle at a far distance, and the binocular cameras with limited “baseline” distance cannot provide accurate and timely depth information [7] in a dynamic and noisy environment as the vehicle would experience. Therefore, we decided to use monocular vision in our system.

Although monocular vision does not provide the distance of obstacles accurately, it has advantages, such as offering fast processing, compact system, and easy implementation. Navigation systems in this category use color information [29–32] to distinguish obstacles and road areas under the assumptions that mobile robots are on the road and the color of the road is homogenous. Another application of monocular vision is to extract land markings [11, 14], road boundaries [13, 33], or directional information of road [1] for navigation.

Our research focuses on the vision-based navigation with monocular vision for ill structured roads. Since ill-structured environments do not have uniform road areas and clear road boundaries, it is difficult to find navigable areas based on color information or directional information alone. Our system uses both color information and directional information to extract travelable roads for mobile robots. During the development of monocular vision system, we find that depth information is necessary to avoid obstacles efficiently. Hence, we expand the monocular vision system considering depth computation.

CHAPTER III

VISION-BASED MOTION PLANNING FOR AN AUTONOMOUS MOTORCYCLE ON ILL-STRUCTURED ROADS *

A. Introduction

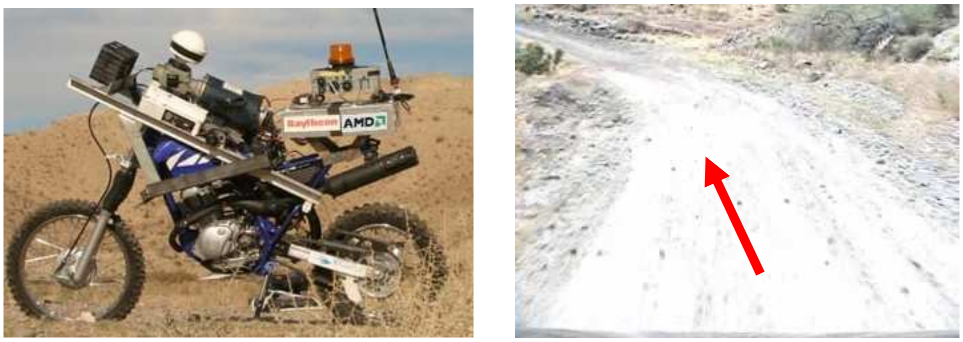
Motivated by the DARPA Grand Challenge 2005, we have developed a vision-based motion planning system for an autonomous motorcycle (Fig. 1) to run across desert terrain or ill-structured roads, where uniform road surface and lane markings do not exist. Since global positioning system (GPS) signals are not enough to guide the vehicle to avoid obstacles, additional sensors and decision-making capabilities are needed. Although the single-track platform (motorcycle) provides us with strong off-road capabilities such as excellent agility and navigation on rough terrain, and an ability to pass through narrow openings, its limited size and power supply do not allow us to install sophisticated sensors such as multiple cameras or multiple laser range finders.

Because of size and power constraints, our motorcycle has one video camera, a GPS receiver, a couple Inertial Measurement Units (IMU), and two on-board computers. The vision system consists of only one camera and one laptop PC while the other computer is dedicated to vehicle balance and low level control. Furthermore, the highly dynamic property of the motorcycle demands very responsive vision data processing. These constraints motivate our research to develop a fast and robust vision-based motion planning system for an ill-structured road.

During the development of the vision-based navigation systems for the autonomous motorcycle, we found that we cannot separate road detection, motion planning, and vehicle

---

*Reprinted with permission from ”Vision-based motion planning for an autonomous motorcycle on ill-structured roads” by Dezhen Song, Hyun Nam Lee, and Jingang Yi, 2007, *Autonomous Robots*,vol. 23, pp. 197-212, Copyright[2007] by Springer.



(a) (b)

Fig. 1. (a) Autonomous motorcycle and (b) unstructured road in desert.

kinematics and dynamics into the isolated individual problems as a conventional approach would do. The dynamic nature of the motorcycle platform determines that all of the above are highly coupled together. For example, a trajectory generated by the algorithm has to consider road conditions, current vehicle status, and vehicle kinematic and dynamic limits. We choose to take an integrated approach that combines all of three elements above in the motion planning.

We found that the fundamental element of the integrated approach is the data representation of ill-structured roads. Although binary maps, lines, and polygons are common data representations in a structured road, none of them are appropriate for an ill-structured road because there is no clear boundaries, no lane markings, and no uniform surfaces. We propose a concept of the vision vector space (V^2-Space), which is a unitary vector set that represents local collision-free directions using a 2D image coordinate system. The V^2-Space is constructed by extracting unit vectors based on the similarity of adjacent pixels, which includes the color information and the directional information from prior vehicle tire tracks and pedestrian footsteps. Designed as an open framework, the V^2-Space can support many existing developments, such as new road detections or machine learning techniques, and still facilitate motion planning with vehicle kinematic and dynamic constraints.

We report how the V^2-Space is constructed using a shadow/illumination invariant color model and a maximum variance color projection method to reduce the impact of varying lighting conditions in outdoor environments. We also show that how the V^2-Space can be used to incorporate vehicle geometric, vehicle dynamic, and time-delay constraints in motioning planning to fit the highly dynamic requirements of the motorcycle. The combined algorithm of the V^2-Space construction and the motion planning runs in $O(n)$ time, where n is the number of pixels in the captured image. Experiments show that it outputs correct robot motion commands more than 90% of the time.

The rest of the chapter is organized as follows, we review exiting work on the vision-based robot motion planning in section B. We propose the V^2-Space in section C. We present an algorithm for the V^2-Space construction and motion planning in section D. Experiments and summary are reported in section E F, respectively.

B. Related Work

The primary challenge of monocular vision for ill-structured roads arises from several aspects: 1) shadow and illumination changes, 2) no clear road boundaries, 3) drastic changes of road surface, and 4) little or no prior knowledge of the roads. The motion blurring and the vibration caused by a fast moving vehicle undermine image quality. To address these issues, researchers approach the problem using different strategies such as color vision [18, 23, 34], road detection [35–37], prior knowledge of road surface [12], pixel voting [1], classifier fusion [22], optical flow [24], neural networks [8], and machine learning [24,28,38,39]. Existing developments provides excellent building blocks for the vision-based navigation. Complementary to existing approaches, we take an integrated approach that combines road/obstacle detection and vehicle dynamics with motioning planning. The new approach starts with looking for a new data representation of ill-structured roads.

Raw vision data cannot be directly used to perform motion planning for a robot. For the structured environments, high level geometric feature representations such as points, lines, surfaces, and polygons can be used to abstract vision data [40, 41]. For the unstructured or ill-structured environments, common feature representations include binary maps and optical flow [24, 42]. Polygons can be used if a road has a clear boundary [43]. The optical flow is the vector field which warps one image into another (usually very similar) image. The vector field captures both the motion of the robot and other moving objects based on the adjacent video frames. It contains information about moving obstacles and the robot, but it does not work for a still vehicle and is sensitive to motion blurring.

Although desert roads have no clear boundaries, the tire tracks and the foot steps left by prior vehicles or pedestrians can provide directional information for vehicle motion planning. Broggi and Berte [10] notice that similar information is provided by lane-markings on urban roads and name it "internal edges". Rasmussen [1] and Zhang et al. [33] name the directional information in ill-structured roads as "dominating directions". Rasmussen uses it to vote for a vanishing point to guide the vehicle. However, one pixel may have more than one dominating direction. A road may fork or intersect with other roads. The directional information can also be trimmed using color information. Our V^2-space is designed to capture the information.

The proposed V^2-Space approach is inspired by the vector field histogram (VFH) [44]. Developed to assist a robot to navigate in a 2D environment with range sensors, the VFH divides the environment according to a Cartesian grid. For each grid cell, the VFH keeps track of a histogram of range sensor readings with respect to angular orientations. The VFH allows the robot to keep track of the obstacles in the environment and enables fast navigation. The VFH and our V^2-Space share the similar aspect of describing obstacle-free space using angular coordinates with respect to each cell or pixel. As pointed out by [44], this representation allows the fast navigation of mobile robots. However, the V^2-Space differs VFH in both its construction and its representation. The V^2-Space is constructed in sensor space, which is the pixel coordinates defined by the camera, instead of using the 2D Cartesian world space. Each entry in V^2-Space is a unitary vector set constructed according to the similarity among local pixels instead of the sensor reading obtain by range sensors in VFH.

C. V^2-Space and Problem Description

1. Nomenclature

- $\{W\}$: 3D Cartesian world coordinate system (x, y, z).
- $\{I\}$: 2D image coordinate system (u, v).
- $\mathcal{F}$: A raw video frame.
- $\mathcal{F}_c$: A video frame after color correction.
- $\mathcal{V}_s$: A video frame after surface verification.
- $\mathcal{V}$: V^2-Space.
- ϕ: Direction to next way point in degrees.
- $\mathcal{T}_W$: Motorcycle trajectory in $\{W\}$.
- $\mathcal{T}$: Motorcycle trajectory in $\{I\}$.
- c_p: The separation color used for surface verification.
- $\mathcal{T}_W^+(\mathcal{T}_W^-)$: $\mathcal{T}_W$'s upper (lower) envelope in $\{W\}$, respectively.
- $\mathcal{T}^+(\mathcal{T}^-)$: $\mathcal{T}$'s upper (lower) envelope in $\{I\}$, respectively.

- f: Road Following Quality (RFQ) function.
- $l_r (w_r)$ Motorcycle length (width), respectively.
- τ]: Trajectory length in time.
- τ_i: Inter-iteration time for motion planning.
- t_m: Measurement time delay.
- t_d: Decision/execution time delay.

2. Assumptions

A pin-hole model [45] is used for modeling the on-broad video camera. It is assumed that the camera is calibrated and that both the camera's intrinsic and extrinsic parameters (with respect to the vehicle) are known. Therefore, we can determine a perspective projection matrix M that projects a point/patch $P = [x\,y\,z\,1]^T$ in the world frame $\{W\}$ to its corresponding pixel in the image frame $\{I\}$ as $p = [u\,v\,1]^T$

$$[u\ v\ 1]^T = M_{3\times4}[x\ y\ z\ 1]^T\,. \tag{3.1}$$

We assume that the lens distortion of the camera is either negligible or is compensated beforehand. It is worth mentioning that perspective projection matrix M depends upon not only static camera mounting mechanisms but also on vehicle orientation including pitch, yaw, and roll angles. As illustrated in Fig. 2, the camera is mounted on the frame of the vehicle. Since a running motorcycle has dynamic pitch, yaw, and roll angles, matrix M is not a constant. We assume that we can read vehicle pitch, yaw, and roll angles from an IMU. We also ignore the camera vertical motions caused by vehicle suspension because the vertical motions can be sensed by another IMU that is attached to the camera and can be compensated later. We assume that the vehicle maintains contact with ground at all time.

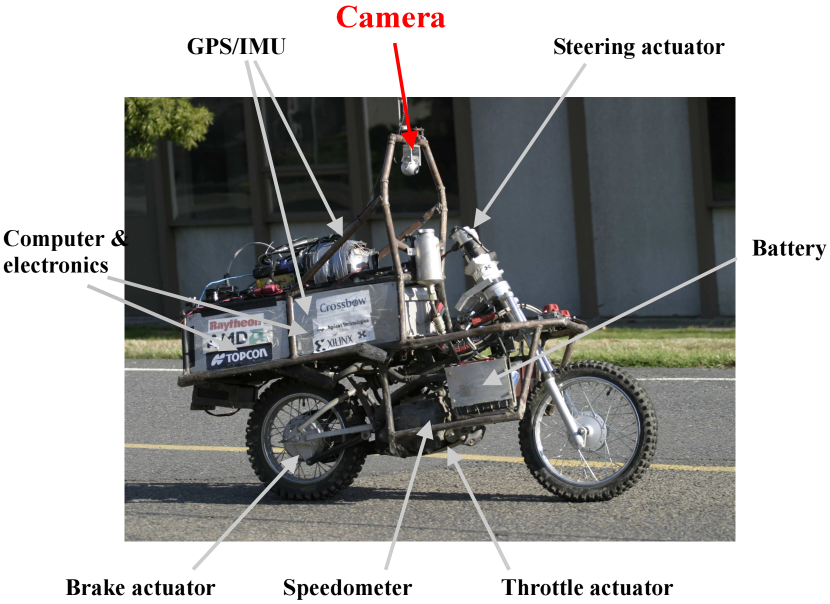


Fig. 2. Camera configuration on the motorcycle.

Therefore, we can obtain M in real time, which means that the dynamic correspondence between $\{I\}$ and $\{W\}$ is known. We also assume that ground surface is relatively flat. Therefore, it can be treated as a ground plane. Depth information can be estimated using intrinsic and extrinsic camera parameters in the monocular vision.

3. Inputs

The video data from a camera are the primary input for our motion planning system. Define the pixel set of a raw video frame $\mathcal{F}$ with $n = l \times h$ pixels as $I = \{(u, v)|1 \leq u \leq l,\ 1 \leq v \leq h, u, v \in \mathbb{N}\}$, where (u, v) are pixel coordinates in $\{I\}$. The video frame $\mathcal{F}$ is a matrix of RGB values

$$\mathcal{F} = (\mathcal{F})_{uv} = ((R, G, B))_{uv},\ (u, v) \in I, \tag{3.2}$$

where $R, G, B \in \mathbb{Z}$ and $0 \leq R, G, B \leq 255$ are integer intensity values for each color channel. Another important input of the system is the direction angle to the next waypoint ϕ obtained from onboard GPS signals.

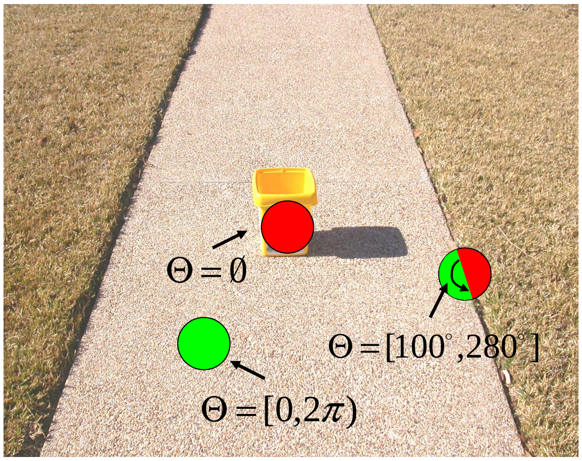


Fig. 3. An example of the collision-free directions $\mathbf{\Theta}$.

4. V^2-Space

The V^2-Space is a collection of unitary vectors that describes local collision-free directions. For frame $\mathcal{F}$, its V^2-Space is,

$$\mathcal{V}(\mathcal{F}) = \{\mathbf{\Theta}(u, v) : \text{ collision-free directions at pixel } (u, v)\}, \tag{3.3}$$

where $\mathbf{\Theta}(u, v) \subseteq [0, 2\pi)$ is a set of collision-free directions at location (u, v),

$$\begin{cases} \mathbf{\Theta}(u, v) = [0, 2\pi), & \text{If pixel } (u, v) \text{ is on the road} \\ \mathbf{\Theta}(u, v) = \emptyset, & \text{If pixel } (u, v) \text{ is an obstacle} \\ \mathbf{\Theta}(u, v) \subset [0, 2\pi), & \text{If pixel } (u, v) \text{ is on boundary.} \end{cases} \tag{3.4}$$

Fig. 3 shows an example of the defined collision-free directions at different pixels. Since the V^2-Space uses the same pixel coordinate of the raw frame in Eq. (3.2), the perspective projection relationship in Eq. (3.1) holds between the V^2-Space and $\{W\}$.

5. Problem Statement

In each iteration, the motion planning system takes an image $\mathcal{F}$ as input and outputs a trajectory. The length of the trajectory in time is defined as τ. Trajectory length τ is not a constant but a function of current vehicle speed and camera coverage. In fact, τ is unknown before a trajectory and its velocity profile are generated. On the other hand, our motion planning algorithm runs every τ_i milliseconds, which is a constant. To ensure that there is always a planned trajectory for the vehicle at any time, the inter-iteration time τ_i has to be strictly less than τ. This constitutes the feasibility condition of the motion planning algorithms. A short τ_i, which usually depends on how fast the algorithm can run, would significantly improve the feasibility of the planning.

Although a trajectory has a length of $\tau > \tau_i$, only the first $[0, \tau_i]$ part of the whole trajectory $[0, \tau]$ will be executed because next trajectory will replace the current trajectory at time τ_i. However, the overlapping part $[\tau_i, \tau)$ is still useful because it helps us to improve the system robustness and connection smoothness among piecewise curvatures. We will explain it below in the section of velocity profile generation.

The problem formulation for each planning iteration (with the time period τ_i) is,

Definition 1 (Motion planning)**.** *Given $\mathcal{F}$ and ϕ, find trajectory*

$$\mathcal{T}_W(\tau) = \{(x(t), y(t)) | \, t \in [0, \tau)\} \tag{3.5}$$

for the robot, where $(x(t), y(t))$ is the robot position in $\{W\}$ at time t.

We propose to solve the above motion planning problem in the image frame $\{I\}$. In the following, we first discuss how to compute $\mathcal{T}_W(\tau)$ using the defined V^2-Space and then find the optimal motorcycle motion trajectory $\mathcal{T}_W(\tau)$.

D. Algorithms

We propose to use a computational approach for the motion planning problem. Because the planning period τ_i is small, we can approximate trajectory $\mathcal{T}_W(\tau)$ by a circular curve starting at the current motorcycle position and tangent to the current vehicle velocity. A circular curve has a constant curvature, which is a special case of a curve with a linear curvature. To ensure the controllability, Ma et al. [46] have proved that a nonholonomic robot such as a motorcycle can only track a piecewise linear curvature under perspective projection. The overall trajectory generated by our algorithm is piecewise a linear curvature, which ensures that motorcycle can execute the planned path [47].

Using such an approximation, we can denote the trajectory $\mathcal{T}_W(\tau)$ by a triplet $(R, d, v^p(t))$ as

$$\mathcal{T}_W(\tau) = \{(R, d, v^p(t)) | R \in [R_{\min}, \infty), d \in \{0, 1\}, t \in [0, \tau)\}, \tag{3.6}$$

where R is the radius of the trajectory, binary variable $d = 0$ (left) or 1 (right) for the trajectory direction with respect to the current velocity direction, $v^p(t)$ is the velocity profile of the trajectory, and $R_{\min}$ is the minimal turning radius of the motorcycle. We compute $\mathcal{T}_W(\tau)$ given by Eq. (3.6) in two steps: first, we compute V^2-Space, $\mathcal{V}$, and then we search for a trajectory in $\mathcal{V}$ using a set of circular candidate curves. We begin with the first step of the V^2-Space construction.

1. V^2-Space Construction

The V^2-Space construction is a non-trivial feature extraction problem. We propose a three step V^2-Space construction-algorithm as illustrated in Fig. 4.

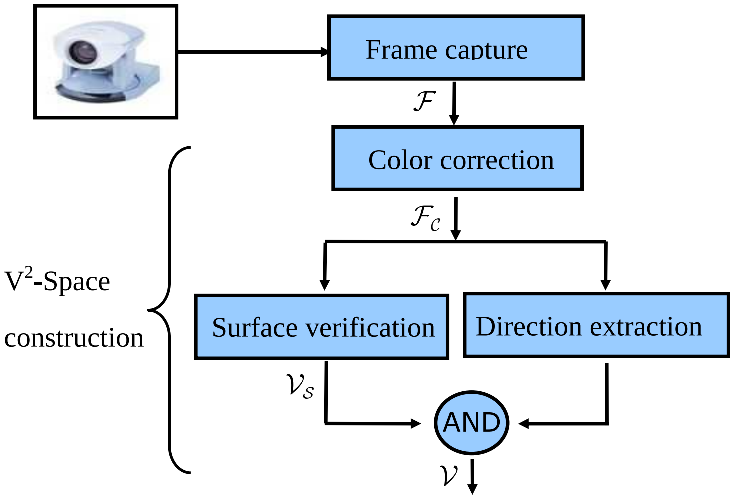


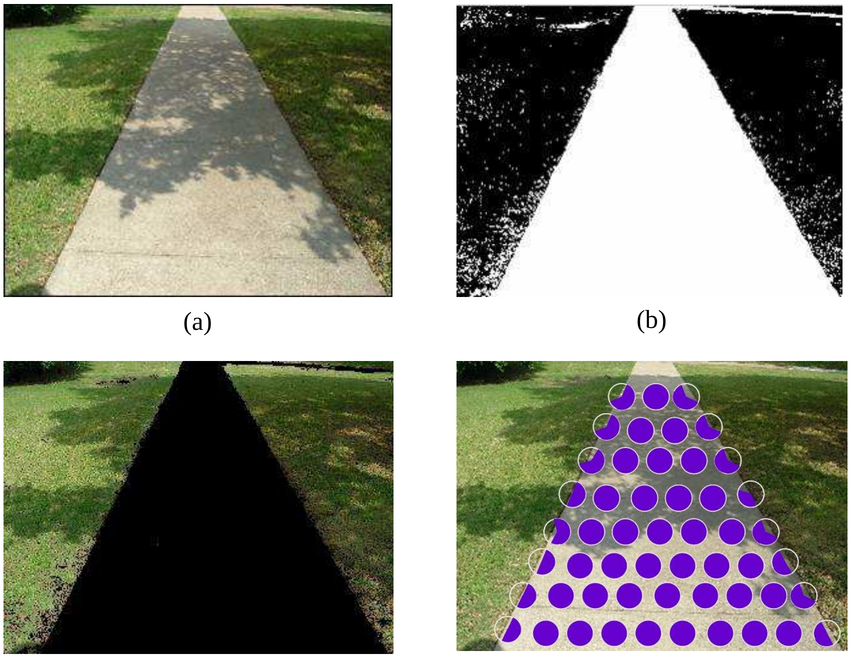
Fig. 4. V^2-Space construction block diagram.

a. Color Correction

The purpose of color correction is to minimize the shadow and illumination change effects. Hue, Saturation, and Intensity (HSI) color models have been used widely in road identification research because they are insensitive to illumination [45]. However, our initial experiments have shown that the HSI color model is not very effective in shadow elimination. We have tested and compared a number of color models such as HSI, normalized Blue [48], and the $l_1l_2l_3$ and the $c_1c_2c_3$ in [49]. Although the $c_1c_2c_3$ color model is originally designed to be shadow-invariant under the indoor lighting conditions, our experiments show that it is the best shadow and illumination invariant color model for outdoor vision algorithms,

$$c_1 = \arctan\left(\frac{R}{\max(G,B)}\right),\ c_2 = \arctan\left(\frac{G}{\max(R,B)}\right), \qquad (3.7)$$
$$c_3 = \arctan\left(\frac{B}{\max(R,G)}\right).$$

Fig. 5(b) shows that the $c_1c_2c_3$ color model is very effective in shadow elimination. With the corrected color information, our content analysis in subsequent steps becomes



(a) (b)

(c) (d)

Fig. 5. An example of the V^2-Space construction. (a) An original video frame with shadow. (b) The classification of the road using c_3 signature in the shadow invariant color model (c_1, c_2, c_3). (c) Output of surface verification. (d) Collision-free direction information $\boldsymbol{\Theta}$ over surface pixels. We omit the regions with $\Theta = \emptyset$.

significantly more robust. Let us define the output of this step as,

$$\mathcal{F}_c = \{(u, v), c_1, c_2, c_3 | (u, v) \in I\}.$$

It takes $O(n)$ time to compute $\mathcal{F}_c$ for an $n = l \times h$-pixel frame.

b. Surface Verification

The purpose of surface verification is to identify obstacles and other non-road regions. It outputs a description of free space that the vehicle can pass through. Let us define such free space as $\mathcal{V}_s$, which takes the same format as Eq. (3.3). The transformation from $\mathcal{F}_c$ to $\mathcal{V}_s$ is a data-reduction process that builds on both prior knowledge and statistic techniques.

As prior knowledge, we find that desert terrain is not completely unstructured. Vegetation can serve as a nice marking of non-road regions. A large portion of $\mathcal{F}_c$ contains only two types of surface: the sandy surface and the vegetated surface. A continuously connected surface of the sandy surface is more likely to be a road. Therefore, our first step is to find an effective color discrimination to separate the two types of surfaces. Since vector (c_1, c_2, c_3) is a 3D point in color space, our conjecture is that there should exist an unknown plane in the color space such that the difference between the two types of surfaces is maximized if we project (c_1, c_2, c_3) to the plane. The question then becomes how to find the plane.

Define (w_1, w_2, w_3) as the unitary normal vector of the plane. The color projection of a pixel (u, v) in $\mathcal{F}_c$ is the inner product of two vectors,

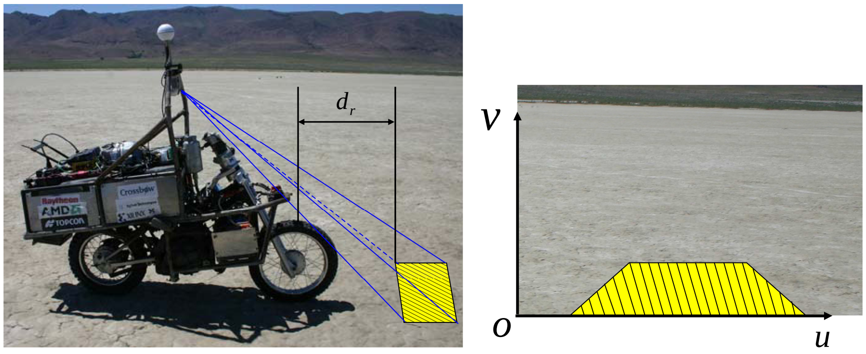
$$c_p(u, v) = w_1 c_1 + w_2 c_2 + w_3 c_3, \tag{3.8}$$

where $c_p(u, v)$ is the separation color that will be used to classify pixels. We employ a data-driven method to estimate (w_1, w_2, w_3) by maximizing the variance,

$$\begin{aligned}(w_1, w_2, w_3) &= \arg \max_{w_1, w_2, w_3} Var(c_p(u, v)) \\ &\quad \text{s.t. } w_1^2 + w_2^2 + w_3^2 = 1.\end{aligned}$$

This can also be viewed as a variation of the unsupervised linear classifier according to [50]. Since the separation color (w_1, w_2, w_3) depends on the lighting conditions and surrounding environments, it usually does not change dramatically in navigation and can be either precomputed or repeated at long intervals (i.e. every 5 minutes). We actually run it as the parallel routine of the main motion planning algorithm.

Now we can reduce $\mathcal{F}_c$ to $\mathcal{F}_p = \{c_p(u, v) | (u, v) \in I\}$. We build on the appearance-based obstacle detection method in [29] to detect obstacles and classify regions. The



(a) (b)

Fig. 6. Appearance-based obstacle detection. (a) The reference region in $\{W\}$. (b) The trapezoid is the reference region in $\{I\}$.

method is based on the assumption that there exists a reference road region in the image. The reference region is believed to be on the road because it is usually the closest region in front of the robot if the robot stays on the road. The trapezoid region (in $\{I\}$) in Fig. 6(b) is used as the reference region. Using the pixels in the reference region, we can construct a Gaussian distribution on the projected color $c_p(u, v)$. The road surface verification step checks the pixels outside the reference region and classifies them as either road or non-road based on the confidence interval constructed from the Gaussian distribution. If pixel (u, v) is located in the confidence interval, then $\mathbf{\Theta}(u, v) = [0, 2\pi)$; otherwise, $\mathbf{\Theta}(u, v) = \emptyset$. Therefore, it takes $O(n)$ to compute the transformation from $\mathcal{F}_c$ to $\mathcal{V}_s$. Fig. 5(c) shows an example of the surface verification output.

A hidden problem in this method is how to guarantee the reference region is really on the road when the motorcycle is running. Fig. 6(a) illustrates a discrepancy d_r between the robot location and the reference region. It causes the planning space to be ahead of the real robot location. Therefore, even if the robot is on the road, the reference region could be outside the road on narrow turns, which can cause the failure of the algorithm. We will address this problem later in Section c.

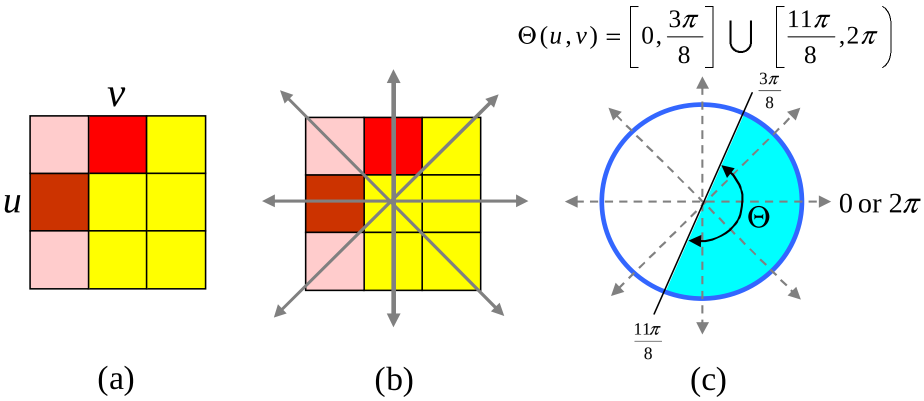

$\Theta(u,v) = \left[0, \frac{3\pi}{8}\right] \bigcup \left[\frac{11\pi}{8}, 2\pi\right)$
v
u
$\frac{3\pi}{8}$
0 or 2π
Θ
$\frac{11\pi}{8}$
(a)
(b)
(c)

Fig. 7. Directions extracted for a pixel. (a) A pixel at (u, v). (b) The similarity comparison along eight neighboring directions. (c) The extracted direction information.

c. Direction Extraction

The purpose of direction extraction is to reduce set $\mathcal{V}_s$ by extracting directional information about the road surface. Although desert roads do not have clear lane markings similar to those on the structured roads, they do contain tracks and footsteps left by previous vehicles or pedestrians. These tracks and footsteps can provide useful directional information.

To extract directional information, we must search local collision-free directions for each pixel in $\mathcal{V}_s$. A straightforward approach is to employ the pixel similarity comparison as shown in Fig. 7. Since each pixel has at most 8 neighboring pixels (Fig. 7(b)), we divide $[0, 2\pi)$ into 8 corresponding subsets. We check each direction for pixel similarity. If the neighboring pixel along one direction is statistically similar to the pixel at (u, v), we update $\boldsymbol{\Theta}(u, v)$ accordingly along that direction. Fig. 7(c) illustrates the output $\boldsymbol{\Theta}(u, v)$ for the example.

To reduce noise effects, in practice we check $5 \sim 10$ pixels along each direction. Our approach takes $O(n)$ in this step.

Remark 1. *We want to point out that the V^2-Space can be constructed at different resolutions. Although the definition of the V^2-Space in Eqs. 3.3 and 3.4 suggests the relationship* *of one-to-one correspondence between each pixel* (u, v) *and a direction set* $\Theta(u, v)$*, the* V^2-*Space can be constructed at lower resolutions. If so, a direction set* Θ *corresponds to a squared-patch of pixels instead of a single pixel. This also allows the introduction of more robust methods for the surface verification or the directional extraction such as the texture matching [51]. Constructing the* V^2-*Space at a lower resolution can greatly improve the robustness to noise at a price of the inability to distinguish small obstacles. However, if we know the vehicle has enough ground clearance, we do not need to consider small obstacles.*

Remark 2. *It is worth mentioning that the* V^2-*Space provides an open framework rather than a fixed method. If the computation power is not constrained, more sophisticated methods can be applied here. For example, we can upgrade the surface verification with the texture classification [51] instead of the straightforward Gaussian method. We can also extract the directional information for each pixel by comparing the texture information instead of just color values. More effective classifiers such as the Polynomial Mahalanobis Distance [36] can be also applied here to extract the road surface. The recent developments [24, 28, 38, 39] on machine learning for the vision-based navigation can also be applied with the* V^2-*Space.*

2. Motion Planning in V^2-Space

With the introduction of the V^2-Space, the motion planning problem for the motorcycle can be quantitatively formulated. To generate timely and accurate robot control commands, we need to consider many factors such as image processing delay and motorcycle geometric, kinematic, and dynamic limits. We begin with the motion planning in V^2-Space using a point robot without time delay and then consider the factors above to form a complete motion planning solution for the autonomous motorcycle.

a. A Point Robot with No Time Delay

Using the perspective projection mapping $\mathcal{P}$ in Eq. (3.1), we can obtain the trajectory $\mathcal{T}$ (projection of $\mathcal{T}_W$ in $\{I\}$) for a set of circular arc trajectories $(\mathbf{R}, \mathbf{d})$ (bold symbols $(\mathbf{R}, \mathbf{d})$ denotes a set of (R, d)s in $\{W\}$) as,

$$\mathcal{T} = \{(u, v) | (u, v) = \mathcal{P}(\mathbf{R}, \mathbf{d}), \mathcal{P} : \text{projection map}\}. \tag{3.9}$$

We need to evaluate $\mathcal{T}$ in $\mathcal{V}$ to obtain an obstacle-free trajectory. Assuming $\mathcal{T}$ overlaps with $\mathcal{V}$ at pixel (u, v), the direction α at pixel (u, v) of the trajectory is,

$$\alpha(u, v) = \text{atan2}\left(\Delta u, \Delta v\right), \quad (u, v) \in \mathcal{T}. \tag{3.10}$$

For a candidate arc $(R, d) \in (\mathbf{R}, \mathbf{d})$, we can calculate α and then evaluate the trajectory by checking how well it fits in $\mathcal{V}$. We define a road following quality (RFQ) function $f(u, v; R, d)$,

$$f(u, v; R, d) = \begin{cases} 0, & \text{if } \mathbf{\Theta}(u, v) = \emptyset \\ 1, & \text{if } \alpha(u, v) \in \mathbf{\Theta}(u, v) \\ |\cos(\theta_d)|, & \text{otherwise} \end{cases} \tag{3.11}$$

where $\theta_d = \inf_{\mathcal{V}} |\alpha(u, v) - \partial\mathbf{\Theta}|$ is the minimum distance between $\alpha(u, v)$ and $\mathbf{\Theta}(u, v)$, and $\partial\mathbf{\Theta}$ is the boundary of $\mathbf{\Theta}$. In other words, θ_d is the distance between a given point $\alpha(u, v)$ and a data set $\mathbf{\Theta}(u, v)$. It characterizes how close the direction $\alpha(u, v)$ is to be collision-free.

Therefore, we formulate the motion planning problem as an optimization problem: looking for a trajectory that maximizes the RFQ function,

$$\max_{\mathcal{T}} \sum_{(u,v) \in \mathcal{V}} f(u, v; R, d). \tag{3.12}$$

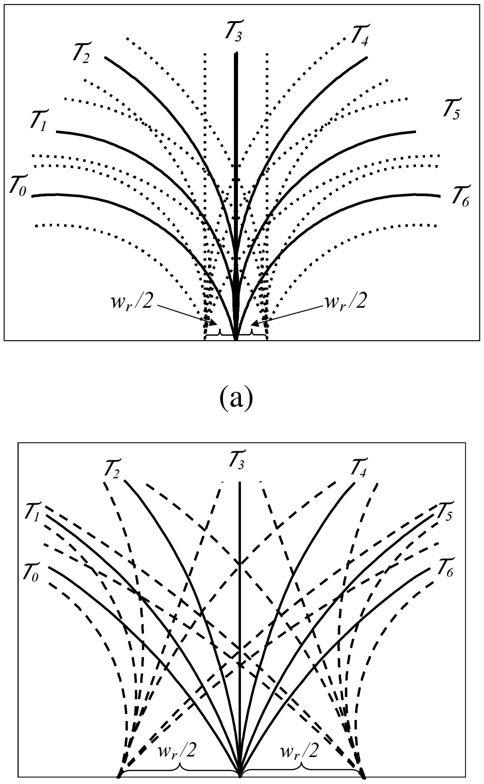
The numerical solution for Eq. (3.12) will not provide a complete obstacle-free trajectory because along $\mathcal{T}$, $f(u, v; R, d)$ could be zero if one pixel (u, v) is an obstacle. Therefore, we should impose the constraint $f(u, v; R, d) > 0,\ t \in [0, \tau_i)$. Since (R, d) uniquely defines a candidate trajectory, then we can find an obstacle-free trajectory by solving the constrained optimization problem,

$$\begin{aligned} (R, d) &= \arg\max_{\mathcal{T}} \sum_{(u,v)\in\mathcal{V}} f(u, v; R, d) \\ & \quad \text{s.t. } f(u, v; R, d) > 0. \end{aligned} \tag{3.13}$$

There are infinite number of candidate trajectories. Finding the exact optimal solution is computationally expensive and unnecessary. Due to the uncertainties in ground frictions and control errors, the motorcycle is not able to follow an "optimal" trajectory exactly. The ability of the trajectory-following determines the necessary accuracy needed in the solution. Therefore, we take an approximate approach by using a predefined candidate solution set, which is initially consisted of seven circular candidate $\mathcal{T}_0, \ldots, \mathcal{T}_6$. The number of candidates can be adjusted with respect to how accurately the vehicle can execute the trajectory. We will take the best trajectory out of the seven candidate solutions as the chosen trajectory. If none of the seven candidate arcs are obstacle-free, we can simply take the best out of the seven because it represents minimum risk of hitting a big obstacle.

Fig. 8 illustrates the seven candidate arcs $\mathcal{T}_0, ..., \mathcal{T}_6$ in the solution space. Candidate arcs $(\mathbf{R}, \mathbf{d})$ are defined in the world coordinate system as illustrated in Fig. 8(a) and the projected image in Fig. 8(b).

After a trajectory (R, d) is chosen, we need to generate its velocity profile $v^p(t)$ along the circular trajectory. We have to consider several factors. First, the motorcycle cannot run too fast for a given trajectory radius R. If we assume the road surface can provide a constant maximum lateral friction force, for a given turning radius R, the maximum allowable velocity $\bar{v}$ to balance the vehicle has to satisfy $\bar{v}(R) = k_f \sqrt{R}$, where the constant k_f



(a)

(b)

Fig. 8. Sample candidate arc trajectories and vehicle boundaries in (a) the world coordinate system $\{W\}$ and (b) the image coordinate system $\{I\}$. The solid arcs are candidate trajectories while dashed arcs are augmented boundaries of the vehicle that characterize the size of the vehicle and the size/location of the reference region defined in Fig. 6. Each solid arc has two corresponding dashed arcs.

is determined by road/tire interaction properties [47]. On the other hand, we also constrain the motorcycle velocity to be faster than its slowest velocity $\underline{v}$ for stability requirements.

Our current approach is to choose a velocity $v(\tau_i)$ at time τ_i and to perform linear interpolation for $v^p(t),\ t \in [0, \tau_i)$. Recall that $\tau_i < \tau$ is the moment that the next planning iteration starts. Bounded velocity $v(\tau_i)$ depends on the quality of road ahead,

$$v(\tau_i) = \min\{\underline{v} + \frac{(\bar{v} - \underline{v})}{S} \sum_{\substack{(u,v)\in\mathcal{V} \\ t\in[\tau_i,\tau)}} \frac{f(u,v;R,d)}{|(u,v) \in \mathcal{V}, t \in [\tau_i, \tau)|}, \bar{v}\}, \tag{3.14}$$

where

$$S = \frac{\sum_{(u,v)\in\mathcal{V},t\in[0,\tau]} f(u,v;R,d)}{|(u,v)\in\mathcal{V},t\in[0,\tau]|}$$

indicates the quality of the whole trajectory using RFQ per pixel in $[0,\tau]$, and term

$$\frac{1}{S}\sum_{\substack{(u,v)\in\mathcal{V}\\ t\in[\tau_i,\tau)}} \frac{f(u,v;R,d)}{|(u,v)\in\mathcal{V},t\in[\tau_i,\tau]|} = s_1$$

indicates relative quality of the road ahead. If RFQ per pixel of the trajectory in $[\tau_i,\tau]$ is better than that of $[0,\tau]$, then $s_1 > 1$ and we take the maximum feasible speed, which is $\bar{v}$, at time τ_i. Otherwise, we reduce the speed to be conservative.

Remark 3. *The constraint in Eq. (3.13) insists a completely obstacle free trajectory in solution. Since the vehicle has some ground clearance, it is worth mentioning that the constraint can be relaxed according to the threshold of the obstacle size.*

b. Incorporating GPS Information

Recall that the GPS input is a direction angle ϕ that points to the next way point. We also need to evaluate each trajectory using ϕ. For $\mathcal{T}$, we have its starting location $(x(0),y(0))$ and the location right before the next iteration $(x(\tau_i),y(\tau_i))$. The overall direction θ_τ in $(0,\tau)$ is,

$$\theta_\tau = \text{atan2}(x(\tau)-x(0),y(\tau)-y(0)).$$

The weight of each trajectory $w(\mathcal{T})$ is based on how much θ and ϕ agree with each other,

$$w(\mathcal{T}) = \cos(\theta_\tau - \phi). \tag{3.15}$$

Therefore, we can use $w(\mathcal{T})$ as a weighted factor of the road following function $f(u,v;R,d)$ in Eq. (3.13) to calculate the best candidate trajectory. It is worth mentioning that this addition actually incorporates the GPS way point navigation capability into our algorithms.

When there are no clear road features, the weight introduced here dominates the output of the algorithm and it becomes a GPS way-point navigation algorithm. This increases the robustness of the algorithm to deal with the situation when the vehicle runs off the road.

c. Vehicle Size and Image Processing Delay

Define l_r and w_r as motorcycle length and width, respectively. To guarantee that the reference region in Fig. 6(b) is on the road, we augment the real motorcycle by adding the reference region with discrepancy distance d_r as part of the robot geometric model. Therefore, l_r and w_r are actually larger than the real robot size.

We also augment the trajectory evaluation to the neighboring regions of the candidate trajectory. Fig. 8 illustrates the neighboring region in dashed arcs. For the trajectory in Eq. (3.6) that starts at $(x(0), y(0))$ in $\{W\}$, the upper envelope of the neighboring region is a concentric arc that starts at $(x(0) + w_r/2, y(0))$ with radius $R + w_r/2$,

$$\mathcal{T}_W^{+}(\tau) = \left\{ \left(R + \frac{w_r}{2}, d \right) |\text{starting at } (x(0) + \frac{w_r}{2}, y(0)) \right\} \tag{3.16}$$

and similarly the lower envelope is

$$\mathcal{T}_W^{-}(\tau) = \left\{ \left(R - \frac{w_r}{2}, d \right) |\text{starting at } (x(0) - \frac{w_r}{2}, y(0)) \right\}. \tag{3.17}$$

With $\mathcal{T}_W^{+}(\tau)$ and $\mathcal{T}_W^{-}(\tau)$, we can compute their projection $\mathcal{T}^{+}$ and $\mathcal{T}^{-}$ using Eq. (3.9). Define $\mathbf{T}$ as the pixels between $\mathcal{T}^{+}$ and $\mathcal{T}^{-}$, which is the set of pixels in the neighboring region of $\mathcal{T}$. $\mathbf{T}$ can be understood as the sweeping region in $\{I\}$ of the augmented motorcycle. For the ith candidate arc $\mathcal{T}_i$, $0 \leq i \leq 6$, its corresponding sweeping region is defined as $\mathbf{T}_i$. We can then modify Eq. (3.12) to incorporate the vehicle size

$$\max_{\mathbf{T}_i} \sum_{(u,v) \in \mathcal{V}} f(u, v; \mathbf{T}_i). \tag{3.18}$$

With the augmentation, RFQ function f also needs to be updated. Recall that f is the function of the angular distance $\alpha(u, v)$ between the tangent line of the candidate arc and the direction set $\Theta(u, v)$ in Eq. (3.11). Eq. (3.10) illustrates how to compute $\alpha(u, v)$ if (u, v) overlaps with the candidate arc. However, a pixel(u, v) might not be located exactly on the candidate arc since we introduced $\mathbf{T}$. For pixels that are not on the candidate arc, $(u, v) \in \mathbf{T} \cap \bar{\mathcal{T}}$, we use a pseudo candidate arc to compute $\alpha(u, v)$. The pseudo candidate arc is an circular curve that shares the same center with $\mathcal{T}$ in $\{W\}$.

Image capturing, processing, communication, and the robot control all take time, and these actions result in a time delay. Such a delay can be further classified as a measurement delay and a decision/execution delay. Measurement delay t_m refers to the elapsed time from the moment that the camera captures a frame to the moment that RGB data enter computer memory. Decision/execution delay t_d refers to the interval between the moment that the system takes the frame from memory to the moment the robot actually executes the resulting control command from the algorithm output.

Assuming $t = 0$ at the beginning of each iteration, motion planning is then based on the frame captured t_m time ago and the command generated will be executed t_d time later. To address such a time discrepancy, we can compensate for the time delay by shifting the starting location of the planned trajectory to its actual location at t_d. Fig. 9 illustrates how to compensate for the delay. Without loss of generality, we assume that $\{W\}$ has its origin at the center of the lower edge of the camera field of view. Then the last known position with respect to $\mathcal{V}$ is $(x(-t_m), y(-t_m)) = (0, -d_r)$, where d_r is the discrepancy distance illustrated in Fig. 6. Therefore, we can estimate $(x(t_d), y(t_d))$ by taking velocity integrals over time period between $-t_m$ and t_d. The estimated $((x(t_d), y(t_d))$ is the new starting point of trajectory $\mathcal{T}_W(\tau)$.

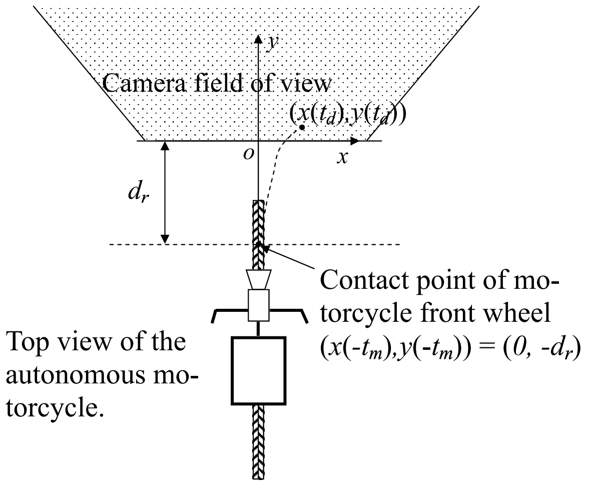

Camera field of view
y
(x(t_d),y(t_d))
o
x
d_r
Contact point of motorcycle front wheel
(x(-t_m),y(-t_m)) = (0, -d_r)
Top view of the autonomous motorcycle.

Fig. 9. A schematic of delay compensation.

3. Algorithm

Combining the analysis above, we have motion planning Algorithm 1. It is clear that the overall algorithm runs in $O(n)$ time.

Algorithm 1: Motion Planning Algorithm

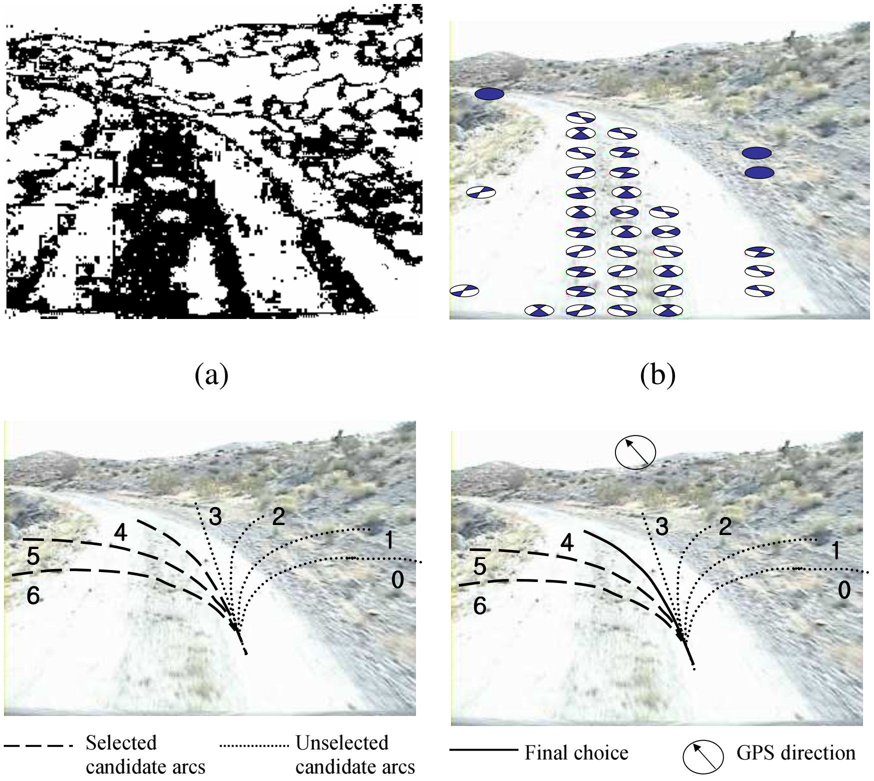
```
input  : An image from the camera, current vehicle velocity, and GPS information
output: Collision-free trajectory $\mathcal{T}_W(\tau)$
Construct the collision-free vector space $\mathcal{V}$;                                  $O(n)$
/* Follow the block diagram in Fig. 4.                                   */
Compute new trajectory start point $(x(t_d), y(t_d))$;                             $O(1)$
/* Based on the current vehicle velocity and the known
   time delays (See Fig. 9 for details).                                 */
Compute $\mathcal{T}_W^+$ and $\mathcal{T}_W^-$ for each candidate arc $\mathcal{T}_i$;                        $O(1)$
/* Using Eqs. (3.16) and (3.17) and the new starting
   point $(x(t_d), y(t_d))$.                                             */
Project $\mathcal{T}_W^+$ and $\mathcal{T}_W^-$ of each $\mathcal{T}_i$ and all $\mathcal{T}_i$'s into the image frame;   $O(1)$
/* Using the perspective projection in Eq. (3.1).  This
   allows us to get $\mathbf{T}_i$ and tag each pixel in $\mathbf{T}_i$.        */
for each $\mathcal{T}_i, 0 \leq i \leq 6$, do
    Compute the GPS weighting factor $w(\mathcal{T}_i)$;                             $O(1)$
    /* Using Eq. (3.15).                                                 */
    Initialize objective function: $F_i = 0$;                                 $O(1)$
    for each pixel $(u, v)$ in $\mathbf{T}_i$ do
        $F_i = F_i + w(\mathcal{T}_i) f(u, v, R, d)$;                              $O(1)$
        /* RFQ function $f$ can be computed according to
           Eq. (3.11).                                                   */
$\mathcal{T} = \arg\max_{\mathcal{T}_i; 0 \leq i \leq 6} F_i$ ;                                   $O(1)$
Generate velocity profile $v^p(t)$ with dynamics constraints ;            $O(n)$
/* Using Eq. (3.14).                                                     */
```

E. Experiments and Results

We have implemented the algorithm on a laptop PC with a 1.6 *GHz* Centrino processor and 512 *MB* RAM. The camera used is a Canon VCC4 camera with a 47.5° horizontal field of view. Based on Microsoft Direct X SDK version 9.0, our algorithm can run with an input from either a live video from the camera or pre-recorded video clips. Our algorithm can process the video at a speed of 5 frames per second.

1. Experiments with Video Clip

The first step of the experiments is to test the algorithm using the video data from the route of DARPA Grand Challenge. Fig. 10 illustrates the algorithm using one of the snapshots in the two hour video clip. Fig. 10(a) illustrates the results of surface verification. Black pixels represent regions that look close to the road surface. It is clear that the data are very noisy because the difference between the road and its surrounding environment is not significant. However, after directional information is extracted, the resulting $\mathcal{V}$ in Fig. 10(b) is quite a good fit of the real road (we use the circular direction to indicate the direction information at each pixel). One thing that was not mentioned early in the paper is that we do not process saturated pixels because they do not contain much information. This can also filter out the bright sky background and improve computation speed. This explains why there are so few vectors in Fig. 10(b). Even there are fewer vectors in Fig. 10(b), the overall $\mathcal{V}$ is sufficient for motion planning. Table I and Fig. 10(c) illustrate the result of candidate arcs evaluation without GPS inputs. The vision algorithm ranks three top choices including arcs 3, 4, and 5. Fig. 10(d) show how a GPS signal is used to identify the final choice. Fig. 11 uses two more examples to further illustrate how a GPS signal can be used to improve the quality of the output of the vision algorithm. Note that the starting points of the arcs in all examples are calculated with the considerations of vehicle kinematic constraints and time-delays.



(a) (b)

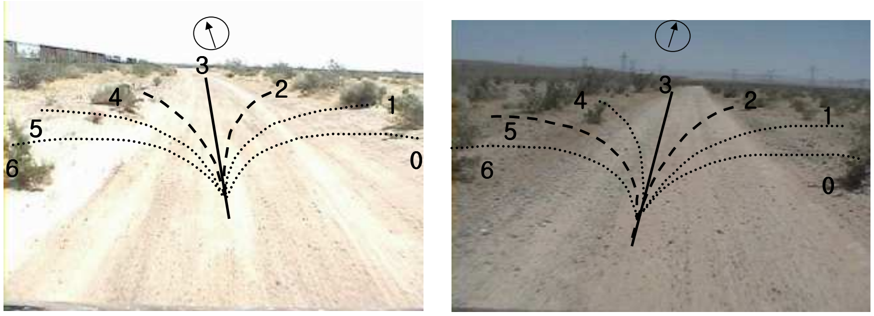
(c) (d)

Fig. 10. An illustration of V^2-Space algorithm using a snapshot of the video clip captured in the Mojave desert. (a) $\mathcal{V}_s$, (b) $\mathcal{V}$, (c) result of the arc evaluation using Eq. (3.18), and (d) final choice of the arc with GPS inputs.

During the test, we found that the algorithm has a successful classification rate of 91%. The successful rate is computed as follows: we extract each frame and its corresponding GPS data from the video clip and log file as input to run our algorithm. At the same time, human inputs, which pick the best candidate arc out of the seven candidate arcs, are used as ground truth. If the algorithm output matches human inputs for a given frame, then it is a success frame. When we classify success frames, more than one candidate arc can be the correct answer due to the road trends. In this case, all possible candidate arcs to satisfy road trends are regarded as the ground truth. These ambiguous cases are classified as successful case.

Table I. The evaluation of candidate arcs with respect to the RFQ functions in Eq. (3.18) for the examples in Figs. 10 and 11.

Fig.	0	1	2	3	4	5	6
10	271.2	129.6	88.0	147.3	717.1	512.2	286.4
11(a)	556.7	524.0	1079.0	1303.8	1330.3	475.4	321.3
11(b)	794.4	960.7	1346.8	1031.3	922.2	1105.6	794.8



(a) (b)

Fig. 11. Two examples using the video data from the Mojave desert.

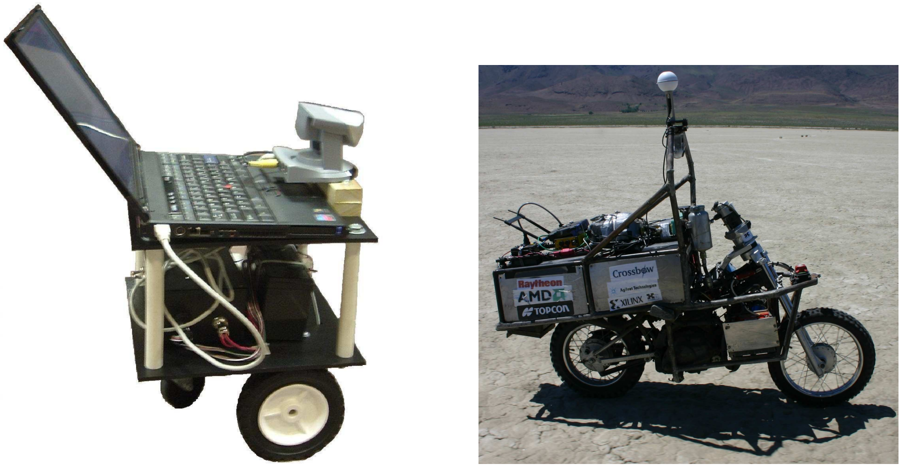
The testing DVD video lasts 23 minutes and 21 seconds. We have verified the algorithm on total 7005 frames. It outputs 6375 successful frames. Therefore, the rate of success is 91%. It is worth mentioning that the result does not mean that the vehicle will fail in the rest 9% of the time. Because there is a significant overlap between adjacent frames, it needs several consecutive failures to lead the vehicle to leave the road. Therefore, the actual performance should be much better than the successful rate on individual frames.

For example, if a road is 3 m in width, a motorcycle traveling speed of 20 km/h is equivalent to 5.555 m/s. Saying that the turn radius of the vehicle has to be limited to more than 5 m due to its dynamics and friction constraints. If the vehicle is in the middle of road, it is 3/2=1.5 m away from the boundary of the road. Given a 5 m turn radius and a velocity of 5.555 m/s, it takes at least 0.716 seconds to hit the road boundary. Given the fact that the motion planning algorithm runs every 200 milliseconds, a duration of more than 0.716 seconds means at least 3 iterations of our motion planning algorithm. It only takes one successful trial out of the three trials for the vehicle to remain on the road. If each trial has a probability of failure of 0.1, the probability of three consecutive failures is $0.1^3 = 0.001$ if each trial is completely independent. Due to the overlap between images, the three runs is not completely independent. The true failure rate is somewhere between 0.1 and 0.001. Of course, the reality is more complicated than this simple computation. However, it reveals the facts that (1) the successful rate metric in our experiment is a conservative metric and (2) the vehicle can survive better if the algorithm can run at a faster speed.

The failure cases tend to happen at the moment when the road surface changes drastically. For example, if part of the road is covered by water, the algorithm cannot distinguish the water from obstacles, which is expected because our existing implementation does not take the sophisticated surface classification into account. Another problem is caused by the inherent limitations of monocular vision. If the surface of the road is identical to the surface of an obstacle, the algorithm fails because it cannot tell the difference.

2. Field Tests

Interestingly, field tests have shown that our algorithm also works with structured environments. Before we tested the algorithm on a running motorcycle, we ran it on a smaller mobile robot. We conducted experiments in a golf course, local parks, and on the university campus, in which both the structured and ill-structured environments exist. The smaller robot is a three wheel robot with two front driving wheels and one rear caster as illustrated in Fig. 12(a). The robot is 30 cm wide and 45 cm tall and can travel at a speed of 25 cm/s with a 25 lbs. payload. It is also equipped with two wheel encoders and a digital compass, which is used to simulate GPS inputs.



(a) (b)

Fig. 12. Robots used for field tests.

Following the same successful rate metric, the robot can make correct decisions at a rate of 92%, which is better than the video clip results because the road conditions are less difficult. We believe this performance can be further improved if our robot had a better wheel encoder. The wheel encoders used in the robot only have a resolution of 32 pulses per revolution, which limits the accuracy of location estimation.

F. Summary

In this chapter, we reported the development of a vision-based navigation system based on road appearance information for a mobile robot equipped with a single camera. To present road features, we propose new type of framework, V^2-Space, which process video data and perform motion planning efficiently due to fast construction. Since the reduction of the impact of varying lighting conditions is one of the major challenges of navigation systems in a outdoor environment, we used a shadow and illumination invariant color model in V^2-Space construction. We extracted directional information from the prior tire tracks and pedestrian footsteps on the road to refine our V^2-Space. We also suggested a motion planning algorithm in V^2-Space. The proposed algorithm considers vehicle kinematic and dynamic limits as well as time-delays during trajectory generation. We conducted experiments to confirm the algorithm both with video clips from the desert and a three-wheel robot. Experimental results showed that robot motion commands are correct at a rate of more than 90%. Failures resulted from the limitations of the appearance information based navigation, such as a lack of depth information. We noticed that depth information is required to avoid obstacles efficiently. Therefore, we expanded the vision-based navigation with geometric information of roads.

CHAPTER IV

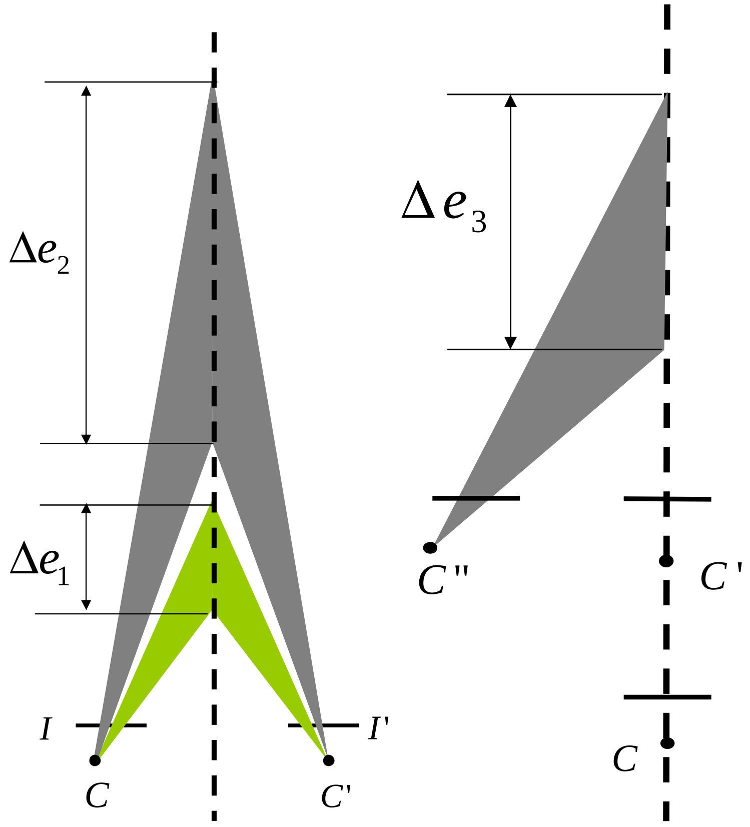
MONOCULAR VISION-BASED DEPTH-ERROR-AWARE ROBOT NAVIGATION

A. Introduction

Vision-based navigation is very important for small and fast-moving mobile robots. Most other navigation sensors, such as sonar rings, laser range finders, and radars, are active sensors, which are usually bulky and not energy-efficient. As a passive sensor, cameras can be very small and energy-efficient because cameras do not emit signals to the surrounding environment. Unlike laser range finders, cameras do not have eye-safe problem and can be used in populated regions. However, images from cameras contain rich information of the environment. Understanding the imaging data is nontrivial. Extracting geometry information from images is critical for obstacle avoidance. Stereo vision approaches are often employed.

As illustrated in Fig. 13, there are two popular types of stereo vision systems. The first type employs multiple cameras with fixed baseline distances. Images taken by different cameras at the same moment are used for stereo reconstruction. The binocular vision system shown in Fig. 13(a) is its representative. This approach has an inherent drawback when the robot gets smaller and moves faster. For a fast moving robot, it is very important to identify obstacles at a distance. Depth error, which is the error of the distance from the robot to the obstacles along robot forwarding direction, characterizes the quality of the stereo information for robot navigation. As illustrated in Fig. 13(a), the depth error range increases very fast ($\Delta e_2 > \Delta e_1$) if the depth of the obstacle gets larger and larger than the baseline distance, which is the distance between the two camera centers C and C'. This is due to the nature of triangulation computation.

Since excessive depth errors can lead to collisions with obstacles, we focus on the



(a) Binocular Vision (b) Structure from Motion

Fig. 13. A 2D conceptual illustration of depth error distribution for two types of vision-based navigation systems. The dash line in each figure is the robot moving direction. The short line segments I and I' represent the imaging planes for camera C and C', respectively. The shaded regions in the figures represent the conic regions generated by back-projecting corresponding pixels from the camera centers to 3D space. The regions determine the depth error ranges.

second type. Commonly referred to as structure from motion (SFM) approach [52], this method constructs depth information using images taken at different perspectives from a single camera [19, 53]. The robot motion changes camera perspectives. Therefore, the baseline distance is no longer limited by the width of the robot. However, this approach has its own problems. The depth of obstacles located at the baseline cannot be obtained because the camera centers and obstacle locations are collinear. Unfortunately, if the robot moves along a straight line, its forward direction is always the baseline direction. Fig. 13(b) for camera perspectives C and C' illustrates the degenerated case. If ignored, the robot will inevitably collide with obstacles.

Therefore, additional perspectives (i.e. camera C'' in Fig. 13(b)) that deviate from the motion direction must be introduced. However, the choice of the additional perspective is a tradeoff among the quality of the sensed information, navigable region limitation, and the energy consumed by the additional travel. To address this problem, we analyze how depth error is distributed on the road plane for a given frame pair. Our model can predict how the regions with excessive depth error are distributed across the joint coverage of multiple views and hence enable us to choose optimal locations to take the additional frame to effectively reduce the overall depth error. We have implemented the algorithm and tested using a three-wheel mobile robot. The experiment results confirm our analysis.

The rest of the paper is organized as follows. We review the related work in Section B. We define the problem in Section C. We analyze depth error and introduce the notion of the untrusted area in Section D. We then propose an overall algorithm in Section E. Experiments and a summary in Section F and G, respectively.

B. Related Work

Our research is related to monocular vision systems for robots, structure from motion (SFM) [52], and active vision [54–56].

Due to its simple configuration, using a monocular vision system is very popular in mobile robots with space and power constraints. The research work in this category can be classified into two types including simultaneous localization and mapping (SLAM) and vision-based navigation. SLAM [57–60] research focuses on the mapping and localization aspect and is often used in structured indoor environments where there are no global positioning system (GPS) signals to assist robots in navigation. SLAM focuses on identifying and managing landmarks/feature points from the scene for map building and localization. Obstacle avoidance is not the concern of SLAM.

Our work is in the category of monocular vision-based navigation that focuses on obstacle detection and avoidance. Due to the inherent difficulty in understanding the environment using monocular vision, many researchers focus on applying machine learning techniques to assist navigation [28, 61–63]. However, those methods are appearance-based and only utilize color and texture. Lack of geometry information limits their ability in obstacle detections.

Our work is a geometry-based approach that uses SFM to obtain geometry information of the environment. SFM can simultaneously estimate both the 3D scene and camera motion information [52]. Since the camera motion information is usually available from on-board sensors such as an inertial measurement unit (IMU) or wheel encoders, the dimensionality of the SFM problem can be reduced to the estimation of the 3D scene only, which is triangulation. The depth error is determined by the image correspondence error and the camera perspectives. To obtain the 3D information, it is necessary to find the corresponding points between the overlapping images. However, due to the fact that images are discrete representations of the environment and the inherent difficulty in image matching, it is unavoidable that matching errors are introduced into the corresponding points [64, 65]. There are many newly developed techniques that can reduce correspondence errors. Such techniques include low-rank approximations [66–68], power factorization [69], closure constraints [70], and covariance-weighted data [71]. In addition, new features, such as planar parallax [72–75] and the probability of correspondence points [76], can be used instead of correspondence points to reduce the correspondence error.

Our work accepts the fact that image correspondence cannot be eliminated completely. We are instead interested in how the depth error is affected by the image correspondence error. Although the magnitude of the image correspondence error is uniformly distributed across image coverage [64, 65], the variance of depth error is not uniformly distributed across the image coverage [77]. Therefore, new robot navigation and camera motion planning should take the depth error distribution information into account and this is the inspiration for our development.

Selection of the appropriate camera perspectives can reduce the impact of image correspondence errors and minimize the depth error [52]. This can be viewed as an "active vision" approach. Introduced by Bajscy [54] and Aloimonos et al. [78, 79], active vision is defined as "an intelligent data acquisition process using optimized camera parameters". Active vision techniques have been widely used in 3D reconstruction [80, 81]. Active vision-based systems determine optimal camera configurations either by maximizing camera visibility [81–86] or by minimizing the 3D estimation error [52, 87–89]. Estimation error is often expressed as a covariance matrix of the 3D estimation error based on the assumption that measurement data have Gaussian distribution [90–92] or a standard deviation of depth error [93]. Error analysis provides the lower boundary of the estimated 3D structure and the camera motion matrix [94, 95], or creates a sensitivity map [96] which shows the estimation error distribution and uses to correct the depth estimation.

Our problem differs from the existing "active vision" research in two aspects. First, the mobile robot does not care about the accuracy of the entire 3D environment; it only cares about the accuracy of the region for next navigation period. There is no need to compute an optimal solution to minimize the 3D reconstruction error. The demanding speed requirement in navigation does not allow it, either. Therefore, we decide to develop a fast depth error range prediction mechanism for next navigation period and plan navigation accordingly. Second, the camera cannot arbitrary select perspectives to minimize 3D reconstruction errors. This is not a free camera placement problem as in [87, 88] because the robot's primary task is navigation instead of the 3D reconstruction.

C. Problem Description

1. Coordinate Systems

Our algorithm runs every τ_0 time. In each period, the robot has a trajectory $T(\tau)$, $\tau \in [0, \tau_0]$. The period length τ_0 is a preset parameter depending on the speed of the robot and the computation time necessary for stereo reconstruction. The most common approach to assist robot navigation is to take a frame $\underline{F}$ at $\tau = 0$ and another frame $\overline{F}$ at $\tau = \tau_0$ for the two-view stereo reconstruction. As a convention, we use underline and overline with variables to indicate their correspondence to $\underline{F}$ and $\overline{F}$, respectively. To clarify the problem, we introduce the following right hand coordinate systems as illustrated in Fig. 14.

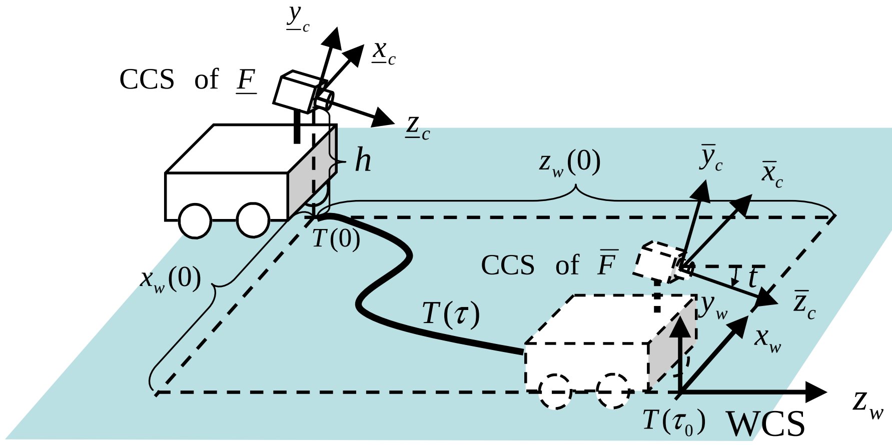


Fig. 14. Definition of coordinate systems and their relationship. The WCS is a fixed coordinate system while a CCS is attached to the moving camera.

- World coordinate system (WCS): A fixed 3D Cartesian coordinate system. Its y-axis is the vertical axis, and its x-z plane is the road plane. Trajectory $T(\tau)$ is located in x-z with $T(\tau_0)$ located at the origin of WCS. Hence, $T(\tau) = [x_w(\tau), z_w(\tau)]^T, 0 \leq \tau \leq \tau_0$ as illustrated in Fig. 15.

- Camera coordinate system (CCS): A 3D Cartesian coordinate system that is attached to a camera mounted on a robot with its origin at the camera optical center. Its z-axis coincides with the optical axis and points to the forward direction of the robot. Its x-axis and y-axis are parallel with the horizontal and vertical directions of the CCD sensor plane, respectively.

- Image coordinate system (ICS): A 2D image coordinate system with the u-axis and v-axis parallel with the horizontal and vertical directions of an image, respectively. Its origin is located at its principle point. Coordinates u and v are discretized pixel readings. When we mention frames such as F, $\underline{F}$ and $\overline{F}$, they are defined in ICS.

Frames such as $\underline{F}$ and $\overline{F}$ have their corresponding CCSs and ICSs. We use the notation $CCS(F)$ to represent the corresponding CCS for frame F. As illustrated in Fig. 14, the origin of $CCS(\overline{F})$ projects to $T(\tau_0)$ on the road plane, which is the origin of WCS. The vertical distance between the origins of the $CCS(\overline{F})$ and the WCS is the camera height h. The origin of $CCS(\underline{F})$ projects to $T(0)$ on the road plane.

2. Nomenclature

- $q = (u, v, 1)^T$: a point in ICS.
- $Q = (x_w, y_w, z_w)^T$: q's position in WCS.
- $\hat{Q} = (\hat{x}_w, \hat{y}_w, \hat{z}_w)^T$: the estimated value of Q through SFM.
- $Q_c = (x_c, y_c, z_c)^T$: q's position in CCS.
- $e = \hat{z}_w - z_w$: depth error.
- Δe: depth error range.
- A_u : the untrusted area in WCS.

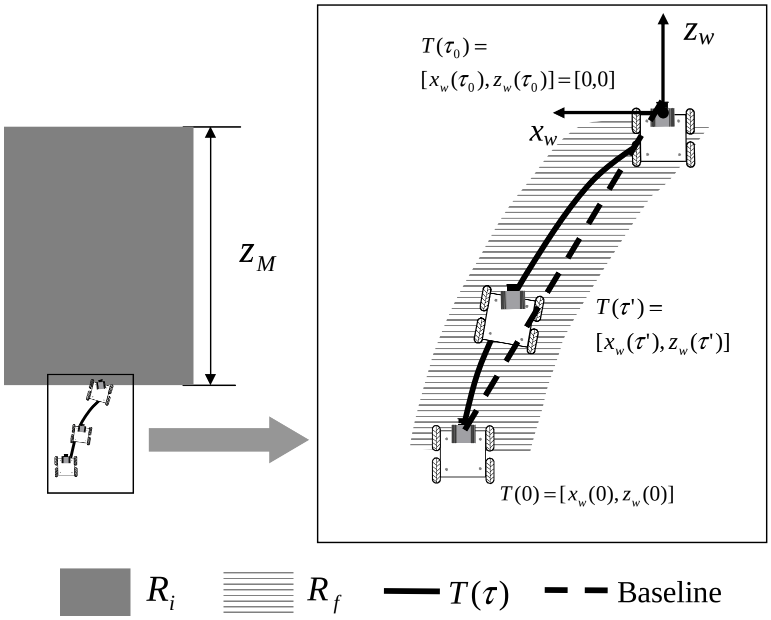
- $T(\tau)$: the navigation trajectory in $\tau \in [0, \tau_0]$.
- R_f : the obstacle free road region around $T(\tau)$.
- R_i : region of interest.
- $\underline{F}$: a frame taken at $T(0)$.
- $\overline{F}$: a frame taken at $T(\tau_0)$.
- F : a frame taken at $T(\tau')$, $0 < \tau' < \tau_0$.

3. Assumptions

- We assume that obstacles in the environment are either static or slow-moving. Therefore, the SFM algorithm can be applied to compute the depth information.
- We assume that intrinsic camera parameters, such as focal length, lens distortion, and CCD sensor size, have been obtained from pre-calibration, and that extrinsic camera parameters, such as camera position and orientational parameters, can be measured using camera angular potentiometers and robot motion sensors. The camera has squared pixels and zero skew factors.
- The robot takes frames periodically for the stereo reconstruction. During each period, we assume that the road surface can be approximated by a plane, which is the x-z plane of WCS as illustrated in Fig. 14.
- We assume that the pixel correspondence error across different frames is uniformly distributed in the ICS. We believe that the pixel correspondence errors do not have an infinite tail distribution in reality and the uniform distribution is a conservative description of the property.

- To simplify the analysis, we assume that the imaging planes of the camera are parallel to each other in the period $[0, \tau_0]$. Although the robot may have different positions and orientations when taking images, we can control the camera pan-and-tilt to ensure that image planes are always parallel to each other within the period. Therefore, all CCSs are iso-oriented with $CCS(\overline{F})$, which is determined by the navigation direction at time τ_0.

4. Problem Context



$T(\tau_0) =$
$[x_w(\tau_0), z_w(\tau_0)] = [0,0]$
z_w
x_w
z_M
$T(\tau') =$
$[x_w(\tau'), z_w(\tau')]$
$T(0) = [x_w(0), z_w(0)]$
R_i
R_f
$T(\tau)$
Baseline

Fig. 15. A robot moves in region R_f to take frames to construct the depth information for region R_i to plan its navigation in R_i. The accuracy of the depth is determined by camera positions when the robot moves in R_f.

a. Frames and Frame Parameters

For frames such as $\underline{F}$ and $\overline{F}$, we need to define their corresponding robot locations and camera parameters. As illustrated in Fig. 14, the camera is mounted at a height of h. Hence the camera position is uniquely defined by its coordinates (x_w, h, z_w) in WCS. Actually (x_w, z_w) determines the baseline distance. In order to have a good coverage of the road, the camera usually tilts towards the ground as illustrated in Fig. 14. The tilt angle is defined as t.

b. Obstacle-free Region

As illustrated in Fig. 15, the previous period provides an obstacle-free road region R_f. The robot needs to stay in R_f and reach $T(\tau_0)$ at the end of the current period. Therefore, the trajectory $T(\tau)$ is not fixed and we have the freedom to adjust it in R_f to reduce the depth error.

c. Region of Interest

A camera frame usually covers a wide range, from adjacent regions to an infinite horizon. For navigational purposes, the robot is not interested in regions that are too far away. As illustrated in Fig. 14, the z-axis of WCS points to the robot's forward direction at time $\tau = \tau_0$ when frame $\overline{F}$ is taken. z_M is defined as the maximum distance that the robot cares about in the next iteration of the algorithm. As illustrated in Fig. 15, the region of interest R_i is as subset of camera coverage,

$$R_i = \{(x_w, z_w) | 0 \le z_w \le z_M, (x_w, z_w) \in \Pi(\overline{F})\}, \tag{4.1}$$

where x_w and z_w are defined in WCS and $\Pi(\overline{F})$ is the coverage of $\overline{F}$ in x-z plane of WCS. We want to reduce the depth error e associated with objects in R_i.

Our research problem is to plan the robot motion in R_f to reduce depth error for R_i before the robot moves and before the actual stereo construction. To study how the depth error is distributed on the road plane, we introduce the *untrusted area* below.

5. Untrusted Area

The computed depth information is not accurate due to the image correspondence error. According to our assumptions, for a given pixel in $\underline{F}$, the corresponding pixel in $\overline{F}$ can be found with an error that is uniformly and independently distributed. Hence, the depth error e is also a random variable with a range, which is defined as the depth error range $|\Delta e|$. Both e and $|\Delta e|$ will be formally defined later. We adopt $|\Delta e|$ as the metric to characterize the quality of the depth information. e_t is the pre-defined threshold for $|\Delta e|$. To facilitate robot navigation, we want to ensure that $|\Delta e| \leq e_t$.

Although the image correspondence error is uniformly and independently distributed in ICS, the influence of the image correspondence error on e is non-uniform due to a non-linear stereo reconstruction process. For the two camera frames $\underline{F}$ and $\overline{F}$ taken from two different camera perspectives, we can construct the depth map for the overlapping regions of the two frames $\Pi(\underline{F} \cap \overline{F})$, where function $\Pi(\cdot)$ refers to the coverage of the camera frame. We define the untrusted area A_u in WCS as

$$A_u(\underline{F}, \overline{F}) = \{(x_w, z_w) | (x_w, z_w) \in \Pi(\underline{F} \cap \overline{F}),\ |\Delta e(x_w, z_w)| > e_t\}, \tag{4.2}$$

because we know that the depth information in A_u is untrustworthy due to the excessive $|\Delta e|$.

6. Problem Definition

For any finite e_t, A_u cannot be eliminated for any two-frame stereo pair due to the degeneracy in triangulation computation. Therefore, the robot has a high probability of colliding with obstacles in $A_u(\underline{F}, \overline{F})$. An immediate solution to this problem is to capture one more frame F at time τ', $0 < \tau' < \tau_0$ such that two additional image pairs $(\underline{F}, F)$ and $(F, \overline{F})$ can be generated[1]. The right hand side of Fig. 15 illustrates this idea. We can obtain two more versions of the 3D information along with two untrusted areas $A_u(\underline{F}, F)$ and $A_u(F, \overline{F})$. The final 3D information can be selected as that with the least $|\Delta e|$ provided by the three image pairs. Therefore, we know that the overall

$$|\Delta e| \leq e_t \iff A_u(\underline{F}, \overline{F}) \cap A_u(\underline{F}, F) \cap A_u(F, \overline{F}) = \emptyset. \tag{4.3}$$

Computing 3D information for all three pairs is time-consuming. In practice, we can reduce the amount of computation by only computing two image pairs $(\underline{F}, \overline{F})$ and $(F, \overline{F})$. Therefore, (4.3) becomes,

$$|\Delta e| \leq e_t \iff A_u(\underline{F}, \overline{F}) \cap A_u(F, \overline{F}) = \emptyset, \tag{4.4}$$

which means that $|\Delta e| \leq e_t$ as long as $A_u(\underline{F}, \overline{F})$ and $A_u(F, \overline{F})$ do not overlap.

As shown later, the location of A_u is a function of the camera parameters and the robot positions. We know that the robot can choose $T(\tau)$ in R_f. This gives us the flexibility to choose the parameters for F to manipulate the position of $A_u(F, \overline{F})$. Since the camera parameters are fixed, the decision variables are $(x_w(\tau'), z_w(\tau'))$, where the robot takes F as illustrated in Fig. 15. There may be multiple $(x_w(\tau'), z_w(\tau'))$s that satisfy the non-overlapping condition in (4.4). Let Φ be the set of all possible solutions. We know that Φ might not exist due to road conditions. In this case, we would like to minimize the area of $A_u(\underline{F}, \overline{F}) \cap A_u(F, \overline{F})$. Therefore, the predictive depth-error-aware robot navigation problem becomes,

Definition 2. *Given e_t, R_f, and R_i and camera parameters, compute Φ if $\Phi \neq \emptyset$. Otherwise, compute the $(x_w(\tau'), z_w(\tau'))$ that minimizes $A_u(\underline{F}, \overline{F}) \cap A_u(F, \overline{F})$.*

---

[1] It is possible to use multiple views simultaneously for stereo reconstructions. However, this is too computationally expensive for real time navigation.

D. Analysis of Depth Error

Def. 2 describes a planning problem based on A_u. However, A_u depends on the depth error range. Let us analyze the depth error and derive the method so that we can predict A_u before the robot makes its move and takes frames.

1. Computing Depth from Two Views

In stereo vision, 3D information is computed through triangulation under the perspective projection based on the extracted correspondence points from each pair of images [97]. Define $\underline{c}$ and $\overline{c}$ be camera centers for frames $\underline{F}$ and $\overline{F}$, respectively. Define $\underline{P}$ and $\overline{P}$ as the camera projection matrix for $\underline{F}$ and $\overline{F}$, respectively. Since the CCSs of $\underline{F}$ and $\overline{F}$ are iso-oriented and only differ from WCS by a tilt value t in orientation, the orientation of WCS with respect to CCSs can be expressed by a rotation matrix

$$R_X(-t) = \begin{bmatrix} 1 & 0 & 0 \\ 0 & c(t) & s(t) \\ 0 & -s(t) & c(t) \end{bmatrix}.$$

Note that we use $s(\cdot)$ and $c(\cdot)$ to denote $\sin(\cdot)$ and $\cos(\cdot)$, respectively. This denotation is used in the rest of the paper. If CCSs are not iso-oriented as assumed, it is not difficult to extend the rotation matrix using Euler angle sets. The origin of WCS with respect to CCSs of $\underline{F}$ and $\overline{F}$ are defined as $\underline{W}$ and $\overline{W}$, respectively. Since $T(0) = [x_w(0), z_w(0)]^T$, $T(\tau_0) = [0, 0]^T$, and the camera height is h, Fig. 14 shows that the camera center positions with respect to WCS are $\underline{c} = [x_w(0), h, z_w(0)]^T$ and $\overline{c} = [0, h, 0]^T$, respectively. Then we have,

$$\underline{W} = -R_X(-t)\underline{c}, \text{ and } \overline{W} = -R_X(-t)\overline{c}.$$

Therefore,

$$\underline{P} = K[R_X(-t)|\underline{W}],\ \overline{P} = K[R_X(-t)|\overline{W}],$$

$$K = diag(f, f, 1),$$

where f is the focal length of the camera divided by the side length of a pixel. Let $\underline{q} = [\underline{u}\ \underline{v}\ 1]^T$ and $\overline{q} = [\overline{u}\ \overline{v}\ 1]^T$ be a pair of corresponding points in $\underline{F}$ and $\overline{F}$, respectively. Define $Q = [x_w, y_w, z_w]^T$ as their corresponding point in WCS. Recall that $\underline{Q}_c = [\underline{x}_c\ \underline{y}_c\ \underline{z}_c]^T$ and $\overline{Q}_c = [\overline{x}_c\ \overline{y}_c\ \overline{z}_c]^T$ are Q's position in CCSs of $\underline{F}$ and $\overline{F}$, respectively. Also, we know that $\underline{Q}_c$ and $\overline{Q}_c$ can be expressed as,

$$\underline{Q}_c = R_X(-t)Q + \underline{W},\ \text{and}\ \overline{Q}_c = R_X(-t)Q + \overline{W}. \tag{4.5}$$

Then the following holds according to the pin-hole camera model,

$$\underline{q} = \frac{1}{\underline{z}_c}\underline{P}\begin{bmatrix} Q \\ 1 \end{bmatrix} = \frac{1}{\underline{z}_c}K\underline{Q}_c, \tag{4.6}$$

and

$$\overline{q} = \frac{1}{\overline{z}_c}\overline{P}\begin{bmatrix} Q \\ 1 \end{bmatrix} = \frac{1}{\overline{z}_c}K\overline{Q}_c. \tag{4.7}$$

From (4.5), we know $\underline{Q}_c = \overline{Q}_c + (\underline{W} - \overline{W})$, which means,

$$\begin{cases} \underline{x}_c = \overline{x}_c - x_w(0) \\ \underline{y}_c = \overline{y}_c - z_w(0)s(t) \\ \underline{z}_c = \overline{z}_c - z_w(0)c(t) \end{cases} \tag{4.8}$$

Plug it into (4.6), we get,

$$\underline{q} = \frac{1}{\overline{z}_c - z_w(0)c(t)}K(\overline{Q}_c + (\underline{W} - \overline{W})). \tag{4.9}$$

Plug (4.7) into (4.9),

$$\underline{q} = \frac{1}{\overline{z}_c - z_w(0)c(t)} \left(\overline{z}_c \overline{q} + K(\underline{W} - \overline{W}) \right). \tag{4.10}$$

Since K, $\underline{W}$, $\overline{W}$, $\underline{q}$, and $\overline{q}$ are known, (4.10) consists of a system of equations with $\overline{z}_c$ as an unknown quantity. There is one unknown variable and a total of two equations since the last row of (4.10) is $1 = 1$. This is an overly-determined equation system. A typical approach would be to apply a least square method [97]. Another method is to simply discard one equation and solve it directly. This method has a speed advantage and is employed by our design. Hence, we have

$$\overline{z}_c = \frac{x_w(0)f - \underline{u} z_w(0)c(t)}{\overline{u} - \underline{u}}. \tag{4.11}$$

From (4.5) and (4.11), we know,

$$z_w = \overline{z}_c \left(\frac{\overline{v}}{f} s(t) + c(t) \right). \tag{4.12}$$

Hence,

$$z_w = \frac{x_w(0)f - \underline{u} z_w(0)}{\overline{u} - \underline{u}} \left(\frac{\overline{v}}{f} s(t) + c(t) \right). \tag{4.13}$$

Depth z_w describes the distance from the robot to an obstacle along the z-axis of WCS. Its error directly affects the robot's collision avoidance performance.

2. Estimating the Depth Error Range

For a given pair of corresponding points ($\underline{q}$, $\overline{q}$) from ($\underline{F}$, $\overline{F}$) with camera centers ($\underline{c}$, $\overline{c}$), the triangulation process works as follows. If we back project a ray from $\underline{c}$ through $\underline{q}$, it intersects with the ray generated by back-projecting from $\overline{c}$ through $\overline{q}$, provided both $\underline{q}$ and $\overline{q}$ are accurate. The intersection point in the 3D space is Q.

However, for a given point $\underline{q}$, finding the accurate $\overline{q}$ is unlikely due to noises and pixelization errors. According to our assumptions, the corresponding errors in $\overline{u}$ and $\overline{v}$

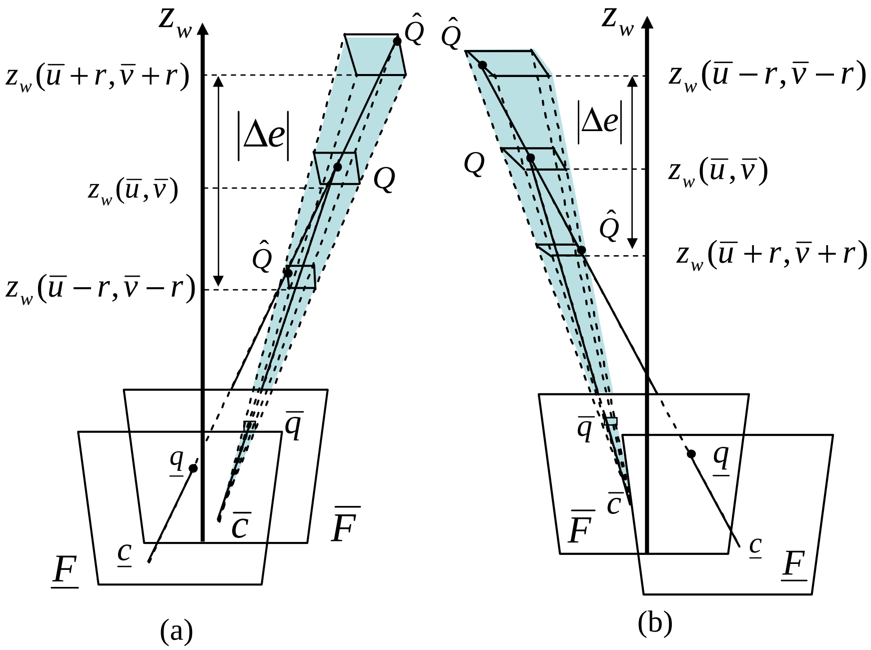


Fig. 16. An illustration of depth error caused by the image correspondence error in $\overline{F}$. The intersection zone between the ray from $\underline{q}$, and the pyramid from $\overline{q}$ is the error range. If the error range projects onto the z axis, it is always bound between $z_w(\overline{u}+r,\overline{v}+r)$ and $z_w(\overline{u}-r,\overline{v}-r)$. (a) and (b) illustrate two subcases.

are independently distributed according to $U(-r,r)$, where r is usually 0.5-2 pixels in length. This means that $\overline{q}$ is distributed in a small square on $\overline{F}$. When we back projects the square, it forms a pyramid in 3D space as illustrated in Fig. 16. When the pyramid meets the ray that is back-projected from $\underline{q}$, it has a range of intersections instead of a single point. The estimated depth z_w is a function of random variables $(\overline{u},\overline{v})$ and can be expressed as $z_w(\overline{u},\overline{v})$. It is apparent that z_w is a random variable that could take any value in this intersection zone. The maximum length of the intersection zone is defined as $|\Delta e|$,

$$|\Delta e| = |z_w(\overline{u}+r,\overline{v}+r) - z_w(\overline{u}-r,\overline{v}-r)|. \tag{4.14}$$

$|\Delta e|$ describes the range of the depth error and is employed as the metric to measure the quality of the stereo reconstruction. To simplify the notation in computing Δe, we define the following intermediate variables for (4.13).

$$\lambda = \beta\overline{v} + c(t), \;\; \zeta_d = \overline{u} - \underline{u}, \;\; \beta = \frac{s(t)}{f}. \tag{4.15}$$

$$\zeta_n = x_w(0)f - \underline{u}z_w(0)c(t), \tag{4.16}$$

then $z_w = \lambda\zeta_n/\zeta_d$ according to (4.13), (4.15) and (4.16). Plug them in to (4.14), and we have,

$$\begin{aligned}\Delta e &= (\lambda + r\beta)\frac{\zeta_n}{\zeta_d + r} - (\lambda - r\beta)\frac{\zeta_n}{\zeta_d - r} \\ &= \zeta_n \frac{2r(\beta\zeta_d - \lambda)}{\zeta_d^2 - r^2}.\end{aligned} \tag{4.17}$$

Eq. (4.17) illustrates Δe in ICS. For robot navigation purpose, we are interested in Δe in x-z plane of WCS. Hence $\underline{u}$, $\overline{u}$ and $\overline{v}$ in (4.17) should be transformed into functions of x_w and z_w. From (4.7), (4.12), and (4.15), we know

$$\overline{u} = \frac{x_w f}{\overline{z}_c} = f\lambda\frac{x_w}{z_w}, \tag{4.18}$$

and $y_w = \left(\frac{\overline{v}}{f}c(t) - s(t)\right)\overline{z}_c + h$. Since we are interested in obstacles on the $x - z$ plane, $y_w = 0$, we have,

$$\overline{v} = \frac{f(z_w s(t) - hc(t))}{z_w c(t) + hs(t)}. \tag{4.19}$$

Similarly, from (4.6), (4.12), and (4.15), we know

$$\underline{u} = \alpha_x x_w + \alpha_0, \tag{4.20}$$

where

$$\alpha_x = \frac{f}{\overline{z}_c - z_w(0)c(t)} = \frac{f\lambda}{z_w - z_w(0)c(t)\lambda},$$

and $\alpha_0 = -x_w(0)\alpha_x$. Plug (4.18), (4.19), and (4.20) into (4.15) and (4.16), and we can derive the intermediate variables $\lambda,\ \zeta_n$, and ζ_d, in terms of x_w and z_w.

$$\lambda = \beta\overline{v} + c(t) = \frac{z_w}{z_w c(t) + hs(t)}. \tag{4.21}$$

$$\zeta_n = x_w(0)f - \underline{u}z_w(0)c(t) = n_x x_w + n_0, \tag{4.22}$$

where $n_x = -z_w(0)c(t)\alpha_x$ and $n_0 = x_w(0)z_w\alpha_x/\lambda$.

$$\zeta_d = \overline{u} - \underline{u} = \frac{n_x\lambda}{z_w}x_w + \frac{n_0\lambda}{z_w}. \tag{4.23}$$

Plug intermediate variables in (4.21), (4.22), and (4.23) into (4.17), and we can get Δe as a function of x_w and z_w,

$$\Delta e = \frac{2r\beta\lambda z_w(n_x x_w + n_0)^2 - 2r\lambda z_w^2(n_x x_w + n_0)}{\lambda^2(n_x x_w + n_0)^2 - r^2 z_w^2}. \tag{4.24}$$

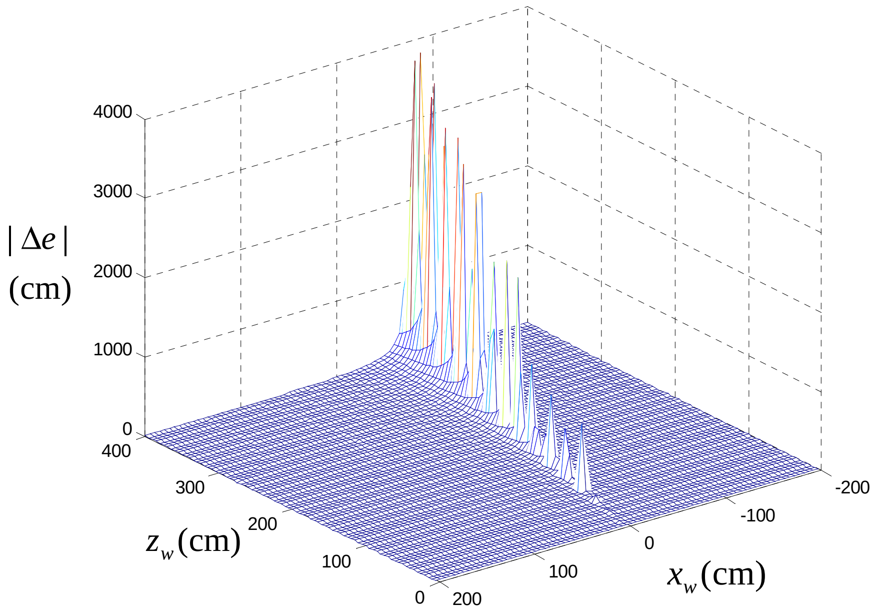


Fig. 17. An illustration of Δe. Robot positions are set to be $x_w(0)$ =10 cm, $z_w(0)$ =-50 cm. The data and parameters are based on a Canon VCC4 camera.

For an obstacle located at $(x_w, 0, z_w)$, (4.24) allows us to estimate Δe. Fig. 17 illustrates how Δe is distributed on the road plane $y_w = 0$. It is clear that the depth error range varies dramatically in different regions and should be considered in robot navigation to avoid obstacles.

3. Predicting Untrusted Area

For a given frame pair with the corresponding robot locations, we can partition R_i using a preset depth error threshold $e_t > 0$. For the region satisfying $|\Delta e| < e_t$, the results from stereo reconstruction are trustable and can be used for motion planning later. However, the *untrusted area* A_u should be avoided. We now ready to predict A_u by computing its boundary using Eq. (4.24).

a. Partition R_i According to the Sign of Δe

To find the regions corresponding to $|\Delta e| < e_t$, there are two possible cases to consider: $\Delta e < 0$ and $\Delta e > 0$. We can rewrite (4.24) as,

$$\Delta e = \frac{2r\lambda z_w(x_w - \mu_{n1})(x_w - \mu_{n2})}{(x_w - \mu_{d1})(x_w - \mu_{d2})}, \tag{4.25}$$

where

$$\begin{aligned}
\mu_{n1} &= \frac{x_w(0)}{z_w(0)\lambda c(t)} z_w, \\
\mu_{n2} &= \frac{x_w(0)}{z_w(0)\lambda c(t)} z_w - \frac{z_w(z_w - z_w(0)\lambda c(t))}{f z_w(0)\lambda\beta c(t)}, \\
\mu_{d1} &= \frac{x_w(0)}{z_w(0)\lambda c(t)} z_w + \frac{r z_w(z_w - z_w(0)\lambda c(t))}{f z_w(0)\lambda^2 c(t)}, \\
\mu_{d2} &= \frac{x_w(0)}{z_w(0)\lambda c(t)} z_w - \frac{r z_w(z_w - z_w(0)\lambda c(t))}{f z_w(0)\lambda^2 c(t)}.
\end{aligned}$$

Recall that t is the camera tilt angle and a typical camera setup has $0 \leq t \leq 30°$. A regular camera would have a focal length of 5-100 mm and pixel side length of 5-10 μm. Therefore,

$f \geq 100$. Since $\beta = s(t)/f$,

$$0 < \beta \leq \sin(30^\circ)/100 \approx 0.005. \tag{4.26}$$

Also we know that

$$\lambda = \beta\overline{v} + c(t) = s(t)\frac{\overline{v}}{f} + c(t) > \beta \tag{4.27}$$

because $|\overline{v}/f| < 1$ for any camera with a vertical field of view less than 90°. Combining this information, we have $0 < \beta < r/\lambda$ and $\beta < \lambda$. For obstacles in R_i, $z_w > 0$ according to the definition of WCS. Also $z_w(0) < 0$ as illustrated in Fig. 14. Hence, we have

$$\frac{z_w(z_w - z_w(0)\lambda c(t))}{f z_w(0)\lambda c(t)} < 0. \tag{4.28}$$

Combining the inequalities above, we can derive the following relationship,

$$\mu_{d1} < \mu_{n1} < \mu_{d2} < \mu_{n2}. \tag{4.29}$$

Combine this with (4.25), and we have,

$$\Delta e > 0 \text{ if } \mu_{n1} < x_w < \mu_{d2} \text{ or } x_w < \mu_{d1}, \tag{4.30}$$

$$\Delta e < 0 \text{ if } \mu_{d2} < x_w < \mu_{n2} \text{ or } \mu_{d1} < x_w < \mu_{n1}. \tag{4.31}$$

We ignore the region $x_w > \mu_{n2}$ in $\Delta e > 0$ as this region is always outside of the camera's coverage. We are now ready to compute A_u for the two cases defined in (4.30) and (4.31).

b. Computing A_u for $\Delta e > 0$

This is the case illustrated in Fig. 16(a). Recall that the untrusted area is the region that satisfies $\Delta e > e_t$. It is worth mentioning that the error threshold e_t is usually not a fixed number but a function of z_w. We define $e_t = \rho z_w$ where ρ is the relative error threshold and $0 < \rho < 1$. There are two sub cases including case (i): $x_w < \mu_{d1}$ and case (ii): $\mu_{n1} < x_w < \mu_{d2}$.

Case (i): when $x_w < \mu_{d1}$, the denominator of Δe in (4.25) is positive. Plug (4.25) into $\Delta e > e_t$, and we have

$$\begin{aligned}
&(e_t\lambda^2 - 2r\beta\lambda z_w)n_x^2 x_w^2 + \\
&(2(e_t\lambda^2 - 2r\beta\lambda z_w)n_x n_0 + 2r\lambda n_x z_w^2)x_w + \\
&(e_t\lambda^2 - 2r\beta\lambda z_w)n_0^2 - e_t r^2 z_w^2 + 2r\lambda n_0 z_w^2 < 0.
\end{aligned} \tag{4.32}$$

The solution to the quadratic inequality(4.32) is,

$$\frac{-\kappa_1 - \sqrt{\kappa_1^2 - 4\kappa_2\kappa_0}}{2\kappa_2} < x_w < \frac{-\kappa_1 + \sqrt{\kappa_1^2 - 4\kappa_2\kappa_0}}{2\kappa_2}, \tag{4.33}$$

where

$$\begin{aligned}
&\kappa_2 = (e_t\lambda^2 - 2r\beta\lambda z_w)n_x^2, \\
&\kappa_1 = 2(e_t\lambda^2 - 2r\beta\lambda z_w)n_x n_0 + 2r\lambda n_x z_w^2, \\
&\kappa_0 = (e_t\lambda^2 - 2r\beta\lambda z_w)n_0^2 - e_t r^2 z_w^2 + 2r\lambda n_0 z_w^2.
\end{aligned}$$

The untrusted area is the region that satisfies (4.33) and $x_w < \mu_{d1}$. To compute the intersection, we need to understand the relationship between the solution in (4.33) and the coefficients in (4.25). Combining them we know,

$$\mu_{d1} - \frac{-\kappa_1 - \sqrt{\kappa_1^2 - 4\kappa_2\kappa_0}}{2\kappa_2} = \frac{r z_w(z_w - z_w(0)\lambda c(t))}{f z_w(0)\lambda^2 c(t)} \left(1 - \frac{\lambda + \sqrt{\lambda^2 + \rho^2\lambda^2 - 2r\beta\lambda\rho}}{\rho\lambda - 2r\beta}\right).$$

We know that $0 < r \leq 2$, $0 < \rho < 1$, β is very small according to (4.26), and $\lambda > 0$ according to (4.27). Therefore, $2r\beta$ and $2r\beta\lambda\rho$ are close to zero. Hence, we approximate

$$\left(1 - \frac{\lambda + \sqrt{\lambda^2 + \rho^2\lambda^2 - 2r\beta\lambda\rho}}{\rho\lambda - 2r\beta}\right) \approx 1 - \frac{2}{\rho} < 0.$$

Combining this equation with (4.28), we know,

$$\mu_{d1} > \frac{-\kappa_1 - \sqrt{\kappa_1^2 - 4\kappa_2\kappa_0}}{2\kappa_2}. \tag{4.34}$$

Similarly, we can obtain

$$\mu_{d1} < \frac{-\kappa_1 + \sqrt{\kappa_1^2 - 4\kappa_2\kappa_0}}{2\kappa_2}. \tag{4.35}$$

According to (4.33), (4.34), (4.35), and $x_w < \mu_{d1}$, the untrusted area for this subcase is

$$\frac{-\kappa_1 - \sqrt{\kappa_1^2 - 4\kappa_2\kappa_0}}{2\kappa_2} < x_w < \mu_{d1}. \tag{4.36}$$

Case (ii): when $\mu_{n1} < x_w < \mu_{d2}$. From (4.29), we know that the denominator of (4.25) is negative. Hence,

$$\kappa_2 x_w^2 + \kappa_1 x_w + \kappa_0 > 0. \tag{4.37}$$

The solution to (4.37) is

$$x_w < \frac{-\kappa_1 - \sqrt{\kappa_1^2 - 4\kappa_2\kappa_0}}{2\kappa_2}, \text{or } x_w > \frac{-\kappa_1 + \sqrt{\kappa_1^2 - 4\kappa_2\kappa_0}}{2\kappa_2}.$$

According to (4.29), $x_w < (-\kappa_1 - \sqrt{\kappa_1^2 - 4\kappa_2\kappa_0})/(2\kappa_2) < \mu_{d1}$ does not satisfy $\mu_{n1} < x_w < \mu_{d2}$ and should be discarded. Hence, only

$$x_w > \frac{-\kappa_1 + \sqrt{\kappa_1^2 - 4\kappa_2\kappa_0}}{2\kappa_2} \tag{4.38}$$

contains the untrusted area. Applying the approximation that $2r\beta$ and $2r\beta\lambda\rho$ are close to zero, we obtain the following inequalities,

$$\mu_{n1} < \frac{-\kappa_1 + \sqrt{\kappa_1^2 - 4\kappa_2\kappa_0}}{2\kappa_2} < \mu_{d2}. \tag{4.39}$$

Knowing $\mu_{n1} < x_w < \mu_{d2}$, the untrusted area for this subcase is

$$\frac{-\kappa_1 + \sqrt{\kappa_1^2 - 4\kappa_2\kappa_0}}{2\kappa_2} < x_w < \mu_{d2}. \tag{4.40}$$

c. Computing A_u for $\Delta e < 0$

In this case, the untrusted area is the region that satisfies $\Delta e < -e_t$. There are also two sub cases including case (i): $\mu_{d2} < x_w < \mu_{n2}$ and case (ii): $\mu_{d1} < x_w < \mu_{n1}$.

Case (i): when $\mu_{d2} < x_w < \mu_{n2}$, the denominator of Δe in (4.25) is positive. Using $\Delta e < -e_t$, we have

$$2r\beta\lambda z_w(n_x x_w + n_0)^2 - 2r\lambda z_w^2(n_x x_w + n_0) < -e_t\lambda^2(n_x x_w + n_0)^2 - e_t r^2 z_w^2. \quad (4.41)$$

The solution to (4.41) is,

$$\frac{-\kappa_1' + \sqrt{{\kappa_1'}^2 - 4\kappa_2'\kappa_0'}}{2\kappa_2'} < x_w < \frac{-\kappa_1' - \sqrt{{\kappa_1'}^2 - 4\kappa_2'\kappa_0'}}{2\kappa_2'}, \quad (4.42)$$

where,

$$\begin{aligned}
\kappa_2' &= (-e_t\lambda^2 - 2r\beta\lambda z_w)n_x^2,\\
\kappa_1' &= 2(-e_t\lambda^2 - 2r\beta\lambda z_w)n_x n_0 + 2r\lambda n_x z_w^2,\\
\kappa_0' &= (-e_t\lambda^2 - 2r\beta\lambda z_w)n_0^2 + e_t r^2 z_w^2 + 2r\lambda n_0 z_w^2.
\end{aligned}$$

Similarly, we obtain the following relationship between the coefficients in (4.25) and the solution in (4.41),

$$\begin{aligned}
\mu_{n2} &> \frac{-\kappa_1' - \sqrt{{\kappa_1'}^2 - 4\kappa_2'\kappa_0'}}{2\kappa_2'},\\
\mu_{d2} &< \frac{-\kappa_1' - \sqrt{{\kappa_1'}^2 - 4\kappa_2'\kappa_0'}}{2\kappa_2'},\\
\mu_{d2} &> \frac{-\kappa_1' + \sqrt{{\kappa_1'}^2 - 4\kappa_2'\kappa_0'}}{2\kappa_2'}.
\end{aligned}$$

Combining the inequalities above with (4.29), (4.42), and $\mu_{d2} < x_w < \mu_{n2}$, the untrusted area for this subcase is,

$$\mu_{d2} < x_w < \frac{-\kappa_1' - \sqrt{{\kappa_1'}^2 - 4\kappa_2'\kappa_0'}}{2\kappa_2'}. \tag{4.43}$$

Case (ii): when $\mu_{d1} < x_w < \mu_{n1}$. We know that the denominator of (4.25) is negative from (4.29). Hence, we have

$$\kappa_2' x_w^2 + \kappa_1' x_w + \kappa_0' < 0$$

The solution to the equation is,

$$x_w < \frac{-\kappa_1' + \sqrt{{\kappa_1'}^2 - 4\kappa_2'\kappa_0'}}{2\kappa_2'}, \text{or } x_w > \frac{-\kappa_1' - \sqrt{{\kappa_1'}^2 - 4\kappa_2'\kappa_0'}}{2\kappa_2'}.$$

Since $x_w > (-\kappa_1' - \sqrt{{\kappa_1'}^2 - 4\kappa_2'\kappa_0'})/(2\kappa_2') > \mu_{d2}$ does not intersect with $\mu_{d1} < x_w < \mu_{n1}$, we discard it. Hence, the solution set is reduced to,

$$x_w < \frac{-\kappa_1' + \sqrt{{\kappa_1'}^2 - 4\kappa_2'\kappa_0'}}{2\kappa_2'}. \tag{4.44}$$

Similarly, we obtain the following relationship between the coefficients in (4.25) and the solution in (4.44),

$$\begin{aligned} \mu_{n1} &> \frac{-\kappa_1' + \sqrt{{\kappa_1'}^2 - 4\kappa_2'\kappa_0'}}{2\kappa_2'}, \\ \mu_{d1} &< \frac{-\kappa_1' + \sqrt{{\kappa_1'}^2 - 4\kappa_2'\kappa_0'}}{2\kappa_2'}. \end{aligned}$$

Combining the inequalities above with (4.29), (4.44), and $\mu_{d1} < x_w < \mu_{n1}$, the untrusted area for this subcase is,

$$\mu_{d1} < x_w < \frac{-\kappa_1' + \sqrt{{\kappa_1'}^2 - 4\kappa_2'\kappa_0'}}{2\kappa_2'}. \tag{4.45}$$

d. Computing the Overall A_u

The overall A_u is the union of four subcases in (4.36), (4.40), (4.43), and (4.45).

$$
\begin{aligned}
A_u = \Bigg\{ &(x_w, z_w) | 0 \le z_w \le z_M, \\
&\frac{-\kappa_1 - \sqrt{\kappa_1^2 - 4\kappa_2\kappa_0}}{2\kappa_2} < x_w < \frac{-\kappa_1' + \sqrt{{\kappa_1'}^2 - 4\kappa_2'\kappa_0'}}{2\kappa_2'} \text{ or} \\
&\frac{-\kappa_1 + \sqrt{\kappa_1^2 - 4\kappa_2\kappa_0}}{2\kappa_2} < x_w < \frac{-\kappa_1' - \sqrt{{\kappa_1'}^2 - 4\kappa_2'\kappa_0'}}{2\kappa_2'} \Bigg\}
\end{aligned}
$$

Let us observe the relationship between the two inner boundaries in A_u,

$$
\begin{aligned}
&\frac{-\kappa_1 + \sqrt{\kappa_1^2 - 4\kappa_2\kappa_0}}{2\kappa_2} - \frac{-\kappa_1' + \sqrt{{\kappa_1'}^2 - 4\kappa_2'\kappa_0'}}{2\kappa_2'} \\
&= \frac{r z_w (z_w - z_w(0)\lambda c(t))}{f z_w(0)\lambda^2 c(t)} \Bigg(\frac{\lambda - \sqrt{\lambda^2 + \rho^2\lambda^2 - 2r\beta\rho\lambda}}{\rho\lambda - 2r\beta} \\
&\quad - \frac{\lambda - \sqrt{\lambda^2 + \rho^2\lambda^2 + 2r\beta\rho\lambda}}{-\rho\lambda - 2r\beta} \Bigg) \approx 0
\end{aligned}
$$

because $2r\beta\rho\lambda \approx 0$ and $2r\beta \approx 0$. Hence, we have

$$
A_u = \Bigg\{ (x_w, z_w) | 0 \le z_w \le z_M,\ \frac{-\kappa_1 - \sqrt{\kappa_1^2 - 4\kappa_2\kappa_0}}{2\kappa_2} < x_w < \frac{-\kappa_1' - \sqrt{{\kappa_1'}^2 - 4\kappa_2'\kappa_0'}}{2\kappa_2'} \Bigg\}. \tag{4.46}
$$

Eq. (4.46) also tells us how to obtain the boundaries of A_u. Represented as a function of x_w, the lower boundary of A_u is

$$
\begin{aligned}
&x_w^l(z_w, x_w(0), z_w(0)) = \\
&\quad \frac{x_w(0) z_w}{z_w(0)\lambda c(t)} + \frac{r z_w (z_w - z_w(0)\lambda c(t))}{f z_w(0)\lambda c(t)(e_t\lambda^2 - 2r\beta\lambda z_w)} \left(z_w\lambda + \sqrt{\lambda^2 z_w^2 + e_t^2\lambda^2 - 2r e_t\beta\lambda z_w} \right).
\end{aligned} \tag{4.47}
$$

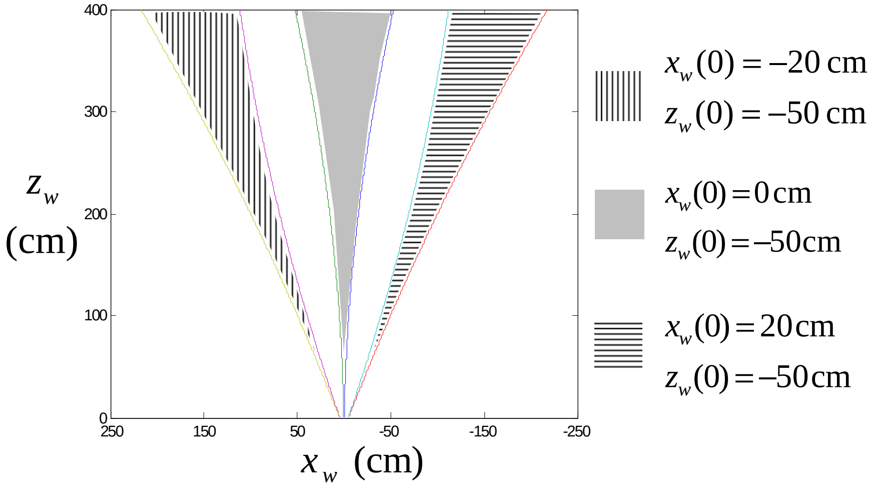


Fig. 18. A_us with different robot positions $(x_w(0), z_w(0))$. We set the threshold $e_t = 0.2z_w$.

The upper boundary of A_u is

$$x_w^h(z_w, x_w(0), z_w(0)) = \frac{x_w(0)z_w}{z_w(0)\lambda c(t)} + \frac{rz_w(z_w - c(t)z_w(0)\lambda c(t))}{fz_w(0)\lambda c(t)(-e_t\lambda^2 - 2r\beta\lambda z_w)}\left(z_w\lambda + \sqrt{\lambda^2 z_w^2 + e_t^2\lambda^2 + 2re_t\beta\lambda z_w}\right). \tag{4.48}$$

As illustrated in (4.47) and (4.48), the boundaries of A_u are a function of the depth z_w and the locations of robots. Fig. 18 gives three examples of A_u for different $(x_w(0), z_w(0))$. It is readily apparent that A_u often overlaps with the robot's forward direction. The risk of collision is high if we do not consider A_u in navigation.

E. Algorithm

The untrusted area A_u predicts how the depth error will be distributed on the road plane for a given frame pair. As we can see, $A_u(\underline{F}, \overline{F})$ always exists for any two-frame pair $\underline{F}$ and $\overline{F}$. At least one additional frame F is required if we want to avoid A_u in robot navigation. However, the new frame introduces a new untrusted area $A_u(F, \overline{F})$. Recall that F is taken at location $(x_w(\tau'), y_w(\tau'))$, which is a point in R_f as illustrated in Fig. 15. Therefore, we

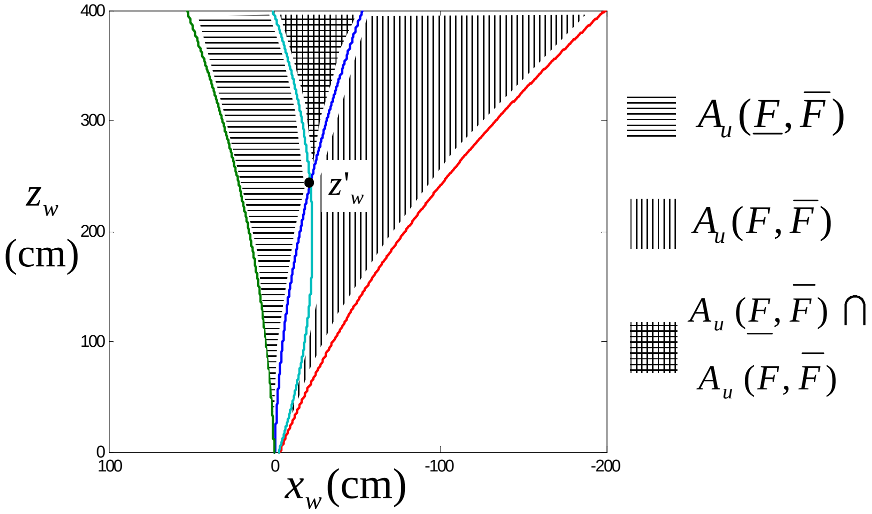


Fig. 19. $A_u(\underline{F},\overline{F})$ and $A_u(F,\overline{F})$ are partially overlapped when F is taken in $\overline{R}_{\varnothing}$.

can partition R_f according to the relationship between $A_u(\underline{F},\overline{F})$ and $A_u(F,\overline{F})$,

- $R_{\varnothing} = \{(x_w(\tau'), z_w(\tau'))|$

  $$(x_w(\tau'), z_w(\tau')) \in R_f, \text{ and } A_u(F,\overline{F}) \cap A_u(\underline{F},\overline{F}) = \emptyset\}.$$

  This is the part of R_f that allows non-overlapping A_us as illustrated in (4.4). We know that the overall depth error would be under the threshold if $R_{\varnothing}$ exists.

- $\overline{R}_{\varnothing} = R_f - R_{\varnothing}$. If a frame F is taken in this region, $A_u(\underline{F},\overline{F})$ and $A_u(F,\overline{F})$ must overlap. The depth error in the overlapped region is beyond the error threshold. We want to avoid this region if possible.

The figure on page 74 illustrates a typical example of $R_{\varnothing}$ and $\overline{R}_{\varnothing}$ on the road. $\overline{R}_{\varnothing}$ is close to the center line of R_f and has a trapezoid-like shape while $R_{\varnothing}$ is outside the trapezoid. Fig. 19 illustrates the relationship between $A_u(\underline{F},\overline{F})$ and $A_u(F,\overline{F})$ when $(x_w(\tau'), z_w(\tau')) \in \overline{R}_{\varnothing}$. We know that the depth error is beyond the threshold in the regions where $A_u(\underline{F},\overline{F})$ and $A_u(F,\overline{F})$ intersect.

The algorithm is a two-step operation. First, we establish wether $R_{\varnothing} \neq \emptyset$. If this is the case then we can use $R_{\varnothing}$ as the output since the depth error is below the threshold as long as we take an F in $R_{\varnothing}$. Therefore, $R_{\varnothing}$ is the entire solution set for $(x_w(\tau'), z_w(\tau'))$. However, $R_{\varnothing}$ does not necessarily exist as R_f may contain obstacles and the road might be narrow. If this is the case then we need to select a solution from $\overline{R}_{\varnothing}$ such that the minimal depth difference between $T(\tau_0) = (0,0)$ and the set $A_u(\underline{F}, \overline{F}) \cap A_u(F, \overline{F})$ is maximized. As an example, z'_w is the minimal depth difference in Fig. 19 . Intuitively, this strategy pushes the closest intersection points between A_us as far as possible, which reduces the possibility of collisions caused by the depth error. The point with the minimal depth difference also tells us when to schedule the next period. Now let us formally introduce this process.

1. Computing $R_{\varnothing}$

Since $R_{\varnothing} \neq \emptyset$ if and only if a location $(x_w(\tau'), z_w(\tau'))$ exists to take an F such that $A_u(\underline{F}, \overline{F}) \cap A_u(F, \overline{F}) = \emptyset$, we need a mechanism to quickly establish wether the two A_us intersect with each other. Eqs. (4.47) and (4.48) give the two boundaries of an A_u. The third boundary of A_u is $z_w = z_M$ because $A_u \subset R_i$ according to (5.1) and (4.2). Directly computing the intersection using the boundaries is not computationally efficient as it involves solving high-order polynomial equations.

Fig. 20 illustrates a quicker method of determining wether the two A_us intersect with each other. Each A_u has three vertices, which are $(0,0)$, $(x_w^l(z_M, x_w(0), z_w(0)), z_M)$, and $(x_w^h(z_M, x_w(0), z_w(0)), z_M)$. We can draw a triangle using these vertices. The following relationship is of interest.

Lemma 1. *For the image frame pair taken at* $(x_w(0), z_w(0))$ *and* $(0,0)$*, the corresponding untrusted area* A_u *is always bounded inside the triangle defined by the vertices* $(0,0)$*,* $(x_w^l(z_M, x_w(0), z_w(0)), z_M)$*, and* $(x_w^h(z_M, x_w(0), z_w(0)), z_M)$*.*

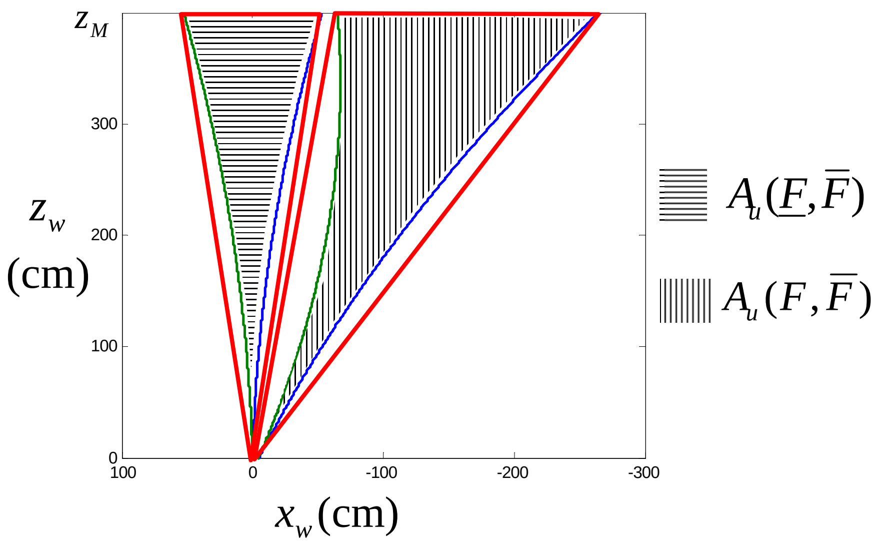


Fig. 20. The relationship between A_us and the triangles generated by connecting three points, $(x_w^l(z_M, x_w(0), z_w(0)), z_M)$, $(x_w^h(z_M, x_w(0), z_w(0)), z_M)$, and $(0, 0)$.

Proof. First, we compare the lower boundary of A_u to the corresponding triangle edge. The lower boundary of A_u described in (4.47) is the right side boundary in Fig. 20, we know this because the positive x_w axis of WCS points to the left. The side of the triangle near the lower boundary of A_u is written as

$$x_t^l(z_w, x_w(0), z_w(0)) = \frac{x_w^l(z_M, x_w(0), z_w(0))}{z_M} z_w. \tag{4.49}$$

We use the subscription t to indicate that this is the boundary function for the triangle.

Recall that $0 < \beta < 0.005$, $2r\beta \approx 0$, and $2r\beta\rho\lambda \approx 0$. Hence, (4.47) is approximated as

$$x_w^l(z_w, x_w(0), z_w(0)) \approx \frac{x_w(0)z_w}{z_w(0)\lambda c(t)} + \frac{rz_w(z_w - z_w(0)\lambda c(t))(1 + \sqrt{1+\rho^2})}{fz_w(0)\lambda^2 c(t)\rho}. \tag{4.50}$$

Plug (4.21) into (4.50), and we get

$$\begin{aligned} x_w^l(z_w, x_w(0), z_w(0)) \approx \frac{1}{z_w(0)c(t)} \Bigg[& x_w(0)(z_w c(t) + hs(t)) \\ & + \frac{r(1+\sqrt{1+\rho^2})}{f\rho} \Big((z_w c(t) + hs(t))^2 - z_w(0)c(t)(z_w c(t) + hs(t)) \Big) \Bigg]. \end{aligned} \tag{4.51}$$

When (4.51) is differentiated twice, it is written as.

$$\frac{\partial^2 x_w^l}{\partial z_w^2} = \frac{1}{z_w(0)} \left(\frac{r(1+\sqrt{1+\rho^2})}{f\rho} 2c(t) \right).$$

We know $z_w(0) < 0$ and $\frac{r(1+\sqrt{1+\rho^2})}{f\rho} > 0$. Hence,

$$\frac{1}{z_w(0)} \left(\frac{r(1+\sqrt{1+\rho^2})}{f\rho} 2c(t) \right) < 0.$$

Therefore, $x_w^l(z_w, x_w(0), z_w(0))$ is a concave function. For any $0 \leq \gamma \leq 1$, we have

$$x_w^l(\gamma z_M, x_w(0), z_w(0)) \geq \gamma x_w^l(z_M, x_w(0), z_w(0)) + (1-\gamma) x_w^l(0, x_w(0), z_w(0)).$$

Since $0 \leq z_w/z_M \leq 1$, we choose $\gamma = z_w/z_M$. Since $x_w^l(0, x_w(0), z_w(0)) = 0$, we have

$$\begin{aligned} x_w^l(z_w, x_w(0), z_w(0)) & \geq \frac{z_w}{z_M} x_w^l(z_M, x_w(0), z_w(0)) \\ & \geq x_t^l(z_w, x_w(0), z_w(0)). \end{aligned} \tag{4.52}$$

For the upper boundary of A_u and the corresponding triangle edge, we have

$$x_t^h(z_w, x_w(0), z_w(0)) = \frac{x_w^h(z_M, x_w(0), z_w(0))}{z_M} z_w.$$

The second order derivative of $x_w^h(z_w, x_w(0), z_w(0))$ is

$$\frac{\partial^2 x_w^h}{\partial z_w^2} \approx -\frac{1}{z_w(0)} \left(\frac{r(1+\sqrt{1+\rho^2})}{f\rho} 2c(t) \right) > 0. \tag{4.53}$$

Therefore, $x_w^h(z_w, x_w(0), z_w(0))$ is a convex function. Similar to the calculation of the lower boundary case, we obtain,

$$\begin{aligned} x_w^h(z_w, x_w(0), z_w(0)) &\leq \frac{z_w}{z_M} x_w^h(z_M, x_w(0), z_w(0)), \\ &\leq x_t^h(z_w, x_w(0), z_w(0)). \end{aligned} \tag{4.54}$$

From (4.52) and (4.54), we can conclude that the triangle always includes A_u because A_u and the triangle share the third boundary on line $z_w = z_M$. This completes the proof. □

From Lemma 1, we can use the relationship between the bounding triangles to determine whether the two A_us intersect. Note that the two bounding triangles share point $(0, 0)$ as one of their vertices and all other vertices are collinear, we thus have the following lemma,

Lemma 2. *The two untrusted areas $A_u(\underline{F}, \overline{F})$ and $A_u(F, \overline{F})$ overlap with each other if and only if their bounding triangles overlap with each other.*

Proof. Necessity: It is apparent that if two A_us intersect with each other then their bounding triangles must intersect with each other.

Sufficiency: As illustrated in Fig. 20, the bounding triangles share the same edge along the line $z_w = z_M$ with the corresponding A_us. Also the bounding triangles share a vertex at point $(0, 0)$. If the two bounding triangles intersect, their edges on the line $z_w = z_M$, which are two collinear intervals, must also intersect. This indicates that the two A_us must intersect. □

Lemmas 1 and 2 provide a quick means of computing $R_\varnothing$. Recall that $A_u(\underline{F}, \overline{F})$ is fixed. The two vertices of $A_u(\underline{F}, \overline{F})$ on the line $z_w = z_M$ are defined as x_M^l and x_M^h and computed using (4.47) and (4.48), respectively,

$$x_M^l = x_w^l(z_M, x_w(0), z_w(0)),$$

$$x^h_M = x^h_w(z_M, x_w(0), z_w(0)).$$

The interval $[x^l_M, x^h_M]$ defines the boundary of the bounding triangle of $A_u(\underline{F}, \overline{F})$ on line $z_w = z_M$. For a new frame F taken at $(x_w(\tau'), z_w(\tau'))$, the corresponding interval for the boundary of $A_u(F, \overline{F})$ on the line $z_w = z_M$ is $[x^l_w(z_M, x_w(\tau'), z_w(\tau'), x^h_w(z_M, x_w(\tau'), z_w(\tau')]$. Since the bounding triangles share the same vertex $(0, 0)$ and share one edge on the line $z_w = z_M$, ensuring that the bounding triangles do not intersect with each other is equivalent to ensuring that the two intervals do not overlap.

The are two ways to ensure that the two intervals do not overlap. One approach is to force,

$$x^l_M > x^h_w(z_M, x_w(\tau'), z_w(\tau')).$$

Plug (4.48) into the equation, and we have

$$x^l_M > \frac{x_w(\tau')z_M}{z_w(\tau')\lambda_M c(t)} + \frac{r z_M(z_M - z_w(\tau')\lambda_M c(t))}{f z_w(\tau')\lambda^2_M c(t)(-\rho\lambda_M - 2r\beta)}\left(\lambda_M + \sqrt{\lambda^2_M + \rho^2\lambda^2_M + 2r\rho\beta\lambda_M}\right).$$

Because $x^h_w(z_M, x_w(\tau'), z_w(\tau'))$ is a function of $x_w(\tau')$ and $z_w(\tau')$, this defines half of $R_\varnothing$. We rewrite the inequality above as,

$$x_w(\tau') > \frac{z_w(\tau')\lambda_M c(t)}{z_M} x^l_M - \frac{r(z_M - z_w(\tau')\lambda_M c(t))}{f\lambda_M(-\rho\lambda_M - 2r\beta)}\left(\lambda_M + \sqrt{\lambda^2_M + \rho^2\lambda^2_M + 2r\rho\beta\lambda_M}\right). \tag{4.55}$$

An alternative approach to ensure that the two intervals do not overlap is to force,

$$x^h_M < x^l_w(z_M, (x_w(\tau'), z_w(\tau')).$$

The inequality leads to the second half of $R_\varnothing$,

$$x_w(\tau') < \frac{z_w(\tau')\lambda_M c(t)}{z_M} x^h_M - \frac{r(z_M - z_w(\tau')\lambda_M c(t))}{f\lambda_M(\rho\lambda_M - 2r\beta)}\left(\lambda_M + \sqrt{\lambda^2_M + \rho^2\lambda^2_M - 2r\rho\beta\lambda_M}\right). \tag{4.56}$$

Fig. 22(a) illustrates the shape of $R_\varnothing$ computed from (4.55) and (4.56). It is not surprising that $\overline{R}_\varnothing$ is close to the camera baseline with less lateral movements. $\overline{R}_\varnothing$ encloses the area where the degeneracy in triangulation occurs during depth reconstruction. $\overline{R}_\varnothing$ is often close to the center of roads, while $R_\varnothing$ is close to the boundary of roads. Although (4.55) and (4.56) provide boundaries for $R_\varnothing$, the obstacles and road width identified in the previous period have not yet been applied. With these constraints, it is possible that the final $R_\varnothing = \emptyset$ and the robot must select a location in $\overline{R}_\varnothing$.

2. Minimizing the Risk of Collisions Caused by $\overline{R}_\varnothing$

Since $\overline{R}_\varnothing$ is close to the center of roads, it is likely that the robot has to stay in $\overline{R}_\varnothing$. However, we need to reduce the risk of collisions with obstacles caused by depth errors. Introducing frame F to an existing frame pair $(\underline{F}, \overline{F})$ determines $A_u(\underline{F}, \overline{F})$ and $A_u(F, \overline{F})$. Based on the A_u pair, we define the minimum depth difference z'_w as,

$$z'_w = \min_{(x_w, z_w) \in A_u(\underline{F}, \overline{F}) \cap A_u(F, \overline{F})} z_w. \tag{4.57}$$

As illustrated in Fig. 19, z'_w is the depth of the nearest point in the forward direction where the depth error is beyond the threshold. Since this position is a function of $(x_w(\tau'), z_w(\tau'))$, we can choose the best $(x_w(\tau'), z_w(\tau'))$ to maximize it in order to reduce the risk of collision,

$$(x^*_w(\tau'), z^*_w(\tau')) = \arg \max_{(x_w(\tau'), z_w(\tau')) \in \overline{R}_\varnothing} z'_w, \tag{4.58}$$

where $(x^*_w(\tau'), z^*_w(\tau'))$ is the optimal solution. As illustrated in Fig. 19, it is apparent that z'_w exists on the intersection point of the boundaries of the A_u's. There are two possibilities. One possibility is that z'_w is the intersection between the lower boundary of $A_u(\underline{F}, \overline{F})$ and the upper boundary of $A_u(F, \overline{F})$, which can be obtained by solving the following equation,

$$x^l_w(z_w, x_w(0), z_w(0)) - x^h_w(z_w, x_w(\tau'), z_w(\tau')) = 0.$$

Plug (4.47) and (4.48) into the equation and recall that $2r\beta\lambda\rho \approx 0$, and $2r\beta \approx 0$. This results in the following solution,

$$z'^{1}_{w} = \frac{f\rho(x_w(\tau')z_w(0) - x_w(0)z_w(\tau'))}{rc(t)(1+\sqrt{1+\rho^2})(z_w(0)+z_w(\tau'))} + \frac{2z_w(0)z_w(\tau')}{z_w(0)+z_w(\tau')} - \frac{hs(t)}{c(t)} \tag{4.59}$$

The other solution is to compute the the intersection between the upper boundary of $A_u(\underline{F}, \overline{F})$ and the lower boundary of $A_u(F, \overline{F})$, which can be obtained by solving the following equation,

$$x^{h}_{w}(z_w, x_w(0), z_w(0)) - x^{l}_{w}(z_w, x_w(\tau'), z_w(\tau')) = 0.$$

This gives the following solution,

$$z'^{2}_{w} = \frac{f\rho(x_w(0)z_w(\tau') - x_w(\tau')z_w(0))}{rc(t)(1+\sqrt{1+\rho^2})(z_w(0)+z_w(\tau'))} + \frac{2z_w(0)z_w(\tau')}{z_w(0)+z_w(\tau')} - \frac{hs(t)}{c(t)} \tag{4.60}$$

According to (4.57), $z'_w = \min\{z'^{1}_{w}, z'^{2}_{w}\}$. Since both (4.59) and (4.60) are continuous and differentiable functions with respect to $x_w(\tau')$ and $z_w(\tau')$, it is straightforward to solve the optimization problem in (4.58).

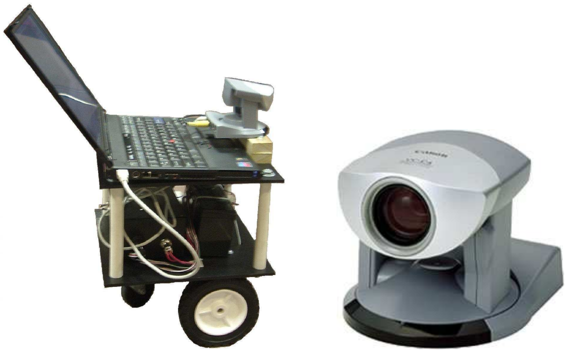
3. Depth-Error-Aware Navigation (DEAN) Algorithm

We summarize the proposed DEAN algorithm below. The algorithm is suppose to be run at $T(0)$ with $T(\tau_0)$ and R_f provided from the previous stereo construction results.

Algorithm 2: DEAN Algorithm

```
input  : Frame $\underline{F}$, $T(0)$, $T(\tau_0)$, camera tilt $t$, $R_f$, and $R_i$
output: Robot positions $(x_w(\tau'), z_w(\tau'))$ for $F$
Compute $x_M^l$ and $x_M^h$ using (4.47) and (4.48);
Compute $R_\varnothing$ boundary using (4.55) and (4.56);
Trim $R_\varnothing$ using the obstacle information from the previous stereo
reconstruction;
if $R_\varnothing \neq \emptyset$ then
    Output $R_\varnothing$ as the solution set;
else
    Compute $z_w'^1$ using (4.59);
    Compute $z_w'^2$ using (4.60);
    $z_w' = \min\{z_w'^1, z_w'^2\}$;
    Solve the optimization problem in (4.58);
    Output $(x_w^*(\tau'), z_w^*(\tau'))$ as the solution;
```

It is worth mentioning that Algorithm 2 is not complete as a robot navigation solution. The planner needs to figure out a trajectory from $T(0)$ to $T(\tau_0)$ using the results from Algorithm 2. Then the robot need to execute the trajectory to navigate to $T(\tau_0)$ and take frames F and $\overline{F}$ at $T(\tau')$ and $T(\tau_0)$, respectively. When the robot reaches $T(\tau_0)$, it needs to perform a stereo reconstruction using frames $\underline{F}$, F, and $\overline{F}$, which provides the information needed to determine $T(\tau_0)$ for the next period. Since the trajectory planning involves dynamic and kinematic constraints of the individual robot and the stereo reconstruction is a well-studied problem, they are not the focus of this paper.



(a) (b)

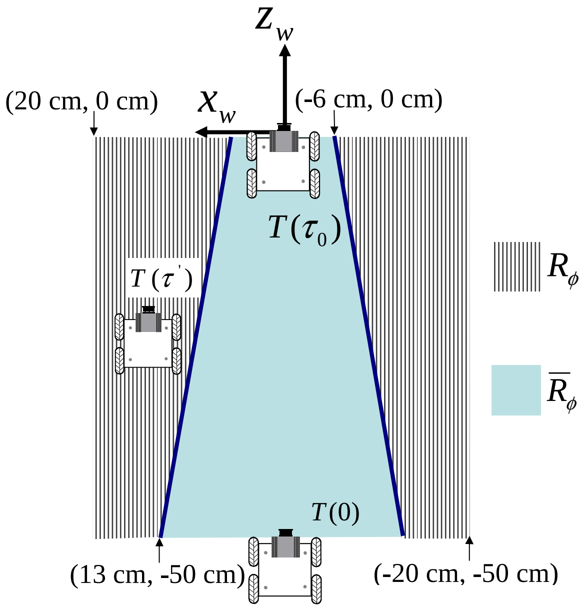
Fig. 21. The robot and the camera used in our experiments.

F. Experiments

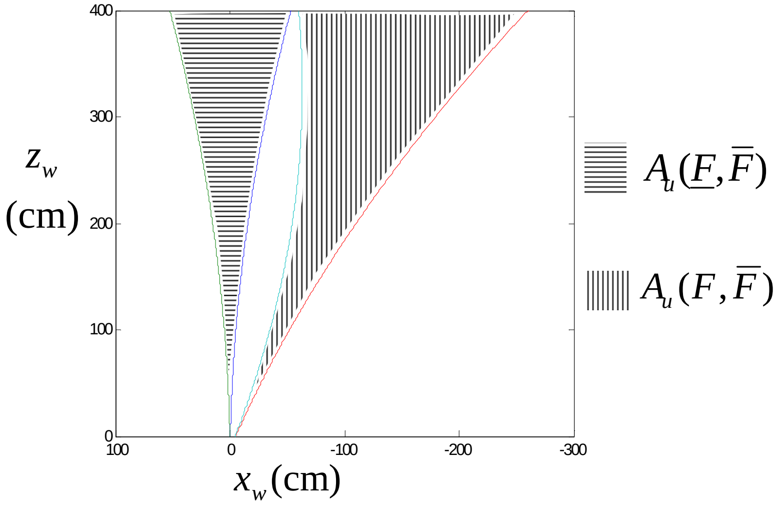
1. Software and Hardware

We have implemented the algorithm on a laptop PC with a 1.6 *GHz* Centrino processor and 512 *MB* RAM. The laptop runs Microsoft Windows XP and the algorithm has been implemented using Matlab. The laptop is mounted on a mobile robot with three wheels. As illustrated in Fig. 21(a), the robot has two front driving wheels and one rear castor. The robot is 30 cm long, 30 cm wide, and 33 cm tall and can travel at a speed of 25 cm/s with a 25 lbs. payload. It is also equipped with two wheel encoders and a digital compass. The camera used is a Canon VCC4 pan-tilt-zoom camera with a 47.5° horizontal field of view as illustrated in Fig. 21(b). The intrinsic camera parameters are estimated using the Matlab calibration toolbox [98], and the extrinsic camera parameters are measured by camera potentiometers and robot motion sensors. During the experiment, we set default $z_M = 4$ m and $t = 15^\circ$. The camera mounting height $h = 44$ cm. We conducted the experiments in the H. R. Bright Bldg. at Texas A&M University campus. The obstacles used in the experiments are books and blocks with a size of 20 cm ×14.5 cm ×10 cm.

2. Two Representative Cases



(a) $R_\varnothing$ and $\overline{R}_\varnothing$.



(b) $A_u(\underline{F}, \overline{F})$ and $A_u(F, \overline{F})$.

Fig. 22. An illustration of robot positions and the relationship between A_us when $R_\varnothing \neq \emptyset$.

We first present two representative cases: $R_\varnothing \neq \emptyset$ and $R_\varnothing = \emptyset$. When the road is relatively wide and with few obstacles, $R_\varnothing \neq \emptyset$. Fig. 22 illustrates the scenario. In this case, we have $x_w(0)$ =0 cm and $z_w(0)$ =-50 cm and

$$R_f = \{(x_w, z_w) |\ \text{-20 cm} \le x_w \le \text{ 20 cm}, \ \ z_w(0) \le z_w \le 0\}.$$

Since R_f is relatively large, $R_\varnothing \neq \emptyset$ according to (4.47) and (4.48). Fig. 22(a) illustrates how $R_\varnothing$ and $\overline{R}_\varnothing$ are distributed in R_f. We chose $(x_w(\tau'), z_w(\tau'))$ = (9.5 cm, -25 cm) as the location to take F and the resulting relationship between $A_u(\underline{F}, \overline{F})$ and $A_u(F, \overline{F})$ is shown in Fig. 22(b). As expected, those two A_us do not overlap. Note that the robot in Figs. 22 and 23 is smaller than its actual size. This is for illustration purposes.

When the road is narrow with obstacles, $R_\varnothing = \emptyset$ may occur as illustrated in Fig. 23. In this case,

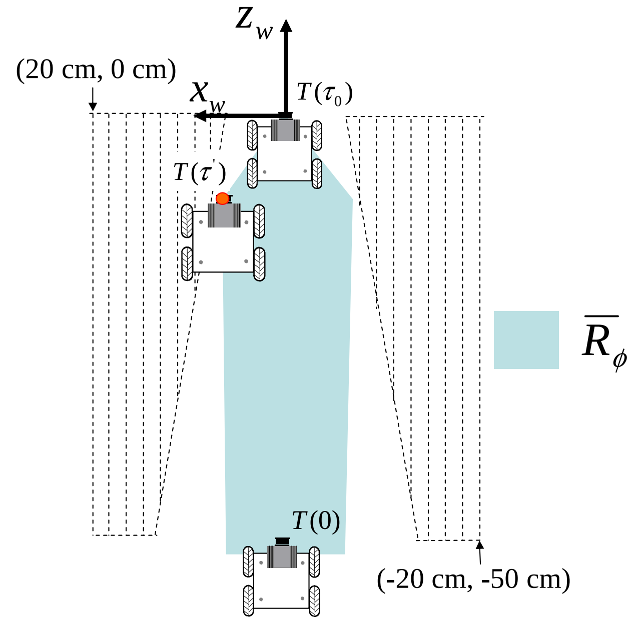
$$\begin{aligned} R_f = \{(x_w, z_w) |\ &\text{-4 cm} \le x_w \le \text{ 4 cm when -50 cm} \le z_w \le \text{ -8 cm;} \\ &\text{-4} + \frac{1}{2}(z_w + 8)\text{ cm} \le x_w \le 4 - \frac{1}{2}(z_w + 8)\text{ cm when -8 cm} \le z_w \le \text{ 0 cm}\}. \end{aligned}$$

The dashed line boxes in Fig. 23(a) represent $(x_w(\tau'), z_w(\tau'))$ that satisfy non-overlapping condition between $A_u(\underline{F}, \overline{F})$ and $A_u(F, \overline{F})$. This is located outside R_f hence $R_\varnothing = \emptyset$. Therefore, we compute $(x^*_w(\tau'), z^*_w(\tau'))$ =(4 cm, -8 cm) using (4.58) and obtain z'_w =222 cm. The position of the red dot in Fig. 23(a) is $(x^*_w(\tau'), z^*_w(\tau'))$. The corresponding z'_w and A_us are illustrated in Fig. 23(b). It is important to notice that z'_w is relatively far away from the origin of the WCS.

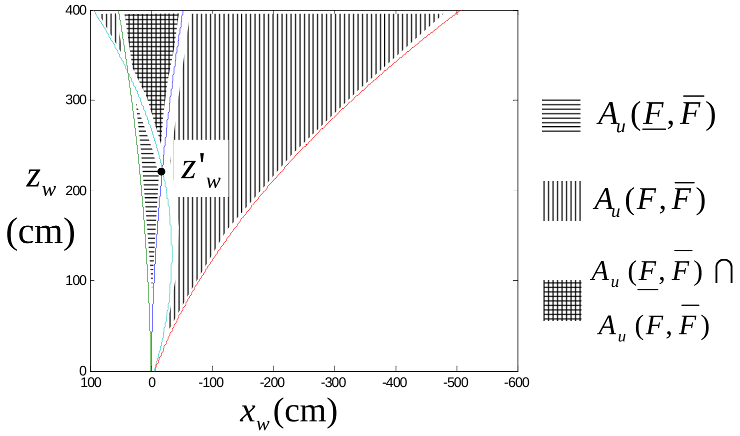
3. Depth Error Reduction Effectiveness

We also verified the effectiveness of the depth error reduction. We defined the relative depth error e_r as,

$$e_r = \frac{|\hat{z}_w - z_w|}{z_w} \times 100, \tag{4.61}$$



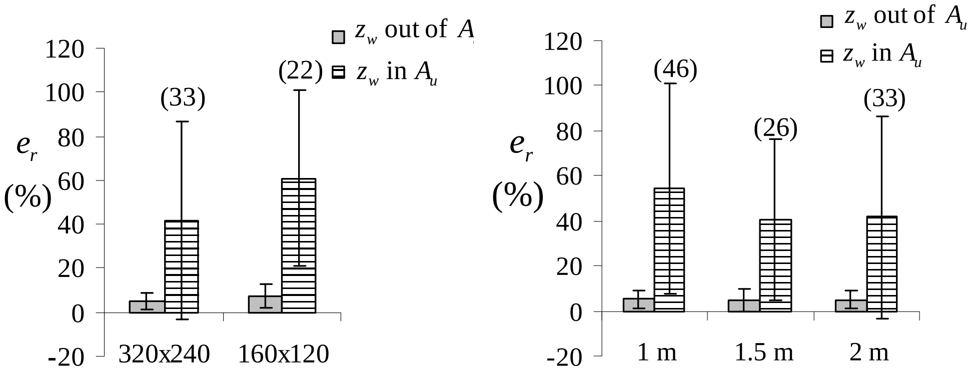
(a) $\overline{R}_\varnothing$ and $(x_w^*(\tau'), z_w^*(\tau'))$.



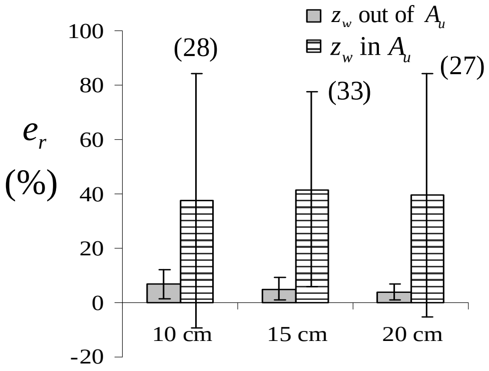
(b) $A_u(\underline{F}, \overline{F})$ and $A_u(F, \overline{F})$.

Fig. 23. An illustration of robot positions and the relationship between A_us when $R_\varnothing = \emptyset$.

where z_w is the measured depth that is used as a ground truth and $\hat{z}_w$ is the computed depth, derived from the stereo reconstruction. Since the basic idea is to avoid overlapping A_us, we compared the relative depth error of the obstacles that are either inside the A_u or outside the A_u. This comparison is performed for several scenarios including different



(a) Image resolution (b) Depth of the obstacle



(c) $z_w(0)$

Fig. 24. The effectiveness of depth error reduction. The height of the bar is the mean value of e_r and the vertical interval represents the variance of e_r. The number in the parenthesis is the trial number.

image resolutions, depth of objects, and robot positions $z_w(0)$. The results are shown in Fig. 24. For each case, we repeat the test over 20 times with different random configurations of obstacles locations. The trial number is shown above the bars in the figure. Note that the mean and the variance of the relative depth error are significantly reduced if the robot stay outside A_u.

G. Summary

In this chapter, we have developed a vision-based navigation system for a mobile robot equipped with a single camera based on geometric information of roads. Depth information was computed using images taken from different camera perspectives, and the depth error range distribution was analyzed across the camera coverage. We showed that the depth error can be excessively large in the region close to the camera baseline, and this degenerated region can cause collisions in robot navigation. We also modeled the untrusted area where the depth error range is beyond a predefined threshold. To reduce the depth error and risk of collision, we propose an algorithm that enables the robot to select its navigation region to avoid the untrusted area. We implemented the algorithm and conducted physical experiments. The results confirmed our analysis. Although we modeled the untrusted area for monocular vision, the untrusted area can also be applied to a robot with multiple cameras or a pan-tilt-zoom camera network to control the depth error. We apply the concept of the untrusted area to 3D reconstruction with a two-view approach.

CHAPTER V

DEPTH-ERROR-AWARE 3D SCENE RECONSTRUCTION WITH A TWO-VIEW APPROACH

A. Introduction

Consider a surveillance system that consists of multiple camera as illustrated in Fig. 25. A surveillance system recognizes objects on the ground surface by stereo reconstruction. A 360 degree view around two cameras can be reconstructed with overlapped images taken from two cameras. Due to the limitation of the camera's field of view, several image pairs are required to reconstruct a 360 degree view around the vision system. As we mentioned in chapter IV, each image pair produces an untrusted area due to the degeneracy introduced by triangulation in the depth computation. This degenerated region causes failures in object recognition. The untrusted area for a 360 degree view can be computed by the union of untrusted areas from each pair of images. We found that the location of untrusted area for a 360 degree view depends on the positions of two cameras. Hence, an additional camera is used to avoid excessive depth error in the untrusted area. We propose an algorithm to control depth error in the untrusted area by adding a camera and computing the location of the additional camera. Therefore, overall depth error can be controlled below a predefined threshold, and the risk of failure in object detection is reduced. We have implemented the proposed algorithm in real environments. Experiments are conducted to confirm the proposed algorithm.

Since the related work for this research overlaps with related work in chapter IV, we omit the related work section. The rest of this chapter is organized as follows. We define our research problem and formulate the untrusted area for the binocular vision system in section B and C, respectively. In section D, we propose an algorithm to select the position

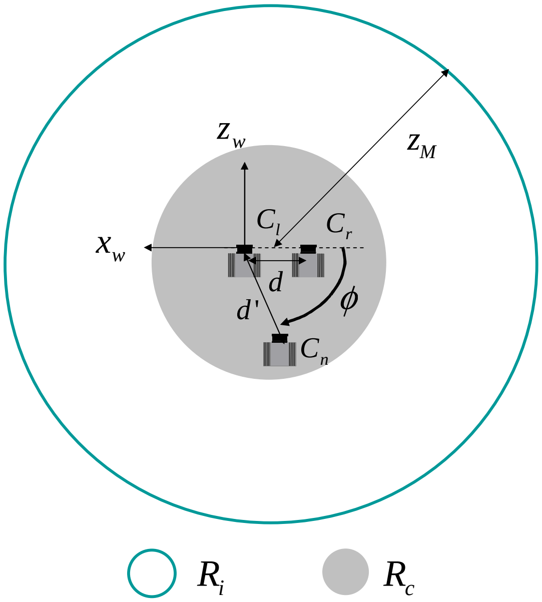


Fig. 25. An illustration of the system configuration. The vision system consists of cameras C_l and C_r reconstructs a 360 degree view within region of interest R_i. To control depth error in the degenerated region, we use an additional camera C_n. C_n should be placed within the predefined available area R_c.

of the additional camera to control depth error. Experiments and summary are presented in section E and F, respectively.

B. Problem Description

1. Coordinate Systems

The given two cameras provides a pair of images for stereo reconstruction. We define C_r and C_l as the right camera and left camera as illustrated in Fig. 26. When we represent variables corresponding to C_r and C_l, we use subscript r and l as a convention in this chapter. Fig. 26 illustrates the right hand coordinate systems and their relationship.

- World coordinate system (WCS): a fixed 3D Cartesian coordinate system. Its x-z plan is the ground surface, and its y-axis is the vertical axis.

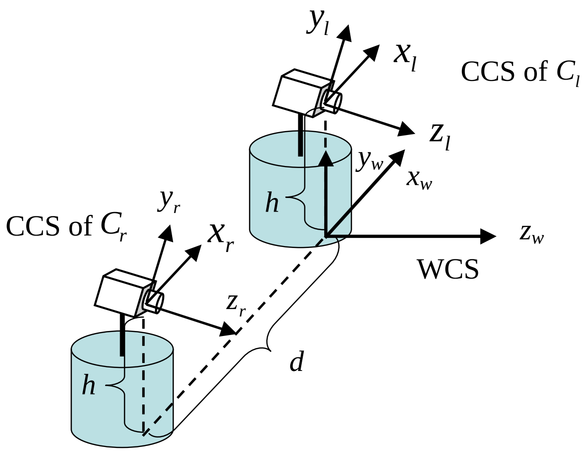


Fig. 26. Definition of coordinate systems and their relationship.

- Camera coordinate system (CCS): a 3D Cartesian coordinate system. It is attached to a camera with its origin at the camera optical center. Its z-axis coincides with the optical axis, and its x-axis and y-axis are parallel with the horizontal direction and the vertical direction of CCD sensor plane, respectively.

- Image coordinate system (ICS): a 2D image coordinate system with its origin at the center of the image. Its u-axis and v-axis are parallel with horizontal and vertical directions of a image, respectively.

As illustrated in Fig. 26, the origin of WCS is placed at the ground plane, where the origin of CCS for C_l is projected on ground plane. The vertical difference between the origins of WCS and CCS for C_l is the camera height h. Define d as the fixed baseline distance between C_l and C_r. Hence, we can define the positions of C_l and C_r as $(0, h, 0)$ and (-$d, h, 0)$, respectively. Cameras usually tilt toward ground to obtain a better coverage of the ground. Define t as the tilt angle.

2. Nomenclature

- $q = (u, v, 1)^T$: a point in ICS.

- $Q = (x_w, y_w, z_w)^T$: q's position in WCS.
- $\hat{Q} = (\hat{x}_w, \hat{y}_w, \hat{z}_w)^T$: the estimated value of Q through stereo reconstruction.
- $Q_c = (x_c, y_c, z_c)^T$: q's position in CCS.
- $e = \hat{z}_w - z_w$: depth error.
- Δe: depth error range.
- A_u : the untrusted area in WCS.
- A_u^c : the untrusted area in WCS when a binocular vision system reconstructs a 360 degree view.
- R_i : region of interest.
- d : baseline distance between two cameras.
- $s(\cdot)$ and $c(\cdot)$ denote $\sin(\cdot)$ and $\cos(\cdot)$, respectively.

3. Assumptions

Since we apply the untrusted area obtained in chapter IV to 3D reconstruction with a two-view approach, we follow assumptions used to analyze depth error, for example, assumptions about calibration of camera parameters, iso-orientation of image planes, and image correspondence error distribution. In addition, we assume that two given cameras are located at fixed position with a given baseline distance.

4. Untrusted Area for a 360 Degree View

We already defined untrusted area A_u with monocular vision in (4.2). We apply the concept of A_u to a 360 degree view reconstruction . Let us define $A_u^c(C_r, C_l)$ as untrusted areas for a 360 degree view that is reconstructed using image pairs taken by camera C_r and C_l with different pan angle. Define F_l and F_r as the image frames taken from C_l and C_r, respectively. When n is the number of image pairs required to reconstruct a 360 degree view, F_{ri} and F_{li} represent i^{th} frames taken from C_r and C_l, respectively. $A_u^c(C_r, C_l)$ can be obtained by the union of untrusted areas from each image pair. Therefore, $A_u^c(C_r, C_l)$ is expressed as

$$A_u^c(C_r, C_l) = \{(x_w, z_w) | (x_w, z_w) \in \bigcup_{i=1}^{n} A_u(F_{li}, F_{ri})\},$$

where x_w and z_w are x-axis and z-axis values in WCS, respectively.

5. Region of Interest

An image frame usually covers both far and near field, but we are not interested in regions that are too far away. Define z_M as the maximal distance that the given two cameras care about for 3D scene reconstruction and $\Pi(F_r \cap F_l)$ as the coverage of the overlapped area between F_r and F_l in 3D space. As illustrated in Fig. 25, the region of interest R_i is defined as the area,

$$R_i = \{(x_w, z_w) | \; (x_w - \frac{d}{2})^2 + z_w^2 \leq z_M^2, \; (x_w, z_w) \in \bigcup_{i=1}^{n} \Pi(F_{ri} \cap F_{li})\}. \qquad (5.1)$$

We want to reduce the depth error e associated with objects in R_i.

6. Problem Definition

A_u^c can be applied in 3D scene reconstruction to reduce depth error. The solution to control depth error in $A_u^c(C_r, C_l)$ is to construct another version of the 3D scene with an additional camera C_n. Since three cameras produce three pairs of images to construct a 3D scene, we can obtain three versions of 3D information with three untrusted areas, $A_u^c(C_r, C_l)$, $A_u^c(C_r, C_n)$, and $A_u^c(C_l, C_n)$. Define e_t as the predefined threshold for depth error range

Δe. To obtain 3D information provided the least Δe, the position of C_n should be satisfied that

$$|\Delta e| \leq e_t \iff A_u^c(C_r, C_l) \cap A_u^c(C_r, C_n) \cap A_u^c(C_l, C_n) = \emptyset. \tag{5.2}$$

Since computing a 3D scene for all three pairs is computationally inefficient, we can reduce the amount of computation by only computing two image pairs taken from (C_r, C_l) and (C_l, C_n). Therefore, (5.2) becomes,

$$|\Delta e| \leq e_t \iff A_u^c(C_r, C_l) \cap A_u^c(C_l, C_n) = \emptyset, \tag{5.3}$$

which means that Δe is bounded below the threshold as long as $A_u^c(C_r, C_l)$ and $A_u^c(C_l, C_n)$ do not overlap.

Since the location and the size of A_u^c in x-z plane are determined based on the baseline distance between two cameras, baseline distance d', which is the relative distance between C_l and C_n as illustrated in Fig. 25, determines the position of C_n to satisfy the non-overlapping condition. There might be multiple d's that satisfy the non-overlapping condition in (5.3). Define R_c as the predefined region where C_n can be placed.

Definition 3. *Given d, e_t, R_c and R_i, compute the additional camera's available locations such that the condition in (5.3) can be satisfied.*

C. Compute Untrusted Area for Two Cameras

Recall that $A_u^c(C_l, C_r)$ is the union of untrusted areas from image pairs taken from C_l and C_r. The size and the location of A_u from each pair of images are expressed using low and high boundaries of the untrusted area in (4.47) and (4.48). A_us for n pairs of images can be categorized into three groups based on the location of A_u: 1) both cameras face z_w-axis (A_u^1), 2) both cameras face x_w-axis (A_u^2), and 3) both cameras face between x_w-axis and z_w-axis (A_u^3). A_u^c is computed by the union of the computed A_us. Let us start with computation of A_u^1.

1. Untrusted Areas When the Cameras Face z_w-axis, A_u^1

A_u^1 is obtained by the union of the untrusted areas from the image pairs captured by cameras that look toward the positive or negative z_w-axis. In this case, the baseline distance between two cameras is the distance difference in x_w-axis. Therefore, $z_w(0) = 0$ and $x_w(0) = d$. We can obtain Δe by plugging $z_w(0)$ and $x_w(0)$ into (4.24).

$$\Delta e = \frac{2r\beta\lambda z_w(n_x x_w + n_0)^2 - 2r\lambda z_w^2(n_x x_w + n_0)}{\lambda^2(n_x x_w + n_0)^2 - r^2 z_w^2}.$$

Plug $z_w(0)$ and $x_w(0)$ into (4.22), we can get $n_x x_w + n_0 = fd$. Hence, Δe becomes a function of z_w and d.

$$\Delta e = \frac{2r\beta\lambda z_w f^2 d^2 - 2r\lambda z_w^2 f d}{\lambda^2 f^2 d^2 - r^2 z_w^2}. \tag{5.4}$$

$\Delta e < 0$ when $d > 0$, and $\Delta e > 0$ when $d < 0$. Since $d < 0$, the untrusted area is obtained when $\Delta e > e_t$. We know that $e_t = \rho z_w$. The boundaries for A_u^1 are obtained by plugging (5.4) into $\Delta e > e_t$.

$$\frac{2r\beta\lambda z_w f^2 d^2 - 2r\lambda z_w^2 f d}{\lambda^2 f^2 d^2 - r^2 z_w^2} > \rho z_w. \tag{5.5}$$

The solution to (5.5) is,

$$z_w > \frac{\lambda f d - df\sqrt{\lambda^2 - \rho(2r\beta\lambda - \rho\lambda^2)}}{\rho r}, \text{ or } z_w < \frac{\lambda f d + df\sqrt{\lambda^2 - \rho(2r\beta\lambda - \rho\lambda^2)}}{\rho r}.$$

Therefore, A_u^1 is written as

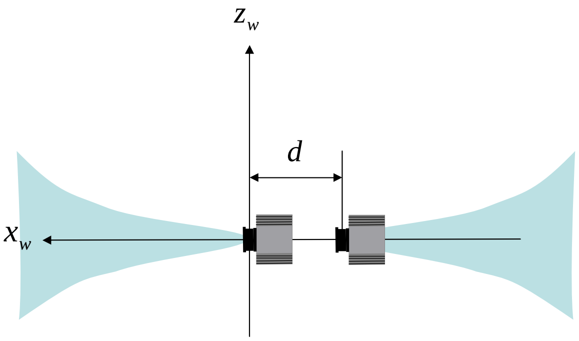


Fig. 27. An illustration of A_u^2.

$$A_u^1 = \{(x_w, z_w) | -z_M + \frac{d}{2} \leq x_w \leq z_M + \frac{d}{2},\ z_w > \frac{\lambda f d - df\sqrt{\lambda^2 - \rho(2r\beta\lambda - \rho\lambda^2)}}{\rho r} \\ \text{or } z_w < \frac{\lambda f d + df\sqrt{\lambda^2 - \rho(2r\beta\lambda - \rho\lambda^2)}}{\rho r}\}. \tag{5.6}$$

2. Untrusted Area When the Cameras Face x_w-axis, A_u^2

A_u^2 consists of untrusted areas from the image pairs taken by cameras that face the positive or negative x_w-axis as illustrated in Fig. 27. First, consider the case that cameras face the positive x_w-axis. Recall that the low and high boundaries of A_u in (4.47) and (4.48) are obtained based on WCS whose z_w-axis coincides with the cameras's optical axis. Since camera's optical axis coincides with the x_w-axis in this case, (4.47) and (4.48) should be modified using the coordinate system transform, especially 90° about the y_w-axis, to compute the high and low boundaries of A_u^2. Define $z_p^l(x_w, d)$ and $z_p^h(x_w, d)$ as the low and high boundaries of A_u^2 when the cameras face the positive x_w-axis. $z_p^l(x_w, d)$ and $z_p^h(x_w, d)$ are computed based on the modified low and high boundaries and baseline distance, $x_w(0) = d$ and $z_w(0) = 0$. $z_p^l(x_w, d)$ and $z_p^h(x_w, d)$ are written as

$$z_p^l(x_w, d) = \frac{r x_w (x_w - d\lambda_x c(t))}{f c(t) d \lambda_x^2 (\rho\lambda_x - 2r\beta)} \left(\lambda_x + \sqrt{\lambda_x^2 + \rho^2\lambda_x^2 - 2r\beta\lambda_x\rho}\right), \tag{5.7}$$

$$z_p^h(x_w, d) = \frac{r x_w (x_w - d\lambda_x c(t))}{f c(t) d \lambda_x^2 (-\rho\lambda_x - 2r\beta)} \left(\lambda_x + \sqrt{\lambda_x^2 + \rho^2\lambda_x^2 + 2r\beta\lambda_x\rho}\right), \tag{5.8}$$

$$\lambda_x = \frac{x_w}{x_w c(t) + h s(t)}.$$

Secondly, consider the case that the cameras face the negative x_w-axis. The low and high boundaries of A_u^2 are obtained using the same procedure to the case that the cameras face the positive x_w-axis. Eq. (4.47) and (4.48) are modified based on the coordinate system transform, especially -90° rotation about the y_w-axis and transition of d along the x_w-axis. Define $z_n^l(x_w, \text{-}d)$ and $z_n^h(x_w, \text{-}d)$ as low and high boundaries of A_u^2 when the cameras are lined up along the negative x_w-axis. In this case, $x_w(0) = \text{-}d$ and $z_w(0) = 0$. $z_n^l(x_w, \text{-}d)$ and $z_n^h(x_w, \text{-}d)$ are written as

$$z_n^l(x_w, \text{-}d) = \frac{r(x_w - d)(x_w - d + d\lambda_x' c(t))}{\text{-}f c(t) d \lambda_x'^2 (-\rho\lambda_x' - 2r\beta)} \left(\lambda_x' + \sqrt{{\lambda_x'}^2 + \rho^2{\lambda_x'}^2 + 2r\beta\lambda_x'\rho}\right), \tag{5.9}$$

$$z_n^h(x_w, \text{-}d) = \frac{r(x_w - d)(x_w - d + d\lambda_x' c(t))}{\text{-}f c(t) d \lambda_x'^2 (\rho\lambda_x' - 2r\beta)} \left(\lambda_x' + \sqrt{{\lambda_x'}^2 + \rho^2{\lambda_x'}^2 - 2r\beta\lambda_x'\rho}\right), \tag{5.10}$$

$$\lambda_x' = \frac{d - x_w}{(d - x_w)c(t) + h s(t)}.$$

The overall A_u^2 is the union of two cases in (5.7-5.10).

$$\begin{aligned} A_u^2 = & \{(x_w, z_w) |\, 0 \le x_w \le z_M + \frac{d}{2},\ z_p^l(x_w, d) \le z_w \le z_p^h(x_w, d)\} \\ & \cup \{(x_w, z_w) |\, \text{-}z_M + \frac{d}{2} \le x_w \le d,\ z_n^l(x_w, \text{-}d) \le z_w \le z_n^h(x_w, \text{-}d)\}. \end{aligned} \tag{5.11}$$

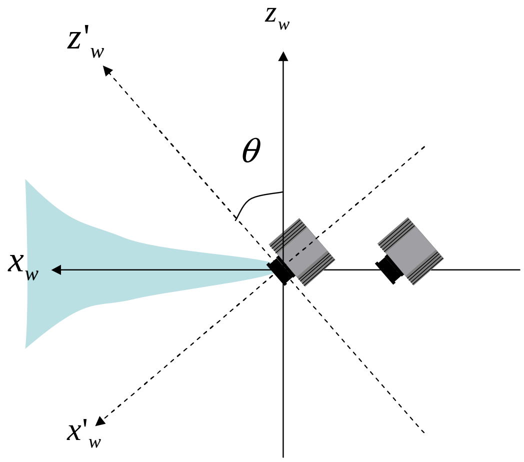


Fig. 28. An illustration of A_u^3. Two cameras are rotated by θ with respect to the y_w-axis.

3. Untrusted Area When the Cameras Face between the x_w-axis and z_w-axis, A_u^3

A_u^3 contains untrusted areas from image pairs captured by cameras that face between the x_w-axis and z_w-axis. To present the rotation of the camera, we defined the angular value as θ. Cameras are rotated by θ with respect to the y_w-axis as shown in Fig. 28. A_u^3 is computed using (5.7-5.10) and baseline distance $x_w(0) = dc(\theta)$ and $z_w(0) = ds(\theta)$. In this case, the cameras' optical axis coincides with the x'_w-axis in Fig. 28. To express A_u^3 based on WCS for $A_u^c(C_l, C_r)$, the coordinate system transformation, θ rotation about the y_w-axis, is required. Define $\overline{z}_w^l(x_w, d)$ and $\overline{z}_w^h(x_w, d)$ as the low and high boundaries of A_u^3 in WCS for A_u^c when $x_w > 0$.

$$\overline{z}^l(x_w, d) = \frac{\gamma_l + 2r\eta_1^l c(\theta)s(\theta)x_w + \sqrt{\gamma_l^2 + 4r\eta_1^l f d\lambda^2 c(t)\eta_2^l c(\theta)s(\theta)x_w}}{2r\eta_1^l s(\theta)^2}, \tag{5.12}$$

$$\overline{z}^h(x_w, d) = \frac{\gamma_h + 2r\eta_1^h c(\theta)s(\theta)x_w + \sqrt{\gamma_h^2 + 4r\eta_1^h f d\lambda^2 c(t)\eta_2^h c(\theta)s(\theta)x_w}}{2r\eta_1^h s(\theta)^2}, \tag{5.13}$$

$$\gamma_l = f d\lambda^2\eta_2^l c(t)c(\theta)^2 + f d\lambda\eta_2^l s(\theta)^2 - r d\lambda\eta_1^l c(\theta)s(\theta), \tag{5.14}$$

$$\gamma_h = f d\lambda^2\eta_2^h c(t)c(\theta)^2 + f d\lambda\eta_2^h s(\theta)^2 - r d\lambda\eta_1^h c(\theta)s(\theta), \tag{5.15}$$

$$\eta_1^l = \lambda + \sqrt{\lambda^2 + \rho^2\lambda^2 - 2r\beta\lambda\rho},\ \ \eta_2^l = \rho\lambda - 2r\beta,$$
$$\eta_1^h = \lambda + \sqrt{\lambda^2 + \rho^2\lambda^2 + 2r\beta\lambda\rho},\ \ \eta_2^h = \text{-}\rho\lambda - 2r\beta.$$

Define $\underline{z}^l(x_w, \text{-}d)$ and $\underline{z}^h(x_w, \text{-}d)$ as the low and high boundaries of A_u^3 in WCS for A_u^c when $x_w < d$. $\underline{z}^l(x_w, \text{-}d)$ and $\underline{z}^h(x_w, \text{-}d)$ are computed using the same procedure.

$$\underline{z}^l(x_w, \text{-}d) = \frac{\gamma_l + 2r\eta_1^l s(\theta)(c(\theta)x_w - d) - \sqrt{\gamma_l^2 + 4r\eta_1^l f d\lambda^2 c(t)\eta_2^l s(\theta)(x_w - dc(\theta))}}{2r\eta_1^l s(\theta)^2}, \tag{5.16}$$

$$\underline{z}^h(x_w, \text{-}d) = \frac{\gamma_h + 2r\eta_1^h s(\theta)(c(\theta)x_w - d) + \sqrt{\gamma_h^2 + 4r\eta_1^h f d\lambda^2 c(t)\eta_2^h s(\theta)(x_w - dc(\theta))}}{2r\eta_1^h s(\theta)^2}. \tag{5.17}$$

Overall A_u^3 is the union of untrusted areas from two cases in (5.12-5.17).

$$\begin{aligned} A_u^3 = &\{(x_w, z_w)|\ 0 \le x_w \le z_M + \frac{d}{2},\ \ \overline{z}^l(x_w, d) \le z_w \le \overline{z}^h(x_w, d)\} \\ &\cup \{(x_w, z_w)|\ \text{-}z_M + \frac{d}{2} \le x_w \le d,\ \ \underline{z}^l(x_w, \text{-}d) \le z_w \le \underline{z}^h(x_w, \text{-}d)\}. \end{aligned} \tag{5.18}$$

Now, we are ready to compute $A_u^c(C_l, C_r)$.

4. $A_u^c(C_l, C_r)$

Recall that $A_u^c(C_l, C_r)$ is the union of A_u^1, A_u^2, and A_u^3. The location of A_u^1 can be adjusted based on d. When we set d to place A_u^1 beyond R_i, we do not need to consider A_u^1 in A_u^c computation. Define d_m as the minimal d to place A_u^1 out of R_i. d_m is computed based on (5.6).

$$\frac{\lambda f d - df\sqrt{\lambda^2 - \rho(2r\beta\lambda - \rho\lambda^2)}}{\rho r} > z_M.$$

d_m is written as

$$d_m < \frac{z_M r \rho}{f(\lambda - \sqrt{\lambda^2 - \rho(2r\beta\lambda - \rho\lambda^2)})}. \tag{5.19}$$

However, A_u^2 and A_u^3 are unavoidable within R_i. Hence, A_u^c is the union of A_u^2 and A_u^3. The following lemma shows the relationship between A_u^2 and A_u^3.

Lemma 3. $A_u^3 \subset A_u^2$.

Proof. A_u^2 and A_u^3 have a pair of untrusted areas: when $x_w > 0$ and $x_w < d$. Since two untrusted areas are symmetric, we compare the untrusted areas when $x_w > 0$ to show the relationship between A_u^2 and A_u^3. In this case, -90° $< \theta <$ 90°.

First, let us compare the low boundaries of A_u^2 $(z_p^l(x_w, d))$ and $A_u^3(\overline{z}^l(x_w, d))$. To satisfy the condition that $A_u^3 \subset A_u^2$, $z_p^l(x_w, d) < \overline{z}^l(x_w, d)$, plug (5.7) and (5.12) into $z_p^l(x_w, d) < \overline{z}^l(x_w, d)$, and we have

$$\frac{\eta_1^l r x_w(x_w - d\lambda c(t))}{fd\lambda^2 c(t)\eta_2^l} < \frac{2r\eta_1^l s(\theta)c(\theta)x_w}{2r\eta_1^l s(\theta)^2} + \frac{\gamma_l}{2r\eta_1^l s(\theta)^2} + \frac{\sqrt{\gamma_l^2 + \kappa_l x_w}}{2r\eta_1^l s(\theta)^2}. \tag{5.20}$$

In order to eliminate the square root value, (5.20) is modified as

$$\begin{aligned} &\left(\frac{\eta_1^l r x_w(x_w - d\lambda c(t))}{fd\lambda^2 c(t)\eta_2^l} 2r\eta_1^l s(\theta)^2 - 2r\eta_1^l s(\theta)c(\theta)x_w\right)^2 \\ &\quad + 4r\eta_1^l s(\theta)^2 \frac{\eta_1^l r x_w(x_w - d\lambda c(t))}{fd\lambda^2 c(t)\eta_2^l}\gamma_l + 4r\eta_1^l s(\theta)c(\theta)\gamma_l x_w - \kappa_1 < 0. \end{aligned} \tag{5.21}$$

Now, we are going to show that (5.21) holds. We can approximate γ_l in (5.14). Since $f > 100$, $fd\lambda\eta_2^l s(\theta)^2 + fd\lambda^2 c(t)\eta_2^l c(\theta)^2 \gg r\eta_1^l d\lambda c(t)c(\theta)s(\theta)$.

$$\begin{aligned} \gamma_l &= fd\lambda\eta_2^l s(\theta)^2 - r\eta_1^l d\lambda c(t)c(\theta)s(\theta) + fd\lambda^2 c(t)\eta_2^l c(\theta)^2 \\ &\approx fd\lambda\eta_2^l (s(\theta)^2 + \lambda c(t)c(\theta)^2). \end{aligned}$$

Eq. (5.21) can be approximated using the approximated γ_l.

$$\begin{aligned}&4r^2{\eta_1^l}^2 x_w^2 s(\theta)^2 \Bigg(\left(\frac{r\eta_1^l s(\theta)(x_w - d\lambda c(t))}{fd\lambda^2 c(t)\eta_2^l} - c(\theta)\right)^2 - \frac{x_w - d\lambda c(t)}{x_w}\frac{s(\theta)^2+\lambda c(t)c(\theta)^2}{\lambda c(t)} \\ &\quad + \frac{fd\lambda c(t)\eta_2^l s(\theta)c(\theta)}{r\eta_1^l x_w}\frac{hs(t)}{x_w c(t)+hs(t)}\Bigg).\end{aligned} \tag{5.22}$$

We know that $d < 0$ and $\eta_2^l > 0$. When $0^\circ < \theta < 90^\circ$,

$$\begin{aligned}&4r^2{\eta_1^l}^2 x_w^2 s(\theta)^2 > 0, \\ &\left(\left(\frac{r\eta_1^l s(\theta)(x_w - d\lambda c(t))}{fd\lambda^2 c(t)\eta_2^l} - c(\theta)\right)^2 - \frac{x_w - d\lambda c(t)}{x_w}\frac{s(\theta)^2+\lambda c(t)c(\theta)^2}{\lambda c(t)} \right) < 0, \\ &\frac{fd\lambda c(t)\eta_2^l s(\theta)c(\theta)}{r\eta_1^l x_w}\frac{hs(t)}{x_w c(t)+hs(t)} < 0,\end{aligned}$$

since

$$\begin{aligned}&0 < \left(\frac{r\eta_1^l (x_w - d\lambda c(t))}{fd\lambda^2 c(t)\eta_2^l} s(\theta) - c(\theta)\right)^2 \le 1, \quad \frac{x_w - d\lambda c(t)}{x_w}\frac{s(\theta)^2+\lambda c(t)c(\theta)^2}{\lambda c(t)} > 1, \\ &0 > \frac{r\eta_1^l (x_w - d\lambda c(t))}{fd\lambda^2 c(t)\eta_2^l} > -0.5.\end{aligned}$$

Therefore, (5.21) holds.

When -90$^\circ < \theta < 0^\circ$,

$$\begin{aligned}&\left(\left(\frac{r\eta_1^l s(\theta)(x_w - d\lambda c(t))}{fd\lambda^2 c(t)\eta_2^l} - c(\theta)\right)^2 - \frac{x_w - d\lambda c(t)}{x_w}\frac{s(\theta)^2+\lambda c(t)c(\theta)^2}{\lambda c(t)} \right) < 0, \\ &\frac{fd\lambda c(t)\eta_2^l s(\theta)c(\theta)}{r\eta_1^l x_w}\frac{hs(t)}{x_w c(t)+hs(t)} > 0.\end{aligned}$$

However,

$$\left(\left(\frac{r\eta_1^l (x_w - d\lambda c(t))}{fd\lambda^2 c(t)\eta_2^l} s(\theta) - c(\theta)\right)^2 + \frac{fd\lambda c(t)\eta_2^l s(\theta)c(\theta)}{r\eta_1^l x_w}\frac{hs(t)}{x_w c(t)+hs(t)} \right) \le 1,$$

since

$$0 < \frac{fd\lambda c(t)\eta_2^l s(\theta)c(\theta)}{r\eta_1^l x_w}\frac{hs(t)}{x_w c(t)+hs(t)} < 0.03.$$

Therefore, (5.21) holds.

In the case of high boundaries, $z_p^h(x_w, d) > \overline{z}^h(x_w, d)$ should be satisfied. Plug (5.8) and (5.13) into $z_p^h(x_w, d) > \overline{z}^h(x_w, d)$, we have

$$\frac{\eta_1^h r x_w (x_w - d\lambda c(t))}{f d\lambda^2 c(t)\eta_2^h} > \frac{2r\eta_1^h s(\theta)c(\theta)x_w}{2r\eta_1^h s(\theta)^2} + \frac{\gamma_h}{2r\eta_1^h s(\theta)^2} - \frac{\sqrt{\gamma_h^2 + \kappa_h x_w}}{2r\eta_1^h s(\theta)^2}. \tag{5.23}$$

Eq. (5.23) is modified to eliminate the square root part.

$$\begin{aligned}&\left(\frac{2{\eta_1^h}^2 r^2 s(\theta)^2 x_w (x_w - d\lambda c(t))}{f d\lambda^2 c(t)\eta_2^h} - 2r\eta_1^h s(\theta)c(\theta)x_w\right)^2 \\ &\quad - \frac{4{\eta_1^h}^2 r^2 s(\theta)^2 x_w (x_w - d\lambda c(t))}{f d\lambda^2 c(t)\eta_2^h}\gamma_h + 4r\eta_1^h s(\theta)c(\theta)\gamma_h x_w - \kappa_h x_w < 0.\end{aligned} \tag{5.24}$$

Since $f d\lambda\eta_2^h s(\theta) + f d\lambda^2 c(t)\eta_2^h c(\theta)^2 \gg r\eta_1^h d\lambda c(t)c(\theta)s(\theta)$, γ_h in (5.15) can be approximated.

$$\begin{aligned}\gamma_h &= f d\lambda\eta_2^h s(\theta)^2 - r\eta_1^h d\lambda c(t)c(\theta)s(\theta) + f d\lambda^2 c(t)\eta_2^h c(\theta)^2 \\ &\approx f d\lambda\eta_2^h (s(\theta)^2 + \lambda c(t)c(\theta)^2).\end{aligned}$$

Eq. (5.24) is approximated using the approximated γ_h.

$$\begin{aligned}&4{\eta_1^h}^2 r^2 s(\theta)^2 x_w^2 \left(\left(\frac{\eta_1^h r s(\theta)(x_w - d\lambda c(t))}{f d\lambda^2 c(t)\eta_2^h} - c(\theta)\right)^2 - \frac{x_w - d\lambda c(t)}{x_w}\frac{s(\theta)^2 + \lambda c(t)c(\theta)^2}{\lambda c(t)} \right. \\ &\left. + \frac{f d\lambda\eta_2^h s(\theta)c(\theta)}{r x_w \eta_1^h}\frac{hs(t)}{x_w c(t) + hs(t)}\right).\end{aligned}$$

We know that $d < 0$ and $\eta_2^h < 0$. When -90° $< \theta <$ 0°,

$$\begin{aligned}&4{\eta_1^h}^2 r^2 s(\theta)^2 x_w^2 > 0, \frac{f d\lambda\eta_2^h s(\theta)c(\theta)}{r x_w \eta_1^h}\frac{hs(t)}{x_w c(t) + hs(t)} < 0, \\ &\left(\frac{\eta_1^h r s(\theta)(x_w - d\lambda c(t))}{f d\lambda^2 c(t)\eta_2^h} - c(\theta)\right)^2 - \frac{x_w - d\lambda c(t)}{x_w}\frac{s(\theta)^2 + \lambda c(t)c(\theta)^2}{\lambda c(t)} < 0,\end{aligned}$$

since

$$0 < \left(\frac{\eta_1^h r s(\theta)(x_w - d\lambda c(t))}{f d\lambda^2 c(t)\eta_2^h} - c(\theta)\right)^2 < 1, \quad \frac{x_w - d\lambda c(t)}{x_w}\frac{s(\theta)^2 + \lambda c(t)c(\theta)^2}{\lambda c(t)} > 1.$$

Therefore, (5.24) holds.

When $0° < \theta < 90°$,

$$\left(\frac{\eta_1^h r(x_w - d\lambda c(t))}{f d\lambda^2 c(t)\eta_2^h} s(\theta) - c(\theta)\right)^2 - \frac{x_w - d\lambda c(t)}{x_w}\frac{s(\theta)^2 + \lambda c(t)c(\theta)^2}{\lambda c(t)}$$
$$+ \frac{f d\lambda\eta_2^h s(\theta)c(\theta)}{r x_w \eta_1^h}\frac{hs(t)}{x_w c(t) + hs(t)} < 0,$$

since

$$\frac{f d\lambda\eta_2^h s(\theta)c(\theta)}{r x_w \eta_1^h}\frac{hs(t)}{x_w c(t) + hs(t)} < 0.03, \;\; 0 < \frac{\eta_1^h r(x_w - d\lambda c(t))}{f d\lambda^2 c(t)\eta_2^h} < 0.5,$$
$$\left(\frac{\eta_1^h r(x_w - d\lambda c(t))}{f d\lambda^2 c(t)\eta_2^h} s(\theta) - c(\theta)\right)^2 + \frac{f d\lambda\eta_2^h s(\theta)c(\theta)}{r x_w \eta_1^h}\frac{hs(t)}{x_w c(t) + hs(t)} \le 1.$$

Therefore, (5.24) holds. □

Due to Lemma 3, $A_u^c(C_l, C_r)$ is written as

$$\begin{aligned} A_u^c(C_l, C_r) &= A_u^1 \cup A_u^2 \cup A_u^3 \\ &= A_u^2. \end{aligned}$$

Hence, $A_u^c(C_l, C_r)$ is

$$\begin{aligned} A_u^c(C_l, C_r) = \{&(x_w, z_w)|\; 0 \le x_w \le z_M + \frac{d}{2},\;\; z_p^l(x_w, d) \le z_w \le z_p^h(x_w, d)\} \\ &\cup \{(x_w, z_w)|\; \text{-}z_M + \frac{d}{2} \le x_w \le d,\;\; z_n^l(x_w, \text{-}d) \le z_w \le z_n^h(x_w, \text{-}d)\}. \end{aligned} \quad (5.25)$$

D. Algorithm

When a 360 degree view is reconstructed using two cameras C_l and C_r, $A_u^c(C_l, C_r)$ is always exited. We know that the location of $A_u^c(C_l, C_r)$ depends on the baseline distance between C_l and C_r. To avoid $A_u^c(C_l, C_r)$, an additional camera C_n is required. C_l and C_n also have untrusted area $A_u^c(C_l, C_n)$. We suggest an algorithm to select the location for C_n that satisfy the non-overlapping condition in (5.3) within the predefined available area R_c.

Define d' and ϕ as the minimal distance between C_l and C_n and the angle of baseline d' from the negative x-axis as shown in Fig. 25. The location of C_n is expressed using d' and ϕ. Define d'_M as the maximal d'. d'_M is from the predefined area R_c. Recall that the location of A_u^1 is adjusted using the baseline distance between two cameras. The minimal d', d'_m, should be computed to place A_u^1 of $A_u^c(C_l, C_n)$ beyond R_i using (5.19). Hence, C_n should be located between the circles with radius d'_m and d'_M as shown in Fig. 25.

Due to Lemmas 1 and 2 in chapter IV, (d', ϕ) set to satisfy non-overlapping condition in (5.3) is obtained by comparing the boundaries of the bounding triangle of $A_u^c(C_l, C_n)$ and $A_u^c(C_l, C_n)$. $A_u^c(C_l, C_n)$ is fixed due to the fixed d. Since R_i is a circle, one side of bounding triangles for A_u^c within R_i is not located on line $x_w = z_M + d/2$ or $x_w =$ -$z_M + d/2$. To compare the bounding triangles, we should compare the intervals from the intersection points between the circle and the untrusted area. We can compute the exact interval of the bounding triangles of A_u^c within R_i. However, it is not computationally efficient, because it includes high order polynomial equations. Hence, we use the approximated intervals when $x_w = z_M + d/2$ and $x_w =$ -$z_M + d/2$. Since the approximated intervals are wider than the exact intervals, the (d', ϕ) set from the approximated intervals is the subset of (d', ϕ) from the exact intervals.

Four vertices of $A_u^c(C_l, C_r)$ on line $x_w = z_M + d/2$ and on line $x_w =$ -$z_M + d/2$ are defined as a_p^l, a_p^h, a_n^l, and a_n^h, and they are computed using (5.7-5.10).

$$\begin{aligned}
a_p^l &= z_p^l(z_M + \frac{d}{2}, d),\\
a_p^h &= z_p^h(z_M + \frac{d}{2}, d),\\
a_n^l &= z_n^l(-z_M + \frac{d}{2}, -d),\\
a_n^h &= z_n^h(-z_M + \frac{d}{2}, -d).
\end{aligned}$$

Since $A_u^c(C_l, C_r)$ and $A_u^c(C_l, C_n)$ are expressed based on different WCS, $A_u^c(C_l, C_n)$ should be transformed with respect to WCS for $A_u^c(C_l, C_r)$ to compute the (d', ϕ) set.

$$\begin{aligned}
b_p^l(x_w, d') = \frac{1}{2r\eta_1^l s(\phi)^2}\Bigg(& r\eta_1^l c(t)d'\lambda s(\phi) - 2r\eta_1^l c(\phi)s(\phi)x_w + fc(\phi)c(t)d'\lambda\eta_2^l\\
& + \sqrt{(r\eta_1^l c(t)d'\lambda s(\phi) + fc(\phi)c(t)d'\lambda\eta_2^l)^2 - 4r\eta_1^l fd'\lambda\eta_2^l c(t)s(\phi)x_w}\Bigg)
\end{aligned} \tag{5.26}$$

$$\begin{aligned}
b_p^h(x_w, d') = \frac{1}{2r\eta_1^h s(\phi)^2}\Bigg(& r\eta_1^h c(t)d'\lambda s(\phi) - 2r\eta_1^h c(\phi)s(\phi)x_w + fc(\phi)c(t)d'\lambda\eta_2^h\\
& - \sqrt{(r\eta_1^h c(t)d'\lambda s(\phi) + fc(\phi)c(t)d'\lambda\eta_2^h)^2 - 4r\eta_1^h fd'\lambda\eta_2^h c(t)s(\phi)x_w}\Bigg)
\end{aligned} \tag{5.27}$$

$$\begin{aligned}
b_n^l(x_w, -d') = \frac{-1}{2r\eta_1^h s(\phi)^2}\Bigg(& r\eta_1^h c(t)d'\lambda s(\phi) + 2r\eta_1^h s(\phi)(x_w c(\phi) - d') + fc(\phi)c(t)d'\lambda\eta_2^h\\
& + \sqrt{(r\eta_1^h c(t)d'\lambda s(\phi) + fc(\phi)c(t)d'\lambda\eta_2^h)^2 + 4r\eta_1^h fd'\lambda\eta_2^h c(t)s(\phi)(x_w - c(\phi)d')}\Bigg)
\end{aligned} \tag{5.28}$$

$$\begin{aligned}
b_n^h(x_w, -d') = \frac{-1}{2r\eta_1^l s(\phi)^2}\Bigg(& r\eta_1^l c(t)d'\lambda s(\phi) + 2r\eta_1^l s(\phi)(x_w c(\phi) - d') + fc(\phi)c(t)d'\lambda\eta_2^l\\
& - \sqrt{(r\eta_1^l c(t)d'\lambda s(\phi) + fc(\phi)c(t)d'\lambda\eta_2^l)^2 + 4r\eta_1^l fd'\lambda\eta_2^l c(t)s(\phi)(x_w - c(\phi)d')}\Bigg)
\end{aligned} \tag{5.29}$$

Four vertices of $A_u^c(C_l, C_n)$ on line $x_w = z_M + d/2$ and on line $x_w = -z_M + d/2$

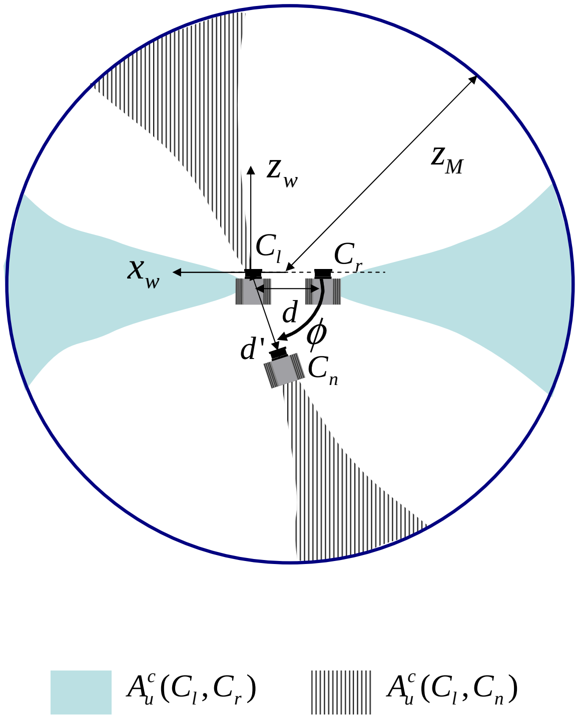


Fig. 29. An illustration of the relationship between $A_u^c(C_l, C_r)$ and $A_u^c(C_l, C_n)$ when $0° < \phi < 90°$.

become $b_p^l(z_M + d/2, d')$, $b_p^h(z_M + d/2, d')$, $b_n^l(\text{-}z_M + d/2, \text{-}d')$, and $b_n^l(\text{-}z_M + d/2, \text{-}d')$. To ensure that the bounding triangles of $A_u^c(C_l, C_r)$ and $A_u^c(C_l, C_n)$ do not overlap each other, $[a_p^l, a_p^h]$ and $[a_n^l, a_n^h]$ do not intersect with $[b_p^l(z_M + d/2, d'), b_p^h(z_M + d/2, d')]$ and $[b_n^l(\text{-}z_M + d/2, d'), b_n^h(\text{-}z_M + d/2, d')]$, respectively.

Since C_n can be located all directions around C_l, the (d', ϕ) set is computed based on four cases: 1) $0° < \phi < 90°$, 2) $90° < \phi < 180°$, 3) $180° < \phi < 270°$, and 4) $270° < \phi < 360°$.

Let us consider the case when $0° < \phi < 90°$. Fig. 29 illustrates the relationship between $A_u^c(C_l, C_r)$ and $A_u^c(C_l, C_n)$ when $0° < \phi < 90°$. To ensure that the intervals do not overlap, the (d', ϕ) set should satisfy two conditions: 1) $a_p^h < b_p^l(z_M + d/2, d')$ when $x_w > 0$ and 2) $a_n^l > b_n^h(\text{-}z_M + d/2, -d')$ when $x_w < d$. Hence, (d', ϕ) is the union of (d', ϕ) sets from each case.

When $x_w > 0$, the relationship between d' and ϕ is obtained by plugging (5.26) into $a_p^h < b_p^l(z_M + d/2, d')$.

$$
\begin{aligned}
a_p^h < \frac{1}{2r\eta_1^l s(\phi)^2}\Bigg(& r\eta_1^l c(t)d'\lambda s(\phi) - 2r\eta_1^l c(\phi)s(\phi)(z_M + \frac{d}{2}) + fc(\phi)c(t)d'\lambda\eta_2^l \\
& + \sqrt{(r\eta_1^l c(t)d'\lambda s(\phi) + fc(\phi)c(t)d'\lambda\eta_2^l)^2 - 4r\eta_1^l fd'\lambda\eta_2^l c(t)s(\phi)(z_M + \frac{d}{2})}\Bigg) \ .
\end{aligned}
$$

The inequality above can be rewritten as

$$
\begin{aligned}
d' &< \frac{\psi_1^2}{2c(t)\lambda\psi_1(r\eta_1^l s(\phi) + f\eta_2^l c(\phi)) - 4r\eta_1^l f\lambda\eta_2^l c(t)s(\phi)(z_M + d/2)}, \qquad (5.30) \\
\psi_1 &= 2a_p^h r\eta_1^l s(\phi)^2 + 2r\eta_1^l c(\phi)s(\phi)(z_M + d/2)
\end{aligned}
$$

When $x_w < d$, the relationship between d' and ϕ is obtained by plugging (5.29) into $a_n^l > b_n^h(\text{-}z_M + d/2, \text{-}d')$.

$$
\begin{aligned}
a_n^l > \frac{1}{2r\eta_1^l s(\phi)^2}\Bigg(& - r\eta_1^l c(t)d'\lambda s(\phi) - 2r\eta_1^l s(\phi)(x_w c(\phi) - d') - fc(\phi)c(t)d'\lambda\eta_2^l \\
& + \sqrt{(r\eta_1^l c(t)d'\lambda s(\phi) + fc(\phi)c(t)d'\lambda\eta_2^l)^2 + 4r\eta_1^l fd'\lambda\eta_2^l c(t)s(\phi)(x_w - c(\phi)d')}\Bigg) \ .
\end{aligned}
$$

This can be rewritten as

$$
\omega_{2f}{d'}^2 + \omega_{1f}d' + \omega_{0f} > 0, \qquad (5.31)
$$

$$
\begin{aligned}
\omega_{2f} &= 4r\eta_1^l s(\phi)(r\eta_1^l s(\phi) + f\lambda\eta_2^l c(t)c(\phi) - r\eta_1^l\lambda c(t)s(\phi) + f\lambda\eta_2^l c(t)c(\phi)) \\
\omega_{1f} &= -8r^2{\eta_1^l}^2 s(\phi)^2(-c(\phi)(z_M - \frac{d}{2}) + s(\phi)a_n^l) - 4r\eta_1^l f\lambda\eta_2^l c(t)s(\phi)(z_M - \frac{d}{2}) \\
&\quad + 4r\eta_1^l s(\phi)(r\eta_1^l c(t)\lambda s(\phi) + fc(\phi)c(t)\lambda b)(s(\phi)a_n^l - c(\phi)(z_M - \frac{d}{2})) \\
\omega_{0f} &= 4r^2{\eta_1^l}^2 s(\phi)^2((z_M - \frac{d}{2})^2 c(\phi)^2 + {a_n^l}^2 - 2s(\phi)a_n^l c(\phi)(z_M - \frac{d}{2})).
\end{aligned}
$$

The solution to (5.31) is,

$$d' < \frac{-\omega_{1f} - \sqrt{\omega_{1f}^2 - 4\omega_{2f}\omega_{0f}}}{2\omega_{2f}} \tag{5.32}$$

$$\text{or } d' > \frac{-\omega_{1f} + \sqrt{\omega_{1f}^2 - 4\omega_{2f}\omega_{0f}}}{2\omega_{2f}}. \tag{5.33}$$

Since $d' > 0$ in (5.33), (d', ϕ) set is obtained from (5.32) when $x_w < d$.

The relationship between d' and ϕ is obtained by the union of (5.30) and (5.32) when $0° < \phi < 90°$.

In case when $0° < \phi <$ -90°, the (d', ϕ) set is obtained using the same procedure. the (d', ϕ) set should satisfy two conditions: 1) $a_p^l > b_p^h(z_M + d/2, d')$ when $x_w > 0$ and 2) $a_n^h < b_n^l(\text{-}z_M + d/2, \text{-}d')$ when $x_w < d$.

When $x_w > 0$, the relationship between d' and ϕ is obtained by plugging (5.27) into $a_p^l > b_p^h(z_M + d/2, d')$.

$$d' < \frac{\psi_2^2}{2c(t)\lambda\psi_2(r\eta_1^h s(\phi) + f\eta_2^h c(\phi)) - 4r\eta_1^h f\lambda\eta_2^h c(t)s(\phi)(z_M + d/2)}, \tag{5.34}$$

$$\psi_2 = 2a_p^l r\eta_1^h s(\phi)^2 + 2r\eta_1^h c(\phi)s(\phi)(z_M + d/2).$$

When $x_w < d$, the relationship between d' and ϕ is obtained by plugging (5.28) into $a_n^h < b_n^l(\text{-}z_M + d/2, \text{-}d')$.

$$\begin{aligned} d' <& \frac{-\omega_{1s} - \sqrt{\omega_{1s}^2 - 4\omega_{2s}\omega_{0s}}}{2\omega_{2s}}, \\ &\omega_{2s} = 4r\eta_1^h s(\phi)(r\eta_1^h s(\phi) + f\lambda\eta_2^h c(t)c(\phi) - r\eta_1^h \lambda c(t)s(\phi) + f\lambda\eta_2^h c(t)c(\phi)), \\ &\omega_{1s} = -8r^2{\eta_1^h}^2 s(\phi)^2(-c(\phi)(z_M - \frac{d}{2}) + s(\phi)a_n^h) - 4r\eta_1^h f\lambda\eta_2^h c(t)s(\phi)(z_M - \frac{d}{2}) \\ &\quad + 4r\eta_1^h s(\phi)(r\eta_1^h c(t)\lambda s(\phi) + fc(\phi)c(t)\lambda b)(s(\phi)a_n^h - (z_M - \frac{d}{2})c(\phi)), \\ &\omega_{0s} = 4r^2{\eta_1^h}^2 s(\phi)^2((z_M - \frac{d}{2})^2 c(\phi)^2 + {a_n^h}^2 - 2s(\phi)a_n^h(z_M - \frac{d}{2})c(\phi). \end{aligned} \tag{5.35}$$

$d' > (-\omega_{1s} + \sqrt{\omega_{1s}^2 - 4\omega_{2s}\omega_{0s}})/(2\omega_{2s})$ is abandoned because $d' > 0$, as in (5.33).

The relationship between d' and ϕ is also obtained by the union of (5.34) and (5.35) when $-90° < \phi < 0°$.

In case when $90° < \phi < 180°$, the (d', ϕ) set is obtained based on two conditions: 1) $a_n^h < b_p^h(\text{-}z_M + d/2, d')$ when $x_w < d$ and 2) $a_p^l > b_n^l(z_M + d/2, \text{-}d')$ when $x_w > 0$. Fig. 30 illustrates the relationship between $A_u^c(C_l, C_r)$ and $A_u^c(C_l, C_n)$ when $90° < \phi < 180°$. (d', ϕ) is also obtained using the same procedure as in the case when $0° < \phi < 90°$.

When $x_w < d$, (5.27) is plugged into $a_n^h < b_p^h(\text{-}z_M + d/2, d')$ to compute the relationship between d' and ϕ.

$$\begin{aligned} d' <& \frac{\psi_3^2}{2c(t)\lambda\phi_3(r\eta_1^h s(\phi) + f\eta_2^h c(\phi)) + 4r\eta_1^h f\lambda\eta_2^h c(t)s(\phi)(z_M - d/2)}, \\ &\psi_3 = 2a_n^h r\eta_1^h s(\phi)^2 - 2r\eta_1^h c(\phi)s(\phi)(z_M - d/2). \end{aligned} \tag{5.36}$$

When $x_w > 0$, (5.28) is plugged into $a_p^l > b_n^l(z_M + d/2, \text{-}d')$ to compute the relation-

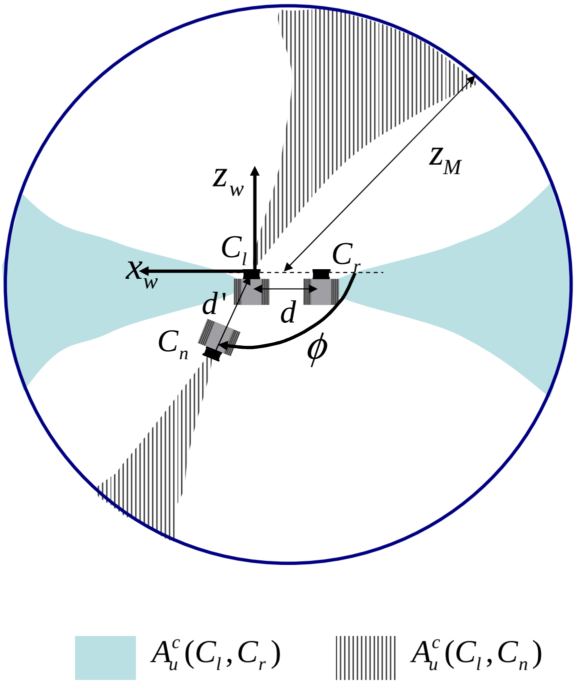


Fig. 30. An illustration of the relationship between $A_u^c(C_l, C_r)$ and $A_u^c(C_l, C_n)$ when $90° < \phi < 180°$.

ship between d' and ϕ.

$$d' < \frac{-\omega_{1t} - \sqrt{\omega_{1t}^2 - 4\omega_{2t}\omega_{0t}}}{2\omega_{2t}}, \tag{5.37}$$

$$\omega_{2t} = 4r\eta_1^h s(\phi)(r\eta_1^h s(\phi) + f\lambda\eta_2^h c(t)c(\phi) - r\eta_1^h \lambda c(t)s(\phi) + f\lambda\eta_2^h c(t)c(\phi)),$$

$$\begin{aligned}\omega_{1t} = {} & -8r^2{\eta_1^h}^2 s(\phi)^2(c(\phi)(z_M + \frac{d}{2}) + s(\phi)a_p^l) + 4r\eta_1^h f\lambda\eta_2^h c(t)s(\phi)(z_M + \frac{d}{2}) \\ & + 4r\eta_1^h s(\phi)(r\eta_1^h c(t)\lambda s(\phi) + fc(\phi)c(t)\lambda b)(s(\phi)a_p^l + (z_M + \frac{d}{2})c(\phi)),\end{aligned}$$

$$\omega_{0t} = 4r^2{\eta_1^h}^2 s(\phi)^2((z_M + \frac{d}{2})^2 c(\phi)^2 + {a_p^l}^2 + 2s(\phi)a_p^l(z_M + \frac{d}{2})c(\phi).$$

The union of the (d', ϕ) sets from (5.36) and (5.37) shows the relationship between d' and ϕ to satisfy the non-overlapping condition when $90° < \phi < 180°$.

In the case when $180° < \phi < 270°$, two conditions for the non-overlapping condition are $a_n^l > b_p^l(\text{-}z_M + d/2, d')$ when $x_w < d$ and $a_p^h < b_n^h(z_M + d/2, \text{-}d')$ when $x_w > 0$.

When $x_w < d$, (5.26) is plugged into $a_n^l > b_p^l(\text{-}z_M + d/2, d')$ to obtain the relationship between d' and ϕ.

$$d' < \frac{\phi_4^2}{2c(t)\lambda\psi_4(r\eta_1^h s(\phi) + f\eta_2^l c(\phi)) + 4r\eta_1^l f\lambda\eta_2^l c(t)s(\phi)(z_M - d/2)}, \tag{5.38}$$
$$\psi_4 = 2a_n^l r\eta_1^l s(\phi)^2 - 2r\eta_1^l c(\phi)s(\phi)(z_M - d/2).$$

When $x_w > 0$, (5.29) is plugged into $a_p^h < b_n^h(z_M{+}d/2, \text{-}d')$. The relationship between d' and ϕ is

$$d' < \frac{-\omega_{1g} - \sqrt{\omega_{1g}^2 - 4\omega_{2g}\omega_{0g}}}{2\omega_{2g}} \tag{5.39}$$
$$\omega_{2g} = 4r\eta_1^l s(\phi)(r\eta_1^l s(\phi) + f\lambda\eta_2^l c(t)c(\phi) - r\eta_1^l \lambda c(t)s(\phi) + f\lambda\eta_2^l c(t)c(\phi))$$
$$\omega_{1g} = -8r^2{\eta_1^l}^2 s(\phi)^2(c(\phi)(z_M + \frac{d}{2}) + s(\phi)a_p^h) + 4r\eta_1^l f\lambda\eta_2^l c(t)s(\phi)(z_M + \frac{d}{2})$$
$$+ 4r\eta_1^l s(\phi)(r\eta_1^l c(t)\lambda s(\phi) + fc(\phi)c(t)\lambda b)(s(\phi)a_p^h + c(\phi)(z_M - \frac{d}{2}))$$
$$\omega_{0g} = 4r^2{\eta_1^l}^2 s(\phi)^2((z_M + \frac{d}{2})^2 c(\phi)^2 + {a_p^h}^2 + 2s(\phi)a_p^h c(\phi)(z_M + \frac{d}{2})).$$

The union of the (d', ϕ) sets from (5.38) and (5.39) shows the relationship between d' and ϕ to satisfy the non-overlapping condition when $180° < \phi < 270°$.

Fig. 31 illustrates the available area of C_n to satisfy the non-overlapping condition within R_c.

E. Experiments

1. Software and Hardware

We have implemented the algorithm on a laptop PC with a 1.6 *GHz* Centrino processor and 512 *MB* RAM. The laptop runs Microsoft Windows XP and the algorithm has been implemented using Matlab. We simulated two-view approach with a single camera which

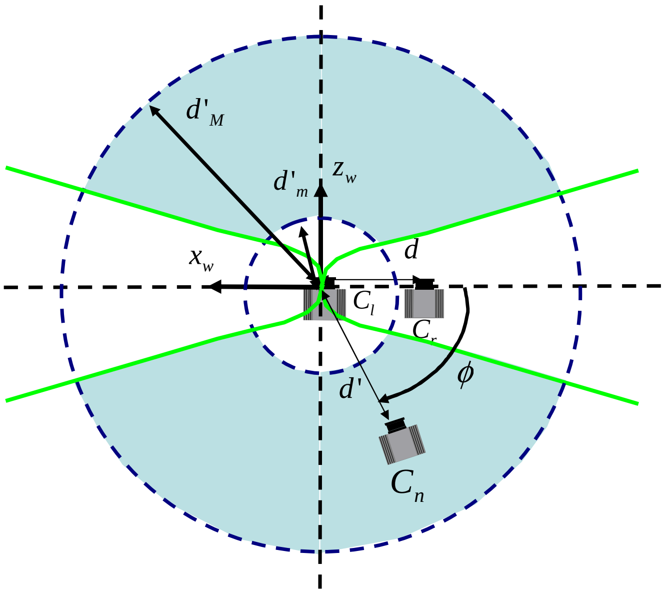


Fig. 31. The shaded area presents available (d', θ) set to satisfy the non overlapping condition between $A^c(C_l, C_r)$ and $A^c(C_l, C_n)$.

Fig. 32. The camera used in experiments.

is a Panasonic HCM 280 networked pan-tilt-zoom camera with a 51° horizontal field of view as illustrated in Fig 32. The intrinsic camera parameters are estimated using the Matlab calibration toolbox [98], and the extrinsic camera parameters are measured by camera potentiometers. During the experiment, we set default z_M =40 m and $t = 15^\circ$. The camera height h =2 m, and the baseline distance between C_l and C_r is d =-50 cm. We conducted the experiments in the corridor of H. R. Bright Bldg. in Texas A&M University. The ob-

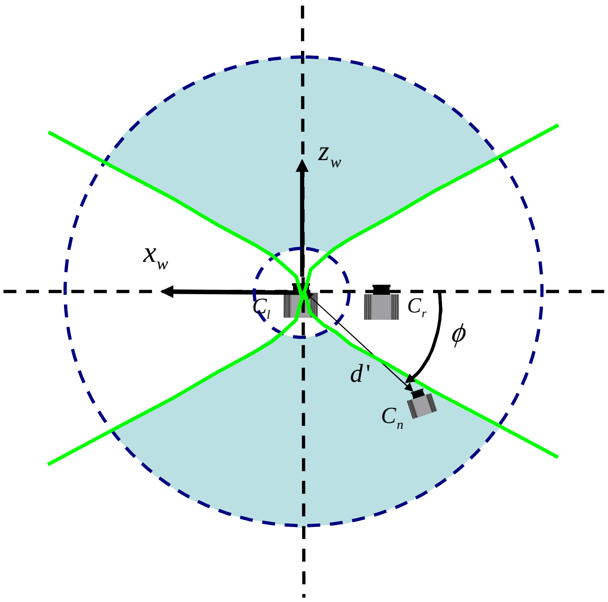


Fig. 33. The shaded area illustrates the available set (d', ϕ) on the x-z plane when d =-50 cm.

stacles used in the experiments are books and blocks with a size of 20 cm×14.5 cm ×10 cm.

2. The Representative Case

We present the available location for an additional camera C_n. The minimal distance d'_m =-23 cm with the given z_M, and the maximal distance d'_M =-150 cm with the given R_c. Fig. 33 shows the available (d', ϕ). Fig. 34 illustrates the relationship between $A^c_u(C_l, C_r)$ and $A^c_u(C_l, C_n)$ when d' =-100 cm and $\phi = 40°$.

3. Depth Error Reduction Effectiveness

We carried out the experiment to verify the effectiveness of the depth error reduction with A^c_u. We simulate two-view approach with a single camera. Since the location for an additional camera is determined based on avoiding overlapping A^c_us, we compared the relative

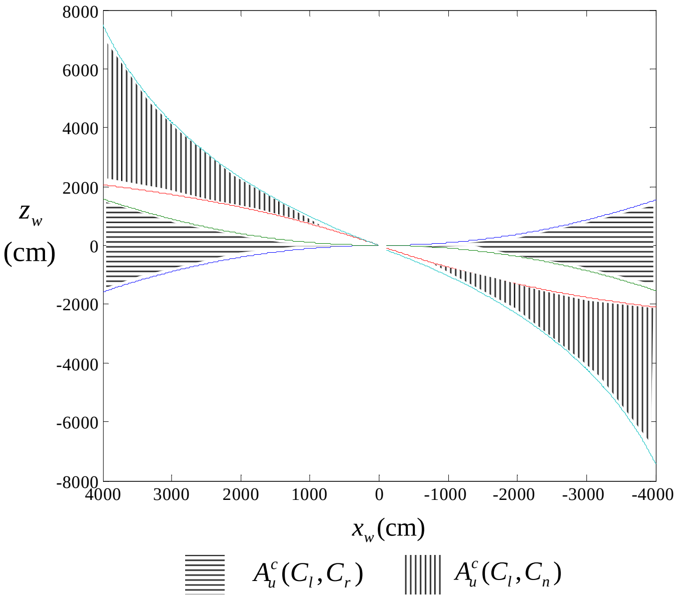


Fig. 34. The relationship between $A_u^c(C_l, C_r)$ and $A_u^c(C_l, C_n)$ when d=-50 cm.

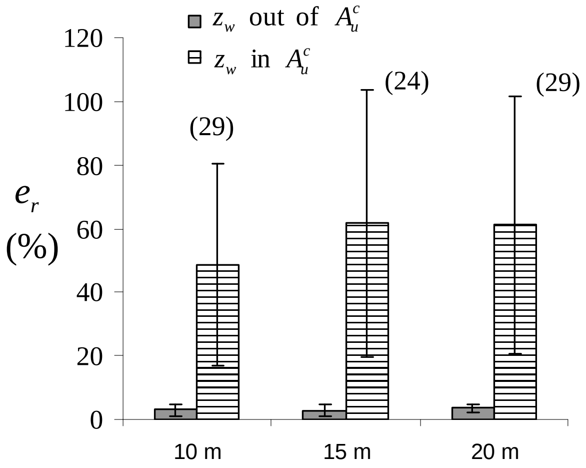


Fig. 35. The effectiveness of depth error reduction with different depth of obstacles. The height of the bar is the mean value of e_r and the vertical interval represents the variance of e_r. The number in the parenthesis is the trial number.

depth error in (4.61) of the obstacles that are either inside the A_u^c or outside the A_u^c. This comparison is performed with different depth of objects. The results are shown in Fig. 35. We repeat the test over 20 times with different random configurations of obstacles locations. The trial number is shown above the bars in the figure. The mean and the variance of the relative depth error are significantly reduced if the robot stay outside A_u^c.

F. Summary

We applied an untrusted area for monocular vision to 3D scene reconstruction with a two-view approach. We presented an untrusted area for a 360 degree view. The untrusted area can result in failures in object detection. Hence, we propose an algorithm to avoid untrusted areas for a 360 degree view by computing the location of the additional camera. This analysis was verified with experiments.

CHAPTER VI

CONCLUSIONS AND FUTURE WORK

A. Conclusions

We addressed a vision-based navigation system for ill-structured roads based on two elements: appearance information and geometric information. In case of the appearance information-based navigation system, we developed a vision-based motion planning algorithm for an autonomous motorcycle. To efficiently process video data and perform motion planning, we propose the V^2-Space, a new framework that represents road features and allows fast construction and motion planning. We used a shadow and illumination invariant color model to construct the V^2-Space to reduce the impact of varying lighting conditions in an outdoor environment. We extracted directional information from the prior tire tracks and pedestrian footsteps on the road to refine our V^2-Space. The V^2-Space also allows us to consider vehicle kinematic, dynamic, and time-delays in motion planning to fit the highly dynamic requirement of the motorcycle. We propose a V^2-Space construction and a motion planning algorithm that runs linear to the number of pixels. The algorithm is tested both with video clips from the desert and in field experiments. It outputted correct robot motion commands at a rate of more than 90%. Failures resulted from the limitations of the appearance information-based navigation that does not provide geometric information. Therefore, we expanded the navigation system to include consideration of geometric information, especially concerning depth.

In the vision-based navigation based on geometric information, depth information is obtained with monocular vision. We computed depth using images taken from different camera perspectives. We analyzed depth error range distribution across the camera coverage for a mobile robot equipped with a single camera. For SFM-based stereo vision for navigation, we showed that the depth error can be excessively large and hence can cause collisions in robot navigation. We modeled the untrusted area where the depth error range is beyond a preset threshold. We then presented an algorithm that enables the robot to select the navigation regions to avoid regions with excessive depth error. We implemented the algorithm and conducted physical experiments. The results confirmed our analysis. Although we computed the untrusted areas based on a monocular vision system, the untrusted area can be applied to vision systems with multiple cameras or a camera network to control the depth error.

The untrusted area for monocular vision was applied to 3D scene reconstruction with a two-view approach. We presented the untrusted area when a 360 degree view is reconstructed. To reduce the risk of failures in object detection within the untrusted area, we propose the algorithm to avoid the untrusted area for a 360 degree view. The proposed algorithm computes the available locations for the additional camera to reduce depth error. Experiments confirmed our analysis.

B. Future Work

We will consider incorporating the V^2-Space in a stereo vision system. We will perform partial construction of the real 3D environment in the V^2-Space to allow fast computations. We will also incorporate machine learning techniques into the V^2-Space to improve the vehicle's capability of adapting to different terrains.

In the geometric information-based navigation, our research uses a single additional frame to control depth error. More frames can be used to further reduce the depth error. There is an interesting tradeoff between the computation time and the accuracy of computation result. We assume known accurate motion information from robot motion sensors and camera potentiometers. This assumption might not be the true for for low-cost miniature robots. It is interesting to extend the model to study how those errors would affect the depth error. We are plane to incorporate the DEAN algorithm in a navigation system to compute the robot position on the fly.

REFERENCES

[1] C. Rasmussen, “Grouping dominant orientations for ill-structured road following,” in *IEEE Computer Society Conference on Computer Vision and Pattern Recognition*, Washington, DC, June 2004, pp. 470–477.

[2] G. N. Desouza and A. C. Kak, “Vision for mobile robot navigation: A survey,” *IEEE Transactions on Pattern Analysis and Machine Intelligence*, vol. 24, no. 2, pp. 237–267, February 2002.

[3] M. Bertozzi, A. Broggi, M. Cellario, A. Fascioli, P. Lombardi, and M. Porta, “Artificial vision in road vehicles,” *Proceedings of IEEE*, vol. 90, pp. 1258–1271, 2002.

[4] A. Remazeilles, F. Chaumette, and P. Gros, “Robot motion control from a visual memory,” in *IEEE International Conference on Robotics and Automation*, New Orleans, LA, USA, April 2004, pp. 4695–4700.

[5] “No hands across america,” Website, 1995, www.cs.cmu.edu/afs/cs/usr/tjochem/www/nhaa/nhaa_home_page.html.

[6] A. Broggi, M. Bertozzi, A. Fascioli, and G. Conte, *Automatic Vehicle Guidance: The Experience of The ARGO Autonomous Vehicle*. New Jersey: World Scientific Co., 1999.

[7] E. R. Davies, *Machine Vision: Theory, Algorithms, Practicalities (2nd edition)*. New York: Academic Press, 1997.

[8] R. Aufrere, J. Gowdy, C. Mertz, C. Thorpe, C.-C. Wang, and T. Yata, “Perception for collision avoidance and autonomous driving,” *Mechatronics*, vol. 13, no. 10, pp. 1149–1161, 2003.

[9] Z. Sun, G. Bebis, and R. Miller, “On-road vehicle detection: A review,” *IEEE Transactions on Pattern Analysis and Machine Intelligence*, vol. 28, no. 5, pp. 694–711, May 2006.

[10] A. Broggi and S. Berte, “Vision-based road detection in automotive systems: A real-time expectation-driven approach,” *Artificial Intelligence Research*, vol. 3, pp. 325–348, 1995.

[11] M. Bertozzi and A. Broggi, “Gold: A parallel real-time stereo vision system for generic obstacle and lane detection,” *IEEE Transactions on Image Processing*, vol. 7, no. 1, pp. 62–81, January 1998.

[12] M. Ekinci, F. W. J. Gibbs, and B. T. Thomas, “Knowledge-based navigation for autonomous road vehicles,” *Turkish Journal of Electrical Engineering & Computer Sciences*, vol. 8, no. 1, pp. 1–29, May 2000.

[13] Y. He, H. Wang, and B. Zhang, “Color-based road detection in urban traffic scenes,” *IEEE Transactions on Intelligent Transportation Systems*, vol. 5, no. 4, pp. 309– 318, December 2004.

[14] D. Pomerleau, “Ralph: Rapidly adapting lateral position handler,” in *Intelligent Vehicles Symposium*, Detroit, MI, September 1995, pp. 506–511.

[15] 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.

[16] M. Freese, S. Singh, E. Fukushima, and S. Hirose, “Bias-tolerant terrain following method for a field deployed manipulator,” in *IEEE International Conference on Robotics and Automation*, Orlando, FL, May 2006, pp. 175–180.

[17] R. Volpe, T. Estlin, S. Laubach, C. Olson, and J. Balaram, "Enhanced mars rover navigation techniques," in *IEEE International Conference on Robotics and Automation*, vol. 1, San Francisco, CA, USA, April 2000, pp. 926–931.

[18] L. M. Lorigo, R. A. Brooks, and W. E. L. Grimsou, "Visually-guided obstacle avoidance in unstructured environments," in *IEEE/RSJ International Conference on Intelligent Robots and Systems*, vol. 1, Grenoble, France, September 1997, pp. 373 – 379.

[19] M. Stephens, R. Blissett, D. Charnley, E. Sparks, and J. Pike, "Outdoor vehicle navigation using passive 3D vision," in *IEEE Computer Society Conference on Computer Vision and Pattern Recognition*, Santa Barbara, CA, June 1989, pp. 556 – 562.

[20] L. Matthies, T. Litwin, K. Owens, A. Rankin, K. Murphy, D. Coorobs, J. Gilsinn, T. Hong, S. Legowik, M. Nashman, and B. Yoshimi, "Performance evaluation of UGV obstacle detection with ccd/flir stereo vision and ladar," in *IEEE ISIC/CIRA/ISAS Joint Conference*, Gaithersburg, MD, September 1998, pp. 658–670.

[21] J. Ibanez-Guzman, X. Jian, A. Malcolm, Z. Gong, C. Chan, and A. Tay, "Autonomous armoured logistics carrier for natural environments," in *IEEE/RSJ International Conference on Intelligent Robots and Systems*, Sendai, Japan, Sept. 2004, pp. 473–478.

[22] C. Dima, N. Vandapel, and M. Hebert, "Classifier fusion for outdoor obstacle detection," in *IEEE International Conference on Robotics and Automation*, vol. 1, New Orleans, LA, USA, April 2004, pp. 665 – 671.

[23] J. Crisman and C. Thorpe, "UNSCARF, a color vision system for the detection of unstructured roads," in *IEEE International Conference on Robotics and Automation*, vol. 3, Sacramento, CA, USA, April 1991, pp. 2496 – 2501.

[24] D. Lieb, A. Lookingbill, and S. Thrun, "Adaptive road following using self-supervised

learning and reverse optical flow," in *Proceediings of Robotics:Science and Systems*, June 2005.

[25] C. Rasmussen, "Combining laser range, color, and texture cues for autonomous raod following," in *IEEE International Conference on Robotics and Automation*, vol. 4, Washington, DC, USA, May 2002, pp. 4320 – 4325.

[26] ——, "A hybrid vision + ladar rural road follower," in *IEEE International Conference on Robotics and Automation*, Orlando, FL, May 2006, pp. 156–161.

[27] M. Kolesnik, G. Paar, A. Bauer, and M. Ulm, "Algorithmic solution for autonomous vision-based off-road navigation," in *SPIE: Enhanced and Synthetic Vision*, vol. 3364, Orlando, FL, April 1998, pp. 230–247.

[28] J. Michels, A. Saxena, and A. Ng, "High speed obstacle avoidance using monocular vision and reinforcement learning," in *22nd International Conference on Machine Learning*, Bonn, Germany, August 2005, pp. 593 – 600.

[29] I. Ulrich and I. Nourbakhsh, "Appearance-based obstacle detection," in *AAAI National Conference on Aritificial Intelligence*, Austin, TX, 2000.

[30] C. Thorpe, M. Herbert, T. Kanade, and S. Shafer, "Vision and navigation for the carnegi-mellon navlab," *IEEE Transactions on Pattern Analysis and Machine Intelligence*, vol. 10, no. 3, pp. 362–372, May 1988.

[31] P. Batavia and S. Singh, "Obstacle detection using adaptive color segmentation and colr stereo homography," in *IEEE International Conference on Robotics and Automation*, vol. 1, Seoul, Korea, May 2001, pp. 705–710.

[32] K. Ohno, T. Tsubouchi, S. Maeyama, and S. Yuta, "A mobile robot campus walkway following with daylight-change-proof walkway color image segmentation," in

IEEE/RSJ International Conference on Intelligent Robots and Systems, vol. 1, Maui, HI, USA, October 2001, pp. 77 – 83.

[33] A. Zhang and R. Russell, "Dominant orientation tracking for path following," in *IEEE/RSJ International Conference on Intelligent Robots and Systems*, Alberta, Canada, August 2005, pp. 3885–3889.

[34] S. Buluswar and B. Draper, "Color machine vision for autonomous vehicles," *International Journal for Engineering Applications of Artificial Intelligence*, vol. 1, no. 2, pp. 245–256, April 1998.

[35] T. Hong, C. Rasmussen, T. Chang, and M. Shneier, "Road detection and tracking for autonomous mobile robots," in *The SPIE 16th Annual International Symposium on Aerospace/Defense Sensing, Simulation, and Controls*, Orlando, FL, April 2002.

[36] G. Grudic and J. Mulligan, "Outdoor path labeling using polynomial mahalanobis distance," in *Robotics: Science and Systems*, Philadelphia, PA, June 2006.

[37] G. Avina-Cervantes, M. Devy, and A. Marin-Hernandez, "Lane extraction and tracking for robot navigation in agricultural applications," in *IEEE International Conference on Advanced Robotics*, Coimbra, Portual, June 2003, pp. 816–821.

[38] H. Dahlkamp, A. Kaehler, D. Stavens, S. Thrun, and G. Bradski, "Self-supervised monocular road detection in desert terrain," in *Robotics: Science and Systems*, Philadelphia, PA, June 2006.

[39] M. Happold, M. Ollis, and N. Johnson, "Enhancing supervised terrain classification with predictive unsupervised learning," in *Robotics: Science and Systems*, Philadelphia, PA, June 2006.

[40] H. Zhang and J. P. Ostrowski, “Visual motion planning for mobile robots,” *IEEE Transactions on Robotics and Automation*, vol. 18, no. 2, pp. 199–208, April 2002.

[41] N. J. Cowan, J. D. Weingarten, and D. E. Koditschek, “Visual servoing via navigation functions,” *IEEE Transactions on Robotics and Automation*, vol. 18, no. 4, pp. 521–533, August 2002.

[42] J. Santos-Victor, G. Sandini, F. Curotto, and S. Garibaldi, “Divergent stereo for robot navigation: Learning from bees,” in *IEEE Computer Society Conference on Computer Vision and Pattern Recognition*, New York, USA, June 1993, pp. 434–439.

[43] D. Mateus, G. Avina, and M. Devy, “Robot visual navigation in semi-structred outdoor environments,” in *IEEE International Conference on Robotics and Automation*, Barcelona, Spain, April 2005, pp. 4691–4696.

[44] J. Borenstein and Y. Koren, “The vector field histogram- fast obstacle avoidance for mobile robots,” *IEEE Transactions on Robotics and Automation*, vol. 7, no. 3, pp. 278–288, June 1991.

[45] D. A. Forsyth and J. Ponce, *Computer Vision A Modern Approach.* New Jersey: Prentice Hall, 2003.

[46] Y. Ma, J. Kosecka, and S. Sastry, “Vision guided navigation for a nonholonomic mobile robot,” *IEEE Transactions on Robotics and Automation*, vol. 15, no. 3, pp. 512–536, June 1999.

[47] J. Yi, D. Song, A. Levandowski, and S. Jayasuriya, “Trajectory tracking and balance stabilization control of autonomous motorcycles,” in *IEEE International Conference on Robotics and Automation*, Orlando, Fl, May 2006, pp. 2583– 2589.

[48] D. Pomerleau, *Neural Network Perception for Mobile Robot Guidance*. Dordrecht: Kluwer Academic, 1993.

[49] T. Gevers and A. W. M. Smeulders, "Color-based object recognition," *Pattern Recognition*, vol. 32, no. 3, pp. 453–464, 1999.

[50] R. Herbrich, *Learning Kernel Classifiers: Theory & Algorithms*. Cambridge, MA: MIT, 2002.

[51] F. Schaffalitzky and A. Zisserman, "Viewpoint invariant texture matching and wide baseline stereo," in *The 18th international Conference on Computer Vision*, Vancouver, Canada, July 2001, pp. 636–643.

[52] F. Chaumette, S. Boukir, P. Bouthemy, and D. Juvin, "Structure from controlled motion," *IEEE Transactions on Pattern Analysis and Machine Intelligence*, vol. 18, no. 5, pp. 492–504, May 1996.

[53] P. Chang and M. Herbert, "Robust tracking and structure from motion with sample based uncertainty representation," in *IEEE International Conference on Robotics and Automation*, vol. 3, Washington, DC, USA, May 2002, pp. 3030 – 3037.

[54] R. Bajcsy, "Active perception," *Proceedings of the IEEE*, vol. 76, pp. 996–1005, August 1988.

[55] B. Zavidovique, "First steps of robotic perception: The turning point of the 1990s," *Proceedings of the IEEE*, pp. 1094–1112, July 2002.

[56] K. Tarabanis, P. Allen, and R. Tsai, "A survey of sensor planning in computer vision," *IEEE Transactions on Robotics and Automation*, vol. 11, no. 1, pp. 86–104, February 1995.

[57] E. Mouragnon, M. Lhuillier, M. Dhome, F. Dekeyser, and P. Sayd, "Monocular vision based slam for mobile robots," in *The 18th International Conference on Pattern Recognition*, Hong Kong, August 2006, pp. 1027–1031.

[58] Z. Chen, R. Rodrigo, and J. Samarabandu, "Implementation of an update scheme for monocular visual slam," in *International Conference on Information and Automation*, Shandong, China, December 2006, pp. 212–217.

[59] E. Mortard, B. Raducanu, V. Cadenat, and J. Vitria, "Incremental on-line topological map learning for a visual homing application," in *IEEE International Conference on Robotics and Automation*, Roma, Italy, April 2007, pp. 2049–2054.

[60] T. Lemaire and S. Lacroix, "Monocular-vision based slam using line segments," in *IEEE International Conference on Robotics and Automation*, Roma, Italy, April 2007, pp. 2791–2796.

[61] E. Royer, J. Bom, B. T. Michel Dhome, M. Lhuillier, and F. Marmoiton, "Outdoor autonomous navigation using monocular vision," in *IEEE/RSJ International Conference on Intelligent Robots and Systems*, Alberta, Canada, August 2005, pp. 1253–1258.

[62] Z. Chen and S. T. Birchfield, "Qualitative vision-based mobile robot navigation," in *IEEE International Conference on Robotics and Automation*, Orlando, FL, May 2006, pp. 2686– 2692.

[63] D. Song, H. Lee, J. Yi, and A. Levandowski, "Vision-based motion planning for an autonomous motorcycle on ill-structured roads," *Autonomous Robots*, vol. 23, no. 3, pp. 197–212, October 2007.

[64] A. Azarbayejani and A. Pentland, "Recursive estimation of motion, structure, and focal length," *IEEE Transactions on Pattern Analysis and Machine Intelligence*, vol. 17,

no. 6, pp. 562–575, June 1995.

[65] T. Jebara, A. Azarbayejani, and A. Pentland, “3D structure from 2D motion,” *IEEE Signal Processing Magazine*, vol. 16, no. 3, pp. 66–84, May 1999.

[66] C. Tomasi and T. Kanade, “Shape and motion from image streams under orthography: A factorization method,” *International Journal of Computer Vision*, vol. 9, no. 2, pp. 137–154, November 1992.

[67] D. Martinec and T. Pajdla, “3D recontruction by fitting low-rank matrices with missing data,” in *IEEE Conference on Computer Vision and Pattern Recognition*, San Diego, CA, June 2005, pp. 198–205.

[68] S. Brandt, “Closed-form solutions for affine reconstruction under missing data,” in *7th European Conference on Computer Vision*, Copenhagen, Denmark, May 2002, pp. 109–114.

[69] R. Hartley and F. Schaffalizky, “Powerfactorization: 3D reconstruction with missing or uncertain data,” in *Australia-Japan Advanced Workshop on Computer Vision*, Australia, September 2003.

[70] N. Guilbert and A. Bartoli, “Batch recovery of multiple views with missing data using direct sparse solvers,” in *British Machine Vision Conference*, Norwich, UK, September 2003.

[71] P. Anandan and M. Irani, “Factorization with uncertainty,” *International Journal of Computer Vision*, vol. 49, no. 3, pp. 101–116, October 2002.

[72] B. Triggs, “Plane+parallax, tensors and factorization,” in *6th European Conference on Computer Vision*, Dublin, Ireland, June 2000, pp. 522 – 538.

[73] M. Irani, P. Anandan, and M. Cohen, “Direct recovery of planar-parallax from multiple frames,” *IEEE Transactions on Pattern Analysis and Machine Intelligence*, vol. 24, no. 11, pp. 1528–1534, November 2002.

[74] C. Rother and S. Carlsson, “Linear multi view reconstruction and camera recovery using a reference plane,” *International Journal of Computer Vision*, vol. 49, no. 3, pp. 117–141, October 2002.

[75] A. Bartoli and P. Sturm, “Contrained structure and motion from multiple uncalibrated views of a piecewise planar scene,” *International Journal of Computer Vision*, vol. 52, no. 1, pp. 45–64, April 2003.

[76] F. Dellaert, S. M. Seitz, C. E. Thorpe, and S.Thrun, “Structure from motion without correspondence,” in *IEEE Conference on Computer Vision and Pattern Recognition*, Hilton Head, SC, June 2000, pp. 557 – 564.

[77] A. K. R. Chowdhury and R. Chellappa, “Statistical bias in 3-D reconstruction from a monocular video,” *IEEE Transactions on Image Processing*, vol. 14, no. 8, pp. 1057 – 1062, August 2005.

[78] J. Aloimonos and I. Weiss, “Active vision,” *International Journal of Computer Vision*, vol. 1, no. 4, pp. 333–356, January 1987.

[79] J. Aloimonos, “Purposive and qualitative active vision,” in *International Conference on Pattern Recognition*, Atlantic City, NJ, June 1990, pp. 346–360.

[80] P. Whaite and F. Ferrie, “Autonomous exploration: Driven by uncertainty,” *IEEE Transactions on Pattern Analysis and Machine Intelligence*, vol. 19, no. 3, pp. 193–205, March 1997.

[81] M. K. Reed and P. K. Allen, “Constraint-based sensor planning for scene modeling,” *IEEE Transactions on Pattern Analysis and Machine Intelligence*, vol. 22, no. 12, pp. 1460–1467, December 2000.

[82] E. Marchand and F. Chaumette, “Active vision for complete scene reconstruction and exploration,” *IEEE Transactions on Pattern Analysis and Machine Intelligence*, vol. 21, no. 1, pp. 65–72, January 1999.

[83] S. Sakai, K. Osuka, T. Maekawa, and M. Umeda, “Active vision of a heavy material handling agricultural robot using robust control: A case study for initial cost problem,” in *2005 IEEE/RSJ International Conference on Intelligent Robots and Systems*, Alberta, Canada, August 2005, pp. 572– 578.

[84] H. Chen and Z. Xu, “Local 3D map building and error analysis based on stereo vision,” in *IEEE Conference on Industrial Electronics Society*, Raleigh, NC, November 2005, pp. 379–382.

[85] Y. F. Li and Z. Liu, “Information entropy-based viewpoint planning for 3-D object reconstruction,” *IEEE Transactions on Robotics*, vol. 21, no. 3, pp. 324–337, June 2005.

[86] E. Dunn, G. Olague, and E. Lutton, “Parisian camera placement for vision metrology,” *Pattern Recognition Letters*, vol. 27, no. 11, pp. 1209–1219, August 2006.

[87] G. Olague and R. Mohr, “Optimal camera placement for accurate reconstruction,” *Pattern Recognition*, vol. 35, pp. 927–944, 2002.

[88] E. Dunn and G. Olague, “Pareto optimal camera placement for automated visual inspection,” in *IEEE International Conference on Robotics and Automation*, Barcelona, Spain, August 2005, pp. 3821 – 3826.

[89] A. Shmuel and M. Werman, "Active vision: 3D from an image sequence," in *10th International Conference on Pattern Recognition*, Atlantic City, NJ, June 1990, pp. 48–54.

[90] R. Szeliski and S. B. Kang, "Shape ambiguities in structure from motion," *IEEE Transactions on Pattern Analysis and Machine Intelligence*, vol. 19, no. 5, pp. 506–512, May 1997.

[91] Z. Sun, V. Ramesh, and A. M. Tekalp, "Error characterization of the factorization method," *Computer Vision and Image Understanding*, vol. 82, no. 2, pp. 110–137, May 2001.

[92] P. Firoozfam and S. Negahdaripour, "Theoretical accuracy analysis of n-ocular vision systems for scene reconstruction, motion estimation, and positioning," in *2nd International Symposium on 3D data processing, visualization, and Transmission*, Thessaloniki, Greece, September 2004, pp. 888–895.

[93] W. Kim, A. Ansar, R. Steele, and R. Steinke, "Performance analysis and validation of a stereo vision system," in *IEEE International Conference on Systems, Man and Cybernetics*, October 2005, pp. 1409 – 1416.

[94] G. Young and R. Chellappa, "Statistical analysis of inherent ambiguities in recovering 3D motion from a noisy flow field," *IEEE Transactions on Pattern Analysis and Machine Intelligence*, vol. 14, no. 10, pp. 995–1013, October 1992.

[95] K. Daniilidis and H. Nagel, "The coupling of rotation and translation in motion estimation of planar surfaces," in *IEEE Conference on Computer Vision and Pattern Recognition*, New York, USA, June 1993, pp. 188–193.

[96] Y. Xiong and L. Matthies, "Error analysis of a real-time stereo system," in *IEEE*

Conference on Computer Vision and Pattern Recognition, San Juan, Puerto Rico, June 1997, pp. 1087–1093.

[97] R. Hartley and A. Zisserman, *Multiple View Geometry in Computer Vision*. UK: Cambridge University Press, 2003.

[98] J. Y. Bouguet, "Camera calibration toolbox for matlab," Website, 2007, www.vision.caltech.edu/bouguetj/calib_doc/index.html.

VITA

Hyun Nam Lee earned a B.S. degree and an M.S. degree in Control and Instrumentation Engineering from Hanyang University, Ansan, Korea in February 1998 and 2000. She worked at LG Electronics located in Seoul, Korea from 2000 to 2002. She received a Ph.D. from Texas A&M University in August 2008. Her research interest is vision-based robot navigation. Hyun Nam Lee can be reached at hyunnamlee@gmail.com, and her permanent address is 462-27 Anyang-8-dong Anyang-si Kyunggi-do South Korea 430-018.

The typist for this dissertation was Hyun Nam Lee.