UNIVERSITY OF CALIFORNIA
RIVERSIDE

LIDAR, Camera and Inertial Sensors Based Navigation Techniques for Advanced Intelligent Transportation Systems

A Dissertation submitted in partial satisfaction
of the requirements for the degree of

Doctor of Philosophy

in

Electrical Engineering

by

Lili Huang

December 2010

Dissertation Committee:

Professor Matthew Barth, Chairperson
Professor Jay Farrell
Professor Amit Roy-Chowdhury

The Dissertation of Lili Huang is approved by

_____

_____

_____

Committee Chairperson

University of California, Riverside

Acknowledgement

The first person I should express my deep gratitude is my advisor, Prof. Matthew Barth, for his utmost guidance, advice, support, patience and encouragement I received throughout my research. He introduced me into the Intelligent Transportation System (ITS) society and helped me tremendously from the first day I joined the Transportation Systems Research (TSR) group. Prof. Barth's sharp insight, system-level thinking, and his talent of seeing the big picture are always inspirations to me. I am grateful, thankful, and honored for all the valuable learning and advices he has given me.

I would like to thank Prof. Jay Farrell, for his extensive knowledge, sense of humor, and his helpful suggestion and comments for making my dissertation even better. His valuable help and encouragement make it possible for me to finish the majority part of the FHWA project in a very short time.

I would also give my appreciation to Dr. Kanok Boriboonsomsin, Mike Todd and Anh Vu for all the help, discussion, and suggestions I got during my Ph.D. I interacted with Dr. Boriboonsomsin and Mike Todd for several years on various research projects and learned a lot from their professionalism of being a researcher. I am happy to have worked with Anh Vu for more than three years. His help in the vision sensor and LIDAR system development makes this dissertation possible.

I am indebted to my committee members Prof. Gerardo Beni and Prof. Amit K. Roy-Chowdhury, for the helpful discussion, patient teaching and the encouragement since the first year in my graduate school.

I am also very grateful to my coworkers and lab mates at CE-CERT and Dept. of Electrical Engineering, especially Amanda Gomes, Dr. Jie Du, Dr. Meng Cao, Dr. Weihua Zhu, George Scora, Alex Vu, Qichi Yang, Dr. Huan Liu, Haiyu Zhang, Yulin Zhang, Anning Chen, Arvind Ramanandan, Behlul Sutarwala, Akshay Morye, Yiqian Li, Lingfei Zhou, Dongfang Zheng, Yiming Chen, Mingyang Li and Teddy Yap Jr. It's my honor to work with them during my time at UC Riverside. I would also thank everyone at CE-CERT for make it a nice place to learn and work.

I have had a wonderful time during my time in Riverside, largely due to my great friends here. I would like to thank my first roommates, Ling Chang and Xiaobin Niu, and their adorable daughter Emma. I cherish you all. Special thanks to Kezhu Hong, who carpools with me from/to Los Angeles and Riverside for more than one year. I also would like to express my gratitude to Haijun Tian and Nan Sun, Bi Song, Yilei Xu, Yunfan Li, Zhuo Zhao, Ning Mi, Pu Liu, Boyuan Yan, Jing Tang, Yi Huang, Yuan Yu, Ting Kong, Yanhua Wu, and Yehong Wang among others. All their help is sincerely appreciated.

I could not have afforded this Ph.D. program without financial support, and I therefore would like to acknowledge the sponsors: The Department of Electrical Engineering of UC Riverside, CE-CERT at UC Riverside, and the UC Transportation Center (UCTC).

Finally, thanks to my parents, my husband and my son, for their endless and unconditional love, faith and support that they have always given me. Special thanks to my husband, Xiaoyan Li, who has stood with me through all the ups and downs. I dedicate this dissertation to them.

ABSTRACT OF THE DISSERTATION

LIDAR, Camera and Inertial Sensors Based Navigation Techniques for Advanced Intelligent Transportation Systems

by

Lili Huang

Doctor of Philosophy, Graduate Program in Electrical Engineering
University of California, Riverside, December 2010
Professor Matthew Barth, Chairperson

During the past decade, numerous research has been carried out in in-vehicle navigation and positioning. All the approaches are trying to solve two problems: "*where am I?*", and "*where are they?*", i.e., the automatic vehicle positioning, and the surrounding vehicle detection and tracking.

Among a variety of sensor based systems, computer vision-based approaches have been one of the most popular and promising techniques, however it suffers from intensity variations, narrow fields of view, and low-accuracy depth information. Laser Ranging Sensor (LIDAR) is another attractive technology due to its high accuracy in ranging, its wide-area view, and low data-processing requirements. However, a major challenge for LIDAR-based systems is that their reliability depends on the distance and reflectivity of different objects. Moreover, LIDAR often suffers from noise issues, making it difficult to distinguish between different kinds of objects. In this dissertation, we address several fundamental problems in integrating LIDAR and camera systems for better navigation and positioning solutions. As part of the research, we present a sensor fusion system to

solve the “where are they” problem. The calibration of the sensor fusion system as well as the vehicle detection and tracking algorithms are proposed to determine the states of surrounding vehicles. The “where am I” solution focuses on the integration of LIDAR and inertial sensors for advanced vehicle positioning. Moreover, a vehicle tracking approach is presented for freeway traffic surveillance system.

Sensor fusion techniques have been used for years to combine sensory data from disparate sources. In this dissertation, a tightly coupled LIDAR/CV integrated system is introduced. LIDAR and camera calibration is the key component of sensor fusion system. A unique multi-planar LIDAR and computer vision calibration algorithm has been developed, which requires that the camera and LIDAR observe a planar pattern at different positions and orientations. Geometric constraints of the different ’views’ of the LIDAR and camera images are resolved as the coordinate transformation and rotation coefficients.

The proposed sensor fusion system is utilized for mobile platform based vehicle detection and tracking. The LIDAR sensor estimates possible vehicle positions. Different Regions of Interest (ROIs) in the imagery are defined based on the LIDAR object hypotheses. An Adaboost object classifier is then utilized to detect the vehicle in ROIs. Finally, the vehicle’s position and dimensions are derived from both the LIDAR and image data. Experimental results are presented to illustrate that this LIDAR/CV system is reliable.

In addition, an autonomous positioning solution for urban environment is provided in this dissertation. The positioning solution is derived by combining measurements from

both LIDAR and inertial sensors, i.e., LIDAR, gyros and accelerometers. The inertial sensors provide the angular velocities as well as the accelerations of the vehicle, while LIDAR detects the landmark structures (posts and surfaces). In our implementation the positioning is performed in known environment, i.e., the map information is assumed to be *a priori* information. Extended Kalman Filter (EKF) is implemented in the positioning estimation.

This dissertation also presents a vehicle tracking approach used in a traffic surveillance system. One of the key challenges with freeway vehicle tracking is dealing with high density traffic, where occlusion often leads to foreground splitting and merging errors. We propose a real-time multi-vehicle tracking approach, which combines both local feature tracking and a global color probability model. In our approach, the corner features are tracked to provide position estimates of moving objects. Then a color probability is calculated in the occluded area to determine which object each pixel belongs to. This approach has been proved to be scalable to both stationary surveillance video and moving camera video.

Contents

List of Figures

List of Table

Chapter 1

Introduction

1.1 Motivation

During the past decade, numerous research work has been carried out in the autonomous vehicle area. Autonomous vehicles are one of the key eventual goals of Intelligent Transportation Systems (ITS). An autonomous vehicle navigates and drives entirely on its own without any human driver or remote control. It is capable of driving in traffic, performing complex maneuvers such as merging, parking, and making intelligent path planning. It is designed to finish the “dull, dirty or dangerous” tasks that can be carried out by machines instead of a human operator [11].

A variety of new sensing technologies have been developed and enhanced to be used for autonomous vehicle navigation. Among all the sensor systems, computer vision-based approach is one of the most widely used and promising techniques. Laser Ranging Sensor (LIDAR) is another attractive technology due to its high accuracy in ranging, wide-area field of view, and low data-processing requirements [1]. The other sensors used in navigation include Global Positioning System (GPS), radar, and loop sensors. A brief comparison of the available sensor technologies is given in Table 1-1.

Sensor fusion is implemented in the vehicle navigation system to gather information

from the far-field and near-field sensors, and combine them in a meaningful way [12]. The output of sensor fusion system should be the vehicle's own position and the state of the objects around it.

Table 1-1. Performance Comparison of Existing Sensor Technologies used in ITS [6].

Sensor Technology	Advantage	Disadvantage
LIDAR	• Detect distance and angle with high accuracy • Low data processing requirement • Operational in fog and rain	• High cost • Limited detection range • Difficult to classify the object
Radar	• Direct measurement of speed or distance • Compact Size • Low data processing requirement	• Relatively low precision • Limited field of view • May have identification problem in multi-lane applications • Medium cost
Video Camera	• Provide real-time image of traffic • Low cost • Multiple lanes observed • No traffic interruption for installation and repair • Large field of view	• High requirement for data processing and storage • Different algorithms required for day and night time • Susceptible to atmospheric obscurants and weather change
Infrared Camera	• Day and night operation • Operational in fog	• High requirement for data processing • Susceptible to weather change
Inductive Loop Detector	• Low cost per-unit • Large experience base	• Installation and maintenance require traffic disruption • Easily damaged

1.2 LIDAR, Camera and Inertial Sensors Based Navigation Systems

The vehicle navigation systems are trying to solve two challenging problems: "*where am I?*" and "*where are they?*". The "*where am I?*" problem is to find the vehicle's

location using GPS, landmarks with *a prior* position information, digital maps or the other sensing technologies. The "*where are they?*" problem is to determine the surrounding vehicle's position, speed and trajectory. The autonomous vehicle is then able to design its short term and long term paths based on the answers to these two questions.

In this dissertation, we propose several multi- sensors based navigation techniques to address these two problems. The approaches we present include calibration of the LIDAR and computer vision systems, vehicle detection using the sensor fusion technique, vehicle automatic localization with *a prior* landmark position, and the freeway traffic surveillance techniques. In the following subsections, we will introduce the problems and briefly discuss the approaches we take for each problem.

1.2.1 LIDAR and Camera Calibration

Mobile sensing systems consisting of one sensor or a suite of sensors are usually used for detecting traffic conditions. These sensors provide real-time measurements and play an important role in the development of Driver Assistant Systems (DAS).

One of the most common sensing techniques used in vehicle navigation and traffic surveillance is the use of computer vision. Computer vision can provide a large amount of information on the surrounding environment. However, it suffers from intensity variations, narrow fields of view, and low-accuracy depth information [2]. On the other side, LIDAR measures the distance and relative angle from the sensor to the target by

calculating the time-of-flight. The accuracy of its measurements depends on the size and reflectivity of the target, so the probability of precise detection decreases with distance [1]. Since their characteristics complement each other, it is logical to utilize both computer vision and LIDAR for detecting different objects.

Sensor fusion systems are commonly used to integrate the sensory data from disparate sources, so that the output will be more accurate and complete in comparison to the output of one sensor. In order to effectively extract and integrate 3D information from both computer vision and LIDAR systems, the relative position and orientation between these two sensor modalities should be obtained. Sensor calibration is to identify the parameters that describe the relative geometric transformation [3][4], which is a key step in the sensor fusion systems. However, current calibration methods work only for visible beam LIDAR, 3D LIDAR and 2D LIDAR. To date, there does not exist any convenient calibration methods for multi-planar 'invisible-beam' LIDAR and computer vision systems.

As part of this dissertation, a novel calibration approach of a camera with a multi-planar LIDAR is proposed, in which the laser beams are invisible to the camera. The camera and LIDAR are required to observe a planar pattern at different positions and orientations. Geometric constraints of the 'views' from the LIDAR and camera images are resolved as the coordinate transformation coefficients. The proposed approach

consists of two stages: solving a closed-form equation, followed by applying a non-linear algorithm based on a maximum likelihood criterion. Compared with the classical methods which use 'beam-visible' cameras or 3D LIDAR systems, this approach is easy to implement at low cost. The calibration method has been applied to a mobile sensing system with two multi-planar LIDAR sensors and a camera. Both simulation and real world experiments have been carried out to evaluate the performance of this approach.

1.2.2 Mobile Platform Based Vehicle Detection

In many driver assistance systems, both LIDAR and computer vision sensors are commonly used to detect surrounding vehicles. LIDAR provides excellent range information to different objects. However, it is difficult to classify these objects from range information alone. On the other hand, computer vision imagery allows for better recognition, but does not provide high-resolution range information. Therefore, sensor fusion techniques have been used for years to combine sensory data from disparate sources.

In the in-vehicle navigation system, it is necessary that the vehicle knows the location of surrounding vehicles. Determining the position and orientation of the surrounding vehicles is known as the "*where are they?*" problem. We proposed a tightly-coupled LIDAR and computer vision system to solve this problem. The sensing system is mounted on a probe vehicle, as is shown in Fig. 1-1. The pair of LIDAR sensors is

mounted on the front bumper of the vehicle, and the camera is behind the front wind shield.

The LIDAR sensor estimates possible vehicle positions. This information is then transformed into the image coordinates. Difference Regions of Interest (ROIs) in the imagery are defined based on the LIDAR object hypotheses. An Adaboost object classifier is then utilized to classify the vehicle in ROIs. A classifier error correction approach is used to choose an optimal position of the detected vehicle. Finally, the vehicle's position and dimensions are derived from both the LIDAR and image data. Experimental results are presented to illustrate that this LIDAR and computer vision integration system is reliable. The tightly coupled sensor fusion system can be used in applications such as traffic surveillance and roadway navigation tasks.

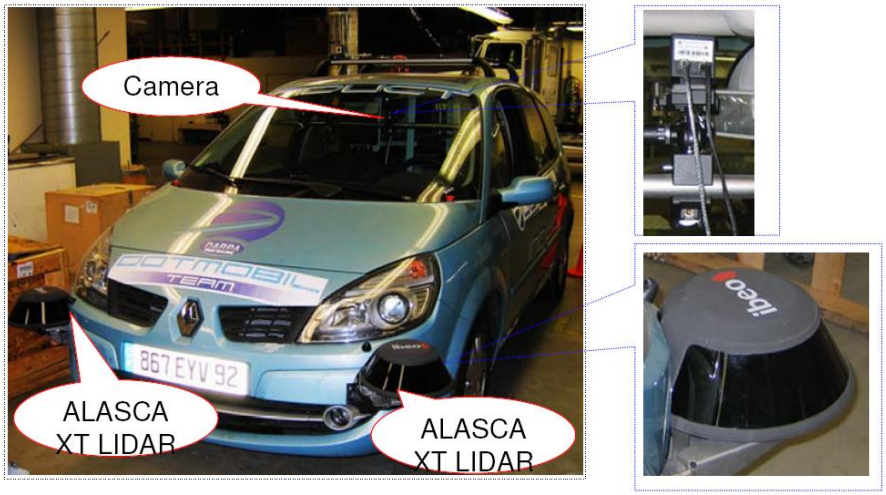


Figure 1-1. The mobile sensing platform. This probe vehicle carries one camera and two IBEO ALASCA XT LIDAR sensors.

1.2.3 LIDAR and Inertial Sensor-Aided Positioning

In order for a vehicle to be able to navigate safely and successfully in a given environment, it is necessary that the vehicle knows its position and orientation. Loss of position information may cause wrong path planning, erroneous behavior, and even hazardous outcomes. Determining the position and orientation is known as the *vehicle localization* problem, or the "*where am I?*" problem. The vehicle localization is considered to be one of the most fundamental and important problems in the intelligent transportation as well as the mobile robotics area [7-9]. A digital map of the environment is usually provided to the vehicle, which is equipped with sensors to perceive itself and the environment.

An autonomous positioning solution for urban environment is provided as part of this dissertation. The positioning solution is derived by combining measurements from both LIDAR and inertial sensors, i.e., LIDAR, gyros, and accelerometers. The inertial sensors provide the angular velocities as well as the accelerations of the vehicle, while LIDAR detects the landmark structures (posts and surfaces). In our implementation the positioning is performed in known environment, i.e., the map information is assumed to be *a priori* information. Extended Kalman Filter (EFK) is used in positioning estimation.

1.2.4 Freeway Traffic Surveillance Using Both Corner and Color Features

Traffic data in general was manually collected by human operators prior to 1970,

when data were manually collected. The manual collecting method has many drawbacks including high cost, low efficiency, and difficulties imposed by staffing limitations [10]. Video image used for traffic surveillance was initiated in U.S., Japan, and Europe [5]. The vision camera output is usually utilized in ITS applications such as vehicle counting, tracking, classification and transportation analysis.

One of the key capabilities that is critical for many surveillance applications is the ability to detect and track vehicles in real-time from a video steam. The vision camera based system works fairly well under light traffic conditions. However, current approaches do not have very good performance under heavy traffic conditions. The main difficulty lies in the fact that the occluded object appearance changes significantly from frame to frame.

As part of this dissertation, we present a vehicle tracking approach used in a traffic surveillance system. In order to help solve the occlusion problem, global features such as color or local features like corners are commonly used for tracking. However, tracking based on global features or local features alone does not work well with a high amount of occlusion. We propose a multi-vehicle tracking approach, which combines both local feature tracking and a global color probability model. In cases with low occlusion, corner feature detection and tracking algorithm is used to estimate vehicle positions and trajectories. When there is a high degree of occlusion, corner features are tracked to

provide position estimates of moving objects. Then a color probability is calculated in the occluded area to determine which object each pixel belongs to. This approach has been proved to be accurate with high efficiency.

1.3 Contributions of the Dissertation

The primary objective of this dissertation is to develop methodologies and systems using LIDAR, computer vision and inertial sensors, in order to determine the position of an instrumental vehicle as well as objects around the vehicle. The dissertation has several major contributions as listed below:

- A unique multi-planar LIDAR and computer vision calibration algorithm has been developed. The camera and LIDAR are required to observe a planar pattern at different positions and orientations. Geometric constraints are solved to obtain the translation and rotation between the two sensors.
- A tightly coupled LIDAR and computer vision integrated system has been developed for vehicle detection and tracking. The LIDAR scanning data are applied for object detection and classifier correction. Moreover, the output of the vision camera is used to provide classification and dimension results.
- An autonomous positioning solution for urban environment is provided in this dissertation. The solution is derived by combining measurements from both LIDAR and inertial sensors. The inertial sensors provide the angular velocities

as well as the accelerations of the vehicle, while LIDAR detects the landmark structures. The positioning is implemented using a EKF.

- This dissertation also presents a vehicle tracking approach used in a traffic surveillance system. In order to solve the occlusion problem in high density traffic, both corner features and color features are utilized in the vehicle tracking. A probability model is proposed to determine which object each pixel belongs to in the occlusion area.

1.4 Organization of the Dissertation

The dissertation is organized as follows: Chapter 2 reviews background and related work, including the calibration methods for LIDAR and camera, sensor fusion based vehicle detection and tracking algorithms, vehicle localization methods, and traffic surveillance techniques on freeway. In Chapter 3, we focus on the calibration of LIDAR and camera system. First of all the coordinates of LIDAR and camera are introduced, followed by the mathematical derivation of the geometric relations between the two sensors. Then the equations are solved in two stages: a close-form solution, followed by applying a non-linear algorithm based on the maximum likelihood criterion. Chapter 4 describes the developed sensor fusion based vehicle detection system. Both hardware and data processing methods of the sensor fusion system are introduced in this chapter. In Chapter 5, a vehicle localization approach using LIDAR and inertial sensors is presented.

Some analytical results are also given. Chapter 6 describes a vehicle tracking method for the high traffic density conditions. Both feature and color models are presented in this chapter, and a probability model is proposed. We conclude the dissertation and describe potential future work in Chapter 7.

Chapter 2

Background and Literature Review

The "*where am I?*" and "*where are they?*" solutions aim to estimate the state of both test vehicle and the surrounding vehicles. The vehicle state information includes position, orientation, speed, and acceleration. The problem of state estimation addresses estimating quantities from sensors that are not directly observable [13]. In this chapter, we review the techniques for vehicle detection and tracking on a moving platform, as well as the vehicle positioning approaches. The sensor fusion techniques and the calibration methods will also be discussed. Moreover, the vehicle tracking methods in high density traffic will also be reviewed in this chapter.

We start by discussing a variety of LIDAR sensors, including one planar, multi-planar, as well as 3D LIDAR. As described in the previous chapter, it is valuable to use both computer vision and LIDAR together for various detection and tracking tasks. One of the key problems to solve is calibrating the two sensors so they can be used together. In the following section, we will describe existing LIDAR and computer vision calibration methods, including visible LIDAR beam-based calibration, 3D LIDAR and camera calibration, and the 2D single planar LIDAR calibration. We then review the application of sensor fusion techniques for vehicle detection and tracking systems on a

moving platform. This is followed by a review of vehicle localization methods using GPS, inertial sensors, and LIDAR. Current camera-based freeway traffic surveillance systems are also discussed in this chapter. The final section recapitulates this chapter.

2.1 LIDAR Sensors

A LIDAR is a device which uses laser beams to determine the distance and azimuth from the sensor to an object. The LIDAR sensor is commonly utilized in vehicle navigation for detecting surrounding vehicles, curbs and obstacles. It can also be used in a vehicle localization solution, either as a single sensor or be combined with GPS and Inertial Navigation System (INS).

An example of the most popular LIDAR sensors is the SICK LMS 2xx series [52]. A SICK LIDAR operates at distance up to 80m with an angular resolution of 0.5^{o} and a measurement accuracy of typically 5cm. The distance between the sensor and an object is calculated by measuring the time interval between an emitted laser pulse and reception of the reflected pulse. The amplitude of the received signal is used to determine the reflectivity of the object surface. Moreover, compared to the CCD cameras and RADAR systems, the view angle of a typical LIDAR sensor is larger, e.g., 180^{o}. Fig. 2-1 (a) illustrates the SICK LMS200 LIDAR.

The HOKUYO UXM-30LN LIDAR is another popular single planar range sensor designed for intelligent robots and vehicles [53]. Its detection range is up to 60m, and the

horizontal field of view is 190^{o}. The distance accuracy is 30mm when the range is less than 10m, and 50mm when the range is between 10 to 30m. The angular resolution is 0.25^{o}. This LIDAR sensor is shown in Fig. 2-1 (b).

As another example, the ALASCA XT LIDAR made by IBEO is a multi-planar LIDAR, which splits the laser beam into four vertical layers. The aperture angle is 3.2^{o}. The distance range is up to 200 meters, and the horizontal field of view is 240^{o} [54]. Fig. 2-1 (c) shows the IBEO LIDAR sensor.

Finally, the Velodyne HDL-64E LIDAR is a 3D sensor which is specifically designed for autonomous vehicle navigation [55]. With 360^{o} horizontal by 25^{o} vertical field of view, 0.09^{o} angular resolution, and 10Hz refresh rate, the Velodyne provides surrounding 3-D traffic information with high accuracy (<5cm resolution) and efficiency. The detection range is 100 meter, and the latency is less than 0.05 milliseconds. Figure 2-1 (d) demonstrates the Velodyne and its 3-D range data.

(a) (b) (c) (d)

Figure 2-1. A variety of LIDARs. (a) SICK LMS200 LIDAR, (b) HOKUYO UXM-30LN LIDAR, (c) IBEO ALASCA XT LIDAR, and (d) Velodyne HDL-64E LIDAR.

Range sensors are originally developed for applications by automatic guided vehicles in an indoor environment. Nowadays the performance of range sensors has been improved so that LIDAR can be utilized in outdoor environments. An application example is the DARPA Urban Challenge 2007, in which the autonomous vehicles were capable of driving in traffic and performing complex maneuvers such as acceleration and braking, lane changes, and parking maneuvers [11]. IBEO and SICK LIDAR sensors were used in many of the finalists for tasks such as object detection and localization. The Velodyne sensor was used by the five out of six of the finishing teams.

2.2 LIDAR and Computer Vision Calibration

Sensor fusion systems are commonly used to combine the sensory data from disparate sources, so that the results will be more accurate and complete in comparison to the output of a single sensor. This is common for autonomous vehicles. An example is BOSS, the winner of the 2007 DARPA Urban Grand Challenge, which was equipped with GPS, long and short-range LIDAR sensors, as well as stereo cameras [14].

In order to effectively extract and integrate 3D information from both computer vision and LIDAR sensors, the relative position and orientation between these two sensor modalities need to be obtained. The relative geometric transformation can be solved through a calibration process [3, 4]. Several approaches have been defined and utilized for LIDAR and computer vision calibration. These techniques can be roughly classified

into three categories: visible beam LIDAR-based calibration, Three Dimensional (3D) LIDAR-based calibration, and Two Dimensional (2D) LIDAR-based calibration. The calibration algorithms will be discussed in the following sections.

2.2.1 Visible Beam Calibration

Visible beam calibration is performed by observing the LIDAR beams or reflection points using a camera. The calibration system usually consists of an active LIDAR and some infrared or near-infrared cameras. The LIDAR system typically projects stripes with a known frequency, while these stripes are visible to the camera [15-17]. For example, the LIDAR beams used in [16] are captured by a 955 frame per second (fps) high-speed camera. The color image of LIDAR beams is generated by letting the vision output go through a beam splitter. This system is commonly used to assist surgery.

The visible beam calibration method requires a high-cost infrared camera, which should be sensitive to the spectral emission band of the LIDAR. Therefore, this method is not suitable for low-cost sensor fusion systems.

2.2.2 Three Dimensional LIDAR Based Calibration

This technique calibrates the computer vision system with a 3D LIDAR sensor. Various features are captured by both the camera and the LIDAR. These features are in the form of planes, corners, or edges of a specific calibration object. An elaborate setup is required. Moreover, dense LIDAR beams in both the vertical and horizontal directions are

necessary for the calibration.

The 3D calibration algorithm presented in [19] uses checkerboard, which is commonly utilized in camera calibration. The coefficients of the checkerboard plane are first calculated by LIDAR, then the coefficients are again computed by a camera in computer vision coordinates. A two stage optimization procedure is implemented to minimize the distance between the calculated results and the measurement output.

When the features are edges or corners, the accuracy of the calibration method depends on the accuracy by which features are localized [20]. When the features are planes, the LIDAR beams must be sufficiently dense [19]. Therefore, these methods cannot easily be applied to single planar or sparse multi-planar LIDAR systems.

2.2.3 Two-Dimensional Planar-Based Calibration

This approach works for the calibration of the camera with a 2D LIDAR system. The calibration system proposed in [18] consists of a gray-level CCD camera and a LIDAR. The orientations of the camera and the LIDAR have been calibrated so their coordinates are parallel to each other, i.e., the rotation matrix is known to be an identity matrix. A "V" shaped pattern is designed to obtain the translation between these two sensors, as is shown in Fig. 2-2. The calibration pattern is composed of two zones: a white zone and a black zone. The LIDAR sensor detects the "V" shape, while the camera identifies the line that separates the white zone and the black zone. The calibration procedure is

implemented in two steps: LIDAR detects the "V" shape and finds the vertex, and camera detects the intersection line which cuts the pattern into two parts.

Another calibration approach is proposed in [21] using checkerboard for calibration. This method is based on observing a plane of an object and solving distance constraints from the camera and LIDAR systems. The solution is first of all obtained by minimizing an algebraic error from the distance constraints, then a nonlinear refinement is used to minimize a re-projection error. This approach works only for a single planar LIDAR.

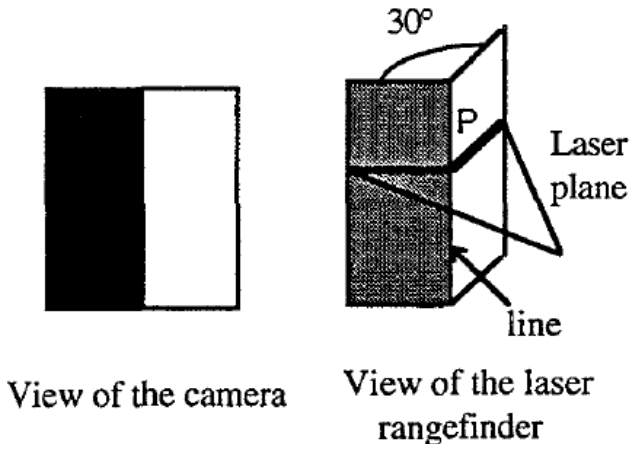


Figure 2-2. The "V" shaped calibration pattern used in 2D LIDAR and computer vision sensor calibration.

To date, there does not exist any convenient calibration method for multi-planar 'invisible-beam' LIDAR and computer vision systems.

2.3 Sensor Fusion Based Vehicle Detection and Tracking

Computer vision is generally used in current mobile platform based object detection and tracking systems, either separately or along with LIDAR sensor [57]. Most of the

current platforms apply a simple segmentation such as background subtraction or temporal difference to detect objects [58]. However, these approaches suffer with the fast background changes due to camera motion. A trainable object detection method is proposed in [59] based on a wavelet template that defines the shape of an object in terms of a subset of the wavelet coefficients of the image. However, the application of vision sensors in vehicle navigation is far from sufficient: clustering, illumination, occlusion, among many other factors, affect the overall performance. Fusion of camera and active sensors such as LIDAR or RADAR is been investigated in the context of on-board vehicle detection and classification.

A LIDAR sensor and a monocular camera based detection and classification system is proposed in [56]. The detection is implemented in the LIDAR space, and the object classification works both in LIDAR space (using a Gaussian Mixture Model classifier) and in computer vision system (using an AdaBoost classifier). A Bayesian decision rule is proposed to combine the results from both classifiers, and thus a more reliable classification is achieved. However, this approach uses the information in LIDAR space for vehicle and pedestrian detection, while the computer vision images are disregarded during the detection process.

Another integration structure is proposed in [60], in which a LIDAR is integrated with a far infrared camera for pedestrian detection. LIDAR data processing and infrared

image processing are performed separately. A LIDAR-based shape extraction method is used to select Regions of Interest (ROIs). The state of the vehicle (position and orientation) w.r.t. the camera sensor is also considered to obtain a global coordinate based pedestrian detection. This system combines a straight forward with a backward loop methodology. Kalman filtering is used as the data fusion algorithm. In this approach, the LIDAR sensor is used alone for initialization, i.e., for ROI generation. It is not integrated with camera in the following pedestrian classification and tracking processes.

A similar technique is presented in [61] for pedestrian detection. It makes use of RADAR and steering sensor to generate the pedestrian position hypotheses. The examination of the hypotheses is implemented by camera with a pedestrian model. The classification is performed using a shape model for either the monocular camera vision or the infrared spectrum images. This method achieves a higher detection accuracy compared to a sole image processing solution. The disadvantage of this method is that one sensor is used in each step: LIDAR for position hypotheses generation, and camera for verification. The two sensors are not integrated as a tightly-coupled sensor fusion system.

The integration of range RADAR, thermopile, and steering angle sensors is proposed in [62]. In this system, range RADAR provides distance from the sensor to pedestrians; thermopile converts thermal energy into electrical energy, which is used to measure

human body temperature; and steering angle is measured by the inertial sensor. Two short range RADAR sensors are mounted in the front bumper of the test vehicle for observing and tracking multiple targets. Spatial distributed thermopile sensors are used to classify the target. RADAR and thermopile are fused using a statistical model. In this case, the target detection and classification can be done either at a lower level (ROIs generated by RADAR and thermopile) or at a higher level (ROIs generated by all the sensors). One contribution of the approach is that instead of using computer vision sensor, in this system the thermopile is utilized for pedestrian detection.

The review of the SAVE-U project [63] compares RADAR-camera and LIDAR-camera sensor fusion systems. It states that the RADAR-camera system becomes unreliable at 10-15m when working in real scenes due to reflection from the surrounding objects. LIDAR, on the other hand, is capable of detecting the targets while providing accurate distance estimates. However, LIDAR is affected by weather conditions, which is not the case for RADAR.

To summarize, LIDAR and computer sensor fusion has been widely used in vehicle and pedestrian detection. The detection is commonly implemented in two steps: hypotheses generation, followed by verification. In current systems, a single sensor is used for one step: LIDAR sensor is utilized to generate position hypotheses, and computer vision sensor is used for verification. Thus, the output of LIDAR sensor and

camera are not tightly integrated in object detection.

2.4 Vehicle Localization Techniques

In driver assistance applications, reliable vehicle-based lane-level position determination techniques are becoming a critical need. The next generation vehicle positioning system requires the fusion a variety of sensors, including GPS, inertial measurement sensors, encoders, and feature-based sensors such as computer vision, RADAR, or LIDAR. The enhanced vehicle positioning capability enables applications such as intersection collision avoidance, road departure warning, and automated vehicle control [40].

There has been increased interest in the sensor aided localization, primarily spawned from recent autonomous vehicle challenges [40]. Simultaneous Localization and Mapping (SLAM) is such a technique using camera or LIDAR to build up a map within an unknown environment (without *a priori* knowledge) or to update a map while at the same time tracking the current location [39]. In this subsection, we discuss several LIDAR based lane-level positioning techniques in which the map information is *a priori* knowledge. The localization approaches can be grouped into several categories w.r.t. the sensors: LIDAR-based, LIDAR and INS based, LIDAR, INS and odometer based, LIDAR and GPS based, and the LIDAR, GPS and INS based localization techniques.

2.4.1 LIDAR-Based Positioning

Vehicle equipped with a LIDAR sensor can locate its own position w.r.t. the landmarks [43, 48, 51]. A very simple navigation framework is proposed in [48], where the map is considered as *a priori* information. An Extended Kalman Filter (EKF) is utilized in [51]. Beacons with reflective tapes are used in [43], where the locations of the beacons are considered as *a priori* information. Least-square minimization methods are applied to calculate the location.

A LIDAR based outdoor navigation system is proposed in [49], in which SLAM is implemented with beacons and natural features. An inverse covariance filter is used for navigation. Outdoor navigation tests using beacons at known locations and SLAM with artificial beacons experiments show that most of the errors are within the 95% confidence bounds.

The LIDAR based positioning techniques work well in indoor environment. However, the overall performance of the LIDAR sensor in outdoor environment may be affected by many factors, such as low target reflection, weather, or small size target such as light poles. Positioning errors will be caused if LIDAR fails to capture the features.

2.4.2 LIDAR and INS Based Positioning

An autonomous navigation solution for urban environment (indoor and outdoor) is proposed in [41, 46], in which a LIDAR sensor and INS are integrated. The line features

detected by a 2D LIDAR is used to determine the location. Line matching is carried out from inertial sensor data. Inertial error calibration is carried out by LIDAR images. The tilt compensation in LIDAR allows for an extension of a 2D case into a partial 3D case by removing the assumption that the scan plane of the LIDAR is parallel to the ground [41]. Estimation of the relative frame position and heading in Earth-Centered-Earth-Fixed (ECEF) frame or the East-North-Up frame allows for the transformation from the sensor coordinated to an absolute coordinate. The experiment shows that in the indoor applications, the relative position error is 1.1% of the distance travelled, where the total distance is 9m. The outdoor test illustrates that the displacement error is 0.8% of the distance travelled. In this approach, the INS is used for LIDAR tilt estimation and landmark identification, while the positioning algorithm depends on a single LIDAR sensor. Therefore, it is not a ‘tightly-coupled’ integration system.

INS and a multiple Airborne Laser Scanner (ALS) are integrated in [44]. A dual laser scanner terrain-referenced dead-reckoning algorithm is proposed in this paper to determine the location. The position drift error has been tested to be approximately 1 meter/minute. This approach requires an Airborne LIDAR to provide 3D bird-view of the terrain, which is a computationally expensive thus prohibiting current positioning algorithm to be applied in vehicle positioning with rapidly changing environment.

2.4.3 LIDAR, INS and Odometer Based Positioning

Inertial sensors, odometer and a LIDAR sensor are utilized in [50] for underground vehicle navigation. A guidance system for an autonomous load, haul and dump truck (LHD) is designed and presented in this paper. In this sensor fusion system, the output of LIDAR is the bearing from the sensor to the target. The sensors are synchronized by a time stamp from a master computer which sends out time signals at a rate of 50Hz. The data from each sensor are recorded for post-processing and individually time stamped. A INS-based dead reckoning system is fused with a SICK LIDAR which provides landmark positions. On the other hand, the odometer based dead reckoning is used to record the distance. These two systems operate in parallel, improving the robustness by providing redundancy. The robustness is also improved by using a variety of sensors, as each sensor has unique failure cases. The navigation is implemented by using discrete EKF. The position error is less than 0.14m during a 140 seconds test. The experiment results show this approach makes the navigation systems for large heavy industrial vehicles much more reliable and robust in harsh uneven terrain [50].

2.4.4 LIDAR and GPS Based Positioning

GPS and LIDAR are combined in [45, 47] for navigation in outdoor environments. In urban areas, if there exists a building wall that blocks the GPS signals, the wall will build a line-feature which can be detected by the LIDAR. On the other hand, for open streets

the GPS signals are not blocked so it provides reliable position information. Thus the GPS and LIDAR can be integrated in either urban or suburban areas.

The integration of GPS and two laser scanners is performed using an EKF in [45]. In this paper, the vehicle's location is determined without *a priori* knowledge of the landmark information. The position error is less than 15cm in the 'forest scenario' when the GPS has a 35m outage, and less than 10cm in the 'urban canyon scenario' when GPS has a 20m outage. In an experiment conducted in an alley in Chicago, the absolute positioning error does not exceed 1.5m over 70m of GPS outage. However, this method has poor performance in urban area where the GPS losses satellite signals.

2.4.5 LIDAR, GPS and INS Based Positioning

GPS, LIDAR and inertial sensors are used for navigation in urban environments, as described in [42]. Compared with the work in [45], in this integration system an INS sensor is utilized for improved solution robustness, i.e., for robust feature association between the LIDAR scan output and the GPS data. Line features are extracted by LIDAR for position determination when the GPS signal is not available. In open areas the GPS is used for navigation. When sufficient GPS or LIDAR measurements are not available, the INS can be utilized. The algorithm to integrate LIDAR and INS data is the same as in [41]. Particularly, the inertial sensor data are applied to identify landmarks in the LIDAR output, and to adjust a 2D scan plan for tilting of the LIDAR platform. It also works in

the cases when GPS losses satellite signals, or insufficient LIDAR measurements are available. The Kalman filter is used in tracking.

This paper uses some urban data to demonstrate the performance of the integration system. Two outdoor tests have been carried out. In the first test, the trajectory is 300m, and the standard deviation of East and North residual components are 30cm and 20cm, respectively. The second trajectory is 240m, and the error is less than 30cm.

To summarize, sensor fusion techniques are commonly used in current vehicle positioning systems. The sensors utilized in positioning include LIDAR, camera, GPS, inertial sensors, and odometer. The performance of LIDAR-based positioning method is sufficient for indoor applications. However, in outdoor environment the LIDAR sensor should be integrated with the other sensors. Camera is another sensor commonly used for navigation. On the other hand, the computation is time consuming since in a large environment there are thousands of features for computer vision sensor to capture. GPS is an efficient sensor to provide an accurate estimation of the position. However, in urban areas it may has poor performance with few satellite connections. The integration of LIDAR and inertial sensors is a reliable solution for outdoor navigation.

2.5 Vehicle Tracking and Its Application in Freeway Traffic Surveillance Systems

Traffic management systems observe the road environment and assist the

transportation administration department in obtaining real-time traffic information. One important capability that is critical for many surveillance applications is the ability to detect and track vehicles from a video stream. This topic has been addressed in video based monitoring systems [22]. The vehicle trajectory data are collected over a length of roadway and time, rather than at a single point and a fixed time. The information from vehicle trajectories is utilized in traffic analysis, driver behavior learning, incident detection, path planning, and emission estimation [23].

The computer vision-based vehicle tracking techniques can be grouped into four main categories: tracking using discrete features, tracking with contours, region-based tracking, and combined tracking.

2.5.1 Tracking Using Discrete Features

The first category is the tracker using only discrete features, i.e., points, collections of edges, and lines. A feature is defined as a local representation of the target. Feature based tracking is commonly implemented by detecting and tracking individual features, and then grouping the features in one foreground blob [25].

Image features are commonly referred to as “interest points”, i.e., the points whose surrounding regions change in a two-dimensional manner [24]. The feature points are selected by checking the intensity change using a local autocorrelation function. Kanade-Lucas-Tomasi (KLT) tracking is used which proposes a density matrix that

captures the local intensity structure [26]. Once the feature point has been selected, the tracking is implemented by solving the image displacement equation.

Tracking using features that describe a larger region have recently been a very popular technique. Color histograms or color distribution models are commonly used as regional descriptors. Color histograms present the probability of each color occurs within an image region. One application is the mean-shift tracking algorithm, which uses a color probability model since the color histogram does not vary much to target rotation of scale variation [27].

Pixels can also be grouped into higher level structures, such as grouping edge pixels into a line, or comparing a set of corner features against the estimated vehicle positions. A commonly used method is to compare the features with a 3D model. The edges are first grouped to line-based features, which are parameterized using the position and orientation. Then the Mahalanobis distance is calculate to compare a line feature with a particular projected model [28, 29].

Discrete features are reliable for vehicle tracking with free flow traffic. When the vehicles are not far away to each other, the features will be visible and can be tracked during several consecutive frames. However, in high density traffic situation when the features are partially occluded, tracking will be problematic.

2.5.2 Tracking Using Contours

A curve that presents the outline of the vehicle is used in contour tracking. Unlike the other tracking methods that represent the vehicle as a rectangle or an ellipse, the contour tracking ideally provide very precise localization of the target and its boundary.

A commonly used contour-based tracker is the ‘snakes’ tracking, which identifies the contour that minimize the shape energy. The energy represents the goodness of a contour. Two assumptions are made in this tracking approach. One is that the contour does not contain protruding or valleys. Furthermore, it is assumed that the length of contour doesn’t change significantly from frame-to-frame [30].

The parameterization process in the snake method is too cumbersome for most tracking systems. Therefore, a regional information based contour tracker is proposed to improve the performance [31]. The so-called ‘depth-adapting algorithm’ proposes a new energy definition. If the local contour perturbation leads to a positive energy change, the contour will continue to move inward or outward. Another ‘adaptive fusion’ algorithm is also proposed in [31], which is an extension of the depth-adapting algorithm to include more global information.

Most of the video sequences used in contour-based tracking dealt with fairly clear, high-contrast edges in minimal cluster, which makes the contour extraction much easier. Therefore, the contour cannot be used in vehicle tracking with high density traffic.

2.5.3 Region-Based Tracking

Region-based Tracking method tracks an area that represents the target. This tracking approach can be classified into several categories: blob tracker, kernel and histogram tracker, and the pixel-wise template tracker.

The blob tracking methods are commonly implemented in the following steps [32]:

- Extract the background from the video, which is a stationary image. The foreground is detected by subtracting the background from the image.
- Detect the changes in foreground from frame to frame. The changes are considered to be the moving objects.
- Associate the currently detected objects with targets detected from previous frames.

Blob tracker is easy to implement with high efficiency. However, the accuracy relies on having an effective background subtraction [33].

Kernel histogram methods use some combination of a weighted kernel and a histogram to track the target frame-to-frame. The most prominent kernel histogram tracking method is the mean shift tracking [27]. The basic component of a mean shift tracking method is the color histogram, which counts the number of pixels with each color and gives the color probabilities. Then the histograms are matched using the Bhattacharyya coefficient [79]. Finally, the best candidate match in the current frame is considered to be the

location of the target.

Pixel-wise template tracking uses some feature descriptors at each pixel for vehicle tracking. The most commonly used feature is the intensity. A template matching tracker performs in the following steps [34, 35]:

- Initialize the template of the target in the first frame.
- Predict where the target will appear in the next frame. It is implemented by a Kalman filter or particle filter.
- Match the template to the image regions which center on the estimated position with a surrounding neighborhood search region. The commonly employed matching methods include standard Sum of Squared Differences (SSD), or the cross-correlation.
- The location that provides the highest matching probability is considered to be the current target.

Region-based tracking is very effective for vehicle tracking when the targets are far away from one another. This approach begins to break down when the vehicles are close to each other, i.e., when the occlusion happens.

2.5.4 Combined Tracking

Nowadays a lot of research has been carried out to combine various vehicle tracking techniques. The combined trackers commonly use a probability model with multiple

target representations.

A tracking model is proposed in [36] to combine a variety of features with a probabilistic framework using particle filter. The features include curves and texture regions. The states in the particle filter consist of the target position in 2D, and its six affine motion parameters. The experiment results in [36] illustrate that when a single feature fails in the tracking, the proposed multi-feature tracking still succeeds.

Another tracking method uses three feature modalities: homogeneous regions, textured regions, and snakes [37]. Likelihood functions have been defined for each of the trackers, respectively. The homogeneous regions-based likelihood function determines the optimal location, size and orientation of the target. Then a Sum-of-Squared-Differences-based matching score is used to define the probability model of the texture tracker. The third tracking modality uses intensity-based snake tracker. The likelihood function is defined based on the differences between the predicted contour points and the closest measurement results. In [37] a joint likelihood filter is proposed which combines the three likelihood functions. Results are shown that the combination of the trackers outperforms the single-modality tracking methods.

However, another work in [38] proposed that the blindly fusing of the results from different sensors or sources is sub-optimal. The combination is generally implemented in a particle filter, which treats each modality evenly and generates an overall likelihood

function. It is proposed in [38] that separate target trackers should be generated using each of the modalities. The different trackers are combined based on the confidence of the obtained tracks.

In summary, in this section we have considered vehicle trackers that make use of discrete features, contours, and regions, as well as the trackers that combines the probability of each feature. However, none of the reviewed approaches works with significant occlusions. In Chapter 6, we will propose a tracker that integrates feature tracking and color histogram probability model to solve the occlusion problem.

2.6 Recapitulation

To summarize, this chapter has presented a number of techniques that are commonly used to estimate the surrounding vehicle's state, or to predict the test vehicle's position. We began with a discussion of LIDAR sensors. Subsequently, the calibration of computer vision and LIDAR sensor has been reviewed. Additionally, the topics of sensor fusion-based vehicle detection approaches were discussed. Vehicle positioning techniques were also presented in this chapter. Finally, we discussed the vehicle tracking methods using stationary computer vision sensor. In the next chapter we will present our solutions to the "*where am I*?" and "*where are they*?" problems in more details.

Chapter 3

A Novel Multi-Planar LIDAR and Computer Vision Calibration Procedure Using 2D Patterns

In this chapter, we describe a novel calibration approach for a LIDAR and computer vision sensor fusion system. This system consists of a camera with a multi-planar LIDAR, where the laser beams are invisible to the camera. This calibration method also works for computer vision and 3D LIDAR systems.

Although several calibration methods have been developed to obtain the geometric relationship between two sensors, few of them have provided a complete sensitivity analysis of the calibration procedure (see background in Chapter 2). As part of the calibration method proposed in this dissertation, we also address the effect of LIDAR noise level as well as the total number of poses on the calibration accuracy.

This chapter is organized as follows: Section 3.1 gives the setup using planar planes and defines the calibration constraint. Section 3.2 describes in detail how to solve this constraint in two steps. Both a closed-form solution and a non-linear minimization solution based on maximum likelihood criterion are introduced. Experimental results with

different poses are provided in Section 3.3. Finally, a brief summary is given in Section 3.4.

3.1 Sensor Alignment

The setup for a multi-planar LIDAR and camera calibration is described here.

3.1.1 Sensor Configuration

In the calibration system, an instrumented vehicle is equipped with two IBEO ALASCA XT LIDAR sensors which are mounted on the front bumper. The LIDAR sensor scans with four separate planes. The distance range is up to 200 meters, the horizontal field of view angle of a single LIDAR is 240^{o}, and the total vertical field of view for the four planes is 3.2^{o}. The camera is mounted on the vehicle behind the front windshield, as is shown in Fig. 1-1.

In order to use the measurements from different kinds of sensors at various positions on the vehicle, the measurements should be transformed from their own coordinate into some common coordinate system. In this section, we focus on obtaining the spatial relationship between video and LIDAR sensors. The geometric sensor model is shown in Fig. 3-1.

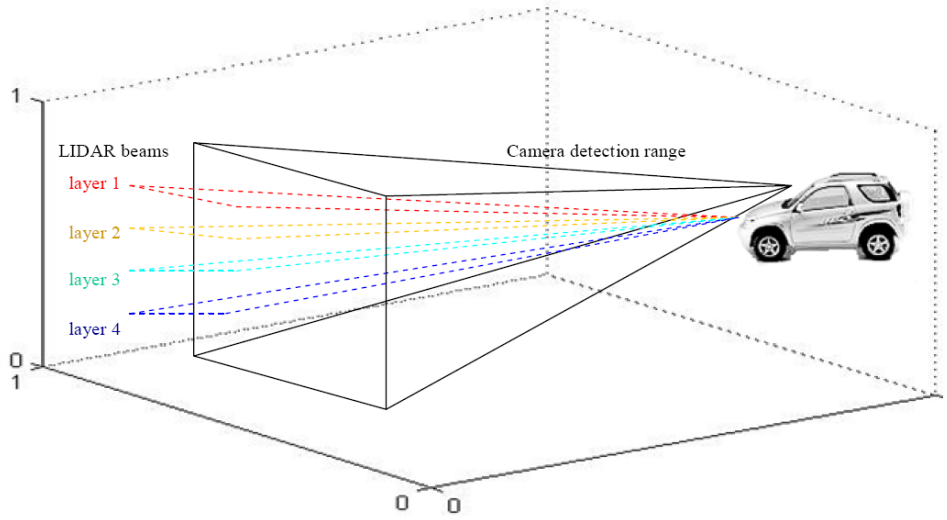


Figure 3-1. Geometric model with the camera and the LIDAR.

3.1.2 Vision, LIDAR and World Coordinate Systems

There are several coordinate systems in the overall system: the camera coordinates, the LIDAR coordinates, and the world coordinate systems.

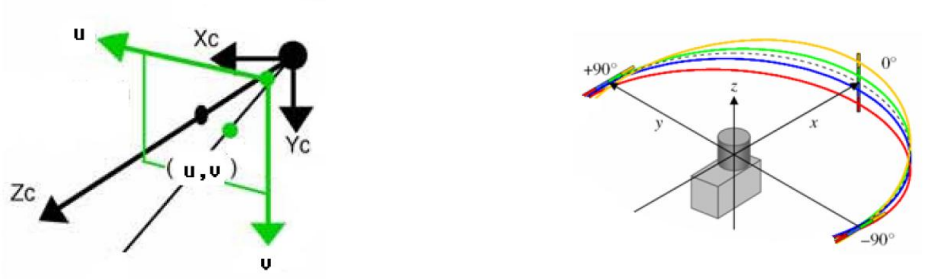
A camera can be represented by the standard pinhole model. One 3D point in the camera coordinate denoted by $\mathbf{P}_c = [X_c\ Y_c\ Z_c]^T$ is projected to a pixel $\mathbf{p} = [u\ v]^T$ in the image coordinate. The pinhole model is given as [64]:

$$s\mathbf{p} \sim \mathbf{A}\begin{bmatrix}\mathbf{R} & \mathbf{t}\end{bmatrix}\mathbf{P}_c \quad \text{with } \mathbf{A} = \begin{bmatrix} \alpha & \gamma & u_0 \\ 0 & \beta & v_0 \\ 0 & 0 & 1 \end{bmatrix} \tag{3-1}$$

where s is an arbitrary scale factor. $\mathbf{A}$ is the camera intrinsic matrix defined by the coordinates of the principal point (u_0, v_0), scale factors α and β in image u and v

axes, and skewness of the two image axes γ. $(\mathbf{R},\ \mathbf{t})$ are called extrinsic parameters. The 3×3 orthonormal rotation matrix $\mathbf{R}$ represents the orientation of the world coordinates to the camera coordinate system. The translation matrix $\mathbf{t}$ is a 3-vector representing the origin of the world coordinates in the camera's frame of reference. In the real world, the lens of the camera may also have image distortion coefficients, which include radial and tangential distortions and are usually stored in a 5-vector [65]. In our experiment, the lens is assumed to have no significant distortion, or the distortion has already been eliminated.

The LIDAR sensor provides distance and direction of each scan point in LIDAR coordinates. Distances and directions can be converted into a 3D point denoted by $\mathbf{P}_l = [X_l\ Y_l\ Z_l]^T$ [54]. The origin of the LIDAR coordinates is the equipment itself. X, Y and Z axes are defined as forward, leftward and upward from the equipment, respectively. The camera and LIDAR reference systems are shown in Fig. 3-2.



(a) (b)

Figure 3-2. Two coordinate systems. (a) The camera coordinates and screen coordinate systems, and (b) the LIDAR coordinate system.

In addition to the camera and LIDAR reference systems, another coordinate system is used in calibration procedure: the world frame of reference. In the calibration process, a checkerboard is placed in front of the sensors. The first grid on the upper-left corner of this board is defined to be the origin of the world coordinates [65].

Suppose we have a fixed point $\mathbf{P}$ in space, which is denoted as $\mathbf{P}_c = [X_c\ Y_c\ Z_c]^T$ in the camera coordinates, and $\mathbf{P}_l = [X_l\ Y_l\ Z_l]^T$ in the LIDAR coordinates. The transformation from LIDAR coordinate to camera coordinate is given as:

$$\mathbf{P}_c = \mathbf{R}_l^c \mathbf{P}_l + \mathbf{t}_l^c \tag{3-2}$$

where $(\mathbf{R}_l^c,\ \mathbf{t}_l^c)$ are the rotation and translation parameters which relate LIDAR coordinate system to the camera coordinate system.

The purpose of our calibration work is to solve Eqn. (3-2) and obtain coefficients $(\mathbf{R}_l^c,\ \mathbf{t}_l^c)$, so that any given point in the LIDAR reference system can be transformed to the camera coordinates.

3.1.3 Basic Geometric Interpretation

A checkerboard visible to both sensors is used for calibration. In the following sections, the planar surface defined by the checkerboard is called the *checkerboard plane*. Without loss of generality, we assume the checkerboard plane is on $Z = 0$ in the world

coordinates. Let $\mathbf{r}_i$ denotes the i-th column of the rotation matrix $\mathbf{R}$. Then $\mathbf{r}_3$ is the surface normal vector of the calibration plane in the camera coordinate system [65].

Note the origin of the world coordinate is the upper-left corner of the checkerboard, and the origin of the camera coordinate is the camera itself. The translation vector $\mathbf{t}$ represents relative position of the checkerboard's upper-left corner in the camera's reference systems. Since both $\mathbf{t}$ and $\mathbf{P}_c$ are points on the checkerboard plane denoted in camera coordinates, a vector $\vec{v}$ is defined as $\vec{v} = \mathbf{P}_c - \mathbf{t}$. Note that $\vec{v}$ is a vector on the checkerboard plane, and $\mathbf{r}_3$ is orthogonal to this plane, we have:

$$\mathbf{r}_3 \cdot \vec{v} = 0 \tag{3-3}$$

where $\cdot$ denotes the inner product. The geometric interpretation for Eqn. (3-3) is illustrated in Fig. 3-3.

By substituting Eqn. (3-2) into Eqn. (3-3), we have:

$$\mathbf{r}_3^T(\mathbf{R}_l^c\mathbf{P}_l + \mathbf{t}_l^c - \mathbf{t}) = 0 \tag{3-4}$$

Since point $\mathbf{P}_l$ in LIDAR coordinates is $[X_l\ Y_l\ Z_l]^T$, from Eqn. (3-4):

$$\mathbf{r}_3^T \begin{bmatrix} \mathbf{R}_l^c & \mathbf{t}_l^c - \mathbf{t} \end{bmatrix} \begin{bmatrix} X_l \\ Y_l \\ Z_l \\ 1 \end{bmatrix} = 0 \tag{3-5}$$

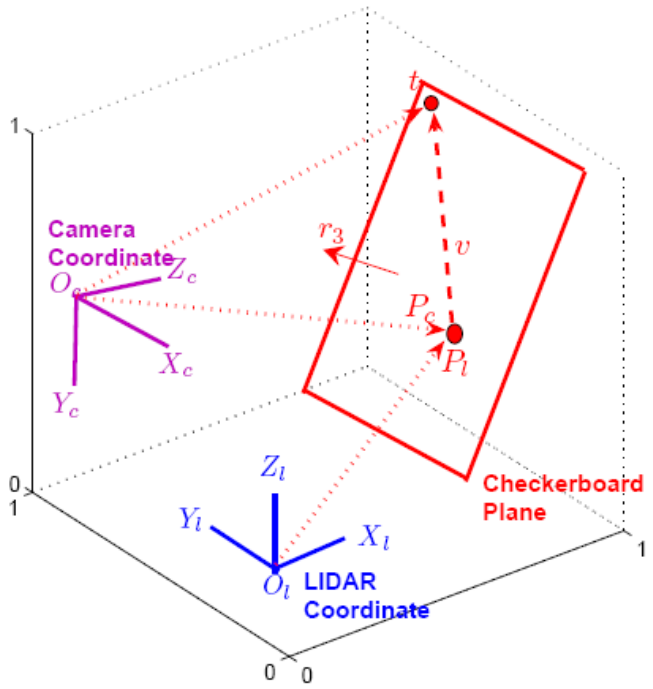


Figure 3-3. Geometric interpretation of the camera coordinates, the LIDAR coordinates, and checkerboard plane.

For each LIDAR point on the checkerboard plane, Eqn. (3-5) explains the geometric relationships and constraints on $(\mathbf{R}_l^c, \mathbf{t}_l^c)$. This is the basic constraints for the calibration from the LIDAR to the vision coordinate system.

3.2 Calibration Solutions

This subsection provides the method to efficiently obtain the calibration coefficients $(\mathbf{R}_l^c, \mathbf{t}_l^c)$. We start with an analytical solution, followed by a nonlinear optimization technique based on the maximum likelihood criterion.

3.2.1 Closed-form Solution

Initially, the camera's intrinsic parameters are calibrated using a standard Camera Calibration Toolbox [65]. For each pose of the checkerboard, there's one set of camera extrinsic parameters $(\mathbf{R}, \mathbf{t})$. Each $(\mathbf{R}, \mathbf{t})$ is determined also using the toolbox, after which $\mathbf{r}_3$ and $\mathbf{t}$ in Eqn. (3-5) are obtained.

For simplicity, let's define $\mathbf{r}_3 = [r_{31} \;\; r_{32} \;\; r_{33}]^T$, $\mathbf{\Delta} = \mathbf{t}_l^c - \mathbf{t} = [\Delta_x \;\; \Delta_y \;\; \Delta_z]^T$, and m_{ij} be the element on the i-th row, j-th column in matrix $\mathbf{R}_l^c$. Suppose for one pose of the checkerboard, there are p LIDAR points on the checkerboard plane, denoted as $\mathbf{P}_{l,1} = [X_{l,1} \;\; Y_{l,1} \;\; Z_{l,1}]^T$, $\mathbf{P}_{l,2} = [X_{l,2} \;\; Y_{l,2} \;\; Z_{l,2}]^T$, ..., $\mathbf{P}_{l,p} = [X_{l,p} \;\; Y_{l,p} \;\; Z_{l,p}]^T$. The geometric interpretation becomes a $\mathbf{Ax} = \mathbf{0}$ problem, where $\mathbf{A}$ is a $N \times 12$ matrix, and $\mathbf{x}$ is a 12-vector to be solved. $\mathbf{A}$ and $\mathbf{x}$ are given in Eqn. (3-6).

$$\mathbf{A} = \begin{bmatrix} r_{31}X_{l,1} & r_{31}Y_{l,1} & r_{31}Z_{l,1} & r_{31} & r_{32}X_{l,1} & r_{32}Y_{l,1} & r_{32}Z_{l,1} & r_{32} & r_{33}X_{l,1} & r_{33}Y_{l,1} & r_{33}Z_{l,1} & r_{33} \\ r_{31}X_{l,2} & r_{31}Y_{l,2} & r_{31}Z_{l,2} & r_{31} & r_{32}X_{l,2} & r_{32}Y_{l,2} & r_{32}Z_{l,2} & r_{32} & r_{33}X_{l,2} & r_{33}Y_{l,2} & r_{33}Z_{l,2} & r_{33} \\ \cdots & & & & & & & & & & & \\ r_{31}X_{l,N} & r_{31}Y_{l,N} & r_{31}Z_{l,N} & r_{31} & r_{32}X_{l,N} & r_{32}Y_{l,N} & r_{32}Z_{l,N} & r_{32} & r_{33}X_{l,N} & r_{33}Y_{l,N} & r_{33}Z_{l,N} & r_{33} \end{bmatrix}$$

$$\mathbf{x} = \begin{bmatrix} m_{11} & m_{12} & m_{13} & \Delta_x & m_{21} & m_{22} & m_{23} & \Delta_y & m_{31} & m_{32} & m_{33} & \Delta_z \end{bmatrix}^T \tag{3-6}$$

By getting the LIDAR points $\mathbf{P}_{l,1}, \mathbf{P}_{l,2}, \cdots, \mathbf{P}_{l,N}$, we can estimate $\mathbf{x}$ using the least square method. In order to avoid the solution $\mathbf{x} = \mathbf{0}$, normalization constraints are proposed. Faugeras and Toscani [66] suggested the constraint $m_{31}^2 + m_{32}^2 + m_{33}^2 = 1$, which

is singularity free. This restriction is proposed from the coincidence that $[m_{31} \;\; m_{32} \;\; m_{33}]$ is the third row of the rotation matrix $\mathbf{R}_l^c$. Thus solving the equation $\mathbf{Ax} = \mathbf{0}$ is transformed into minimizing the norm of $\mathbf{Ax}$, i.e., minimizing $|\mathbf{Ax}|$ with the restriction $m_{31}^2 + m_{32}^2 + m_{33}^2 = 1$.

$|\mathbf{Ax}|$ can be minimized using a Lagrange method [67]. Let $\mathbf{m}_3 = [m_{31} \;\; m_{32} \;\; m_{33}]$, and $\mathbf{m}_9$ be a vector containing the remaining 9 elements in $\mathbf{x}$. The Lagrange equation is written as:

$$L = \mathbf{A}_9\mathbf{m}_9 + \mathbf{A}_3\mathbf{m}_3 + \lambda\left(\mathbf{m}_3^T\mathbf{m}_3 - 1\right) \tag{3-7}$$

where $\mathbf{A}_3$ contains the 9-th to 11-th columns of $\mathbf{A}$, and $\mathbf{A}_9$ contains the remaining 9 columns corresponding to $\mathbf{m}_9$.

The closed-form linear solution is:

$$\begin{aligned} \lambda\mathbf{m}_3 &= \left(\mathbf{A}_3^T\mathbf{A}_3 - \mathbf{A}_3^T\mathbf{A}_9\left(\mathbf{A}_9^T\mathbf{A}_9\right)^{-1}\mathbf{A}_9^T\mathbf{A}_3\right)\mathbf{m}_3 \\ \mathbf{m}_9 &= -\left(\mathbf{A}_9^T\mathbf{A}_9\right)^{-1}\mathbf{A}_9^T\mathbf{A}_3\mathbf{m}_3 \end{aligned} \tag{3-8}$$

It's well known that $\mathbf{m}_3$ is the eigenvector of the symmetric positive definite matrix $\mathbf{A}_3^T\mathbf{A}_3 - \mathbf{A}_3^T\mathbf{A}_9\left(\mathbf{A}_9^T\mathbf{A}_9\right)^{-1}\mathbf{A}_9^T\mathbf{A}_3$ associated with the smallest eigenvalue. $\mathbf{m}_9$ is obtained after $\mathbf{m}_3$. Once $\mathbf{m}_3$ and $\mathbf{m}_9$ are known, the rotation and translation matrix $(\mathbf{R}_l^c, \mathbf{t}_l^c)$ is available.

Because of data noise, the rotation matrix $\mathbf{R}_l^c$ may not in general satisfy $(\mathbf{R}_l^c)^T\mathbf{R}_l^c = \mathbf{I}$. One solution is to obtain $\hat{\mathbf{R}}_l^c$, which is the best approximation of given $\mathbf{R}_l^c$. This $\hat{\mathbf{R}}_l^c$ has the smallest Frobenius norm of the difference $\hat{\mathbf{R}}_l^c - \mathbf{R}_l^c$, subject to $(\hat{\mathbf{R}}_l^c)^T\hat{\mathbf{R}}_l^c = \mathbf{I}$ [64].

3.2.2 Maximum Likelihood Estimation

The closed-form solution is obtained by minimizing an algebraic distance $|\mathbf{Ax}|$, which is not physically meaningful. In this subsection, we refine this problem through maximum likelihood function using multi-pose checkerboard planes, which is more meaningful.

In this camera calibration approach, differences of image points and the corresponding projection of the ground truth point in an image are minimized [64]. This method is also valid for visible-beam LIDAR calibration [21]. In our test, the Euclidean distances from camera to the checkerboard are checked. Note that Eqn. (3-4) can be written as:

$$\mathbf{r}_3^T\left(\mathbf{R}_l^c\mathbf{P}_l + \mathbf{t}_l^c\right) = \mathbf{r}_3^T\mathbf{t} \tag{3-9}$$

where both $\mathbf{R}_l^c\mathbf{P}_l + \mathbf{t}_l^c$ and $\mathbf{t}$ are points on the calibration plane surface, and $\mathbf{r}_3$ is the normal vector to this surface. Therefore, both the left and right sides of Eqn. (3-9) are the distance between the checkerboard plane and the origin of the camera reference system.

Suppose we have n poses of the calibration plane. For the i-th pose, there is a set of $(\mathbf{r}_3, \mathbf{t})$ denoted as $(\mathbf{r}_3^i, \mathbf{t}^i)$. We assume the LIDAR points are corrupted by Gaussian distributed noise. Then the maximum likelihood function can be defined by minimizing the sum of the difference between $\mathbf{r}_3^T(\mathbf{R}_l^c\mathbf{P}_l + \mathbf{t}_l^c)$ and $\mathbf{r}_3^T\mathbf{t}$ for all the LIDAR points. Suppose for the i-th plane, there are p_i LIDAR points. Then the solution satisfies:

$$\arg\min_{\mathbf{R}_l^c, \mathbf{t}_l^c} \sum_{i=1}^{n} \frac{1}{p_i} \sum_{j=1}^{p_i} \left((\mathbf{r}_3^i)^T \left(\mathbf{R}_l^c \mathbf{P}_{l,j}^i + \mathbf{t}_l^c \right) - (\mathbf{r}_3^i)^T \mathbf{t}_i \right)^2 \tag{3-10}$$

where $\mathbf{R}_l^c\mathbf{P}_{l,j}^i + \mathbf{t}_l^c$ is the coordinate of $\mathbf{P}_{l,j}^i$ in camera reference system, according to Eqn. (3-2).

By using the Rodriguez formula [66], the rotation matrix $\mathbf{R}_l^c$ is transformed into a vector, which is parallel to the rotation axis and whose magnitude is equal to the rotation angle. Thus $(\mathbf{R}_l^c, \mathbf{t}_l^c)$ forms a vector. Eqn. (3-10) is solved using the Levenberg-Marquardt algorithm (LMA) [68, 69], which provides numerical solutions to the problem of minimizing nonlinear functions. LMA requires an initial guess for the parameters to be estimated. In our algorithm,$(\mathbf{R}_l^c, \mathbf{t}_l^c)$ in the closed form is used as this initial state. For each pose, a set of $(\mathbf{R}_l^c, \mathbf{t}_l^c)$ is obtained. The weighted average is used as an initial guess, where the scalar weight is normalized as a relative contribution of each checkerboard

pose. Then LMA gives a robust solution even if the initial state starts far off the final solution.

3.2.3 Summary of Calibration Procedure

The calibration procedure proposed in this approach can be summarized as:

1) Place the checkerboard in view of the camera and LIDAR systems. Make sure that the plane is within the detection zone of both sensors. The different poses of checkerboard cannot be parallel to each other, otherwise the parallel poses do not provide enough constraints on $\mathbf{R}_l^c$.

2) Take a few measurements (images) of the checkerboard under different orientations. For each orientation, read the LIDAR points on this plane from the output.

3) Estimate the coefficients using the closed-form solution given in Section 3.2.1.

4) Refine all the coefficients using the maximum likelihood estimation in Section 3.2.2.

3.3 Experimental Results

The proposed vision-LIDAR calibration algorithm has been tested on both a computer simulation platform and with real world data.

3.3.1 Computer Simulations

The camera is assumed to have been calibrated. It is simulated to have the following property: $\alpha = 1200$, $\beta = 1000$, and the skewness coefficient $\gamma = 0$. The principal point is (320, 240), and the image resolution is 640×480. The calibration checkerboard consists of 10×10 grids. The size of each square grid is $5\text{cm} \times 5\text{cm}$. The position and orientation of the LIDAR relative to the camera have also been defined. The LIDAR's position in camera coordinates is $\mathbf{t}_l^c = [10 \;\; 150 \;\; 100]^T$ centimeters, and the rotation matrix $\mathbf{R}_l^c$ is parameterized by a 3-vector rotation vector $[-85^o \;\; 10^o \;\; -80^o]^T$.

The LIDAR points are calculated based on the location of the camera and relative position and orientation of the checkerboard. Gaussian noise is added to the points.

A. Performance w.r.t. the noise level

The checkerboard plane is placed in front of the camera and LIDAR. Three poses are used here. All of them have $\mathbf{t} = [-20 \;\; -20 \;\; -550]^T$. The three rotation matrix are defined by the rotation vectors as $\mathbf{r}_1 = [170^o \;\; -5^o \;\; 85^o]^T$, $\mathbf{r}_2 = [170^o \;\; 15^o \;\; 85^o]^T$, $\mathbf{r}_3 = [170^o \;\; -25^o \;\; 85^o]^T$, respectively. Gaussian noise with zero mean and σ standard deviation (from 1 to 10cm) is added to the LIDAR points. The estimation results are then compared with the ground truth. For each noise level, we carried out 100 independent

random trials. The averaged calibration error is shown in Fig. 3-4, where the calculation results are denoted as $\hat{\mathbf{R}}_l^c$ and $\hat{\mathbf{t}}_l^c$, respectively. As we can see from this figure, the calibration error increases with noise level. For $\delta < 7$cm (which is larger than the normal standard deviation for most LIDAR sensors), the error of norm of $\mathbf{R}_l^c$ is less than 0.01. With three checkerboard poses, the relative translation error is less than 5% when $\sigma < 5$cm.

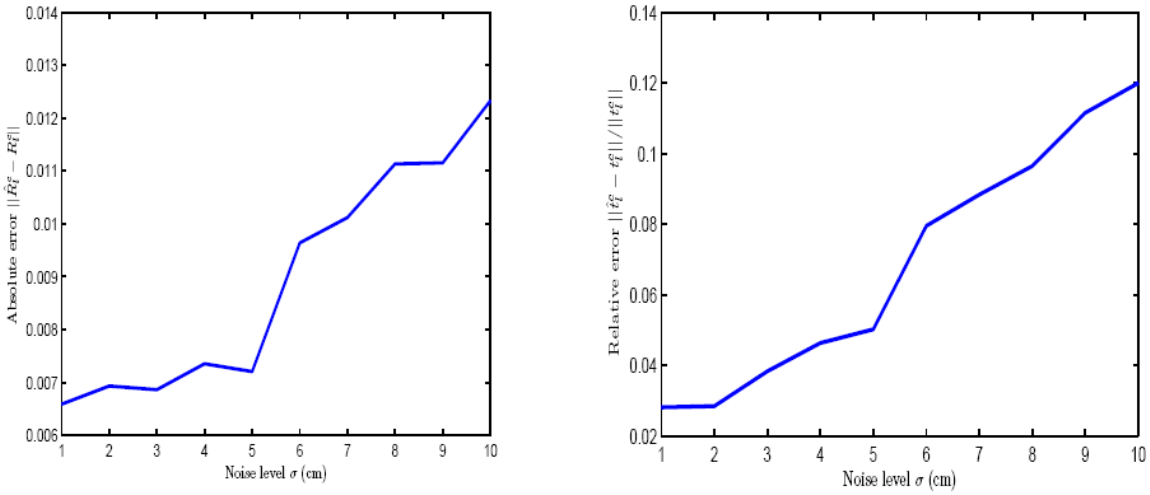


Figure 3-4. Rotation and translation error w.r.t. the noise level.

B. Performance w.r.t. the number of checkerboard positions

The checkerboard is originally setup parallel to the image plane. Then it is rotated by 30^o, where the rotation axis is randomly selected in a uniform sphere. The number of checkerboards used for calibration varies from 4 to 20. Gaussian noise with zero mean and standard deviation $\sigma = 4$cm is added to the LIDAR points. For each number of posi-

tion, 100 trials of independent rotation axis are implemented. The averaged result is illustrated in Fig. 3-5. This figure shows that when the number of checkerboard positions increases, the calibration error decreases.

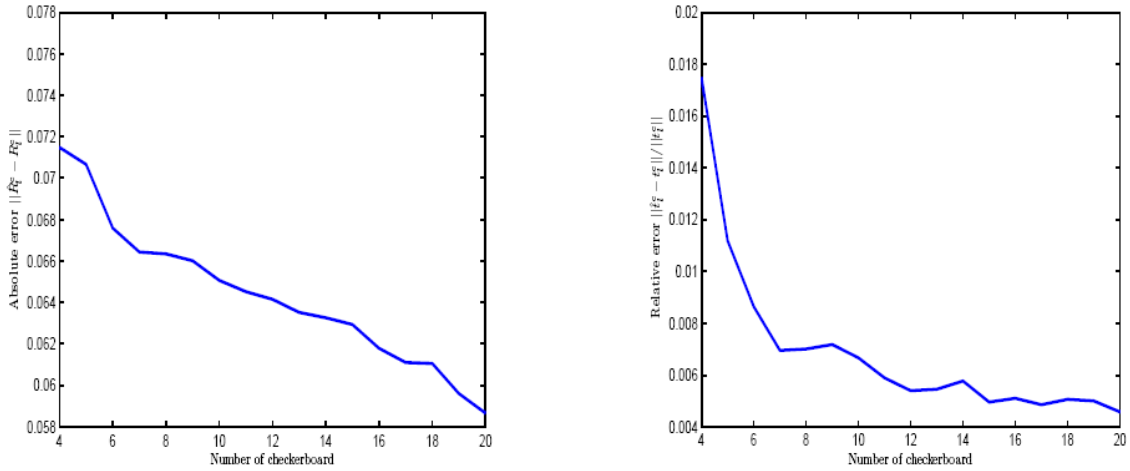


Figure 3-5. Rotation and translation error w.r.t. number of checkerboard positions.

C. Performance w.r.t. the number of checkerboard positions

The checkerboard plane is initially set as parallel to the image plane. It is then rotated around a randomly chosen axis with angle θ. The rotation axis is randomly selected from a uniform sphere. The rotation angle θ varies from 10° to 80°, and 10 checkerboards are used for each θ. Gaussian noise with zero mean and standard deviation $\sigma = 4$cm is added to the LIDAR points. For each rotation angle, 100 trials are repeated and the average error is calculated. The simulation result is shown in Fig. 3-6. The calibration error

decreases when the rotation angle increases. When the rotation angle is too small, the calibration planes are almost parallel to each other, which causes a large error. When the rotation angle is too large, the calibration plane is almost perpendicular to the image plane, which makes the LIDAR measurement less precise.

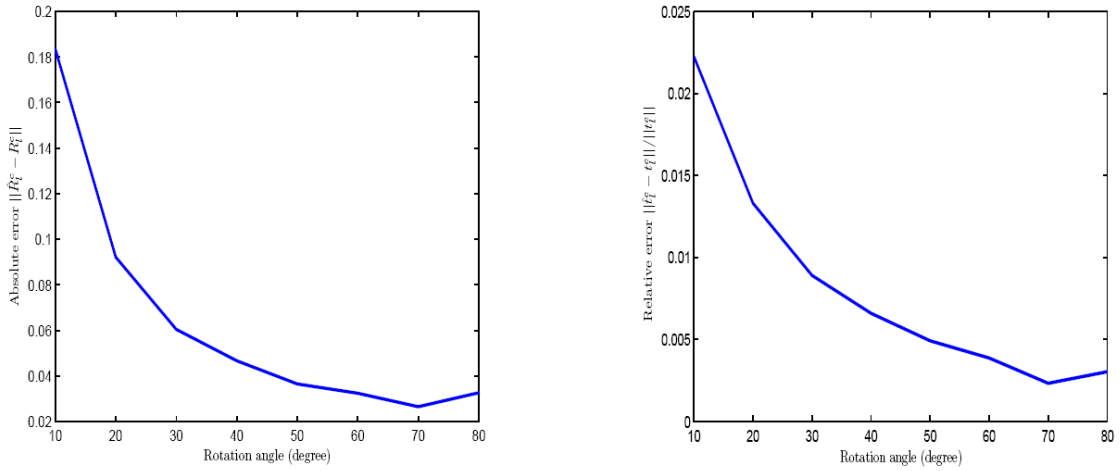


Figure 3-6. Rotation and translation error w.r.t. the orientation of the checkerboard plane.

3.3.2 Real Data Calibration

The calibration method is checked using an IBEO ALASCA XT LIDAR system and a Sony CCD digital camera with a 6mm lens. The image resolution is 640×480. The checkerboard plane consists of a pattern of 16×16 squares, so there are totally 256 grids on the plane. The size of each grid is $2.54\text{cm} \times 2.54\text{cm}$ (1 inch $\times$ 1 inch).

20 images of the plane were taken with different orientations, and the LIDAR points are recorded simultaneously. Two examples of the calibration results are shown in Fig. 3-7, where the LIDAR points are mapped to image reference system using estimated $\mathbf{R}_l^c$ and $\mathbf{t}_l^c$. Although the ground truth of $\mathbf{R}_l^c$ and $\mathbf{t}_l^c$ are not known, Fig. 3-7 shows that the estimation results are pretty reasonable.

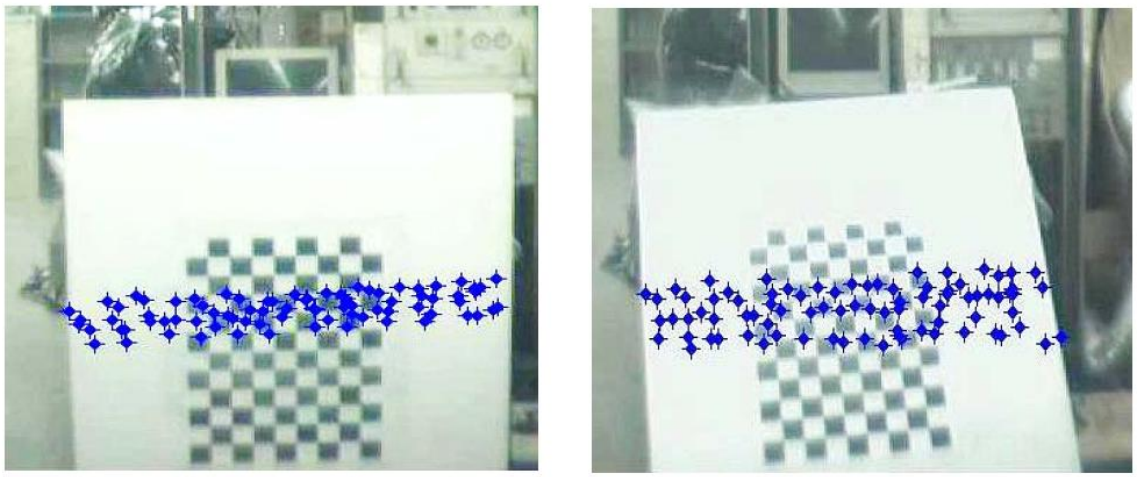


Figure 3-7. Two checkerboard positions. The LIDAR points are indicated by blue dots. The calibration method proposed in this chapter is used to estimate $\mathbf{R}_l^c$ and $\mathbf{t}_l^c$.

3.3.3 Application in Automated Navigation

The calibration method has been integrated into our mobile sensing system. This mobile sensing system is designed to detect and track surrounding vehicles, which is the first and fundamental step for any of the automatic traffic surveillance systems. However, object detection is a big challenge for the moving platform. Both the foreground and the

background are rapidly changing, which makes it difficult to extract the foreground regions from the background. In our research, the sensor fusion technique is used to compensate for the spatial motion of the moving platform. Fig. 3-8 gives two images from the vehicle detection video.

Figure 3-8. Two sensor fusion image frames. The red rectangle is an enlarged image of the detected area.

In Fig. 3-8(a), there are totally four vehicles detected by the LIDAR, where the farthest vehicle is 55 meters away from our mobile sensing system. It's hard to obtain the vehicle's distance and orientation from an image only. The LIDAR points provide a reliable estimation of this vehicle's position. In Fig. 3-8(b), one car parallel to the probe vehicle is detected by the LIDAR. Meanwhile, it is partially visible in the image, together with its shadow. Although this vehicle is hardly recognizable in the image, with a wide angle of view, the LIDAR data provide enough information to reconstruct the location of the vehicle.

Our experiment results illustrate that the calibration algorithm provides good results. For the automatic navigation application, the LIDAR provides reliable distance information. The sensor fusion system combining LIDAR and computer vision information sources presents distance and orientation information. This system is helpful for vehicle detection and tracking applications.

3.4 Summary

In this chapter, we developed a novel calibration algorithm to obtain the geometry transformation between a multi-plane LIDAR system and the camera vision system. This calibration method requires the LIDAR and camera to observe a checkerboard simultaneously. A few checkerboard poses are observed and recorded. The calibration approach has two stages: closed-form solution followed by a maximum likelihood criterion based optimization. Both simulation and real world experiments have been carried out. The experiment results show that the calibration approach is reliable. This approach will be used in the vehicle detection and tracking.

Chapter 4

Tightly-Coupled LIDAR and Computer Vision Integration for Vehicle Detection

Computer vision sensors are generally used in current mobile platform based object detection and tracking systems. However, the application of vision sensors is far from sufficient: clustering, illumination, occlusion, among many other factors, affect the overall performance. In contract, a LIDAR sensor provides range and azimuth measurements from the sensor to the targets. However, the accuracy of its measurements depends on the reflectivity of the targets and the weather. The fusion of camera and active sensors such as LIDAR is being investigated in the context of on-board vehicle detection and tracking.

In this chapter, we propose a tightly coupled LIDAR/CV system, in which the LIDAR scanning points are used for hypothesizing regions of interest and for providing error correction to the classifier, while the vision image provides object classification information. LIDAR object points are first transformed into image space. ROIs are generated using the LIDAR feature detection method. An Adaboost classifier based on computer vision systems [74] is then used to detect vehicles in the image space. Dimensions and

distance information of the detected vehicles are calculated in body-frame coordinates. This approach provides a more complete and accurate map of surrounding vehicles in comparison to the single sensors used separately. One of the key features of this technique is that it uses LIDAR data to correct the Adaboost classification pattern. Moreover, the Adaboost algorithm is utilized both for vehicle detection and for vehicle distance and dimension extraction. Then the classification results provide compensatory information to the LIDAR measurements.

This chapter is organized as follows: in Section 4.1 a brief introduction of the vehicle detection system is given, followed by a description of the LIDAR system. Section 4.2 describes the vision based system. The vehicle detection algorithm is introduced in Section 4.3. A vehicle tracking approach using particle filter is proposed in Section 4.4. Experimental results of vehicle detection are provided in Section 4.5, followed by conclusions and future work in Section 4.6.

4.1 Overview of the Vehicle Detection System

Vehicle detection is the first and fundamental step for any Driver Assistant System (DAS) or automatic driving applications. With sensors mounted on a moving platform,

the detected data change rapidly, making it difficult to extract objects of interest. In our approach, we compensate for spatial motion of the moving platform.

4.1.1 Multi-Module Architecture

The input of our vehicle detection system consists of two LIDAR sensors and a single camera, as is shown in Fig. 1-1. The specifications of the LIDAR sensor and the camera have been presented in Chapter 3. The areas covered by the two LIDAR sensors overlap with each other. The camera is placed inside of the vehicle, i.e., behind the rear-view mirror. The field of view of this camera is fully covered by the LIDAR ranging space.

Fig. 4-1 presents the flow chart of the proposed vehicle detection system. It consists of four subsystems: a LIDAR-based subsystem, a coordinate transformation subsystem, a vision-based subsystem, and a vehicle detection subsystem. The LIDAR-based subsystem as well as the coordinate transformation subsystem is introduced in this section. The offline vision training subsystem and the coordinate transformation subsystem will be introduced in the next section.

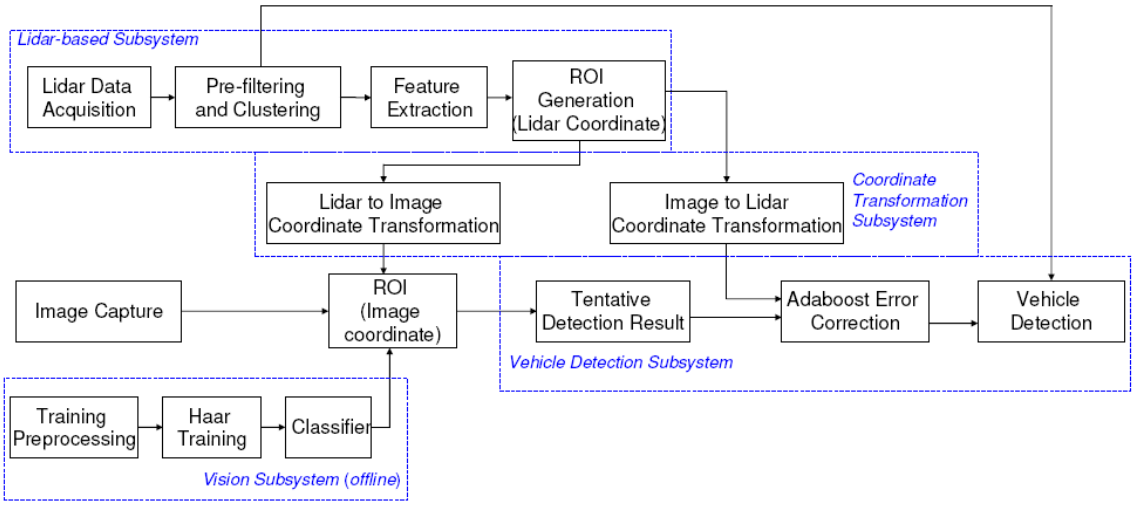


Figure 4-1. Flow chart of the mobile sensing system.

4.1.2 LIDAR-Based Subsystem

The LIDAR Data Acquisition Module uses the IBEO External Control Unit (ECU) to communicate, collect and combine data from the two LIDAR sensors.

The Prefiltering and Clustering Module aims to transform the scan data from distances and azimuths to positions, and cluster the incoming data into segments using a Point-Distance-Based Methodology (PDBM) [21]. If there exists any segment consisting of less than three points, and the distance of this segment is greater than the given threshold, these points are considered as noise. Then the segment will be disregarded.

The Feature Extraction Module extracts the primary features in the cluster. The main feature information in one segment is its geometrical representation, such as lines and

circles. One of the advantages of these geometrical features is that they occupy far less space than storing all the scanned points.

A vehicle may have any possible orientation. The contour of a vehicle is constructed by four sides: front, back, left-side and right-side. The LIDAR sensor can capture one side, or two neighboring sides, as is shown in Fig. 4-2.

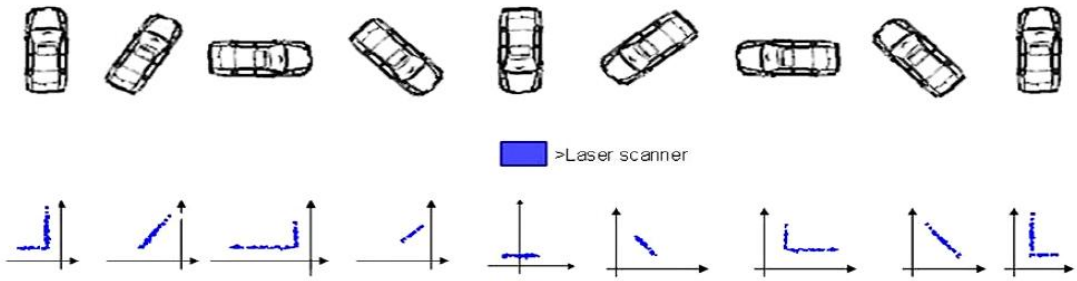


Figure 4-2. The orientation of the vehicle significantly changes its appearance in the scan data frame. The rectangles show the position of the LIDAR.

When the object is close to the probe vehicle, the extracted feature provides enough information for object classification. However, if the target is far away, it may be represented by only one or two scanning points. For those objects with only a few LIDAR points, it's difficult to get reliable size, location, or orientation information from the scan data alone. Note that the computer vision image also contains size and orientation information, which can be extracted by the object classification technique. Therefore, the LIDAR scan data and Adaboost output are complimentary to each other.

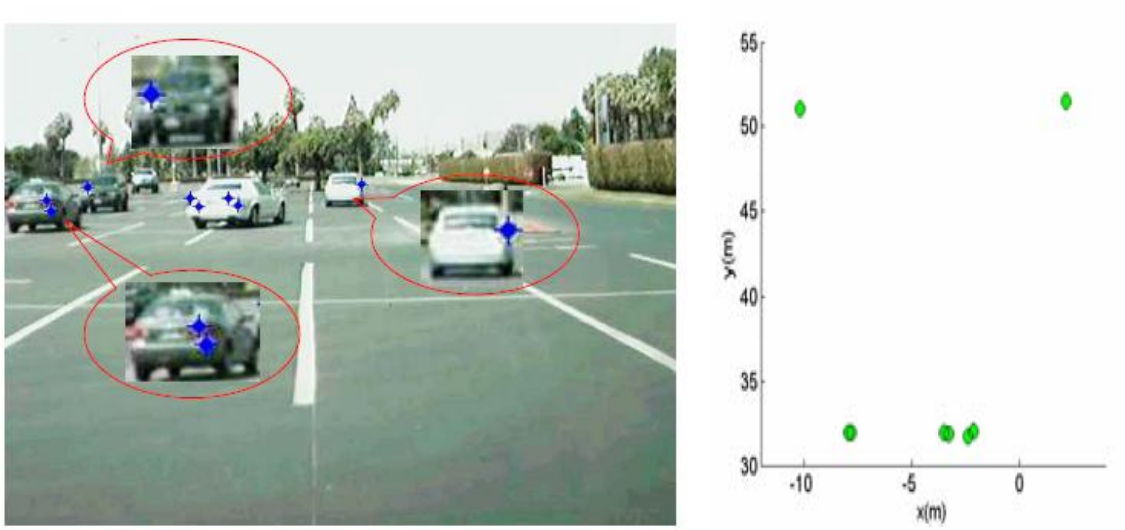
The ROI Generation Module calculates the position of ROI bounding boxes in LIDAR coordinates. It is worth mentioning that the ROI is not defined by the LIDAR points alone, since the scan points on one target may not be able to represent its full dimension. In our algorithm, the width and length of ROI are defined by both LIDAR data points and the maximum dimension of a potential vehicle.

Each ROI is defined as a rectangular area in the image. The bottom of the rectangle is the flat ground. The top of the rectangle is set to be the maximum height of a car. The left and right edges are obtained from the furthest left and right scanning points in a cluster, as well as the typical width of a car.

4.1.3 Coordinate Transformation Subsystem

The LIDAR to Image Coordinate Transformation Module transforms each of the LIDAR points into the image frame. The relative position and orientation between the sensors should be obtained for the transformation. We propose a unique multi-planar LIDAR and computer vision calibration algorithm in Chapter 3, which calculates the geometry transformation matrix between the 'multiple invisible beams' of the LIDAR sensors and the camera. This calibration approach is applied in the sensor fusion systems.

During the road test, each LIDAR scan point $\mathbf{P}_l$ is transformed into camera coordinate as $\mathbf{P}_c$. $\mathbf{P}_c$ is then transformed into point $\mathbf{p}_c$ in the image plane. Fig. 4-3 illustrates the LIDAR scan points and its transformation result in the image reference systems.



(a) (b)

Figure 4-3. The LIDAR scan points. (a) Points in the LIDAR coordinate system. Each green dot is one scan point. (b) These points are transformed to the image frame. Each blue star in the image is one LIDAR point. Vehicles in the red circles are the enlarged image of the detected area.

After the LIDAR to image transformation coefficients are calculated, ROIs generated in the LIDAR-based subsystem is converted into the image frame. A larger ROI is generated due to the inaccuracy of the transformation from LIDAR data to image data.

The Image to LIDAR Coordinate Transformation Module is called to correct the Adaboot classification result. More details will be given in the following sections.

4.2 Vision-Based System

Object classification from the hypothesized ROIs is required for vehicle detection purpose. The feature representations are used for object classifiers by an Adaboost algorithm [74]. Viola et al. proposed that the object is detected based on a boosted cascade of feature classifiers, which performs feature extraction and combines features as simple weak classifiers to a strong one.

The Adaboost classifier should be off-line trained using target as well as non-target images. In our system, the target images, i.e., the rear view of the vehicles, are called positive samples; while the non-vehicles are named as negative samples. Fig. 4-4 illustrates some of the positive as well as negative samples in the training dataset. The training samples are taken from both the Caltech vehicle image dataset [75] and the video collected by our test vehicle. The positive samples include passenger cars, vans, trucks and trailers. The negative sample sets include roads, traffic signs, buildings, plants and pedestrians.

4.2.1 Image Training Preprocessing

Both the positive and negative samples are used by the vision system for data training. They are originally colored images. In order to remove the effects of various illumination conditions and camera differences, grey-level transformation is required as a preprocess-

Figure 4-4. Positive and negative samples used in Adaboost training.

ing step. In our system, the grey-level normalization method is applied to the whole image, which transforms grey-level of the image to be in [0, 1] domain. The color image is transformed by [76]:

$$I^*(x,y) = \frac{I(x,y) - I_{min}}{I_{max} - I_{min}} \tag{4-1}$$

where $I(x,y)$ is the intensity of pixel (x,y), I_{min} and I_{max} are the minimum and maximum values in this image, respectively. $I^*(x,y)$ is the normalized grey-level value.

The next step is to normalize the size of all the positive samples. It is implemented before training since different resolution may cause different number of features to be

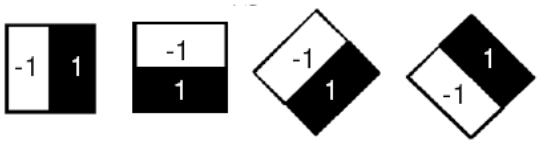
counted. Size of the normalized positive samples determines minimum size of the object that can be detected [77]. In our test, the normalized image size is set to be 25×25 pixels.

4.2.2 Haar Training

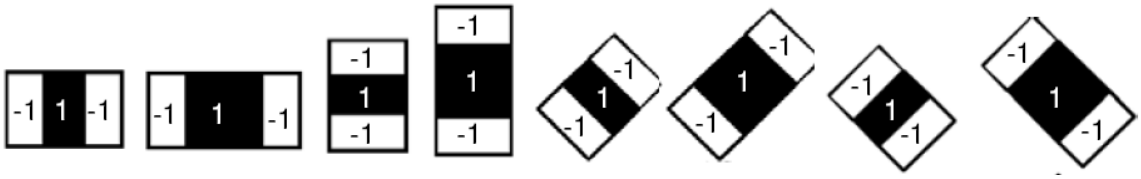
In the vision-based system, Haar-like features are used to classify objects. This approach combines more complex classifiers in a "cascade" which quickly discard the background regions while spending more computation on the Haar-like area [74].

More specifically, we use 14 feature prototypes for the Haar training [78]. These features represent the characteristic properties like edge, line, and symmetry. The features prototypes can be grouped into three sets:

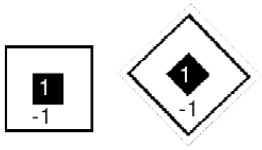
- Edge features: the difference between the sum of the pixels within two rectangular regions.



- Line features: the sum within two outside rectangular regions subtracted from the sum in the center rectangular.



- Center-surround features: the difference between the sum of the center rectangular and the outside area.



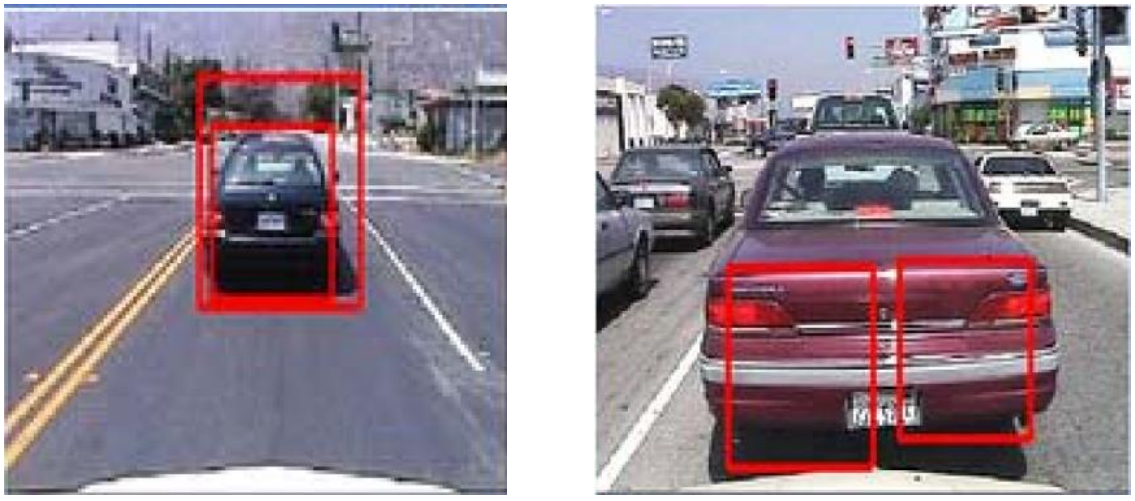
Here black areas with '-1' have negative weights, and white areas with '1' have positive weights.

After the weak classifier has been trained at each stage, the classifier is able to detect almost all the targets of interest while rejecting certain non-target objects. Then a cascade of classifiers is generated to form a decision tree. In our training process, there are totally 15 stages. Each stage was trained to eliminate 60% of the non-vehicle patterns, and the hit rate in each stage is set to be 0.998. Therefore, the total false alarm rate for this cascade classifier is supposed to be $0.4^{15} \approx 1.07e-06$, and the hit rate should be around $0.998^{15} \approx 0.97$.

4.3 Moving Vehicle Detection System

The Adaboost algorithm designs a strong classifier that can detect multiple objects in the given image. However, there is no guarantee that this strong classifier is optimal for the object detection. In contrast to the classic Adaboost algorithm, in our test there is only one vehicle in each ROI defined by the LIDAR clustering algorithm. Therefore, it is not necessary for the Adaboost algorithm to detect several possible targets in one ROI. The classification correction technique is proposed to utilize the LIDAR scanning data to reduce the redundancy in the Adaboost detection results.

There are two kinds of redundancy errors in the classification results. Fig. 4-5 gives some examples of these two cases. Both have detected more than one object, while the ground truth is that there is only one vehicle. One kind of error is that the Adaboost detects two possible targets, while the area of the smaller one is almost covered by the larger one, as is shown in Fig. 4-5(a). Another error shown in Fig. 4-5(b) is that all the detected areas belong to the same object, while none of them covers the full body of the target.



(a) (b)

Figure 4-5. Two kinds of redundancy errors in Adaboost detection.

Suppose in the i-th ROI, there exists a LIDAR point cluster $\mathfrak{R}_i$, which has the following features in the LIDAR coordinate system: c_i^{LIDAR} as the center of the cluster, w_i^{LIDAR} as the width of the object, and l_i^{LIDAR} as the possible length of the vehicle. On the image side, there are n detected target candidates denoted as d_1, d_2, ..., d_n. Initially, d_1, d_2, ..., d_n are transformed from image coordinate to camera coordinate, then to LIDAR coordinates.

The scan points in LIDAR coordinates are denoted as D_1, D_2, ..., D_n. The j -th candidate D_j has center $c_{i,j}^{DETECT}$, width $w_{i,j}^{DETECT}$ and height $h_{i,j}^{DETECT}$. Then in the LIDAR coordinate frame, two vectors are defined as $\mathbf{m}_i^{LIDAR} = (c_i^{LIDAR}, w_i^{LIDAR})$ and $\mathbf{m}_i^{DETECT} = (c_{i,1}^{DETECT}, w_{i,1}^{DETECT}, ..., c_{i,n}^{DETECT}, w_{i,n}^{DETECT})$. Here $\mathbf{m}_i^{LIDAR}$ is the

information measured by LIDAR, and $\mathbf{m}_i^{DETECT}$ consists of measurements in LIDAR coordinate systems which are transformed from the image reference system. The coefficient $\mathbf{w}_i$ for mapping multiple target areas to the LIDAR information satisfies:

$$\mathbf{w}_i = \text{argmin}||\mathbf{m}_i^{LIDAR} - \sum_{j=1:n} \mathbf{w}_{i,j}\mathbf{m}_{i,j}^{DETECT}|| \tag{4-2}$$

where $||.||$ is the Euclidean norm. $\mathbf{w}_i$ is then used as a weight to recalculate the detected area, which is a combination of all the detected objects in one ROI.

The LIDAR scanning points and the Adaboost results are then combined to generate a complete map of vehicles. A summary of the vehicle detection process is given here:

Algorithm 1: Vehicle Detection

Given LIDAR points cluster $\mathfrak{R}_i$ with features $(c_i^{LIDAR}, w_i^{LIDAR}, L_i^{LIDAR})$, detected target candidate d_1, d_2, ..., d_n in the image;

if no object detected in the ROI
  enlarge ROI and search again
else
  define $\mathbf{m}_i^{LIDAR} = (c_i^{LIDAR}, w_i^{LIDAR})$
  transform d_1, ..., d_n in the image coordinate frame to LIDAR reference frame
  define $\mathbf{m}_i^{DETECT} = (c_{i,1}^{DETECT}, w_{i,1}^{DETECT}, ...)$
  calculate the weight vector which minimize $||\mathbf{m}_i^{LIDAR} - \sum_{j=1:n} \mathbf{w}_{i,j}\mathbf{m}_{i,j}^{DETECT}||$
end if

The detected vehicle is located at $\sum_{j=1:n} \mathbf{w}_{i,j}\mathbf{c}_{i,j}^{DETECT}$.

The vehicle detection system proposed in this chapter can be summarized as:

- The Adaboost classifier training is done offline with both positive and negative samples.
- ROI is defined by the LIDAR scan data. No more than one vehicle is assumed to exist in each ROI.
- Use the Adaboost classifier to make a preliminary vehicle detection.
- LIDAR data is used to correct the Adaboost redundancy error, and to merge the detect area in one ROI.
- Combine the Adaboost detected area (in LIDAR coordinate) and the LIDAR output to generate a complete vehicle distance and dimension map.

4.4 Vehicle Tracking System

The LIDAR and computer vision sensors are integrated in a probabilistic manner for vehicle tracking. In our research, a Sampling Importance Resampling (SIR) Particle filter is used as the tracker, which is a sophisticated model estimation technique [80]. Unlike the commonly used Kalman filter and EKF, the particle filter does not assume that the linear dynamic system is perturbed by Gaussian noise. The key idea of the particle filter

is to represent the estimation by a set of random samples (they are called *particles*) with associated weights.

4.4.1 Particle Filter

Let $\mathbf{x}_t^{(i)}$ be the i-th sample of the position and velocity of the target at time t, $i = 1, 2, ..., N_s$, where N_s is the total number of samples or particles in the particle filter. $w_t^{(i)}$ is the *i*-th weight at time t associated with $\mathbf{x}_t^{(i)}$. The procedure of the SIR particle filter is defined as follows:

(1) Initial Particle Generation

Generate N_s particles $\{\mathbf{x}_0^{(i)}, w_0^{(i)}\}$, $i = 1, 2, ..., N_s$. Here $\mathbf{x}_0^{(i)}$ is obtained from vehicle detection results, and $w_0^{(i)} = 1/N_s$.

(2) Particle Updating

For each particle $\mathbf{x}_{t-1}^{(i)}$ at time $t-1$, generate a particle $\mathbf{x}_t^{(i)}$ at time t. This step corresponds to the prediction step in Kalman filter and EKF. However, in Kalman filter and EKF, at time t the state is updated only once. Here in the particle filter, each of the particles should be updated, so totally N_s particle updating calculations are implemented. In our system $\mathbf{x}_t^{(i)} = [\mathbf{p}_t^{(i)} \ \ \mathbf{v}_t^{(i)}]$ is the sample of the position and velocity, so a linear model is used to update the particles:

$$\mathbf{p}_t^{(i)} = \mathbf{p}_{t-1}^{(i)} + \mathbf{v}_{t-1}^{(i)} T + n \tag{4-3}$$

where T is the time interval, and n is the noise.

(3) Particle Weighting

Each particle $\mathbf{x}_0^{(i)}$ at time t is associated with the weight $w_t^{(i)}$, which is also called the *importance factor*. The weight at time t is a function of the weight at time $t-1$, and the probability function of measurements and states at time t. The importance factor is commonly calculated as [80]:

$$w_t^{(i)} = w_{t-1}^{(i)} p(\mathbf{z}_t | \mathbf{x}_t^{(i)}) \tag{4-4}$$

where $\mathbf{z}_t$ is the measurement at time t. In our sensor fusion system, both LIDAR and computer vision sensors are utilized for vehicle tracking. Therefore, the measurement is $\mathbf{z}_t = \{{}^{l}\mathbf{z}_t; {}^{c}\mathbf{z}_t\}$, where ${}^{l}\mathbf{z}_t$ is the output of LIDAR, and ${}^{c}\mathbf{z}_t$ is the measurement of the camera. This step corresponds to the update step in Kalman filter and EKF. The probability $p(\mathbf{z}_t | \mathbf{x}_t^{(i)})$ will be discussed in the following section.

(4) Resampling

A common problem with the particle filter is the degeneracy, which is the phenomenon that after a few iterations only one particle has non-negligible weight [80]. $\widehat{N_{eff}}$ has been defined to measure the degeneracy, which is calculated as [80]:

$$\widehat{N_{eff}} = \frac{1}{\sum_{i=1}^{N_s} (w_t^{(i)\prime})^2} \tag{4-5}$$

where $w_t^{(i)\prime}$ is the normalized weight of the i-th particle at time t with $w_t^{(i)\prime} = \frac{w_t^{(i)}}{\sum_{i=1}^{N_s} w_t^{(i)}}$. If $\widehat{N_{eff}}$ is less than a given threshold N_T, the degeneracy is detected.

Resampling is performed at each iteration. It is designed to eliminate particles that have small weights and replicate particles that have large weights. Particles that have large weights are considered to be the "good" particles while the particles with small weights are "bad" particles. The resampled weights are set as $w_t^{(i)} = 1/N_s$.

After obtaining $w_t^{(i)}$, the posterior filtered density can be approximated as [80]:

$$p(\mathbf{x}_t|\mathbf{z}_{1:t}) \approx \sum_{i=1}^{N_s} w_k^i \delta(\mathbf{x}_t - \mathbf{x}_t^{(i)}) \tag{4-6}$$

4.4.2 The Sensor Model

The sensor model describes the process by which the sensor measurements are made in the physical world. It relates the sensor output to the state of the vehicle. In our applications it is defined as the conditional probability $p({}^l\mathbf{z}_t; {}^c\mathbf{z}_t|\mathbf{x}_t^{(i)})$, which is the probability of the LIDAR and computer vision sensor measurements given the state of the vehicle.

A LIDAR sensor model is described in [82], which represents the probability as a mixture of four distributions corresponding to four types of measurement errors: the

small measurement noise, errors due to unexpected objects or obstacles, errors due to failure to detect objects, and random unexplained noise.

Let $z_t^{(k)*}$ denote the true distance to an obstacle, $z_t^{(k)}$ denote the recorded measurement, and $z_{\max}$ denote the maximum possible reading. The small measurement error is defined as a Gaussian distribution p_{hit} over the range $[0,\ z_{\max}]$ with mean $z_t^{(k)*}$ and standard deviation σ_{hit}.

The LIDAR detection zone is often blocked by the moving vehicles, which leads to the measurement whose length is shorter than the true length. This particular type of measurement error is modeled by a truncated exponential distribution p_{short} with the coefficient λ_{short}.

LIDAR sometimes fails to detect obstacles due to low reflectivity of the target. The errors due to failure to detect objects is defined as a pseudo point-mass distribution $p_{\max}$ centered at $z_{\max}$.

Finally, unexplainable measurements may be returned by the LIDAR sensor, which is caused by interference. This type of error is modeled by a uniform distribution p_{rand} over the entire measurement range.

$p(\mathbf{z}_t|\mathbf{x}_t^{(i)})$ is calculated as a combination of the four types of errors as:

$$p({}^{l}\mathbf{z}_t|\mathbf{x}_t^{(i)}) = \alpha_{\text{hit}} p_{\text{hit}}({}^{l}\mathbf{z}_t|\mathbf{x}_t^{(i)}) + \alpha_{\text{short}} p_{\text{short}}({}^{l}\mathbf{z}_t|\mathbf{x}_t^{(i)}) + \alpha_{\text{max}} p_{\text{max}}({}^{l}\mathbf{z}_t|\mathbf{x}_t^{(i)}) + \alpha_{\text{rand}} p_{\text{rand}}({}^{l}\mathbf{z}_t|\mathbf{x}_t^{(i)}) \tag{4-7}$$

where α_{hit} , α_{short} , α_{max} and α_{rand} are the weights for p_{hit} , p_{short} , p_{max} and p_{rand}, respectively. $\alpha_{\text{hit}} + \alpha_{\text{short}} + \alpha_{\text{max}} + \alpha_{\text{rand}} = 1$. The parameters in Eqn. (4-7) are commonly used as the *a priori* information, which are obtained by data training.

A camera weight model is proposed in [81] as:

$$p({}^{c}\mathbf{z}_t|\mathbf{x}_t^{(i)}) = \begin{cases} S & \text{the object is in the camera detection zone.} \\ 1-S & \text{the object is out of the camera detection zone.} \end{cases} \tag{4-8}$$

where S is a constant, $0 \leq S \leq 1$. This model is based on the assumption that the camera is able to detect all the objects in the detection zone.

The sensor fusion probability model is calculated based on the LIDAR probability model as well as the camera probability model. It is proposed in [81] that ${}^{l}\mathbf{z}_t$ and ${}^{c}\mathbf{z}_t$ are independent measurements. So

$$p({}^{l}\mathbf{z}_t; {}^{c}\mathbf{z}_t|\mathbf{x}_t^{(i)}) = p({}^{l}\mathbf{z}_t|\mathbf{x}_t^{(i)}) p({}^{c}\mathbf{z}_t|\mathbf{x}_t^{(i)}) \tag{4-9}$$

However, in our system LIDAR and the camera are not two independent sensors. They have been calibrated to observe the same target, and the geometric relationships are given as *a priori* information. Moreover, the classification result of the camera is corrected by the LIDAR output.

In this dissertation, a novel sensor fusion model is proposed. As defined in [82], the LIDAR tracking process is modeled as a mixture of four types of errors: the small measurement noise, unexpected objects detection error, detection failure error, and random unexplained noise. In our research, the small measurement noise error is considered to be the exclusive error source of the LIDAR sensor. The other types of error are removed by the integration of LIDAR and camera. The LIDAR tracking model is:

$$p({}^{l}\mathbf{z}_t|\mathbf{x}_t^{(i)}) = \begin{cases} \frac{1}{\sqrt{2\pi}\delta}\exp -\frac{({}^{l}\mathbf{z}_t - {}^{l}\mathbf{z}_t^*)^2}{2\delta^2} & 0 \le {}^{l}\mathbf{z}_t \le z_{\max} \\ 0 & \text{otherwise} \end{cases} \tag{4-10}$$

in which the parameters are the same as the coefficients defined in Thrun's LIDAR model [82].

The computer vision-based vehicle tracking is implemented by KLT tracking [74, 85]. A function $s(x_{t(i)}^m)$ is used to determine if a predicted corner $x_{t(i)}^m$ is close to an observed corner $z_{t(i)}$. $s(x_{t(i)}^m)$ is defined as $s(x_{t(i)}^m) = -\sum_{j=1}^{M}\exp\left(d_{t(i,j}^m\right)^2$, where M is the total number of corners, $\left(d_{t(i,j}^m\right)^2 = ||z_{t(j)} - x_{t(i)}^m||^2$ is the Euclidean distance between the *i*-th detected corner of the *m*-th particle $x_{t(i)}^m$ and the *j*-th extracted corner $z_{t(j)}$ [85]. The camera model is defined as [85]:

$$p({}^{c}\mathbf{z}_t|\mathbf{x}_t^{(i)}) = \exp\left(-\sum(s(x_{t(i)}^m) - 1)^2\right) \tag{4-11}$$

where S is a constant, $0 \leq S \leq 1$. S is the successful feature tracking rate, which is obtained by the training image data.

Finally, the sensor fusion tracing system has totally three measurement situations: 1) the vehicle is tracked by both LIDAR and camera. In this case the single sensor tracking error is eliminated by the sensor integration technique proposed in Section 4.3; 2) the vehicle is out of camera detection zone, so it is tracked by the LIDAR alone; 3) and the case that the vehicle is tracked by the camera but not detected by the LIDAR sensor due to factors such as distance or weak reflection. The sensor fusion model is given as:

$$p({}^{l}\mathbf{z}_t; {}^{c}\mathbf{z}_t|\mathbf{x}_t^{(i)}) = \begin{cases} \alpha p({}^{l}\mathbf{z}_t|\mathbf{x}_t^{(i)}) + \beta p({}^{c}\mathbf{z}_t|\mathbf{x}_t^{(i)}) & \text{target is tracked by both LIDAR and camera} \\ p({}^{l}\mathbf{z}_t|\mathbf{x}_t^{(i)}) & \text{target is tracked by LIDAR alone} \\ p({}^{c}\mathbf{z}_t|\mathbf{x}_t^{(i)}) & \text{target is tracked by camera alone} \end{cases} \tag{4-12}$$

where the weight $0 \leq \alpha \leq 1$ and $0 \leq \beta \leq 1$ are two coefficients obtained by data training, $\alpha + \beta = 1$, which allows to balance the LIDAR and computer vision information.

The weight $w_t^{(i)}$ can be calculated using Eqn. (4-4). The particle filter is summarized in Algorithm 2. Unlike the Kalman filter or EKF, particle filter can track vehicle state with multi-model or arbitrary distributions.

4.5 Experiment Results

The experiment results of vehicle detection are discussed in this section. To evaluate the performance of the proposed system, a dataset of 377 images was used for training and testing. These images are taken from both the Caltech vehicle image dataset [75] and the video samples collected by the probe vehicle. Another test dataset consists of 526 images with synchronized scanning data are used for performance evaluation. The test dataset was recorded in a local parking lot on different days during different seasons.

Algorithm 2: Particle Filter for Sensor Fusion Systems

Input: $\{\mathbf{x}_{t-1}^{(i)}, w_{t-1}^{(i)}\}$, $i = 1, 2, ..., N_s$: set of weighted particles at time $t-1$
$\mathbf{z}_t = ({}^{l}\mathbf{z}_t; {}^{c}\mathbf{z}_t)$: LIDAR and computer vision measurement at time t
Output: $\{\mathbf{x}_t^{(i)}, w_t^{(i)}\}$, $i = 1, 2, ..., N_s$: set of weighted particles at time t
Process:
for $i = 1$ to N_s do
  Predict $\mathbf{x}_t^{(i)}$ as $\mathbf{p}_t^{(i)} = \mathbf{p}_{t-1}^{(i)} + \mathbf{v}_{t-1}^{(i)} T + n$
  $w_t^{(i)} = w_{t-1}^{(i)} p(\mathbf{z}_t | \mathbf{x}_t^{(i)})$
end for
calculate $\widehat{N_{eff}}$
if $\widehat{N_{eff}} < N_T$
  for $i = 1$ to N_s do
    computer $w_t^{(i)}$ using Eqn. (4-4), in which $p({}^{l}\mathbf{z}_t; {}^{c}\mathbf{z}_t | \mathbf{x}_t^{(i)})$ is given in Eqn. (4-10)
    update the particle with $\{\mathbf{x}_t^{(i)}, 1/N_s\}$
  end for
  else
    $Z = \sum_{i=1}^{N_s} w_t^{(i)}$
    for $i = 1$ to N_s do
      update the particle with $\{\mathbf{x}_t^{(i)}, Z^{-1} w_t^{(i)}\}$
    end for
end if

The Hit rate (HR), false alarm rate (FAR) and region detection rate (RDA) are used to evaluate the performance of this system. Here HR is the number of detected vehicles over the total number of vehicles. RDA denotes the percentage of 'real' vehicle detection rate. A ‘real’ vehicle detection is that most area of the vehicle is covered by a rectangle, and there is only one rectangle that covers this object. Therefore, a target that is hit may not be a region 'really' detected; and a region detected object is always a hit. The higher the RDA is, the more accurate the detection result will be. Fig. 4-6 illustrates several cases for hit, false alarm, and region detection. In this figure, TR represents the target region in the image, and DR is the detected region by the classifier.

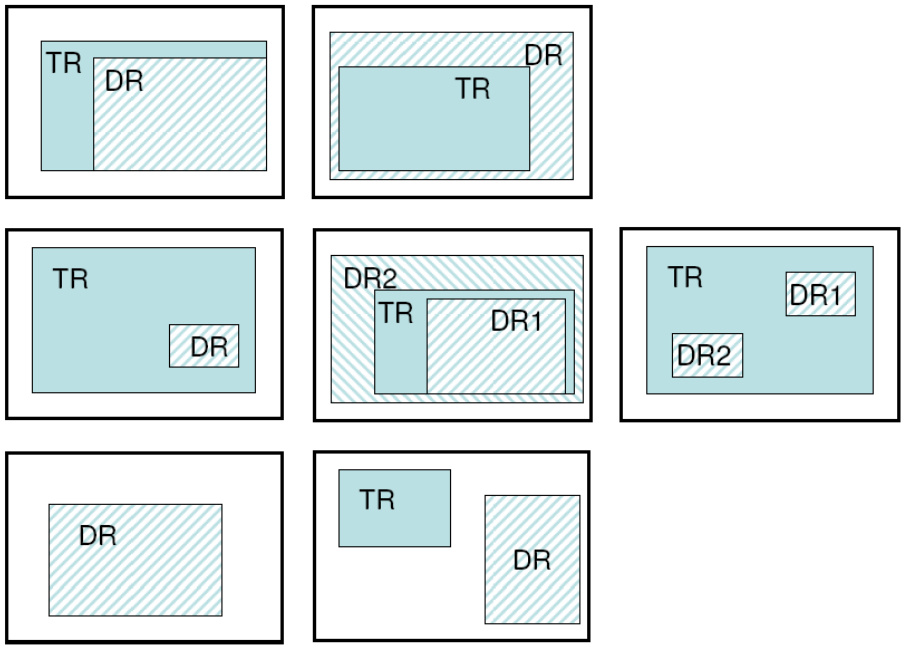


Figure 4-6. Three target detection cases. The first row is region detected, the second row is hit but not region detected, and the third row is false alarm.

Table 4-1 gives the detection performance of the Adaboost classifier (detection with camera only), the classic LIDAR-camera sensor fusion system, and our LIDAR and computer vision-based detection/error correction/combination approach. Here in both the classic sensor fusion technique and our approach, LIDAR data are utilized for ROI generation. The difference lies in the fact that, in our approach LIDAR data help correct the classification result. Table 4-1 shows that our approach both improved the hit rate and reduced the false alarm rate in comparison with the Adaboost classifier. Compared with the classic LIDAR-camera fusion system, our approach improves the region detect rate from 84.85% to 89.32%, since for each hit but not accurately covered object the LIDAR scanning data helps to recomputed position of the target. Most of the overlapping or partial target detection areas are merged during the LIDAR correction process.

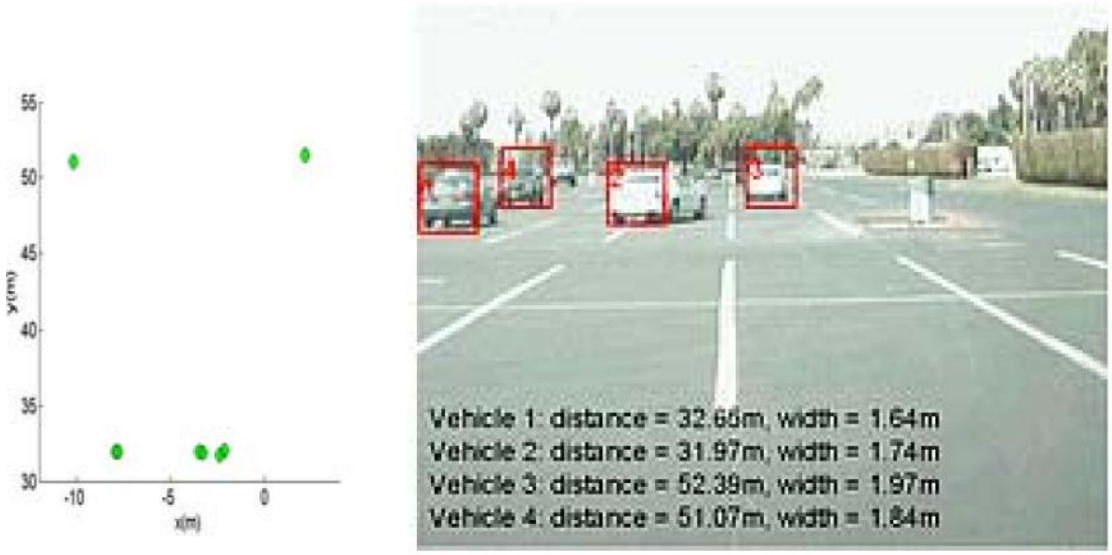
Table 4-1. Detection Result

Type	HR	FAR	RDA
Adaboost	84.17%	3.27%	78.00%
Classic LIDAR-camera Fusion	91.33%	1.78%	84.85%
Our Approach	91.33%	1.78%	89.32%

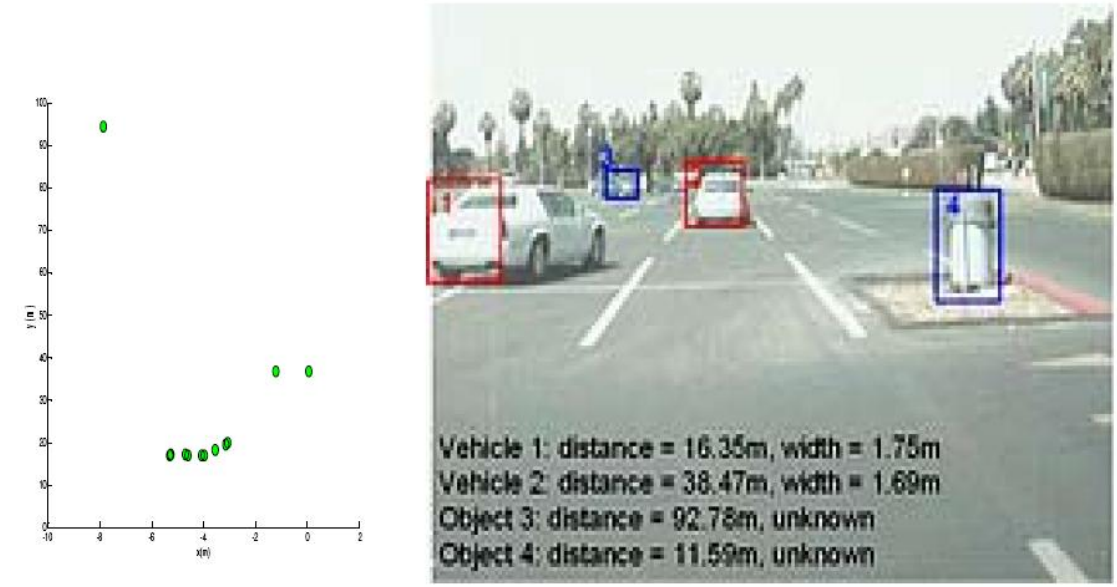
Fig. 4-7 shows some results of the vehicle detection system. The left column is the LIDAR scan points, and the right column illustrates the camera image with information

from the sensor fusion system. In Fig. 7(a), all the vehicles are detected and are marked with a rectangle. Fig. 7(b) shows that the classifier found two vehicles only, which are bounded in a red rectangle. The other two ROIs (shown in blue rectangles) are classified to have non-vehicle objects. In fact, one of them is a trash can. The other is a vehicle at a distance of 92.78 meters. This vehicle is too far away and too small in the image for the classifier to recognize.

During the test, we find out that the hit rate decreases when the distance between our probe vehicle and the target vehicles increases. In our experiment, the targets are detected frame by frame. Therefore, the target vehicles may not be recognized by the classifier in certain frames even if it was recognized in the last frame. Vehicle tracking technique help solve this problem. By running a particle filter based tracking algorithm, the target will be detected in the initial frame or in several initial frames, then be tracked in the following frames. This approach both improves the detection accuracy and reduces the required amount of calculation. HR and RDA will be further improved by vehicle tracking.



(a)



(b)

Figure 4-7. LIDAR scan points and the final vehicle detection results.

4.6 Summary and Discussions

In this chapter, we have proposed a novel vehicle detection system based on tightly integrating LIDAR and computer vision sensor. Distances to the objects are first defined by the LIDAR sensors, and then the object is classified based on computer vision images. In addition, the data from these two complementary sensors are combined for classifier correction and vehicle detection. The experimental results have indicated that, when compared with image-based and classic sensor fusion based vehicle detection systems, our approach has a higher hit rate and a lower false alarm rate. This system is quite useful for modeling and prediction of the traffic conditions over a variety of roadways. Thus it may be used in future autonomous navigation systems.

Chapter 5

LIDAR and Inertial Sensor-Aided Vehicle Positioning

In order for a vehicle to be able to navigate safely and successfully in a given environment, it is necessary that the vehicle knows its position and orientation. Determining the position and orientation is known as the *vehicle localization* problem, or the "*where am I?*" problem. A digital map of the environment is usually provided to the vehicle, which is equipped with a variety of sensors to perceive itself and the environment. Sensor fusion systems are commonly used to integrate the sensory data from disparate sources, so that the output will be more accurate and complete in comparison to the output of one sensor.

In this chapter, a LIDAR and inertial sensor-based vehicle positioning framework is proposed, in which the inertial sensors are used to estimate the position and orientation of the vehicle, while the LIDAR sensor provides very accurate distance and azimuth measurements from itself to the landmarks. An EFK is used to combine the two sensors and predict the position of the vehicle. This approach provides a more accurate positioning estimation in comparison to the single sensor-based method. One of the key features of

this technique is that we do not make the assumption that the navigation is in 2D plane. In our application, the navigation is in 3D coordinates. Moreover, a novel INS and LIDAR calibration method is proposed to transform the landmark in INS coordinate to LIDAR frame of reference, and vice versa.

This chapter is organized as follows: in Section 5.1 a brief introduction of the frames used in this system is given, followed by a description of high speed sensor models. Section 5.2 describes the calibration method, which calculates the geometric relationship between INS and LIDAR systems. The time propagation model is introduced in Section 5.3. LIDAR sensor model is proposed in Section 5.4. Finally, the EKF is provided in Section 5.5, followed by conclusions and future work in Section 5.6.

5.1 Sensor Modeling

The LIDAR and inertial sensor models are discussed in this section. We first briefly describe the frame of reference, then describe the output of each sensor.

5.1.1 Frames

There are basically four frames in the vehicle positioning system. The frames and the quantities presented in these frames are discussed below.

- Inertial frame: Inertial frame is defined by the inertial sensors. Quantities in the inertial frame have a superscript i.
- Tangent frame: Tangent frame is considered to be the fundamental frame. Quantities in the tangent frame have a superscript t. The axes in the body frame are north, east, and down.
- Body frame: Body frame is the coordinate of the vehicle. Quantities in the body frame have a superscript b. The axes in the body frame are forward, right and down.
- LIDAR frame: The output of LIDAR is represented in this frame of reference. Quantities in the LIDAR frame have a superscript l. The axes in the LIDAR frame are defined as right, forward and up.

5.1.2 High Speed Sensor Model

The high speed sensor in this system is the INS, which consists of three orthogonal gyros and three orthogonal accelerometers. The gyro provides change of Euler angles, and the accelerometers give the specific forces.

A. Gyro Model

The gyro measurement is modeled as:

$$ {}^{b}\tilde{\omega}_{ib} = {}^{b}\omega_{ib} + \mathbf{b}_g + \gamma_g \tag{5-1} $$

where ${}^{b}\omega_{ib}$ is the angular velocity of the gyro relative to the inertial frame represented in body frame, ${}^{b}\tilde{\omega}_{ib}$ is the measurement value of ${}^{b}\omega_{ib}$. $\mathbf{b}_g$ is the gyro bias, which is modeled as a random constant plus random noise:

$$ \delta\dot{\mathbf{b}}_g = \omega_{bg} \quad \omega_{bg} \sim \mathcal{N}\left(0, \sigma_{bg}^2\mathbf{I}\right) \tag{5-2} $$

γ_g is the white Gaussian measurement noise. We assume that $\gamma_g \sim \mathcal{N}(0, \sigma_g^2\mathbf{I})$.

B. Accelerometer Model

The accelerometer measurement is modeled as:

$$ {}^{b}\tilde{\mathbf{a}}_b = \mathbf{f}^b + \mathbf{b}_a + \gamma_a \tag{5-3} $$

where γ_a is the measurement noise with Gaussian distribution $\mathcal{N}(0, \sigma_a^2\mathbf{I})$, and $\mathbf{b}_a$ is the accelerometer bias. $\mathbf{b}_a$ is modeled as a random constant plus random noise:

$$ \delta\dot{\mathbf{b}}_a = \omega_{ba} \quad \omega_{ba} \sim \mathcal{N}\left(0, \sigma_{ba}^2\mathbf{I}\right) \tag{5-4} $$

$\mathbf{f}^b$ is the specific force vector in the body frame [83, eqn. (12.11)]:

$$ \begin{aligned} \mathbf{f}^b &= \mathbf{R}_i^b\mathbf{a}^i - \mathbf{g}^b \\ &= {}^{b}\mathbf{a}_b - \mathbf{g}^b \end{aligned} \tag{5-5} $$

where ${}^{b}\mathbf{a}_b = \mathbf{R}_i^b\mathbf{a}^i$ is the acceleration of the vehicle relative to the inertial frame represented in the body frame, $\mathbf{a}^i$ is the acceleration of the accelerometer w.r.t. the iner-

tial frame, g^b is the local gravity vector at the vehicle location represented in the body frame, and $\mathbf{R}_i^b$ is the rotation matrix from the inertial coordinate to the body coordinate.

5.1.3 Low Speed Sensor Model

The LIDAR measurements are a set of points, in which each scan point is defined in a polar coordinate. The output of LIDAR is the distance and azimuth from the target to the sensor. In our system the LIDAR sensor is used to detect landmarks. There are two kinds of landmarks: points (e.g., a light pole) and lines (e.g, walls). The detection output of LIDAR is segmented and classified as points or lines. The LIDAR sensor model will be discussed in section 5.4.

5.2 LIDAR and Inertial Sensor Calibration

In order to effectively extract and integrate 3D information from both inertial sensors and LIDAR systems, the relative position and orientation between these two sensor modalities should be obtained.

The calibration is implemented in three steps: calibration of LIDAR and tangent coordinates, transformation from INS to tangent frame of reference, and the calibration from INS to LIDAR coordinates. The steps are introduced in the following sections.

5.2.1 Calibration of LIDAR and Tangent Frames

Suppose a point ${}^{t}P$ in the tangent frame is located at ${}^{l}P$ in the LIDAR coordinate system, the transformation from the tangent coordinate to the LIDAR coordinate system is:

$$ {}^{l}P = \mathbf{R}_{t}^{l}\ {}^{t}P + {}^{l}\mathbf{T}_{LT} \tag{5-6} $$

where $\mathbf{R}_{t}^{l}$ is a 3×3 orthonormal matrix representing the rotation from tangent frame to LIDAR frame, and ${}^{l}\mathbf{T}_{LT}$ is the translation vector. The subscript L is the origin of the LIDAR coordinate, and T is the origin of the tangent frame. Our goal is to calculate $\mathbf{R}_{t}^{l}$ and ${}^{l}\mathbf{T}_{LT}$.

A. Geometric Constraint

The calibration is implemented through the observation of LIDAR to a planar board. This planar board is placed in front of the LIDAR sensor, which is called the *calibration plane* in this dissertation. The LIDAR measurement points on the calibration plane are called *LIDAR points*, which are not visible to human eyes or computer vision sensors.

A field survey is carried out by GPS to get the position of the calibration plane in tangent frame. At least three points are taken on the plane, whose coordinates in ECEF frame are defined by GPS. Then the coordinates in ECEF are transformed to tangent

frame. The calibration plane is represented in tangent frame by a 3-vector ${}^{t}\mathbf{r}$, which is a unit normal vector to the checkerboard plane in tangent frame.

A geometric constraint can be obtained between LIDAR points and the checkerboard. Since LIDAR points lie on the calibration plane, and ${}^{t}\mathbf{r}$ is the normal vector to the plane, we have:

$$ {}^{t}\mathbf{r} \cdot {}^{t}P = d \tag{5-7} $$

which ${}^{t}P$ is the coordinate of LIDAR point in tangent frame. d is a scalar representing the distance from the origin of tangent frame to the calibration board, which is calculated from the position and orientation of the calibration board. From Eqn. (5-6) we know that

$$ {}^{t}P = (\mathbf{R}_{t}^{l})^{-1}({}^{l}P - {}^{l}\mathbf{T}_{LT}) \tag{5-8} $$

By substituting Eqn. (5-8) into Eqn. (5-7), we have:

$$ {}^{t}\mathbf{r}(\mathbf{R}_{t}^{l})^{-1}({}^{l}P - {}^{l}\mathbf{T}_{LT}) = d \tag{5-9} $$

For any given LIDAR point and calibration board position, Eqn. (5-9) gives a constraint on $\mathbf{R}_{t}^{l}$ and ${}^{l}\mathbf{T}_{LT}$. It will be solved in two consecutive steps: a linear solution, followed by a non-linear optimization.

B. Linear Solution

The LIDAR scan plane in the LIDAR coordinate is ${}^{l}Z = 0$. Each scan point can be represented as ${}^{l}P = [{}^{l}P_x \;\; {}^{l}P_y \;\; 0]^T$. Then Eqn. (5-9) is rewritten as:

$$\begin{aligned} & {}^{t}_{l}\mathbf{r}(\mathbf{R}^l_t)^{-1}\left(\begin{bmatrix} {}^{l}P_x \\ {}^{l}P_y \\ 0 \end{bmatrix} - {}^{t}_{l}\mathbf{T}_{LT}\right) \\ = \;& {}^{t}_{l}\mathbf{r}(\mathbf{R}^l_t)^{-1}\begin{bmatrix} 1 & 0 & \\ 0 & 1 & -{}^{t}_{l}\mathbf{T}_{LT} \\ 0 & 0 & \end{bmatrix}\begin{bmatrix} {}^{l}P_x \\ {}^{l}P_y \\ 1 \end{bmatrix} \\ = \;& d \end{aligned} \tag{5-10}$$

Let's define $\mathbf{X} = (\mathbf{R}^l_t)^{-1}\begin{bmatrix} 1 & 0 & \\ 0 & 1 & -{}^{t}_{l}\mathbf{T}_{LT} \\ 0 & 0 & \end{bmatrix}$, and ${}^{l}\bar{P} = [{}^{l}P_x \;\; {}^{l}P_y \;\; 1]^T$, Eqn. (5-10) is rewritten as

$${}^{t}_{l}\mathbf{r}\mathbf{X}\,{}^{l}\bar{P} = d \tag{5-11}$$

in which $\mathbf{X}$ is the parameter to be solved, which is the integration of two unknown parameters $\mathbf{R}^l_t$ and ${}^{t}_{l}\mathbf{T}_{LT}$.

Eqn. (5-11) can be solved using the standard linear least square algorithm. In order to obtain a better result, multiple calibration planes should be used in the calibration. Suppose in the calibration we use totally N calibration planes, with $M_i, i = 1, ..., N$ LIDAR points on the *i*-th plane. Eqn. (5-11) can be rewritten as a $\mathbf{Ax} = \mathbf{b}$ problem, where $\mathbf{A}$ is a $(\sum_{i=1}^{N}\sum_{j=1}^{M_i}) \times 9$ matrix, and $\mathbf{b}$ is a $(\sum_{i=1}^{N}\sum_{j=1}^{M_i}) \times 1$ vector. Let $\mathbf{X} = \begin{bmatrix} x_{11} & x_{12} & x_{13} \\ x_{21} & x_{22} & x_{23} \\ x_{31} & x_{32} & x_{33} \end{bmatrix}$, the normal vector for the *i*-th plane ${}^{t}_{l}\mathbf{r}_i = [r_{i,1}, r_{i,2}, r_{i,3}]$, the dis-

tance from origin of the tangent frame and the *i*-th calibration plane is d_i, and the *j*-th LIDAR point on the *i*-th calibration plane is ${}^l\bar{P}_{ij} = [{}^lP_{ij,x} \;\; {}^lP_{ij,y} \;\; 1]^T$, Eqn. (5-11) is re-written as:

$$\begin{bmatrix} r_{1,1}{}^lP_{11,x} & r_{1,2}{}^lP_{11,x} & r_{1,3}{}^lP_{11,x} & r_{1,1}{}^lP_{11,y} & r_{1,2}{}^lP_{11,y} & r_{1,3}{}^lP_{11,y} & r_{1,1} & r_{1,2} & r_{1,3} \\ r_{1,1}{}^lP_{12,x} & r_{1,2}{}^lP_{12,x} & r_{1,3}{}^lP_{12,x} & r_{1,1}{}^lP_{12,y} & r_{1,2}{}^lP_{12,y} & r_{1,3}{}^lP_{12,y} & r_{1,1} & r_{1,2} & r_{1,3} \\ \cdots & & & & & & & & \\ r_{1,1}{}^lP_{1M_1,x} & r_{1,2}{}^lP_{1M_1,x} & r_{1,3}{}^lP_{1M_1,x} & r_{1,1}{}^lP_{1M_1,y} & r_{1,2}{}^lP_{1M_1,y} & r_{1,3}{}^lP_{1M_1,y} & r_{1,1} & r_{1,2} & r_{1,3} \\ r_{2,1}{}^lP_{21,x} & r_{2,2}{}^lP_{21,x} & r_{2,3}{}^lP_{21,x} & r_{2,1}{}^lP_{21,y} & r_{2,2}{}^lP_{21,y} & r_{2,3}{}^lP_{21,y} & r_{2,1} & r_{2,2} & r_{2,3} \\ \cdots & & & & & & & & \\ r_{2,1}{}^lP_{2M_2,x} & r_{2,2}{}^lP_{2M_2,x} & r_{2,3}{}^lP_{2M_2,x} & r_{2,1}{}^lP_{2M_2,y} & r_{2,2}{}^lP_{2M_2,y} & r_{2,3}{}^lP_{2M_2,y} & r_{2,1} & r_{2,2} & r_{2,3} \\ \cdots & & & & & & & & \\ r_{N,1}{}^lP_{NM_N,x} & r_{N,2}{}^lP_{NM_N,x} & r_{N,3}{}^lP_{NM_N,x} & r_{N,1}{}^lP_{NM_N,y} & r_{N,2}{}^lP_{NM_N,y} & r_{N,3}{}^lP_{NM_N,x} & r_{N,1} & r_{N,2} & r_{N,3} \end{bmatrix} \cdot \begin{bmatrix} x_{11} \\ x_{21} \\ x_{31} \\ x_{12} \\ x_{22} \\ x_{32} \\ x_{13} \\ x_{23} \\ x_{33} \end{bmatrix}$$
$$= \begin{bmatrix} d_1 & d_1 & \ldots & d_1 & d_2 & \ldots & d_2 & \ldots & d_N \end{bmatrix}^T \tag{5-12}$$

Therefore for each scan point we have a linear equation which is a row in Eqn. (5-12). $\mathbf{X}$ can be calculated using the standard linear least square algorithm. Then $\mathbf{R}_t^l$ and ${}^lT_{LT}$ will be obtained from $\mathbf{X}$.

Let $\mathbf{R}_i$ be the *i*-th row of $\mathbf{R}_t^l$, then $(\mathbf{R}_t^l)^{-1} = (\mathbf{R}_t^l)^T = [\mathbf{R}_1^T \; \mathbf{R}_2^T \; \mathbf{R}_3^T]$. So

$$\begin{aligned} \mathbf{X} &= [\mathbf{R}_1^T \; \mathbf{R}_2^T \; \mathbf{R}_3^T] \begin{bmatrix} 1 & 0 & \\ 0 & 1 & -{}^l\mathbf{T}_{LT} \\ 0 & 0 & \end{bmatrix} \\ &= \begin{bmatrix} \mathbf{R}_1^T \; \mathbf{R}_2^T & -(\mathbf{R}_t^l)^T {}^l\mathbf{T}_{LT} \end{bmatrix} \end{aligned} \tag{5-13}$$

Note that $\mathbf{R}_t^l$ is a rotation matrix, so the columns in $\mathbf{R}_t^l$ are orthogonal to each other. From Eqn. (5-13) we get:

$$\mathbf{R}_t^l = [\mathbf{X}_1 \;\; \mathbf{X}_2 \;\; \mathbf{X}_1 \times \mathbf{X}_2]^T \tag{5-14}$$

where $\mathbf{X}_i$ is the i-th column of the 3×3 matrix $\mathbf{X}$. Moreover, from Eqn. (5-13) we know that:

$$\mathbf{X}_3 = -(\mathbf{R}_t^l)^{-1} {}^l\mathbf{T}_{LT} \tag{5-15}$$

So

$$\begin{aligned} {}^l\mathbf{T}_{LT} &= -\mathbf{R}_t^l \mathbf{X}_3 \\ &= -[\mathbf{X}_1 \;\; \mathbf{X}_2 \;\; \mathbf{X}_1 \times \mathbf{X}_2]^T \mathbf{X}_3 \end{aligned} \tag{5-16}$$

Now $\mathbf{R}_t^l$ and ${}^l\mathbf{T}_{LT}$ have been solved by using Eqn. (5-14) and Eqn. (5-16).

$\mathbf{R}_t^l$ is a rotation matrix, which has the property that $(\mathbf{R}_t^l)^T \mathbf{R}_t^l = \mathbf{I}$. However, the calculated $\mathbf{R}_t^l$ may not meet the property of a rotation matrix. In our calibration a rotation matrix $\hat{\mathbf{R}}_t^l$ is computed by minimizing the Frobenius norm of the difference $\hat{\mathbf{R}}_t^l - \mathbf{R}_t^l$ subject to $(\hat{\mathbf{R}}_t^l)^T \hat{\mathbf{R}}_t^l = \mathbf{I}$ [86], i.e., our goal is to find the $\hat{\mathbf{R}}_t^l$ that minimizing $\mathrm{Tr}\left(\left(\hat{\mathbf{R}}_t^l - \mathbf{R}_t^l\right)^T \left(\hat{\mathbf{R}}_t^l - \mathbf{R}_t^l\right)\right)$ subject to $(\hat{\mathbf{R}}_t^l)^T \hat{\mathbf{R}}_t^l = \mathbf{I}$. The solution is the polar decomposition of $(\mathbf{R}_t^l)^T \mathbf{R}_t^l$ [87].

C. Nonlinear Solution

The linear solution is obtained by the least square method, which aims to minimize an algebraic distance. In this subsection, a nonlinear minimization method is proposed to minimize the differences between the measured Euclidean distances as well as the calcu-

lated distance from the LIDAR points to the calibration plane, which is physically meaningful.

Eqn. (5-9) gives two types of distances: d is the distance from the calibration plane to the origin of tangent frame obtained by field survey, and ${}^{t}\mathbf{r}(\mathbf{R}_t^l)^{-1}({}^{l}P - {}^{l}\mathbf{T}_{LT})$ is the calculated distance. The difference between these two distances is defined as the distance error for one calibration plane pose. The nonlinear solution aims to minimize the sum function of the distance errors for all the plane positions. The sum function is defined as:

$$\sum_{i=1}^{N}\sum_{j=1}^{M_i}[{}^{t}\mathbf{r}_i(\mathbf{R}_t^l)^{-1}({}^{l}P_{ij} - {}^{l}\mathbf{T}_{LT}) - d_i]^2 \tag{5-17}$$

where $\mathbf{r}_i$ defines the *i*-th calibration plane, d_i is the distance from the *i*-th plane to the center of tangent frame, and P_{ij} is the *j*-th LIDAR point with the *i*-th calibration plane. The pair of $\mathbf{R}_t^l$ and ${}^{l}\mathbf{T}_{LT}$ that minimize Eqn. (5-17) are considered to be the rotation and translation matrix to be calculated. Eqn. (5-17) can be minimized as a nonlinear optimization problem by using the Levenberg-Marquardt method [68]. The Levenberg-Marquardt algorithm requires an initial guess of $\mathbf{R}_t^l$ and ${}^{l}\mathbf{T}_{LT}$, which is obtained using the method in 5.2.1 B.

When we get the translation and rotation matrix between tangent frame and LIDAR, any point in the tangent frame can be converted to a point in the LIDAR frame using Eqn. (5-6).

5.2.2. Calibration of LIDAR and INS Systems

The calibration of LIDAR and INS is to obtain the geometric relationship between LIDAR coordinate and body frame.

The INS consists of three gyros and three accelerometers. The gyro provides change of Euler angles, while the accelerometers give the specific force. By integrating the output of the gyros and the accelerometers, we can obtain the rotation and translation matrix between tangent frame and body frame. Let the rotation matrix be $\mathbf{R}_b^t$, and translation vector be ${}^t\mathbf{T}_{TB}$ in which the subscript B is the origin of the body frame, a point in body can be converted to a point in tangent frame as:

$$ {}^tP = \mathbf{R}_b^t \, {}^bP + {}^t\mathbf{T}_{TB} \tag{5-18} $$

where ${}^tP$ is the point in tangent frame, and ${}^bP$ is the point in body frame.

Finally, by substituting Eqn. (5-6) into Eqn. (5-18) we have:

$$ {}^lP = \mathbf{R}_t^l \, (\mathbf{R}_b^t \, {}^bP + {}^t\mathbf{T}_{TB}) + {}^l\mathbf{T}_{LT} \tag{5-19} $$

which is the transformation from body frame to LIDAR frame.

5.3 Time Propagation Error Modeling

The kinematic equations, navigation mechanization equations and the time propagation error model will be given in this section.

5.3.1 Kinematic Equations

The standard form of the kinematic model is $\dot{\mathbf{x}} = \mathbf{F}(\mathbf{x}, \mathbf{u})$, where $\mathbf{x}$ is the state and $\mathbf{u}$ is the input. In our application $\mathbf{x} = \begin{bmatrix} {}^{t}\mathbf{T}_{TB} & {}^{b}\mathbf{v} & \mathbf{\Theta} \end{bmatrix}^T$, where ${}^{t}\mathbf{T}_{TB}$ is the vector from the tangent frame origin to the body frame origin in the tangent coordinate, ${}^{b}\mathbf{v}$ is the velocity of the vehicle in body coordinate, and $\mathbf{\Theta} = \begin{bmatrix} \phi & \theta & \psi \end{bmatrix}^T$ is the roll, pitch and yaw of the vehicle. The system kinematic model is [83, eqn. (12.5)-(12.7)]:

$$\begin{aligned} {}^{t}\dot{\mathbf{T}}_{TB} &= \mathbf{R}_b^{t\,b}\mathbf{v} \\ {}^{b}\dot{\mathbf{v}} &= {}^{b}\mathbf{a}_b - (\mathbf{\Omega}_{ie}^{b} + \mathbf{\Omega}_{ib}^{b})\,{}^{b}\mathbf{v} \\ \dot{\mathbf{\Theta}} &= \mathbf{\Omega}_E^{-1}\,{}^{b}\omega_{tb} \end{aligned} \tag{5-20}$$

where $\mathbf{R}_b^t$ is the rotation matrix from body frame to tangent frame. $\mathbf{f}^b$ and $\mathbf{g}^b$ have been defined in Eqn. (5-5), and ${}^{b}\mathbf{a}_b = \mathbf{f}^b + \mathbf{g}^b$. $\mathbf{\Omega}_E^{-1}$ is defined as [83, eqn. (2.74)]:

$$\mathbf{\Omega}_E^{-1} = \begin{bmatrix} 1 & \sin\phi\tan\theta & \cos\phi\tan\theta \\ 0 & \cos\phi & -\sin\phi \\ 0 & \frac{\sin\phi}{\cos\theta} & \frac{\cos\phi}{\cos\theta} \end{bmatrix} \tag{5-21}$$

${}^{b}\omega_{tb}$ is the angular rate of the body frame w.r.t. the tangent frame represented in body frame with ${}^{b}\omega_{tb} = {}^{b}\omega_{ib} - {}^{b}\omega_{it}$, where ${}^{b}\omega_{ib}$ is the angular rate measurement by the gyro

represented in body frame. ${}^{b}\omega_{it} = \mathbf{R}_t^{bt}{}^{t}\omega_{it}$ with ${}^{t}\omega_{it} = \omega_{ie}\left[\begin{array}{ccc}\cos\bar{\phi} & 0 & -\sin\bar{\phi}\end{array}\right]^T$, where $\bar{\phi}$ represents the latitude.

5.3.2 Navigation Mechanization Equations

The standard form of the navigation mechanization equations is $\dot{\hat{\mathbf{x}}} = \mathbf{F}(\hat{\mathbf{x}}, \tilde{\mathbf{u}})$, where $\hat{\mathbf{x}}$ is the estimate value of the state $\mathbf{x}$, and $\tilde{\mathbf{u}}$ is the measured input. The navigation mechanization equations are given as [83, Eqn. (12.20)-(12.22)]:

$$\begin{aligned} {}^{t}\dot{\hat{\mathbf{T}}}_{TB} &= \hat{\mathbf{R}}_b^{tb}\hat{\mathbf{v}} \\ {}^{b}\dot{\hat{\mathbf{v}}} &= {}^{b}\hat{\mathbf{a}}_b - (\hat{\mathbf{\Omega}}_{ie}^{b} + \hat{\mathbf{\Omega}}_{ib}^{b})^{b}\hat{\mathbf{v}} \\ \dot{\hat{\mathbf{\Theta}}} &= \hat{\mathbf{\Omega}}_E^{-1}{}^{b}\hat{\omega}_{tb} \end{aligned} \tag{5-22}$$

where ${}^{b}\hat{\omega}_{tb} = {}^{b}\hat{\omega}_{ib} - {}^{b}\hat{\omega}_{it}$, ${}^{b}\hat{\omega}_{it} = \hat{\mathbf{R}}_t^{bt}{}^{t}\hat{\omega}_{it}$ with ${}^{t}\hat{\omega}_{it} = \omega_{ie}\left[\begin{array}{ccc}\cos\hat{\bar{\phi}} & 0 & -\sin\hat{\bar{\phi}}\end{array}\right]^T$, $\hat{\bar{\phi}}$ represents the computed latitude. ${}^{b}\hat{\omega}_{ib}$ is defined as:

$${}^{b}\hat{\omega}_{ib} = {}^{b}\tilde{\omega}_{ib} - \hat{\mathbf{b}}_g \tag{5-23}$$

where $\hat{\mathbf{b}}_g$ is the estimation of $\mathbf{b}_g$. We have $\dot{\hat{\mathbf{b}}}_g = 0$.

Finally, ${}^{b}\hat{\mathbf{a}}_b$ is defined as:

$$\begin{aligned} {}^{b}\hat{\mathbf{a}}_b &= {}^{b}\tilde{\mathbf{a}}_b + \hat{\mathbf{g}}_b - \hat{\mathbf{b}}_a \\ &= {}^{b}\mathbf{a}_b - \mathbf{g}^{b} + \mathbf{b}_a + \gamma_a + \hat{\mathbf{g}}_b - \hat{\mathbf{b}}_a \end{aligned} \tag{5-24}$$

where $\hat{\mathbf{g}}_b$ is the local gravity vector calculated at the vehicle location, and $\hat{\mathbf{b}}_a$ is the estimation of $\mathbf{b}_a$.

5.3.3 Time Propagation Error Modeling

The error is given by subtracting the mechanization eqn. (5-22) from the kinematic eqn. (5-20).

A. Position Error Analysis

The error in position is [83, Eqn. (12.34)]:

$$\begin{aligned} {}^t\delta\dot{\mathbf{T}}_{TB} &= \mathbf{R}_b^t\ {}^b\mathbf{v} - \hat{\mathbf{R}}_b^t\ {}^b\hat{\mathbf{v}} \\ &= \hat{\mathbf{R}}_b^t\ {}^b\delta\mathbf{v} - [{}^t\hat{\mathbf{v}}\times]\rho \end{aligned} \tag{5-25}$$

where $\mathbf{R}_b^t = \hat{\mathbf{R}}_b^t\,(\mathbf{I} + [\rho\times])$, ρ is the tangent plane tilt error, and ${}^b\mathbf{v} = {}^b\hat{\mathbf{v}} + {}^b\delta\mathbf{v}$. The second order error has been eliminated.

B. Velocity Error Analysis

The error in velocity is modeled as [83, eqn. (12.35)-(12.36)]:

$$\begin{aligned} {}^b\delta\dot{\mathbf{v}} &= {}^b\dot{\mathbf{v}} - {}^b\dot{\hat{\mathbf{v}}} \\ &= \mathbf{a}^b - (\boldsymbol{\Omega}_{ie}^b + \boldsymbol{\Omega}_{ib}^b){}^b\mathbf{v} - \hat{\mathbf{a}}^b + (\hat{\boldsymbol{\Omega}}_{ie}^b + \hat{\boldsymbol{\Omega}}_{ib}^b){}^b\hat{\mathbf{v}} \\ &= \hat{\mathbf{R}}_t^b \left(\frac{\partial \mathbf{g}^t}{\partial {}^t\mathbf{T}_{TB}} + [{}^t\hat{\mathbf{v}}\times]\frac{\partial {}^t\omega_{ie}}{\partial {}^t\mathbf{T}_{TB}} \right) |_{{}^t\hat{\mathbf{T}}_{TB}}\, {}^t\delta\mathbf{T}_{TB} - (\hat{\boldsymbol{\Omega}}_{ie}^b + \hat{\boldsymbol{\Omega}}_{ib}^b){}^b\delta\mathbf{v} \\ &+ \hat{\mathbf{R}}_t^b([\mathbf{g}^t\times] + {}^t\omega_{ie}({}^t\hat{\mathbf{v}})^T - ({}^t\omega_{ie})^T{}^t\hat{\mathbf{v}}\mathbf{I})\rho - \delta\mathbf{b}_a - \left[{}^b\hat{\mathbf{v}}\times\right]\delta\mathbf{b}_g - \gamma_a - \left[{}^b\hat{\mathbf{v}}\times\right]\gamma_g \end{aligned} \tag{5-26}$$

where we used the property that $\mathbf{g}^b = \mathbf{R}_t^b\mathbf{g}^t = \hat{\mathbf{R}}_t^b(\mathbf{I} - [\rho\times])\mathbf{g}^t$ and $\hat{\mathbf{g}}^b = \hat{\mathbf{R}}_t^b\hat{\mathbf{g}}^t$.

C. Attitude Error Analysis

The attitude error between the body and geographic frames is given in [83, eqn. (12.45)] as:

$$\dot{\rho} = -\hat{\boldsymbol{\Omega}}_{it}^t\rho - \delta{}^t\omega_{it} - \hat{\mathbf{R}}_b^t\delta\mathbf{b}_g - \hat{\mathbf{R}}_b^t\gamma_g \tag{5-27}$$

where $\delta^t\omega_{it}$ is calculated by [83, eqn. (12.44)]

$$\delta^t\omega_{it} = -\omega_{ie} \begin{bmatrix} \sin\bar{\phi} \\ 0 \\ \cos\bar{\phi} \end{bmatrix} \frac{\partial\bar{\phi}}{\partial {}^t\mathbf{T}_{TB}} {}^t\delta\mathbf{T}_{TB} \tag{5-28}$$

D. Error Model

The state of the system is

$$\mathbf{x} = \begin{bmatrix} {}^t\mathbf{T}_{TB} & {}^b\mathbf{v} & \mathbf{\Theta} & \mathbf{b}_a & \mathbf{b}_g \end{bmatrix}^T \tag{5-29}$$

and the input is

$$\mathbf{u} = \begin{bmatrix} \gamma_a & \gamma_g & \omega_{ba} & \omega_{bg} \end{bmatrix}^T \tag{5-30}$$

The error state vector is:

$$\delta\mathbf{x} = \begin{bmatrix} {}^t\delta\mathbf{T}_{TB} & {}^b\delta\mathbf{v} & \rho & \delta\mathbf{b}_a & \delta\mathbf{b}_g \end{bmatrix}^T \tag{5-31}$$

where each delta term is defined by subtracting the computed value from the true value, e.g., ${}^t\delta\mathbf{T}_{TB} = {}^t\mathbf{T}_{TB} - {}^t\hat{\mathbf{T}}_{TB}$.

The state space model of the system is represented in the format $\delta\dot{\mathbf{x}} = \mathbf{F}\delta\mathbf{x} + \mathbf{G}\mathbf{u}$. Based on Eqns. (5-2), (5-4), and (5-25, 5,26, 5-27), the state error model is:

$$\begin{bmatrix} {}^t\delta\dot{\mathbf{T}}_{TB} \\ {}^b\delta\dot{\mathbf{v}} \\ \dot{\rho} \\ \delta\dot{\mathbf{b}}_a \\ \delta\dot{\mathbf{b}}_g \end{bmatrix} = \begin{bmatrix} \mathbf{0} & \hat{\mathbf{R}}_b^t & -[{}^t\hat{\mathbf{v}}\times] & \mathbf{0} & \mathbf{0} \\ \mathbf{F}_{VT} & -\left(\hat{\mathbf{\Omega}}_{ie}^b + \hat{\mathbf{\Omega}}_{ib}^b\right) & \mathbf{F}_{V\rho} & -\mathbf{I} & -[{}^b\hat{\mathbf{v}}\times] \\ \mathbf{F}_{\rho T} & \mathbf{0} & -\hat{\mathbf{\Omega}}_{it}^t & \mathbf{0} & -\hat{\mathbf{R}}_b^t \\ \mathbf{0} & \mathbf{0} & \mathbf{0} & \mathbf{0} & \mathbf{0} \\ \mathbf{0} & \mathbf{0} & \mathbf{0} & \mathbf{0} & \mathbf{0} \end{bmatrix} \begin{bmatrix} {}^t\delta\mathbf{T}_{TB} \\ {}^b\delta\mathbf{v} \\ \rho \\ \delta\mathbf{b}_a \\ \delta\mathbf{b}_g \end{bmatrix} + \begin{bmatrix} \mathbf{0} & \mathbf{0} & \mathbf{0} & \mathbf{0} \\ -\mathbf{I} & -[{}^b\hat{\mathbf{v}}\times] & \mathbf{0} & \mathbf{0} \\ \mathbf{0} & -\hat{\mathbf{R}}_b^t & \mathbf{0} & \mathbf{0} \\ \mathbf{0} & \mathbf{0} & \mathbf{I} & \mathbf{0} \\ \mathbf{0} & \mathbf{0} & \mathbf{0} & \mathbf{I} \end{bmatrix} \begin{bmatrix} \gamma_a \\ \gamma_g \\ \omega_{ba} \\ \omega_{bg} \end{bmatrix} \tag{5-32}$$

where

$$\mathbf{F}_{VT} = \hat{\mathbf{R}}_t^b \left(\frac{\partial \mathbf{g}^t}{\partial {}^t\mathbf{T}_{TB}} + [{}^t\hat{\mathbf{v}}\times] \frac{\partial {}^t\omega_{ie}}{\partial {}^t\mathbf{T}_{TB}} \right) |_{{}^t\hat{\mathbf{T}}_{TB}} \approx \hat{\mathbf{R}}_t^b \frac{\partial \mathbf{g}^t}{\partial {}^t\mathbf{T}_{TB}} |_{{}^t\hat{\mathbf{T}}_{TB}}$$

$$\mathbf{F}_{V\rho} = \hat{\mathbf{R}}_t^b ([\mathbf{g}^t\times] + {}^t\omega_{ie}({}^t\hat{\mathbf{v}})^T - ({}^t\omega_{ie})^T {}^t\hat{\mathbf{v}}\mathbf{I}) \approx \hat{\mathbf{R}}_t^b [\mathbf{g}^t\times]$$

and

$$\mathbf{F}_{\rho T} = \omega_{ie} \begin{bmatrix} \sin\bar{\phi} \\ 0 \\ \cos\bar{\phi} \end{bmatrix} \frac{\partial \bar{\phi}}{\partial {}^t\mathbf{T}_{TB}} \approx 0.$$

5.4 Aiding Sensor Model

LIDAR sensor is able to detect three types of landmarks: point features, arc features, and line features. The aiding sensor models for both types of feature detection are discussed in this section.

5.4.1 Point Feature Detection

The LIDAR scans a single laser beam through 180°. Therefore, a post which will be modeled as a vertical line in the tangent frame will appear as a point in the LIDAR frame. Similarly, a wall which will be modeled as a vertical plane in the tangent frame will appear as a line in the LIDAR frame of references. The LIDAR position and point feature detection are shown in Fig. 5-1.

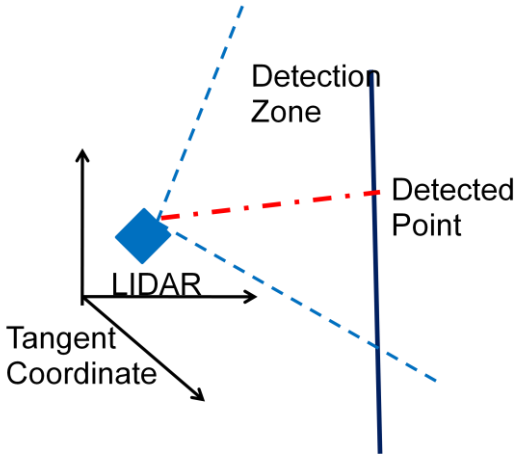


Figure 5-1. The LIDAR position and point feature detection.

Consider a mapped vertical post at P_0. The post will appear as a point in the LIDAR return. The vector from T to P_0 in tangent frame is ${}^{t}\mathbf{T}_{TP_0} = [N_0 \quad E_0 \quad 0]^T$, which is known in the survey. The line is denoted as:

$$ {}^{t}\mathbf{T}_{TP}(m) = {}^{t}\mathbf{T}_{TP_0} + m\mathbf{e}_3 \tag{5-33} $$

where ${}^{t}\mathbf{T}_{TP}(m)$ is the vector in tangent frame from T to a point P at height m on the post, m is a scalar with $m \in (0, +\infty)$. The vector $\mathbf{e}_3$ is parallel to the post. In our

application, all the posts and planes are modeled as vertical. For a vertical post, $\mathbf{e}_3 = [0 \;\; 0 \;\; 1]^T$ in the tangent frame.

For any feature point P, the vector from LIDAR to the point is:

$$\mathbf{T}_{LP} = \mathbf{T}_{TP} - (\mathbf{T}_{TB} + \mathbf{T}_{BL}) \tag{5-34}$$

as is shown in Fig. 5-2. Eqn. (5-34) is valid for all the frame of references.

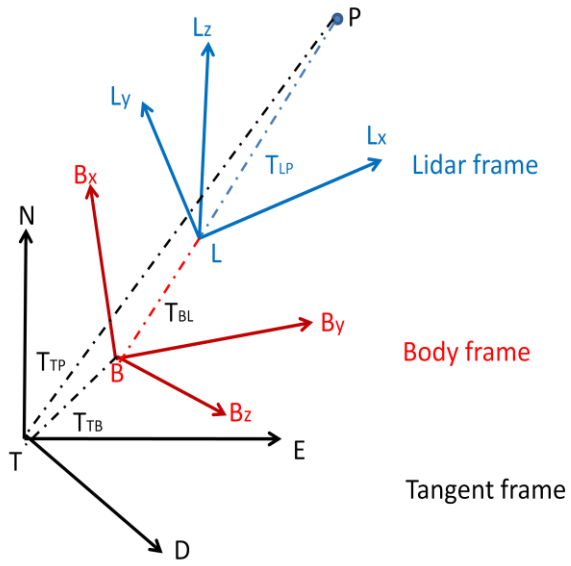


Figure 5-2. The LIDAR, body and tangent frames as well as the feature point P.

In Eqn. (5-34), $\mathbf{T}_{TP}$ is a constant in tangent frame known from the feature survey. $\mathbf{T}_{TB}$ will be calculated in the tangent frame. $\mathbf{T}_{BL}$ in the body frame is known, which can be determined by the pre-calibration. $\mathbf{R}_b^l$ is the rotation matrix from the body frame to the LIDAR frame, which is determined by the pre-calibration by using the method in

Section 5.2. So the vector from LIDAR to a feature point in LIDAR coordinate is denoted as:

$$ {}^{l}\mathbf{T}_{LP} = \mathbf{R}_b^l(\mathbf{R}_t^b({}^{t}\mathbf{T}_{TP} - {}^{t}\mathbf{T}_{TB}) - {}^{b}\mathbf{T}_{BL}) \tag{5-35} $$

By substituting Eqn (5-33) into (5-35) we have:

$$ {}^{l}\mathbf{T}_{LP}(m) = \mathbf{R}_b^l(\mathbf{R}_t^b({}^{t}\mathbf{T}_{TP_0} + m\mathbf{e}_3 - {}^{t}\mathbf{T}_{TB}) - {}^{b}\mathbf{T}_{BL}) \tag{5-36} $$

which is the vector in the LIDAR coordinate from the origin of the LIDAR sensor to a point with height m on the post. By expanding Eqn. (5-36), the coordinate of ${}^{l}\mathbf{T}_{LP}(m)$ on ${}^{l}z$- direction is:

$$ \begin{aligned} z(m) &= \mathbf{e}_3^T\, {}^{l}\mathbf{T}_{LP}(m) \\ &= m\mathbf{e}_3^T\mathbf{R}_b^l\mathbf{R}_t^b\mathbf{e}_3 + \mathbf{e}_3^T\mathbf{R}_b^l(\mathbf{R}_t^b({}^{t}\mathbf{T}_{TP_0} - {}^{t}\mathbf{T}_{TB}) - {}^{b}\mathbf{T}_{BL}) \end{aligned} \tag{5-37} $$

Eqn. (5-37) is the theoretical equation. Note that for a single-planar LIDAR, the scan plane in LIDAR coordinate is ${}^{l}z = 0$. Therefore, for all the LIDAR points we have ${}^{l}z = 0$. We can use this fact to calculate m as:

$$ m = -\frac{\mathbf{e}_3^T\mathbf{R}_b^l(\mathbf{R}_t^b({}^{t}\mathbf{T}_{TP_0} - {}^{t}\mathbf{T}_{TB}) - {}^{b}\mathbf{T}_{BL})}{\mathbf{e}_3^T\mathbf{R}_b^l\mathbf{R}_t^b\mathbf{e}_3} \tag{5-38} $$

By substituting Eqn. (5-38) into (5-36), the detected point is:

$$ {}^{l}\mathbf{T}_{LP} = \mathbf{R}_b^l\left(\mathbf{R}_t^b\left({}^{t}\mathbf{T}_{TP_0} - \frac{\mathbf{e}_3^T\mathbf{R}_b^l(\mathbf{R}_t^b({}^{t}\mathbf{T}_{TP_0} - {}^{t}\mathbf{T}_{TB}) - {}^{b}\mathbf{T}_{BL})}{\mathbf{e}_3^T\mathbf{R}_b^l\mathbf{R}_t^b\mathbf{e}_3}\mathbf{e}_3 - {}^{t}\mathbf{T}_{TB}\right) - {}^{b}\mathbf{T}_{BL}\right) \tag{5-39} $$

Eqn. (5-39) will be used to predict the LIDAR measurement.

LIDAR sensor provides both range and bearing angles as an output when a point is detected. The range and bearing measurements are given as:

$$\begin{aligned} y_{L1} &= ||{}^{l}\mathbf{T}_{LP}|| + n_{L1} \\ y_{L2} &= \operatorname{atan2}\left(\mathbf{e}_2^T \cdot {}^{l}\mathbf{T}_{LP}, \mathbf{e}_1^T \cdot {}^{l}\mathbf{T}_{LP}\right) + n_{L2} \end{aligned} \tag{5-40}$$

where n_{L1} and n_{L2} are the measurement noise for y_{L1} and y_{L2}, respectively. $\mathbf{e}_1 = [1 \;\; 0 \;\; 0]^T$ and $\mathbf{e}_1 = [0 \;\; 1 \;\; 0]^T$.

Let $\delta\mathbf{y} = \mathbf{H}\delta\mathbf{x}$, $\mathbf{y} = [y_{L1} \;\; y_{L2}]^T$. $\mathbf{H}$ is to be solved for the aiding sensor model using partial derivative. See Appendix A for the calculation of $\mathbf{H}$. For point feature detection,

$$\mathbf{H} = \begin{bmatrix} {}^{l}\vec{\mathbf{T}}_{LP} \\ \frac{\mathbf{u}_1\mathbf{e}_2^T - \mathbf{u}_2\mathbf{e}_1^T}{||{}^{l}\mathbf{T}_{LP}||^2} \end{bmatrix} [\mathbf{F}_{yT} \;\; \mathbf{0} \;\; \mathbf{F}_{y\rho} \;\; \mathbf{0} \;\; \mathbf{0}] \tag{5-41}$$

where

$$\mathbf{F}_{y\rho} = \mathbf{R}_b^l \hat{\mathbf{R}}_t^b \left([{}^{t}\mathbf{T}_{TP0} \times] - \frac{\mathbf{e}_3\left(\mathbf{e}_3^T \mathbf{R}_b^l \hat{\mathbf{R}}_t^b ([{}^{t}\mathbf{T}_{TP0}\times] - [{}^{t}\hat{\mathbf{T}}_{TB}\times])\right)}{\hat{\mathbf{F}}} + \frac{(\hat{\mathbf{B}} - \hat{\mathbf{C}} - \hat{\mathbf{D}})\left(\mathbf{e}_3^T \mathbf{R}_b^l \hat{\mathbf{R}}_t^b [\mathbf{e}_3\times]\right)}{\hat{\mathbf{F}}^2} - \left[\frac{\hat{\mathbf{B}} - \hat{\mathbf{C}} - \hat{\mathbf{D}}}{\hat{\mathbf{F}}} \times\right] - [{}^{t}\hat{\mathbf{T}}_{TB} \times] \right)$$

$$\begin{aligned} \mathbf{F}_{yT} &= \mathbf{R}_b^l \hat{\mathbf{R}}_t^b \left(\frac{\mathbf{e}_3 \left(\mathbf{e}_3^T \mathbf{R}_b^l \hat{\mathbf{R}}_t^b \right)}{\hat{\mathbf{F}}} - \mathbf{I} \right) \\ \hat{\mathbf{B}} &= \mathbf{e}_3^T \mathbf{R}_b^l \hat{\mathbf{R}}_t^b \, {}^{t}\mathbf{T}_{TP_0} \mathbf{e}_3 \\ \hat{\mathbf{C}} &= \mathbf{e}_3^T \mathbf{R}_b^l \hat{\mathbf{R}}_t^b \, {}^{t}\hat{\mathbf{T}}_{TB} \mathbf{e}_3 \\ \hat{\mathbf{D}} &= \mathbf{e}_3^T \mathbf{R}_b^l \, {}^{b}\mathbf{T}_{BL} \mathbf{e}_3 \\ \hat{\mathbf{F}} &= \mathbf{e}_3^T \mathbf{R}_b^l \hat{\mathbf{R}}_t^b \mathbf{e}_3 \\ \mathbf{u}_1 &= \mathbf{e}_1^T \, {}^{l}\hat{\mathbf{T}}_{LP} \\ \mathbf{u}_2 &= \mathbf{e}_2^T \, {}^{l}\hat{\mathbf{T}}_{LP} \end{aligned}$$

5.4.2 Line Feature Detection

For the line feature, the basic detection method is to decompose the data into segments of points, such that each segment can be modeled as a line, or a cluster of lines. The Iterative End Point Fit (IEPF) algorithm has been widely used for line extraction [84]. Here we have an assumption that the data has been processed into segments and the line has been extracted for each segment. The LIDAR position, a wall in the real world, and the line feature are shown in Fig. 5-3.

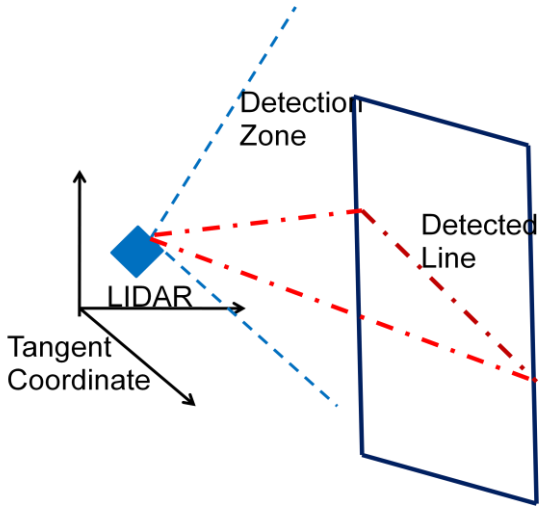


Figure 5-3. The LIDAR position, a wall in the real world, and the line features.

There are several features of the line, such as position, orientation, length, start and end position of the line, etc. Fig. 5-4 shows that it is very difficult to determine the length

and end points of lines, since in some cases only part of the line can be detected. Therefore, in this report we use the expression of the line in the LIDAR frame as features.

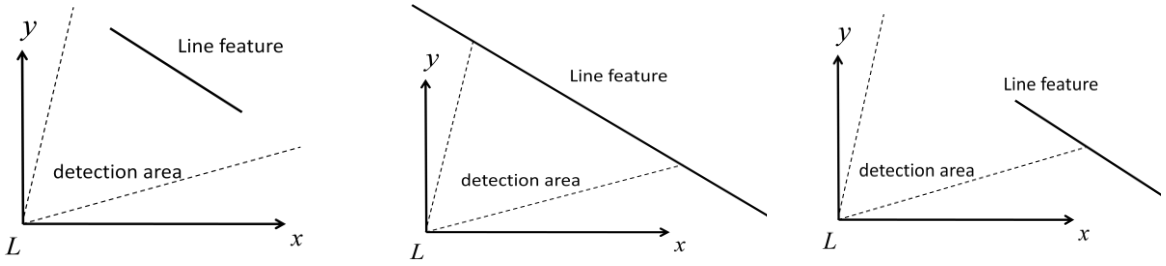


Figure 5-4. Three line detection cases. The line features are represented by solid lines. In (a), the line is covered by the detection area so it can be fully detected. In (b) and (c), part of the line can be covered by the LIDAR detection area. Both ends are out of detection zone in (b). One end can be detected in (c).

A line feature detected by the LIDAR is a plane in the world coordinate. In our application we only model, map, and consider vertical planes. Let P_0 be a point on this plane, and ${}^{t}\mathbf{r}$ be the normal vector of this plane in tangent frame. Point P on this plane has the following property in tangent frame:

$$({}^{t}\mathbf{T}_{TP} - {}^{t}\mathbf{T}_{TP_0})^{T}{}^{t}\mathbf{r} = 0 \tag{5-42}$$

and in LIDAR plane

$$({}^{l}\mathbf{T}_{LP} - {}^{l}\mathbf{T}_{LP_0})^{T}{}^{l}\mathbf{r} = 0 \tag{5-43}$$

where ${}^{l}\mathbf{r}$ is the normal vector in LIDAR frame. From Eqn. (5-35) we know that

$${}^{l}\mathbf{T}_{LP_0} = \mathbf{R}_b^l(\mathbf{R}_t^b({}^{t}\mathbf{T}_{TP_0} - {}^{t}\mathbf{T}_{TB}) - {}^{b}\mathbf{T}_{BL}) \tag{5-44}$$

After rotation from tangent coordinate to LIDAR coordinate, the normal vector $\mathbf{r}$ in LIDAR frame of reference is ${}^{l}\mathbf{r} = \mathbf{R}_b^l\,\mathbf{R}_t^b\,{}^{t}\mathbf{r}$. So the plane in LIDAR coordinate satisfies:

$$\left({}^{b}\mathbf{T}_{LP} - \left(\mathbf{R}_b^l \left(\mathbf{R}_t^b \left({}^{t}\mathbf{T}_{TP_0} - {}^{t}\mathbf{T}_{TB}\right) - {}^{b}\mathbf{T}_{BL}\right)\right)\right)^T \left(\mathbf{R}_b^l \, \mathbf{R}_t^b \, {}^{t}\mathbf{r}\right) = 0 \tag{5-45}$$

The LIDAR detection plane is ${}^{l}z = 0$. By substituting ${}^{l}z = 0$ in Eqn. (5-45) the detected line in LIDAR coordinate is:

$$\begin{cases} \mathbf{e}_1^T(\mathbf{R}_b^l\mathbf{R}_t^b{}^{t}\mathbf{r}){}^{l}x + \mathbf{e}_2^T(\mathbf{R}_b^l\mathbf{R}_t^b{}^{t}\mathbf{r}){}^{l}y - \left(\mathbf{R}_b^l \left(\mathbf{R}_t^b \left({}^{t}\mathbf{T}_{TP_0} - {}^{t}\mathbf{T}_{TB}\right) - {}^{b}\mathbf{T}_{BL}\right)\right)^T \left(\mathbf{R}_b^l \, \mathbf{R}_t^b \, {}^{t}\mathbf{r}\right) = 0 \\ {}^{l}z = 0 \end{cases} \tag{5-46}$$

where $\mathbf{e}_1 = [1 \quad 0 \quad 0]^T$, $\mathbf{e}_2 = [0 \quad 1 \quad 0]^T$, $\mathbf{e}_3 = [0 \quad 0 \quad 1]^T$, ${}^{l}x = \mathbf{e}_1^T{}^{b}\mathbf{T}_{LP}$, ${}^{l}y = \mathbf{e}_2^T{}^{b}\mathbf{T}_{LP}$, ${}^{l}z = \mathbf{e}_3^T{}^{b}\mathbf{T}_{LP}$, and ${}^{b}\mathbf{T}_{LP} = [{}^{l}x \;\; {}^{l}y \;\; {}^{l}z]^T$.

The LIDAR measurements are usually calculated by the following method. In the survey, the line feature as denoted as $ax + by + c = 0$ in the LIDAR frame, where *a*, *b* and *c* are scalars. *d* and α are defined as $d = ||c/\sqrt{a^2 + b^2}||$, and $\alpha = \text{atan2}(b, a)$, where d is the distance from LIDAR origin to the line, and α is the angle between the *x*- axis and the dash line perpendicular to the feature line, as is shown in Fig. 5-5. Then in the LIDAR detection, the point data on the same line will be collected. By using least square fitting method the points are fitted into a straight line as $\hat{a}x + \hat{b}y + \hat{c} = 0$. Then $\hat{d} = ||\hat{c}/\sqrt{\hat{a}^2 + \hat{b}^2}||$ and $\hat{\alpha} = \text{atan2}(\hat{b}, \hat{a})$ will be calculated. $\hat{d}$ is the single measurement used in [41]. In our positioning system $(\hat{d}, \hat{\alpha})$ is considered to be a pair of measurements. With a unique $(\hat{d}, \hat{\alpha})$, the position of the line feature can be determined.

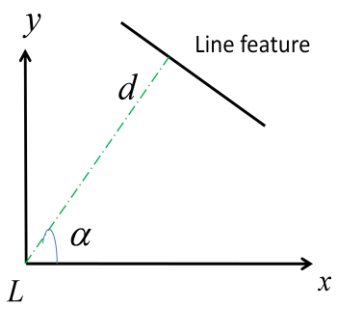


Figure 5-5. Measurement d and α in the line feature detection.

Since d, α are derived from a, b, and c, while $\hat{d}$ and $\hat{\alpha}$ are derived from $\hat{a}$, $\hat{b}$, and $\hat{c}$, instead of using $(\hat{d}, \hat{\alpha})$ we use $(\hat{b}/\hat{a}, \hat{c}/\hat{a})$ as the pair of measurements. $(\hat{b}/\hat{a}, \hat{c}/\hat{a})$ is also able to determine the position of the line feature. Moreover, the calculation of $(\hat{b}/\hat{a}, \hat{c}/\hat{a})$ is more effective and less time-consuming compared with the calculation of $(\hat{d}, \hat{\alpha})$. To the best of our knowledge, this approach is the first method to use $(\hat{b}/\hat{a}, \hat{c}/\hat{a})$ as the measurements.

On the other hand, $(\hat{b}/\hat{a}, \hat{c}/\hat{a})$ is not able to determine the unique position of the line feature if another line feature is an extension of the detected line (which is also true for the $(\hat{d}, \hat{\alpha})$ features). In order to solve this problem, the start point $P_{i,\text{start}}$ and $P_{i,\text{end}}$ of the *i*-th line feature are utilized for identification. If two line features have been detected with the same $(\hat{b}/\hat{a}, \hat{c}/\hat{a})$, the line between $P_{i,\text{start}}$ and $P_{i,\text{end}}$ will be considered to be the *i*-th line feature, as is shown in Fig. 5-6.

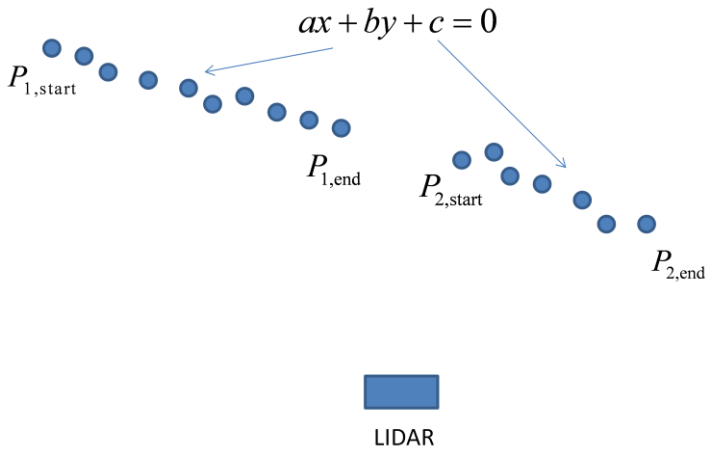


Figure 5-6. Two line features with the same measurement. The start and end points of each line feature are used for identification.

The line feature in LIDAR coordinate is provided in Eqn. (5-46). In our implementation, the measurements are defined as

$$\begin{aligned} y_{L1} &= -\frac{\left(\mathbf{R}_b^l\left(\mathbf{R}_t^b\left({}^t\mathbf{T}_{TP_0} - {}^t\mathbf{T}_{TB}\right) - {}^b\mathbf{T}_{BL}\right)\right)^T\left(\mathbf{R}_b^l\mathbf{R}_t^{bt}\mathbf{r}\right)}{\mathbf{e}_1^T\left(\mathbf{R}_b^l\mathbf{R}_t^{bt}\mathbf{r}\right)} + n_{L1} \\ y_{L2} &= \frac{\mathbf{e}_2^T\left(\mathbf{R}_b^l\mathbf{R}_t^{bt}\mathbf{r}\right)}{\mathbf{e}_1^T\left(\mathbf{R}_b^l\mathbf{R}_t^{bt}\mathbf{r}\right)} + n_{L1} \end{aligned} \tag{5-47}$$

where n_{L1} and n_{L1} are the measurement noise for y_{L1} and y_{L2}, respectively.

Let $\delta\mathbf{y} = \mathbf{H}\delta\mathbf{x}$, $\mathbf{y} = [y_{L1} \;\; y_{L2}]^T$. will be solved using partial derivative (see Appendix B). Here for line feature detection, $\mathbf{H}$ is:

$$\mathbf{H} = \begin{bmatrix} \mathbf{F}_{y_1T} & \mathbf{0} & \mathbf{F}_{y_1\rho} & \mathbf{0} & \mathbf{0} \\ \mathbf{0} & \mathbf{0} & \mathbf{F}_{y_2\rho} & \mathbf{0} & \mathbf{0} \end{bmatrix} \tag{5-48}$$

where

$$\mathbf{F}_{y_1T} = \frac{\left(\mathbf{R}_b^l\hat{\mathbf{R}}_t^{bt}\mathbf{r}\right)^T\left(\mathbf{R}_b^l\hat{\mathbf{R}}_t^b\right)}{\mathbf{G}}$$

$$
\begin{aligned}
\mathbf{F}_{y_1\rho} &= \frac{\left(\mathbf{R}_b^l\hat{\mathbf{R}}_t^{bt}\mathbf{r}\right)^T\left(\mathbf{R}_b^l\hat{\mathbf{R}}_t^b\left(\left[{}^b\hat{\mathbf{T}}_{TB}\times\right]-\left[{}^b\mathbf{T}_{TP_0}\times\right]\right)\right)-\left(\mathbf{R}_b^l\hat{\mathbf{R}}_t^b\left({}^b\mathbf{T}_{TP_0}-{}^b\hat{\mathbf{T}}_{TB}\right)-\mathbf{R}_b^l{}^b\mathbf{T}_{BL}\right)^T\left(\mathbf{R}_b^l\hat{\mathbf{R}}_t^b\left[{}^t\mathbf{r}\times\right]\right)}{\mathbf{G}} \\
&- \frac{K}{\mathbf{G}^2}\mathbf{e}_1^T\mathbf{R}_b^l\hat{\mathbf{R}}_t^b\left[{}^t\mathbf{r}\times\right] \\
\mathbf{F}_{y_2\rho} &= \left(\frac{\mathbf{e}_2^T}{\mathbf{G}}-\frac{\mathbf{J}}{\mathbf{G}^2}\mathbf{e}_1^T\right)\mathbf{R}_b^l\hat{\mathbf{R}}_t^b\left[{}^t\mathbf{r}\times\right] \\
\mathbf{G} &= \mathbf{e}_1^T\left(\mathbf{R}_b^l\hat{\mathbf{R}}_t^{bt}\mathbf{r}\right) \\
\mathbf{J} &= \mathbf{e}_2^T\left(\mathbf{R}_b^l\hat{\mathbf{R}}_t^{bt}\mathbf{r}\right) \\
\mathbf{K} &= -\left(\mathbf{R}_b^l\left(\mathbf{R}_t^b\left({}^b\mathbf{T}_{TP_0}-{}^b\mathbf{T}_{TB}\right)-{}^b\mathbf{T}_{BL}\right)\right)^T\left(\mathbf{R}_b^l\mathbf{R}_t^{bt}\mathbf{r}\right)
\end{aligned}
$$

5.4.3 Arc Feature Detection

A cylinder in the tangent frame will appear as an arc in the LIDAR frame. The cylinder can be a traffic light pole, or a trash can. When the LIDAR is near a cylinder-shaped landmark, the detected feature is an arc; while the LIDAR is far away, the feature will be a point. Therefore, the arc feature and the point feature can be detected by LIDAR sensor from the same landmark. The LIDAR position, a cylinder in the real world, and the arc feature are shown in Fig. 5-7.

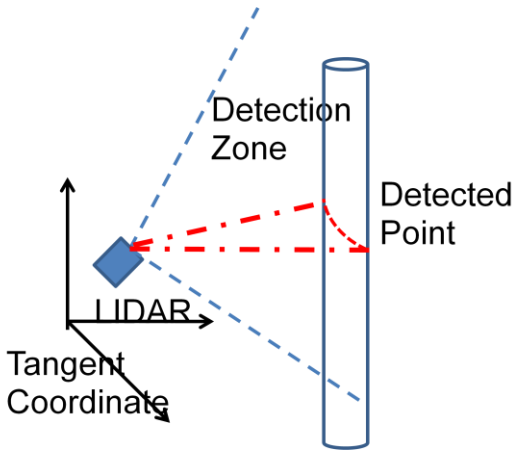


Figure 5-7. LIDAR and arc feature detection.

The center of the arc in the LIDAR coordinate is calculated from the LIDAR scanning points. The measurement of LIDAR sensor is the distance and azimuth to the center point, which is the same as the point feature. Therefore, the sensor model for point feature is also valid for arc feature detection.

5.4.4 Feature Identification

The vehicle navigation is implemented by detecting the landmarks from one frame to the next frame. In order to calculate the position and orientation of the vehicle from frame to frame, the features must be detected in both scans and each feature should have a constant ID. This is done by estimating the position of the feature in the next frame and identifying the detected feature position with the estimated feature locations.

In this system, the transformation between two consecutive frames is estimated by the output of the INS. By mapping the transformation from body frame to LIDAR frame, the relative position of the features in next scan in LIDAR coordinates can be predicted. For point features, the information to be estimated is the position of the point. The information to be predicted for arc feature is the location of the center of the arc. As to the line features, the coordinate of its start and end points will be estimated. After the estimation, a search window is generated around the prediction results to allow for uncertainties.

5.5 Extended Kalman Filter

This section derives the Extended Kalman Filter (EKF) prediction procedures to propagate the estimation error.

5.5.1 System Model in Continuous and Discrete Time Domain

The error model in continuous time domain from Eqn. (5-32) is:

$$\begin{aligned} \delta\dot{\mathbf{x}}(t) &= \mathbf{F}\delta\mathbf{x}(t) + \mathbf{G}\mathbf{u}(t) \\ \delta\mathbf{y}(t) &= \mathbf{H}\delta\mathbf{x}(t) + \mathbf{v}(t) \end{aligned} \tag{5-49}$$

In the discrete time domain, the time propagation model is:

$$\begin{aligned} \delta\mathbf{x}_{k+1} &= \mathbf{\Phi}\delta\mathbf{x}_k + \delta\mathbf{w}_k \\ \mathbf{P}_{k+1}^{-} &= \mathbf{\Phi}\mathbf{P}_k^{+}\mathbf{\Phi}^T + \mathbf{Qd} \end{aligned} \tag{5-50}$$

where $\mathbf{\Phi} = e^{\mathbf{F}T}$ with $||\mathbf{F}T|| \ll 1$, and $\mathbf{Qd} \approx \mathbf{G}\mathbf{Q}\mathbf{G}^T T$. Using the Taylor expansion we have $\mathbf{\Phi} \approx \mathbf{I} + \mathbf{F}T + (\mathbf{F}T)^2/2 \approx \mathbf{I} + \mathbf{F}T$. So from Eqn. (5-32):

$$\begin{aligned} \mathbf{\Phi} &\approx \mathbf{I} + \mathbf{F}T \\ &= \begin{bmatrix} \mathbf{I} & \hat{\mathbf{R}}_b^t T & -\left[{}^t\hat{\mathbf{v}}\times\right] T & \mathbf{0} & \mathbf{0} \\ \mathbf{F}_{VT} T & \mathbf{I} - \left(\hat{\mathbf{\Omega}}_{ie}^b + \hat{\mathbf{\Omega}}_{ib}^b\right) T & \mathbf{F}_{V\rho} T & -\mathbf{T} & -\left[{}^b\hat{\mathbf{v}}\times\right] T \\ \mathbf{F}_{\rho T} T & \mathbf{0} & \mathbf{I} - \hat{\mathbf{\Omega}}_{it}^t T & \mathbf{0} & -\hat{\mathbf{R}}_b^t T \\ \mathbf{0} & \mathbf{0} & \mathbf{0} & \mathbf{I} & \mathbf{0} \\ \mathbf{0} & \mathbf{0} & \mathbf{0} & \mathbf{0} & \mathbf{I} \end{bmatrix} \end{aligned} \tag{5-51}$$

5.5.2 Aided Navigation Process

The aided navigation is implemented in two steps. The first step is time propagation, in which we integrate the high rate sensor output and get the estimation of current state.

In the second step, the low speed sensor measurement is collected to correct the estimation in step 1.

Step 1: Time propagation

1. At time t_k, given $\hat{\mathbf{x}}_k^+$ and $\hat{\mathbf{P}}_k^+$, integrate $\dot{\hat{\mathbf{x}}} = \mathbf{f}(\hat{\mathbf{x}}, \hat{\mathbf{u}})$ over $t \in (t_k, t_{k+1})$ to obtain $\hat{\mathbf{x}}_{k+1}^-$.

2. Calculate $\mathbf{P}_{k+1}^- = \mathbf{\Phi}\mathbf{P}_k^+\mathbf{\Phi}^T + \mathbf{Qd}$ where $\mathbf{Qd} \approx \mathbf{GQG}^T T$, $T = t_{k+1} - t_k$ and $\mathbf{\Phi} = e^{\mathbf{F}T}$.

Step 2: Measurement update

1. At time t_k, predict the measurement $\hat{\mathbf{y}}_{k+1}^- = \mathbf{h}(\hat{\mathbf{x}}_{k+1}^-)$.

2. The measurement residual is $\delta\mathbf{y}_{k+1} = \tilde{\mathbf{y}}_{k+1} - \hat{\mathbf{y}}_{k+1}^-$.

3. The correction gain is $\mathbf{K}_{k+1} = \mathbf{P}_{k+1}^-\mathbf{H}_{k+1}^T \left(\mathbf{H}_{k+1}\mathbf{P}_{k+1}^-\mathbf{H}_{k+1}^T + \mathbf{R}_{k+1}\right)^{-1}$.

4. Correct the state $\mathbf{x}_{k+1}^+ = \hat{\mathbf{x}}_{k+1}^- + \mathbf{K}_{k+1}\delta\mathbf{y}_{k+1}$.

5. Correct the error covariance:

$$\mathbf{P}_{k+1}^+ = \mathbf{K}_{k+1}\mathbf{R}_{k+1}\mathbf{K}_{k+1}^T + (\mathbf{I} - \mathbf{K}_{k+1}\mathbf{H}_{k+1})\mathbf{P}_{k+1}^-(\mathbf{I} - \mathbf{K}_{k+1}\mathbf{H}_{k+1})^{-1}$$

5.6 Summary and Future Work

In this chapter a LIDAR and inertial sensor-based vehicle positioning approach is proposed. First of all, the high speed as well as low speed sensor models were introduced.

A calibration method between the LIDAR and the inertial sensors has also been presented in this chapter. The LIDAR sensor is supposed to detect several types of features: the point feature, the arc feature and the line feature. Finally, an EKF is utilized to estimate the position and orientation of the vehicle. Further field tests will be implemented to test the performance of the calibration method as well as the vehicle positioning approach.

Chapter 6

Multi-Vehicle Tracking Based on Feature Detection and Color Probability Model

One important capability that is critical for traffic surveillance systems is the ability to track vehicles from a video stream. Current vehicle tracking systems have been described in Chapter 2. However, none of the current approaches work well when significant occlusions occur.

In this chapter, a new vehicle tracking framework is proposed, which is based on both feature tracking and a color probability model. The general approach is to estimate the target motion by detecting and tracking the corner features. The tracking result provides an initial guess of the target location. Subsequently, the occlusion is detected, and a probability model is used to re-estimate the target position by only considering the pixels that lie in the occluded area. Our approach is more accurate yet efficient, since the time-consuming probability calculation is called only for cases of occlusion.

This chapter is organized as follows: a description of the vehicle tracking system is given in Section 6.1, followed by the probability model in Section 6.2. Experimental re-

sults are provided in Section 6.3. Finally, we present the conclusion and future work in Section 6.4.

6.1 Overview of the Vehicle Tracking System

Vehicle motion tracking is one of the fundamental components of a traffic surveillance system. With a video camera mounted above the freeway, the foreground image changes rapidly and the background makes little change. Therefore, it is easy to extract the foreground from the background. Then the vehicles on the freeways can be detected and tracked. The flow of this tracking system is shown in Figure 6-1.

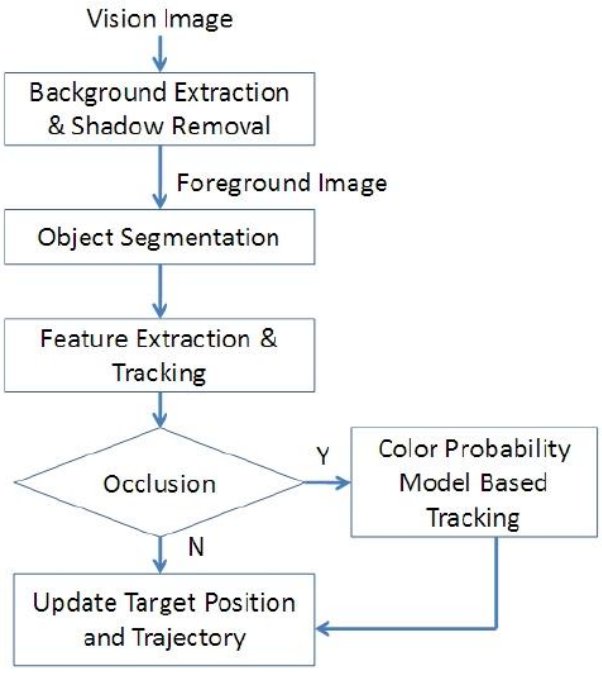


Figure 6-1. Block diagram of the vehicle tracking system.

6.1.1 Background Extraction

For a given video image series, the gray scale of each pixel is modeled by a Gaussian mixture model. For each pixel, the value in the image series is described as a mixture of Gaussian distributions. The probability of current pixel value is given by [70]:

$$p(X) = \sum_{i=1}^{K} \omega_{i,t} \cdot \mathcal{N}(X; \mu_{i,t}, \Sigma_{i,t}) \tag{6-1}$$

where K is the total number of distributions, and $\omega_{i,t}$ is the weight of the i^{th} Gaussian distribution at time t. $\mathcal{N}(X; \mu_{i,t}, \Sigma_{i,t})$ is the i^{th} normal distribution at time t, with its mean value $\mu_{i,t}$ and variance $\Sigma_{i,t}$. Eqn. (6-1) defines the distribution of values of each pixel. A new pixel value represented by one of the major components of the model will be used to update the mixture model.

In the background extraction process, the parameters $\omega_{i,t}$, $\mu_{i,t}$ and $\Sigma_{i,t}$ are to be determined. It was proposed in [70] that the first B distribution of the mixture model best represents the background process. Here B is defined as:

$$B = \arg\min_{K_b}(\Sigma_{k=1}^{K_b}\omega_k > T) \tag{6-2}$$

where T is a constant to determine the minimum portion of the background in the scene.

When the background has been extracted from the video stream, foreground can be obtained by subtract the background from the image.

6.1.2 Shadow Removal

One of the challenges is shadow identification, which moves along with the vehicle. Shadows may cause detection errors when the shadow pixels are segmented and grouped as foreground pixels. A very simple method is used in our implementation to remove the shadows.

Suppose a pixel in the image at position (x, y) is denoted as $I(x, y)$, and the pixel in the background is $B(x, y)$. The absolute difference $|I(x, y) - B(x, y)|$ is considered to be the foreground, which in fact consists of both the moving vehicles and their shadow. In the shadow removal process, the pixel difference is compared with the shadow value threshold as:

$$F(x,y) = \begin{cases} 0 & T_1 \le |I(x,y) - B(x,y)| \le T_2 \\ |I(x,y) - B(x,y)| & \text{otherwise} \end{cases} \quad (6\text{-}3)$$

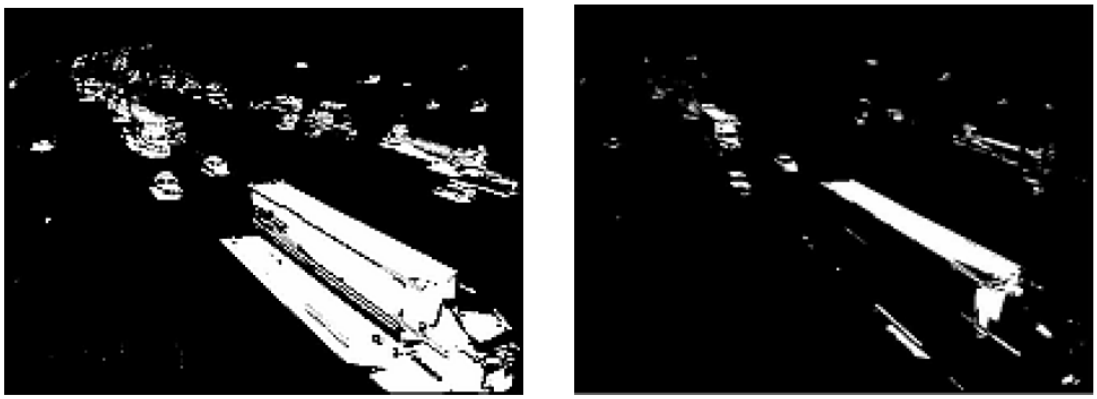
where T_1 and T_2 are the lower and upper threshold defined by the user. If the difference is between the upper and lower bound, this pixel is a shadow pixel. The masks before and after shadow removal are shown in Figure 6-2.

For the foreground extraction, we follow a standard procedure:

(1) Threshold the difference between foreground and background,

(2) Apply the morphological operation (erosion and dilation), and

(3) Remove small regions, fill in the connected components, and detect the object blobs.



(a) (b)

Figure 6-2. Foreground (a) before, and (b) after the shadow removal.

Now the foreground is ready for vehicle detection.

6.1.3 Feature Detection and Tracking

The corner points in regions with rich enough texture are extracted as features. The features are detected and tracked by applying the Kanade-Lucas-Tomasi (KLT) feature detection and tracking method [26, 71]. Lucas and Kanade developed a feature detection method by finding the eigenvalues of the local sums of the weighted gradients [71]. The windowed second moment matrix is calculated by averaging a 2×2 matrix in a spatial window. Suppose the windowed area is $S(p)$, and the image gradient matrix is ∇I, then the second moment matrix in $S(p)$ is $\nabla I \cdot \nabla I^T$. If both the eigenvalues of $\nabla I \cdot \nabla I^T$

are larger than a threshold, the point is assumed to be a feature point, i.e., a 'trackable' point. Here the threshold is defined by the user. In our implementation, the eigenvalue threshold is set as $0.05 \times \max(\min(\lambda_1, \lambda_2)_{x,y})$, where $\min(\lambda_1, \lambda_2)_{x,y}$ is the minimum of eigenvalues λ_1 and λ_2 at point (x, y). Tomasi and Kanade used the same techniques to track the positions of the feature points in the following frame [26].

The feature detection and tracking are implemented in every frame. A feature point is considered to be 'valid' only when it has been tracked in a couple of frames. A feature that does not overlap with the tracking result of an existing feature is considered to be a new one, and will be tracked in the next frame.

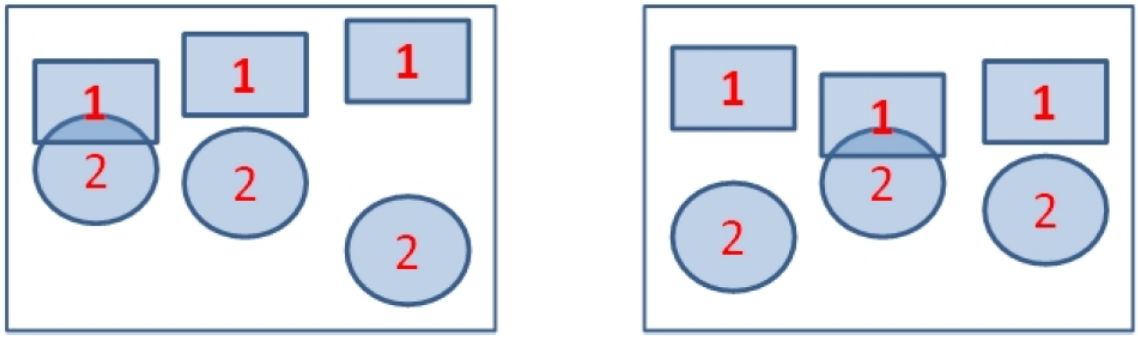
In order to detect multiple targets, the connected components in the foreground have been extracted and considered as blobs. Suppose for object O, there are n features $f_1^k, f_2^k, \ldots, f_n^k$ in the k-th frame. These features are detected inside blob B_i^k. Here B_i^k is the contour of object O in frame k. The corresponding features tracked in frame $k+1$ are $f_1^{k+1}, f_2^{k+1}, \ldots, f_n^{k+1}$. So the blob B_i^{k+1} that covers the features $f_1^{k+1}, f_2^{k+1}, \ldots, f_n^{k+1}$ is considered to be target O in the $k+1$ frame.

In the current vehicle tracking methods, corner features are capable of representing the target. Thus they are sufficient for vehicle tracking. However, in high density traffic

features may be obstructed by the neighboring vehicles. In the following subsection we will discuss how to detect the occlusion situation.

6.1.4 Occlusion Detection

The targets are represented as blobs. Each blob may represent one real-world object, or multiple objects. Occlusions are detected by considering the overlap of multiple objects on one blob. There are two cases of occlusions, as is shown in Figure 6-3.



(a) (b)

Figure 6-3. wo occlusion cases. The first case (a) is that when the blob appears, it is an overlap of two objects. In this case, the two targets will be tracked when they are separated. The second case (b) is that the two targets are two separated blobs in the first frame that they show up. In this case, the blob which is the overlap of the two targets will be split into two objects by the tracking algorithm.

The first case is shown in Figure 6-3(a). Each new detected blob is considered to be a single target. If the blob shows up as an overlap of two targets, we cannot tell whether it is a single object or multiple targets until the blob is split into two objects. The two ob-

jects are tracked independently only when they are separated. The first case will not be discussed in this dissertation.

Figure 6-3(b) illustrates the second case, in which the two targets appear to be two separate blobs in the first frame. In this case, we are able to detect the occlusion, and split the merged blob into two objects.

The association of object and blob has been introduced in Section 6.1.3. Here the object position is estimated by feature tracking. If multiple objects are associated with one blob, then the occlusion is detected. This paper proposed a color probability model to solve the occlusion problem, which is introduced in Section 6.2.

6.2 Color Probability Model

In order to solve the occlusion problem, two representations are used to demonstrate the property of the target. One is the feature-contour model, which generates a hypothesis of the target position in the overlapped blob. The other is a color representation of the target.

The combination of both corner features and color models has the following benefits. The corner features provide texture information, while the color representation gives a

full object mask. In occluded cases, the feature points are used to provide an estimation of the target position. Moreover, the color model estimates the probability of the occluded region associated with the target label. When the two models are combined together, they provide a robust joint probability to assign each pixel to a particular object.

6.2.1 Feature-Contour Model

Jie Yu *et al.* proposed a corner-center model [72], in which a spatial vector is defined as the distance from the feature points to the center. This vector does not vary with the target's position or orientation. However, in our implementation we found out that: 1) the feature points are not evenly distributed in the blob. They may concentrate in a small region. So the distance between the features and the center cannot represent the shape of the blob. 2) The distance vector changes with the target's position and orientation.

In this dissertation we propose a feature-contour model. In our implementation, each blob is represented by an ellipse. A set of features within the target i are denoted as $\{\mathbf{F}_{i,1}^k, \mathbf{F}_{i,2}^k, ...\mathbf{F}_{i,n}^k\} = \{(x_{i,1}^k, y_{i,1}^k), ..., (x_{i,n}^k, y_{i,n}^k)\}$, where $(x_{i,j}^k, y_{i,j}^k)$ is the coordinate of feature point $\mathbf{F}_{i,j}^k$ in the k-th frame. The center of the ellipse is $\mathbf{P}_i^k = (p_{x,i}^k, p_{y,i}^k)$. The axes length of the ellipse on $x-$ and $y-$ directions are $E_{x,i}^k$ and $E_{y,i}^k$, respectively. In the $(k+1)$ -th frame, the feature point position is updated as

$\{\mathbf{F}_{i,1}^{k+1}, ..., \mathbf{F}_{i,n}^{k+1}\} = \{(x_{i,1}^{k+1}, y_{i,1}^{k+1}), ..., (x_{i,n}^{k+1}, y_{i,n}^{k+1})\}$. We define $\delta x = \sum_{j=1}^{n}(x_{i,j}^{k+1} - x_{i,j}^{k})/n$ and $\delta y = \sum_{j=1}^{n}(y_{i,j}^{k+1} - y_{i,j}^{k})/n$. So the center of the target i in frame $k+1$ is estimated as $\mathbf{P}_i^{k+1} = (p_{x,i}^{k} + \delta x, p_{y,i}^{k} + \delta y)$. The two axes of the ellipse are $E_{x,i}^{k+1} = E_{x,i}^{k} - [\max(x_{i,j}^{k+1} - x_{i,j}^{k}) - \min(x_{i,j}^{k+1} - x_{i,j}^{k})]$ and $E_{y,i}^{k+1} = E_{y,i}^{k} - [\max(y_{i,j}^{k+1} - y_{i,j}^{k}) - \min(y_{i,j}^{k+1} - y_{i,j}^{k})]$. From $\mathbf{P}_i^{k+1}$, $E_{x,i}^{k+1}$ and $E_{y,i}^{k+1}$, a hypothesis of the target position in frame $k+1$ is generated. Figure 6-4 explains the position estimation.

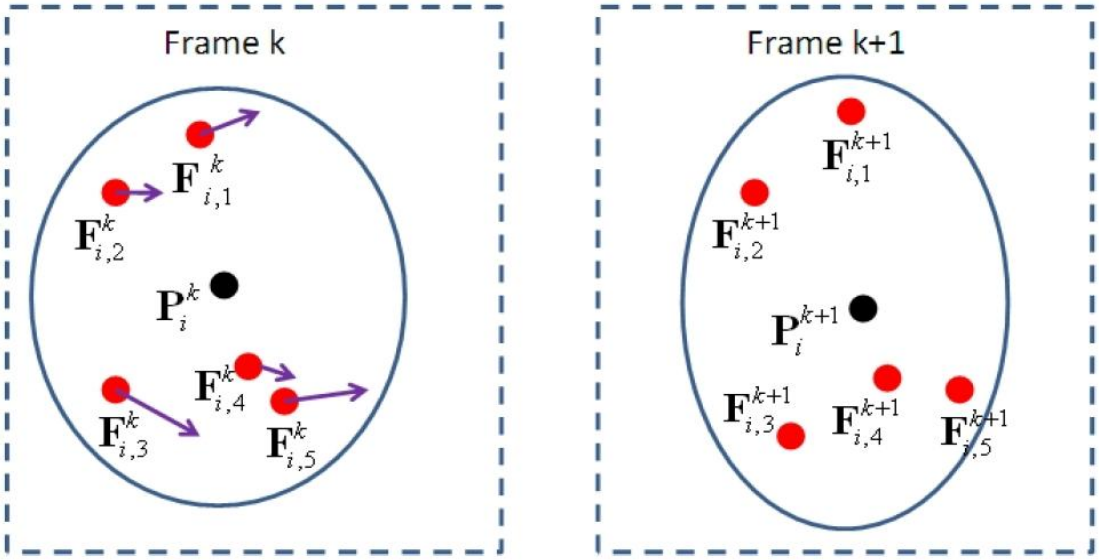


Figure 6-4. The feature-contour positions in frame *k* and *k*+1. Here the arrows represent the feature shift from frame *k* to frame *k*+1.

6.2.2 Color Model

Color histograms have become extremely popular to describe a large image region. Since it does not vary much to target translation, rotation, or the target scale variation, it

has been used in many tracking applications. Color histogram describes the color distribution in a given region, which is robust against partial occlusions. However, color histogram fails to provide spatial information of the object being modeled. It is also susceptible to illumination changes. In this chapter, the color model is used to estimate the target label while the position hypothesis has been established by the feature-contour model. Therefore, the color histogram works in a limited region.

In our implementation, the color model of mean-shift tracking algorithm is used to represent the histograms. Let $\{\mathbf{x}_i\}_{i=1,...n}$ be the pixel positions normalized to range [-1, 1]. We found out that the peripheral pixels are the least reliable in the occlusion region. So the probability of color $\mathbf{C}$ in the target is modeled as a kernel profile, which assigns a larger weight to the pixels close to the center and smaller weights to the peripheral locations [27]. The robustness of the estimation is increased by the weighted algorithm, since the largest weights are allocated to the center pixels, i.e., the most reliable ones. Variable b associates the pixel at location $\{\mathbf{x}_i\}$ to the index $b(\mathbf{x}_i)$ of the histogram bin, which corresponds to the color of that pixel. The probability of the color $\mathbf{C}$ in the target is derived as [27]:

$$p_{\mathbf{C}} = C \sum_{i=1}^{n} k(||\mathbf{x}_i||^2)\delta[b(\mathbf{x}_i) - \mathbf{C}] \tag{6-4}$$

where δ is the Kronecker delta function. The constant C is to normalize the probability $p_{\mathbf{C}}$, which is computed as [27]:

$$C = \frac{1}{\sum_{i=1}^{n} k(||\mathbf{x}_i||^2)} \tag{6-5}$$

since $\sum_{\mathbf{C}} p_{\mathbf{C}} = 1$. k is the profile of a kernel K, $K(\mathbf{x}) = k(||\mathbf{x}||^2)$. There are usually two kinds of kernel profiles. One is the multivariate Epanechnikov profile, the other is the normal profile. In our implementation, the Epanechnikov profile is used, which is defined as [73, pp.139]:

$$k_E(x) = \begin{cases} \frac{1}{2} c_d^{-1} (d+2)(1-x) & x < 1 \\ 0 & \text{otherwise} \end{cases} \tag{6-6}$$

where $d = 2$. c_d is the volume of the unit d-dimensional sphere.

6.2.3 Color Probability Tracking Model

Occlusion is caused by two or more objects. Their contours are merged as one foreground blob. The tracking model is to classify which object (among all the objects that participate in the occlusion) each pixel belongs to. In order to reduce the calculation amount and promote the speed, the occlusion blob is divided into multiple small patches. So our model estimates the posterior probability of each patch belonging to a particular target.

We found that although the whole blob is overlapped by two or more targets, some of the patches are covered by only one object. Therefore, the patches in the blob can be divided into two categories: 'non-occluded patches' and 'occluded patches'. A hypothesis generation algorithm of the target position has been proposed in Section 6.2.1. Then each patch is labeled as 'non-occluded' or 'occluded' (see Figure 6-5).

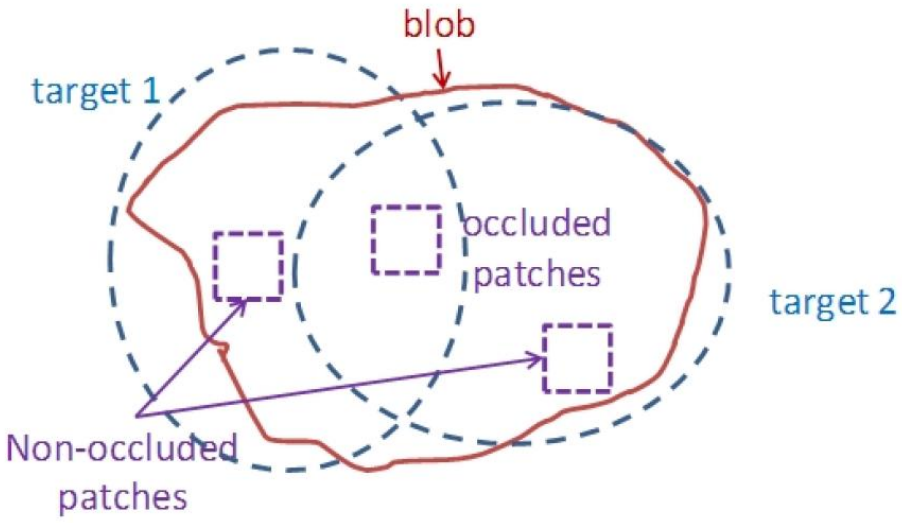


Figure 6-5. The overlap blob, hypothesis of two target positions, and the non-occluded as well as occluded patches.

The non-occluded patches are covered by only one target. Therefore, when a patch is defined to be non-occluded, the object it belongs to is determined. In the following sub-sections the probability of occluded patches will be discussed.

Suppose there are n objects participating in the occlusion, marked as $O_1, O_2, ..., O_n$. For each patch, we have two observations: position and color. The coordinate of the patch

center is denoted as $\mathbf{P}$. The color in this patch is $\mathbf{C}$. So the probability of one patch belonging to object O_i is $p(O_i|\mathbf{P},\mathbf{C})$. By using the Bayesian inference, we have:

$$p(O_i|\mathbf{P},\mathbf{C}) = \frac{p(\mathbf{C}|O_i,\mathbf{P})p(O_i|\mathbf{P})}{p(\mathbf{C}|\mathbf{P})} \tag{6-7}$$

where $p(\mathbf{C}|\mathbf{P})$ is the probability of color observation at position $\mathbf{P}$ in the image. With known position and color, the value of $p(\mathbf{C}|\mathbf{P})$ has no effect on the final object label determination. So the probability in Eqn. (6-7) is factorized as:

$$p(O_i|\mathbf{P},\mathbf{C}) \propto p(\mathbf{C}|O_i,\mathbf{P})p(O_i|\mathbf{P}) \tag{6-8}$$

Now the object probability for each patch is defined by two factors: the probability of object label given position, i.e., $p(O_i|\mathbf{P})$; and the probability of color given both object label and position, i.e., $p(\mathbf{C}|O_i,\mathbf{P})$.

A. The Probability of Object Label Given Position $p(O_i|\mathbf{P})$

Let the occluded patch being covered by n objects. The probability of object label given position is $p(O_i|\mathbf{P}) = 1/n$. In other words, the patch that is covered by objects gets a likelihood of $1/n$ for each of the objects. In the vehicle tracking application, we usually have $n = 2$, which means that the occluded patch is often covered by two vehicles. So $p(O_i|\mathbf{P}) = 1/2$ is assumed to be a constant. Eqn. (6-8) is simplified as:

$$p(O_i|\mathbf{P},\mathbf{C}) \propto p(\mathbf{C}|O_i,\mathbf{P}) \tag{6-9}$$

which means that the object label only depends on the probability of color given both object label and position.

B. The Probability of Object Color Given Position and Label $p(\mathbf{C}|O_i, \mathbf{P})$

In frame k the color distribution probability is written as $p_k(\mathbf{C}|O_i, \mathbf{P}_k)$, which is the color distribution of object O_i at image coordinate $\mathbf{P}_k$. Let the original position of the occluded patch in frame $k-1$ be $\mathbf{P}_{k-1}$. The color observation in the occluded region is defined by:

$$p_k(\mathbf{C}|O_i, \mathbf{P}_k) = p_{k-1}(\mathbf{C}|O_i, \mathbf{P}_{k-1}) \tag{6-10}$$

where $\mathbf{P}_{k-1}$ can be calculated by the corner feather shift from frame $k-1$ to k. Note that the color distribution represents the global information. Without loss of generality, the original position of the occluded region (which was occluded in frame k) in frame $k-1$ is considered to have the same color distribution. In another word, the region is occluded in frame k, and the color probability is calculated in frame $k-1$. Suppose $\mathbf{\Phi}_{i,k-1}$ is the original location of the occluded region of target O_i in the $(k-1)$-th frame. From Eqn. (6-10) we have:

$$p_k(\mathbf{C}|O_i, \mathbf{P}_k) = p_{k-1}(\mathbf{C}|O_i, \mathbf{\Phi}_{i,k-1}) = p_{k-1}(\mathbf{C}|\mathbf{\Phi}_{i,k-1}) \tag{6-11}$$

where $p_{k-1}(\mathbf{C}|\mathbf{\Phi}_{i,k-1})$ is defined by the color model in Eqn. (6-4).

For each of patch in the occlusion region, the probability of the patch belonging to each object is calculated by Eqn. (6-11). Then the patch is assigned to the target with maximum probability. An example is shown in Figure 6-6.

6.3 Experiment Results

To evaluate the performance of our system, a traffic monitoring video was used. This video was captured by a freeway surveillance camera on the top of the 215-N freeway in Southern California. The video is about 80 seconds, consisting of totally 1200 frames. The resolution of this video is 640×480, and the frame rate is 15 fps. The video clip is challenging, with multiple trucks and clusters of vehicles. Some video tracking samples are shown in Figure 6-7.

The performance of our approach is evaluated together with the standard KLT [26] and mean-shift tracking algorithms [27]. The ground truth was obtained by manual labeling. Three aspects are relevant for the performance comparison: the hit rate, false alarm rate, and missing rate. Here the hit rate assesses the ability of the system to detect the target at the beginning, and track the object during frame-wise without losing the position or changing the ID. The false alarm rate illustrates the rate to detect non-target objects as the

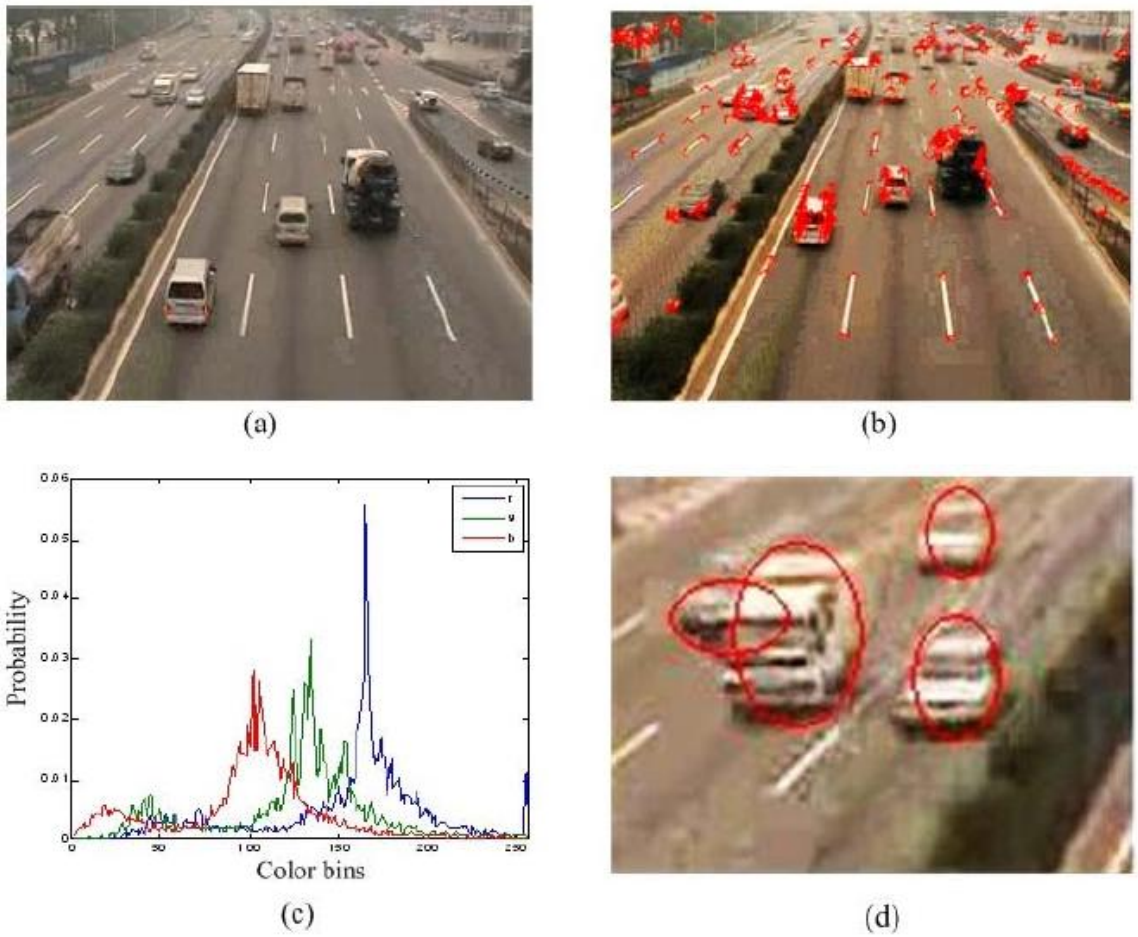


Figure 6-6. An example of the probability tracking model. (a) Original image. (b) Feature points. The vector of the features from current frame to the next frame is shown as arrows. (c) The color probability of the occlusion region. Here the color is divided into 256 bins. (d) The vehicle tracking result in the occlusion area.

the targets. Furthermore, the missing rate is the total number of targets missed by the tracking system vs. the total number of the vehicles. The tracking results are summarized in Table 6-1.

Table 6-1 shows that our approach is a significant improvement compared with the KLT and mean-shift tracking algorithms. The hit rate of our approach is 93.55%, which

shows that the method we proposed outperforms KLT and mean-shift tracking. Both the false alarm rate and the missing rate of KLT are higher than those of the mean-shift tracking. As an improvement of the KLT in occlusion cases, our approach achieves a better false alarm as well as a better missing rate. The false alarm and missing rate of our approach are close to those of mean-shift tracking method. The experiment results illustrate that by combining the features used in KLT and mean-shift tracking methods, the robustness of the tracking performance is increased.

Table 6-1. Vehicle Detection and Tracking Results.

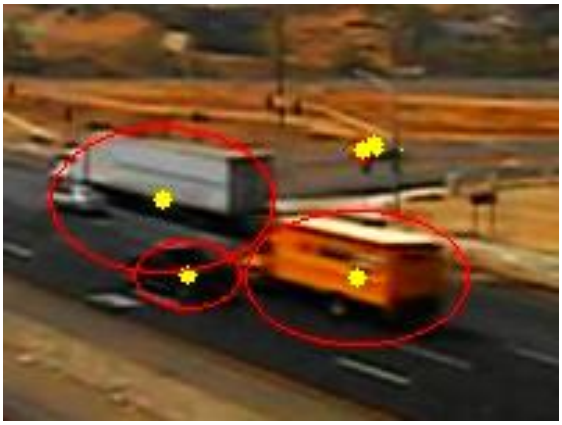
Detection and Tracking Approach	Total Frame Number	Total Vehicle Number (frame-wise)	Hit Rate	False Alarm Rate	Missing Rate
Our Approach	1200	14276	93.55%	3.0%	2.92%
Standard KLT			87.31%	3.02%	3.77%
Mean-shift Tracking			89.53%	2.87%	2.86%

The main false alarm errors and missing errors happen due to the following reasons:

- This method does not work very well in dense crowded regions, where a reasonable background subtraction is not achieved.

- The features are not grouped correctly. The features are grouped as an object by comparing their positions with the blob contours. The feature grouping errors are usually caused by blob segmentation errors.

Frame 107



Frame 300

Frame 642

Frame 654

Frame 764

Frame 1083

Figure 6-7. The vehicle tracking results. They are part of the video frames where occlusion is detected. The ellipses are the position of the targets. The points represent the trajectories.

- Long trucks with multiple trailers are sometimes classified as two objects.
- As is discussed in Section 6.1.4, if the blob is an overlap of two objects in the first frame it appears we cannot correctly split the blob into two separate targets.
- The video stream is unstable due to camera vibration. It can be solved by image stabilization.

6.4 Summary and Discussion

A camera-based vehicle detection and tracking approach is proposed in this chapter. We introduced an integration of the feature tracking and mean-shift tracking in occlusion cases, which can be applied in real-time applications. Promising results were presented with a transportation video clip. This method can handle both the stationary camera and moving camera video streams.

The performance can be further improved by integrating vehicle classification together with vehicle tracking. If the vehicle type is known, size and speed of the vehicle can be constrained. Furthermore, camera calibration provides projection from the image plane to the real world coordinate. By camera calibration the tracking results can be implemented in the real world plane, and vehicle trajectories can be collected.

Chapter 7

Conclusions and Future Work

This dissertation has presented a multi-sensor equipped vehicle navigation system that was developed to specifically obtain the state of surrounding vehicles and its own position. It involves the development of a tightly-coupled LIDAR and computer vision system, the calibration approach of a pair of multi-planar LIDAR sensors and the camera system, the sensor fusion-based vehicle detection and tracking technique, the estimation of the test vehicle's position, and the methodology of vehicle tracking applied in high density traffic surveillance.

This chapter provides a brief summary of the dissertation, as well as the possible future work.

7.1 Summary

Automatic vehicle navigation techniques are becoming an essential part of our daily lives. They open up many potential opportunities but they also come with challenges in terms of sensing capability and accuracy. In this dissertation, we have addressed two problems: *where am I?* and *where are they?*, and demonstrated our approaches to each of the problems in a traffic environment.

The goal of this research is to provide a solution to measure the state of the

surrounding vehicles as well as the test vehicle. The state of the vehicle includes position, orientation, speed, and acceleration. Sensor fusion techniques are utilized to provide a direct measurement of the state. A variety of sensors have been used in this dissertation, including LIDAR, computer vision, as well as an inertial measurement unit. The goal is to quantitatively show that the integration of the sensors will provide a more accurate and effective estimation of the vehicle state. The developed system has successfully met this goal.

The developed multi-planar LIDAR and computer vision sensor calibration approach, as to the author's best knowledge, is the first calibration method for a 'invisible-beam' multi-planar LIDAR and a camera. In comparison to other calibration methods that require a infrared camera to 'see' the LIDAR beams or a special designed calibration shape, this approach is easy to implement with low cost. It has been theoretically and experimentally proven to be able to estimate the geometric relationships between the two sensors.

Based on this unique calibration method, a sensor fusion-based vehicle detection and tracking system was designed and implemented. It consists of three major components: 1) ROIs are generated by the LIDAR sensor; 2) vehicle classification using a computer vision-based Adaboost algorithm, and 3) vehicle position is verified using the output of the LIDAR sensor. A vehicle tracking model is also presented in this dissertation, which

uses a joint probability model-based particle filter to predict the state of the vehicle. The experiment result shows that the designed sensor fusion system achieves higher detection rate and lower positive as well as negative error rates compared with a single sensor-based detection method. Then the relative positions of detected vehicles in the surrounding environment have been represented in the vehicle coordinate to generate a local traffic map.

In addition, a LIDAR and inertial sensor-based vehicle localization approach has been presented in this dissertation. The positioning solution is derived by combining measurements from both LIDAR and inertial sensors, i.e., gyros and accelerometers. Calibration between the two sensors is implemented by the transformation from body coordinate to ECEF coordinate, and from ECEF coordinate to the LIDAR frame of reference. In vehicle positioning system, the angular velocities as well as the accelerations of the vehicle are used to measure the state of the vehicle, while the LIDAR measurements to the landmark structures (posts and surfaces) are considered to be observations. An Extended Kalman Filter is used in position estimation.

Finally, a vehicle tracking technique has been developed for the freeway traffic surveillance system. Vision techniques have been more and more involved in vehicle tracking and counting in the ITS area. However, currently there is no solid solution to track vehicles in high density traffic conditions. The vehicle tracking approach proposed

in this dissertation integrates the corners features and the color histogram probability for vehicle tracking. If occlusion is not detected, feature-based tracking is implemented; otherwise if occlusion occurs, the color probability model is used to determine which object each pixel in the occlusion area belongs to. The experimental results show that the vehicle tracking technology performs well in high density traffic situations.

Taken together, the results in this dissertation demonstrate that a good navigation performance can be achieved using a LIDAR, computer vision, and inertial sensors-based moving platform. Such results are especially important for vehicle navigation systems that are equipped with multiple sensors.

7.2 Future Work

Thus far, only a few experimental LIDAR and computer vision sensor-based vehicle detection data have been collected, within limited time and regions. More data should be collected for Haar training. Further long-term and extensive vehicle tracking videos and LIDAR data will be collected and processed in future research.

In Chapter 5, a LIDAR and inertial sensor calibration method has been proposed. Simulation results are also given in this dissertation. In addition, the performance of the calibration method will be evaluated with field test.

A sensor fusion-based vehicle positioning method has also been discussed in this dissertation. In addition, more experimental data will be collected to evaluate the

performance of the vehicle positioning approach, both on the local road and on the freeways.

Finally, more robust and reliable vehicle tracking techniques are worthy to be explored to provide complete and effective traffic surveillance results.

Bibliography

[1] Young-Kee Jung and Yo-Sung Ho, "Traffic Parameter Extraction Using Video-based Vehicle Tracking," *IEEE/IEEJ/JSAI International Conference on Intelligent Transportation Systems*, pp. 764-769, 1999.

[2] M. Y. Kim, H. Cho and H. Lee, "An Active Trinocular Vision System for Sensing Mobile Robot Navigation Environments," *IEEE Intelligent Robots and Systems Proceedings*, pp. 1698-1703, 2004.

[3] H. Baltzakis, A. Argyros, and P. Trahanias, "Fusion of Laser and Visual Data for Robot Motion Planning and Collision Avoidance," *Machine Vision and Application*, pp. 431-441, 2003.

[4] Y. Liu and R. Emery, "Using EM to Learn 3D Environment Models with Mobile Robots," *Proc. 18th International Conference on Machine Learning*, 2001.

[5] P. G. Michalopoulos, "Vehicle Detection Video Through Image Processing: the Autoscope System," *IEEE Transactions on Vehicular Technology*, vol. 40(1), pp. 21–29, Feb 1991.

[6] H. S. Mahmassani, C. Haas, S. Zhou, and J. Peterman, "Evaluation of Incident Detection Methodologies," *Technical Report FHWA/TX-00/1795-1*, Center of Transportation Research, The University of Texas at Austin, Oct. 1998.

[7] I. J. Cox and G. T. Wilfong, *Autonomous Robot Vehicles*, Springer-Verlag, New York, USA, September 1990.

[8] Ingemar J. Cox, "Blanche–an Experiment in Guidance and Navigation of an Autonomous Robot Vehicle," *IEEE Transactions on Robotics and Automation*, vol. 7(2), pp. 193–204, April 1991.

[9] Johann Borenstein and Liqiang Feng, "UMBmark: A Benchmark Test for Measuring Odometry Errors in Mobile Robots," *Proceedings of the 1995 SPIE Conference on Mobile Robots*, pp. 113–124, October 1995.

[10] Minnesota department of transportation, office of traffic engineering/its section, "Evaluation of Non-intrusive Technologies for Traffic Detection," Oct. 2001.

[11] DARPA Urban Challenge, http://www.darpa.mil/grandchallenge/index.asp.

[12] University of Central Florida, I2Lab, "TeamUCF – DARPA Urban Challenge Technical Report," Jun. 2007.

[13] Sebastian Thrun, "Learning Occupancy Grid Maps with Forward Sensor Models," *Autonomous Robots*, vol. 15(2), pp.111–127, Sep. 2003.

[14] W. R. Whittaker, "Red Team DARPA Grand Challenge 2005 Technical Paper," Carnegie Mellon

University, *CSC Technical Report*, August 2005.

[15] Olli Jokinen, "Self-Calibration of a Light Striping System by Matching Multiple 3-D Profile Maps," *Proc. Second International Conference on 3-D Digital Imaging and Modeling*, pp. 180-190, 1999.

[16] Mitsuhiro Hayashibe and Yoshihiko Nakamura, "Laser-Pointing Endoscope System for Intra-Operative 3D Geometric Registration," *Proc. International Conference on Robotics and Automation*, pp. 1543-1548, May 2001.

[17] Mirko Mahlisch, Roland Schweiger, Werner Ritter and Klaus Dietmayer, "Sensorfusion Using Spatio-Temporal Aligned Video and Lidar for Improved Vehicle Detection," *Proc. of Intelligent Vehicles Symposium*, pp. 424-429, Jun. 2006.

[18] S. Wasielewski and O. Strauss, "Calibration of a Multi-sensor System Laser Rangefinder / Camera," *Proc. of Intelligent Vehicles Symposium*, pp. 472-477, 1995.

[19] R. Unnikrishnan and M. Hebert, "Fast Extrinsic Calibration of a Laser Rangefinder to a Camera," *Tech. report CMU-RI-TR-05-09, Robotics Institute*, Carnegie Mellon University, July, 2005.

[20] D. Cobzas, H. Zhang and M. J¨agersand, "A Comparative Analysis of Geometric and Image-based Volumetric and Intensity Data Registration Algorithms," *IEEE Intl. Conference on Robotics and Automation (ICRA)*, 2002.

[21] Qilong Zhang and Robert Pless, "Extrinsic Calibration of a Camera and Laser Range Finder (Improves Camera Calibration)," *IEEE Proceedings of International Conference on Intelligent Robots and Systems*, pp. 2301-2306, Sep. 2004.

[22] Benjamin Coifman, David Beymer, Philip McLauchlan and Jitendra Malik, "A Real-time Computer Vision System for Vehicle Tracking and Traffic Surveillance," *Transportation Research Part C 6*, pp. 271-288, 1998.

[23] Matthew Barth and Kanok Boriboonsomsin, "Traffic Congestion and Greenhouse Gases," *ACCESS Magazine*, pp.2-9, 2009.

[24] Schmid, C., Mohr, R. & Bauckhage, C., "Evaluation of Interest Point Detectors," *International Journal of Computer Vision*, vol. 37(2), pp. 151-172, 2000.

[25] W. Forstner and E. Gulch, "A Fast Operator for Detection and Precise Location of Distinct Points, Corners, and Centers of Circular Features," *Proceedings Intercommision Conference on Fast Processing of Photogrammetric Data*, pp. 281-305, 1987.

[26] Carlo Tomasi and Tekeo kanade, "Detection and Tracking of Point Features," *Technical Report CMU-CS-91-132*.

[27] Comaniciu, D., Ramesh, V. and Meer, P., "Real-time Tracking of Non-rigid Objects Using Mean

Shift," *IEEE Computer Conference on Computer Vision and Pattern Recognition*, vol. 2, pp. 142-149, 2000.

[28] Deriche, R. & Faugeras, O., "Tracking Line Segments," *Image and Vision Computing*, vol. 8(4), pp. 261-270, 1991.

[29] Koller, D., Daniilidis, K. and Nagel, H. H., "Model-based Object Tracking in Monocular Image Sequences of Road Traffic Scenes," *International Journal of Computer Vision*, vol.10(3), pp. 257-281, 1993.

[30] Kass, M., Witkin, A. and Terzopoulos, D., "Snakes: Active Contour Models," *International Journal of Computer Vision,* pp. 321-331, 1988.

[31] Ronfard, R., "Region-based Strategies for Active Contour Models," *International Journal of Conputer Vision,* vol. 13(2), pp. 229-251, 1994.

[32] Wren, C., Azarbayejani, A., Darrell, T. and Pentland, A., "Pfinder: Real-time Tracking of the Human Body," *IEEE Transactions on Pattern Analysis and Machine Intelligence*, vol. 19(7), pp. 780-785, 1997.

[33] Stauffer, C. and Grimson, W, "Learning Patterns of Activity Using Real-time Tracking," *IEEE Transactions on Pattern Analysis and Machine Intelligence*, vol. 22(8), pp. 747-757.

[34] Brown, M. Z., Burschka, D. & Hager, G. D, "Advances in Computational Stereo," *IEEE Transactions on Pattern Analysis and Machine Intelligence*, vol. 25(8), pp. 993-1008, 2003..

[35] Derpanis, K., "Characterizing Image Motion," *Technical Report CS-2006-06*, York University, Department of Computer Science, 2006.

[36] Serby, D., Meier, E. K. & Gool, L. V., "Probabilistic Object Tracking Using Multiple Features," *IEEE International Conference on Pattern Recognition,* pp. 184-187, 2000.

[37] Rasmussen, C. & Hager, G. D., "Probabilistic Data Association Methods for Tracking Complex Visual Objects," *IEEE Transactions on Pattern Analysis and Machine Intelligence*, vol. 23(6), pp. 560-576, 2001.

[38] Han, B., Joo, S. W. & Davis, L. S., "Probabilistic Fusion Tracking Using Mixture Kernel-Based Bayesian Filtering," *IEEE International Conference on Computer Vision,* pp. 1-8, 2007.

[39] Davison, A.J., I.D. Reid, N.D. Molton, and O. Stasse, "Monoslam: Real-Time single camera SLAM", *IEEE Transactions of Pattern Analysis and Machine Intelligence*, vol. 29, no. 6, pp. 1052-1067, 2007.

[40] DARPA, "The DARPA Urban Grand Challenge," see http://www.darpa.mil/GRANDCHALLENGE/, January 2009.

[41] A. Soloviev, D. Bates, F. van Graas, "Tight Coupling of Laser Scanner and Inertial Measurements for a Fully Autonomous Relative Navigation Solution," *NAVIGATION: Journal of the Institute of Navigation*, Vol. 53, No. 3, 2007.

[42] A. Soloviev, "Tight Coupling of GPS, Laser Scanner, and Inertial Measurements for Navigation in Urban Environments," *IEEE/ION Position, Location and Navigation*, 2008.

[43] Oskarsson, M. and K. Astrom, "Accurate and Automatic Surveying of Beacon Positions for a Laser Guided Vehicle," *Progress in Industrial Mathematics at ECMI*, 1999.

[44] Vadlamanai, A. K. and M. Uijt de Haag, "Use of Laser Range Scanners for Precise Navigation in Unknown Environments," *Proceedings of the ION GNSS*, 2006.

[45] Joerger, M. and B. Pervan, "Range-Domain Integration of GPS and Laser-scanner Measurements for Outdoor Navigation," *Proceedings of the ION GNSS*, 2006.

[46] Maarten Uijt de Haag, Zhen Zhu, Andrey Soloviev and Frank van Graas, "Tightly-Integrated LADAR/INS algorithm Development to Support Urban Operations," *2nd year project report*, 2008.

[47] M. Joerger, and B. Pervan, "Autonomous Ground Vehicle Navigation Using Integrated GPS and Laser Scanner Measurements," *Proc. ION/IEEE PLANS*, April 2006.

[48] Ali Siadat, Axel Kaske, Siegfried Klausmann, Michel Dufaut and Rene Husson, "An Optimized Segmentation Method for a 2D Laser-Scanner Applied to Mobile Robot Navigation," *3rd IPAC Symposium on Intelligent Components and Instruments for Control Applications*, Jun. 1997.

[49] Jose Guivant, Eduardo Nebot and Stephan Baiker, "Autonomous Navigation and Map Building Using Laser Range Sensors in Outdoor Applications," *Journal of Robotics Systems*, vol. 17, No. 10, pp. 565-583, Oct. 2000.

[50] Steven Scheding, Gamini Dissanayake, Eduardo Mario Nebot and Hugh Durrant-Whyte, "An Experiment in Autonomous Navigation of an Underground Mining Vehicle," *IEEE Transactions on Robotics and Automation*, vol. 15, no. 1, pp. 85-95, 1999.

[51] Lim Chee Wang, Lim Ser Yong and Marcelo H. Ang, "Mobile Robot Localization for Indoor Environment," *SIMTech Technical Report (AT/02/015/MECH)*, 2002.

[52] SICK U.S.A., see http://www.sick.com/us/en-us/home/Pages/Homepage1.aspx.

[53] HOKUYO UXM-30LN LIDAR, see http://www.hokuyo-aut.jp/02sensor/07scanner/uxm_30ln.html.

[54] The Laser Scanner Product Overview, see http://www.ibeoas.com/english/products.asp.

[55] Velodyne HDL-64E LIDAR, http://www.hizook.com/blog/2009/01/04/velodyne-hdl-64e-laser-rangefinder-lidar-pseudo-disassembled.

[56] Premebida C., Monteiro G., Nunes U., and Peixoto, P., "A Lidar and Vision-based Approach for Pedestrian and Vehicle Detection and Tracking," *IEEE Intelligent Transportation Systems Conference*, pp. 1044-1049, 2007.

[57] J. Neira, J.D. Tard, J. Horn and G. Schmidt, "Fusing Range and Intensity Images for Mobile Robot Localization," *IEEE Trans. Robotics and Automation*, Vol. 15, No. 1, pp 76- 84, 1999.

[58] D. Toyh and T. Aach, "Detection and Recognition of Moving Objects Using Statistical Motion Detection and Fourier Descriptors," *12th International Conference on Image Analysis and Processing*, pp. 430-435, 2003.

[59] M. Oren, C. Papageorgiou and P. Sinha, "Pedestrian Detection Using Wavelet Templates," *IEEE Proc. on Computer Vision and Pattern Recognition*, 1997.

[60] B. Fardi, U. Schuenert, and G. Wanielik, "Shape and Motion-Based Pedestrian Detection in Infrared Images: A Multi Sensor Approach," *Proc. IEEE Intelligent Vehicles Symp.*, pp. 18-23, 2005.

[61] S. Milch and M. Behrens, "Pedestrian Detection with Radar and Computer Vision," *Proc. Conf. Progress in Automobile Lighting*, 2001.

[62] D. Linzmeier, M. Skutek, M. Mekhaiel, and K. Dietmayer, "A Pedestrian Detection System Based on Thermopile and Radar Sensor Data Fusion," *Proc. Int'l Conf. Information Fusion*, vol. 2, 2005.

[63] P. Marchal, M. Dehesa, D. Gavrila, M.-M. Meinecke, N. Skellern, and R. Viciguerra, "SAVE-U. Final Report," *Technical report, Information Soc. Technology Programme of the EU*, 2005.

[64] Zhengyou Zhang, "A Flexible New Technique for Camera Calibration," *IEEE Trans. On Pattern Analysis and Machine Intelligence*, pp. 1330-1334, 2000.

[65] Jean-Yves Bouguet, "Camera Calibration Toolbox for Matlab," 2003.

[66] Faugeras, O.D. and Toscani, G., "Camera Calibration for 3D Computer Vision," *Proc. International Workshop on Industrial Applications of Machine Vision and Machine Intelligence*, pp. 240-247, 1987.

[67] Carlos Ricolfe Viala and Antonio Jose Sanchez Salmeron, "Performance Evaluation of Linear Camera Calibration Techniques," *IEEE Proceedings on World Automation Congress*, Vol. 18, pp. 49-54, 2004.

[68] K. Levenberg, "A Method for the Solution of Certain Problems in Least Squares," *Quarterly Applied Math*, pp. 164-168, 1944.

[69] J. More, "The Levenberg-Marquardt Algorithm, Implementation and Theory," In G.A. Watson, editor, *Numerical Analysis, Lecture Notes in Mathematics 630*, Springer-Verlag, 1977.

[70] C. Stauffer and W.E.L. Grimson, "Adaptive Background Mixture Models for Real-time Tracking," *Proceedings of IEEE Conference on Computer Vision and Pattern Recognition*, vol. 2, pp. 246-252, 1999.

[71] Bruce D. Lucas and Takeo Kanade, "An Iterative Image Registration Technique with an Application to Stereo Vision," *International Joint Conference on Artificial Intelligence*, pp. 674-679, 1981.

[72] Jie Yu, Dirk Farin and Hartmut S. Loos, "Multi-cue Based Visual Tracking in Clutter Scenes with

Occlusions," *6th IEEE International Conference on Advanced Video and Signal Based Surveillance*, pp. 158-163, 2009.

[73] D.W. Scott, *Multivariate Density Estimation*, New York: Wiley, 1992.

[74] Paul Viola and Michael J. Jones, "Rapid Object Detection Using a Boosted Cascade of Simple Features," *IEEE Computer Vision and Pattern Recognition Proceeding*, pp. 511-518, 2001.

[75] Caltech Vehicle Image Dataset, see http://www.vision.caltech.edu/htmlfiles.

[76] A. Haselhoff, A. Kummert and G. Schneider, "Radar-Vision Fusion with an Application to Car-Following Using an Improved Adaboost Detection Algorithm," *IEEE Intelligent Transportation Systems Conference*, pp. 854-858, 2007.

[77] Tutorial OpenCV Haartraining, see http://sourceforge.net/projects/opencvlibrary.

[78] Rainer Lienhart, Alexander Kuranov, and Vadim Pisarevsky, "Empirical Analysis of Detection Cascades of Boosted Classifiers for Rapid Object Detection," *25th Pattern Recognition Symposium*, pp. 297-304, Sep. 2003.

[79] A. Djouadi, O. Snorrason and F. D. Garber, "The Quality of Training Sample Estimates of the Bhattacharyya Coefficient," *IEEE Trans. On Pattern Analysis and Machine Intelligence*, vol. 12, No. 1, pp. 92-97, 1990.

[80] Arulampalam, M.S., Maskell, S., Gordon, N., and Clapp, T., "A tutorial on particle filters for online nonlinear/non-Gaussian Bayesian tracking," *IEEE Transactions on Signal Processing*, vol. 50 (2), pp. 174–188, 2002.

[81] Ryo Kurazume, Hiroyuki Yamada, Kouji Murakami, Yumi Iwashita and Tsutomu Hasegawa, "Target Tracking Using SIR and MCMC Particle Filters by Multiple Cameras and Laser Range Finders," *IEEE/RSJ International Conference on Intelligent Robots and Systems*, pp. 3838-3844, 2008.

[82] Sebastian Thrun, Wolfram Burgard, and Dieter Fox, *Probabilistic Robotics*, The MIT Press, Cambridge, Massachusetts, USA, 2005.

[83] Farrell, J. A., *Aided Navigation: GPS with High Rate Sensors*, 2008. ISBN: 978-0-07-149329-8. McGraw-Hill.

[84] P. V. C. Hough, *A Method and Means for Recognizing Complex Patterns*, US Patent, vol. 3,069,654,1962.

[85] Alessio Dore, Andrea Beoldo and Carlo S. Regazzoni, "Multitarget Tracking with a Corner-based Particle Filter," *IEEE 12th International Conference on Computer Vision Workshops*, pp. 1251-1258, 2009.

[86] Golub, Gene; Charles F. Van Loan, *Matrix Computations - Third Edition*, The Johns Hopkins University Press, 56-57. ISBN 0-8018-5413-X.

[87] Conway, J.B., *A Course in Functional Analysis: Graduate Texts in Mathematics*, New York: Springer 1990

Appendix

A. Calculation of H in point feature detection

For point feature detection, we have

$$\begin{aligned} y_{L1} &= \|{}^{l}\mathbf{T}_{LP}\| + n_{L1} \\ y_{L2} &= \operatorname{atan2}(\mathbf{e}_2^T \cdot {}^{l}\mathbf{T}_{LP}\,, \mathbf{e}_1^T \cdot {}^{l}\mathbf{T}_{LP}) + n_{L2} \end{aligned}$$

Therefore

$$\begin{aligned} \frac{\partial y_{L1}}{\partial \mathbf{x}} &= \frac{{}^{l}\mathbf{T}_{LP}}{\|{}^{l}\mathbf{T}_{LP}\|} \cdot \frac{\partial {}^{l}\mathbf{T}_{LP}}{\partial \mathbf{x}} \\ &= {}^{l}\vec{\mathbf{T}}_{LP} \cdot \frac{\partial {}^{l}\mathbf{T}_{LP}}{\partial \mathbf{x}} \end{aligned} \tag{A1}$$

where ${}^{l}\vec{\mathbf{T}}_{LP}$ is a normal vector representing the direction of the vector ${}^{l}\mathbf{T}_{LP}$.

$$\begin{aligned} \frac{\partial y_{L2}}{\partial \mathbf{x}} &= \frac{1}{1 + \left(\frac{\mathbf{e}_2^T \cdot {}^{l}\mathbf{T}_{LP}}{\mathbf{e}_1^T \cdot {}^{l}\mathbf{T}_{LP}}\right)^2} \cdot \frac{\frac{\partial \mathbf{e}_2^T \cdot {}^{l}\mathbf{T}_{LP}}{\partial \mathbf{x}}(\mathbf{e}_1^T \cdot {}^{l}\mathbf{T}_{LP}) - \frac{\partial \mathbf{e}_1^T \cdot {}^{l}\mathbf{T}_{LP}}{\partial \mathbf{x}}(\mathbf{e}_2^T \cdot {}^{l}\mathbf{T}_{LP})}{(\mathbf{e}_1^T \cdot {}^{l}\mathbf{T}_{LP})^2} \\ &= \frac{1}{(\mathbf{e}_1^T \cdot {}^{l}\mathbf{T}_{LP})^2 + (\mathbf{e}_2^T \cdot {}^{l}\mathbf{T}_{LP})^2} \cdot \left(\frac{\partial \mathbf{e}_2^T \cdot {}^{l}\mathbf{T}_{LP}}{\partial \mathbf{x}}(\mathbf{e}_1^T \cdot {}^{l}\mathbf{T}_{LP}) - \frac{\partial \mathbf{e}_1^T \cdot {}^{l}\mathbf{T}_{LP}}{\partial \mathbf{x}}(\mathbf{e}_2^T \cdot {}^{l}\mathbf{T}_{LP}) \right) \\ &= \frac{1}{\|{}^{l}\mathbf{T}_{LP}\|^2} [\mathbf{u}_1 \quad -\mathbf{u}_2] \begin{bmatrix} \frac{\partial \mathbf{u}_2}{\partial \mathbf{x}} \\ \frac{\partial \mathbf{u}_1}{\partial \mathbf{x}} \end{bmatrix} \end{aligned} \tag{A2}$$

where $\mathbf{u}_1 = \mathbf{e}_1^T \cdot {}^{l}\mathbf{T}_{LP}$ and $\mathbf{u}_2 = \mathbf{e}_2^T \cdot {}^{l}\mathbf{T}_{LP}$. So we need to solve $\partial {}^{l}\mathbf{T}_{LP}/\partial \mathbf{x}$.

From Eqn. (5-39) we know

$$\begin{aligned} {}^{l}\mathbf{T}_{LP} &= \mathbf{R}_b^l \left(\mathbf{R}_t^b \left({}^{t}\mathbf{T}_{TP0} - \frac{\mathbf{e}_3^T \mathbf{R}_b^l \left(\mathbf{R}_t^b ({}^{t}\mathbf{T}_{TP0} - {}^{t}\mathbf{T}_{TB}) - {}^{b}\mathbf{T}_{BL} \right)}{\mathbf{e}_3^T \mathbf{R}_b^l \mathbf{R}_t^b \mathbf{e}_3} \mathbf{e}_3 - {}^{t}\mathbf{T}_{TB} \right) - {}^{b}\mathbf{T}_{BL} \right) \\ &= \mathbf{R}_b^l \mathbf{R}_t^b \, {}^{t}\mathbf{T}_{TP0} - \mathbf{R}_b^l \mathbf{R}_t^b \left(\frac{\mathbf{e}_3^T \mathbf{R}_b^l \mathbf{R}_t^b \, {}^{t}\mathbf{T}_{TP0} - \mathbf{e}_3^T \mathbf{R}_b^l \mathbf{R}_t^b \, {}^{t}\mathbf{T}_{TB} - \mathbf{e}_3^T \mathbf{R}_b^l \, {}^{b}\mathbf{T}_{BL}}{\mathbf{e}_3^T \mathbf{R}_b^l \mathbf{R}_t^b \mathbf{e}_3} \mathbf{e}_3 \right) \\ &\quad - \mathbf{R}_b^l \mathbf{R}_t^b \, {}^{t}\mathbf{T}_{TB} - \mathbf{R}_b^l \, {}^{b}\mathbf{T}_{BL} \end{aligned} \tag{A3}$$

We will represent the error ${}^{l}\delta\mathbf{T}_{LP} = {}^{l}\mathbf{T}_{LP} - {}^{l}\widehat{\mathbf{T}}_{LP}$ in terms of the error state $\delta\mathbf{x}$. ${}^{l}\delta\mathbf{T}_{LP}$ will be computed term by term. The second order terms will be eliminated.

Let $\mathbf{A} = \mathbf{R}_b^l \mathbf{R}_t^b \, {}^t\mathbf{T}_{TP0}$, $\mathbf{B} = \mathbf{e}_3^T \mathbf{R}_b^l \mathbf{R}_t^b \, {}^t\mathbf{T}_{TP0}\mathbf{e}_3$, $\mathbf{C} = \mathbf{e}_3^T \mathbf{R}_b^l \mathbf{R}_t^b \, {}^t\mathbf{T}_{TB}\mathbf{e}_3$, $\mathbf{D} = \mathbf{e}_3^T \mathbf{R}_b^l \, {}^b\mathbf{T}_{BL}\mathbf{e}_3$, $\mathbf{E} = \mathbf{R}_b^l \mathbf{R}_t^b \, {}^t\mathbf{T}_{TB}$, and $\mathbf{F} = \mathbf{e}_3^T \mathbf{R}_b^l \mathbf{R}_t^b \mathbf{e}_3$, then

$$ {}^{l}\mathbf{T}_{LP} = \mathbf{A} - \mathbf{R}_b^l \mathbf{R}_t^b \frac{\mathbf{B} - \mathbf{C} - \mathbf{D}}{\mathbf{F}} - \mathbf{E} - \mathbf{R}_b^l \, {}^b\mathbf{T}_{BL} \tag{A4} $$

Similarly, ${}^{l}\widehat{\mathbf{T}}_{LP} = \widehat{\mathbf{A}} - \mathbf{R}_b^l \widehat{\mathbf{R}}_t^b \frac{\widehat{\mathbf{B}} - \widehat{\mathbf{C}} - \widehat{\mathbf{D}}}{\widehat{\mathbf{F}}} - \widehat{\mathbf{E}} - \mathbf{R}_b^l \, {}^b\mathbf{T}_{BL}$, where $\widehat{\mathbf{A}} = \mathbf{R}_b^l \widehat{\mathbf{R}}_t^b \, {}^t\mathbf{T}_{TP}$, $\widehat{\mathbf{B}} = \mathbf{e}_3^T \mathbf{R}_b^l \widehat{\mathbf{R}}_t^b \, {}^t\mathbf{T}_{TP0}\mathbf{e}_3$, $\widehat{\mathbf{C}} = \mathbf{e}_3^T \mathbf{R}_b^l \widehat{\mathbf{R}}_t^b \, {}^t\widehat{\mathbf{T}}_{TB}\mathbf{e}_3$, $\widehat{\mathbf{D}} = \mathbf{e}_3^T \mathbf{R}_b^l \, {}^b\mathbf{T}_{BL}\mathbf{e}_3$, $\widehat{\mathbf{E}} = \mathbf{R}_b^l \widehat{\mathbf{R}}_t^b \, {}^t\widehat{\mathbf{T}}_{TB}$, and $\widehat{\mathbf{F}} = \mathbf{e}_3^T \mathbf{R}_b^l \widehat{\mathbf{R}}_t^b \mathbf{e}_3$. So

$$ \begin{aligned} \delta\mathbf{A} &= \mathbf{A} - \widehat{\mathbf{A}} \\ &= \mathbf{R}_b^l \mathbf{R}_t^b \, {}^t\mathbf{T}_{TP0} - \mathbf{R}_b^l \widehat{\mathbf{R}}_t^b \, {}^t\mathbf{T}_{TP0} \\ &= \mathbf{R}_b^l \widehat{\mathbf{R}}_t^b (\mathbf{I} - [\boldsymbol{\rho}\times] - \mathbf{I}) \, {}^t\mathbf{T}_{TP0} \\ &= \mathbf{R}_b^l \widehat{\mathbf{R}}_t^b [{}^t\mathbf{T}_{TP0}\times]\boldsymbol{\rho} \end{aligned} \tag{A5} $$

and

$$ \begin{aligned} \delta\mathbf{E} &= \mathbf{R}_b^l \mathbf{R}_t^b \, {}^t\mathbf{T}_{TB} - \mathbf{R}_b^l \widehat{\mathbf{R}}_t^b \, {}^t\widehat{\mathbf{T}}_{TB} \\ &= \mathbf{R}_b^l \left(\widehat{\mathbf{R}}_t^b (\mathbf{I} - [\boldsymbol{\rho}\times]) \left({}^t\widehat{\mathbf{T}}_{TB} + {}^t\delta\mathbf{T}_{TB} \right) - \widehat{\mathbf{R}}_t^b \, {}^t\widehat{\mathbf{T}}_{TB} \right) \\ &= \mathbf{R}_b^l \left(\widehat{\mathbf{R}}_t^b \left[{}^t\widehat{\mathbf{T}}_{TB}\times \right]\boldsymbol{\rho} + \widehat{\mathbf{R}}_t^b \, {}^t\delta\mathbf{T}_{TB} \right) \end{aligned} \tag{A6} $$

For $\mathbf{B}/\mathbf{F}$ we have:

$$ \frac{\mathbf{B}}{\mathbf{F}} = \frac{\widehat{\mathbf{B}} + \delta\mathbf{B}}{\widehat{\mathbf{F}} + \delta\mathbf{F}} = \frac{\widehat{\mathbf{B}} + \delta\mathbf{B}}{\widehat{\mathbf{F}}\left(\mathbf{I} + \frac{\delta\mathbf{F}}{\widehat{\mathbf{F}}}\right)} \approx \frac{\widehat{\mathbf{B}} + \delta\mathbf{B}}{\widehat{\mathbf{F}}}\left(\mathbf{I} - \frac{\delta\mathbf{F}}{\widehat{\mathbf{F}}}\right) \approx \frac{\widehat{\mathbf{B}}}{\widehat{\mathbf{F}}} + \frac{\delta\mathbf{B}}{\widehat{\mathbf{F}}} - \frac{\widehat{\mathbf{B}}\delta\mathbf{F}}{\widehat{\mathbf{F}}^2} \tag{A7} $$

in which we use the property that $1/(1+\mathrm{x}) \approx 1-\mathrm{x}$ when $|\mathrm{x}| \ll 1$. So

$$\delta\frac{\mathbf{B}}{\mathbf{F}} = \frac{\mathbf{B}}{\mathbf{F}} - \frac{\widehat{\mathbf{B}}}{\widehat{\mathbf{F}}} = \frac{\delta\mathbf{B}}{\widehat{\mathbf{F}}} - \frac{\widehat{\mathbf{B}}\delta\mathbf{F}}{\widehat{\mathbf{F}}^2} \tag{A8}$$

So error of the second term in eqn. (A3) is:

$$\begin{aligned}
&\mathbf{R}_{\mathrm{b}}^{\mathrm{l}}\mathbf{R}_{\mathrm{t}}^{\mathrm{b}}\frac{\mathbf{B}-\mathbf{C}-\mathbf{D}}{\mathbf{F}} - \mathbf{R}_{\mathrm{b}}^{\mathrm{l}}\widehat{\mathbf{R}}_{\mathrm{t}}^{\mathrm{b}}\frac{\widehat{\mathbf{B}}-\widehat{\mathbf{C}}-\widehat{\mathbf{D}}}{\widehat{\mathbf{F}}} \\
&= \mathbf{R}_{\mathrm{b}}^{\mathrm{l}}\widehat{\mathbf{R}}_{\mathrm{t}}^{\mathrm{b}}(\mathbf{I}-[\boldsymbol{\rho}\times])\left(\frac{\widehat{\mathbf{B}}+\delta\mathbf{B}}{\widehat{\mathbf{F}}+\delta\mathbf{F}} - \frac{\widehat{\mathbf{C}}+\delta\mathbf{C}}{\widehat{\mathbf{F}}+\delta\mathbf{F}} - \frac{\widehat{\mathbf{D}}+\delta\mathbf{D}}{\widehat{\mathbf{F}}+\delta\mathbf{F}}\right) - \mathbf{R}_{\mathrm{b}}^{\mathrm{l}}\widehat{\mathbf{R}}_{\mathrm{t}}^{\mathrm{b}}\frac{\widehat{\mathbf{B}}-\widehat{\mathbf{C}}-\widehat{\mathbf{D}}}{\widehat{\mathbf{F}}} \\
&= \mathbf{R}_{\mathrm{b}}^{\mathrm{l}}\widehat{\mathbf{R}}_{\mathrm{t}}^{\mathrm{b}}\left(\delta\frac{\mathbf{B}}{\mathbf{F}} - \delta\frac{\mathbf{C}}{\mathbf{F}} - \delta\frac{\mathbf{D}}{\mathbf{F}}\right) \\
&-\mathbf{R}_{\mathrm{b}}^{\mathrm{l}}\widehat{\mathbf{R}}_{\mathrm{t}}^{\mathrm{b}}[\boldsymbol{\rho}\times]\left(\left(\frac{\widehat{\mathbf{B}}}{\widehat{\mathbf{F}}} + \frac{\delta\mathbf{B}}{\widehat{\mathbf{F}}} - \frac{\widehat{\mathbf{B}}\delta\mathbf{F}}{\widehat{\mathbf{F}}^2}\right) - \left(\frac{\widehat{\mathbf{C}}}{\widehat{\mathbf{F}}} + \frac{\delta\mathbf{C}}{\widehat{\mathbf{F}}} - \frac{\widehat{\mathbf{C}}\delta\mathbf{F}}{\widehat{\mathbf{F}}^2}\right) - \left(\frac{\widehat{\mathbf{D}}}{\widehat{\mathbf{F}}} + \frac{\delta\mathbf{D}}{\widehat{\mathbf{F}}} - \frac{\widehat{\mathbf{D}}\delta\mathbf{F}}{\widehat{\mathbf{F}}^2}\right)\right) \\
&\approx \mathbf{R}_{\mathrm{b}}^{\mathrm{l}}\widehat{\mathbf{R}}_{\mathrm{t}}^{\mathrm{b}}\left(\delta\frac{\mathbf{B}}{\mathbf{F}} - \delta\frac{\mathbf{C}}{\mathbf{F}} - \delta\frac{\mathbf{D}}{\mathbf{F}}\right) + \mathbf{R}_{\mathrm{b}}^{\mathrm{l}}\widehat{\mathbf{R}}_{\mathrm{t}}^{\mathrm{b}}\left[\frac{\widehat{\mathbf{B}}-\widehat{\mathbf{C}}-\widehat{\mathbf{D}}}{\widehat{\mathbf{F}}}\times\right]\boldsymbol{\rho}
\end{aligned} \tag{A9}$$

$\delta\mathbf{B}$, $\delta\mathbf{C}$, $\delta\mathbf{D}$ and $\delta\mathbf{F}$ are calculated as:

$$\begin{aligned}
\delta\mathbf{B} &= \mathbf{B} - \widehat{\mathbf{B}} \\
&= \mathbf{e}_3^{\mathrm{T}}\,\mathbf{R}_{\mathrm{b}}^{\mathrm{l}}\mathbf{R}_{\mathrm{t}}^{\mathrm{b}}\,{}^{\mathrm{t}}\mathbf{T}_{\mathrm{TP0}}\,\mathbf{e}_3 - \mathbf{e}_3^{\mathrm{T}}\,\mathbf{R}_{\mathrm{b}}^{\mathrm{l}}\widehat{\mathbf{R}}_{\mathrm{t}}^{\mathrm{b}}\,{}^{\mathrm{t}}\mathbf{T}_{\mathrm{TP0}}\,\mathbf{e}_3 \\
&= \mathbf{e}_3^{\mathrm{T}}\,\mathbf{R}_{\mathrm{b}}^{\mathrm{l}}\,\widehat{\mathbf{R}}_{\mathrm{t}}^{\mathrm{b}}(\mathbf{I}-[\boldsymbol{\rho}\times])\,{}^{\mathrm{t}}\mathbf{T}_{\mathrm{TP0}}\,\mathbf{e}_3 - \mathbf{e}_3^{\mathrm{T}}\,\mathbf{R}_{\mathrm{b}}^{\mathrm{l}}\widehat{\mathbf{R}}_{\mathrm{t}}^{\mathrm{b}}\,{}^{\mathrm{t}}\mathbf{T}_{\mathrm{TP0}}\,\mathbf{e}_3 \\
&= \mathbf{e}_3^{\mathrm{T}}\,\mathbf{R}_{\mathrm{b}}^{\mathrm{l}}\widehat{\mathbf{R}}_{\mathrm{t}}^{\mathrm{b}}[{}^{\mathrm{t}}\mathbf{T}_{\mathrm{TP0}}\times]\,\boldsymbol{\rho}\,\mathbf{e}_3 \qquad \text{(A10)} \\
&= \mathbf{e}_3\left(\mathbf{e}_3^{\mathrm{T}}\,\mathbf{R}_{\mathrm{b}}^{\mathrm{l}}\widehat{\mathbf{R}}_{\mathrm{t}}^{\mathrm{b}}[{}^{\mathrm{t}}\mathbf{T}_{\mathrm{TP0}}\times]\right)\boldsymbol{\rho} \qquad \text{(A11)}
\end{aligned}$$

From (A8) to (A9) we used the property that $\mathbf{e}_3^{\mathrm{T}}\,\mathbf{R}_{\mathrm{b}}^{\mathrm{l}}\widehat{\mathbf{R}}_{\mathrm{t}}^{\mathrm{b}}[{}^{\mathrm{t}}\mathbf{T}_{\mathrm{TP0}}\times]\,\boldsymbol{\rho}$ is a scalar.

Similarly we have:

$$\delta\mathbf{C} = \mathbf{C} - \widehat{\mathbf{C}}$$

$$= \mathbf{e}_3^T \mathbf{R}_b^l \mathbf{R}_t^b \, {}^t\mathbf{T}_{TB}\mathbf{e}_3 - \mathbf{e}_3^T \mathbf{R}_b^l \widehat{\mathbf{R}}_t^b \, {}^t\widehat{\mathbf{T}}_{TB}\mathbf{e}_3$$

$$= \mathbf{e}_3^T \mathbf{R}_b^l \, \widehat{\mathbf{R}}_t^b (\mathbf{I} - [\boldsymbol{\rho}\times]) \left({}^t\widehat{\mathbf{T}}_{TB} + {}^t\delta\mathbf{T}_{TB} \right)\mathbf{e}_3 - \mathbf{e}_3^T \mathbf{R}_b^l \widehat{\mathbf{R}}_t^b \, {}^t\widehat{\mathbf{T}}_{TB}\mathbf{e}_3$$

$$= \mathbf{e}_3^T \mathbf{R}_b^l \, \widehat{\mathbf{R}}_t^b \left[{}^t\widehat{\mathbf{T}}_{TB} \times \right]\boldsymbol{\rho} \, \mathbf{e}_3 + \mathbf{e}_3^T \mathbf{R}_b^l \, \widehat{\mathbf{R}}_t^b \, {}^t\delta\mathbf{T}_{TB}\mathbf{e}_3$$

$$= \mathbf{e}_3 \left(\mathbf{e}_3^T \mathbf{R}_b^l \, \widehat{\mathbf{R}}_t^b \left[{}^t\widehat{\mathbf{T}}_{TB} \times \right] \right) \boldsymbol{\rho} + \mathbf{e}_3 \left(\mathbf{e}_3^T \mathbf{R}_b^l \, \widehat{\mathbf{R}}_t^b \right) \, {}^t\delta\mathbf{T}_{TB} \tag{A12}$$

$\delta\mathbf{D}$ is calculated as

$$\delta\mathbf{D} = \mathbf{D} - \widehat{\mathbf{D}} = \mathbf{0} \tag{A13}$$

and

$$\delta\mathbf{F} = \mathbf{e}_3^T \mathbf{R}_b^l \mathbf{R}_t^b \mathbf{e}_3 - \mathbf{e}_3^T \mathbf{R}_b^l \widehat{\mathbf{R}}_t^b \mathbf{e}_3$$

$$= \mathbf{e}_3^T \mathbf{R}_b^l \left(\widehat{\mathbf{R}}_t^b (\mathbf{I} - [\boldsymbol{\rho}\times]) - \widehat{\mathbf{R}}_t^b \right)\mathbf{e}_3$$

$$= \mathbf{e}_3^T \mathbf{R}_b^l \widehat{\mathbf{R}}_t^b [\mathbf{e}_3 \times] \, \boldsymbol{\rho} \tag{A14}$$

Therefore,

$$\delta\frac{\mathbf{B}}{\mathbf{F}} = \frac{\delta\mathbf{B}}{\widehat{\mathbf{F}}} - \frac{\widehat{\mathbf{B}}\delta\mathbf{F}}{\widehat{\mathbf{F}}^2} = \frac{\mathbf{e}_3 \left(\mathbf{e}_3^T \mathbf{R}_b^l \widehat{\mathbf{R}}_t^b [{}^t\mathbf{T}_{TP0} \times] \right)}{\widehat{\mathbf{F}}} \boldsymbol{\rho} - \frac{\widehat{\mathbf{B}} \left(\mathbf{e}_3^T \mathbf{R}_b^l \widehat{\mathbf{R}}_t^b [\mathbf{e}_3 \times] \right)}{\widehat{\mathbf{F}}^2} \boldsymbol{\rho}$$

$$\delta\frac{\mathbf{C}}{\mathbf{F}} = \frac{\delta\mathbf{C}}{\widehat{\mathbf{F}}} - \frac{\widehat{\mathbf{C}}\delta\mathbf{F}}{\widehat{\mathbf{F}}^2}$$

$$= \frac{\mathbf{e}_3 \left(\mathbf{e}_3^T \mathbf{R}_b^l \, \widehat{\mathbf{R}}_t^b \left[{}^t\widehat{\mathbf{T}}_{TB} \times \right] \right)}{\widehat{\mathbf{F}}} \boldsymbol{\rho} + \frac{\mathbf{e}_3 \left(\mathbf{e}_3^T \mathbf{R}_b^l \, \widehat{\mathbf{R}}_t^b \right)}{\widehat{\mathbf{F}}} \, {}^t\delta\mathbf{T}_{TB} - \frac{\widehat{\mathbf{C}} \left(\mathbf{e}_3^T \mathbf{R}_b^l \widehat{\mathbf{R}}_t^b [\mathbf{e}_3 \times] \right)}{\widehat{\mathbf{F}}^2} \boldsymbol{\rho}$$

$$\delta\frac{\mathbf{D}}{\mathbf{F}} = \frac{\delta\mathbf{D}}{\widehat{\mathbf{F}}} - \frac{\widehat{\mathbf{D}}\delta\mathbf{F}}{\widehat{\mathbf{F}}^2} = -\frac{\widehat{\mathbf{D}} \left(\mathbf{e}_3^T \mathbf{R}_b^l \widehat{\mathbf{R}}_t^b [\mathbf{e}_3 \times] \right)}{\widehat{\mathbf{F}}^2} \boldsymbol{\rho} \tag{A15}$$

From eqns. (A4)-(A15), the error ${}^l\delta\mathbf{T}_{LP}$ is computed as:

$${}^l\delta\mathbf{T}_{LP} = \delta\mathbf{A} - \left(\mathbf{R}_b^l \widehat{\mathbf{R}}_t^b \left(\delta\frac{\mathbf{B}}{\mathbf{F}} - \delta\frac{\mathbf{C}}{\mathbf{F}} - \delta\frac{\mathbf{D}}{\mathbf{F}} \right) + \mathbf{R}_b^l \widehat{\mathbf{R}}_t^b \left[\frac{\widehat{\mathbf{B}} - \widehat{\mathbf{C}} - \widehat{\mathbf{D}}}{\widehat{\mathbf{F}}} \times \right] \boldsymbol{\rho} \right) - \delta\mathbf{E}$$

$$= \mathbf{R}_b^l \widehat{\mathbf{R}}_t^b [{}^t\mathbf{T}_{TP0} \times] \boldsymbol{\rho} - \mathbf{R}_b^l \widehat{\mathbf{R}}_t^b \left(\frac{\mathbf{e}_3 \left(\mathbf{e}_3^T \mathbf{R}_b^l \widehat{\mathbf{R}}_t^b [{}^t\mathbf{T}_{TP0} \times] \right)}{\widehat{\mathbf{F}}} - \frac{\widehat{\mathbf{B}} \left(\mathbf{e}_3^T \mathbf{R}_b^l \widehat{\mathbf{R}}_t^b [\mathbf{e}_3 \times] \right)}{\widehat{\mathbf{F}}^2} \right) \boldsymbol{\rho}$$

$$+\mathbf{R}_b^l\hat{\mathbf{R}}_t^b\left(\frac{\mathbf{e}_3\left(\mathbf{e}_3^T\,\mathbf{R}_b^l\,\hat{\mathbf{R}}_t^b\left[{}^t\hat{\mathbf{T}}_{TB}\times\right]\right)}{\hat{\mathbf{F}}}-\frac{\hat{\mathbf{C}}\left(\mathbf{e}_3^T\,\mathbf{R}_b^l\hat{\mathbf{R}}_t^b[\mathbf{e}_3\times]\right)}{\hat{\mathbf{F}}^2}\right)\boldsymbol{\rho}$$

$$+\mathbf{R}_b^l\hat{\mathbf{R}}_t^b\left(\frac{\mathbf{e}_3\left(\mathbf{e}_3^T\,\mathbf{R}_b^l\,\hat{\mathbf{R}}_t^b\right)}{\hat{\mathbf{F}}}\right)\,{}^t\delta\mathbf{T}_{TB}-\mathbf{R}_b^l\hat{\mathbf{R}}_t^b\left(\frac{\hat{\mathbf{D}}\left(\mathbf{e}_3^T\,\mathbf{R}_b^l\hat{\mathbf{R}}_t^b[\mathbf{e}_3\times]\right)}{\hat{\mathbf{F}}^2}\right)\boldsymbol{\rho}$$

$$-\mathbf{R}_b^l\hat{\mathbf{R}}_t^b\left[\frac{\hat{\mathbf{B}}-\hat{\mathbf{C}}-\hat{\mathbf{D}}}{\hat{\mathbf{F}}}\times\right]\boldsymbol{\rho}-\mathbf{R}_b^l\hat{\mathbf{R}}_t^b\left[{}^t\hat{\mathbf{T}}_{TB}\times\right]\boldsymbol{\rho}-\mathbf{R}_b^l\hat{\mathbf{R}}_t^b\,{}^t\delta\mathbf{T}_{TB}$$

$$=\mathbf{R}_b^l\hat{\mathbf{R}}_t^b\left(\left[{}^t\mathbf{T}_{TP0}\times\right]-\frac{\mathbf{e}_3\left(\mathbf{e}_3^T\,\mathbf{R}_b^l\hat{\mathbf{R}}_t^b\left(\left[{}^t\mathbf{T}_{TP0}\times\right]-\left[{}^t\hat{\mathbf{T}}_{TB}\times\right]\right)\right)}{\hat{\mathbf{F}}}\right.$$

$$\left.+\frac{\left(\hat{\mathbf{B}}-\hat{\mathbf{C}}-\hat{\mathbf{D}}\right)\left(\mathbf{e}_3^T\,\mathbf{R}_b^l\hat{\mathbf{R}}_t^b[\mathbf{e}_3\times]\right)}{\hat{\mathbf{F}}^2}-\left[\frac{\hat{\mathbf{B}}-\hat{\mathbf{C}}-\hat{\mathbf{D}}}{\hat{\mathbf{F}}}\times\right]-\left[{}^t\hat{\mathbf{T}}_{TB}\times\right]\right)\boldsymbol{\rho}$$

$$+\mathbf{R}_b^l\hat{\mathbf{R}}_t^b\left(\frac{\mathbf{e}_3\left(\mathbf{e}_3^T\,\mathbf{R}_b^l\,\hat{\mathbf{R}}_t^b\right)}{\hat{\mathbf{F}}}-\mathbf{I}\right)\,{}^t\delta\mathbf{T}_{TB} \qquad \text{(A16)}$$

Let $\mathbf{F}_{y\rho}=\mathbf{R}_b^l\hat{\mathbf{R}}_t^b\left(\left[{}^t\mathbf{T}_{TP0}\times\right]-\frac{\mathbf{e}_3\left(\mathbf{e}_3^T\,\mathbf{R}_b^l\hat{\mathbf{R}}_t^b\left(\left[{}^t\mathbf{T}_{TP0}\times\right]-\left[{}^t\hat{\mathbf{T}}_{TB}\times\right]\right)\right)}{\hat{\mathbf{F}}}+\frac{\left(\hat{\mathbf{B}}-\hat{\mathbf{C}}-\hat{\mathbf{D}}\right)\left(\mathbf{e}_3^T\,\mathbf{R}_b^l\hat{\mathbf{R}}_t^b[\mathbf{e}_3\times]\right)}{\hat{\mathbf{F}}^2}-\left[\frac{\hat{\mathbf{B}}-\hat{\mathbf{C}}-\hat{\mathbf{D}}}{\hat{\mathbf{F}}}\times\right]-\left[{}^t\hat{\mathbf{T}}_{TB}\times\right]\right)$, and $\mathbf{F}_{yT}=\mathbf{R}_b^l\hat{\mathbf{R}}_t^b\left(\frac{\mathbf{e}_3\left(\mathbf{e}_3^T\,\mathbf{R}_b^l\,\hat{\mathbf{R}}_t^b\right)}{\hat{\mathbf{F}}}-\mathbf{I}\right)$, Eqn. (A16) is

$${}^l\delta\mathbf{T}_{LP}=[\mathbf{F}_{yT},\mathbf{0},\mathbf{F}_{y\rho},\mathbf{0},\mathbf{0}]\delta\mathbf{x} \qquad \text{(A17)}$$

Then (A1) and (A2) will be rewritten as:

$$\frac{\partial y_{L1}}{\partial\mathbf{x}}={}^l\vec{\mathbf{T}}_{LP}\cdot[\mathbf{F}_{yT},\mathbf{0},\mathbf{F}_{y\rho},\mathbf{0},\mathbf{0}]$$

$$\frac{\partial y_{L2}}{\partial\mathbf{x}}=\frac{1}{\|{}^l\mathbf{T}_{LP}\|^2}\begin{bmatrix}\mathbf{u}_1 & -\mathbf{u}_2\end{bmatrix}\begin{bmatrix}\mathbf{e}_2^T\\ \mathbf{e}_1^T\end{bmatrix}[\mathbf{F}_{yT},\mathbf{0},\mathbf{F}_{y\rho},\mathbf{0},\mathbf{0}]$$

$$=\frac{\mathbf{u}_1\mathbf{e}_2^T-\mathbf{u}_2\mathbf{e}_1^T}{\|{}^l\mathbf{T}_{LP}\|^2}[\mathbf{F}_{yT},\mathbf{0},\mathbf{F}_{y\rho},\mathbf{0},\mathbf{0}] \qquad \text{(A18)}$$

B. Calculation of H in Line Feature Detection

For point feature detection, we have

$$y_{L1} = \frac{\mathbf{K}}{\mathbf{G}} + n_{L1}, \quad y_{L2} = \frac{\mathbf{J}}{\mathbf{G}} + n_{L2} \tag{A19}$$

where

$$\mathbf{G} = \mathbf{e}_1^T\left(\mathbf{R}_b^l \mathbf{R}_t^{bt} \mathbf{r}\right)$$

$$\mathbf{J} = \mathbf{e}_2^T\left(\mathbf{R}_b^l \mathbf{R}_t^{bt} \mathbf{r}\right)$$

$$\mathbf{K} = -\left(\mathbf{R}_b^l\left(\mathbf{R}_t^b({}^t\mathbf{T}_{TP0} - {}^t\mathbf{T}_{TB}) - {}^b\mathbf{T}_{BL}\right)\right)^T\left(\mathbf{R}_b^l \mathbf{R}_t^{bt} \mathbf{r}\right)$$

Therefore

$$\frac{\partial y_{L1}}{\partial \mathbf{x}} = \frac{\frac{\partial \mathbf{K}}{\partial \mathbf{x}}\mathbf{G} - \frac{\partial \mathbf{G}}{\partial \mathbf{x}}\mathbf{K}}{\mathbf{G}^2}, \qquad \frac{\partial y_{L2}}{\partial \mathbf{x}} = \frac{\frac{\partial \mathbf{J}}{\partial \mathbf{x}}\mathbf{G} - \frac{\partial \mathbf{G}}{\partial \mathbf{x}}\mathbf{J}}{\mathbf{G}^2} \tag{A20}$$

We need to calculate $\partial\mathbf{G}/\partial\mathbf{x}$, $\partial\mathbf{J}/\partial\mathbf{x}$ and $\partial\mathbf{K}/\partial\mathbf{x}$.

$$\begin{aligned}
\delta\mathbf{G} &= \mathbf{e}_1^T\left(\mathbf{R}_b^l \mathbf{R}_t^{bt} \mathbf{r}\right) - \mathbf{e}_1^T\left(\mathbf{R}_b^l \widehat{\mathbf{R}}_t^{bt} \mathbf{r}\right) \\
&= \mathbf{e}_1^T\left(\mathbf{R}_b^l \widehat{\mathbf{R}}_t^b (\mathbf{I} - [\boldsymbol{\rho}\times])^t\mathbf{r} - \mathbf{R}_b^l \widehat{\mathbf{R}}_t^{bt} \mathbf{r}\right) \\
&= \mathbf{e}_1^T\left(\mathbf{R}_b^l \widehat{\mathbf{R}}_t^b [{}^t\mathbf{r}\times]\boldsymbol{\rho}\right) \\
&= \mathbf{e}_1^T \mathbf{R}_b^l \widehat{\mathbf{R}}_t^b [{}^t\mathbf{r}\times]\boldsymbol{\rho}
\end{aligned} \tag{A21}$$

Similarly,

$$\delta\mathbf{J} = \mathbf{e}_2^T \mathbf{R}_b^l \widehat{\mathbf{R}}_t^b [{}^t\mathbf{r}\times]\boldsymbol{\rho} \tag{A22}$$

For $\partial\mathbf{K}/\partial\mathbf{x}$, we have:

$$\begin{aligned}
\delta\mathbf{K} &= -\left(\mathbf{R}_b^l\left(\mathbf{R}_t^b({}^t\mathbf{T}_{TP0} - {}^t\mathbf{T}_{TB}) - {}^b\mathbf{T}_{BL}\right)\right)^T\left(\mathbf{R}_b^l \mathbf{R}_t^{bt} \mathbf{r}\right) \\
&\quad + \left(\mathbf{R}_b^l\left(\widehat{\mathbf{R}}_t^b({}^t\mathbf{T}_{TP0} - {}^t\widehat{\mathbf{T}}_{TB}) - {}^b\mathbf{T}_{BL}\right)\right)^T\left(\mathbf{R}_b^l \widehat{\mathbf{R}}_t^{bt} \mathbf{r}\right) \\
&= -\left(\left(\mathbf{R}_b^l \mathbf{R}_t^{b\,t}\mathbf{T}_{TP0}\right)^T\left(\mathbf{R}_b^l \mathbf{R}_t^{b\,t}\mathbf{r}\right) - \left(\mathbf{R}_b^l \widehat{\mathbf{R}}_t^{b\,t}\mathbf{T}_{TP0}\right)^T\left(\mathbf{R}_b^l \widehat{\mathbf{R}}_t^{bt}\mathbf{r}\right)\right)
\end{aligned}$$

$$+\left(\left(\mathbf{R}_b^l\mathbf{R}_t^b\,{}^t\mathbf{T}_{TB}\right)^T\left(\mathbf{R}_b^l\mathbf{R}_t^b\,{}^t\mathbf{r}\right)-\left(\mathbf{R}_b^l\widehat{\mathbf{R}}_t^b\,{}^t\widehat{\mathbf{T}}_{TB}\right)^T\left(\mathbf{R}_b^l\widehat{\mathbf{R}}_t^b\,{}^t\mathbf{r}\right)\right)$$

$$+\left(\left(\mathbf{R}_b^l\,{}^b\mathbf{T}_{BL}\right)^T\left(\mathbf{R}_b^l\mathbf{R}_t^b\,{}^t\mathbf{r}\right)-\left(\mathbf{R}_b^l\,{}^b\mathbf{T}_{BL}\right)^T\left(\mathbf{R}_b^l\widehat{\mathbf{R}}_t^b\,{}^t\mathbf{r}\right)\right) \quad \text{(A23)}$$

where the first term in Eqn. (A23) is

$$-\left(\left(\mathbf{R}_b^l\mathbf{R}_t^b\,{}^t\mathbf{T}_{TP0}\right)^T\left(\mathbf{R}_b^l\mathbf{R}_t^b\,{}^t\mathbf{r}\right)-\left(\mathbf{R}_b^l\widehat{\mathbf{R}}_t^b\,{}^t\mathbf{T}_{TP0}\right)^T\left(\mathbf{R}_b^l\widehat{\mathbf{R}}_t^{bt}\mathbf{r}\right)\right)$$

$$=-\left(\mathbf{R}_b^l\widehat{\mathbf{R}}_t^b(\mathbf{I}-[\boldsymbol{\rho}\times])\,{}^t\mathbf{T}_{TP0}\right)^T\left(\mathbf{R}_b^l\widehat{\mathbf{R}}_t^b(\mathbf{I}-[\boldsymbol{\rho}\times])^t\mathbf{r}\right)-\left(\mathbf{R}_b^l\widehat{\mathbf{R}}_t^b\,{}^t\mathbf{T}_{TP0}\right)^T\left(\mathbf{R}_b^l\widehat{\mathbf{R}}_t^{bt}\mathbf{r}\right)$$

$$=-\left(\mathbf{R}_b^l\widehat{\mathbf{R}}_t^b[{}^t\mathbf{T}_{TP0}\times]\,\boldsymbol{\rho}\right)^T\left(\mathbf{R}_b^l\widehat{\mathbf{R}}_t^{bt}\mathbf{r}\right)-\left(\mathbf{R}_b^l\widehat{\mathbf{R}}_t^{bt}\mathbf{T}_{TP0}\right)^T\left(\mathbf{R}_b^l\widehat{\mathbf{R}}_t^b[{}^t\mathbf{r}\times]\boldsymbol{\rho}\right) \quad \text{(A24)}$$

$$=-\left(\mathbf{R}_b^l\widehat{\mathbf{R}}_t^{bt}\mathbf{r}\right)^T\left(\mathbf{R}_b^l\widehat{\mathbf{R}}_t^b\,[{}^t\mathbf{T}_{TP0}\times]\right)\boldsymbol{\rho}-\left(\mathbf{R}_b^l\widehat{\mathbf{R}}_t^{bt}\mathbf{T}_{TP0}\right)^T\left(\mathbf{R}_b^l\widehat{\mathbf{R}}_t^b[{}^t\mathbf{r}\times]\right)\boldsymbol{\rho} \quad \text{(A25)}$$

From Eqn. (A24) to (A25) we use the property that if χ is a scalar, then $\chi=\chi^T$. The second term in eqn. (A23) is

$$\left(\mathbf{R}_b^l\mathbf{R}_t^b\,{}^t\mathbf{T}_{TB}\right)^T\left(\mathbf{R}_b^l\mathbf{R}_t^b\,{}^t\mathbf{r}\right)-\left(\mathbf{R}_b^l\widehat{\mathbf{R}}_t^b\,{}^t\widehat{\mathbf{T}}_{TB}\right)^T\left(\mathbf{R}_b^l\widehat{\mathbf{R}}_t^b\,{}^t\mathbf{r}\right)$$

$$=\left(\mathbf{R}_b^l\widehat{\mathbf{R}}_t^b(\mathbf{I}-[\boldsymbol{\rho}\times])\left({}^t\widehat{\mathbf{T}}_{TB}+{}^t\delta\mathbf{T}_{TB}\right)\right)^T\left(\mathbf{R}_b^l\widehat{\mathbf{R}}_t^b(\mathbf{I}-[\boldsymbol{\rho}\times])^t\mathbf{r}\right)$$

$$-\left(\mathbf{R}_b^l\widehat{\mathbf{R}}_t^b\,{}^t\widehat{\mathbf{T}}_{TB}\right)^T\left(\mathbf{R}_b^l\widehat{\mathbf{R}}_t^{bt}\mathbf{r}\right)$$

$$=\left(\mathbf{R}_b^l\widehat{\mathbf{R}}_t^b[\,{}^t\widehat{\mathbf{T}}_{TB}\times]\boldsymbol{\rho}\right)^T\left(\mathbf{R}_b^l\widehat{\mathbf{R}}_t^{bt}\mathbf{r}\right)+\left(\mathbf{R}_b^l\widehat{\mathbf{R}}_t^b\,{}^t\delta\mathbf{T}_{TB}\right)^T\left(\mathbf{R}_b^l\widehat{\mathbf{R}}_t^{bt}\mathbf{r}\right)$$

$$+\left(\mathbf{R}_b^l\widehat{\mathbf{R}}_t^b\,{}^t\widehat{\mathbf{T}}_{TB}\right)^T\left(\mathbf{R}_b^l\widehat{\mathbf{R}}_t^b[\,{}^t\mathbf{r}\times]\boldsymbol{\rho}\right)$$

$$=\left(\mathbf{R}_b^l\widehat{\mathbf{R}}_t^{bt}\mathbf{r}\right)^T\left(\mathbf{R}_b^l\widehat{\mathbf{R}}_t^b[\,{}^t\widehat{\mathbf{T}}_{TB}\times]\right)\boldsymbol{\rho}+\left(\mathbf{R}_b^l\widehat{\mathbf{R}}_t^{bt}\mathbf{r}\right)^T\left(\mathbf{R}_b^l\widehat{\mathbf{R}}_t^b\right)^t\delta\mathbf{T}_{TB}+$$

$$+\left(\mathbf{R}_b^l\widehat{\mathbf{R}}_t^b\,{}^t\widehat{\mathbf{T}}_{TB}\right)^T\left(\mathbf{R}_b^l\widehat{\mathbf{R}}_t^b[\,{}^t\mathbf{r}\times]\right)\boldsymbol{\rho} \quad \text{(A26)}$$

The third term in Eqn. (A23) is

$$\left(\mathbf{R}_b^l\,{}^b\mathbf{T}_{BL}\right)^T\left(\mathbf{R}_b^l\mathbf{R}_t^b\,{}^t\mathbf{r}\right)-\left(\mathbf{R}_b^l\,{}^b\mathbf{T}_{BL}\right)^T\left(\mathbf{R}_b^l\widehat{\mathbf{R}}_t^b\,{}^t\mathbf{r}\right)$$

$$= \left(\mathbf{R}_{\mathrm{b}}^{\mathrm{l}}{}^{\mathrm{b}}\mathbf{T}_{\mathrm{BL}}\right)^{\mathrm{T}}\left(\mathbf{R}_{\mathrm{b}}^{\mathrm{l}}\widehat{\mathbf{R}}_{\mathrm{t}}^{\mathrm{b}}(\mathbf{I}-[\boldsymbol{\rho}\times])^{\mathrm{t}}\mathbf{r}\right)-\left(\mathbf{R}_{\mathrm{b}}^{\mathrm{l}}{}^{\mathrm{b}}\mathbf{T}_{\mathrm{BL}}\right)^{\mathrm{T}}\left(\mathbf{R}_{\mathrm{b}}^{\mathrm{l}}\widehat{\mathbf{R}}_{\mathrm{t}}^{\mathrm{b}\,\mathrm{t}}\mathbf{r}\right)$$

$$= \left(\mathbf{R}_{\mathrm{b}}^{\mathrm{l}}{}^{\mathrm{b}}\mathbf{T}_{\mathrm{BL}}\right)^{\mathrm{T}}\left(\mathbf{R}_{\mathrm{b}}^{\mathrm{l}}\widehat{\mathbf{R}}_{\mathrm{t}}^{\mathrm{b}}[^{\mathrm{t}}\mathbf{r}\times]\right)\boldsymbol{\rho} \tag{A27}$$

From Eqns. (A23)-(A27),

$$\delta\mathbf{K} = \left(\mathbf{R}_{\mathrm{b}}^{\mathrm{l}}\widehat{\mathbf{R}}_{\mathrm{t}}^{\mathrm{b}\,\mathrm{t}}\mathbf{r}\right)^{\mathrm{T}}\left(\mathbf{R}_{\mathrm{b}}^{\mathrm{l}}\widehat{\mathbf{R}}_{\mathrm{t}}^{\mathrm{b}}\left(\left[{}^{\mathrm{t}}\widehat{\mathbf{T}}_{\mathrm{TB}}\times\right]-\left[{}^{\mathrm{t}}\mathbf{T}_{\mathrm{TP0}}\times\right]\right)\right)\boldsymbol{\rho}$$
$$-\left(\mathbf{R}_{\mathrm{b}}^{\mathrm{l}}\widehat{\mathbf{R}}_{\mathrm{t}}^{\mathrm{b}}\left({}^{\mathrm{t}}\mathbf{T}_{\mathrm{TP0}}-{}^{\mathrm{t}}\widehat{\mathbf{T}}_{\mathrm{TB}}\right)-\mathbf{R}_{\mathrm{b}}^{\mathrm{l}}{}^{\mathrm{b}}\mathbf{T}_{\mathrm{BL}}\right)^{\mathrm{T}}\left(\mathbf{R}_{\mathrm{b}}^{\mathrm{l}}\widehat{\mathbf{R}}_{\mathrm{t}}^{\mathrm{b}}[^{\mathrm{t}}\mathbf{r}\times]\right)\boldsymbol{\rho}+\left(\mathbf{R}_{\mathrm{b}}^{\mathrm{l}}\widehat{\mathbf{R}}_{\mathrm{t}}^{\mathrm{b}\,\mathrm{t}}\mathbf{r}\right)^{\mathrm{T}}\left(\mathbf{R}_{\mathrm{b}}^{\mathrm{l}}\widehat{\mathbf{R}}_{\mathrm{t}}^{\mathrm{b}}\right){}^{\mathrm{t}}\delta\mathbf{T}_{\mathrm{TB}} \tag{A28}$$

So from Eqns. (A20), (A21), (A22) and (A28),

$$\partial \mathrm{y}_{\mathrm{L1}} = \frac{\frac{\partial\mathbf{K}}{\partial\mathbf{x}}\mathbf{G}-\frac{\partial\mathbf{G}}{\partial\mathbf{x}}\mathbf{K}}{\mathbf{G}^2}$$

$$= \left(\frac{\left(\mathbf{R}_{\mathrm{b}}^{\mathrm{l}}\widehat{\mathbf{R}}_{\mathrm{t}}^{\mathrm{b}\,\mathrm{t}}\mathbf{r}\right)^{\mathrm{T}}\left(\mathbf{R}_{\mathrm{b}}^{\mathrm{l}}\widehat{\mathbf{R}}_{\mathrm{t}}^{\mathrm{b}}\left(\left[{}^{\mathrm{t}}\widehat{\mathbf{T}}_{\mathrm{TB}}\times\right]-\left[{}^{\mathrm{t}}\mathbf{T}_{\mathrm{TP0}}\times\right]\right)\right)-\left(\mathbf{R}_{\mathrm{b}}^{\mathrm{l}}\widehat{\mathbf{R}}_{\mathrm{t}}^{\mathrm{b}}\left({}^{\mathrm{t}}\mathbf{T}_{\mathrm{TP0}}-{}^{\mathrm{t}}\widehat{\mathbf{T}}_{\mathrm{TB}}\right)-\mathbf{R}_{\mathrm{b}}^{\mathrm{l}}{}^{\mathrm{b}}\mathbf{T}_{\mathrm{BL}}\right)^{\mathrm{T}}\left(\mathbf{R}_{\mathrm{b}}^{\mathrm{l}}\widehat{\mathbf{R}}_{\mathrm{t}}^{\mathrm{b}}[^{\mathrm{t}}\mathbf{r}\times]\right)}{\mathbf{G}}\right.$$

$$\left.-\frac{\mathbf{K}}{\mathbf{G}^2}\mathbf{e}_1^{\mathrm{T}}\mathbf{R}_{\mathrm{b}}^{\mathrm{l}}\widehat{\mathbf{R}}_{\mathrm{t}}^{\mathrm{b}}[^{\mathrm{t}}\mathbf{r}\times]\right)\boldsymbol{\rho}+\frac{\left(\mathbf{R}_{\mathrm{b}}^{\mathrm{l}}\widehat{\mathbf{R}}_{\mathrm{t}}^{\mathrm{b}\,\mathrm{t}}\mathbf{r}\right)^{\mathrm{T}}\left(\mathbf{R}_{\mathrm{b}}^{\mathrm{l}}\widehat{\mathbf{R}}_{\mathrm{t}}^{\mathrm{b}}\right)}{\mathbf{G}}\,{}^{\mathrm{t}}\delta\mathbf{T}_{\mathrm{TB}} \tag{A29}$$

$$\partial \mathrm{y}_{\mathrm{L2}} = \frac{\frac{\partial\mathbf{J}}{\partial\mathbf{x}}\mathbf{G}-\frac{\partial\mathbf{G}}{\partial\mathbf{x}}\mathbf{J}}{\mathbf{G}^2}$$

$$= \left(\frac{\mathbf{e}_2^{\mathrm{T}}}{\mathbf{G}}-\frac{\mathbf{J}}{\mathbf{G}^2}\mathbf{e}_1^{\mathrm{T}}\right)\mathbf{R}_{\mathrm{b}}^{\mathrm{l}}\widehat{\mathbf{R}}_{\mathrm{t}}^{\mathrm{b}}[^{\mathrm{t}}\mathbf{r}\times]\boldsymbol{\rho} \tag{A30}$$