Research on Traffic Object 3D Pose Estimation Integrated with Prior Knowledge

崔志超

A Thesis submitted to Tokushima University
in partial fulfillment of the requirements
for the degree of Doctor of Philosophy

March 2020

Tokushima University
Graduate School of Advanced Technology and Science
Information Science and Intelligent Systems

Contents

List of Figures

List of Tables

Acknowledgment

Here, I would like to give sincere thanks to those who help me in the aspects of my life and research study during the period of my Tokushima's Ph.D program. To begin with, I want to thank to my supervisor, professor Fuji Ren, director of the Tokushima University Ren Laboratory for giving me great supports in my researches. With his patient guidances and meticulous cares, I can successfully finish my Ph.D program and experience the Japanese culture. Furthermore, Prof. Ren often provided the chances to attend the international conference to broaden my horizons and increase my knowledge.

Next, I want to appreciate Dr. Shun Nishide, Dr. Xin Kang and Ms. Asada for their helps in my individual affairs. Especially for Dr. Shun Nishide, we often discussed and exercised, namely play the badminton or table tennis, together, which not only reduces my pressures but also has benefits for study and health. Besides, Dr. Xin Kang and Ms. Asada often helped me with my life and school affairs.

Next, I appreciate my parents for their great supports and understanding. During the period of the doctoral course, they encourage me and provide all kinds of helps as far as possible. Although they live far away from Japan, I still feel warm and encouraged all the time.

Then, I am deeply thankful for my friends and research mates (Ning Liu, Chao Li, Xudong Zhang, Duo Feng, Mengjia He, Jiahui Xue, Siyuan Xue, Xin Lu, Ming Zou, QianLu Huang, Qian Zhang, Kai Yang, Shuda Xing). They gave me supports, loves, helps, cares, discussion, accompaniments during the two years' Japanese life. It is interesting to understand the thoughts, cultures and customs from different people and cities.

Finally, I give thanks to myself. I thank to myself for selecting this research road and continuously giving myself courages to overcome the difficulty and move ahead. I believe this research experience will be one of most great wealth in my future life.

Abstract

Pose estimation is an important research field of computer vision, which has contributed to many domains, such as industry, medical care, education, and autonomous driving. This thesis aims to investigate the problem of pose estimation of traffic objects (i.e. traffic sign, road) from the image sequences, which can be applied in the application of unmanned vehicle technologies or traffic scenario construction. The general methods for pose estimation are often through establishing a map between input images and the pose. This thesis explores the space features of specific kinds of traffic objects. These features are utilized as constraints for pose estimation of corresponding objects. Thus, the methods are designed for specific traffic objects and details are as follows:

Firstly, the homography-based method for traffic sign pose estimation is put forward . This method begins with acquiring robust feature correspondences based on homography constraints from image pairs. Then the objective function is designed to integrate the feature correspondences to optimize the parameters of the traffic sign plane in the 3D coordinate. Finally, the sign plane is utilized for attribute estimation (i.e pose estimation). In addition, we provide an extension for the raw KITTI dataset, which can be utilized for 3D tasks of traffic sign localization and pose estimation. In experiments, three state-of-the-art methods are employed for comparisons based on the publicly available BelgiumTS and KITTI dataset.

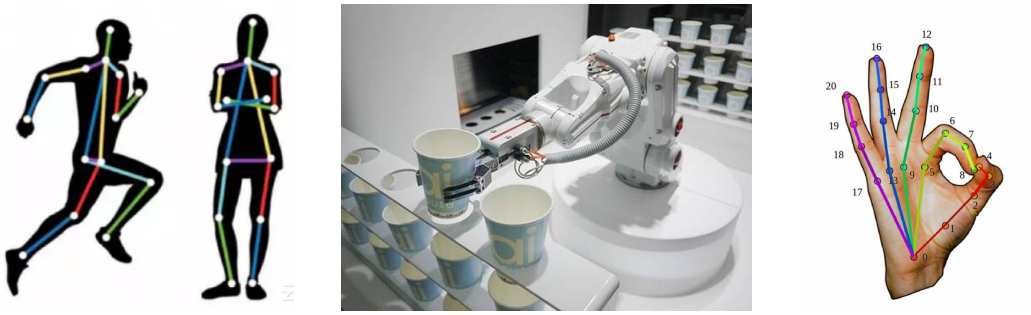
Subsequently, the 3D-2D registration method is proposed for road pose estimation. Before this method, we provide a convenient and free way to generate the 3D wire-frame road model from the Google Earth software. The 3D points from road model and the corresponding lines on the images are utilized to establish the correspondences. The registration method estimates the pose of the road model through two coarse-to-fine stages. The first stage estimates the coarse pose through exploiting the ICP to match the viewpoint and road model. The second stage utilizes the objective function to combine the point-to-line correspondences for refining pose. The both quantitative and qualitative experiments prove the effectiveness and superiority by comparison with state-of-the-art methods.

Finally, based on the 3D-2D registration method, we import the central line of a road to replace the wire-frame road model. Thus, a central-line based method is proposed for reconstructing the road structure and estimating the road pose simultaneously. Before this method, the parameterized road representation is achieved based on the central line of the road from the Google Earth. Meanwhile, the correspondences are established based on the central-line road model. The objective function utilizes the correspondences to estimate the width and pose of the road. The experiments on the public dataset prove the effectiveness of our method.

Chapter 1

Introduction

Pose estimation is an old, traditional, classical but useful research field in the computer vision community. The applications of this research direction have covered many industrial domains, such as intelligent robots, self-driving cars. The definitions of pose estimation problem are various with the specific tasks. Since the poses of objects have plenty applications in the various industrial engineering, the poses have different definitions and understanding. For example, as shown in Fig. 1.1(a), the investigation on human pose, which utilizes the articulations and its connection for representation, is basic for action recognition, understanding and forecast. Besides, human pose estimation can be applied in action imitation and interaction for intelligent robots. Furthermore, in the subdivision field, the hand pose, expressed as finger knuckles in Fig. 1.1(c), is a crucial fashion for human-computer interaction. Distinguished from aforementioned tasks, the pose estimation in the measurement field is to estimate the object's position and orientation. Figure 1.1(b) displays the robotic manipulator to grasp the paper cup. This manipulation need to acquire the accurate object position and orientation before. Thus, the pose estimation task has various definitions according to the specific tasks. Among these various definitions, this thesis mainly estimate the 6-DoF pose of traffic objects, which means to obtain the object's position and orientation. The following parts will introduce the research content of thesis (including definition of the 6-DoF pose and oriented objects), the application and significance.



(a) Human Pose (b) Object Pose (c) Hand Pose

Figure 1.1: The pose estimation tasks in various fields

1.1 Research Content

1.1.1 6DoF Pose Definition

As stated above, this thesis aims to estimate the 6-DoF pose (position and orientation) of traffic objects from traffic images. Here, we first introduce what is the 6-DoF pose in the computer vision. In physics, we often utilize the centroid to represent an object in the three dimension with a prerequisite of regardless of its shape. Hence, the centroid is generally regarded as the position of an object. For the three dimension space, the $3-vector$ is enough to represent a position, which also means the position is determined by a 3-DoF vector. When the shape and structure of an object are taken into consideration, the orientation is regarded as pitch, yaw and roll. Hence, the orientation of an object is also determined by a 3-DoF vector. Combination of the position and orientation, any pose of an object can be determined by a 6-DoF vector. As shown in Fig. 1.2, the object has pitch, yaw and roll degree.

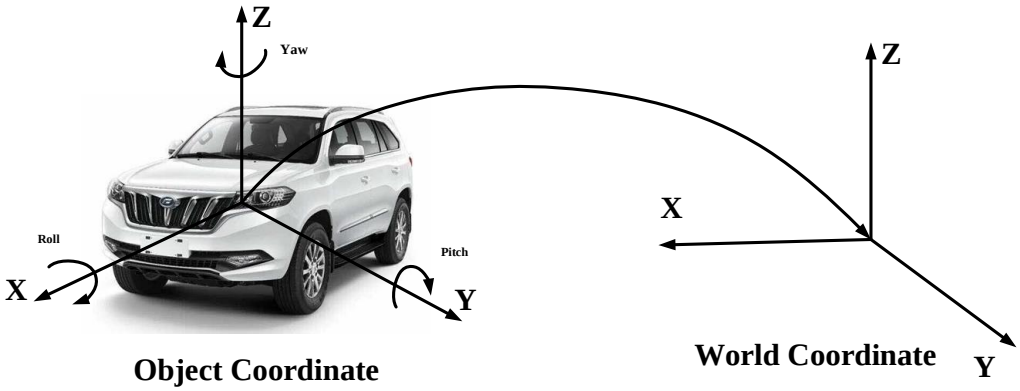


Figure 1.2: An example for explaining the pose estimation problem

Generally, the position and orientation (6DoF) of an object is relative to a coordinate. Without loss of generation, any rigid object can be represented in a measurement system, which is called object coordinate (Fig. 1.2). This coordinate is changed with the object, which results in any point in that the object is relatively static in the object coordinate. Meanwhile, the pose estimation is converted to be the relationship between the object coordinate and world coordinate. Generally, there exists some prevalent and effective ways to describe this relationship.

(1) **Matrix**: Since the object changes with the object coordinate, the pose estimation can be regarded as the transformation between object and world coordinate. Generally, the transformation between two coordinates is determined by a rotation matrix $\mathbf{R}$ and translation vector $\mathbf{t}$. However, since the matrix $\mathbf{R}$ contains nine variables but only determines by three degree of freedom. Besides, rotation obey orthogonal property, which results in complex solution process.

(2) **Euler Angles**: combination of euler angle and translation vector can completely determine the transformation between two coordinate. Similar to the pitch, yaw and roll degree, the rotation matrix can be determined by three angles rotating along any three corresponding orthogonal axes. For example, the rotation matrix $\mathbf{R}$ can be calculated by the euler angle (α, θ and β). However, the same rotation matrix has many solutions of eular angle, which also called gimbal lock. Hence, utilizing function or mapping to determine the eular angle sometimes can not work without some constraints.

(3) **Quaternions**: Quaternions utilize the complex number to represent a point and rotation axis. The rotation operation are executed in the complex domain, which solves the gimbal lock problem.

(4) **Lie group**: Lie group is to map the rotation matrix and translation vector to Lie group. Besides, the Lie group and matrix can be converted into each other. In the Lie group, series of operations were proposed instead of the matrix operations.

Besides these, there also other ways (such as Rodrigues [19], which combines the rotated axis and angle) to express the transformation between coordinates. Hence, pose estimation

between two coordinates need to be calculated according to the specific form.

1.1.2 Oriented Object

After the definition of the 6DoF pose, the oriented object will be introduced here. Actually, the traffic environment contains lots of traffic objects moving with their trajectories. It is hard to estimate all the objects emerging in the traffic environment. Thus, the methods of some specific objects are investigated in this thesis. In terms of a traffic environment, some objects are necessary according to the best of our knowledges. Combination of traffic roads, signs, lights constructs the basic traffic environment. Based on this environment, the moving vehicles, cyclists, pedestrians makes the traffic more complex. Thus, this thesis mainly aims to estimate the pose of traffic signs and roads.

1.2 Application and Significance

As mentioned above, the pose estimation is able to applied in the many research fields. Thus, we mainly introduce the two main significances of this thesis's study: autonomous vehicle technologies and traffic scenario augmented reality.

1.2.1 Autonomous Vehicle Technologies

With the rapid development of unmanned vehicle technologies, the related intelligent algorithms are needed to guarantee the safety of the autonomous vehicles during driving. The mainly vital algorithms for autonomous vehicles can be classified into three categories: recognition, decision and control. The basic aim of 'recognition' is to know the traffic objects of the traffic environment around the self-driving car. The 'decision' helps unmanned vehicle to design the moving trajectory including moving direction, velocity and acceleration etc. The 'Control' algorithms aim to compute the control parameters and control the autonomous vehicles for safety driving. The study of pose estimation mainly contributes to the algorithm of 'recognition' and 'control' parts. The obvious applications of pose estimation include 'route planning of the vehicle' and 'behavior analysis of traffic Objects', which are elucidated in detail as follows.

Route Planning of the Vehicle

Route planning is a vital and necessary ability for autonomous vehicles. The route planning is to design a path for the autonomous vehicle to drive along. Before route planning, the starting point and destination should be provided in advance. Since the autonomous vehicle is dynamically moving with time, continuously changing position and orientation of the autonomous vehicle should be provided for real-time route planning. Currently, there exist many ways to provide the position and orientation information. For example, the GPS with IMU equipment mounted on autonomous vehicles can acquire the position and orientation in real time. Another way is to utilize the intelligent algorithms to localize the autonomous vehicles with respective to a global map, namely estimate the poses of autonomous vehicle relative to the map. According to the sensors types, the localization algorithms are designed with the various data (such as images, 3D cloud points). Thus, the pose estimation of vehicles from images is also the basis of the route planning investigation.

Behavior Analysis of Traffic Objects

Behavior analysis of traffic objects aims to understand the object behaviors and forecast the subsequent behaviors (such as go straight, turn left, turn right). This helps the autonomous vehicles avoid collision in advance and safety driving. The behavior analysis mainly includes two researches.

The behavior recognition and analysis generally construct the mapping between object behaviors and semantic labels. Thus, the behavior representation of traffic objects is important for recognition. One way of constructing the object representation utilizes the historical pose of the object, namely, the continuous location and orientation of traffic objects establishes the moving trajectory. Since the pose estimation aims to estimate the location and orientation of traffic object, the moving trajectory of traffic objects can be obtained by the estimated pose. Thus, the accurate pose estimation makes big differences to the analysis and forecast of traffic objects.

1.2.2 Traffic Scenario Augmented Reality

As mentioned above, the autonomous vehicles need to be completely tested for safety driving. Currently, the popular fashions for testing include field test and simulation test.

The field test is to construct the real traffic environment for testing the autonomous vehicles. AstaZero [3] is first test field in the world, which is located in the suburban district of Sweden. This field occupies 200 million square meter and simulates the situation of many roads in the urban cities, highways and countries. The match of Intelligent Vehicles Future Challenge (IVFC) has been hold for testing the unmanned vehicles in Changshu city of China. Besides, DARPA [75] was organized for testing unmanned vehicles through setting various missions (Fig. 1.3).

Figure 1.3: DARPA match

The field test is the most popular fashion for evaluating the ability of autonomous vehicles with the advantages of reality. However, this test fashion is easily affected by the field condition, weather and climate etc. In addition, this fashion has the limitation of high resource consumption.

With the rapid development of computer graphics, the simulation tests are gradually applied to autonomous vehicles. These fashions aim to utilize the computer graphical technologies to simulate the real traffic environments. Currently, in the simulation test, there exist two ways for evaluating the autonomous vehicles. One is construct the virtual traffic environments for testing. The another is to utilize the augmented reality method to generate traffic data. These two aspects are individual and related with each others. The details of each aspect are elaborated as follows.

Figure 1.4: The simulated traffic environment constructed by PreScan software

Virtual Traffic Environment

Recently, researchers attempt to utilize the computer technologies to simulate the real traffic environments. The basic requirement is to construct the virtual dynamic traffic including scene construction, traffic element (vehicle, cyclist) behavior simulation and multi-sensor (camera, lidar) simulation. The famous example is PreScan software [65], which provides the functions of 'environment construction', algorithm test, road test etc. Meanwhile, the PreScan software support further development in the Matlab. Figure 1.4 displays traffic scenarios in the PreScan software.

Besides, Google has developed a Matrix-style system [57], which simulates the road environment of California by importing the road maps. The Matrix-style system is utilized for testing the autonomous vehicles before driving on the road. Recently, the Matlab platform released the 'Automated Driving System Toolbox' in the 2017a edition. This toolbox includes scenario generation, sensor simulation, algorithm validation, ground truth labeling and visualization.

In order to simulate a realistic traffic scenario, the traffic condition should obey the real traffic environment. One of an importation traffic condition includes the numbers,

types, features and poses(position and orientation) of objects in the real traffic scenario. Generally, these information can be obtained from the multi-modal data of the traffic environment. Thus, the pose estimation of each traffic objects is also the parts of traffic environment simulation. Since the traffic environment is the big, complex, dynamical collections of traffic objects, the objects can be classified into two categories (i.e. moving and static objects) roughly. With respect to the static objects (traffic sign, static vehicle etc.), plenty of researchers aim to estimate the pose of these static objects utilizing some specific feature of the objects. Meanwhile, as for the moving traffic objects, it is somewhat different from the static objects. Since the poses of moving objects are continuously changing, the poses of objects are need to be estimated and generated the continuous trajectory. Thus, the pose estimation is basis of traffic environment simulation.

Augmented Reality

The evaluations of autonomous vehicles is vital to guarantee the safety driving before putting into production. One of a prevalent and convenient ways for evaluating autonomous vehicles is utilizing the multi-modal data captured in the real traffic scenarios to test the related algorithms. The existing famous open-sourced traffic datasets include KITTI [29], ApolloScape [41], CityScape [20] etc. However, these datasets hardly overlap the all situations of the traffic scenarios, which can not test the algorithms completely.

The augmented reality methods are utilized for generating the newly traffic data, specifically traffic images. The advantage of augmented reality is that the traffic scenarios can be designed according to test requirements. The aim of augmented reality is to seamlessly insert the virtual object into the scene. Thus, the vital component of augmented reality is to acquire the accurate pose of traffic scenarios and elements (such as traffic sign, vehicle, cyclist). The pose estimation of traffic objects can be applied in the traffic augmented reality for synthesizing new images.

1.3 Thesis Organization

This thesis mainly aims to investigate the problem of pose estimation of traffic objects (i.e. traffic sign and road) from images. According to various traffic objects, the corresponding

methods are designed, Meanwhile, the experiments results and discussion is displayed in this thesis. The details of the rest are shown as follows.

Chapter 2: aim to investigate the pose estimation methods for traffic sign. Meanwhile, a dataset for evaluating the pose estimation algorithm of traffic sign is proposed. The comparative experiments and analysis are displayed in this chapter.

Chapter 3: the pose estimation method of road model is presented utilizing the 3D-2D registration. Besides, a convenient fashion for making road model is displayed. Finally, the qualitative and quantitative experimental results are displayed and discussed.

Chapter 4: In order to simplify the process of road generation, the central line parameterized model is utilized to represent a road. Meanwhile, the objective function is displayed for not only determining the road parameters and pose simultaneously. Finally, the experimental results are displayed in the end.

Chapter 5: conclude the whole thesis and discuss the future works.

Chapter 2

Homography-Based Traffic Sign Pose Estimation

This chapter focuses on the pose estimation of traffic signs. Taking in account the specific shapes of traffic signs, we explore how to utilize this shape information in the pose estimation. Based on the homography constraint, the objective function is proposed for pose estimation. Besides, aiming at lack of the effective dataset for evaluation, we propose the corresponding dataset (KITTI^{+}) based on the KITTI. Subsequently, according to the experimental results, the effectiveness of our method is discussed and analyzed. Finally, we summarize this chapter.

2.1 Introduction

In the past, most of researches pay attention to the traffic sign recognition (TSR) task, which has great influence on driving assistance system (DAS) [71], driver decision making of unmanned vehicle [94], path planning for unmanned vehicles [89] and traffic sign management etc. Nowadays such much progress has been achieved that some algorithms can approximately reach to 100% recognition accuracy [60]. However, with the further study, more researchers realize that combination of content and position information of traffic signs is of great importance, and the study of sign localization have the following significances:

(1) Some digit maps (such as Here), which autonomous vehicles rely on for navigation

or route planing, are sometimes outdated or unavailable. In this condition the 3D position information of signs can be supplementary to the dynamic map for DAS [91].

(2) 3D positions of traffic signs help autonomous vehicles with navigation and safety driving [43]. For instance, the semantic information and 3D position of guide traffic signs contribute to accurate navigation. 3D Positions information of some warning signs, such as falling rocks, slippery road, warn drivers the dangerous position and make the vehicles avoid dangers.

(3) 3D Position information of traffic signs and their contents can be utilize for performing and maintaining the asset inventory [84].

Currently, many vision-based methods [91, 79, 36] has been proposed for estimating the position of traffic signs. With further study, some researchers, such as Hu *et al.* [39, 40] realize only 3D position information is not enough for road signs inventory, and extends the position to sign attributes (including position, distance, height, title angle and shape) computation, which is overall but redundant.

In this chapter, we first define the space attributes of a traffic sign as 3D position (the centroid of traffic sign), pose (the normal vector of traffic sign plane), which is slightly different from the pose definition in the Sect.1.1.1. Since the traffic signs are almost located on a space plane in the 3D coordinate, the normal vector of the plane rather than the object coordinate (in Sect.1.1.1) is enough to represent the pose. The designed method aims to estimate traffic sign attributes from image sequences by monocular or binocular cameras. Different from previous methods which compute attributes directly, the proposed method estimates the attributes through the 3D sign plane. As displayed in Fig. 3.1, our method first builds effective feature correspondences from image pairs. Then the objective function exploits obtained feature correspondences and homography constraints between multi-view images to estimate the 3D plane which traffic signs are located on. Based on the estimated parameters of 3D plane, the attributes of signs are determined finally.

Reviewing the experiment parts of previously related works, the limitation and weakness can be summarized in three cases: 1) some researches lack experiments to evaluate the accuracy of their methods. 2) some researchers [36, 40, 84, 91] exploited their own vehicle-mounted multi sensors including cameras, GPS etc. to perform field tests. However, the

field tests are high-cost and demand lots of resources. 3) some researchers [76] conducted the experiments through simulation in the visual space of the computer. Nevertheless, computer simulation experiments often simplify the real complex traffic environments, which leads to the unreal performance. In view of these shortages, A public dataset, called BelgiumTS, for traffic sign detection and localization task was first proposed and the relevant experiments was conducted by Timofte *et al.* [79]. Indeed, Belgium dataset [79] only has the ground truth of 3D position. Hence, this chapter proposes a series of processing to provide an extension to the KITTI dataset [30]. After processing, this dataset can be applied to the tasks of traffic sign position and pose evaluation. In experiments, we adopt both BelgiumTS dataset [79] and KITTI dataset [30] from the real traffic environment to conduct the comparative experiments with sate-of-the-art methods. The experiments on the dataset from real traffic environment not only reflects the real performance of methods, but also saves resources.

2.2 Related Works

According to existing relevant works, the vision-based methods for the 3D object localization and pose estimation can be classified into two categories according to whether providing the prior knowledge or not, namely bottom-to-up and top-down methods. The main idea of top-down methods, such as [72], is to project the 3D object model to the image and localize through guaranteeing the consistency between the projected region and the object region in the image. For instance, PWP3D method [68] models the region of the 3D object by level-set model for pose estimation. Zeeshan Zia *et al.* [99] utilized a class of models to achieve 3D localization of cars in the image sequences. However, the model-based methods often need many constrains to achieve good performance. Concretely, the 3D model has the same shape, scale with the real object as much as possible. In reality, it's hard to find the totally same object model with that of the image, especially for large-scale scenes. Due to the limitation of acquiring real object models, many bottom-to-up methods are advocated for supplements. Among these methods, representative cases generally utilize bottom features, such as points, lines and textures, to estimate the 3D position and pose of objects. For example, Bao *et al.* [5] proposed a semantic SfM method for both

recognition and localization of points, regions and objects. Song *et al.* [77] combined the SfM framework and detection cues to localize vehicles from traffic environmental videos.

In terms of traffic sign localization and pose estimation, research [69] utilizes the 3D model to estimate the poses of traffic signs based on the PWP3D method [68]. Krsák *et al.* [44] proposed a TSR system with an algorithm to estimate the 3D position of the traffic sign by the mobile vehicle location received from GPS. In actual, the long distance between the vehicle and sign leads to inaccurate 3D localization. André Welzel *et al.* [91] proposed two methods to estimate the 3D position of the sign, which need hypothesize the height of the traffic sign as priori knowledges. Wang *et al.* [84] provided a stereo-vision based method which can apply to the survey vehicle with either monocular or binocular instruments. Hu *et al.* [40] developed the homography-based method to estimate the attributes (height, title, distance) of a traffic sign. Hu *et al.* [39] recovered the pavement plane for traffic sign localization. Balali *et al.* [4] utilized the SIFT [54] feature correspondences to triangulate sparse points and expand them to dense point cloud for localization. Hazelhoff *et al.* [35] established the linear view degree in horizontal and vertical direction of panoramic image and located the sign by interacting the 3D lines from multi-view images.

2.3 Proposed Method

2.3.1 Feature Extraction

The study of the feature detection and descriptor matching has been well developed in the computer vision community for many years. Many famous feature detection and matching methods [87, 85, 88, 86] have emerged for various vision tasks. Although existing features and descriptors, such as SIFT [54], SURF [7] and BRISK [48], can overcome the influence of illumination, rotation and scale variance, it is hard to apply these feature detection methods to the task of attributes estimation directly. The reasons are as follows: 1) Most of the traffic sign regions from the whole images are of low resolution, which results in extracting a small amount of feature points. 2) In spite of existing the high matching score of two descriptors, some wrong matching between two features still occur. As the yellow points shown in the Fig. 2, the wrong feature correspondences laying the symmetrical positions also have the similar descriptors. 3) As the red feature point in Fig. 2.1, some

features points detected in the background are useless and hard to be removed.

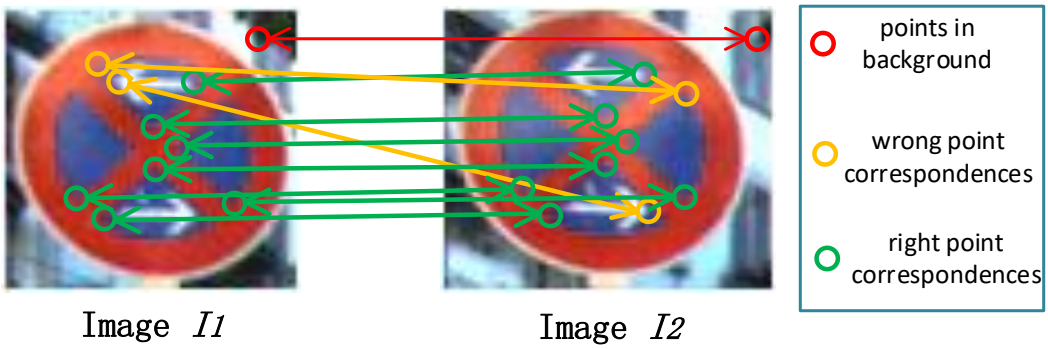


Figure 2.1: An instance of feature correspondence establishment

In order to overcome these problems, we first utilize both SIFT [54] and SURF [7] features as well as their descriptors to increase the number of feature points. In the matching stage, the best feature correspondences are established through two steps. 1) The epipolar constraint is utilized to acquire feature correspondences for each point, which can eliminate the influence of the symmetrical shape. Given the ith feature points $\mathbf{p}_{I_1}^i$ from image I_1, the KNN method [9] is utilized to find the k feature candidates $\{\mathbf{p}_{I_2}^{j_1}, ..., \mathbf{p}_{I_2}^{j_k}\}$ and k corresponding matching scores in the image I_2. Then the fundamental matrix $\mathbf{F}_{I_1 I_2}$ can be calculated by intrinsic and extrinsic parameters of two image views [34] and the epipolar lines $\mathbf{l}_{I_2}^i$ on image I_2 can be estimated by $\mathbf{F}_{I_1 I_2}\mathbf{p}_{I_1}^i$ [34]. The feature point, which has the least distance to the line $\mathbf{l}_{I_2}^i$, is the best correspondence. 2) The homography constraint is employed to eliminate the feature correspondence in the background. After acquiring lots of feature correspondences, the homography matrix $\mathbf{H}_{I_1 I_2}$ between images I_1 and I_2 is calculated by the RANSAC (Random Sample Consensus) algorithm [81]. Finally, the homography matrix is employed to shift the feature correspondences and eliminate the outliers simultaneously.

2.3.2 Objective Function

In this section, objective function is constructed to integrate the plane $\mathbf{\Pi} = \begin{bmatrix}\mathbf{n}^\top & d\end{bmatrix}^\top$ and feature correspondences for traffic sign plane estimation. The objective function will be deduced through three parts: homography matrix deduction, objective function construction and the optimization.

Homography matrix deduction

The homography matrix induced by a space plane $\mathbf{\Pi}$ indicates the relationship between two image views. In this case, the feature correspondence ($\mathbf{p}_i$, $\mathbf{p}_j$) from the image pair (i, j) is projected by a 3D point $\mathbf{P}$ in the traffic sign and satisfies the homography constraint. Given the rotation matrix $\mathbf{R}_i$, transform vector $\mathbf{t}_i$ and intrinsic matrix $\mathbf{K}_i$ parameters of image i, the line $\mathbf{p}(\gamma)$ passing through optical center of camera, $\mathbf{p}_i$ and $\mathbf{P}$ can be determined with a parameter γ in Eq. (2.1) [34]. As a result, any point in the line $\mathbf{p}(\gamma)$ can be expressed with γ.

$$\mathbf{p}(\gamma) = \begin{bmatrix} \mathbf{R}_i^{-1}\mathbf{K}_i^{-1}\mathbf{p}_i \\ 0 \end{bmatrix} + \gamma \begin{bmatrix} \text{-}\mathbf{R}_i^{-1}\mathbf{t}_i \\ 1 \end{bmatrix} \tag{2.1}$$

According to the constraint that the 3D point $\mathbf{P}$ is located in plane $\mathbf{\Pi}$, the parameter γ_0 can be determined by $\begin{bmatrix} \mathbf{n}^\top & d \end{bmatrix} \mathbf{p}(\gamma) = 0$. Through replacing the γ with γ_0 in Eq. (2.1), the point $\mathbf{P}$ can be expressed in Eq. (2.2).

$$\mathbf{P} = \begin{bmatrix} \mathbf{R}_i^{-1}\mathbf{K}_i^{-1} - \frac{\mathbf{R}_i^{-1}\mathbf{t}_i\mathbf{n}^\top\mathbf{R}_i^{-1}\mathbf{K}_i^{-1}}{\mathbf{n}^\top\mathbf{R}_i^{-1}\mathbf{t}_i - d} \\ \frac{\mathbf{n}^\top\mathbf{R}_i^{-1}\mathbf{K}_i^{-1}}{\mathbf{n}^\top\mathbf{R}_i^{-1}\mathbf{t}_i - d} \end{bmatrix} \mathbf{p}_i \tag{2.2}$$

After acquiring the point $\mathbf{P}$ by Eq. (2.2), we project the point $\mathbf{P}$ into the image j according to the camera parameters (rotation matrix $\mathbf{R}_j$, transform vector $\mathbf{t}_j$ and intrinsic matrix $\mathbf{K}_j$). The final relationship between $\mathbf{p}_i$ and $\mathbf{p}_j$ from images i and j is shown in the Eq. (2.3). Therefore, as shown in Eq. (2.4), it is obvious that the homography matrix $\mathbf{H}_{ij}$ between two images pairs is only determined by the sign plane $\mathbf{\Pi}$ by providing the camera parameters.

$$\mathbf{p}_j = \mathbf{K}_j(\mathbf{R}_j\mathbf{R}_i^{-1} + \frac{(\mathbf{R}_j\mathbf{R}_i^{-1}\mathbf{t}_i - \mathbf{t}_j)\mathbf{n}^\top\mathbf{R}_i^{-1}}{\mathbf{n}^\top\mathbf{R}_i^{-1}\mathbf{t}_i - d})\mathbf{K}_i^{-1}\mathbf{p}_i \tag{2.3}$$

$$\mathbf{H}_{ij}(\mathbf{\Pi}) = \mathbf{K}_j(\mathbf{R}_j\mathbf{R}_i^{-1} + \frac{(\mathbf{R}_j\mathbf{R}_i^{-1}\mathbf{t}_i - \mathbf{t}_j)\mathbf{n}^\top\mathbf{R}_i^{-1}}{\mathbf{n}^\top\mathbf{R}_i^{-1}\mathbf{t}_i - d})\mathbf{K}_i^{-1} \tag{2.4}$$

Objective function construction

The objective function is based on feature correspondences from image pairs, the homography matrices, and plane parameter. Here, we have a set of the image pairs, denoted as $\{1, 2, \cdots, S-1, S\}$. A set of corresponding homography matrices $\mathbb{H} = \{\mathbf{H}_1(\mathbf{\Pi}), \mathbf{H}_2(\mathbf{\Pi}), \cdots, \mathbf{H}_S(\mathbf{\Pi})\}$ between image pairs can be estimated in Sec. 2.3.1. It is obvious that all the homography matrices between different image pairs are induced by the traffic sign plane $\mathbf{\Pi}$. For any image pair s, we obtain the number of K_s feature correspondences according to Sec. 2.3.1, and points $\mathbf{p}_k^s$ and $\mathbf{p}_k^{s\prime}$ denote the kth ($0 \leq k \leq K_s$) feature correspondences of image pairs s. From the Sec. 2.3.1, the $\mathbf{p}_k^{(s)}$, $\mathbf{p}_k^{(s)\prime}$ and $\mathbf{H}_s(\mathbf{\Pi})$ satisfies the equation $\mathbf{p}_k^{(s)\prime} = \mathbf{H}_s(\mathbf{\Pi})\mathbf{p}_k^{(s)}$. However, because of existing the noise, the equation above cannot be unequal. We utilize the $\mathbf{L}_2$-norm to calculate the geometry distance between $\mathbf{p}_k^{(s)\prime}$ and $\mathbf{H}_s(\mathbf{\Pi})\mathbf{p}_k^{(s)}$. As shown in the Sec. 2.3.1, any pixel point $\mathbf{p}$ from the image is expressed in the homogeneous form. In order to eliminate the influence of z component, we import the vector $\mathbf{m} = \begin{bmatrix} 0 & 0 & 1 \end{bmatrix}^\top$ to extract the z component of the point and make sure that the value of the z component is 1, which is shown in Eq. (2.5).

$$\frac{\mathbf{H}_s(\mathbf{\Pi})\mathbf{p}_k^s}{\mathbf{m}^\top \mathbf{H}_s(\mathbf{\Pi})\mathbf{p}_k^s} \tag{2.5}$$

Then, the geometry distance of the kth feature correspondence from the image pair s are shown as follows.

$$D_k^s(\mathbf{\Pi}) = \|\mathbf{p}_k^{s\prime} - \frac{\mathbf{H}_s(\mathbf{\Pi})\mathbf{p}_k^s}{\mathbf{m}^\top \mathbf{H}_s(\mathbf{\Pi})\mathbf{p}_k^s}\| \tag{2.6}$$

In order to acquire the geometry distances of all feature correspondences, we sum the geometry distances of all feature correspondences from all image pairs to formulate the objective function (Eq. (2.7)). In Eq. (2.7), the S and K_s are the number of image pairs and the number of feature correspondences of the image pair s.

$$E(\mathbf{\Pi}) = \sum_{s=1}^{S} \sum_{k=1}^{K_s} D_k^s(\mathbf{\Pi}) \tag{2.7}$$

Finally, Replacing the $D_k^s(\mathbf{\Pi})$ by Eq. (2.6) to form the final objective function in the

following Eq. (2.8), where the vector $\mathbf{n}$ in the parameter $\mathbf{\Pi}$ is constraint to the unit vector.

$$\begin{aligned} E(\mathbf{\Pi}) &= \sum_{s=1}^{S}\sum_{k=1}^{K_s}\|\mathbf{p}_k^{s\prime} - \frac{\mathbf{H}_s(\mathbf{\Pi})\mathbf{p}_k^s}{\mathbf{m}^\top\mathbf{H}_s(\mathbf{\Pi})\mathbf{p}_k^s}\| \\ & s.t. \|\mathbf{n}\| = 1 \\ where\ \mathbf{H}_s(\mathbf{\Pi}) &= \mathbf{K}_s(\mathbf{R}_s{\mathbf{R}'_s}^{-1} + \\ & \frac{(\mathbf{t}_s - \mathbf{R}_s{\mathbf{R}'_s}^{-1}\mathbf{t}'_s)\mathbf{n}^\top{\mathbf{R}'_s}^{-1}}{\mathbf{n}^\top{\mathbf{R}'_s}^{-1}\mathbf{t}'_s - d})\mathbf{K}_s^{-1} \end{aligned} \tag{2.8}$$

Optimization

The gradient descent method [64] based on interactive scheme is employed for optimization. However, since the objective function is non-convex, a good initial value and suitable step size for the objective function is must. For initial value of traffic sign plane. we estimate the 3D positions of the feature correspondences by triangle method [34]. Then, the 3D position of feature correspondences are utilized for plane fitting [81], which is taken as the initial value. The newton method and Hessian matrix estimated by "bfgs" method [18] are employed to estimate the optimization step. The details are presented in [11].

2.3.3 Attributes Recovery

After acquiring the plane parameters $\mathbf{\Pi} = \begin{bmatrix}\mathbf{n}^\top & d\end{bmatrix}^\top$. Then the attributes (position, pose) of a sign are recovered through the plane.

Position Estimation

As is defined in the introduction section, the centroid point of the traffic sign in the 3D coordinate is regarded as the 3D position. Because the centroid of a traffic sign in the 3D coordinate is projected into pixel centroid of the sign in the image. So the pixel centroid of the sign in the image is back-projected to line according to Eq. (2.1). The intersection between the back-projected line and the sign plane is the corresponding centroid point in the 3D coordinate, which is the position of the traffic sign.

Pose Estimation

The definition of the sign pose in the introduction section is not detailed, since all traffic signs have two facades: the front face with the pattern, the back face without the pattern. The normal vectors are generally perpendicular to the plane and has two opposite directions. This chapter defines that the normal vector directing to the back face of the sign is the traffic sign pose. In reality, the estimated normal vector $\mathbf{n}$ is sometimes opposite to the pose of the traffic sign. To eliminate the ambiguity, we utilize the relative relationship between $\mathbf{n}$ and the image view. Concretely, the angle α between vector $\mathbf{n}$ and z axis of camera can be adjusted to meet the property that the pattern of traffic signs can be seen in the image. Namely, if the angle α is less than $\pi/2$, the pose of the traffic sign is $-\mathbf{n}$. Otherwise, the pose is represented as $\mathbf{n}$.

2.4 KITTI+ Dataset

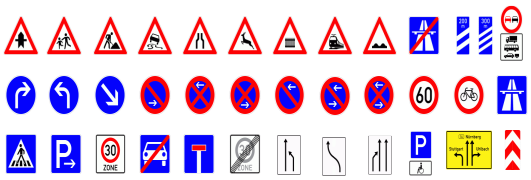
2.4.1 KITTI+ Introduction

KITTI+ dataset[1] is based on the raw KITTI dataset [31] and aim at the 3D task of the traffic signs, namely traffic sign pose, position estimation. KITTI+ dataset contains traffic signs from the various multiple views and the ground truth is extracted from the lidar data by Velodyne HDL-64E. As is shown in Table 2.1, KITTI+ dataset has 76 traffic sign instances from the "City", "Residential" and "Road" raw data of KITTI, which are also divided into ellipse, rectangle and triangle signs according to the shapes. Each traffic sign includes several original images, ROI images, ROI files, camera matrices and the ground truth. ROI images are the rectangle regions including traffic signs, which are extracted from the original images manually. ROI file contains the left-top pixel position and the size (width, height) of ROI rectangle region in the image. Camera matrices consist of the intrinsic matrix and extrinsic matrices, which are in accordance with the corresponding frames. The ground truth has the 3D position and pose of the traffic sign in the 3D coordinate. The signs of KITTI+ are shown in the "Instances" item of Table 2.1, and we utilize the signs of "Maximum Speed Limit of 60" to stand for all "Speed Limit Signs". Different from the existing traffic sign dataset, the KITTI+ dataset is new in three aspects.

[1] "https://github.com/czc-convolution/KITTIDatasaet"

i) The ground truth, namely the 3D position and pose of signs, are extracted from the lidar data of traffic environments, which has the error within centimeter. To the best of our knowledge, there has not been a 3D task dataset for traffic signs, whose ground truth is extracted from lidar data. ii) We propose series of processing methods, which can get accurate camera parameters, 3D positions and pose. iii) The KITTI+ dataset contains the diverse traffic signs in various traffic conditions, such as straight road, curved road, cross road, etc.

Table 2.1: The content of KITTI+ dataset

KITTI Dataset for Traffic Signs 3D Task						
Category:	City	Residential	Road	Ellipse	Rectangle	Triangle
Amount:	16	30	30	33	20	23
Content:	1.The original image sequences 2.Camera parameters (intrinsic and extrinsic) 3.The regions of traffic signs and their positions in the image 4.The ground truth of 3D position and pose					
Instances:						

2.4.2 Processing Method

To begin with, the coordinate concerning KITTI dataset are introduced.

(1) Original coordinate: the original coordinate is a global coordinate defined by GPS/IMU. The position and pose of the vehicle at any time are related to the original coordinate through GPS/IMU.

(2) Vehicle coordinate: the vehicle coordinate is defined by GPS/IMU and is changing with the moving vehicle. Besides, the vehicle coordinate at any time has transformation relationship with the original coordinate, which can make vehicle coordinates converted into each others.

(3) Lidar coordinate: the lidar coordinate defined by Velodyne HDL-64E describes the position and pose of lidar devices at any time. The lidar data are measured in this

coordinate.

(4) Camera coordinate: the camera coordinate is the 3D coordinate and determines the position and pose of camera at any time. The images are captured in this coordinate.

(5) World coordinate of signs: this coordinate is one of vehicle coordinates and utilized for KITTI$^+$ dataset.

(6) Sign plane coordinate: this coordinate is in the 2D sign plane, which is utilized for data generation in Sect. 2.4.2.

As is known in [31], the camera, lidar and GPS/IMU devices have been calibrated in the vehicle. The relationships between the vehicle coordinate and the original coordinate, the vehicle coordinate and lidar coordinate, the lidar coordinate and camera coordinate are shown in Eq. (2.9), (2.10), (2.11) [31]. In these equations, the $\mathbf{P}_{imu(i)}$, $\mathbf{P}_{velo(i)}$, $\mathbf{P}^{j}_{cam(i)}$ and $\mathbf{P}_{ori}$ means the point value in the vehicle coordinate, the lidar coordinate, the jth camera coordinate($j = 1, \cdots, 4$, totally four cameras in KIT vehicle) at the time of i and original coordinate. The rotation matrices $\mathbf{R}^{ori}_{imu(i)}$, $\mathbf{R}^{velo(i)}_{imu(i)}$, and transform matrices $\mathbf{t}^{ori}_{imu(i)}$, $\mathbf{t}^{velo(i)}_{imu(i)}$ are from the vehicle coordinate to the original coordinate, from the vehicle coordinate to the lidar coordinate at the i time. The conversion from the lidar coordinate to the jth camera coordinate at the time of i is through Eq. (2.11), where $\mathbf{R}^{j}_{rect(i)}$, $\mathbf{R}^{cam(i)}_{velo(i)}$and $\mathbf{t}^{j}_{rect(i)}$ $\mathbf{t}^{cam(i)}_{velo(i)}$ are rotation matrix, transform vector; $\mathbf{P}^{j}_{rect(i)}$ is camera parameter from [31]; f_u is the focal length of the camera.

$$\begin{bmatrix} \mathbf{P}_{ori} \\ 1 \end{bmatrix} = \begin{bmatrix} \mathbf{R}^{ori}_{imu(i)} & \mathbf{t}^{ori}_{imu(i)} \\ \mathbf{0} & 1 \end{bmatrix} \begin{bmatrix} \mathbf{P}_{imu(i)} \\ 1 \end{bmatrix} \tag{2.9}$$

$$\begin{bmatrix} \mathbf{P}_{velo(i)} \\ 1 \end{bmatrix} = \begin{bmatrix} \mathbf{R}^{velo(i)}_{imu(i)} & \mathbf{t}^{velo(i)}_{imu(i)} \\ \mathbf{0} & 1 \end{bmatrix} \begin{bmatrix} \mathbf{P}_{imu(i)} \\ 1 \end{bmatrix} \tag{2.10}$$

$$\begin{bmatrix} \mathbf{P}^{j}_{cam(i)} \\ 1 \end{bmatrix} = \begin{bmatrix} \mathbf{R}^{j}_{rect(i)} & \mathbf{t}^{j}_{rect(i)} \\ \mathbf{0} & 1 \end{bmatrix} \begin{bmatrix} \mathbf{R}^{cam(i)}_{velo(i)} & \mathbf{t}^{cam(i)}_{velo(i)} \\ \mathbf{0} & 1 \end{bmatrix} \begin{bmatrix} \mathbf{P}_{velo(i)} \\ 1 \end{bmatrix} \tag{2.11}$$

$$Where \qquad \mathbf{t}^{j}_{rect(i)} = \mathbf{P}^{j}_{rect(i)}(1:3,4)/f_u$$

The framework of KITTI$^+$ dataset processing is shown in Fig. 2.2, which mainly contains data selection, synchronous processing and data generation.

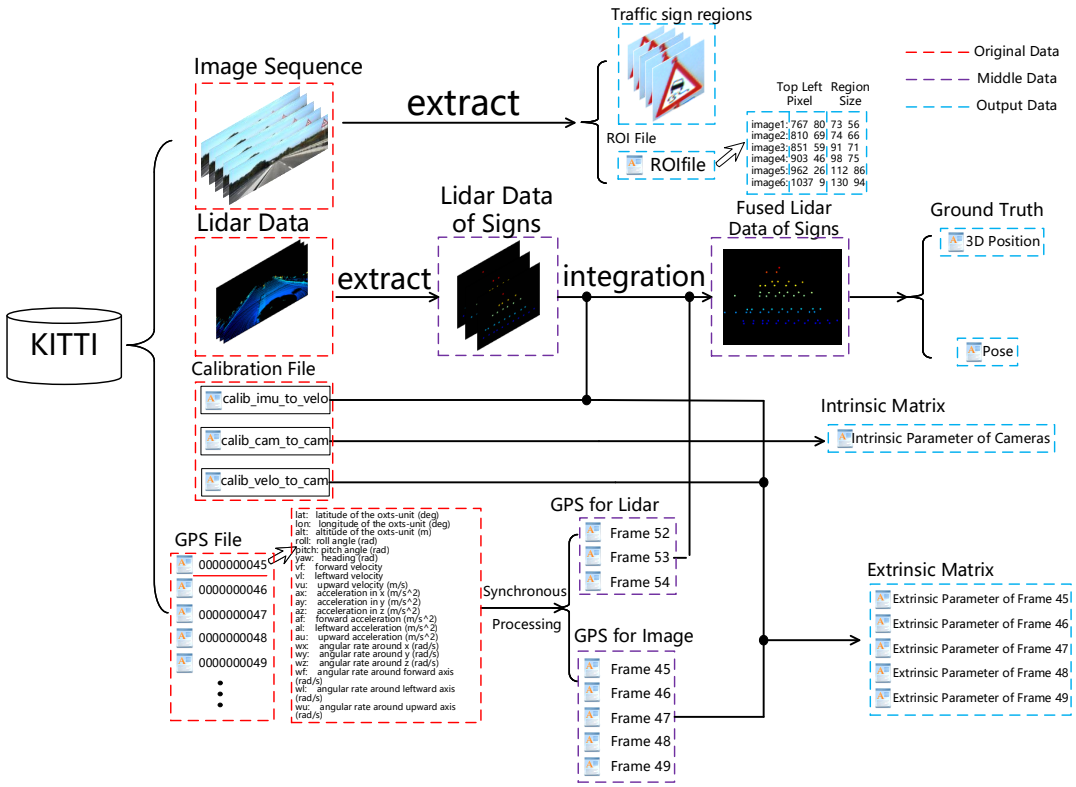


Figure 2.2: The flow chart of processing for KITTI+

Data Selection

The each rectified data from KITTI, contains four consecutive image sequences, lidar data, GPS frames from Velodyne HDL-64E and the calibrated parameters between equipped sensors, which are displayed as the red dot box in Fig. 2.2. The data selection aims to manually choose the appropriate raw data for the post processing. The procedure of selection mainly contains image selection, lidar data selection and parameters selection. For image selection, the images with content of traffic signs will be selected manually, and for each traffic sign, the number of selected images is ranged from four to eight. In terms of lidar data, in generally, the data containing the corresponding traffic signs will be selected. However, because of the limitation of the lidar field view, signs are often out of lidar range in height when the vehicle is near. When the vehicle is far, the lidar points on the sigh are too sparse to represent the signs due to the fixed scan resolution of lidar sensors. To solve this contradiction, we select three or four consecutive lidar frames, which are captured at the far distance for post processing. In this way, although each lidar frame contains sparse points of the sign, three or four sparse points from all consecutive

lidar frames can be integrated into dense one. Besides the GPS data corresponding to the image frames and lidar frames, and the calibrated matrices between sensors are chosen for post processing.

Synchronous Processing

In the beginning of Sect. 2.4.2, it has been indicated that the coordinates can be converted into each other. However, as is shown in Fig. 2.3, multi-model data, such as images, lidar and GPS, received at the different time are hard to fuse together. Therefore, synchronous processing is needed to integrate the multi-model data in the same coordinate.

The main idea of synchronous processing is to utilize existing GPS frames to interpolate the vehicle coordinate at the time of capturing the lidar frame or the image. After automatically carrying out Kalman filter [6] to optimize the GPS/IMU information in Velodyne HDL-64E device, we can acquire the parameters of *roll*, *pitch*, *yaw*, *latitude*, *longitude* and *height* etc. (shown in "GPSFile" of Fig. 2.2). The interpolation process includes two parts: pose and position.

For the pose interpolation, without loss of generality, we define the time of capturing either the lidar frame or image as t_s, and the capturing time of left and right GPS frames is t_{Gl} and t_{Gr} (shown in Fig. 2.3). We assume that the rotation angles of the vehicle are linearly changing. Therefore, the rotation angles $\mathbf{A}_{t_s}$ of the vehicle can be estimated by Eq. (2.12). Finally the rotation matrix $\mathbf{R}^{ori}_{imu(t_s)}$ can be obtained by [92].

$$\begin{aligned}
&\mathbf{A}_{t_s} = (\mathbf{A}_{Gl} - \mathbf{A}_{Gr})/(t_{Gl} - t_{Gr}) * (t_s - t_{Gr}) + \mathbf{A}_{Gr} \\
&where\ \mathbf{A}_{t_s} = \begin{bmatrix} roll_{t_s} & pitch_{t_s} & yaw_{t_s} \end{bmatrix}^\top; \\
&\qquad\qquad \mathbf{A}_{t_{Gl}} = \begin{bmatrix} roll_{t_{Gl}} & pitch_{t_{Gl}} & yaw_{t_{Gl}} \end{bmatrix}^\top; \\
&\qquad\qquad \mathbf{A}_{Gr} = \begin{bmatrix} roll_{t_{Gr}} & pitch_{t_{Gr}} & yaw_{t_{Gr}} \end{bmatrix}^\top;
\end{aligned} \tag{2.12}$$

For the position of the vehicle, we first estimate the position in the original coordinate at the time of t_{Gl} and t_{Gr}, namely $\mathbf{t}_{imu(t_{Gl})}$ and $\mathbf{t}_{imu(t_{Gr})}$ in Eq. (2.9). Besides, the speed $\mathbf{v}$ and acceleration $\mathbf{a}$ (shown in "GPSFile" of Fig. 2.2) at the time of t_{Gl} and t_{Gr} are provided by GPS/IMU devices. Thus, the approximating position $\mathbf{X}_{t_{Gl}}$, $\mathbf{X}_{t_{Gl}}$ of the vehicle can be

estimated by Eq. (2.13), where the Δt are replaced by $t_s - t_{Gr}$ and $t_s - t_{Gl}$. Finally, the position $\mathbf{t}^{ori}_{imu(t_s)}$ of vehicle in the Eq. (2.9) is the mean of $\mathbf{X}_{t_{Gl}}$ and $\mathbf{X}_{t_{Gl}}$.

$$
\begin{aligned}
\mathbf{X} &= \begin{bmatrix} \mathbf{X} & \mathbf{v} & \mathbf{a} \end{bmatrix} \begin{bmatrix} 1 \\ \Delta t \\ \frac{\Delta t^2}{2} \end{bmatrix}; \\
where\ \mathbf{X} &= \begin{bmatrix} x_f & x_l & x_u \end{bmatrix}^\top; \\
\mathbf{v} &= \begin{bmatrix} v_f & v_l & v_u \end{bmatrix}^\top \\
\mathbf{a} &= \begin{bmatrix} a_f & a_l & a_u \end{bmatrix}^\top
\end{aligned}
\tag{2.13}
$$

After acquiring the pose and position of the vehicle, the relationship between the t_s vehicle coordinate and the original coordinate can be achieved by Eq. (2.9).

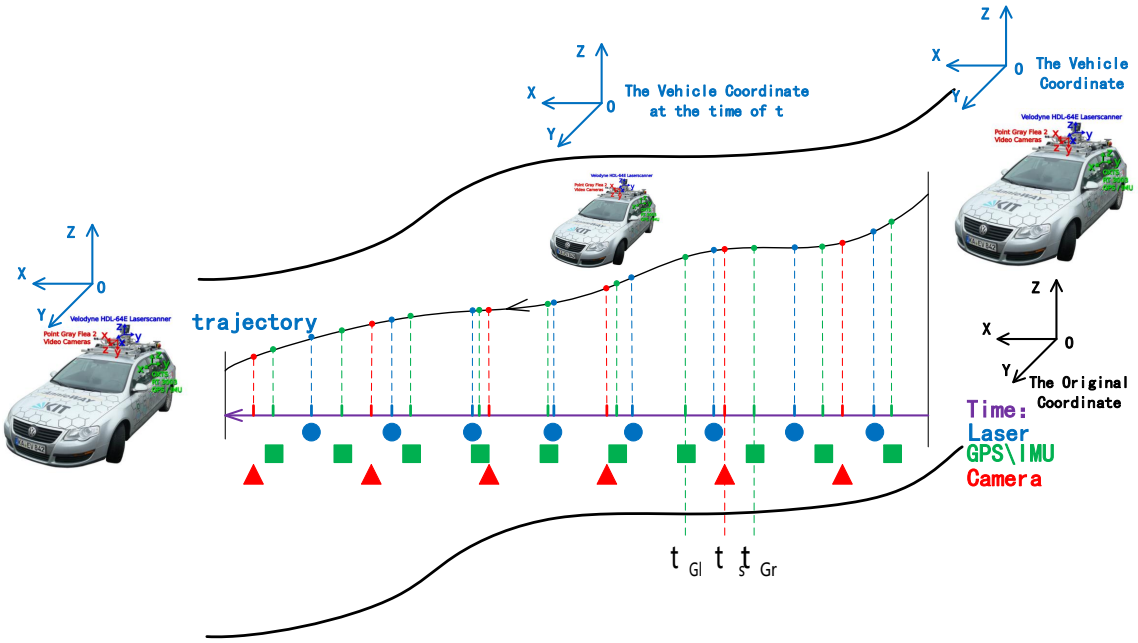


Figure 2.3: The time of capturing data by GPS, lidar and camera sensors equipped in vehicle

Dataset Generation

The procedure of dataset generation contains two parts: Input dataset generation (image sequence, ROIFile, intrinsic and extrinsic matrices in Fig. 2.2) and ground truth generation (3D position and pose in "Ground Truth" of Fig.2.2). In terms of input dataset

generation, the traffic sign regions are extracted from image sequences manually to produce the ROI images and the positions of regions in images (the position of left, top pixel and the size, i.e. width, height of the ROI region) are recorded in "ROIfile" (Fig. 2.2). For the calibration matrices, the vehicle coordinate of the first selected frame is regarded as the world coordinate, which is mentioned in Sec. 4.2. Next the rotation matrix $\mathbf{R}_{imu(i)}^{world}$ and translation vector $\mathbf{t}_{imu(i)}^{world}$ from the ith vehicle coordinate to the world coordinate are calculated by the bridge of the original coordinate in Eq. (2.9). Finally, as is shown in Eq. (2.14), the extrinsic matrices ($\mathbf{R}_{ex}$, $\mathbf{t}_{ex}$) of each frame are transformed by the bridge of the matrix from the camera to the lidar, and the lidar to GPS/IMU. For the intrinsic matrix of any frame, the matrix $\mathbf{P}_{rect(i)}^{j}$ mentioned in Eq. (2.11) [31] can be converted into the intrinsic matrix $\mathbf{K}$ according to Eq. (2.15).

$$\begin{aligned}
\mathbf{R}_{ex} &= \mathbf{R}_{rect(i)}^{j}\ \mathbf{R}_{velo(i)}^{cam(i)}\ \mathbf{R}_{imu(i)}^{velo(i)}\ \mathbf{R}_{imu(i)}^{world\ \top} \\
\mathbf{t}_{ex} &= \mathbf{R}_{rect(i)}^{j}\mathbf{R}_{velo(i)}^{cam(i)}\mathbf{R}_{imu(i)}^{world\ \top}\mathbf{t}_{imu(i)}^{world} \\
&\quad + \mathbf{R}_{rect(i)}^{j}\mathbf{R}_{velo(i)}^{cam(i)}\mathbf{t}_{imu(i)}^{velo(i)} \\
&\quad + \mathbf{R}_{rect(i)}^{j}\mathbf{t}_{velo(i)}^{cam(i)} + \mathbf{t}_{rect(i)}^{j}
\end{aligned} \tag{2.14}$$

$$\mathbf{K} = \mathbf{P}_{rect(i)}^{j}(1:3, 1:3) \tag{2.15}$$

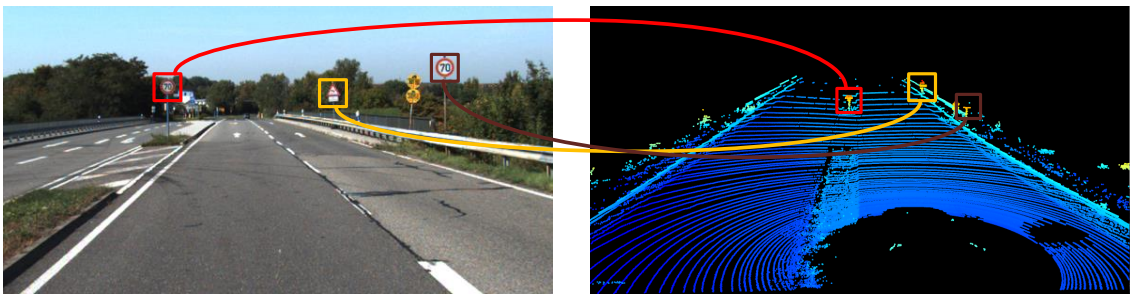


Figure 2.4: An instance for the image and lidar correspondence

In terms of ground truth processing, lidar frames are employed to extract dense points of a traffic sign. After the manipulation of rotation, translation, zoom and removing some unrelated points (such as points in the back of vehicle) in the MeshLab software [17], a lidar frame can be shown in the view of an image (like Fig. 2.4). Thus, according to the relative image frame, the lidar points of the traffic sign can be extracted by the MeshLab

software [17] (illustrated in Fig. 2.4). However, as is illustrated in "Lidar Data of signs" of Fig. 2.2, the lidar points of the traffic sign from each frame are so sparse. To overcome the sparsity problem, the lidar points of the traffic sign from each frame are transferred into the world coordinate to form the dense points. Eq. (2.16) indicates that the points $\mathbf{P}_{velo(i)}$ of the ith frame lidar are converted into the world coordinate $\mathbf{P}_{world}$.

$$\begin{aligned} &\mathbf{P}_{world} = \mathbf{R}_{imu(i)}^{world}\ {\mathbf{R}_{imu(i)}^{velo(i)}}^{\top}\ \mathbf{P}_{velo(i)} + \mathbf{T} \\ &where\ \mathbf{T} = \mathbf{t}_{imu(i)}^{world} - \mathbf{R}_{imu(i)}^{world} {\mathbf{R}_{imu(i)}^{velo(i)}}^{\top} \mathbf{t}_{imu(i)}^{velo(i)} \end{aligned} \tag{2.16}$$

An instance of visualization of dense points is shown in "Fused Radar Data of Sign" of Fig. 2.2. Then, the dense points are utilized to estimate the sign plane by [81]. As the red ones in Fig. 2.5, some points, whose distances to the plane are larger than 0.12 meter, are regarded as outliers and removed. After projecting dense points into the sign plane, we utilize the projected points to acquire the ground truth of position and pose. To begin with, any three projected non-collinear points are utilized to construct the x and y axises of the traffic sign coordinate, which has been defined in the beginning of Sec. 2.4.2. The z axis is computed based on x and y axises by right hand rule. Besides, the rotation matrix $\mathbf{R}_{world}^{sign}$ and translation vector $\mathbf{t}_{world}^{sign}$ can be acquired. Similar to the pose estimation part in Sec. 3.4, if the angle between z axis of the sign coordinate and z axis of the camera coordinate is bigger than $\pi/2$, the ground truth of the pose is the direction of z axis of the sign coordinate. Otherwise the ground truth of the pose is the opposite of z axis of the sign coordinate. Next, we estimate the contour of the sign in the sign coordinate. However, in general, the projected dense points often distribute the bottom of signs and hardly overlap the top of the sign (Fig. 2.5). For the incomplete overlap problem, we add the geometry constraint to the projected dense points to estimate the contours. Generally, the traffic signs has three kinds of shapes: triangle, square, and circle.

(i) Signs of the triangle shape: based on the point cloud of traffic sign, the method [28] is employed to obtain the lines candidates. Any three lines and their intersection points constitute the set $L_t = \{(\mathbf{l}_i^1, \mathbf{l}_i^2, \mathbf{l}_i^3, \mathbf{A}_i, \mathbf{B}_i, \mathbf{C}_i)\}$, where $\mathbf{l}_i^1, \mathbf{l}_i^2, \mathbf{l}_i^3$ are the ith group of three lines; $\mathbf{A}_i, \mathbf{B}_i, \mathbf{C}_i$ are the intersection points of the three lines. We try to find

one of three lines to satisfy the following equation.

$$\begin{gathered}\angle \mathbf{A}_i\mathbf{B}_i\mathbf{C}_i \approx \angle \mathbf{B}_i\mathbf{A}_i\mathbf{C}_i \approx \angle \mathbf{B}_i\mathbf{C}_i\mathbf{A}_i \approx \frac{\pi}{3} \\ S_{\triangle \mathbf{A}_i\mathbf{P}\mathbf{B}_i} + S_{\triangle \mathbf{A}_i\mathbf{P}\mathbf{C}_i} + S_{\triangle \mathbf{B}_i\mathbf{P}\mathbf{C}_i} \approx S_{\triangle \mathbf{A}_i\mathbf{B}_i\mathbf{C}_i}\end{gathered} \tag{2.17}$$

In Eq. (2.17), the first sub equation aims to guarantee that the contour of the traffic sign is of equilateral triangle shape. In the second sub equation, the $\mathbf{P}$ is any projected point of signs and this equation aims to guarantee that the contour of the traffic sign contains all projected lidar points.

(ii) Signs of the square shape: the method for signs of the square shape is almost the same with that of the triangle, except for the set $L_s = \{(\mathbf{l}_i^1, \mathbf{l}_i^2, \mathbf{l}_i^3, \mathbf{l}_i^4, \mathbf{A}_i, \mathbf{B}_i, \mathbf{C}_i, \mathbf{D}_i)\}$ and the shape constraints in Eq. (2.18), where the third sub equation make sure that the contour of the sign is of the square shape.

$$\begin{gathered}\angle \mathbf{A}_i\mathbf{B}_i\mathbf{C}_i \approx \angle \mathbf{B}_i\mathbf{C}_i\mathbf{D}_i \approx \angle \mathbf{C}_i\mathbf{D}_i\mathbf{A}_i \approx \angle \mathbf{D}_i\mathbf{A}_i\mathbf{B}_i \approx \frac{\pi}{2} \\ S_{\triangle \mathbf{A}_i\mathbf{P}\mathbf{B}_i} + S_{\triangle \mathbf{A}_i\mathbf{P}\mathbf{C}_i} + S_{\triangle \mathbf{B}_i\mathbf{P}\mathbf{C}_i} \approx S_{\mathbf{A}_i\mathbf{B}_i\mathbf{C}_i\mathbf{D}_i} \\ l_i^1 \approx l_i^2 \approx l_i^3 \approx l_i^4\end{gathered} \tag{2.18}$$

(iii) Signs of the circle shape: any three points are utilized to determine a circle, which is represented as the center of the circle (a_i, b_i) and radius R_i. The element of set $L_c = \{(a_i, b_i, R_i)\}$, which satisfy Eq. (2.19) for all points (x, y), to determine the contour.

$$\sqrt{(x-a)^2 + (y-b)^2} \leq R \tag{2.19}$$

Finally, based on the contours of the signs, the centroid of the traffic sign can be estimated in the sign coordinate. The ground truth position and pose in the sign coordinate are converted into the world coordinate by $\mathbf{R}_{world}^{sign}$ and $\mathbf{t}_{world}^{sign}$ matrices

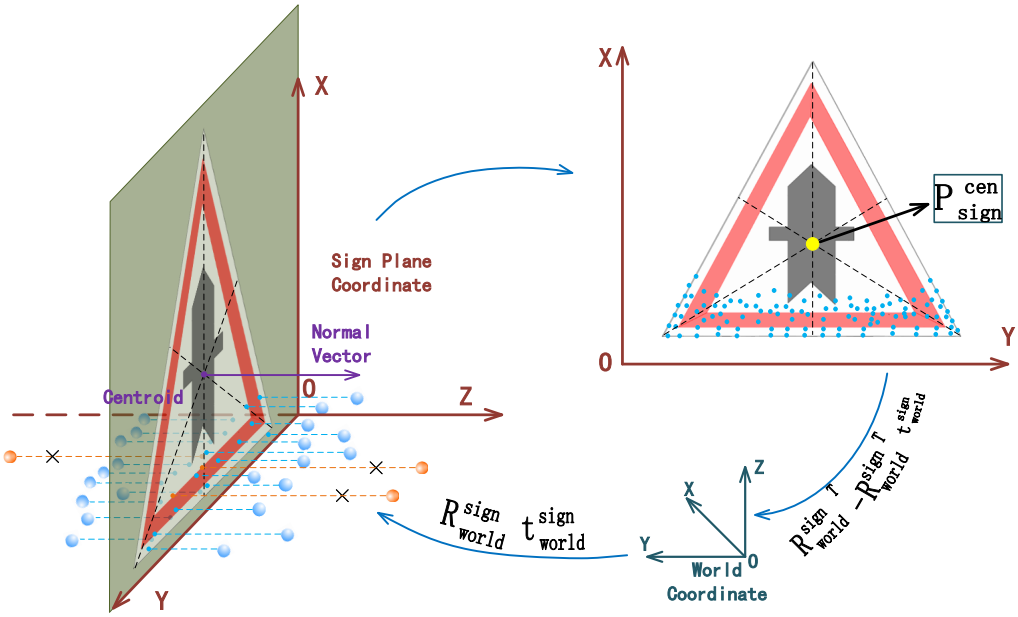


Figure 2.5: The figure of projecting points to the sign plane

2.5 Experiments

To evaluate the performance of the proposed method, both the localization, pose estimation experiments have been conducted in this section. For the localization experiments, we select both BelgiumTS dataset [79] and KITTI$^+$ dataset to comprehensively evaluate the proposed method. In addition, the methods in [76, 91, 36] are applied in two datasets for the comparison. In terms of pose experiments, since the BelgiumTS [79] dataset lacks the ground truth of the sign pose, we only utilize the KITTI$^+$ dataset to test the proposed method and the method in [76].

2.5.1 Experimental Preparation

Dataset

Both the BelgiumTS dataset [79] and KITTI$^+$ dataset should be processed for experiments. Specifically, BelgiumTS [79] dataset aims at the tasks of traffic sign detection, recognition and localization. We extract the consecutive image frames, the intrinsic matrix, extrinsic matrices and the ground truth of 3D positions from the BelgiumTS [79] dataset. The regions of traffic signs in image frames are extracted manually. For KITTI$^+$ dataset, it has been processed according to Sect. 2.4. Both the BelgiumTS [79] and KITTI$^+$ dataset

are also divided according to the shape and sequence. Concretely, BelgiumTS [79]: shape of the circle, triangle, rectangle signs and other shape signs; the 01 to 04 sequences. KITTI: shape of the circle, triangle, rectangle signs; the city, residential and road sequences .

Methods

We employ three different methods in [36, 76, 91] for comparisons. These three methods are denoted as Hazelhoff [36], Soheilian [76], Welzel [91] respectively for simplification.

For Hazelhoff [36] method, we implement the localization method and adjust the parameters to adapt to the BelgiumTS [79] and KITTI$^+$ dataset. Specifically, the inputs in [36] are panoramic images, which demands that $\phi_{vertical}$ ranges from $-\pi/2$ to $\pi/2$, and $\phi_{horizontal}$ ranges from $-\pi$ to π. In experiments, $\phi_{vertical}$ ranges from $-\arctan\frac{v_0}{f_y}$ to $\arctan\frac{Height-v_0}{f_y}$ and $\phi_{horizontal}$ ranges from $-\arctan\frac{u_0}{f_x}$ to $\arctan\frac{Width-u_0}{f_x}$, where u_0, v_0, f_x and f_y are principal point and focal length acquired from the camera intrinsic matrix; and $Height$, $Width$ are the width and height of images in pixel.

The Soheilian [76] method utilizes the vertexes of traffic signs to achieve localization and pose estimation, which can not apply to the signs of the circle and irregular shapes. Thus we manually label the vertexes of signs in the two dataset except the signs of the circle and irregular shapes. The Soheilian [76] method acquires the vertex point of signs $\mathbf{p}_1$, $\mathbf{v_1}$ and $\mathbf{v_2}$ vectors for either triangle or rectangle signs after optimization. For the position estimation, the vertex $\mathbf{p}_1$, $\mathbf{v_1}$ and $\mathbf{v_2}$ can be utilized to estimate the 3D position based on the shape information. In terms of pose estimation, the vector $\mathbf{v_n} = \frac{\mathbf{v_1}\times\mathbf{v_2}}{\|\mathbf{v_1}\times\mathbf{v_2}\|}$ represents the normal vector of traffic signs.

The Welzel [91] method utilizes the real size of the traffic signs and the that in the image to estimate the relative distance. The positions of traffic signs are determined by relative distances and the bearing. According to the [91], the algorithm has been implemented by ourself for comparisons.

In order to conduct a fair comparison, we utilize two different feature points in our method. One is to utilize the feature points labeled manually for localization and pose estimation, which is compared to the Soheilian method [76]. The other is to utilize the SIFT [54] and SURF [7] feature points for localization, which is compared to the Hazelhoff [36] and Welzel [91] method. We call these twos as "**Given Points**" and "**SIFT+SURF**"

in the following article for convenience.

Metric

We utilize the Euclidean distance of the estimated the 3D position and ground truth, the degree of the estimated pose and ground truth to evaluate the localization and pose estimation error of each sign. For overall performance evaluation, two different metrics are employed. The first one is to measure the average error of each item. The second one is to measure the number of instances, whose error is less than a threshold.

2.5.2 Localization Results

Experimental results w.r.t. sequence in BelgiumTS

In this experiment, image sequences from 01 to 04 are utilized for comparison experiments of ours, Hazelhoff [36] and Welzel [91]. Meanwhile, the image sequences form 01 to 04 without the signs of ellipse and irregular shape are employed for comparison of ours and Soheilian [76]. The statistical localization results are depicted in Table 2.5.2. Our approaches (Given Points and SIFT+SURF) outperform Sohelian [76], Hazelhoff [36] and Welzel [91] approaches in sequence 01 to 04. Fig. 2.6 (a) (b) present the average accuracy of ours (Given Points, SIFT+SURF), Hazelhoff [36], Soheilian [76] and André [91] methods from sequence 01 to 04. It is obvious that ours significantly outperform other twos except that Soheilian [76] approach localizes more accurately than SIFT+SURF does in the sequence 04. Comparing Soheilian method [76] and "Given Point", "Given Points" outperforms Soheilian [76] significantly. In addition, it reflects that "Given point" are more accurate than "SIFT+SURF". The average errors of ours ("Given Points", "SIFT+SURF") are 0.41 and 0.44 meters respectively.

Table 2.2: The localization results of BelgiumTS

Sequence	Method	Accuracy (m)			
		< 0.3	< 0.6	< 0.9	< 1.2
	Hazelhoff	36.76%	75.00%	85.29%	94.12%
Continued on next page					
Seq01					

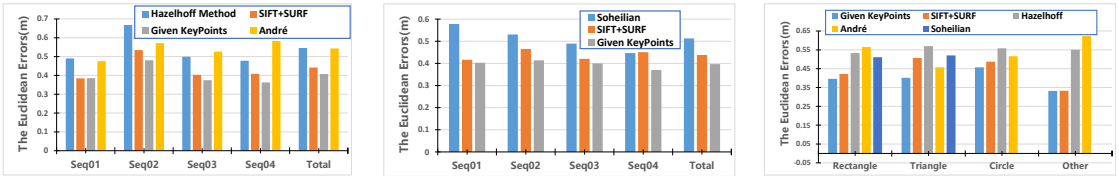
Sequence	Method	Accuracy (m)			
		< 0.3	< 0.6	< 0.9	< 1.2
Seq02	Given Points	**51.47%**	89.71%	**94.12%**	**95.59%**
	SIFT+SURF	47.76%	**92.54%**	92.54%	95.52%
	Welzel	30.03%	72.73%	92.42%	**96.97%**
	Soheilian	39.22%	76.47%	82.35%	88.24%
	Given Points	**50.98%**	90.20%	**92.16%**	**94.12%**
	SIFT+SURF	44.00%	**92.00%**	92.00%	94.00%
Seq03	Hazelhoff	28.57%	58.16%	75.51%	83.67%
	Given Points	**45.92%**	**79.59%**	**93.88%**	**97.96%**
	SIFT+SURF	40.82%	75.51%	88.78%	96.94%
	Welzel	20.62%	59.79%	85.57%	92.78%
	Soheilian	16.00%	64.00%	**90.00%**	**98.00%**
	Given Points	**44.00%**	**82.00%**	**90.00%**	**98.00%**
	SIFT+SURF	36.00%	74.00%	**90.00%**	96.00%
Seq04	Hazelhoff	40.58%	71.01%	84.06%	95.65%
	Given Points	**49.28%**	**84.06%**	**97.10%**	**100.00%**
	SIFT+SURF	46.38%	76.81%	**97.10%**	**100.00%**
	Welzel	24.64%	69.57%	88.41%	97.10%
	Soheilian	41.18%	74.51%	80.39%	96.08%
	Given Points	**43.14%**	**82.35%**	**98.04%**	**100.00%**
	SIFT+SURF	37.25%	76.47%	**98.04%**	**100.00%**
	Hazelhoff	32.43%	71.62%	93.24%	97.30%

Continued on next page

Sequence	Method	Accuracy (m)			
		< 0.3	< 0.6	< 0.9	< 1.2
	Given Points	**48.65%**	**90.54%**	**95.95%**	**98.64%**
	SIFT+SURF	44.59%	85.14%	94.59%	95.95%
	Welzel	32.86%	62.86%	81.43%	91.43%
	Soheilian	28.26%	82.61%	89.13%	97.83%
	Given Points	**45.65%**	**89.13%**	**95.65%**	**100.00%**
	SIFT+SURF	36.96%	78.26%	93.48%	95.65%

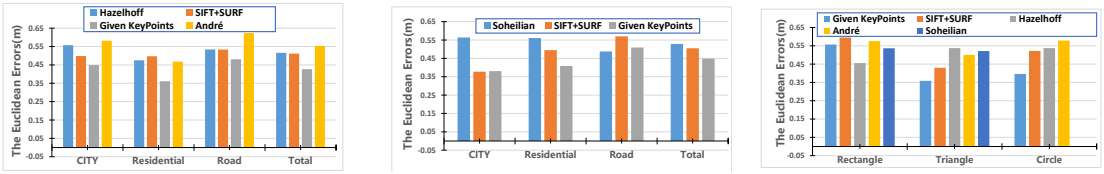
Localization results w.r.t. sign shapes in BelgiumTS

In this experiment, we aim to evaluate the effectiveness of our approach to diverse shapes of the traffic signs. Thus the rectangle and triangle signs are employed for comparison of ours ("Given Points") and Soheilian [76] method based on the manually labeled feature points. Besides, the signs of the rectangle, triangle, ellipse and other irregular shapes are employed for comparing ours ("SIFT+SURF") with Hazelhoff [36] and Welzel [91] without the pre-labeled points. In Table 2.5.2, it is illustrated that our approach ("Given Points") outperforms the Soheilian [76] totally. Compared with Hazelhoff [36] and André [91], ours ("SIFT+SURF") has better performance in the mosts items except "< 0.9", "< 1.2" in the triangle and "< 0.9" in the circle. As the average errors shown in Fig. 2.6 (c), ours including "Given Points" and "SIFT+SURF" outperforms the other methods except the item of the triangle signs.



(a) (b) (c)

Figure 2.6: The localization results by ours, Hazelhoff [36], Soheilian [76] and Welzel [91] in BelgiumTS Dataset



(a) (b) (c)

Figure 2.7: The localization results by ours, Hazelhoff [36], Soheilian [76] and Welzel [91] in KITTI$^+$ Dataset

Table 2.3: The results w.r.t. diverse shape in BelgiumTS

Shape	Method	Accuracy (m)			
		< 0.3	< 0.6	< 0.9	< 1.2
	Hazelhoff	39.29%	64.29%	80.95%	89.29%
Circle	Hazelhoff	34.16%	71.43%	84.47%	93.17%
	Welzel	27.39%	61.78%	83.44%	92.36%
Rectangle	SIFT+SURF	**37.89%**	**80.12%**	**93.79%**	**96.89%**
	Given Points	**45.96%**	**85.71%**	**93.79%**	**98.14%**
	Soheilian	31.06%	74.53%	85.71%	96.27%
Triangle	Hazelhoff	32.43%	64.86%	83.78%	91.89%
	Welzel	30.56%	75.11%	**97.22%**	**100.00%**
	SIFT+SURF	**40.54%**	**78.38%**	89.19%	91.89%
	Given Points	**45.95%**	**86.49%**	**94.59%**	**97.30%**
	Soheilian	32.43%	72.97%	83.78%	89.19%

Shape	Method	Accuracy (m)			
		< 0.3	< 0.6	< 0.9	< 1.2
	Given Points	**53.57%**	**83.33%**	**96.43%**	**97.62%**
	SIFT+SURF	**53.57%**	82.14%	83.33%	**97.62%**
	Welzel	24.10%	69.88%	91.57%	97.59%
Others	Hazelhoff	18.52%	62.96%	88.89%	92.59%
	Given Points	51.85%	88.89%	**100.00%**	**100.00%**
	SIFT+SURF	**55.56%**	**92.59%**	96.30%	**100.00%**
	Welzel	22.22%	59.62%	77.78%	88.89%

Localization results w.r.t. traffic environments in KITTI$^+$

We compare our approach, Hazelhoff [36] and Welzel [91] approaches in diverse traffic environment, i.e. city, residential and road, of KITTI dataset. Like Sec. 5.2.1, the circle signs are removed from KITTI to test ours and Soheilian [76] approach based on the manually labeled points. The localization results are shown in Table 2.4. Compared to the Soheilian [76], ours ("Given Points") basically outperforms the Soheilian [76] in most items except "< 0.6" in the city and "< 0.9" in the road. Besides ours based on SIFT and SURF features has better performance than other twos in the most items of "CITY". The average error results of three methods in "City", "Residential", "Road" sequences are shown in Fig. 2.7 (a), (b). Results in Fig. 2.7 (a) verify that ours outperforms Hazelhoff [36] and Welzel [91] in "City", "Road" and "Total" items. Fig. 2.7 (b) shows that ours outperforms Soheilian [76] in the KITTI$^+$ dataset except for the "Road" sequence. The average errors of our methods ("Given Points", "SIFT+SURF") in KITTI dataset are 0.42 and 0.51 meter.

Localization results w.r.t. shape in KITTI$^+$

In order to verify the availability of our method in the diverse traffic sign shapes, we compare ours ("Given Points") with Soheilian [76] method in the shapes of rectangle, triangle signs of KITTI$^+$ dataset based on the manually labeled points. In addition, the Hazelhoff [36] and Welzel [91] approaches are exploited to compare with ours ("SIFT+SURF") in the all shapes of the signs in the KITTI dataset without labeled points. As the localization results shown in Table 2.5.2, the "Given Points" almost outperforms the Soheilian [76] method. "SIFT+SURF" has better performances than other twos in the most items of triangle and circle traffic signs. The average error results of the three methods in diverse shapes are shown in Fig. 2.7 (c). Obviously, ours outperforms the other threes in all shapes except for rectangle traffic signs.

Table 2.5: The results w.r.t. diverse shapes in KITTI$^+$

Shape	Method	Accuracy (m)			
		< 0.3	< 0.6	< 0.9	< 1.2
Rectangle	Hazelhoff	**30.00%**	**85.00%**	**95%**	**100%**
	Welzel	5.00%	60.00%	90.00%	100%
	SIFT+SURF	10.00%	65.00%	90.00%	95.0%
	Given Points	15.00%	**80.00%**	**85.00%**	**95.0%**
	Soheilian	**20.00%**	65.00%	85.00%	95.0%
Triangle	Hazelhoff	20.83%	66.67%	87.5%	87.50%
	Welzel	16.67%	70.83%	**100%**	100%
	SIFT+SURF	**33.33%**	**79.17%**	95.83%	**100%**
	Given Points	**45.83%**	**83.33%**	**100%**	**100%**
	Soheilian	12.50%	70.83%	95.83%	**100%**
Circle	Given Points	34.38%	**84.38%**	**100%**	**100%**

Continued on next page

Shape	Method	Accuracy (m)			
		< 0.3	< 0.6	< 0.9	< 1.2
	SIFT+SURF	**35.48%**	70.97%	87.10%	90.32%
	Hazelhoff	12.50%	65.63%	93.75%	**100%**
	Welzel	9.09%	63.64%	87.88%	96.97%

2.5.3 Pose Results

In this section, the experiments are conducted for testing our method in the aspect of pose estimation. Meanwhile, the Soheilian [76] method is employed for comparison. Table 2.5.3 shows the pose estimation results in KITTI$^+$ dataset. "Given Points" totally outperforms Soheilian [76] within errors of 30 degree in "City" and 20 degree in "Road". "Given Points" has the same performance with Soheilian [76] in the signs of the rectangle shape. Fig. 2.8 shows that the average degree errors of the two methods. "Given Points" outperforms Soheilian [76] in the items of "Road" and "Rectangle".

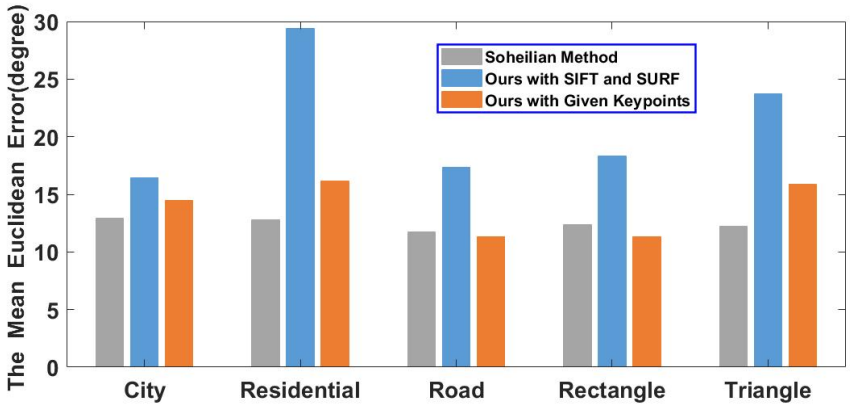


Figure 2.8: The pose estimation results of traffic signs in KITTI$^+$

Table 2.6: The statical pose results in KITTI^{+}

Item	Method	Accuracy (Deg)		
		< 10	< 20	< 30
City	Given Points	**33.33%**	66.67%	**100%**
	SIFT+SURF	**33.33%**	55.56%	89%
	Soheilian	**33.33%**	**88.89%**	89%
Residential	Given Points	26.67%	66.67%	93%
	SIFT+SURF	14.29%	35.71%	64%
	Soheilian	**40.00%**	**86.67%**	**100%**
Road	Given Points	55.00%	**90.00%**	**95%**
	SIFT+SURF	35.00%	60.00%	85%
	Soheilian	**57.14%**	85.71%	**95%**
Rectangle	Given Points	**57.14%**	**85.71%**	**95.24%**
	SIFT+SURF	38.10%	66.67%	86%
	Soheilian	**57.14%**	**85.71%**	**95.24%**
Triangle	Given Points	25.00%	70.83%	**95.83%**
	SIFT+SURF	17.39%	34.78%	74%
	Soheilian	**37.50%**	**87.50%**	**95.83%**

Table 2.4: The results w.r.t. traffic environment in KITTI^{+}

Item	Method	Accuracy (m)			
		< 0.3	< 0.6	< 0.9	< 1.2
CITY	Given Points	**31.25%**	**81.25%**	**100%**	**100%**
	SIFT+SURF	**31.25%**	81.25%	87.50%	93.75%
	Hazelhoff	18.75%	62.50%	87.50%	**100%**
	Welzel	0%	62.50%	87.50%	**100%**
	Given Points	**44.44%**	88.89%	**100%**	**100%**
	SIFT+SURF	33.33%	**100%**	**100%**	**100%**
	Soheilian	11.11%	66.67%	**100%**	**100%**
RESID-ENTIAL	Given Points	**46.67%**	**90.00%**	**96.67%**	96.67%
	SIFT+SURF	37.93%	79.31%	89.66%	89.66%
	Hazelhoff	23.33%	76.67%	93.33%	**100%**
	Welzel	23.33%	80.00%	93.33%	**100%**
	Given Points	**46.67%**	**86.67%**	**93.33%**	**93.33%**
	SIFT+SURF	28.57%	78.57%	92.86%	92.86%
	Soheilian	20.00%	66.67%	80.00%	**93.33%**
ROAD	Given Points	**20.00%**	**76.67%**	**93.33%**	**100%**
	SIFT+SURF	10.00%	60.00%	**93.33%**	**100%**
	Hazelhoff	16.67%	70.00%	**93.33%**	**100%**
	Welzel	3.33%	53.33%	**93.33%**	96.67%
	Given Points	**15.00%**	**75.00%**	90.00%	**100%**
	SIFT+SURF	5.00%	55.00%	90.00%	**100%**
	Soheilian	**15.00%**	70.00%	**95.00%**	**100%**

2.6 Summary

This chapter proposes a novel vision-based method for recovering the attributes (i.e. 3D position and pose) of traffic signs from image sequences. The dot features and their descriptors (SIFT, SURF) are extracted from all images. The process of feature matching and shifting is interactive to establish robust feature correspondences. Then the designed objective function can build the homography relationship between feature correspondences and the parameters of the sign plane. The 3D plane of the traffic sign is determined by optimizing the objective function. Finally the attributes of the sign are estimated based on the sign plane. In addition, this chapter proposes a series of processing methods to make KITTI$^+$ dataset for the tasks of 3D localization and pose estimation of traffic signs. Finally, based on the BelgiumTS and KITTI$^+$ datasets, the compared experiments with methods in [91], [36] and [76] are conducted respectively to validate the effectiveness of our method in the aspects of localization and pose estimation. This research makes the following contributions:

1) A novel vision-based method has been proposed to recover the attributes of traffic signs in three steps. The method first extracts effective features and establishes feature correspondences from images. Secondly, the method combines the feature correspondences and the homography constraints to form objective function for the traffic sign plane estimation, and finally estimate 3D position, pose at the same time.

2) In order to make an extension of KITTI dataset [30] for 3D tasks, a series of processing methods are put forward. Different from the BelgiumTS dataset [79], the ground truth of 3D position and pose are extracted from lidar frames.

3) In experiments, we utilize both KITTI [30] and BelgiumTS [79] datasets to conduct the comparison experiments with state-of-the-art methods. This off-line experiments have the advantages of easily comparing with other methods and consuming less resources.

Chapter 3

3D-2D Registration for Road Pose Estimation

Pose estimation of traffic roads is still challenging due to the big noises for multi-modal data. This chapter explores to estimate the pose of the traffic road through 3D-2D registration method. Before this, we propose a convenient fashion to obtain the 3D wire-frame road model. Then, the objective function is deduced based on the feature correspondences for coarse-to-fine pose estimation. Subsequently, comparative experimental results are displayed and discussed to prove effectiveness of ours. Finally, we review and summarize this chapter.

3.1 Introduction

The problem of 3D registration aims to estimate the absolute pose (i.e. orientation and position) of cameras with respect to the 3D model, given a set of features from 3D model and the corresponding 2D projection on the image. Solutions to the pose estimation problem has been widely applied in the fields of visual odometry [10, 63], image-based localization and navigation [45], and augmented reality [56] etc. The pose estimation from a single image has been formulated two main types according to the feature correspondences. Ones solve the pose estimation by providing the n 2D-3D point correspondences, which is also called *Perspective-n-Point* (PnP) problem. Many efficient solutions including EPnP [61] were put forward and succeeded in the PnP problem to some degree. With

huge emergency of line feature extraction methods [53], the line correspondences are taken into account for this task. Many researchers [67, 95] refer to the pose estimation by line correspondences as *Perspective-n-Line* (PnL) problem. No matter what kinds of correspondences, the transformation estimation of a single camera is totally called perspective pose estimation.

In recent year, with increasing utilization of multi-camera systems in the intelligent robots, autonomous vehicles etc, the solutions for single camera pose estimation can not be applied into the multi-camera systems. Thus, the perspective pose estimation has been extended and investigated for multi-camera system, which is called non-perspective pose estimation. The essential difference between the perspective and non-perspective pose estimation is that for the latter, there exists more than one camera centers [47]. Besides, the latter problem is converted to the former one, when multi-camera system degenerate to a single camera. Thus, the solutions to non-perspective pose estimation are generally efficient for the perspective pose estimation [47]. According to correspondences, these problems are still classified into point-based and line-based non-perspective pose estimation. The point-based non-perspective pose problem, which is also known as *non-perspective n point* (NPnP), was relatively fully studied in the recent decade. On the contrary, line-based non-perspective pose estimation, which is called *non-perspective n line* (NPnL), was first put forward by Lee [47] in 2016. Thus, the NPnL problem remains relatively un-investigated.

In this chapter, we aim to estimate the relative pose (i.e. position and orientation) between the cameras and the model of the traffic road, which also belongs to the problem of non-perceptive pose estimation. The main reason for the traffic scenario is that one of important application is to utilize the augmentation reality to synthesize and render new traffic images according to individual demands. The newly synthesized traffic images can be supplemented to the open-sourced image data (such as KITTI [29], RobotCar [55], ApolloScape [41]) for testing the visual algorithms (such as vehicle detection [24, 90, 97]) of self-driving car. This task is still full of challenges in the following reasons. 1) The 3D road models of traffic scenes are hard to acquire in actual. The prevalent fashions utilize the lidar sensors to capture the data of traffic scene. The model is generated by amount of manual processing, which is time-consuming, expensive. We proposes 3D wire-frame

road model from the GIS information (Google Earth) instead. The processing method for such model is relatively time-saving and low-cost. 2) Based on the aforementioned wire-frame road model, the existing non-perspective pose estimation solutions from line correspondences are not robust and accurate enough to achieve the augmentation reality. Thus, a two-stage coarse-to-fine registration method is proposed to estimate the poses of non-perspective cameras from point-to-line correspondences. Firstly, the proposed method adopts the iterative closest point (ICP) algorithm to match the road model and the viewpoints of multi cameras to estimate the coarse pose. Secondly, the objective function combines the point-to-line feature correspondences to refine the coarse pose.

3.2 Related Works

As stated in the previous section, the solutions to pose estimation from line correspondences can be classified into two categories (i.e. PnL and NPnL) according to the number of cameras. The literatures will be given to discuss solutions to both PnL and NPnL problem.

3.2.1 PnL Solution

Generally, the PnL problem aims to estimate the pose of a camera with respective to a model from line correspondences. This problem has been well-investigated in the last two decades. The prevalent PnL solutions are generally achieved by iterative and algebraic fashions. Iterative approaches regard the PnL as the nonlinear least squares problem. They often utilize the geometrical and algebraic errors to construct the cost function, which estimates the pose of the camera through minimization. Zhang et al. [101] *et al.* utilized an orthogonal constraint to filter the line correspondences and subsequently achieve pose estimation. Although iterative approaches often achieve accurate pose estimation, they suffer from two shortcomings. Firstly, the iterative approaches have a high requirement of the initial pose value for refinement. Slow convergences or even failures will emerge when initialization pose is inaccurate. Secondly, the iterative schemes often take much time to for pose refinement, which is hardly applied in the real-time tasks.

On the contrary, the algebraic approaches estimate the camera pose by solving a de-

duced polynomial equations. Thus, these solutions have no demand of both initialization and iterative optimization. The earliest approaches including [23, 14] proposed a closed-form solutions for the minimal configuration of three line correspondences. The approaches totally obtain eight solution candidates from three line correspondences (P3L), which is not shifted in [23, 14]. Recently, a global method [59] was proposed to handle the P3L problem as a polynomial system with 27 solution candidates. This method [59] has huge improvements in the computational cost O(n) and robustness in the image noises. More recently, the Robust PnL (RPnL) [100] and accurate subset based PnL (ARPnL) [96] was put forward to study and analyze the PnL problem further. Besides the solutions of polynomial system, another series of solutions for PnL problem are known as DLT-based methods. The famous DLT-based solution was called DLT-Lines, which utilizing the endpoint of 3D lines and 2D line to establish the correspondences. Subsequently, [66] employed the Plücker coordinates to parameterize a 3D line. The proposed DLT-Plücker-Lines solution [66] has more accurate pose estimation than DLT-Lines does. Recently, a DLT-Combined-Lines solution [67] was proposed to combine line and point-to-line correspondences in the DLT framework to acquire significant improvements in accuracy. However, this method [67] need to adjust a hyper parameter to combine the influence of line and point-to-line correspondence.

3.2.2 NPnL Solution

NPnL problem aims to estimate the poses of multi cameras with respective to a model fixed in the world coordinate by providing the line correspondences. Since the earliest definition of the NPnL problem was derived from 2016 by [47], NPnL problem is relatively un-investigated. Thus, some recent works are introduced with the time order. The minimal configuration including three cameras and line correspondences was first proposed by Lee [47]. Besides, the Plücker coordinate are utilized to parameterize a 3D line for estimating the rotation and translation separately [47]. This solution [47] has an advantage of being compatible with both PnL and NPnL problems. Subsequently, [58] combines both point and line correspondences for pose estimation from minimal configuration. Besides, the close-form solution for the configuration of two points and one line was first proposed by [58]. Recently, some researchers aim to apply the NPnL solutions to systems (such as

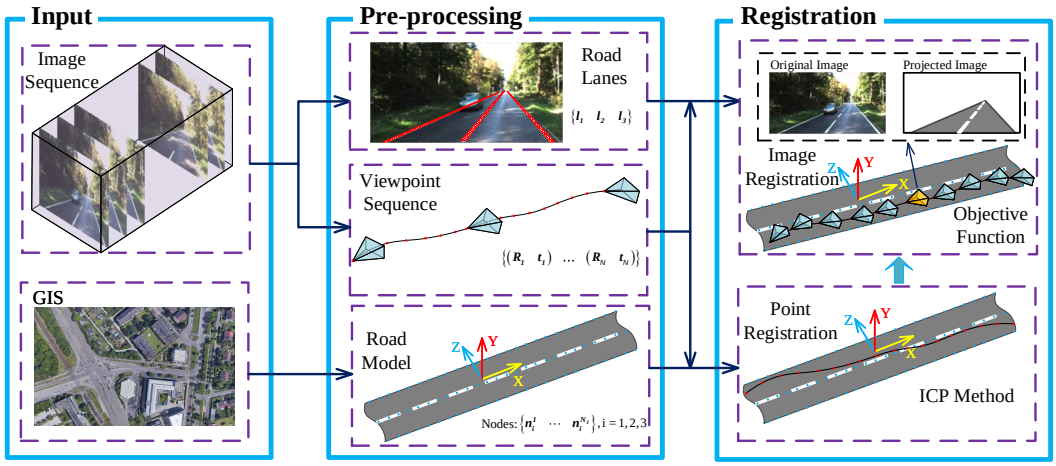


Figure 3.1: The pipeline of the proposed registration and augmented framework. **Input** includes two videos from binocular cameras, road information from GIS and camera parameters. **Preprocessing** obtains the road model and correspondence establishment, where blue points in the model are labeled from GIS; the labeled red lines in the image and the blue points from the model construct correspondences. In **Registration**, the poses of viewpoints are estimated first in viewpoint registration; the pose of the road is estimated in road registration.

self-driving car) equipped with IMU sensors. Hence, some solutions combining the vertical direction information from IMU were proposed to convert the pose to four freedom of degree (one for rotation, three for translation) estimation. The famous solution [38] represents a 3D line as combination of the point and direction, and deduce the problem to cubic polynomial solution. Although the successful solutions have been proposed, the estimated pose is not accurate enough to meet the task of augmented reality in traffic scenarios. Thus, we propose a two-stage coarse-to-fine registration method from line correspondences. We represent a 3D line as a set of 3D points and estimate the pose through minimizing the distance between image line and projected point. Experiments prove our accuracy exceeds others in the real traffic scenario.

3.3 Overviews

3.3.1 Problem Formulation

As mentioned in the introduction, finding the rigid transformation between multi cameras and road model from line correspondences belongs to non-perspective pose estimation and non-perspective n lines (NPnL) problem. The solutions to these problems are also the

'Road Registration' module of the proposed framework in Fig.3.1. As defined in [47], we are given the K 3D line features from the road model in the world coordinate $\{W\}$, and its K corresponding 2D projected lines by N cameras. The N cameras are fixed in the reference coordinate $\{G\}$, and the conversion from reference to ith camera coordinate is known and denoted as $\mathbf{T}_G^{C_i}$. The intrinsic parameters $\mathbf{K}_i, (i = 1, ...N)$ of ith cameras $(i = 1, ..., N)$ has been calibrated. Thus, the rigid transformation between cameras and road model can be determined by combination of the transformation $\mathbf{T}_W^G$ and $\mathbf{T}_G^{C_i}$. In the Eq. 3.1, the point $\mathbf{P}_G$ in the world coordinate is converted to $\mathbf{P}_{C_i}$ in the ith camera coordinate through the transformation $\mathbf{T}_w^G$, $\mathbf{T}_G^{C_i}$. Figure 3.2 gives an example to explain the problem, where the 3D red line in the model and 2D red line on the image establish a correspondence and can be converted to each other through the $\mathbf{T}_W^G$ and $\mathbf{T}_G^{C_i}$. Utilization of the relationship between line correspondences can estimate the transformation $\mathbf{T}_W^G$. As the black points located on the red line in Fig. 3.2, the 3D line is expressed as a set of 3D points in our method. Thus, It is equivalent to solve the NPnL problem utilizing the point-to-line correspondences.

$$\begin{aligned} \begin{bmatrix} \mathbf{P}_{C_i} \\ 1 \end{bmatrix} &= \mathbf{T}_G^{C_i} \mathbf{T}_W^G \begin{bmatrix} \mathbf{P}_W \\ 1 \end{bmatrix} \\ where \quad \mathbf{T}_W^G &= \begin{bmatrix} \mathbf{R}_W^G & \mathbf{t}_W^G \\ \mathbf{0} & 1 \end{bmatrix}; \mathbf{T}_G^{C_i} = \begin{bmatrix} \mathbf{R}_G^{C_i} & \mathbf{t}_G^{C_i} \\ \mathbf{0} & 1 \end{bmatrix} \end{aligned} \tag{3.1}$$

3.3.2 Framework

As shown in Fig. 3.1, the pipeline of the proposed augmented reality framework contains four stages: input, pre-processing, registration and video augmentation.

1) The **input** includes traffic videos, camera intrinsic matrix, road information from GIS and camera baseline B.

2) The **pre-processing** stage contains wire-frame road model generation (Sect. 3.4.1) and point-to-line feature correspondence establishment (Sect. 3.4.2).

3) The **registration** contains two parts: viewpoint registration, road registration. The viewpoint registration (Sect. 3.5.1) is to estimate the rigid transformation of all cameras

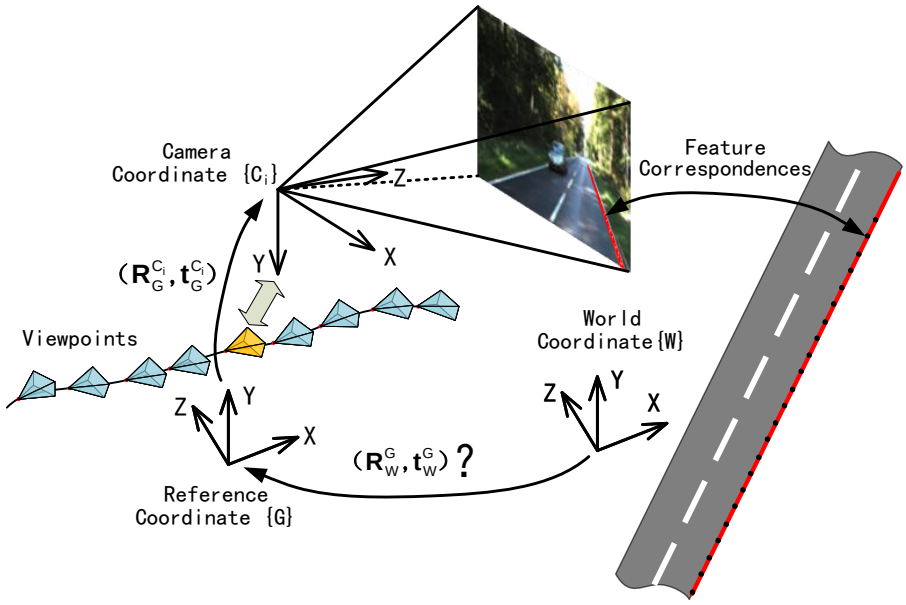


Figure 3.2: An instance of non-perspective pose estimation for traffic road.

with respective to aforementioned reference coordinate (i.e. $\mathbf{T}_G^{C_i}$), which is the input of NPnL problem. Road registration takes the feature correspondences, camera poses with respective to reference coordinate ($\mathbf{T}_G^{C_i}$), intrinsic matrix of camera as the input to solve the NPnL problem mentioned in problem formulation section (Sect. 3.3.1). As shown in Fig. 3.1, the two-stage registration method estimates the transformation $\mathbf{T}_W^G$ from initialization to refinement (Sect. 3.5.2).

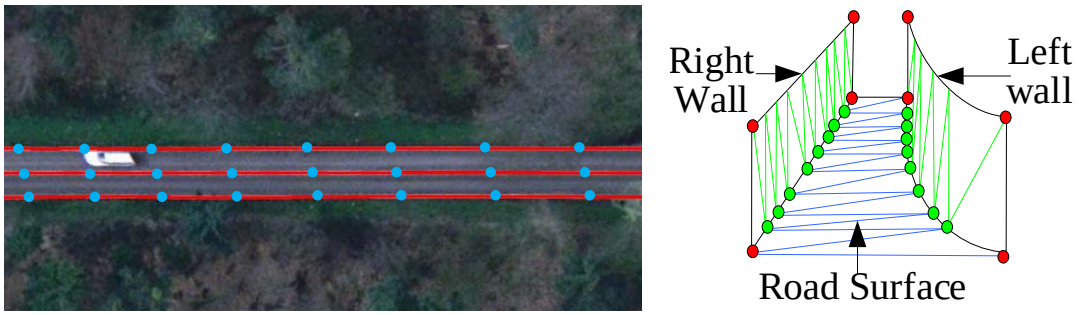
3.4 Multi-modal data preprocessing

The prerequisite of non-perspective pose estimation is the 3D-2D feature correspondences. Furthermore, the 3D feature is generally derived from 3D road model. Thus, this section mainly includes two aspects: 1) the wire-frame road model generation from road information. 2) Feature correspondence establishment between the traffic image and road model.

3.4.1 Road Model

Inspired by [51], the corridor road model is employed in this chapter. The road model consists of three parts: road surface, right and left walls. As shown in Fig. 4.2, the road surface is wire-frame model consisted of triangle meshes. The left and right walls are composed of the triangle mesh planes, which are perpendicular to the road surface and

pass the boundary lines. The reason why utilizing this model is that this structure can provide a movement space for traffic elements. Specifically, the traffic elements move on the road surface and within the boundaries of left and right walls.



(a) Road in Google Earth (b) Road model

Figure 3.3: Utilizing the Google Earth software to extract the road model. (a) Road in Google Earth. The red lines are the labeled lanes and boundaries. (b) The wire-frame road model composed of three parts: right wall, left wall and road surface, where the green points are labeled in Google Earth.

To make such a road model, the Google Earth Software containing abundant GIS information is employed to obtain the road information. The road information includes the boundary and lane lines of a road, which are manually labeled by points in the Google Earth software. As shown in Fig. 3.3(a), the red lines including lanes and boundaries on the road surface are labeled by blue points. After acquiring the road information, the points on the lines are converted from the WGS84 (World Geodetic System 1984) [25] to ENU (local East, North, Up) coordinate. In terms of the points on left and right walls, the normal direction of the road surface is estimated based on the labeled road points by [81]. The points on the walls can be calculated through boundary points and the normal direction. After obtaining the points of the road model relative to the ENU coordinate, the triangle meshes of the surface, left and right walls are generated by [46] respectively.

3.4.2 Feature Correspondence Establishment

Accurate feature correspondences are crucial for 3D-2D road registration. As mentioned in introduction of this chapter, the point-to-line feature correspondences are employed for the pose estimation. In terms of the 3D point feature, the 3D points of road boundaries and lane lines have been labeled and calculated relative to the ENU coordinate.

With respect to the 2D lines in the images, the straight road lines including lanes and boundaries are manually labeled by points. Subsequently, the parameters of each line can be calculated by linearly fitting the points. Without decreasing the accuracy of the registration algorithm, we only label lines from some images of each sequence for reducing workloads. The reasons are as follows: 1) The information of the road between consecutive frames is redundant. There is no need to label all frames of a sequence. 2) Since the road geometry is a strong constraint in the model, only labeling few frames can achieve good registration performances. These two reasons indicate the frames to be labeled have a long interval. Finally, the 3D points and the corresponding lines in the frames construct the feature correspondences.

3.5 Registration

As shown in Fig. 3.1, the registration module consists of two parts: viewpoint registration and road registration. Viewpoint registration aims to estimate the camera poses with respect to reference coordinate (i.e. $\mathbf{R}_G^{C_i}$, $\mathbf{t}_G^{C_i}$). Road registration is to find the transformation between the reference and world coordinate, (i.e. $\mathbf{R}_W^G$, $\mathbf{t}_W^G$).

3.5.1 Viewpoint Registration

Viewpoint registration is to recover the poses (i.e. $\mathbf{R}_G^{C_i}$, $\mathbf{t}_G^{C_i}$) of the viewpoints of all frames in the reference coordinate. There exist plenty of perfect algorithms, such as structure from motion (SfM), stereo vision and SLAM (Simultaneous localization and mapping), to solve this problem. This chapter utilizes the ORB feature [70] based SLAM method (i.e. ORB-SLAM2 [63]) to estimate the viewpoints from image sequences by giving the camera intrinsic matrix $\mathbf{K}_i$ and camera baseline B. To extract the ORB features, we set a six-level pyramid and extract one thousand amount of mutli-scale features [70] for an image. Through this method, the poses of viewpoints (i.e. $\mathbf{R}_G^{C_i}$, $\mathbf{t}_G^{C_i}$) are estimated relative to the reference coordinate.

3.5.2 Road Registration

Initialization

The aim of this paragraph is to estimate a rough pose $(\mathbf{R}_W^G, \mathbf{t}_W^G)$ of the road, which provides a robust and suitable initial value for optimization. First of all, the road is represented by points of a lane line in the model. The registered viewpoints of an image sequence can be also regarded as a 3D point cloud. Thus, the "Point-to-Point" based ICP algorithm [16, 8] can be employed to match the road and viewpoints. Meanwhile, the viewpoints can be registered in the road model by the estimated transformation parameters $\mathbf{R}_W^G, \mathbf{t}_W^G$.

Point Selection

The feature correspondences can not be utilized in the following objective function directly, since some points are not located in the image after projection in actual. Thus, the aim of this paragraph is to select the 3D point $\mathbf{P}_i^{j_k}$ (the kth feature point in the jth line) belonging to the ith frame. The principle to select points of the road model for a specific viewpoint is that the points of the road can be seen in this viewpoint. In other words, the points must locate in the region of the view frustum. Figure 3.4 illustrates the view frustum, which is determined by five planes (up plane, down plane, left plane, right plane, image plane). Given the intrinsic matrix $\mathbf{K}_i$ and image size, the five planes can be expressed as Eq. (3.2), where $\mathbf{p}_{i,i\in\{ul,dl,dr,ur\}}$ denote the four corner points (up-left, down-left, down-right, up-right point) in the pixel coordinate; $\mathbf{n}_{i,i\in\{d,l,r,u\}}$ denote the normal vectors of four planes (shown in Fig. 3.4); f is the focal length of the camera and $\mathbf{P}_f$ denote any 3D point located on the view frustum. Thus, the region of the view frustum $D(f)$ can be expressed in Eq. (3.3).

$$
\begin{aligned}
& z - f = 0; \mathbf{n}_i^\top \mathbf{P}_f = 0; i \in \{u, d, l, r\} \\
& where,\ \mathbf{n}_l = (\mathbf{K}_i^{-1}\mathbf{p}_{ul}) \times (\mathbf{K}_i^{-1}\mathbf{p}_{dl}); \\
& \mathbf{n}_u = (\mathbf{K}_i^{-1}\mathbf{p}_{ul}) \times ((\mathbf{K}_i^{-1}\mathbf{p}_{ur})) \\
& \mathbf{n}_r = (\mathbf{K}_i^{-1}\mathbf{p}_{dr}) \times ((\mathbf{K}_i^{-1}\mathbf{p}_{ur})); \\
& \mathbf{n}_d = (\mathbf{K}_i^{-1}\mathbf{p}_{dr}) \times ((\mathbf{K}_i^{-1}\mathbf{p}_{dl})
\end{aligned}
\tag{3.2}
$$

$$D(f) = \{\mathbf{p} | \mathbf{n}_i^\top \mathbf{p} > 0; z - f > 0; i \in \{l, r, u, d\}\} \tag{3.3}$$

In terms of selecting the suitable points for the ith frame, all the points of road model firstly are converted into the ith camera coordinate by the rotation matrix $\mathbf{R}_G^{C^i}$ and translation vector $\mathbf{t}_G^{C^i}$. Then the point $\mathbf{p}$ satisfying the Eq. (3.3) is located in the view frustum $D(f)$ (like the red points in Fig. 3.4). Finally, an amount of Q points near to the viewpoint of the ith frame are selected to construct the feature correspondence $\mathbf{l}_i^j$ and $\mathbf{P}_i^{j_k}$ $(k = 1, ..., Q)$.

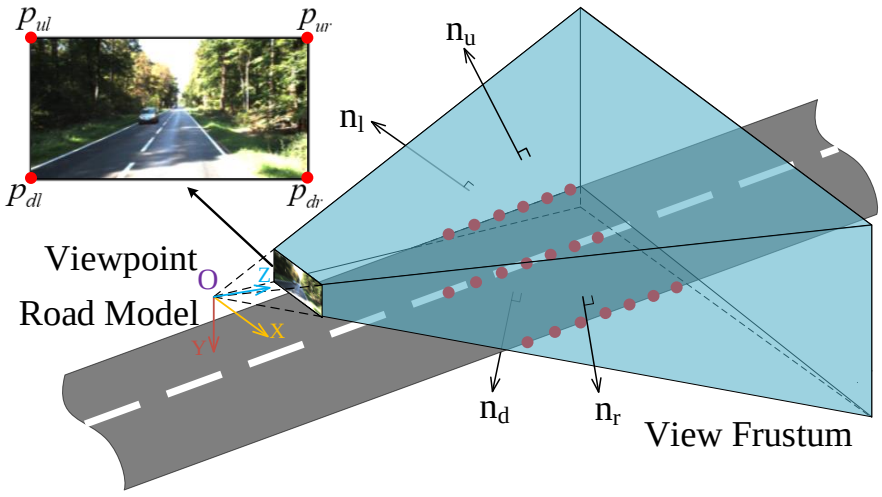


Figure 3.4: Utilizing the view frustum of each frame to select points of the road. The blue region is the frustum determined by five planes. The red points on the road model are located in the frustum.

Pose Refinement

The objective function is designed to refine the transformation matrix $\mathbf{R}_W^G$, $\mathbf{t}_W^G$ of the road model in this paragraph. In terms of the ith frame, there exist an amount of N feature lines $\{\mathbf{l}_i^1, \mathbf{l}_i^2, ..., \mathbf{l}_i^N\}$, an amount of Q 3D points $\mathbf{P}_i^{j_k}$ $(k{=}1,...,Q)$ corresponding to the feature line $\mathbf{l}_i^j$. The pose of the ith viewpoint, i.e. $\mathbf{R}_G^{C_i}$, $\mathbf{t}_G^{C_i}$, has been obtained in section 5.1. As shown in Fig. 3.5, the pixel point $\tilde{\mathbf{p}}_i^{j_k}$ in the ith frame is calculated after transformation and projection by providing the transformation parameters $\mathbf{R}_G^{C_i}$, $\mathbf{t}_G^{C_i}$, $\mathbf{R}_W^G$,

$\mathbf{t}_W^G$ and intrinsic matrix $\mathbf{K}_i$ (Eq. (4.3)).

$$\begin{aligned} \tilde{\mathbf{p}}_i^{j_k} &= \mathbf{K}_i \begin{bmatrix} \mathbf{R}_G^{C_i} & \mathbf{t}_G^{C_i} \end{bmatrix} \begin{bmatrix} \mathbf{R}_W^G & \mathbf{t}_W^G \\ \mathbf{0} & 1 \end{bmatrix} \begin{bmatrix} \mathbf{P}_i^{j_k} \\ 1 \end{bmatrix} \\ s.t. \quad & \mathbf{R}_W^G \in SO(3) \end{aligned} \tag{3.4}$$

However, the $\tilde{\mathbf{p}}_i^{j_k}$ is in homogeneous format. Thus, the vector $\mathbf{m} = \begin{bmatrix} 0 & 0 & 1 \end{bmatrix}$ is employed to extract the z value of the projected point $\tilde{\mathbf{p}}_i^{j_k}$, and the final point in the pixel coordinate is $\mathbf{p}_i^{j_k} = \frac{\tilde{\mathbf{p}}_i^{j_k}}{\mathbf{m}\tilde{\mathbf{p}}_i^{j_k}}$. Then, the Euclidean distance between the projected point and corresponding line is calculated by the following equation.

$$D_i^{j_k} = \|(\mathbf{l}_i^j)^T \mathbf{p}_i^{j_k}\|_2 \tag{3.5}$$

In terms of all viewpoints, the summation of all the Euclidean distances forms the objective function as follows:

$$\begin{aligned} E(\mathbf{R}_W^G, \mathbf{t}_W^G) &= \sum_{i=1}^{M} \sum_{j=1}^{N} \sum_{k=1}^{Q} D_i^{j_k} \\ s.t. \quad & \mathbf{R}_W^G \in SO(3) \end{aligned} \tag{3.6}$$

In the ideal situation, the projected point should be located on the corresponding line after a series of transformations. Thus, the accurate pose $(\mathbf{R}_W^G, \mathbf{t}_W^G)$ of the road model is determined through minimizing the objective function. The matrix $\mathbf{R}_W^G$ has only three degrees of freedom (DoF) and satisfies the constraint $SO(3)$ in Eq. 3.6. Thus, the matrix $\mathbf{R}_W^G$ is replaced by the rotation axis $\mathbf{v}$ and angle θ, namely Rodrigues [19] equation $\mathbf{R}_W^G = cos(\theta)\mathbf{I} + sin(\theta)[\mathbf{v}]_\times + (1 - cos(\theta))\mathbf{v}\mathbf{v}^T$. After the substitution, the final format of the objective function is shown in Eq. (4.7), where the rotation axis $\mathbf{v}$ is the unit vector; the axis degree θ is in the range of $-\pi/3$ and $\pi/3$ radian; and the components of the

translation vector $\mathbf{t}_W^G$, i.e. $t_{W,x}^G$, $t_{W,y}^G$, $t_{W,z}^G$, are in range of -5 and 5 meter.

$$
\begin{aligned}
E(\mathbf{v},\theta,\mathbf{t}_W^G) &= \min \sum_{i=1}^{M}\sum_{j=1}^{N}\sum_{k=1}^{Q} D_i^{jk}(\mathbf{v},\theta,\mathbf{t}_W^G) \\
s.t.\ \|\mathbf{v}\|_2 &= 1 \\
-\pi/3 < \theta &< \pi/3 \\
-5 < \mathbf{t}_{W,i}^G &< 5,\ i \in x,y,z
\end{aligned}
\tag{3.7}
$$

For optimization, it is hard to get close-form solution for the objective function (Eq. (4.7)). Thus, the gradient descent method based iterative scheme is employed for optimization. Meanwhile, a good initial value and suitable step size is must. In actual, the initial value of the rotation axis $\mathbf{v}$, rotation angle θ and translation vector $\mathbf{t}_m$ are the normal vector of the road surface, 0 degree and zero vector respectively. The interior algorithm [83] is exploited for iterative updating. To avoid obtaining the local minimum by single initial value, we exploit the scatter-search mechanism [82] to acquire multiple initial values in the scope of constraints. The global minimum can be found by comparing all the local optimums. The whole process is

After estimating the accurate pose of the road model, the road and viewpoints have been registered in the world coordinate. Besides, they can be converted into the global coordinate (WGS84). The conversion is achieved by two steps: from ENU to ECEF and from ECEF to WGS84, which is the inverse process of that in section 4.1.

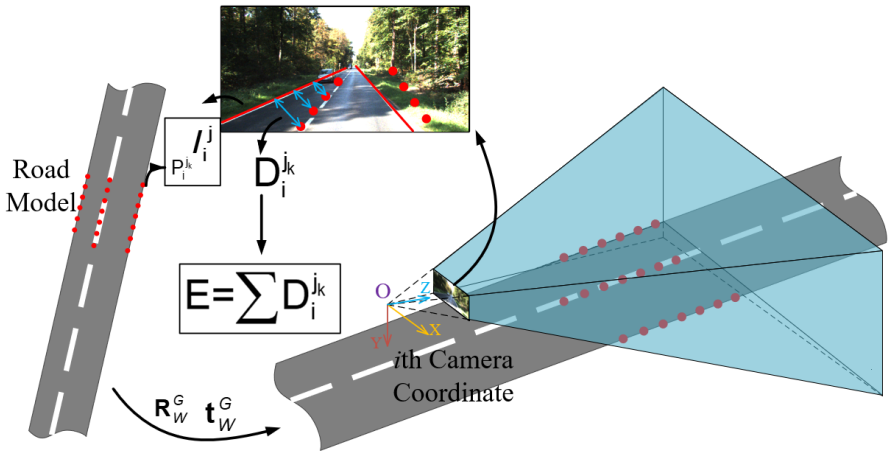


Figure 3.5: The explanation figure for objective function construction

Algorithm 1 The coarse-to-fine registration for road pose estimation

Require: The line correspondence, the transformation $(\mathbf{R}_G^{C_i}, \mathbf{t}_G^{C_i})$ between cameras and reference coordinate and intrinsic matrix $(\mathbf{K}_i)$ of ith camera.
Ensure: the transformation $(\mathbf{R}_W^G, \mathbf{t}_W^G)$ between reference coordinate and word coordinate.
1: **Step1:** utilizing the ICP method [16, 8] to estimate the $\mathbf{R}_W^G$, $\mathbf{t}_W^G$ coarsely
2: **Step2:** projecting the 3D lines into camera coordinates. For ith camera and , finding the point-to-line correspondence $(\mathbf{l}_i^j, \mathbf{P}_i^{j_k}, (k = 1, ..., Q))$ by Eq. (3.2), (3.3).
3: **Step3:** constructing the objective function (Eq. (4.7)) with the point-to-line correspondences, viewpoint poses $(\mathbf{R}_G^{C_i}, \mathbf{t}_G^{C_i})$ and camera intrinsic parameter $(\mathbf{R}_i)$.
4: **Step4:** the interior algorithm [83] is employed for refining the coarse pose $(\mathbf{R}_W^G, \mathbf{t}_W^G)$ by minimizing the objective function (Eq. (4.7)).
5: **return** $\mathbf{R}_W^G$, $\mathbf{t}_W^G$;

3.6 Experiments

In this section, two kinds of experiments are conducted to validate the effectiveness of the proposed method. One is the quantitative experiments to evaluate and analyze the 3D-2D registration accuracy. Another is the qualitative experiments to illustrate the 3D registration results. Meanwhile, the comparison results with state of the art are conducted in both quantitative and qualitative experiments to prove our superiority. Before these, the experimental setting including the dataset, metric, ground truth and comparison methods, are introduced first.

3.6.1 Experiment Setting

Dataset

The famous traffic dataset KITTI [29], containing four image sequences, lidar and IMU/GPS, is employed to evaluate the proposed method. As shown in table 3.1, six groups of raw dataset are selected, each of which contains 227, 214, 240, 240, 188, 169 frames respectively.

3.6.2 Quantitative Experiments

In this section, the aforementioned IoU metric is utilized to evaluate the registration results first. Then we explore how 'viewpoint registration' module of the proposed framework contribute to the registration results. Finally, the comparison experiments with state-of-the-art are conducted to valuate the superiority.

Table 3.1: The description of the selected KITTI dataset

KITTI	City0926_0014	City0926_0056	City0926_0096	Road0926_0015	Road0926_0027	Road0926_0028
Frames	227	214	240	240	188	169
description	straight	right curved	right curved	left curved	straight	right curved

Table 3.2: The statistical results of IoU scores in "city" scene

Method	City0926_0014				City0926_0056				City0926_0096			
	IoUscores			Mean	IoUscores			Mean	IoUscores			Mean
	>90%	>94%	>98%		>90%	>94%	>98%		>90%	>94%	>98%	
Ours	**92.07%**	39.65%	**3.08%**	**93.35%**	**100%**	**97.20%**	**48.60%**	**97.37%**	**96.67%**	**77.50%**	19.58%	**95.70%**
GPS	81.06%	**40.09%**	0.88%	93.06%	78.04%	58.41%	9.35%	93.94%	80.42%	60.83%	**41.67%**	94.96%
proSlam	37.44%	1.32%	0%	86.18%	89.25%	49.07%	0%	93.23%	90%	66.25%	0.42%	94.14%

Table 3.3: The statistical results of IoU scores in "road" scene

Method	Road0926_0015				Road0926_0027				Road0926_0028			
	IoUscores			Mean	IoUscores			Mean	IoUscores			Mean
	>90%	>94%	>98%		>90%	>94%	>98%		>90%	>94%	> 98%	
Ours	**94.17%**	**72.08%**	**34.17%**	**95.65%**	**100%**	**100%**	**36.70%**	**97.55%**	**100%**	**82.25%**	**10.65%**	**95.84%**
GPS	82.92%	63.75%	15.83%	94.37%	64.89%	32.45%	0.00%	90.46%	69.23%	47.93%	7.10%	91.94%
proSlam	0.00%	0.00%	0%	73.05%	51.60%	32.45%	1%	90.84%	69.82%	33.14%	0.00%	91.63%

Registration Results

The registration results of the framework are shown in Fig. 3.6. The top of each sub figure shows the road boundaries (red point), lanes (blue point) and viewpoints (black point) in the vertical view. Obviously, the viewpoints are located between the right road boundary and central lane, which indicates that the vehicle keeps right for driving and accords with the reality. In the bottom of each sub figure, the IoU scores of the proposed method of six image sequences are displayed by red line. Furthermore, the statistical results of six image sequences are summarized in Table 3.2, 3.3, with percentages of frames achieving predefined IoU scores. In Table 3.2, 3.3, the proposed method can achieve more than 93% mean of IoU scores. What's more, the IoU scores of more than 94% frames exceed 90% in six image sequences. As a result, the projected regions of the 3D road model basically accord with the road regions of the image sequences after registration.

Component Analysis

The aim of this experiment is to test how the results of 'viewpoint registration' (i.e. $\mathbf{R}_G^{C_i}, \mathbf{t}_G^{C_i}$) impact the registration accuracy. Thus, the three different methods (i.e. ORB-SLAM2[63], GPS and proSlam [73]) are realized while the rest of the framework keeps invariant.

The compared results of the framework with three various methods are shown in Fig. 3.6. The bottom of each sub figure shows the IoU scores of each frame of six sequences by three methods. It is obvious that the framework with ORB-SLAM2 [63] (red lines) totally has better performance than the other twos (blue and cyan lines). Besides, the framework with proSlam [73] has the worst performance. In addition, compared to the other two methods, the framework with ORB-SLAM2 [63] is more stable. In Table 3.2, 3.3, it is also proven that the proposed method outperforms the other twos.

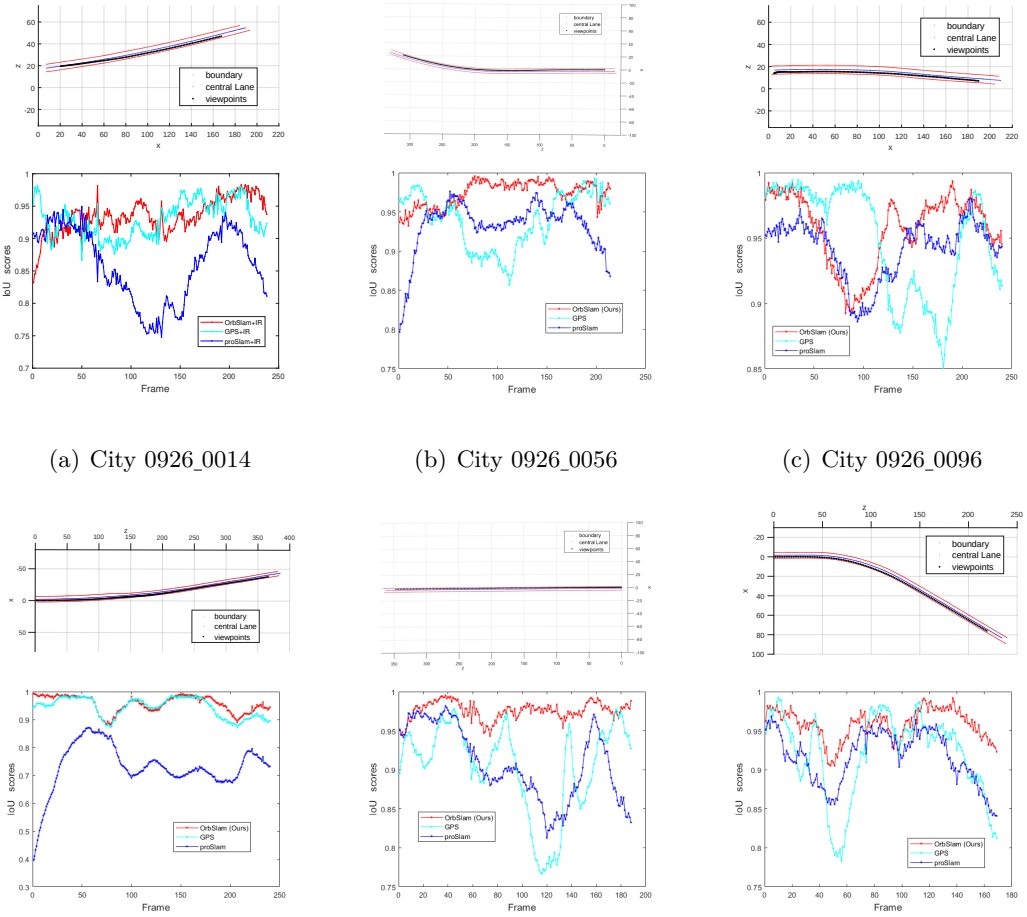
In actual, the performance of ORB-SLAM2 [63] is better than that of proSlam [73], which indicates the accurate intermediate results (i.e. viewpoint poses) contribute to the good registration.

Table 3.4: Compared to the state of the art in the “city” scene

Method	City0926_0014				City0926_0056				City0926_0096			
	IoUscores			Mean	IoUscores			Mean	IoUscores			Mean
	>90%	>94%	>98%		>90%	>94%	>98%		>90%	>94%	>98%	
Ours	**92.07%**	**39.65%**	**3.08%**	**93.35%**	**100%**	**97.20%**	**48.60%**	**97.37%**	**96.67%**	**77.50%**	**19.58%**	**95.70%**
Lee [47]	0%	0%	0%	67.75%	32.24%	21.50%	8.41%	85.46%	44.58%	15.83%	2.5%	90.27%
NPnLupC [38]	14.10%	0.44%	0%	81.35%	**100%**	83.18%	36.92%	96.53%	79.17%	57.92%	16.67%	94.22%
NPnLupL [38]	28.19%	11.89%	0%	85.89%	0%	0%	0%	38.77%	9.58%	0%	0%	86.26%

Table 3.5: Compared to the state of the art in the “road” scene

Method	Road0926_0015				Road0926_0027				Road0926_0028			
	IoUscores			Mean	IoUscores			Mean	IoUscores			Mean
	>90%	>94%	>98%		>90%	>94%	>98%		>90%	>94%	> 98%	
Ours	94.17%	**72.08%**	**34.17%**	**95.65%**	**100%**	**100.00%**	**36.70%**	**97.55%**	**100.00%**	**82.25%**	**10.65%**	**95.84%**
Lee [47]	89.17%	36.25%	2.92%	93.16%	54.26%	27.13%	0.00%	89.51%	69.23%	38.46%	3.55%	92.07%
NPnLupC [38]	**95.83%**	71.25%	5.83%	95.18%	54.26%	32.45%	0%	89.86%	90.53%	50.30%	9.47%	93.83%
NPnLupL [38]	25.42%	13.33%	0%	72.82%	0%	0%	0%	44.84%	11.83%	5.33%	0.00%	32.27%



(a) City 0926_0014 (b) City 0926_0056 (c) City 0926_0096

(d) Road 0926_0015 (e) Road 0926_0027 (f) Road 0926_0028

Figure 3.6: IoU scores of image sequences

Metric

The proposed method is evaluated in the image sequence space. Specifically, as long as the 3D road model is located in the accurate position after registration, the projection of the road model will strictly consist with the road region of each frame of the video. Thus, the overlapped ratio between the projected region of the road model and the road region of the image is decided by the pose of the road model. One of famous metric to measure the overlapped ratio is "intersection over union" (IoU), which has been widely utilized in the tasks of 3D object tracking, pose estimation of 3D objects and object detection.

Ground Truth

As stated in the metric part, the IoU metric is employed for evaluation. Thus, the ground truth of each image is expressed as a set of pixels of the road region. To acquire the ground truth for each frame, we label the boundaries of the road by a set of points and utilize "inpolygon" matlab function to acquire all the pixels of the road region.

Comparison Method

The state-of-the-art methods including Lee [47], NPnLupC [38] and NPnLupL [38], are employed for comparison. For the fair comparison, the state-of-the-art methods are re-implemented by the Matlab 2019a software.

Comparison to State-of-the-Art

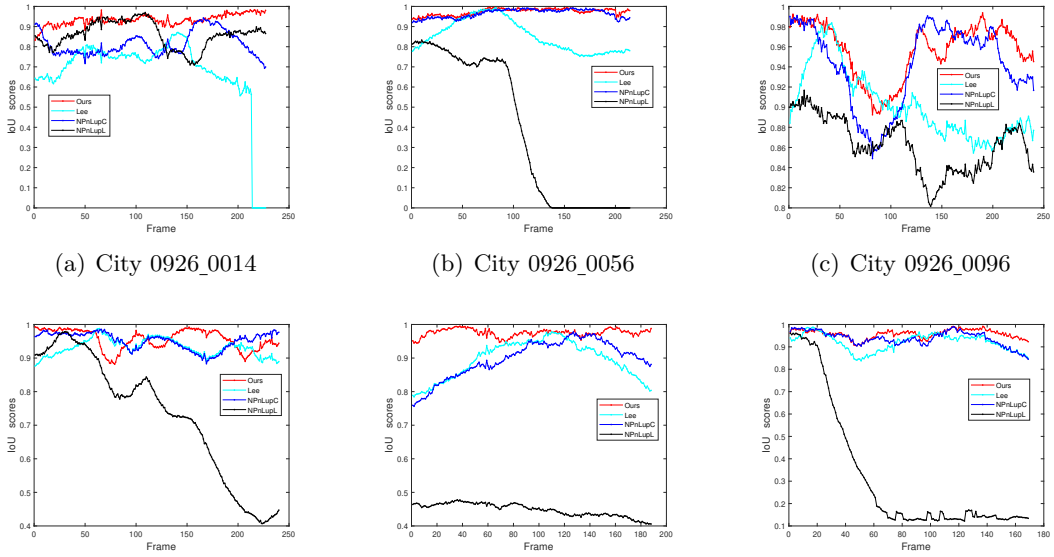
This experiment aims to validate the superiority of ours by comparing to methods in Lee [47], NPnLupC [38] and NPnLupL [38]. For fair comparisons, the feature correspondences and the relative poses of cameras (i.e. $\mathbf{T}_G^{C_i}$) estimated by ORB-SLAM2 [63] are kept the same and put into ours, Lee[47], NPnLupC [38] and NPnLupL [38] methods.

The IoU score of each frame in traffic videos are shown in Fig. 3.7. The IoU scores by NPnLupL [38] method falls lower than 0.5 in the traffic video of 'City 0926_0056', 'Road 0926_0015', 'Road 0926_0027' and 'Road 0926_0028'. Besides, the IoU scores by Lee [47] method reach to zeros in the end of 'City0926_0014', which indicates this method fails in this traffic video. In total, our method has better performances than other threes. The statistical IoU results are displayed in Table 3.4, 3.5. It is obvious that our method outperforms other threes in almost six traffic videos.

3.6.3 Qualitative Experiments

The registration results of two sequences (i.e. Road0926_0027, Road 0926_0028) are shown in Fig. 3.8. The top of the Fig. 3.8(a)(b) shows the road and viewpoints of cameras in the vertical direction after registration. The red points are the viewpoints of all frames, which also indicates the trajectory of the vehicle. The black points in the top of Fig. 3.8(a)(b) represent the road boundaries. Besides, Fig. 3.8(a),(b) utilize the red pixels

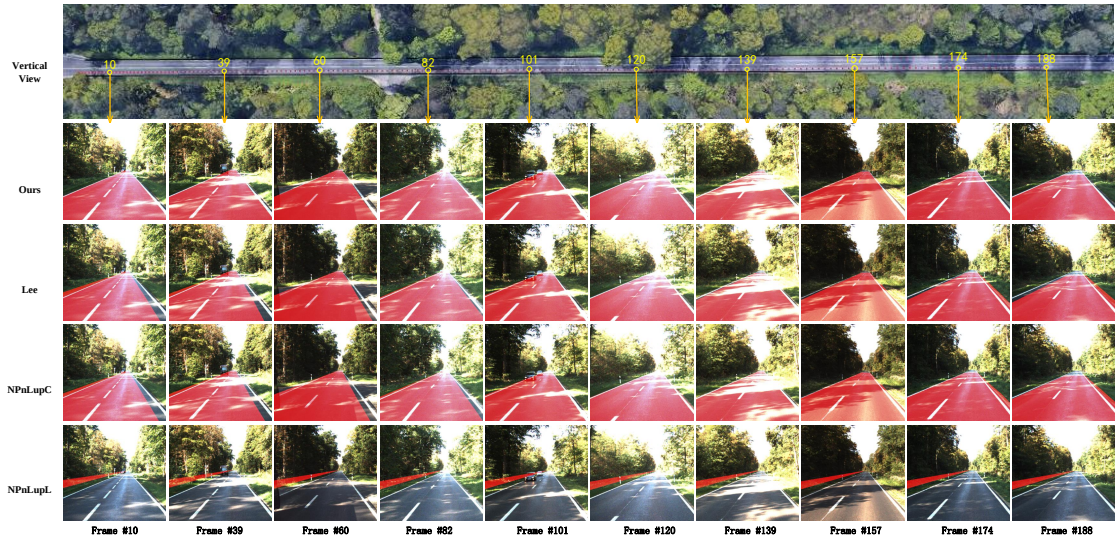
to show the projection region of the road model by our method, Lee [47], NPnLupC [38] and NPnLupL [38]. In Fig. 3.8(a), since NPnLupL [38] method fails in the image sequence of 'Road 0926_0027', the projected rejoin does not consist with the road region completely. Besides, our performance is slightly better than those of Lee [47] and NPnLupC [38] in the 157th, 174th and 188th frame. Furthermore, our method obviously outperforms the Lee [47], NPnLupC [38] in the 10th, 39th, 60th, 82th frame. In terms of the 'Road0926_0015', our performance exceeds those of Lee [47], NPnLupL [38] in the 158, 188, 210 and 230 frame. Besides, our method is a little better than NpnLupC [38] in the 188, 210 frame.



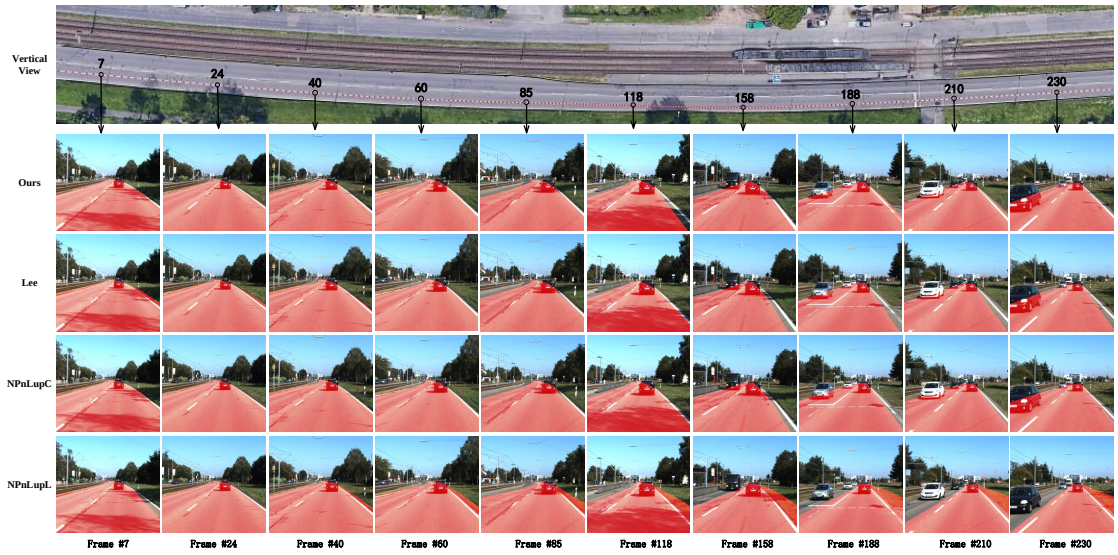
(a) City 0926_0014 (b) City 0926_0056 (c) City 0926_0096

(d) Road 0926_0015 (e) Road 0926_0027 (f) Road 0926_0028

Figure 3.7: IoU scores of image sequences by ours, Lee [47], NPnLupC [38] and NPnLupL [38]



(a) Road0926_0027



(b) Road0926_0015

Figure 3.8: The visualization of registration results in the Google software. The black and red points indicate the road boundaries and the camera trajectory respectively in the vertical view. The black points in each frame are the boundary points by projecting road model. (a) The registration results of Road0926_0027 (b) The registration results of Road0926_0015.

3.7 Summary

This chapter proposes a coarse-to-fine registration method for non-perspective pose estimation problem in the traffic scenario, which mainly contains two steps: 1) The ICP method is employed to estimate a coarse pose. 2) The coarse pose is iteratively refined by the pre-defined objective function. Besides, a traffic video augmentation framework combining multi-modal data (traffic videos and 3D road model) is put forward by three stages. Firstly, the preprocessing includes 3D-2D feature correspondence acquisition and 3D road model. Subsequently, the 3D-2D registration is realized by estimating the poses of all the viewpoints and road model. Finally, the registered viewpoint, road model and 3D model of new traffic elements are put into the 3D graphic engine for traffic simulation. The augmented videos are generated by fusing the projected traffic elements and original frames. Both quantitative and qualitative experiments prove the superiority of our method compared with state of the art in the KITTI dataset. The contributions of this chapter are presented as follows:

1) To the best of our knowledge, it's the first time to introduce the non-perspective pose estimation in the traffic scenario. Besides, we propose a new coarse-to-fine two-stage scheme to estimate the road pose. Within the scheme, an objective function combining the 3D-2D pre-defined feature correspondences is designed for pose refinement. The comparison experiments verify the effectiveness and superiority of our method.

2) This chapter provides a convenient method to make the wire-frame road model from GIS information. Generally, many GIS systems, such as Google Earth, contain the free and global road information. Through this way, the wire-frame model of any road can be extracted and made instead of field measurement.

Chapter 4

Central-line Model based Road Structure and Pose Estimation

In the problem of road pose estimation, since the road model is hard to obtain, this chapter attempt to utilize the central line of a road instead. In the content of this chapter, we first study how to represent a parameterized road based on the central line information. Then, the objective function combining the parameterized road and pose is deduced step by step. The details of experiments including metric, dataset, results and analysis are displayed. Finally, we summarize this chapter.

4.1 Introduction

3D road model has been widely applied in the fields of the road surface inspection [26, 27], autonomous driving navigation [33], traffic scene simulation [50, 102, 52, 98], and multi-modularity data augmentation [2, 49, 41]. Thus, many researchers are devoted to construct such 3D road models through various methods. Most early, researchers utilize vision-based methods to construct the road from remote sensing images [104], which generally contain a road network of a region. These methods are restricted by the resolution of the remote images. Currently, with the rapid development of measurement sensors (lidars scans, cameras), many institutions and corporations [31, 20, 21] utilize the vehicle-mounted lidar scans and cameras to capture multi-modality data of the urban or rural road, which can be utilized for not only recognition algorithm evaluation but also road reconstruction. At

present, according to the various categories of the traffic data, the reconstruction methods can be classified into vision-based, lidar-based and multi-modal based methods. Laser scanners can acquire the 3D information of the traffic environment, which makes it easy to reconstruct the road. However, due to lack of color information, the difficulty of the road construction is detecting the road from the whole traffic environment without obvious road edges or boundaries. Compared to the laser scanner, vehicles equipped with cameras has the advantages of the low cost and acquisition of the color information of the traffic environments. Thus, those including stereo-vision [26, 62] based, SfM-based (Structure from Motion) and CNN-based [103] methods have been proposed for the traffic scene reconstruction. In order to complement advantages of various sensors, some researchers adopt MLS systems [13] (mobile laser scanning), which includes laser scanners, cameras etc., to capture multi-modality traffic data and reconstruct the road in real-time. Taking advantages of traffic images and 3D cloud points contributes to improve the reconstruction performance.

Whatever kinds of traffic data, those methods should solve three problems to achieve the road reconstruction: detection, construction and localization. Generally, detection is to determine the road part from the traffic data. Construction is to reconstruct a road segment from the one or few data frames. Since the road segments reconstructed from few data frames are too short to satisfy some application demands (surface inspection, autonomous driving etc.), localization is needed to estimate the position and orientation to joint the adjacent road segments. Currently, most existing reconstruction methods aim to recover the layout of the whole scene. As for the object of the scene (such as roads), accurate detection makes a big difference to this task. In terms of images or videos, due to the illumination, occlusion and shadow, the state-of-the-art methods hardly achieve accurate detection in these conditions, which severely affects the reconstruction performance. Hence, it's challenging to reconstruct the road from the images based on the some false road detection.

This chapter also aims to reconstruct the traffic road from the traffic videos. In order to overcome the influence of false road detection from the images, this chapter imports the GIS (Geographic Information System) information as a constraint. Specifically, we import the central line of a road from the GIS information and utilize the central line to

parameterize the road boundaries. Theoretically, the projection of the parameterized road boundaries should strictly match with those from the images after projection. Thus, the parametric road boundaries and those from images establish the feature correspondences. Meanwhile, the objective function integrates the feature correspondences to minimize the projection errors for road reconstruction. However, the roads detected from the images inevitably exist false. Thus, in the optimizing stage, the RANSAC (Random Sample Consensus) [81] method is exploited to eliminate the false detection and estimate the optimizing parameters simultaneously. Different from the previous methods, the roads are reconstructed through estimating the attributes: position, orientation and structure. The designed objective function exploits all the frames of the traffic videos to globally estimate the road attributes.

4.2 Related works

At present, 3D scene construction has been a tradition and practical study in the computer vision community. The amounts of existing vision-based 3D reconstruction methods can be classified two categories: bottom-to-top and top-down methods.

Those belonging to bottom-to-top methods generally extract the bottom pixel features and establish the correspondences for 3D reconstruction, which has no assumptions and limitations for the scene. Thus, these methods have advantages of being applied in various scenes. One kind of these methods is called SfM (Structure from Motion), which generally extracts and establishes feature correspondences from a set of non-sequential images, utilizes the triangle methods to recover the 3D cloud points from the correspondences and register the images simultaneously. These kinds of methods have three various strategies in matching feature correspondences, which totally called incremental SfM [1, 78, 74], global SfM [93, 12] and hybrid SfM [22, 105]. With the rapid development of autonomous driving technologies, SLAM [42] (simultaneous localization and mapping) technologies are advocated to widely study. These technologies generally construct the scenes and determine the location of the vehicles, which contributes to navigations. Currently, the deep learning technologies have made a revolutionary breakthrough in many fields of the computer vision. Hence, many researchers attempt to train the neural networks to recon-

struct the scene. For example, Zhou et al. [103] trained the pose and depth neural nets simultaneously by unsupervised learning.

In terms of top-down methods, these methods generally have the information about the scene structure, which are utilized as prior knowledges or constraints to improve reconstruction performance. Hence, according to the various scenes, these methods can add flexible and regular structure constraints. A typical example is the "floor-wall" model in [52], which assume the traffic scenes consist of three components: floor, wall, sky. The "floor", "wall" and "road" represent the road surface, the objects in the both sides of the road (buildings, trees) and sky respectively. Through the "floor-wall" model, the traffic environment is simulated and augmented.

The proposed method in this chapter belongs to the second category. In actual, plenty of roads are man-made and of the regular shapes. Thus, 3D roads first are modeled by the road width parameter and the central line of the road. The reason utilizing such modeling way is that the central lines of the roads are convenient and free to extract from the GIS softwares, such as OSM (Open Street Map), Google Earth. Besides, it is effective way to express a road utilizing the central lines and road width. In addition, as stated in the Sect. 4.1, combination of the central lines and the objective function has effects on reducing the influence of false road detection.

4.3 Proposed Method

4.3.1 Problem Formulation

As stated in Sect. 4.1, given two videos O_l, O_r by binocular cameras, camera parameters (including intrinsic matrix $\mathbf{K}$ and baseline of two cameras ($\mathbf{B}$)) and central lines from GIS information (G_c), the road is constructed through estimating the attributes of the road. This chapter introduces two coordinates to measure the poses of cameras and the road respectively. Thus, the poses of the road (i.e. $\mathbf{R}_m$, $\mathbf{t}_m$) can be regarded as the transformation between these two coordinates. The road structure is jointly determined by the central lines and road width w. Hence, the structure of the road can be determined by estimating the road width.

As shown in Fig. 4.1, the poses of the viewpoints $\mathbb{T}$ are estimated from the traffic

videos O_l, O_r by binocular cameras. Meanwhile, the road boundaries $\mathbb{L}$ in each frame of the traffic videos are extracted by the neural network. Based on the central lines from the GIS (G_c) and road width w, the 3D road boundaries are parameterized as $\mathbb{P}(w)$. The pixel distances between the projection of the road boundaries $\mathbb{P}(w)$ and boundary line $\mathbb{L}$ formulate the objective function. Thus, the formulation of this problem is presented in Eq. 4.1, which is detailed in Para. 4.3.3. The parameters can be estimated through minimizing this objective function.

$$\mathbf{R}_m, \mathbf{t}_m, w = \underset{\mathbf{R}_m, \mathbf{t}_m, w}{\arg\min} \quad F(\mathbb{L}, \mathbb{P}(w), \mathbf{K}, \mathbf{B}, \mathbf{R}_m, \mathbf{t}_m, w) \tag{4.1}$$

4.3.2 Feature Correspondences

3D Parametric Road Model

This paragraph introduces how to model a road based on the central line. Besides, in the process of modeling, the parameterized representation of a road is also provided for the following usage.

As is shown in Fig. 4.2, the process of modeling a road includes four stages: input, parameter computation, boundary representation and surface generation. The **input** is the central line of a road from GIS information. As presented red points in Fig. 4.2(b), the central line is denoted as a set of ordered points $\mathbb{P}_c = \{\mathbf{P}_c^i | i = 1, .., N\}$. In the stage of **parameter computation**, the lateral vectors of the road are estimated based on the points of the central line $\mathbb{P}_c$. As illustrated in the Fig. 4.2(c), the lateral vectors (green

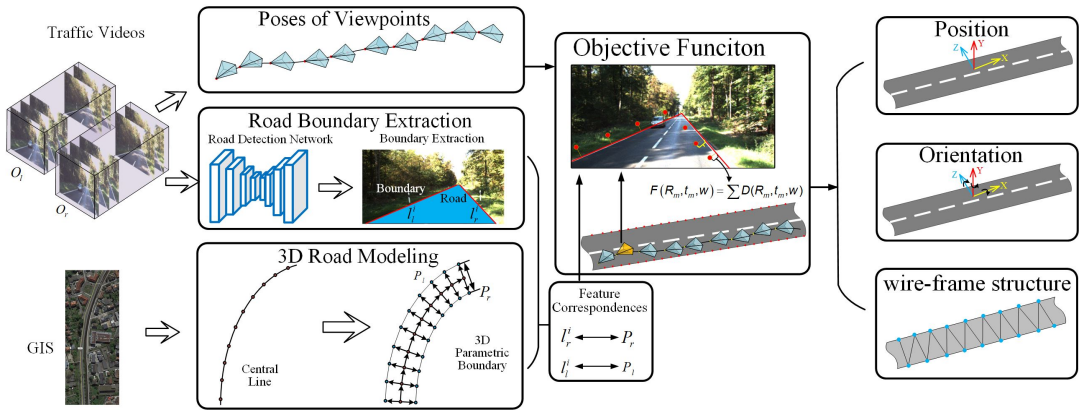


Figure 4.1: The pipeline of road reconstruction

vectors) are parallel to the road surface and vertical to the forward vectors (purple vectors) of the road. The process of estimating the lateral vectors is as follows. To estimate the lateral vectors of each point on the central line, the normal vector of the road surface is firstly estimated by utilizing [81] method on the points of the central line. Secondly, the forward vectors of each point on the central line are calculated by subtracting the adjacent points. Finally, the lateral vector of each point is calculated by the cross product of the normal and forward vector. After acquiring the lateral vectors $\mathbf{v}_c^i (i = 1, ..., N)$ of each point, the boundary point is estimated in the stage of **boundary representation**. The points on boundary lines (blue points in Fig. 4.2(d)) are calculated by the Eq. (4.2), where $\mathbf{P}_l^i$, $\mathbf{P}_r^i$ and w represent the ith left, right boundary points and width of the road respectively.

$$\begin{cases} \mathbf{P}_l^i = \mathbf{P}_c^i + \frac{w}{2} \times \mathbf{v}_c^i \\ \mathbf{P}_r^i = \mathbf{P}_c^i - \frac{w}{2} \times \mathbf{v}_c^i \end{cases} \tag{4.2}$$

In the end, the wire-frame model of a road is constructed by utilizing the algorithm to generate triangle meshes on the points of the boundaries (Fig. 4.2(e)).

According to the process of the road model, the road construction depends on two aspects: the central line of a road and width w of the road. Given the central line from GIS information, the structure of a road can be determined by estimating the road width w. For simplification, we utilize the sets $\mathbb{P}_l(w) = \{\mathbf{P}_l^i | i = 1, ..., N\}$ and $\mathbb{P}_r(w) = \{\mathbf{P}_r^i | i = 1, ..., N\}$ to represent left and right boundaries of a road.

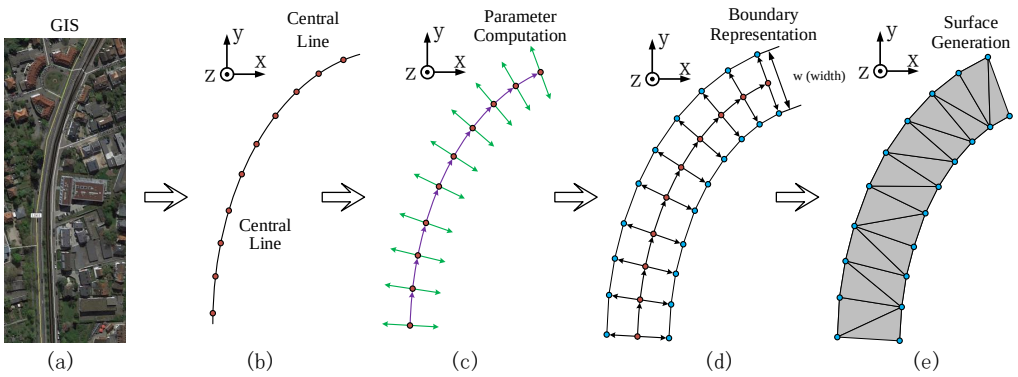


Figure 4.2: Road model based on the central line

Road Boundary Extraction

In order to extract the accurate road boundaries automatically from each frame of the traffic videos, we firstly detect the road surface. Recently, most deep learning methods have outstanding performance in the field of the semantic segmentation of traffic images. Thus, the one of the famous method (i.e. deeplab v3plus [15]) is exploited for road surface detection with some modifications. Concretely, this model consists of three parts: "encoder", "aspp" and "decoder". In the component of the "encoder" part, the resnet101 [37] net is exploited as the "backbone". Besides, a new softmax layer is added to the last layer of "decoder" to classify which category (road or non road) each pixel belongs to. After training, the road surface region of each image can be detected through the net.

After acquiring the detection results of the road surface, the road boundaries of each image is extracted based on the road region. Firstly, the max set $\mathbb{E}$ of the connective pixels is selected for succedent selection (as stated in step 1-2 of the Algorithm 2). Then, in the ith row of the image, the left and right pixels of the road region are regarded as points of left and right boundaries (as stated in step 3 of the Algorithm 2). As is shown in Fig. 4.3, the $\mathbf{p}_l^i$ and $\mathbf{p}_r^i$ stands for the left and right pixel in the ith row. Subsequently, the sets of left boundary $\mathbb{S}_l = \{\mathbf{p}_l^i | i = 1, ..., N\}$ and right boundary $\mathbb{S}_r = \{\mathbf{p}_r^i | i = 1, ..., N\}$ are obtained row by row. Finally, according to the point sets $\mathbb{S}_l$ and $\mathbb{S}_r$, the left and right boundaries $\mathbf{l}_l$, $\mathbf{l}_r$ can be linearly fitted using RANSAC method to eliminate the outliers (step 4 of algorithm 2). For the traffic video, we utilize the network to detect the road surface and extract the boundaries frame by frame. After road boundary extraction, the left and right boundaries of stereo cameras are denoted as $\mathbb{L}_l^l = \{\mathbf{l}_l^{l_i} | i = 1, ..., N\}$, $\mathbb{L}_l^r = \{\mathbf{l}_l^{r_i} | i = 1, ..., N\}$, $\mathbb{L}_r^l = \{\mathbf{l}_r^{l_i} | i = 1, ..., N\}$, $\mathbb{L}_r^r = \{\mathbf{l}_r^{r_i} | i = 1, ..., N\}$, where the subscript and superscript mean which boundaries and videos the lines belong to respectively.

Feature Correspondence

This paragraph aims to establish the point-to-line feature correspondences. Concretely, the 3D points on the road boundaries match with the road boundary line in the images. As stated in the paragraph 4.3.2, 4.3.2, the 3D points of left and right road boundaries are represented as sets $\mathbb{P}_l(w)$ and $\mathbb{P}_r(w)$; the road boundary lines of each frame of left and right

Figure 4.3: Road surface and boundary detection in the traffic image

Algorithm 2 Road boundary extraction from the detection results

Require: The road detection mask of image sequences, which defined as $M_1, .., M_N$
Ensure: The right $(l_r^1, ..., l_r^N)$ and left $(l_l^1, ..., l_l^N)$ road boundaries of each image.

1: **for** i=1 to the end of the mask sequence **do**
2: **Step1:** For mask M_N, finding the adjacent pixels from the road region and building the sets for each adjacent pixels.
3: **Step2:** Finding the set $\mathbb{E}$ with the max number of pixels elements.
4: **Step3:** Extracting the left and right boundary pixels from the the set $\mathbb{E}$ row by row. the pixel point sets of left and right boundaries are expressed as $\mathbb{S}_l$, $\mathbb{S}_r$
5: **Step4:** Utilizing the RANSAC method to fit the left l_l^i and right l_r^i boundary lines.
6: **end for**
7: **return** $\mathbb{L}_l^l = \{\mathbf{l}_l^{l_i} | i = 1, ..., N\}$, $\mathbb{L}_l^r = \{\mathbf{l}_l^{r_i} | i = 1, ..., N\}$, $\mathbb{L}_r^l = \{\mathbf{l}_r^{l_i} | i = 1, ..., N\}$, $\mathbb{L}_r^r = \{\mathbf{l}_r^{r_i} | i = 1, ..., N\}$;

videos are denoted as $\mathbb{L}_l^l = \{\mathbf{l}_l^{l_i} | i = 1, ..., N\}$, $\mathbb{L}_l^r = \{\mathbf{l}_l^{r_i} | i = 1, ..., N\}$, $\mathbb{L}_r^l = \{\mathbf{l}_r^{l_i} | i = 1, ..., N\}$, $\mathbb{L}_r^r = \{\mathbf{l}_r^{r_i} | i = 1, ..., N\}$. For the left video, the $\mathbb{P}_l(w)$ and $\mathbb{L}_l^l$, $\mathbb{P}_r(w)$ and $\mathbb{L}_r^l$, establish the feature correspondences, which is same with the right video.

4.3.3 Road Construction

Poses of Viewpoints

This paragraph aims to recover the poses (i.e. position and orientation) of viewpoints of the image sequences. As stated in Sect.4.3.1, the input contains two traffic image sequences by the binocular cameras, intrinsic parameters of the cameras and baseline **B** between the optical centers of binocular cameras. Based on these input data, the poses of cameras can be estimated by the ORB-SLAM2 [63] method. Without loss of generality, the $\mathbf{R}_i$, $\mathbf{t}_i$ and

$\mathbf{R}_i$, $\mathbf{t}_i + \mathbf{B}$ denote the poses of the viewpoints of the ith left image and right image.

Objective Function

In this paragraph, the objective function is designed to combine the poses of the viewpoints, feature correspondences for estimating the position, orientation and width of the road. The main principle of the objective function is to calculate the distances between the projected points of the 3D road boundaries and boundary lines in the traffic images. The parameters (position, orientation and width) are accurately estimated through minimizing the objective function.

Here, the derivation process regarding the left image sequence is introduced first. In the Sect. 4.3.2, the ith 3D left and right boundaries are denoted as $\mathbf{P}_l^i$ and $\mathbf{P}_r^i$ from the sets $\mathbb{P}_l$ and $\mathbb{P}_r$. Given the intrinsic $\mathbf{K}$, extrinsic parameters $\mathbf{R}_j$, $\mathbf{t}_j$ (Sect. 4.3.3) and the transformation $\mathbf{R}_m$, $\mathbf{t}_m$, the projected points in the jth left image coordinate is presented as follows

$$\tilde{\mathbf{p}}_l^i = \mathbf{K}(\mathbf{R}_j(\mathbf{R}_m\mathbf{P}_l^i + \mathbf{t}_m) + \mathbf{t}_j) \tag{4.3}$$

However, the point $\tilde{\mathbf{p}}_l^i$ is in the homogeneous format. Thus, the vector $\mathbf{m} = \begin{bmatrix} 0, & 0, & 1 \end{bmatrix}$ is exploited to extract the z value of projected point $\tilde{\mathbf{p}}_l^i$. The final point in the pixel coordinate is $\mathbf{p}_l^i = \frac{\tilde{\mathbf{p}}_l^i}{\mathbf{m}\tilde{\mathbf{p}}_l^i}$. Then the Euclidean distance between the point $\mathbf{p}_l^i$ and the correspondent line $\mathbf{l}_l^{l_j}$ in the jth left image is calculated by the following equation.

$$D_{l_i}^{l_j} = \|(\mathbf{l}_i^{l_j})^T\mathbf{p}_l^i\|_2 \tag{4.4}$$

Equation 4.4 is the distance of a feature correspondence (i.e. the ith left projected point and left boundary line in the jth image). The distances D_l^l between all the left feature correspondences in the left image sequence is calculated by summation of the distance of each correspondence (Eq. (4.5)).

$$D_l^l = \sum_{i,j} D_{l_i}^{l_j} \tag{4.5}$$

In the similar way, the D_l^r, D_r^l and D_r^r, which denote the projection errors of the left, right, right boundary points on the right, left, right images respectively, are calculated as

the same with Eq. 4.5 except that the translation vector $\mathbf{t}_j$ is replaced by $\mathbf{t}_j + \mathbf{B}$ for the jth right images.

Finally, as shown in Eq. (4.6), the summation of all distances in the left and right sequences formulates the objective function.

$$F(\mathbf{R}_m, \mathbf{t}_m, w) = D_l^l + D_l^r + D_r^l + D_r^r \tag{4.6}$$

As is known, the rotation matrix $\mathbf{R}_m$ contains nine parameters but only has three degrees of freedom. Thus, we utilize the rotation axis $\mathbf{v}_m$ and angle θ instead of the rotation matrix $\mathbf{R}_m$, namely Rodrigues equation $\mathbf{R}_m = cos(\theta)\mathbf{I} + sin(\theta)[\mathbf{v}_m]_\times + (1 - cos(\theta))\mathbf{v}_m\mathbf{v}_m^T$ [19]. The final format of the objective function is presented by Eq. (4.7). Besides, some constraints are supplemented for restricting the parameters. These constraints are that the rotation axis $\mathbf{v}_m$ is limited to an unit vector, the angle ranges from $-\pi/3$ to $\pi/3$ radian, and the components of the translation vector $\mathbf{t}_m$, i.e. $t_{m,x}$, $t_{m,y}$, $t_{m,z}$, are in range of -5 and 5 meter, the road width w is larger than 0 to accord with the actual situation.

$$\begin{aligned} F(\mathbf{v}_m, \theta, \mathbf{t}_m, w) &= D_l^l(\mathbf{v}_m, \theta, \mathbf{t}_m, w) + D_l^r(\mathbf{v}_m, \theta, \mathbf{t}_m, w) \\ &\quad + D_r^l(\mathbf{v}_m, \theta, \mathbf{t}_m, w) + D_r^r(\mathbf{v}_m, \theta, \mathbf{t}_m, w) \\ s.t.\ &\|\mathbf{v}_m\|_2 = 1 \\ & w > 0 \\ & -\pi/3 < \theta < \pi/3 \\ & -5 < \mathbf{t}_{m,i} < 5,\ i \in x, y, z \end{aligned} \tag{4.7}$$

Algorithm 3 The process of optimizing the objective function

Require: The road boundaries of left and right image sequences ($\mathbb{L}_l^l = \{\mathbf{l}_l^{l_1}, ..., \mathbf{l}_l^{l_N}\}$, $\mathbb{L}_l^r = \{\mathbf{l}_l^{r_1}, ..., \mathbf{l}_l^{r_N}\}$, $\mathbb{L}_r^l = \{\mathbf{l}_r^{l_1}, ..., \mathbf{l}_r^{l_N}\}$, and $\mathbb{L}_r^r = \{\mathbf{l}_r^{r_1}, ..., \mathbf{l}_r^{r_N}\}$). The intrinsic parameter $\mathbf{K}$ and baseline of two cameras $\mathbf{B}$. The parametric left and right road boundaries $\mathbb{P}_l(w)$, $\mathbb{P}_r(w)$.

Ensure: The best model parameters($\mathbf{M}_b$). The best error of the model (E_b).

▷

iter=0: the current iteration times.
iter_times=200: the total iteration times.
$\mathbb{C} = \emptyset$.
while iter<iter_times **do**
    Step1: Dividing each of the sets $\mathbb{L}_l^l$, $\mathbb{L}_l^r$, $\mathbb{L}_r^l$, $\mathbb{L}_r^r$ into four sub sets respectively according to the frame sequence, (such as $\mathbb{L}_l^l(1 - N/4)$, $\mathbb{L}_l^l(N/4 + 1 - N/2)$, $\mathbb{L}_l^l(N/2 + 1 - 3N/4)$, $\mathbb{L}_l^l(3N/4 + 1 - N)$). The frame number N of each image sequence.
    Step2: Randomly selecting the four groups of road boundaries from the four divided sets. For example, one group of road boundaries for frames (1-N/4) are $\mathbf{l}_l^{l_i}, \mathbf{l}_l^{r_i}, \mathbf{l}_r^{l_i}, \mathbf{l}_r^{r_i}, i < N/4$.
    Step3: Utilizing the four road boundaries to optimize the objective function (Eq. 4.7) by the interior algorithm [83]; Set $\mathbb{C} = \mathbb{C} \cup$ four groups of road boundaries.
    Step4:
    for $\mathbf{l}_l^{l_i}, \mathbf{l}_l^{r_i}, \mathbf{l}_r^{l_i}, \mathbf{l}_r^{r_i} \notin \mathbb{C}$ **do**
        if $F(\mathbf{l}_l^{l_i}, \mathbf{l}_l^{r_i}, \mathbf{l}_r^{l_i}, \mathbf{l}_r^{r_i}) < 10$ **then**
            $\mathbb{C} = \mathbb{C} \cup \{\mathbf{l}_l^{l_i}, \mathbf{l}_l^{r_i}, \mathbf{l}_r^{l_i}, \mathbf{l}_r^{r_i}\}$
        end if
    end for

    Step5: **if** crad($\mathbb{C}$)> $N/2$ **then**
        Utilizing the all the boundaries of the set $\mathbb{C}$ to minimize the objective function (Eq. 4.7) by the interior algorithm [83]. The estimated parameters and minimization error are $\mathbf{M}_c$ and E_c.
    end if
    if $E_c < E_b$ **then**
        $E_b = E_c$
        $\mathbf{M}_b = \mathbf{M}_c$
    end if
    iter=iter+1
end while
return $\mathbf{M}_b, E_b$;

4.3.4 Solution

Initialization

In actual, as stated in Sect. 4.3.1, the road model and viewpoints are located in their own local coordinates. Before estimating the attributes of the road, the viewpoints and central lines of the road should be registered in the same coordinate. Here, the "Point-to-Point" based ICP (Iterative Closest Points) [16, 8] method is employed to match the central lines of the road and the viewpoints with the transformation matrix $\mathbf{R}_m$, $\mathbf{t}_m$. This transformation guarantees the central lines of the road is near to the viewpoints, which is better for acquiring the global minimization. Besides, for the initialization, the rotation axis $\mathbf{v}_m$, angle θ, are the z axis, 0 radian; the translation $\mathbf{t}_m$ is zero vector; the width w of the road is 1.5 meter.

Optimization

As stated in Sect. 4.1, due to the illumination, shadow, etc., the road boundaries extracted from the video frames are not accurate all the time. The inaccurate road boundaries will affect the parameter estimation when iteratively optimizing the objective function. Therefore, the RANSAC (Random Sample and Consensus) [81] method is exploited to eliminate outliers of the road boundaries during optimization. Besides, the interior algorithm [83] is employed for interactive updating. To avoid falling into the local optimum, we acquire multiple initial values in the scope of the constraints using scatter-search mechanism [82]. The details of the optimization process are shown in the Algorithm 3.

4.3.5 Attribute Estimation

According to Sect. 4.1, the road is constructed by estimating the parameters (i.e. rotation $\mathbf{R}_m$, translation $\mathbf{t}_m$ and road structure). After optimizing the objective function, the rotated vector $\mathbf{v}_m$, rotation angle θ, translation vector $\mathbf{t}_m$ and road width w have been determined. The road attributes should be calculated based on the estimated parameters except the translation vector $\mathbf{t}_m$. Concretely, the rotation axis $\mathbf{v}_m$ and angle θ are utilized to calculate the rotation matrix $\mathbf{R}_m$ by Rodrigues equation [19]. In terms of the wire-frame road structure, the points of the road boundaries are estimated by the central points and

road width w. The road surface consists of series of triangle patches, which are generated based on the boundary points. The details are presented in the paragraph 4.3.2. Through this way, any point of the road surface can be interpolated by the vertexes of the nearby triangle meshes.

4.4 Experiments

In this section, both the quantitative and qualitative experiments are conducted to evaluate the proposed method. In the quantitative experiment, since the proposed method is to estimate the attributes of the roads, we evaluate the proposed method through two aspects: the poses (position and orientation) of the road and road width. In the qualitative experiments, the reconstructed wire-frame model is put into the Google Earth for display. Before these, the experiment preparation including dataset, metric etc. is introduced first.

4.4.1 Experiment Preparation

Parameter Setting

As stated in the paragraph 4.3.2, the deeplab v3plus [15] network is employed for the road detection and boundary extraction. The road detection task of KITTI [32] is exploited for training. Before training, all the input images are resized as 375×621. The deeplab v3 plus is utilized for the road detection after training 400 epochs.

Dataset

The raw KITTI [31] is exploited and processed to evaluate the proposed method. We select four groups (i.e. city_0926_96, city_0926_14, road_0926_15, road_0926_27) of data from KITTI. The details are shown in Table 4.1. Each group data contains two image sequences from the binocular color cameras, the intrinsic matrices and baseline of the binocular cameras.

Metric

As for the poses of the road, the performance of our method is evaluated in the 2D space. The reason is that the projected region of the road model by the accurate poses is the

Table 4.1: The introduction of the dataset

Category	Road		City	
Number	0926_15	0926_27	0926_96	0926_14
Frame	240	188	240	238
Description	Curved	Straight	Curved	Curved
Content	two image sequences, intrinsic matrix, Baseline			

same as the region of road in image. Thus, the evaluation is converted to compare the projected region of the 3D road model and the region of the road from each image. Like [80], we utilize the intersection over union (IoU) between the reprojected region and the ground truth of the road region, which is manually labeled in each frame, to judge whether the pose of the road is accurate or not.

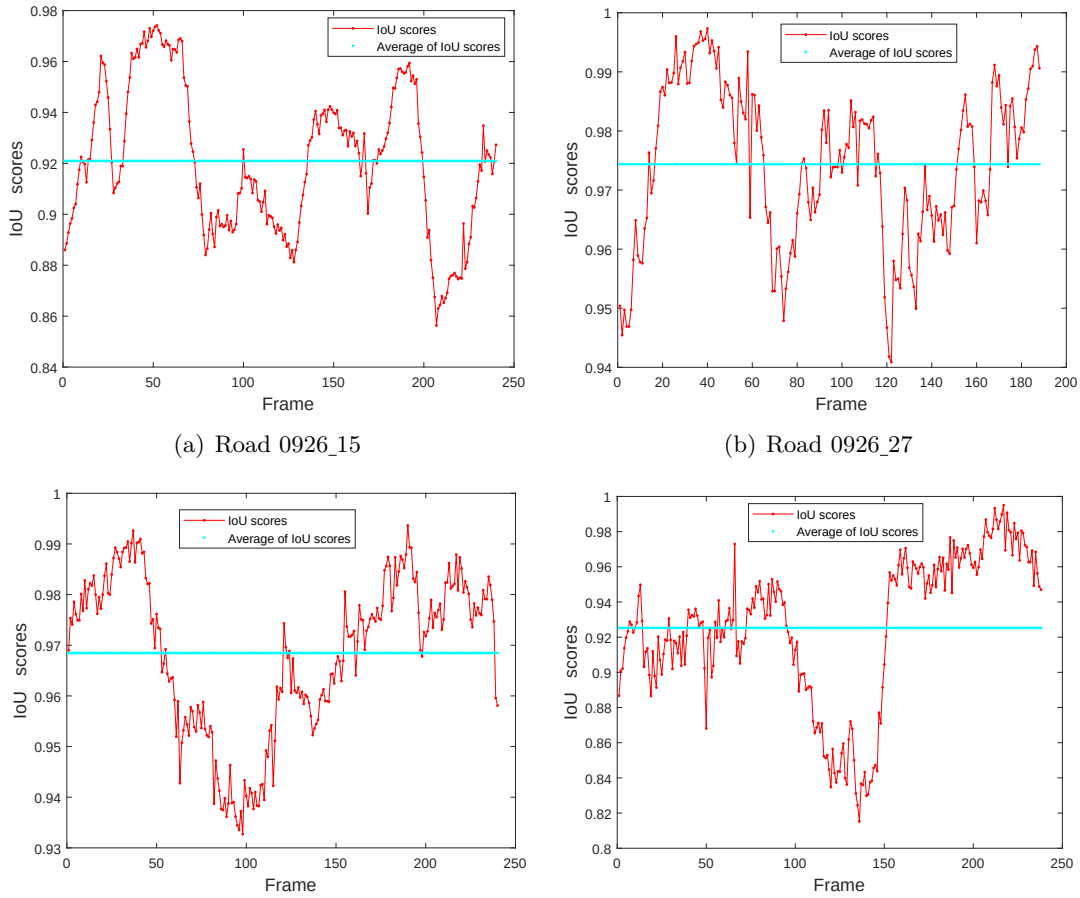
In terms of the road width, the ground truth is acquired from the GIS information. Concretely, we utilize the measurement tools in the "Google Earth" to measure a road width for many times. Then the ground truth of a road is the average value of the measurements. Finally, the relative error is exploited to evaluate the accuracy of the estimated road width.

4.4.2 Quantitative Experiments

In this paragraph, we conduct the quantitative experiments to evaluate the proposed method in two aspects. First, after acquiring attributes of the road model, we will project the model into each frame of the traffic video to verify the accuracy of the position and orientation. Second, the estimated road width is evaluated with the ground truth from Google Earth.

Road Pose

The IoU scores of each frame in four traffic videos is shown in Fig. 4.4, where the red line indicates the IoU tendency with the frame; the blue lines indicate the average IoU score of each video. The IoU scores approximately range from 0.85 and 0.97, from 0.94 and 0.99, from 0.93 and 0.99 and from 0.81 and 0.99 in Fig.4.4(a),(b),(c),(d) respectively. The average IoU scores of each sequence are 0.921, 0.975, 0.968, 0.928 respectively. As a



(a) Road 0926_15

(b) Road 0926_27

(c) City 0926_96

(d) City 0926_14

Figure 4.4: the IoU scores of each frames in four traffic videos

result, the projected region of the reconstructed road are basically consistent with the road regions of image sequence, which indicates the proposed method has good performance in both straight and curved road.

The statistics of four traffic videos are summarized in Table 4.2, with percentage of frames achieving predefined IoU scores. The IoU scores of "Road 0926_27" and "City 0926_14" have the better performances. Besides, the percentages of traffic frames, whose IoU scores exceed 90%, are more than approximate 70% in the all traffic videos. Besides, the performance in "Road0926_27" obviously outperforms others threes, which indicates not only the geometry structure of straight road ("Road0926_27") is simpler than those of curved roads, but also the proposed method has better performance in straight road.

Table 4.2: The statistical IoU scores of each video

Traffic Video	IoU scores				
	87%	>90%	>94%	>97%	Mean
Road0926_15	96.67%	72.08%	26.25%	2.92%	92.09%
Road0926_27	100%	100%	100%	61.17%	97.44%
City0926_96	85.71%	76.47%	43.28%	12.61%	96.85%
City0926_14	100%	100%	92.50%	54.58%	92.53%

Road Width

The results of the road width experiments are shown in Table 4.3, where the first, second and third columns are estimated width, ground truth and relative error respectively. The relative error of the road width is 5.7%, 2.1%, 4.2% and 4.9%, which basically accords with the reality. Meanwhile, the estimated width of the straight road is more accurate than those of curved ones.

Table 4.3: The relative error of road width

	Width (m)	Ground Truth (m)	Relative Error
Road0926_0015	6.41	6.8	5.70%
Road0926_0027	5.48	5.6	2.10%
City0926_0096	8.4	8.06	4.20%
City0926_0014	7.34	7	4.90%

(a) City0926_0014

(b) City0926_0096

Figure 4.5: The display of the road model in the Google Earth software

4.4.3 Qualitative Experiments

In the qualitative experiments, we will display the reconstructed road model in the Google Earth. After estimating the attributes and acquiring the wire-frame model of the road, the points on the road model are converted to the longitude, latitude, and altitude in the WGS-84 (World Geodetic System 1984) Coordinate. As is shown in the Fig. 4.5, the roads from the traffic videos "City0926_96" and "City0926_14" are reconstructed and displayed as the red meshes in the Google Earth. Obviously, the reconstructed roads overlap and consist with the roads of the Google Earth in the aspects of the position, orientation and structure.

4.5 Summary

A novel road reconstruction method from traffic videos is proposed in this chapter. The road construction is realized by estimating the attributes (i.e. position, orientation and structure). In this method, the central lines of a road is imported from GIS information. Subsequently, the 3D parametric road boundaries and 2D detected road boundaries from images establish feature correspondences. Besides, the relative poses of each image frame is estimated by SLAM-based method. Then, the parameter is determined by objective function, which combines feature correspondences and relative poses. Finally, the road attributes are computed by the parameters. The experiments are conducted on the public KITTI dataset, which prove the effectiveness of the proposed method. The contributions of this chapter are shown as follows:

1) The vision-based methods mentioned above rely on the road detection results, which makes a big difference to the reconstructed road. However, due to the factors of strong illumination, shadow and various object occlusion on the road surface, the road detection algorithms are hardly always effective. Hence, we import the central line of a road from GIS as a constraint to reduce the effective of the false road detection of some frames.

2)In the stage of the road reconstruction, different from the methods those execute frame by frame, the proposed method designs a new objective function to estimate the road attributes. This objective function utilizes all image frames to globally optimize the estimated parameters, which can reduce the accumulative errors.

Chapter 5

Conclusion and Future Works

5.1 Conclusion

In this thesis, aiming to the demands of unmanned vehicle technologies, some related methods are designed for perusing more accurate pose estimation of traffic objects. Due to the limitation of time and length of the thesis, the two kinds traffic objects (i.e. traffic signs and roads) are mainly taken into consideration for pose estimation. In order to overcome the influences of noises by sensors and vehicle's motion, the specific methods are proposed for corresponding objects utilizing the its shape and structure information. In experiments, the comparative results on the public available dataset of real traffic scenarios validate the superiority of the proposed methods. The concrete summaries and contributions are displayed as follows:

(1) The method for pose estimation of traffic signs is put forward firstly. The homography constraint utilized in the method assumes that the traffic signs captured by various viewpoints are stable and localized in the 3D plane. The feature correspondences are established and shifted through a series of processing. The objective function is established to combine the shifted feature correspondences and traffic sign plane for optimization. Meanwhile, in order to test the proposed method, the KITTI$^+$ dataset is proposed after space-time synchronization. The comparative experiments with the state-of-the-art methods have been conducted on the KITTI$^+$ and BelgiumTS datasets. The quantitative results prove that the proposed method has better localization performance than others do.

(2) The pose estimation of a traffic road problem has been investigated and converted into the problem of non-perspective pose estimation. Thus, for the traffic road, the two-stage coarse-to-fine registration method is proposed to estimate the road pose from line correspondences. Firstly, the ICP method is employed for estimating the road pose coarsely. Secondly, the objective function is designed to combine the point-to-line correspondences for refining the coarse pose. Besides, we propose a convenient and free way to acquire the wire-frame road from GIS information, which can be applied in this task. The quantitative and qualitative experiments are conducted to validate the proposed method. Besides, in these experiments, the state-of-the-art methods are re-implemented for comparison. The results prove that our method estimates the road pose more accurate than other state-of-the-art methods do.

(3) Since the way of acquiring the wire-frame road model is relatively time-consuming, we propose a new framework to acquire the structure and estimate the pose of a road simultaneously. The main improvement is that the proposed method only need the central line of a road rather than the whole wire-frame model. To begin with, the road boundaries are parameterized by the central line of the road. The point-to-line correspondences are established with the 3D points of parameterized boundaries and the 2D lines on the images. Then, the objective function utilizes the correspondences to estimate the pose and structure of the road. The experimental results prove the effectiveness of the proposed method.

5.2 Future Works

As mentioned in the introduction, the traffic objects include vehicles, pedestrians, cyclists except for traffic signs and roads. Due to the limitation of time, the pose estimation of some important traffic objects (vehicles, pedestrians, cyclists) have been un-investigated in this thesis. Distinguished from the traffic road and signs, some traffic objects (such as pedestrians, cyclists) are dynamically moving with time in traffic scenarios. Thus, the pose estimation of dynamical traffic objects is needed for investigation. Since the pose of moving objects is changing with time, the task of dynamically moving objects pose estimation is more hard than the static objects. For the moving objects, the pose of each

frame need to be estimated. Thus, the way of utilizing the space-time continuity and the constraint of specific object still need to be investigated.

Bibliography

[1] S. Agarwal, N. Snavely, I. Simon, S. M. Seitz, and R. Szeliski. Building rome in a day. In *IEEE International Conference on Computer Vision*, 2009.

[2] H. A. Alhaija, S. K. Mustikovela, L. Mescheder, A. Geiger, and C. Rother. Augmented reality meets computer vision: Efficient data generation for urban driving scenes. *International Journal of Computer Vision*, (2):1–12, 2017.

[3] AstaZero. Astazero: Active safety test area(sp sveriges tekniska forskningsinstitut).

[4] V. Balali, M. Golparvar-Fard, and J. M. de la Garza. Video-based highway asset recognition and 3d localization. *Computing in civil engineering*, 386, 2013.

[5] S. Y. Bao, M. Bagra, Y.-W. Chao, and S. Savarese. Semantic structure from motion with points, regions, and objects. In *2012 IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, pages 2703–2710. IEEE, 2012.

[6] Y. Bar-Shalom, X. Li, and T. Kirubarajan. *Estimation with Applications to Tracking and Navigation: Theory, Algorithms and Software*. Wiley, 2001.

[7] H. Bay, T. Tuytelaars, and L. V. Gool. SURF: speeded up robust features. In *Computer Vision - ECCV 2006, 9th European Conference on Computer Vision, Graz, Austria, May 7-13, 2006, Proceedings, Part I*, pages 404–417, 2006.

[8] P. J. Besl, N. D. McKay, et al. A method for registration of 3-d shapes. *IEEE Transactions on pattern analysis and machine intelligence*, 14(2):239–256, 1992.

[9] C. M. Bishop. Pattern recognition. *Machine Learning*, 128, 2006.

[10] M. Buczko, V. Willert, J. Schwehr, and J. Adamy. Self-validation for automotive visual odometry. In *2018 IEEE Intelligent Vehicles Symposium, IV 2018, Changshu, Suzhou, China, June 26-30, 2018*, pages 1–6, 2018.

[11] R. H. Byrd, J. C. Gilbert, and J. Nocedal. A trust region method based on interior point techniques for nonlinear programming. *Math. Program.*, 89(1):149–185, 2000.

[12] A. Chatterjee and V. M. Govindu. Efficient and robust large-scale rotation averaging. *2013 IEEE International Conference on Computer Vision*, pages 521–528, 2013.

[13] D. Chen and X. He. Fast automatic three-dimensional road model reconstruction based on mobile laser scanning system. *Optik - International Journal for Light and Electron Optics*, 126(7-8):725–730, 2015.

[14] H. H. Chen. Pose determination from line-to-plane correspondences: Existence condition and closed-form solutions. *IEEE Trans. Pattern Anal. Mach. Intell.*, 13(6):530–541, 1991.

[15] L.-C. Chen, Y. Zhu, G. Papandreou, F. Schroff, and H. Adam. Encoder-decoder with atrous separable convolution for semantic image segmentation. In *ECCV*, 2018.

[16] Y. Chen and G. Medioni. Object modelling by registration of multiple range images. *Image and vision computing*, 10(3):145–155, 1992.

[17] P. Cignoni, M. Callieri, M. Corsini, M. Dellepiane, F. Ganovelli, and G. Ranzuglia. MeshLab: an Open-Source Mesh Processing Tool. In V. Scarano, R. D. Chiara, and U. Erra, editors, *Eurographics Italian Chapter Conference*. The Eurographics Association, 2008.

[18] T. F. Coleman and Y. Li. On the convergence of reflective newton methods for large-scale nonlinear minimization subject to bounds. *Mathematical programming*, 67:189–224, 1994.

[19] W. contributors. Rotation matrix — wikipedia, the free encyclopedia, 2017. [Online; accessed 17-January-2018].

[20] M. Cordts, M. Omran, S. Ramos, T. Rehfeld, M. Enzweiler, R. Benenson, U. Franke, S. Roth, and B. Schiele. The cityscapes dataset for semantic urban scene understanding. In *Proc. of the IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, 2016.

[21] M. Cordts, M. Omran, S. Ramos, T. Scharwächter, M. Enzweiler, R. Benenson, U. Franke, S. Roth, and B. Schiele. The cityscapes dataset. In *CVPR Workshop on The Future of Datasets in Vision*, 2015.

[22] H. Cui, G. Xiang, S. Shen, and Z. Hu. Hsfm: Hybrid structure-from-motion. In *IEEE Conference on Computer Vision & Pattern Recognition*, 2017.

[23] M. Dhome, M. Richetin, J. Lapresté, and G. Rives. Determination of the attitude of 3d objects from a single perspective view. *IEEE Trans. Pattern Anal. Mach. Intell.*, 11(12):1265–1278, 1989.

[24] N. D. Doulamis. Coupled multi-object tracking and labeling for vehicle trajectory estimation and matching. *Multimedia Tools Appl.*, 50(1):173–198, 2010.

[25] S. Drake. Converting gps coordinates to navigation coordinates (enu), 2002.

[26] R. Fan, X. Ai, and N. Dahnoun. Road surface 3d reconstruction based on dense subpixel disparity map estimation. *IEEE Transactions on Image Processing*, 27(6):3025–3035, 2018.

[27] R. Fan, Y. Liu, X. Yang, M. J. Bocus, N. Dahnoun, and S. Tancock. Real-time stereo vision for road surface 3-d reconstruction. In *2018 IEEE International Conference on Imaging Systems and Techniques (IST)*, pages 1–6. IEEE, 2018.

[28] L. A. Fernandes and M. M. Oliveira. Real-time line detection through an improved hough transform voting scheme. *Pattern Recognition*, 41(1):299–314, 2008.

[29] A. Geiger, P. Lenz, C. Stiller, and R. Urtasun. Vision meets robotics: The kitti dataset. *International Journal of Robotics Research (IJRR)*, 2013.

[30] A. Geiger, P. Lenz, C. Stiller, and R. Urtasun. Vision meets robotics: The kitti dataset. *The International Journal of Robotics Research*, 32(11):1231–1237, 2013.

[31] A. Geiger, P. Lenz, C. Stiller, and R. Urtasun. Vision meets robotics: The KITTI dataset. *I. J. Robotics Res.*, 32(11):1231–1237, 2013.

[32] A. Geiger, P. Lenz, and R. Urtasun. Are we ready for autonomous driving? the KITTI vision benchmark suite. In *2012 IEEE Conference on Computer Vision and Pattern Recognition, Providence, RI, USA, June 16-21, 2012*, pages 3354–3361, 2012.

[33] A. Guiducci. *Parametric Model of the Perspective Projection of a Road with Applications to Lane Keeping and 3D Road Reconstruction*. 1999.

[34] A. Harltey and A. Zisserman. *Multiple view geometry in computer vision (2. ed.)*. Cambridge University Press, 2006.

[35] L. Hazelhoff, I. Creusen, and P. H. de With. Robust detection, classification and positioning of traffic signs from street-level panoramic images for inventory purposes. In *2012 IEEE Workshop on Applications of Computer Vision (WACV)*, pages 313–320. IEEE, 2012.

[36] L. Hazelhoff, I. M. Creusen, et al. Exploiting street-level panoramic images for large-scale automated surveying of traffic signs. *Machine Vision and Applications*, 25(7):1893–1911, 2014.

[37] K. He, X. Zhang, S. Ren, and J. Sun. Deep residual learning for image recognition. In *2016 IEEE Conference on Computer Vision and Pattern Recognition, CVPR 2016, Las Vegas, NV, USA, June 27-30, 2016*, pages 770–778, 2016.

[38] N. Horanyi and Z. Kato. Multiview absolute pose using 3d - 2d perspective line correspondences and vertical direction. In *2017 IEEE International Conference on Computer Vision Workshops, ICCV Workshops 2017, Venice, Italy, October 22-29, 2017*, pages 2472–2480, 2017.

[39] Z. Hu. Intelligent road sign inventory (irsi) with image recognition and attribute computation from video log. *Computer-Aided Civil and Infrastructure Engineering*, 28(2):130–145, 2013.

[40] Z. Hu and Y. J. Tsai. Homography-based vision algorithm for traffic sign attribute computation. *Computer-Aided Civil and Infrastructure Engineering*, 24(6):385–400, 2009.

[41] X. Huang, X. Cheng, Q. Geng, B. Cao, D. Zhou, W. Peng, Y. Lin, and R. Yang. The apolloscape dataset for autonomous driving. 2018.

[42] Z. Ji and S. Singh. Visual-lidar odometry and mapping: Low-drift, robust, and fast. In *IEEE International Conference on Robotics & Automation*, 2015.

[43] J. Jiao. Machine learning assisted high-definition map creation. In *2018 IEEE 42nd Annual Computer Software and Applications Conference (COMPSAC)*, volume 01, pages 367–373, July 2018.

[44] E. Krsák and S. Toth. Traffic sign recognition and localization for databases of traffic signs. *Acta Electrotechnica et Informatica*, 11(4):31, 2011.

[45] L. Lecrosnier, R. Boutteau, P. Vasseur, X. Savatier, and F. Fraundorfer. Vision based vehicle relocalization in 3d line-feature map using perspective-n-line with a known vertical direction. In *2019 IEEE Intelligent Transportation Systems Conference (ITSC)*, pages 1263–1269, Oct 2019.

[46] D.-T. Lee and B. J. Schachter. Two algorithms for constructing a delaunay triangulation. *International Journal of Computer & Information Sciences*, 9(3):219–242, 1980.

[47] G. H. Lee. A minimal solution for non-perspective pose estimation from line correspondences. In *Computer Vision - ECCV 2016 - 14th European Conference, Amsterdam, The Netherlands, October 11-14, 2016, Proceedings, Part V*, pages 170–185, 2016.

[48] S. Leutenegger, M. Chli, and R. Siegwart. BRISK: binary robust invariant scalable keypoints. In *IEEE International Conference on Computer Vision, ICCV 2011, Barcelona, Spain, November 6-13, 2011*, pages 2548–2555, 2011.

[49] W. Li, C. Pan, R. Zhang, J. Ren, and R. Yang. Aads: Augmented autonomous driving simulation using data-driven algorithms. 2019.

[50] Y. Li, Y. Liu, Y. Su, G. Hua, and N. Zheng. Three-dimensional traffic scenes simulation from road image sequences. *IEEE Transactions on Intelligent Transportation Systems*, 17(4):1121–1134, April 2016.

[51] Y. Li, Y. Liu, C. Zhang, D. Zhao, and N. Zheng. The "floor-wall" traffic scenes construction for unmanned vehicle simulation evaluation. In *Intelligent Transportation Systems (ITSC), 2014 IEEE 17th International Conference on*, pages 1726–1731. IEEE, 2014.

[52] Y. Li, Y. Liu, C. Zhang, D. Zhao, and N. Zheng. The "floor-wall" traffic scenes construction for unmanned vehicle simulation evaluation. In *17th International IEEE Conference on Intelligent Transportation Systems (ITSC)*, pages 1726–1731, Oct 2014.

[53] Y. Liu, Z. Xie, and H. Liu. LB-LSD: A length-based line segment detector for real-time applications. *Pattern Recognition Letters*, 128:247–254, 2019.

[54] D. G. Lowe. Distinctive image features from scale-invariant keypoints. *International Journal of Computer Vision*, 60(2):91–110, 2004.

[55] W. Maddern, G. Pascoe, C. Linegar, and P. Newman. 1 Year, 1000km: The Oxford RobotCar Dataset. *The International Journal of Robotics Research (IJRR)*, 36(1):3–15, 2017.

[56] É. Marchand, H. Uchiyama, and F. Spindler. Pose estimation for augmented reality: A hands-on survey. *IEEE Trans. Vis. Comput. Graph.*, 22(12):2633–2651, 2016.

[57] Matrix-Style. Google's testing self-driving cars in a matrix-style simulation.

[58] P. Miraldo, T. J. Dias, and S. Ramalingam. A minimal closed-form solution for multi-perspective pose estimation using points and lines. In *Computer Vision - ECCV 2018 - 15th European Conference, Munich, Germany, September 8-14, 2018, Proceedings, Part XVI*, pages 490–507, 2018.

[59] F. M. Mirzaei and S. I. Roumeliotis. Globally optimal pose estimation from line correspondences. In *IEEE International Conference on Robotics and Automation, ICRA 2011, Shanghai, China, 9-13 May 2011*, pages 5581–5588, 2011.

[60] A. Mogelmose, M. M. Trivedi, and T. B. Moeslund. Vision-based traffic sign detection and analysis for intelligent driver assistance systems: Perspectives and survey. *IEEE Transactions on Intelligent Transportation Systems*, 13(4):1484–1497, 2012.

[61] F. Moreno-Noguer, V. Lepetit, and P. Fua. Accurate non-iterative o(n) solution to the pnp problem. In *IEEE 11th International Conference on Computer Vision, ICCV 2007, Rio de Janeiro, Brazil, October 14-20, 2007*, pages 1–8, 2007.

[62] M. G. Mozerov and V. D. W. Joost. Accurate stereo matching by two-step energy minimization. *IEEE Transactions on Image Processing*, 24(3):1153–1163, 2015.

[63] R. Mur-Artal and J. D. Tardós. ORB-SLAM2: an open-source SLAM system for monocular, stereo, and RGB-D cameras. *IEEE Trans. Robotics*, 33(5):1255–1262, 2017.

[64] M. J. Powell. A fast algorithm for nonlinearly constrained optimization calculations. *Numerical analysis*, 1978.

[65] PreScan. Simulation of adas and active safety.

[66] B. Pribyl, P. Zemcík, and M. Cadík. Camera pose estimation from lines using plücker coordinates. In *Proceedings of the British Machine Vision Conference 2015, BMVC 2015, Swansea, UK, September 7-10, 2015*, pages 45.1–45.12, 2015.

[67] B. Pribyl, P. Zemcík, and M. Cadík. Absolute pose estimation from line correspondences using direct linear transformation. *Computer Vision and Image Understanding*, 161:130–144, 2017.

[68] V. A. Prisacariu and I. D. Reid. Pwp3d: Real-time segmentation and tracking of 3d objects. *International journal of computer vision*, 98(3):335–354, 2012.

[69] V. A. Prisacariu, R. Timofte, K. Zimmermann, I. Reid, and L. Van Gool. Integrating object detection with 3d tracking towards a better driver assistance system. In *2010 20th International Conference on Pattern Recognition (ICPR)*, pages 3344–3347. IEEE, 2010.

[70] E. Rublee, V. Rabaud, K. Konolige, and G. Bradski. Orb: An efficient alternative to sift or surf. In *Computer Vision (ICCV), 2011 IEEE international conference on*, pages 2564–2571. IEEE, 2011.

[71] A. Ruta, F. Porikli, S. Watanabe, and Y. Li. In-vehicle camera traffic sign detection and recognition. *Machine Vision and Applications*, 22(2):359–375, 2011.

[72] S. Savarese and L. Fei-Fei. 3d generic object categorization, localization and pose estimation. In *2007 IEEE 11th International Conference on Computer Vision*, pages 1–8. IEEE, 2007.

[73] D. Schlegel, M. Colosi, and G. Grisetti. Proslam: Graph SLAM from a programmer's perspective. In *2018 IEEE International Conference on Robotics and Automation, ICRA 2018, Brisbane, Australia, May 21-25, 2018*, pages 1–9, 2018.

[74] J. L. Schönberger and J. M. Frahm. Structure-from-motion revisited. In *IEEE Conference on Computer Vision & Pattern Recognition*, 2016.

[75] G. Seetharaman, A. Lakhotia, and E. P. Blasch. Unmanned vehicles come of age: The darpa grand challenge. *Computer*, 39(12):26–29, 2007.

[76] B. Soheilian, N. Paparoditis, and B. Vallet. Detection and 3d reconstruction of traffic signs from multiple view color images. *ISPRS Journal of Photogrammetry and Remote Sensing*, 77:1–20, 2013.

[77] S. Song and M. Chandraker. Joint sfm and detection cues for monocular 3d localization in road scenes. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, pages 3734–3742, 2015.

[78] C. Sweeney, T. Sattler, M. Turk, T. Hoellerer, and M. Pollefeys. Optimizing the viewing graph for structure-from-motion. In *IEEE International Conference on Computer Vision*, 2015.

[79] R. Timofte, K. Zimmermann, and L. Van Gool. Multi-view traffic sign detection, recognition, and 3d localisation. *Machine Vision and Applications*, 25(3):633–647, 2014.

[80] H. Tjaden, U. Schwanecke, and E. Schömer. Real-time monocular pose estimation of 3d objects using temporally consistent local color histograms. In *IEEE International Conference on Computer Vision, ICCV 2017, Venice, Italy, October 22-29, 2017*, pages 124–132, 2017.

[81] P. H. S. Torr and A. Zisserman. MLESAC: A new robust estimator with application to estimating image geometry. *Computer Vision and Image Understanding*, 78(1):138–156, 2000.

[82] Z. Ugray, L. S. Lasdon, J. C. Plummer, F. W. Glover, J. P. Kelly, and R. Martí. Scatter search and local NLP solvers: A multistart framework for global optimization. *INFORMS Journal on Computing*, 19(3):328–340, 2007.

[83] R. A. Waltz, J. L. Morales, J. Nocedal, and D. Orban. An interior algorithm for nonlinear optimization that combines line search and trust region steps. *Math. Program.*, 107(3):391–408, 2006.

[84] K. C. Wang, Z. Hou, and W. Gong. Automated road sign inventory system based on stereo vision and tracking. *Computer-Aided Civil and Infrastructure Engineering*, 25(6):468–477, 2010.

[85] Q. Wang, M. Chen, F. Nie, and X. Li. Detecting coherent groups in crowd scenes by multiview clustering. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, pages 1–1, 2018.

[86] Q. Wang, Z. Qin, F. Nie, and X. Li. Spectral embedded adaptive neighbors clustering. *IEEE Transactions on Neural Networks and Learning Systems*, 30(4):1265–1271, April 2019.

[87] Q. Wang, J. Wan, and X. Li. Robust hierarchical deep learning for vehicular management. *IEEE Transactions on Vehicular Technology*, 68(5):4148–4156, May 2019.

[88] Q. Wang, J. Wan, F. Nie, B. Liu, C. Yan, and X. Li. Hierarchical feature selection for random projection. *IEEE Transactions on Neural Networks and Learning Systems*, 30(5):1581–1586, May 2019.

[89] X. Wang, J. Wen, Z. Nan, J. Shi, L. Xu, and N. Zheng. A general and practical path planning framework for autonomous vehicles. In *2016 IEEE 19th International Conference on Intelligent Transportation Systems (ITSC)*, pages 1397–1402, Nov 2016.

[90] Y. Wang. Real-time moving vehicle detection with cast shadow removal in video based on conditional random field. *IEEE Trans. Circuits Syst. Video Techn.*, 19(3):437–441, 2009.

[91] A. Welzel, A. Auerswald, and G. Wanielik. Accurate camera-based traffic sign localization. In *17th International IEEE Conference on Intelligent Transportation Systems (ITSC)*, pages 445–450. IEEE, 2014.

[92] Wikipedia. Rotation matrix — wikipedia, the free encyclopedia, 2017. [Online; accessed 10-August-2017].

[93] K. Wilson and N. Snavely. Robust global translations with 1dsfm. In *European Conference on Computer Vision*, 2014.

[94] W. Xiao, L. Xu, H. Sun, J. Xin, and N. Zheng. On-road vehicle detection and tracking using mmw radar and monovision fusion. *IEEE Transactions on Intelligent Transportation Systems*, 17(7):2075–2084, 2016.

[95] C. Xu, L. Zhang, L. Cheng, and R. Koch. Pose estimation from line correspondences: A complete analysis and a series of solutions. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 39(6):1209–1222, June 2017.

[96] C. Xu, L. Zhang, L. Cheng, and R. Koch. Pose estimation from line correspondences: A complete analysis and a series of solutions. *IEEE Trans. Pattern Anal. Mach. Intell.*, 39(6):1209–1222, 2017.

[97] Young-Kee Jung and Yo-Sung Ho. Traffic parameter extraction using video-based vehicle tracking. In *Proceedings 199 IEEE/IEEJ/JSAI International Conference on Intelligent Transportation Systems (Cat. No.99TH8383)*, pages 764–769, Oct 1999.

[98] J. Yuan, Y. Li, H. Pan, Z. Cui, and Y. Liu. 3d traffic scenes construction and simulation based on scene stages. In *2018 Chinese Automation Congress (CAC)*, pages 1334–1339, Nov 2018.

[99] M. Zeeshan Zia, M. Stark, and K. Schindler. Are cars just 3d boxes?-jointly estimating the 3d shape of multiple objects. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, pages 3678–3685, 2014.

[100] L. Zhang, C. Xu, K. Lee, and R. Koch. Robust and efficient pose estimation from line correspondences. In *Computer Vision - ACCV 2012 - 11th Asian Conference on Computer Vision, Daejeon, Korea, November 5-9, 2012, Revised Selected Papers, Part III*, pages 217–230, 2012.

[101] X. Zhang, Z. Zhang, Y. Li, X. Zhu, Q. Yu, and J. Ou. Robust camera pose estimation from unknown or known line correspondences. *Applied Optics*, 51(7):936.

[102] D. Zhao, Y. Liu, C. Zhang, and Y. Li. Autonomous driving simulation for unmanned vehicles. In *2015 IEEE Winter Conference on Applications of Computer Vision*, pages 185–190, Jan 2015.

[103] T. Zhou, M. Brown, N. Snavely, and D. G. Lowe. Unsupervised learning of depth and ego-motion from video. 2017.

[104] T. Zhou, C. Sun, and H. Fu. Road information extraction from high-resolution remote sensing images based on road reconstruction. *Remote Sensing*, 11(1):79, 2019.

[105] S. Zhu, T. Shen, L. Zhou, R. Zhang, J. Wang, T. Fang, and L. Quan. Parallel structure from motion from local increment to global averaging. 2017.