

Cooperative Perception of Roadside Unit and Onboard Equipment with Edge Artificial Intelligence for Driving Assistance

USDOT Award No. 69A3551747124

August 2021



Cooperative Perception of Roadside Unit and Onboard Equipment with Edge Artificial Intelligence for Driving Assistance

C2SMART Center is a USDOT Tier 1 University Transportation Center taking on some of today's most pressing urban mobility challenges. Using cities as living laboratories, the center examines transportation problems and field tests novel solutions that draw on unprecedented recent advances in communication and smart technologies. Its research activities are focused on three key areas: Urban Mobility and Connected Citizens; Urban Analytics for Smart Cities; and Resilient, Secure and Smart Transportation Infrastructure.

Some of the key areas C2SMART is focusing on include:

Disruptive Technologies

We are developing innovative solutions that focus on emerging disruptive technologies and their impacts on transportation systems. Our aim is to accelerate technology transfer from the research phase to the real world.

Unconventional Big Data Applications

C2SMART is working to make it possible to safely share data from field tests and non-traditional sensing technologies so that decision-makers can address a wide range of urban mobility problems with the best information available to them.

Impactful Engagement

The center aims to overcome institutional barriers to innovation and hear and meet the needs of city and state stakeholders, including government agencies, policy makers, the private sector, non-profit organizations, and entrepreneurs.

Forward-thinking Training and Development

As an academic institution, we are dedicated to training the workforce of tomorrow to deal with new mobility problems in ways that are not covered in existing transportation curricula.

Led by the New York University Tandon School of Engineering, C2SMART is a consortium of five leading research universities, including Rutgers University, University of Washington, the University of Texas at El Paso, and The City College of New York.

c2smart.engineering.nyu.edu

Yinhai Wang
Principal Investigator
University of Washington
ORC-ID: 0000-0002-4180-5628

Wei Sun
Co-Principal Investigator
University of Washington
ORC-ID: 0000-0002-2191-5352

Chenxi Liu
Student Researcher
University of Washington
ORC-ID: 0000-0002-5447-4768

Zhiyong Cui
Student Researcher
University of Washington
ORC-ID: 0000-0002-5780-4312

Meixin Zhu
Student Researcher
University of Washington
ORC-ID: 0000-0003-3291-3616

Ziyuan Pu
Student Researcher
University of Washington
ORC-ID: 0000-0002-9488-9175

Disclaimer

The contents of this report reflect the views of the authors, who are responsible for the facts and the accuracy of the information presented herein. This document is disseminated in the interest of information exchange. The report is funded, partially or entirely, by a grant from the Federal Transit Administration. However, the U.S. Government assumes no liability for the contents or use thereof.

Executive Summary

Recently, 2D detection in images has made significant progress owing to the emergence of a convolutional neural network (CNN), which can extract high-level features from the images. However, detecting objects in 3D instead of 2D space is an essential topic when building perception systems for autonomous driving. An autonomous vehicle (AV) needs to perceive the objects present in the 3D scene from its sensors to plan its motion safely. Therefore, a series of solutions based on Stereo Cameras, Light Detection, and Ranging (LiDAR) is proposed in existing CAV systems to provide 3D location information to the vehicles. However, all the technologies suffer from a fatal challenge on edge computing: computing resource limitation. As a result, even though powerful, heavy, and expensive on-vehicle computers are equipped with CAVs, some unavoidable calculation errors or delays often happen and result in some severe consequences. To address the challenge and provide more reliable real-time localization services for the CAVs, the team proposed a smart roadside unit (SRSU) for driving or parking assistance with advanced computer vision technologies. The SRSU sensor is a multi-source traffic sensing roadside unit developed by STAR Lab at the UW. It can transmit data through 4G/5G data plan or Long Range (LoRa) and Narrow Band Internet of Things (NB-IoT) data communication protocols. In this research, the smart roadside unit, Mobile Unit for Traffic Sensing (MUST) developed by the research team, is employed for both data analysis and communication with surrounding vehicles.

The key accomplishment of the project can be summarized in follow. The team developed a customized MUST sensor, a cooperative perception method that realizes accurate 3D vehicle localization based on a single image for driving or parking assistance. The customized Computer Vision (CV) algorithm based on Mask-RCNN is implemented on the IoT device for 2D car keypoints detection (X and Y coordinates) based on the image or video data collected by the monocular cameras. Cooperating with the Car Keypoints Prediction (CKP) algorithm, the Depth Detection Algorithm (DDA) developed by the team can extract the third-dimension information (Z coordinates) through the detected car keypoints. Finally, the fusion algorithm integrates the Mask-RCNN Model and DDA results to provide more reliable localization services to all the vehicles in the sensing range. As a result, the SRSU 3D Vehicle Localization System can 1) provide more reliable vehicle localization information; 2) increase calculation efficiency to reduce the information delay; 3) serve more vehicles in a more extensive sensing range.

Table of Contents

Cooperative Perception of Roadside Unit and Onboard Equipment with Edge Artificial Intelligence for Driving Assistance	i
Executive Summary	1
Table of Contents.....	2
List of Figures.....	3
List of Tables	5
Section 1 Introduction	5
Subsection 1.1 Background.....	5
Subsection 1.2 Problem Statement	13
Subsection 1.3 Solution Introduction	14
Section 2 Literature Review	17
Subsection 2.1 Cooperative Perception.....	17
Subsection 2.2 3D Object Detection	21
Subsection 2.3 Datasets for Cooperative Perception Research	25
Section 3 System Design	34
Subsection 3.1 Vehicle Detection and Classification	35
Subsection 3.2 Vehicle Keypoints Detection and Prediction.....	35
Subsection 3.3 Automatic 2D Keypoints to 3D Projection.....	38
Section 4 Prototype Development.....	46
Subsection 4.1 MUST Sensor Prototype Design	46
Subsection 4.2 Data Stream Design	47
Subsection 4.3 Prototype Preliminary Tests.....	48
Section 5 Field Test.....	52
Subsection 5.1 In-lab Tests	52
Subsection 5.2 Field Test Design.....	59
Subsection 5.2 Data Collection and Training	60
Subsection 5.3 Car Keypoints Detection Results	61
Subsection 5.4 Camera Calibration Results	63
Subsection 5.5 Results Validation.....	66
Section 6 Outreach and Technical Transfer	70
Section 7 Conclusion and Discussion	71
Subsection 7.1 Research Conclusion	71
Subsection 7.1 Discussion and Future Work.....	71
References	73

List of Figures

Figure 1-1 A Model of Autonomous Vehicle Taking Advantage of 3D Sensing Technologies Developed by GM Technology	6
Figure 1-2 Latest Research on 3D Sensing Technologies.....	7
Figure 1-3 Point Cloud Generated by Small Lidar Sensor Working with Raspberry Pi	8
Figure 1-4 Point Cloud Generated by Velodyne Lidar Sensor Installed on Autonomous Vehicles..	9
Figure 1-5 Experiment of Distance Measurement based on RSS Technology	11
Figure 1-6 Classic Three Towers Device Localization based on TDOA.....	12
Figure 1-6 Classic Two Towers Localization based on AOA	13
Figure 1-7 MUST Sensor Based Solution Structure	15
Figure 1-8. Illustration of the MUST sensor	16
Figure 2-1: Cooperative Perception Results	18
Figure 2-2: Cooperative Perception with Elevated LiDAR Along the Road	19
Figure 2-3: Cooperative 3D Object Detection System.....	20
Figure 2-4: A review of Lidar and Camera Fusion Algorithm	22
Figure 2-5: Overview of the Mono3D approach (Chen et al., 2016)	24
Figure 2-6: Overview of the state-of-art RTM3D approach (Li et al., 2020)	25
Figure 2-7: Cooperative 3D object detection system proposed by Arnold et al. (2019)	26
Figure 2-8. Five Categories in KITTI data sample.....	30
Figure 2-9. Object detection in KITTI data sample	31
Figure 2-10. Waymo data in diverse environment.....	32
Figure 2-11. Semantic Map Sample in Lyft Dataset.....	33
Figure 3-1 The Structure Design of the System.....	34
Figure 3-2 Fourth-order Hourglass Module Architecture.....	36
Figure 3-3 Residual Module Architecture	37
Figure 3-4 Structure of Graph Neural Network (GNN) for Invisible Car Keypoints Prediction.....	37
Figure 3-5 Structure of 2D to 3D Projection	39
Figure 3-6 Convert between the real-world coordinate system and the camera coordinate system.....	40
Figure 3-7 Convert between the camera coordinate system and the image coordinate system..	41
Figure 3-8 Bias between image coordinate and pixel coordinate.....	42
Figure 4-1 Customized MUST Sensor Design	46
Figure 4-2 Customized MUST Sensor Rendering.....	47
Figure 4-3 Produced Customized MUST sensor	47

Figure 4-4 MUST Prototype System DataStream	48
Figure 4-5 MUST Quality Tests.....	49
Figure 4-6 Sample Practical Test Result in City of Bellevue for Vehicle Detection in Different Conditions (Daylight, Night Vision, Rainy, and Snowy).....	50
Figure 4-7 Comparison of Night Vision between IP camera (left) and USB camera (right) in the same dark room	51
Figure 4-8 Object Detection in Complete Darkness	51
Figure 5-1 In-lab Camera Calibration Tests.....	52
Figure 5-2 In-lab Camera Calibration Coordinates	53
Figure 5-3 Camera Calibration Coordinates in Second In-lab Tests	55
Figure 5-4 FHWA 13 Vehicle Category Classification.....	57
Figure 5-5 Identify seven points and obtain their pixels number for calibration.....	58
Figure 5-6 Same Truck in Another Snapshot.....	59
Figure 5-7 MUST Sensor Installed in Bellevue Test Bed	60
Figure 5-8 Example Results of 3D-SISS on Self-collected Data in Bellevue Test Bed.....	62
Figure 5-9 Accuracy vs α with Various Number of Invisible Car Keypoints	62
Figure 5-10 Camera Calibration Coordinates in Field Test	63
Figure 5-11 Vehicle Dimension with Picked Seven Keypoints.....	64
Figure 5-12 Vehicle Localization Results Validation	68

List of Tables

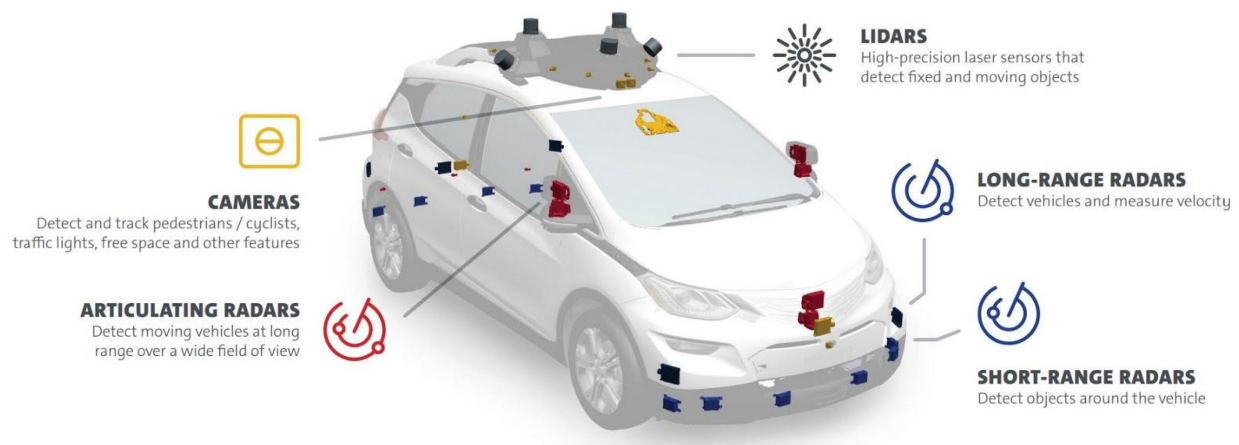
Table 2-1. Self-driving and Object Detection Datasets Overview	29
Table 3-1 MUST sensor quality tests	49
Table 5-1 Camera Calibration Results in First In-lab Tests.....	54
Table 5-2 Camera Calibration Results in Second Lab Tests.....	58
Table 5-3 Sample Results of Camera Calibration	65
Table 5-4 Camera Calibration Results after 100 Runs	65
Table 5-5 Camera Calibration Results Validation with Ground Truth Data	67
Table 5-6 Vehicle Localization Results Validation	69

Section 1 Introduction

Subsection 1.1 Background

Three Dimension (3D) sensing technology has been a hot research topic for decades. In Intelligent Transportation System (ITS) community, 3D sensing technology plays more and more important roles in many applications like Connected and Autonomous Vehicles (CAV) (see Figure 1-1), Mobile Mapping Systems (MMS), Advanced Driving Assistance Systems (ADAS), and Augmented Reality (AR). Compared with traditional 2D traffic sensing technologies, 3D sensing methods can make use of the depth information to estimate more details about the surrounding environment like objects dimension and location. The projection from 2D image coordinate to 3D real world coordinate enables computers to perceive the surrounding environments like human eyes, which can provide more comprehensive, detailed, and accurate information about all the target objects in the scene. As a result, in many

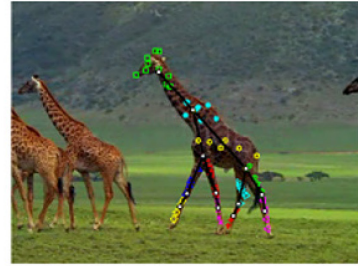
transportation scenarios like driving, monitoring, and planning, 3D sensing algorithms can replace traditional 2D sensing methods or human operators to complete the task pretty good.



However, 3D sensing remains a critical challenge since the lack of depth information. In 2D sensing methods, images can provide rich information on color and texture, however, in 3D sensing, they lack depth information caused by the perspective projection, especially when using a monocular camera. Though it would suffer from the difficulties of estimating 3D locations from images, it is possible to exploit deep learning to get an acceptable result of 3D object detection and localization. Recent years, taking advantage of the development of advanced sensors like stereo camera, radar sensor, and LiDAR sensor, studies related to 3D sensing technology have made great progress in civil engineering, robotics, computer vision, and other related fields based on sensor fusion technologies. Figure 1-2 shows some examples of the latest research on 3D sensing technologies.



[Yan et al., PAMI 2018]



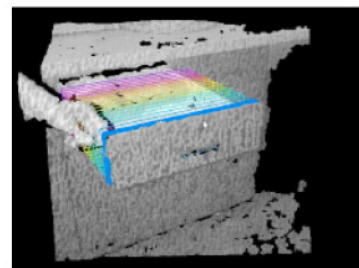
[Ross et al., IJCV 2020]



[Katz et al., ISER 2020]



[Sturm et al., IJCAI 2019, IROS 2020]



[Sturm et al., ICRA 2020]

Existing 3D sensing technologies in ITS can be classified into two categories: local sensing and global sensing (Zhang et al., 2020). 3D local sensing indicates the sensing technologies applied to on-vehicle sensors for surrounding objects or obstacles detection, while 3D global sensing can perceive and monitor the entire traffic scene based on the surveillance system. Recent years, with the development of CAV, more and more attentions are focused on 3D local sensing for driving assistance systems and autonomous driving systems. As a result, most of existing 3D sensing research focused on 3D local sensing technologies based on Lidar, Radar, or other advanced sensors and discusses how to improve its accuracy or efficiency. However, comparing with global sensing, local sensing is much more "shortsighted", because the perception range is limited in a small surrounding area by the field of views of the sensors mounted on the vehicles. Also, in local sensing, the occlusion can be a serious issue because of the limited on-board sensors installation heights. As a result, some potential dangers beyond the detection range cannot be aware by the vehicle computer, which may result in serious car crash. Recent years, global sensing attracts more and more attentions in ITS community for entire traffic scene perception, which can address the two aforementioned limitations of 3D local sensing. More importantly, the potential of 3D global sensing in traffic monitoring and management is exposed through the studies, which can be used as a comprehensive ITS solution for many traditional transportation challenges like vehicle localization, speed measurement, and collision warning. Sensing systems based on 3D global sensing methods can not only cover the functions of most traditional traffic

sensors, but also provide more information like driving direction, collision detection, height detection, and so on. The following paragraphs introduce some popular 3D local and global sensing methods.

Subsection 1.1.1 3D Local Sensing

- **Lidar**

LiDAR stands for “Light Detection and Ranging.” This technology can be mounted to aerial vehicles or roadside units to send laser pulses to the other objects, and once the laser returns to the sensor, the LiDAR system will record data based on the information received. (Hecht et al., 2018) Each data recorded by LiDAR is a 3D point that has its own set of X, Y, and Z coordinates and, in some cases, additional attributes. Point Clouds are a collection of points that represent a 3D shape or feature. Therefore, compared with other detection methods, LiDAR provides reliable depth information that can be used to localize objects accurately and characterize their shapes (Dong et al., 2017) (Behroozpour et al., 2017).

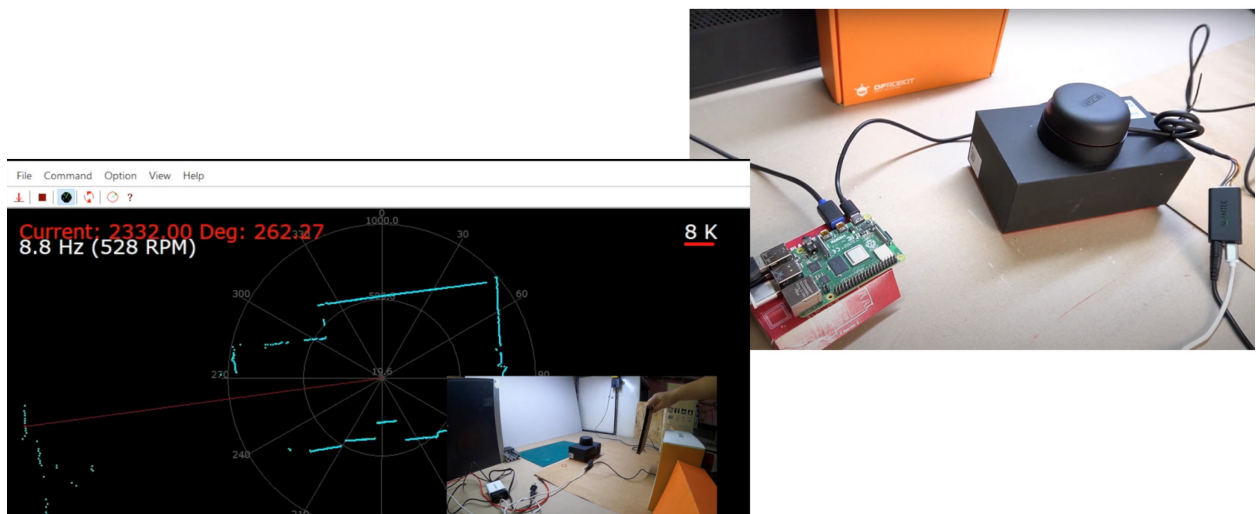


Figure 1-3 Point Cloud Generated by Small Lidar Sensor Working with Raspberry Pi

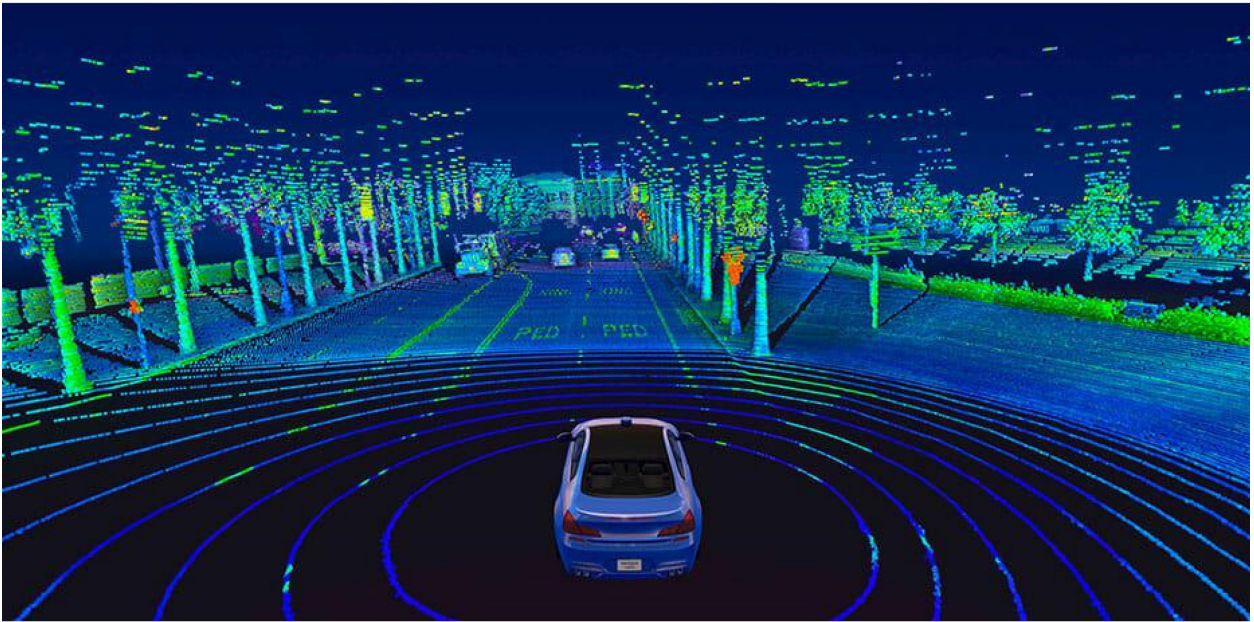


Figure 1-4 Point Cloud Generated by Velodyne Lidar Sensor Installed on Autonomous Vehicles

However, LiDAR, a 3D Sensor that outputs a set of point clouds, gives precise 3D coordinates based on the light rays that reflect on the surroundings, which is vulnerable. Although the accuracy of the LiDAR-based 3D object detections has been hugely improved, LiDAR point clouds suffer from sparse observations at long range and the lack of semantic information. Thus, it is reasonable to integrate camera and LiDAR to yield more robust detection results. Recently, there are promising approaches based on multi-sensor fusion that can provide more accurate 3D object detection results. However, both LiDAR and camera are not reliable in adverse weather conditions, and a fusion of them cannot solve this challenge.

- **Radar**

In general, radar is more reliable in harsh environments, such as rain, fog, and other low visibility conditions. The frequency modulated continuous wave (FMCW) radar works in the millimeter wave (MMW) frequency band (30-300GHz) below the visible light. Thus, the following strengths of MMW are worth mentioning: 1) MMW has a strong ability to penetrate fog, smoke, and dust; 2) The wide bandwidth and higher working frequency endow FMCW radar with excellent detection capability. Significant research in radar object classification has demonstrated its feasibility as a good alternative when cameras fail to provide good results. Despite these good characteristics, it is still very challenging for radars to distinguish different types of objects since it's hard to extract and analyze the semantic information from radio signals. Also, another real challenge mission is to

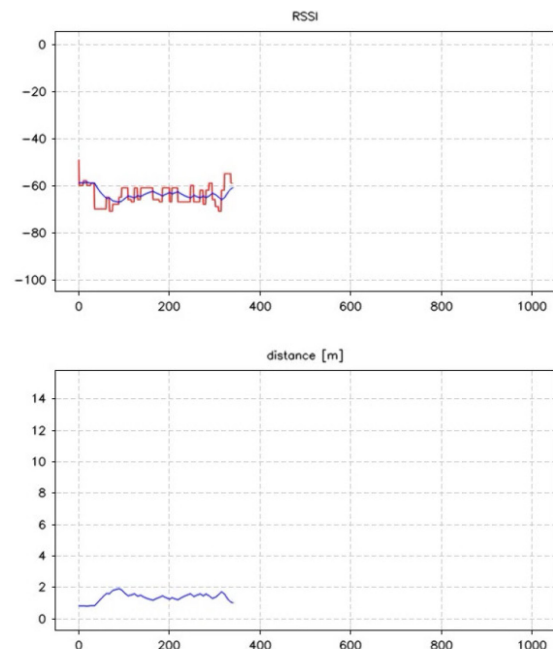
distinguish features of various objects from a radar frame extracted. The surrounding obstacles, such as roadside trees, signs, and even buildings, also can bring in a lot of noise.

Previous research studies show promising and encouraging potential for using radar for object detection. By extracting features manually, (Heuel et al., 2011) use support vector machines (SVM) to classify objects to distinguish cars from pedestrians. Then, (Angelov et al., 2018) uses neural networks to extract features from short-time Fourier transform (STFT) heat maps and evaluates the three categories of cars, pedestrians, and cyclists. However, this method only focuses on classification, which assumes that only one object is properly recognized in the scene and is not suitable for complex autonomous driving scenes. Recently, several methods on radar and image fusion target detection have been proposed in (Gao et al., 2019) (Richards et al., 2005) (Simonyan et al., 2014), which combines the statistical detection algorithm with the neural network classifier. However, such approaches are prone to generate false positives, i.e., obstacles are recognized as target objects. In addition, the laborious human annotations required for this method are often not available given the resource constraint.

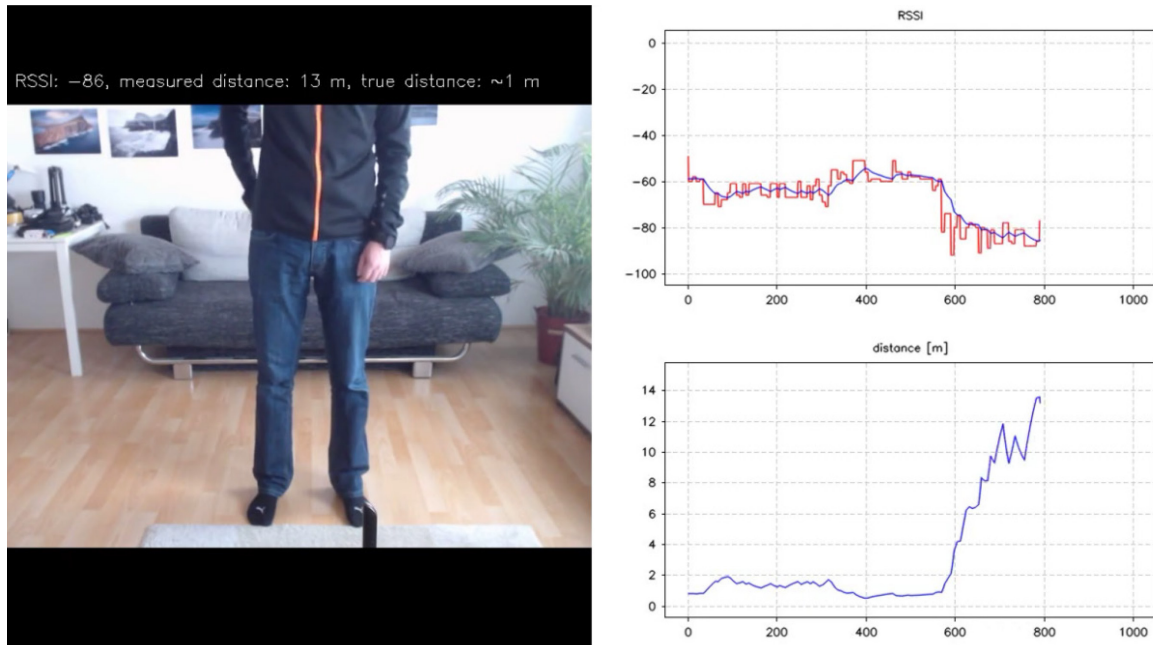
Subsection 1.1.2 3D Global Sensing

- **Ranging Technologies**

Ranging technology is usually used in the telecommunication field for device localization. Some road-side unit-based localization schemes use ranging techniques to estimate the distance between the vehicle and the RSU. The three most common ranging technologies are: Received Signal Strength (RSS), Time of Arrival (TOA), and Angle of Arrival (AOA).



(A)



(B)

Figure 1-5 Experiment of Distance Measurement based on RSS Technology

1) RSS method is a well-known, low-cost method for hardware-constrained systems, in which the distance between the transmitter and the receiver is estimated based on the received signal strength by using theoretical radio propagation models. However, the reliability of the RSSI estimates cannot be guaranteed in environments with multi-path and shadowing effects because of the resulting attenuation of the received signal. Figure 1-5 shows the experiment about Bluetooth signal strength test. When the tester puts a phone in front, the distance measured by the receiver is about two meters (see Figure 1-5 A), while when the tester puts the phone to the back, the detected distance soars to fourteen meters (see Figure 1-5 B).

2) Time of arrival (TOA), sometimes called time of flight (TOF), is the travel time of a radio signal from a single transmitter to a remote single receiver. (Also, in some scenarios, there is no special equipment installed on the receiver.) However, the method highly relies on the accuracy of the time, and therefore, the accuracy is relatively low in practice. Therefore, an improved method Time Difference of Arrival (TDOA) is proposed to reduce the error caused by the time synchronization. Compared with TOA, more research implements the method for the distance measurement. However, three stations are necessary to localize the vehicle.

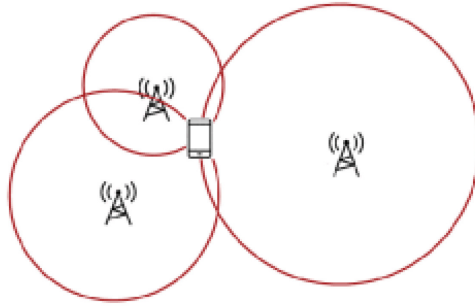


Figure 1-6 Classic Three Towers Device Localization based on TDOA

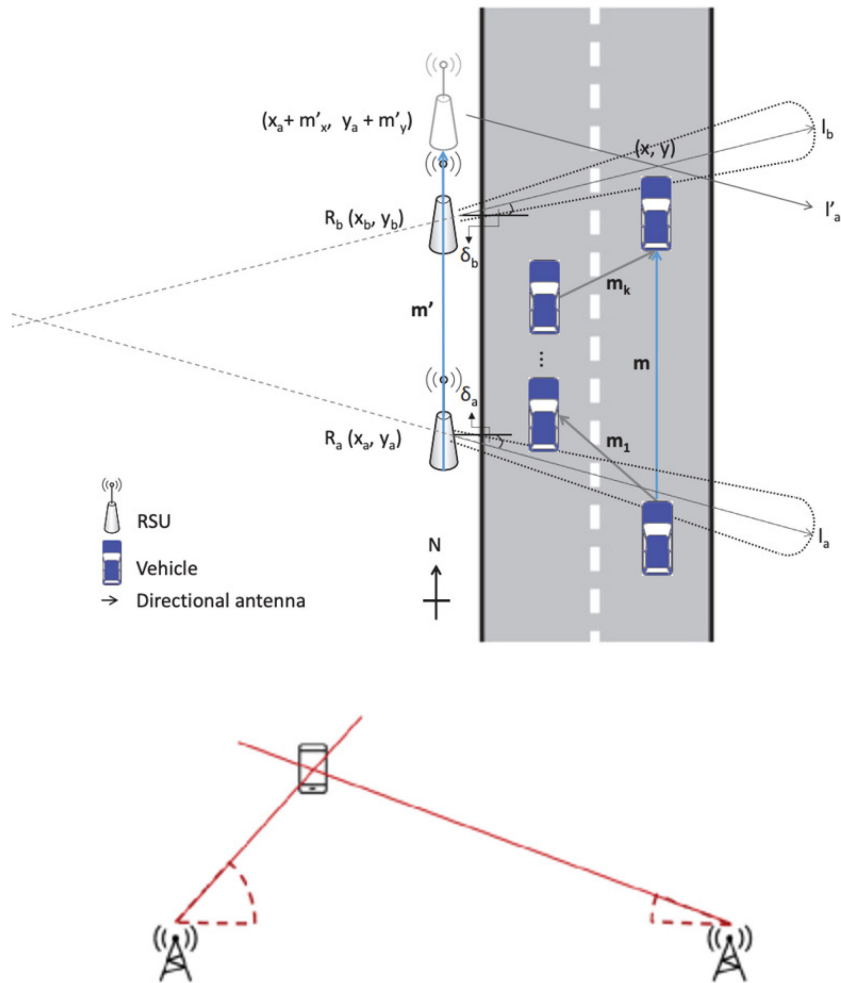


Figure 1-6 Classic Two Towers Localization based on AOA

3) The angle of arrival (AOA) of a signal is the direction from which the signal (e.g., radio, optical or acoustic) is received. Measurement of AOA can be done by determining the direction of propagation of a radio-frequency wave incident on an antenna array or determined from maximum signal strength during antenna rotation. AOA is an advanced telecommunication method for vehicle localization based on only two RSU, while the powerful directional antennas are required by the method.

Subsection 1.2 Problem Statement

Advanced driver assistance systems (ADAS) are on-board systems that assist drivers while driving and parking (Lindgren et al., 2006). Typical ADAS subsystems include adaptive cruise control (ACC), forward

collision warning (FCW), pedestrian collision warning (PCW), and parking assistance system (PAS). They are believed to be able to enhance the vehicle systems for safety and better driving.

Environment perception and understanding remains a critical challenge for ADAS (Wang et al., 2019). Specifically, precise 3D sensing of vehicles and pedestrians (Arnold et al., 2019), and understanding road users' intentions (Song et al., 2016), are needed for roadway traffic decision making and parking assistance. Current advanced driving systems mainly rely on local sensing methods environmental perception (Geiger et al., 2012), which work well under certain circumstances, but has the following limitations:

- Local sensing relies on only onboard sensors, which makes the perception system limited in both spatial (limited horizontal and vertical fields of views and suffering from occlusions) and temporal (only observing other traffic participants for a limited amount of time and hard to fully understand their intentions) dimensions.
- On-vehicle smart sensors are expensive, require a high communication bandwidth for transmitting the generated huge amount of point cloud data and suffer from degraded performances in cases of rain, snow, fog, and dust.
- Onboard cameras are suitable for 2D vehicle and pedestrian detections (Ren et al., 2018) (Redmon et al., 2018), but have poor performances when used for 3D sensing.

On the other side, road-side units have been being quickly developed during the last decades (Pu et al., 2019). With the emerging of low-latency, high-speed communication technologies like 5G (Hetzer et al., 2019), the affluent real-time data collected by the road-side sensing equipment empowers the global sensing methods, which can be used as complementary data resources to broaden the views of current driving perception systems in both spatial and temporal dimensions. Multiple challenges (such as occlusion issues and low precisions of monocular 3D detections) can be solved by fusing local sensing and global sensing instead of solely relying on onboard sensing technologies. Therefore, the team proposes a global structural information sensing system (3D-SISS) based on Mobile Unit for Sensing Traffic (MUST) cooperating with the on-board sensors to address the challenges mentioned before

Subsection 1.3 Solution Introduction

The structure of the solution proposed by the UW team is shown in Figure 1-7. The system contains two components, customized MUST sensor installed on site and the back server in UW STAR Lab.

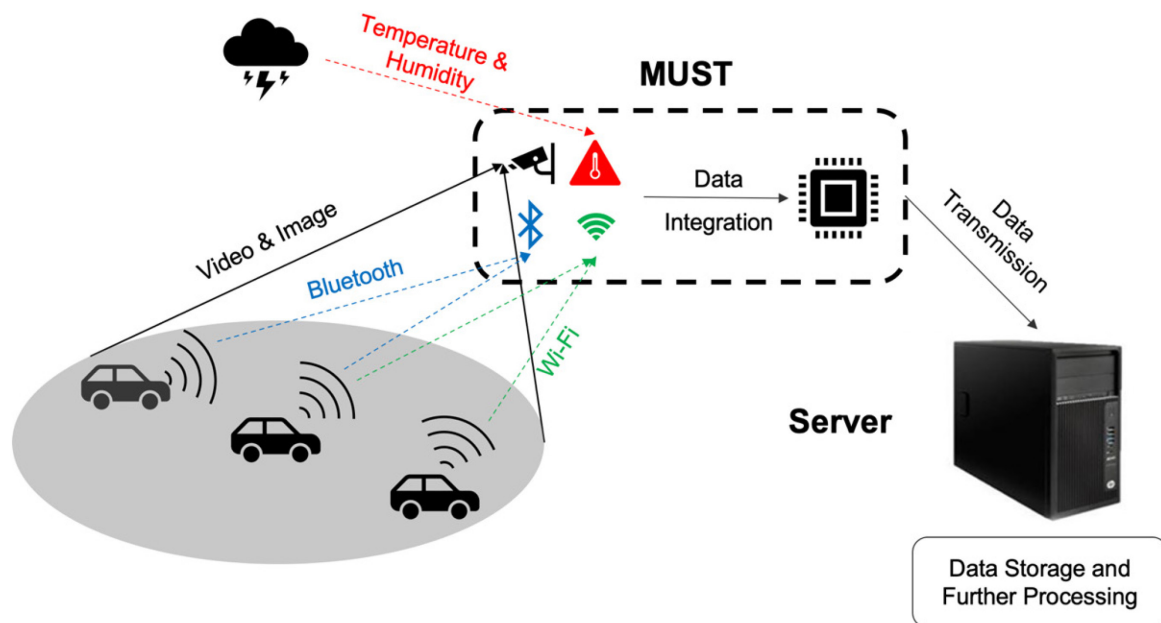


Figure 1-7 MUST Sensor Based Solution Structure

Customized MUST sensor is a representative roadside device and collects data by both road-side devices and on-board sensing equipment for algorithms development. The MUST sensor (see Figure 1-8) is a multi-source traffic sensing road-side unit developed by STAR Lab at the UW. The sensing modules of the MUST sensor include Wi-Fi and Bluetooth sensing, image camera, thermal camera, and environmental sensing. The MUST sensor can transmit data through 4G/5G data plan, or through Long Range (LoRa) and Narrow Band Internet of Things (NB-IoT) data communication protocols. In this research, the MUST sensor will be employed as the road-side unit for data collection and communication, and on-board IoT devices, e.g., Raspberry Pi or NVIDIA Jetson, connected with a monocular camera will be employed as on-board equipment. The research team will design a system by integrating road-side and on-board units for driving assistance.

Specifications

- **Weight:** 10 pounds
- **Dimensions:** 170 mm (length), 170 mm (width), 300 mm (height)
- **Power supply:** 12 V (DC)
- **Power consumption:** < 5W
- **Communication:** Ethernet, 3G/ 4G/ 5G
- **Temperature range:** -40°F to 158 °F
- **IP rating:** IP 65
- **CPU:** ARM 1176 JZF S 700 MHz
- **System:** Linux
- **Configuration:** remote software update



Figure 1-8. Illustration of the MUST sensor

In the project, a MUST sensor was installed on the road site to realize real-time traffic scene monitoring and data analysis. During the process, the team introduces 3D-GSISS into the system to realize 3D global sensing. 3D-SISS proposed in the project can project the 2D pixel coordinate to 3D real-world coordinate by matching the car keypoints detected and the real car dimension. The report will introduce the system in three steps. Firstly, a Region-based Convolutional Neural Network (RCNN) is applied to the original image/video data for vehicle detection and classification. Then, the ROI extracted by RCNN is input to the hourglass network and graph neural network (GNN) to estimate all keypoints. Finally, based on the classification results obtained from RCNN, 3D-SISS matches 2D keypoints with average 3D vehicle dimension to get all demanded parameter matrices for coordinate projection. Additionally, a self-learning algorithm is introduced in 3D-SISS to update the parameter matrices for better performance with the increasing number of passing vehicles.

Finally, the collected image data and the processed results are transmitted back to the server in STAR Lab for storage and further analysis. Also, in the project, the team used the data to validate the sensing results and found the accuracy can reach 95%.

Section 2 Literature Review

Environment perception and understanding remains a challenge for (Advanced Driver Assistance Systems) ADAS. Specifically, precise 3D detection of vehicles and pedestrians, and understanding road users' intentions, are needed for roadway traffic decision making and parking assistance. As mentioned in the proposal, this project aims to address the perception challenge by developing a cooperative perception method that fuses sensor information from vehicle-onboard and roadside devices, to extend sensing ranges and improve sensing reliability. Considering the limited computational and storage resources on the edge-side, edge-device-based artificial intelligence algorithms will be employed for enhancing the efficiency of the proposed methods without losing accuracy.

Considering that the main technical approach for this project is to first merge information obtained by road-side devices and on-board sensing equipment and then conduct 3D object detection, literature in the following topics will be reviewed and summarized: cooperative perception and 3D object detection.

Subsection 2.1 Cooperative Perception

An accurate representation of the driving environment is the core feature of an intelligent vehicle's perception system and is essential for the safe operation of vehicles in complex environments. Most of the existing perception methods rely on fusion of data from different sensor modalities (e.g., lidar and camera) to overcome the limitations of single sensor types and increase detection performance. However, multimodal sensor data fusion from a single point of view has limitations including occlusions and restricted view. To address this, cooperative perception methods that integrate spatially distributed sensors have been proposed. Basically, cooperative perception refers to a technology that increases perception range and field-of-view by exchanging sensor information between vehicles (and roadside devices). In this report, we will review cooperative perception studies from the following aspects: image based cooperative perception, point cloud based cooperative perception, resource allocation for cooperative perception, and datasets for cooperative perception research.

Subsection 2.1.1 Image based Cooperative Perception

Image based cooperative perception studies emerged around the 2010s. They integrate cameras from multiple vehicles to overcome occlusion issues. (Bagnell et al., 2010) developed a method that combines vehicle onboard data with static satellite imagery for path planning. The major limitation of this method is that static images cannot capture vehicle and pedestrian dynamics. (Kim et al., 2013) presented a framework for multi-vehicle cooperative perception and identified several major challenges: map merging, vehicle identification, sensor multi-modality, the impact of communications, and impact on path planning. (Liu et al., 2013) further considered the motion planning problem given long-range cooperative sensing information.

(Kim et al., 2014) proposed a multi-vehicle cooperative perception system to provide see-through, lifted-seat, satellite and all-around views to drivers. Based on the cooperative perception system, they then designed a cooperative driving system with examples of see-through forward collision warning, overtaking/lane-changing assistance, and automated hidden obstacle avoidance. Figure 2-1 shows the visualization results for their cooperative perception study, where subfigure (b)-(d) shows the original camera data from multiple vehicles that are in the situation of car following, (a) shows the map merging results, and (e)-(h) demonstrates the see-through function (e.g., the ego vehicle can sense the second leader with its view occluded by the first leader). While this study tends to be one of the most influential studies in the field of cooperative perception in the 2010s, it has the two limitations: (1) peer-to-peer data fusion may create a huge burden to both communication and computation systems and (2) their object detection methods are based traditional image features instead of convolutional neural networks, which means the system's performance can be further improved by incorporating current state-of-art object detection methods.

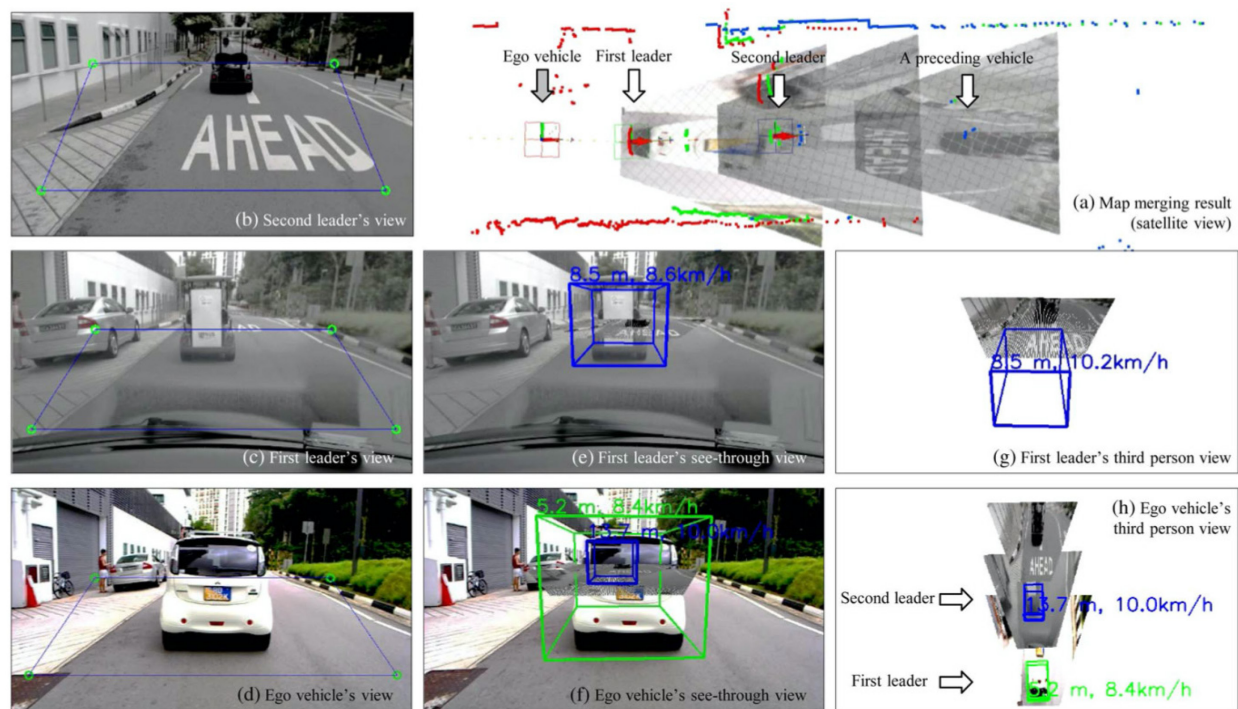


Figure 2-1: Cooperative Perception Results

Subsection 2.1.2 Point Cloud based Cooperative Perception

A point cloud is a set of data points in space, which is generally produced by LiDAR (Light Detection And Ranging) and depth cameras. Points clouds are popular for 3D object detection because depth and pose information can be easily extracted compared to monocular cameras. Cooperative perception research based on point clouds emerged in recent years.

To reduce the cost and data storage burden induced by having each vehicle equipped with a standalone LiDAR device, (Jayaweera et al., 2019) proposed to develop a coordinated set of LiDARs along the road at an elevation that can give an integrated view with a much larger field of vision, as shown in Figure 2-2. For their system, each Elevated LiDAR (ELiD) unit comprises two stationary LiDAR sensors angled towards relevant road sections. The ELiD module is mounted in an elevated position in road infrastructure to create a bird's-eye view. Each ELiD module can receive supportive sensor data from moving vehicles and transmit decision commands to vehicles. A series of ELiDs are connected by fiber and can collectively produce an HD road map. The major limitation of this study is that they only conducted experimental computations to verify the feasibility of the system, instead of implementing the system.

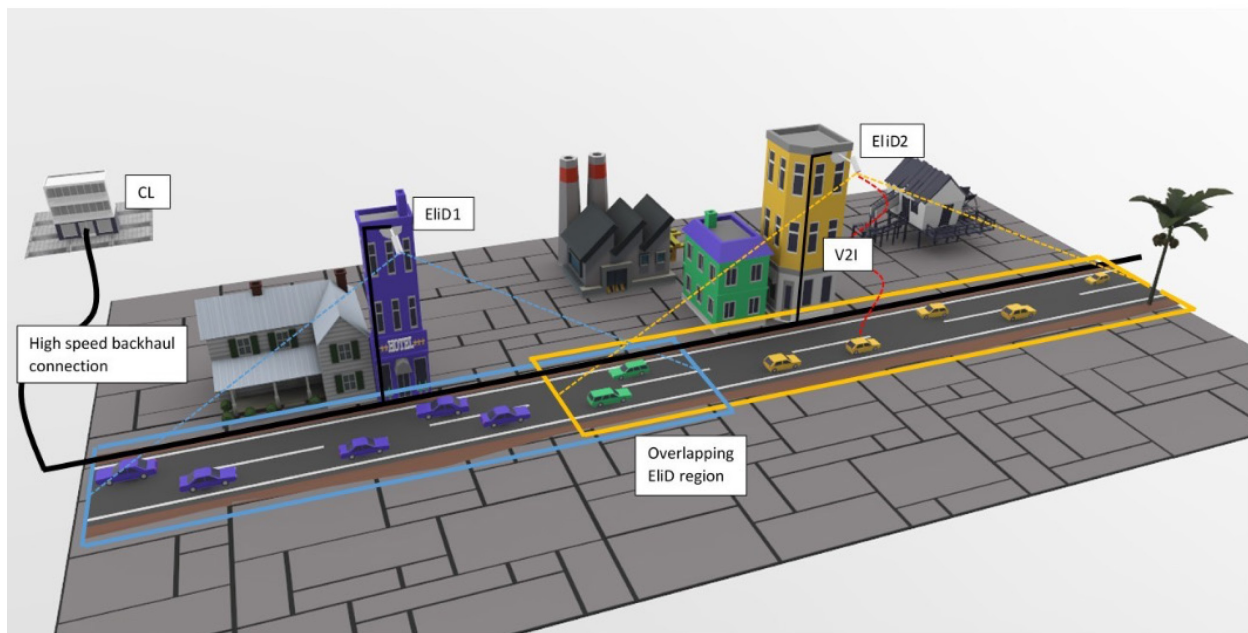


Figure 2-2: Cooperative Perception with Elevated LiDAR Along the Road

(Chen et al., 2019a) proposed a raw-data level cooperative perception approach that fuses Lidar sensor data collected from different positions and angles of connected vehicles. A point cloud-based 3D object detection method was then developed to work on the aligned point clouds. Based on the KITTI benchmark, they found that the proposed system can extend sensing area, improve detection accuracy and promote augmented results. Considering that raw point cloud data may create a burden for network bandwidth and computation power, a follow-up study from (Chen et al., 2019b) proposed a point cloud feature based cooperative perception framework. They claimed that feature-based data is sufficient for object detection training, and the small size of feature based data can facilitate real-time edge computing. Based on experiments on the KITTI dataset, the system provided around 10% improvement for detection within 20 meters and 30% for further distances. It also achieved faster edge computing with a low communication delay, requiring 71 milliseconds in certain feature selections. (Hurl

et al., 2019) researched the trust issue in cooperative 3D object detection and proposed TruPercept to avoid malicious attacks on cooperative perception systems.

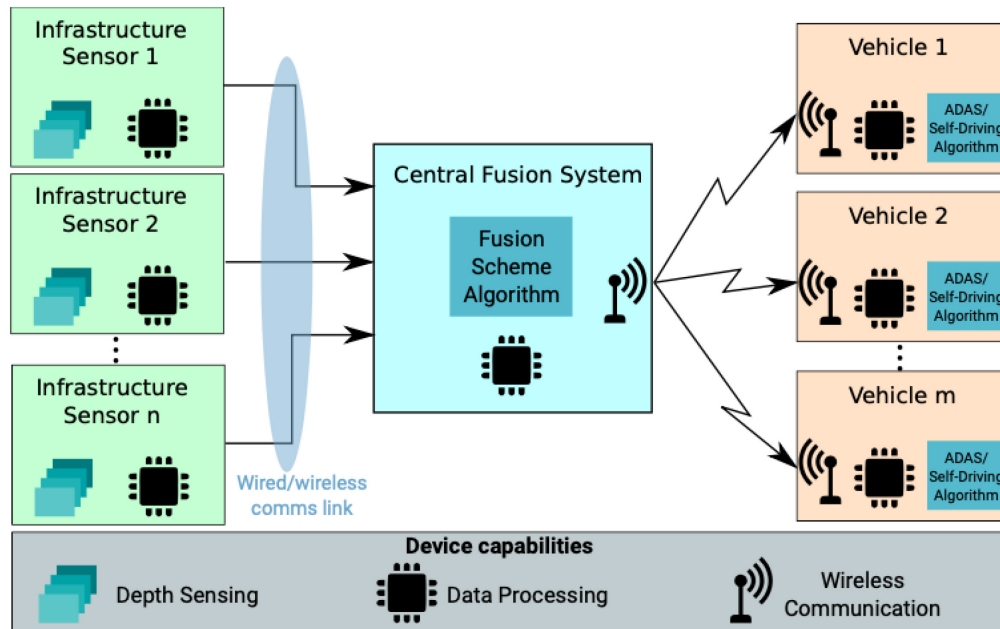


Figure 2-3: Cooperative 3D Object Detection System

To address the limitation of the aforementioned studies that they share peer-to-peer (V2V) information and fuse data locally, (Arnold et al., 2019) proposed a central system that can fuse data from multiple infrastructure-side LiDAR sensors which can reduce both sensor and processing costs through shared resources. Their system architecture is shown in Figure 2-3. Several infrastructure LiDAR sensors, each equipped with a local processor, are linked to a central fusion system through wired or wireless data links. The central fusion system is equipped with a powerful processor to fuse data and a wireless broadcast system to send cooperative perception messages to vehicles nearby. Two data fusion schemes were tried: early fusion scheme that combines point clouds directly and late fusion scheme that combines independently estimated bounding boxes. By testing with a synthetic dataset, they found that the early fusion scheme outperformed the late fusion by a large margin, and the cooperative perception can recall more than 95% of the objects compared to 30% for single-point sensing in the most challenging scenario.

Subsection 2.1.3 Resource Allocation for Cooperative Perception

Since cooperative perception requires data sharing and data fusion computation, there are several studies (Kong et al., 2020) focusing on how to optimize resource allocation to minimize latency. (Jayaweera et al., 2020) studied the resource allocation problem for the Elevated LiDAR system. They used decoder error probability and the number of channels used to parameterize the reliability and

latency of the system. To analytically evaluate the efficiency of the ELiD device under different vehicle densities, a maximum decoder error probability minimization problem and a total energy minimization problem were considered. To effectively assign data to computing servers for ELiD systems, (Lucic et al., 2020) formulated a mixed-integer programming problem which minimizes the uplink and downlink hop-by-hop transmission average latency plus computation time for each ELiD while considering different bandwidth allocation schemes. (Aoki et al., 2020) used reinforcement learning to select which kinds of data to transmit so as to minimize packet loss and increase detection accuracy

Subsection 2.2 3D Object Detection

3D object detection is the core function of the intelligent driving perception system. It refers to the estimation of 3D bounding boxes specifying the size, 3D pose (position and orientation) and class (vehicle, pedestrian) of the objects in the driving environment. Based on the data source, there are two types of 3D object detection methods: image based and point cloud based.

Subsection 2.2.1 LiDAR based 3D Object Detection

LiDAR stands for “Light Detection and Ranging.” This technology can be mounted to aerial vehicles or roadside units to send laser pulses to the other objects, and once the laser returns to the sensor, the LiDAR system will record data based on the information received. (Hecht et al., 2018) Each data recorded by LiDAR is a 3D point that has its own set of X, Y, and Z coordinates and, in some cases, additional attributes. Point Clouds are a collection of points that represent a 3D shape or feature. Therefore, compared with other detection methods, LiDAR provides reliable depth information that can be used to localize objects accurately and characterize their shapes (Dong et al., 2017) (Behroozpour et al., 2017). As a result, in the passing decade, many studies and algorithms have been developed (Chen et al., 2017) (Qi et al., 2018) (Yang et al., 2018) to process the point clouds data collected by LiDAR sensor for 3D object detection, which benefits many applications like 3D mapping (Brummelen et al., 2018), lane change detection, obstacle detection in the transportation field.

However, unlike images, LiDAR point clouds are sparse and have highly variable point density (Brummelen et al., 2018), due to factors such as non-uniform sampling of the 3D space, effective range of the sensors, occlusion, and the relative pose. To address the problem, sensor fusion technology about cameras and LiDAR is proposed as an effective solution in recent studies. The image data collected by the cameras can provide rich texture descriptions of the surrounding objects while the depth information is hard to obtain. LiDAR is a relatively low-cost sensor to get accurate depth information compared with stereo cameras. Additionally, all the 3D data can be provided by the LiDAR sensor, which can work as the ground truth information when we apply the computer vision algorithm to cameras. As a result, more and more researchers pay attention to LiDAR and camera fusion technology in the

intelligent transportation field. Figure 2-4 shows the recent algorithms proposed for the research topic in various practical applications.

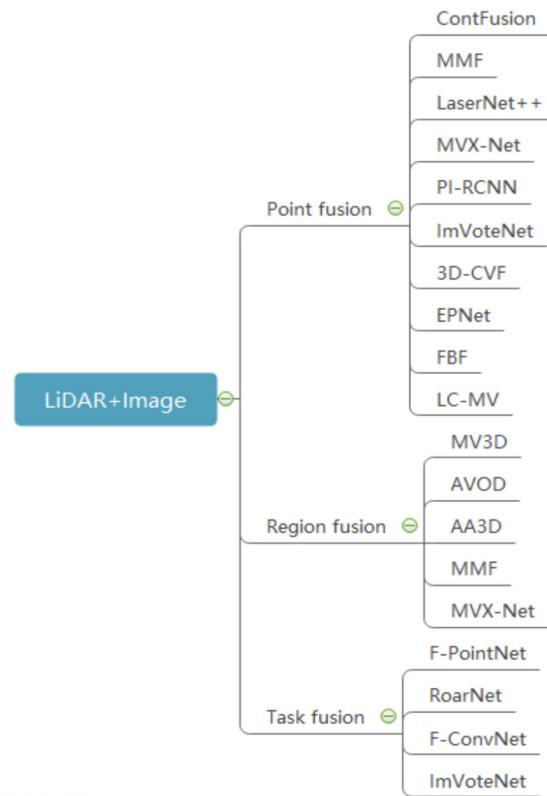


Figure 2-4: A review of Lidar and Camera Fusion Algorithm

There are three categories of point cloud-based 3D object detection methods: multi-view, voxel, and point based methods (Yang et al., 2019). For multi-view methods, (Chen et al., 2017) proposed the MV3D method which projects LiDAR point cloud to the bird's-eye view and trains a region proposal network (RPN) to predict 3D bounding boxes for objects. The limitations of multi-view based methods are that they cannot well handle small objects like pedestrians and cyclists, and they do not deal with cases with several objects in depth direction. Voxel based methods project point clouds into voxel grid and generate 3D proposals by conducting 3D convolutions on voxels. Representative methods include VoxelNet (Zhou and Tuzel, 2018), SECOND (Yan et al., 2018), and PointPillars (Lang et al., 2019). For point based methods, they rely on the famous PointNet for data representation learning. F-PointNet (Qi et al., 2018) uses frustum proposal from 2D object detection as candidate 3D bounding boxes, which means its performance is hugely impacted by 2D object detectors. Recent studies tried to generate 3D proposals directly from point clouds, e.g., PointRCNN (Shi et al., 2019).

Subsection 2.2.2 Radar based 3D Object Detection

In general, radar is more reliable in harsh environments, such as rain, fog, and other low visibility conditions. The frequency modulated continuous wave (FMCW) radar works in the millimeter wave (MMW) frequency band (30-300GHz) below the visible light. Thus, the following strengths of MMW are worth mentioning: 1) MMW has a strong ability to penetrate fog, smoke, and dust; 2) The wide bandwidth and higher working frequency endow FMCW radar with excellent detection capability. Significant research in radar object classification has demonstrated its feasibility as a good alternative when cameras fail to provide good results. Despite these good characteristics, it is still very challenging for radars to distinguish different types of objects since it's hard to extract and analyze the semantic information from radio signals. Also, another real challenge mission is to distinguish features of various objects from a radar frame extracted. The surrounding obstacles, such as roadside trees, signs, and even buildings, also can bring in a lot of noise.

Previous research studies show promising and encouraging potential for using radar for object detection. By extracting features manually, (Heuel et al., 2011) use support vector machines (SVM) to classify objects to distinguish cars from pedestrians. Then, (Angelov et al., 2018) uses neural networks to extract features from short-time Fourier transform (STFT) heat maps and evaluates the three categories of cars, pedestrians, and cyclists. However, this method only focuses on classification, which assumes that only one object is properly recognized in the scene and is not suitable for complex autonomous driving scenes. Recently, several methods on radar and image fusion target detection have been proposed in (Gao et al., 2019) (Richards et al., 2005) (Simonyan et al., 2014), which combines the statistical detection algorithm with the neural network classifier. However, such approaches are prone to generate false positives, i.e., obstacles are recognized as target objects. In addition, the laborious human annotations required for this method are often not available given the resource constraint.

Subsection 2.2.3 Monocular Camera based 3D Object Detection

Because of the lack of accurate depth information, monocular camera-based 3D object detection is more challenging or even an ill-posed problem compared to using LiDAR or multi-camera systems. However, there is still a growing trend in monocular 3D detection research because monocular cameras have the advantage of low-cost, low-power, and easy to deploy in vehicles. There are mainly two types of monocular 3D object detection algorithms: prior knowledge-based and geometry constraints based.

For prior knowledge-based methods, Mono3D (Chen et al., 2016) utilized the fact that 3D objects are on the ground plane. They also used prior 3D shapes of vehicles to reconstruct 3D bounding boxes, as shown in Figure 2-5. They first sample candidate 3D bounding boxes with typical physical sizes in the 3D space by assuming objects are on the ground plane. Then 3D candidate boxes are projected into 2D image plane for search. By using predefined physical sizes, they avoided multi-scale search in the image.

During the search, they used multiple features (class semantic, instance semantic, contour, object shape, context, and local) to score the object proposals. Non-maximum suppression was used to generate final object detection results.

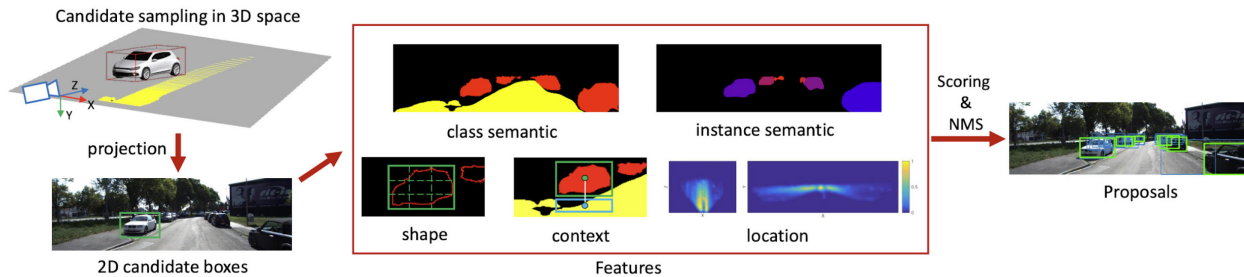


Figure 2-5: Overview of the Mono3D approach (Chen et al., 2016)

Deep MANTA (Chabot et al., 2017) predicts information for objects in 3D using key points and 3D CAD models. SubCNN (Xiang et al., 2017) learns from 3D CAD models viewpoint-dependent subcategories to capture patterns of both shape, viewpoint, and occlusion. (Barabanau et al., 2019) used CNN to estimate the correspondences between detected 2D key points and 3D counterparts. For all object instances from an image, 3D-RCNN (Kundu et al., 2018) introduces an inverse-graphic framework. A differentiable loss of rendering and comparison was proposed to learn 3D results via 2D information. There is another line of research (Ma et al., 2019; Manhardt et al., 2019; Wang et al., 2019; Xu and Chen, 2018) that first use standalone network for depth estimation or instance segmentation, and then use CNN to extract 3D bounding boxes, they rely on ground-truth depth labels in the training stage.

Compared to prior based methods, geometrical constraints-based methods use only the RGB images as input rather than relying on external data, network structures or pre-trained depth estimation models. Deep3DBox (Mousavian et al., 2017) infers 3D knowledge from a 2D bounding box taking into account the projection geometric constraints. OFTNet (Roddick et al., 2018) presents an orthographic feature which transforms image-based characteristics into an orthographic 3D space. MonoGRNet (Qin et al., 2019) predicts 3D object locations from a monocular RGB image, taking into account the 2D projection geometric reasoning and the unobserved dimension depth.

Current state-of-art monocular 3D object detection methods include RTM3D (Li et al., 2020), SMOKE (Liu et al., 2020), and MonoPair (Chen et al., 2020). For RTM3D, they first calculate the nine key points of the 3D bounding box in the image space and then use the 3D and 2D viewpoint geometric relationship to recover the 3D space dimension, position and orientation, as shown in Figure 2-6. They claimed that their method is the first real-time one for monocular image 3D detection and achieved state-of-art performances.

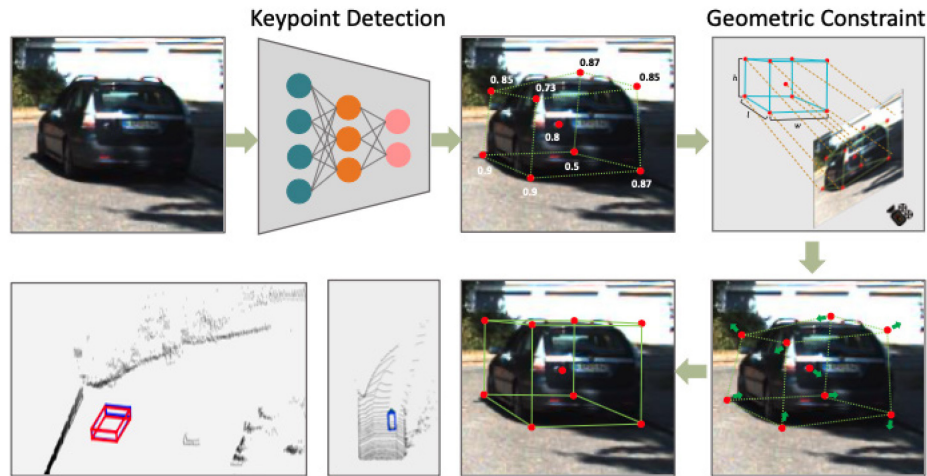


Figure 2-6: Overview of the state-of-art RTM3D approach (Li et al., 2020)

For SMOKE, the authors argue that 2D detection is not needed and will introduce non-negligible noise for monocular 3D object detection. Instead, they predict 3D bounding boxes by combining a single key point estimate with regressed 3D variables. They also used a multi-step disengagement method for the construction of a 3D bounding box which will greatly enhance both training convergence and detection accuracy. This method's main advantage is that it does not require complicated pre-/post-processing, additional data and refining.

For MonoPair, the authors argue that previous detectors consider each 3D object as an independent training part which will result in a lack of information for occluded objects. Therefore, they proposed a novel approach to enhance the detection of monocular 3D objects by taking into account the paired sample relationship. This helps one to encode spatial restrictions from its neighboring neighbors for partially-occluded objects.

Subsection 2.3 Datasets for Cooperative Perception Research

Compared to single-point object detection, there are limited datasets for cooperative perception. Chen et al. (2019a, b) merged two sequential frames from the KITTI dataset to simulate a cooperative perception dataset. However, this method can only create limited scenes where all other objects remain static. (Hurl et al., 2019) generated a synthetic cooperative perception dataset based on a game engine called Grand Theft Auto V (GTA V) in urban scenarios. (Arnold et al., 2019) used the CARLAR simulator to generate cooperative LiDAR sensing dataset in T-junction and roundabout scenarios, as shown in Figure 2-7 where subfigures (a), (c) show the sensors configuration and subfigures (b), (d) show the fused point clouds, with each color representing a sensor. To the best of our knowledge, there is no public available field data for cooperative perception research.

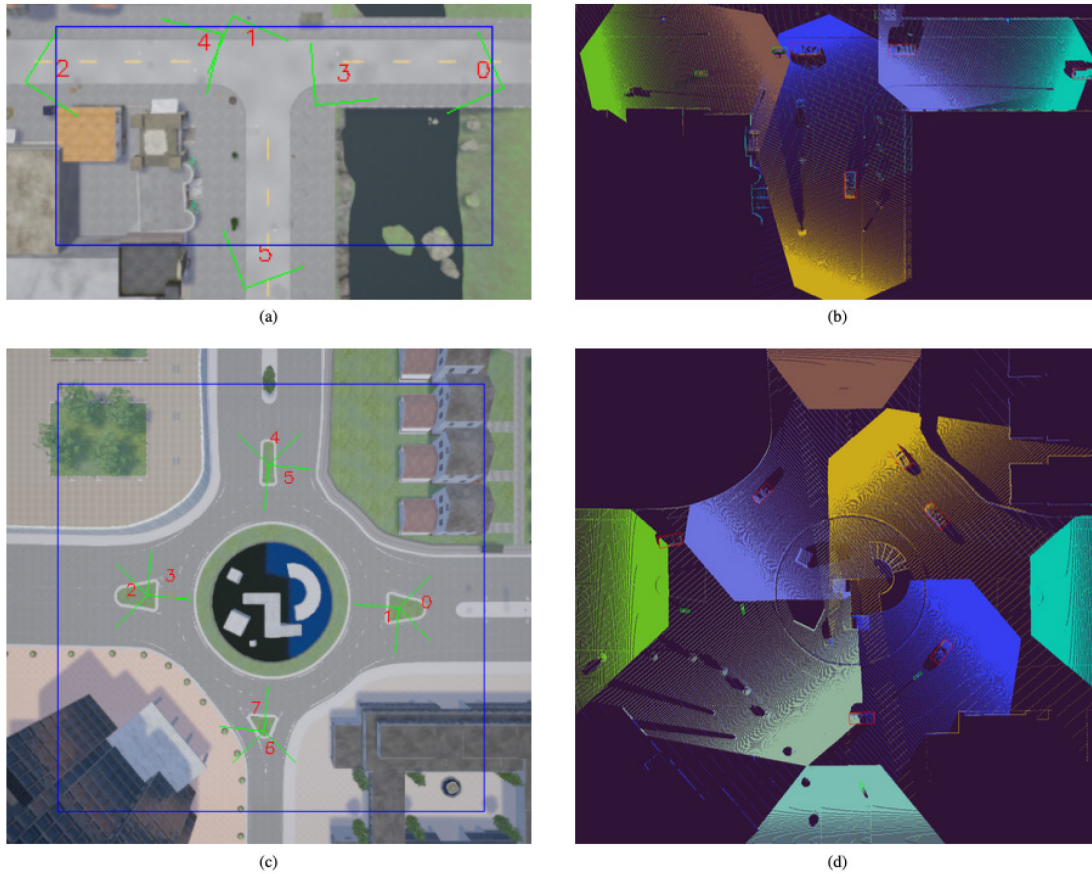


Figure 2-7: Cooperative 3D object detection system proposed by Arnold et al. (2019)

Driving Assistance System requires high accuracy Computer Vision Technologies which rely on large amounts of high quality and propriate image data for the algorithm training. However, before the practical tests, it is not easy to record the data on our own. Therefore, in the project, the team investigated multiple open data resources in the field and tried to search a suitable dataset to fit the demands of the project. Finally, three open datasets are introduced in the project to assist the team complete the task. At this time point, the team has just started the testing and the results will be reported in the next task report. Figure 2-8 shows all the self-driving and object detection datasets. The following part introduces the three datasets we finally selected to show their advantages for the project.

Name	Provider	Cit.	Year	Location	Diversity					Traffic					Sensors					Annot.	Track.	Frames(k)	Classes		
					R	S	N	D	u	H	Re	Ru	c	Vi	Li	GPS	Th	If	Ra	2D	3D				
en	Hesai & Scale	N/A	2019	Silicon Valley	?	?	?	?	?	?	?	?	?	X	X**	X					X	X		60	28
	Aptiv	16	2019	Boston, Singapore	X		X	X	X					X	X	X				X	X	X	X	40	23
	Waymo Inc.	N/A	2019	United States	X		X	X	X	X	X	X		X	X						X	X	X	200	4/5
	Lyft	N/A	2019	Palo Alto, London, Munich	?	?	?	?	?	?	?	?	?	X	X	X					X	X	X	55	23
	Honda Res. Inst.	4	2019	San Francisco					X						X	X	X					X	X	27	8
ge	Bosch N.A. Res.	0	2019	Unknown	X			X	X	X				X							X	X		200	1
	VCCIV	2	2019	Changshu			X		X					X	X							X	X	120	3
	University of Tokyo	31	2018	Japan	X				X		X				X*						X			9	8
	KAIST	13	2018	Seoul			X	X	X		X			X	X	X	X	X			X		X	8.9	6
	ispectral	U. of California	103	2018	San Francisco, New York	X	X	X	X	X	X	X	X		X		X				X		X	100	10
n Platform	Baidu Inc.	N/A	2018	China	X				X	X	X	X		X	X					X	X	X	20	4/5	
	Delft U. of Tech.	6	2018	European Cities	X		X	X	X					X							X		X	47	7
al Sensing	FLIR Sytems Inc.	N/A	2018	Santa Barbara			X	X	X	X				X			X				X		X	14	5
	Oxford U.	3	2018	Germany, Netherlands, UK	X	X	X	X	X		X			X							X		X	279	4

TuSimple	TuSimple	N/A	2017	Unknown	X					X									X		X	12	it
NEXET	Nexar	N/A	2017	Around the globe	X	X	X		X	X	X	X		X					X			55	5
Multi-spectral Object Detection	U. of Tokyo	5	2017	Tokyo			X	X					X	X			X		X			7.5	5
Bosch Small Traffic Lights	Bosch N. A. Res.	45	2017	San Francisco	X				X		X	X		X				X		X		13	15
City Persons	Max Planck Inst. (Info.)	105	2017	Germany, France, Switz.					X					X		X			X			5	4
Udacity	Udacity	N/A	2016	Mountain View					X					X					X		X	34	3
JAAD	York University	16	2016	Ukraine, Canada	X	X	X	X	X					X					X		X	88	itt
Elektra (CVC-14)	Auton. U. of Barcelona	N/A	2016	Barcelona			X		X					X			X		X		X	10	1
Tsinghua-Daimler Cyclist	Daimler AG	41	2016	Beijing					X					X					X			15	6
KAIST Multispectral Pedestrian	KAIST	195	2015	Seoul			X	X	X					X		X	X		X		X	95	3
Highway Workzones	CMU	21	2015	United States	X					X				X					X		X	17	9
KITTI	Karlsruhe Inst of Tech.	5415	2013	Karlsruhe					X	X	X	X	X	X	X	X			X	X	X	15	2
German Traffic Sign	Ruhr U.	298	2013	Germany	X			X	X	X	X	X		X*					X			5	43
LISA Traffic Sign	U. of California	382	2012	San Diego			X	X	X					X*					X			6.6	47
TME Motorway	Czech Tech. U.	127	2011	Northern Italy						X				X					X		X	30	2
Belgian Traffic Sign	ETH Zurich	261	2011	Belgium					X					X*					X			9	62

LISA Vehicle Detection	U. of California	333	2010	California						X	X								X		X	2.2	1
TUD-Brussels Pedestrian	Max Planck Inst. (Info)	182	2009	Belgium						X									X		X	1.6	1
ETH Pedestrian	ETH Zurich	547	2009	Zurich						X									X	X	X	4.8	1
Caltech Pedestrian	Caltech	1009	2009	Los Angeles						X									X		X	250	1
Daimler Pedestrian	Daimler AG	1177	2008	Beijing						X									X		X	21	1

Table 2-1. Self-driving and Object Detection Datasets Overview

Subsection 2.3.1 KITTI Dataset

The KITTI dataset is jointly founded by Karlsruhe Institute of Technology in Germany and Toyota's American Institute of Technology. It is currently the world's largest evaluation dataset for computer vision algorithms in autonomous driving scenarios (Chen et al., 2019). The data set is used to evaluate the performance of computer vision technologies such as stereo, optical flow, visual odometry, 3D object detection and 3D tracking in a car environment . KITTI contains real image data collected from urban, rural and highway scenes, with up to 15 vehicles and 30 pedestrians in each image, as well as various degrees of occlusion and truncation. The entire dataset consists of 389 pairs of stereo images and optical flow maps, a 39.2 km visual ranging sequence, and images of objects labeled with more than 200k 3D frames (Geiger et al., 2012). The data is sampled and synchronized at a frequency of 10 Hz. The original dataset is classified as "Road", "City", "Residential", "Campus" and "Person". Figure 2-8 (Geiger et al., 2013) shows the five categories of the dataset. The original data was collected in 5 days in 2011, and the size of the data can reach 180GB.



Figure 2-8. Five Categories in KITTI data sample

In the Driving Assistance System development, 3D object detection is a significant task. In this aspect, KITTI data has a unique advantage. In the data, the labels are divided into car, van, truck, pedestrian (walking), pedestrian (sitting), cyclist, tram and misc. (Everingham et al., 2011). For each image data, a txt file is attached to describe the objects in the image (See Figure 2-9).

In summary, KITTI data is a good dataset with a large amount (i.e., 5 days duration) of high quality images in diverse environments. Additionally, the high precision and detailed labels can accelerate the back propagation process which help the team address the problems in the algorithm. However, the diversity (i.e., various driving environments like raining or snowing) of the KITTI dataset are not good enough (see Figure 2-9). In general, it is still worth to try the dataset in the algorithm development.



Figure 2-9. Object detection in KITTI data sample

Subsection 2.3.2 Waymo Dataset

Waymo dataset is generated by the Waymo self-driving vehicles in over 10 million autonomous miles driving in 25 cities. Therefore, in the official website, Waymo claimed “It is one of the largest, richest, and most diverse self-driving datasets ever released for research”. At present, the released open data consists of 1000 driving segments. Each segment captures 20 seconds of continuous driving, corresponding to 200,000 frames at 10 Hz per sensor. The size of the data is not as large as KITTI’s, while it covers a more diverse driving environment. Figure 2-10 shows the sample data proposed by Waymo. This dataset covers dense urban and suburban environments across Phoenix, AZ, Kirkland, WA, Mountain View, CA and San Francisco, CA capturing a wide spectrum of driving conditions (day and night, dawn and dusk, sun and rain).



Figure 2-10. Waymo data in diverse environment

For the 3D object detection, the dataset includes lidar frames and images with 4 categories: vehicles, pedestrians, cyclists, and signage carefully labeled, capturing a total of 12 million 3D labels and 1.2 million 2D labels. Compared with the KITTI dataset, Waymo data doesn't have the detailed labels for all the objects captured in the image. Additionally, only four kinds of labels are introduced in the process.

In summary, Waymo is a good dataset for our algorithm training. It consists of a large amount of high-quality image data in diverse driving environments, which can help the algorithm learn the features of the image in various scenarios. Though it contains 220k detection frames, it only classifies them as 4 categories. In general, it is still good enough for our algorithm training in the project.

Subsection 2.3.3 Lyft Dataset

The Lyft dataset is the latest one of the three, as a result, the data is generated by 7 cameras and 3 lidars. Additionally, the size of the dataset is the biggest: it consists of more than 1000 hours of movement images in 16K miles driving from various traffic agents including cars, pedestrians, and cyclists with frequency of 10Hz. Compared with the other datasets, the Lyft dataset contains the high-dimensional semantic map to provide context about traffic agents and their motion. Figure 2-11 shows the semantic map sample in Lyft dataset. The map features over 4,000 manually annotated semantic elements, including lane segments, pedestrian crosswalks, stop signs, parking zones, speed bumps, and speed humps. Taking advantage of the road network information provided by the semantic map, the team can do further investigation and research like route planning to assist the drivers.

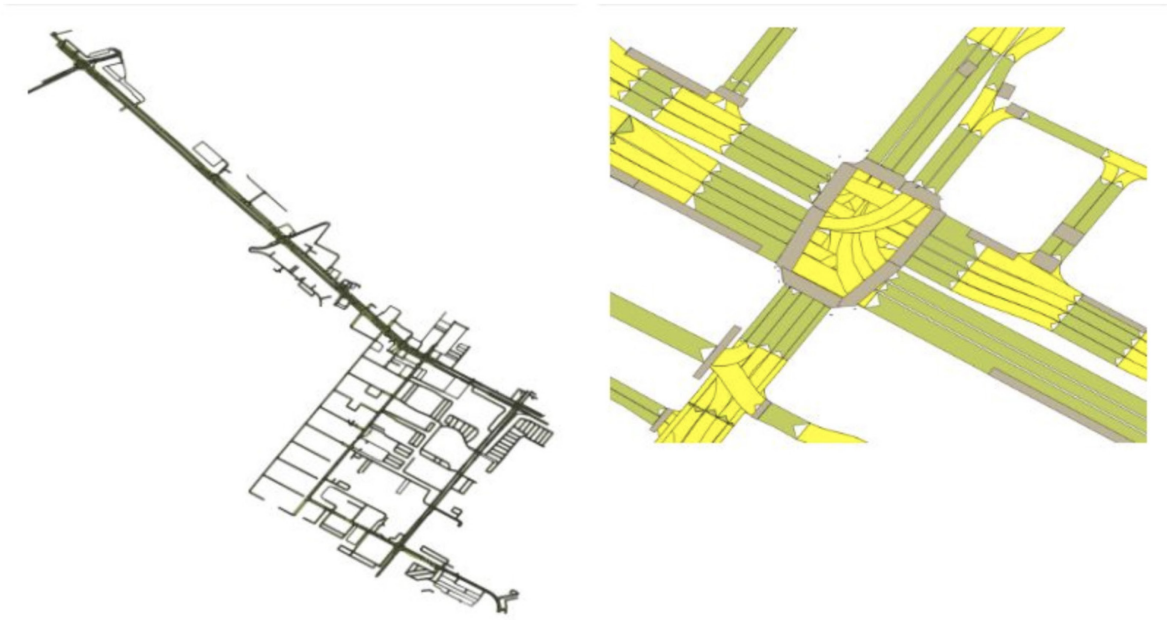


Figure 2-11. Semantic Map Sample in Lyft Dataset

Like the previous two datasets, Lyft dataset can provide large amount of high-quality data captured by 7 cameras and 3 lidars in various angles. Additionally, it provides a semantic map as the reference for every trip. Therefore, it is worth to try the dataset in the project.

Section 3 System Design

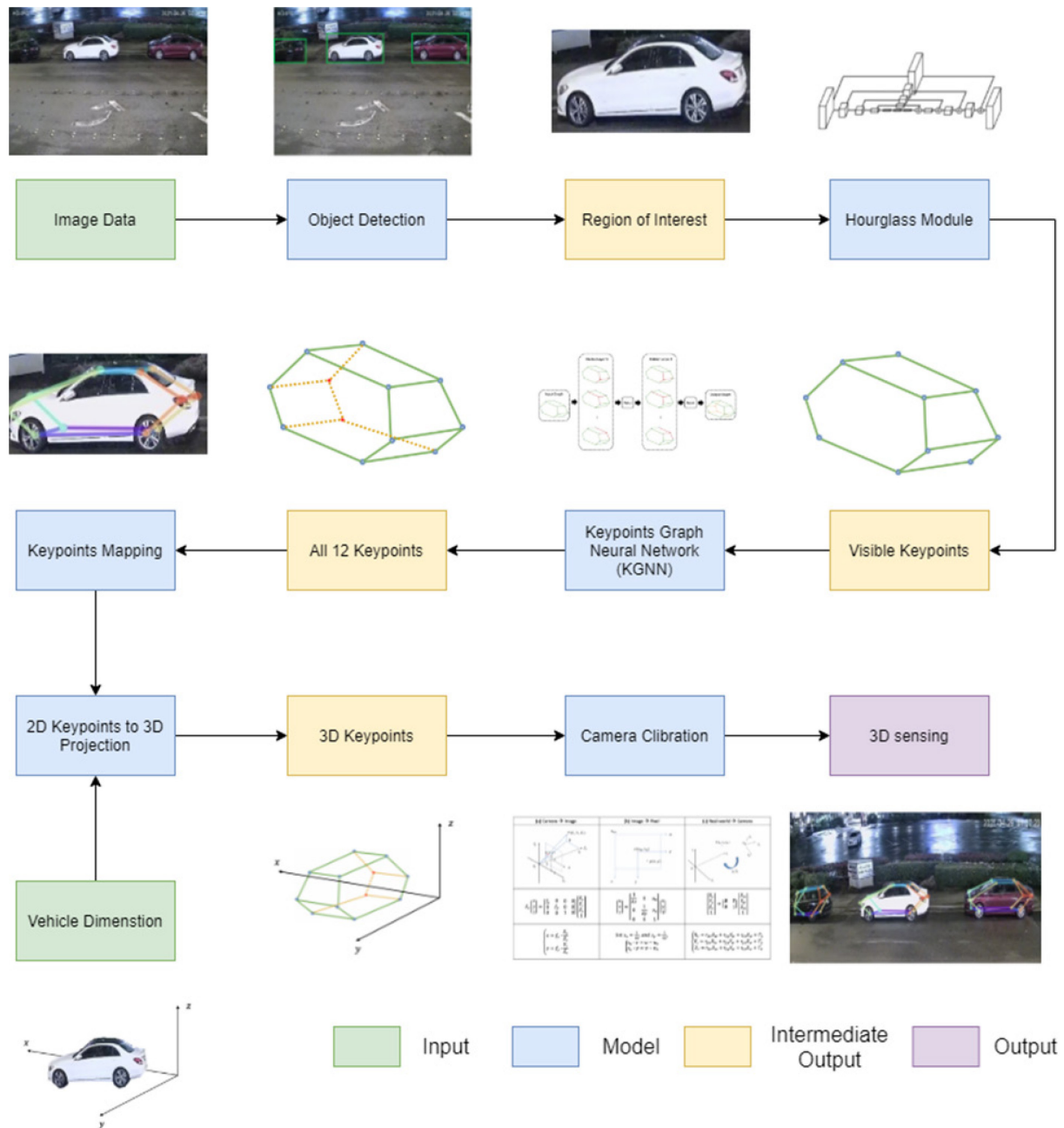


Figure 3-1 The Structure Design of the System

The system can project the 2D pixel coordinate to 3D real-world coordinate by matching the car key points and the real car dimension. The report introduces the system in three steps. Firstly, a Region-based Convolutional Neural Network (RCNN) is applied to the original image/video data for vehicle detection and classification. Then, the ROI extracted by RCNN is input to the hourglass network (Newell et al., 2016) and graph neural network (Wu et al., 2020) (GNN) to estimate all key points. Finally, based on the classification

results obtained from RCNN, 3D-SISS matches the 2D key points with average 3D vehicle dimensions to get all demanded parameter matrices for coordinate projection. Additionally, a self-learning algorithm is introduced in 3D-SISS to update the parameter matrices for better performance with the increasing number of passing vehicles.

- **Vehicle Detection and Classification:** Any object detection algorithm like SSD, YOLO, Mask-RCNN can be used in the system to realize vehicle detection. The input of the step is the image data collected by the surveillance camera system, and the output contains the region of interest (ROI) and the classification result (category with the highest confidence score).
- **Visible Keypoints Detection and Prediction:** The output ROI will work as the input in the step. Hourglass network is employed in the algorithm for vehicle 2D visible keypoints detection. Then the paper takes advantage of Keypoint Graph Neural Network (2D-KGNN) (Reddy et al., 2019) to predict the occluded keypoints based on the detected visible keypoints. The output of the step is the pre-defined vehicle keypoints (including visible and invisible keypoints).
- **Automatic 2D Keypoints to 3D Projection:** There are two inputs of the step: 1) vehicle category determined by object detector; 2) 12 vehicle keypoints determined by Hourglass Network and 2D-KGNN. Based on the vehicle category, the mean shape and the corresponding standard deviation of the vehicle are used to project 2D keypoints to 3D. Then, the camera calibration matrix is applied to the 3D keypoints to realize automatic camera calibration.

Subsection 3.1 Vehicle Detection and Classification

Object Detection and Classification is the most basic work in the CV field. In Common, present object detection algorithms can be divided into two parts: ROI detection and classification, which are needed in 3D-SISS. As a result, nearly all the existing detectors can be used in 3D-SISS for vehicle detection. However, 3D-SISS targets on 3D structural information sensing, which has high requirements on detection accuracy. As a result, the paper uses the Region-based Convolutional Neural Network (RCNN) architecture (Girshick et al., 2014) proposed in 2014, whose efficiency and accuracy have been proved by many studies (He et al., 2017). This paper retrains the model to a particular category of target object (i.e., vehicle) with human supervision. Because the vehicle detection algorithm is the most basic network in computer vision (CV), the details of the vehicle detection will not be presented in the report.

Subsection 3.2 Vehicle Keypoints Detection and Prediction

The paper uses hourglass neural network (HGNN) (Newell et al., 2016) for visible vehicle keypoints estimation. HGNN is used for human pose estimation in previous works (Li et al., 2019) vector, zhu2019exploring, xu2021graph} and their results has proved that HGNN has great performance on human keypoints (i.e., joints) detection. The idea of HGNN is motivated by the demands for feature extraction in each scale level. While local information is essential for identifying features, a final keypoints estimation

requires a coherent understanding of the full entity. In human pose estimation, the person's orientation, the arrangement of their limbs, and the relationships of adjacent joints are among the many cues that are best recognized at different scales in the image. HGNN is a simple, minimal design that has the capacity to capture all of these features and bring them together to output pixel-wise predictions. The paper replant HGNN into the transportation community for vehicle keypoints estimation based on the similar idea.

The architecture of Fourth order HGNN used in the paper is shown in Figure 3-2. The green rectangles indicate the down sampling operation which is aimed to reduce the picture from high resolution to low resolution. The red rectangles indicate the up-sampling operation which is aimed to increase the down-sampled image to original resolution. Finally, the blue rectangle indicates the residual module (see Figure 3-3), which is used for feature extraction. It is obvious that HGNN is a symmetric neural network, which can be divided into two parts: Bottom-up part and Top-down part. In the Bottom-up part, the paper uses max pooling as the down sampling method to reduce the image resolution four times. With the decrease of resolution, global features of vehicles are much easier to be captured, while the local detail features are blurred. At each max pooling step, the network branches off and applies more convolutions at the original pre-pooled resolution. After reaching the lowest resolution, the paper follows the process of (Tompson et al., 2014) and uses nearest neighbor interpolation for up sampling. Therefore, in the top-down part, the network does up sampling process and integrates the features across scales.

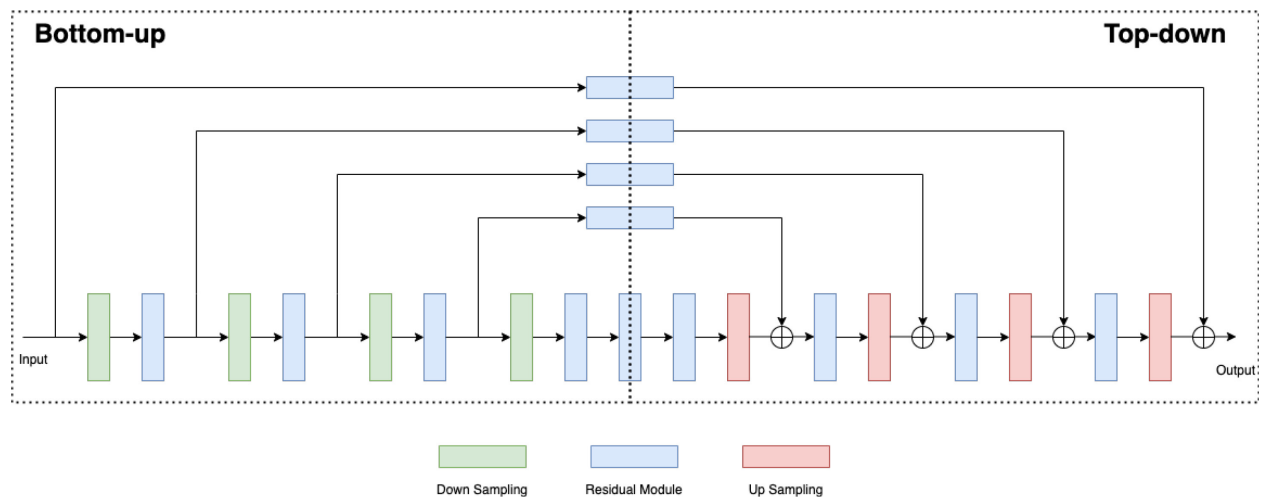


Figure 3-2 Fourth-order Hourglass Module Architecture

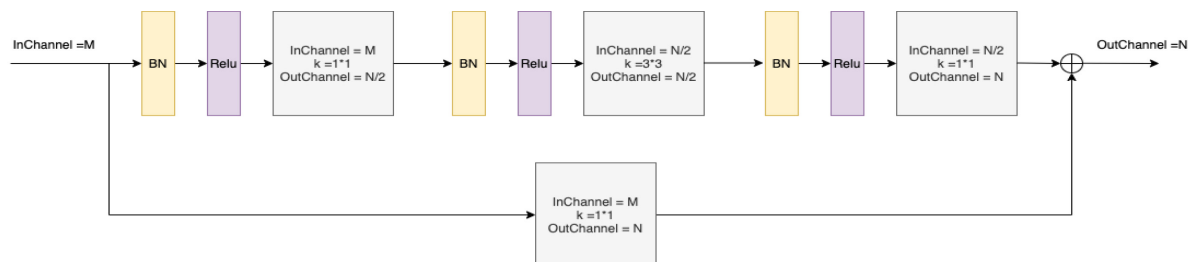


Figure 3-3 Residual Module Architecture

The input of the HGNN is the vehicle image extracted by RCNN and the output is the heat map which can show all the visible keypoints estimated. The loss function used in the network is Minimal Square Error (MSE) which is aimed to minimize the distance between the detected keypoints and labeled keypoints.

In the previous research work, 36 keypoints are predefined for accurate 3D vehicle reconstruction, however, in our case, we only select 12 of 36 keypoints of vehicles in the network training. Compared with 3D sensing technologies deployed on vehicles for autonomous driving, we do need location and structural information rather than accurate shape or rebuild the vehicle model. Comparing with the accuracy, processing efficiency is more important for 3D-SISS. There are thousands of images transmitted from surveillance systems all over the city every second. If 3D-SISS cannot process them with high efficiency, there will be serious delays in the system. As a result, based on multiple tests, the team selected 12 keypoints to represent the location and structural information about the vehicle.

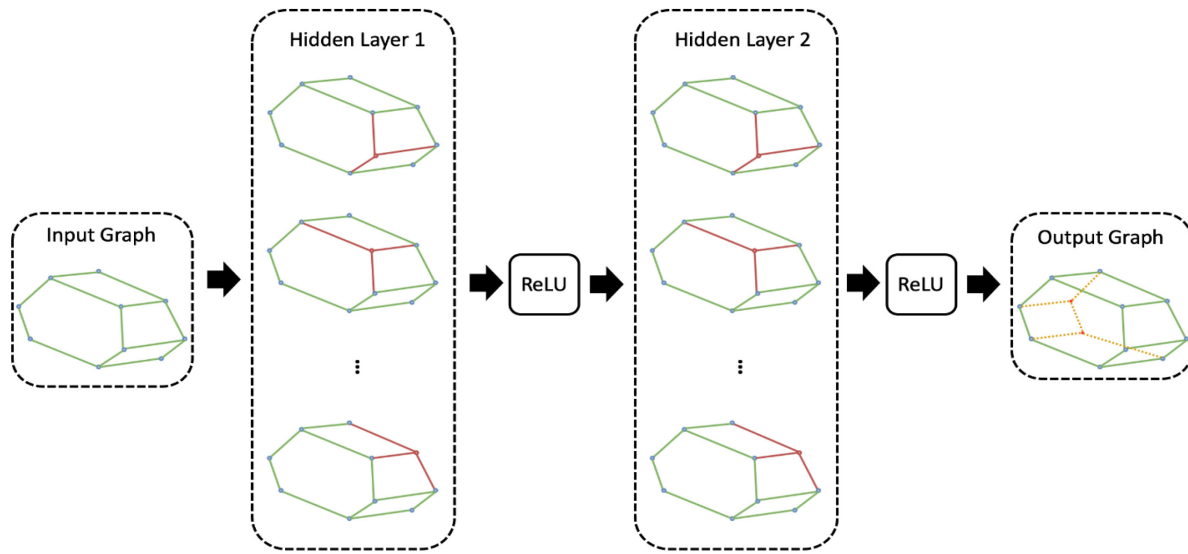


Figure 3-4 Structure of Graph Neural Network (GNN) for Invisible Car Keypoints Prediction

To predict the invisible keypoints, a 2D keypoint graph network (2D-KGNN) is introduced in the network to locate all keypoints from the keypoint heatmap generated by HGNN in the previous step. The report predefined 12 car keypoints when labeling the data. Therefore, based on the input heat map, we can categorize the 12 keypoints into two sets: visible keypoints set v and invisible keypoints set w . We denote the vertex of the graph as $V = (v_1, v_2, \dots, v_{12})$ for 12 predefined keypoints. The relationship between all nodes is encoded in the edge $E_{ij} = \{v_i, v_j\}$, where:

$$E_{ij} = \begin{cases} 1, & \text{if } i \in v \text{ and } j \in w \\ 0, & \text{otherwise} \end{cases} \quad (3-1)$$

Based on Eq. (3-1), we can convert the heatmap into a graph by encoding the location and confidence of each keypoint into a node feature. The feature for keypoint i , can be more formally represented as $v_i = \{x_i, y_i, c_i, t_i\}$, where x_i and y_i indicate the location of the keypoints in image coordinate; c_i represents the confidence score of the keypoints estimation which get from previous step; and t_i is defined as the type of the keypoint. To predict the underlying keypoints, we use the GNN to predict the latent graph structure. Figure 3-4 shows the structure of GNN. In the network, two hidden layers are used to predict the invisible edges. The process is modeled as $q(V) = \text{softmax}(f_{enc}(V))$ where $f_{enc}(V)$ is a GNN acting on the fully connected graph produced from the heatmaps. The edge loss for the process is the cross-entropy loss between the predicted edges and the ground truth edges, given in Eq. (3-2). $\varepsilon_{i,j}^l$ indicates the visibility statistics for each edge computed from the labeled keypoints.

$$Loss = \sum \varepsilon_{i,j} \log(\varepsilon_{i,j}^l) \quad (3-2)$$

Subsection 3.3 Automatic 2D Keypoints to 3D Projection

The last part of the system is to project the information we get from car key points detection to 3D coordinate. To project the image coordinate to real world coordinate, both internal and external information about the surveillance camera is necessary. As a result, in the paper, we divide the process into two parts: internal camera parameters estimation and external camera parameters estimation. In the first part, we project the 2D keypoints information in 2D pixel coordinate to 3D coordinate with the focus of the camera as the origin (i.e., camera coordinate) (Ke et al., 2017). Then, in the second part, based on the location of the camera, we transform the information in camera coordinate to 3D real-world coordinate.

The projection process has three components, consisting of lens distortion measurement, internal camera parameters determination and external camera parameters determination. The three components are illustrated in Figure 3-5. Since the camera is placed near the roadway, which is considered close enough to the detected vehicle, the lens distortion measurement is neglected at this stage. In the camera internal parameter calibration, the parameters (f_x, f_y) are focal length in pixels and (o_x, o_y) are optical centers in pixels. In this way, the overall process of camera calibration can be interpreted as converting the pixel in an image (x, y) to the coordinate (X_c, Y_c, Z_c) in a camera coordinate system, and then to the coordinate (X_w, Y_w, Z_w) in the real-world coordinate system. In this section, the coordinate of in the real-world coordinate system is denoted as characters with subscript w , and the coordinates in a camera coordinate system is denoted as characters with subscript c .

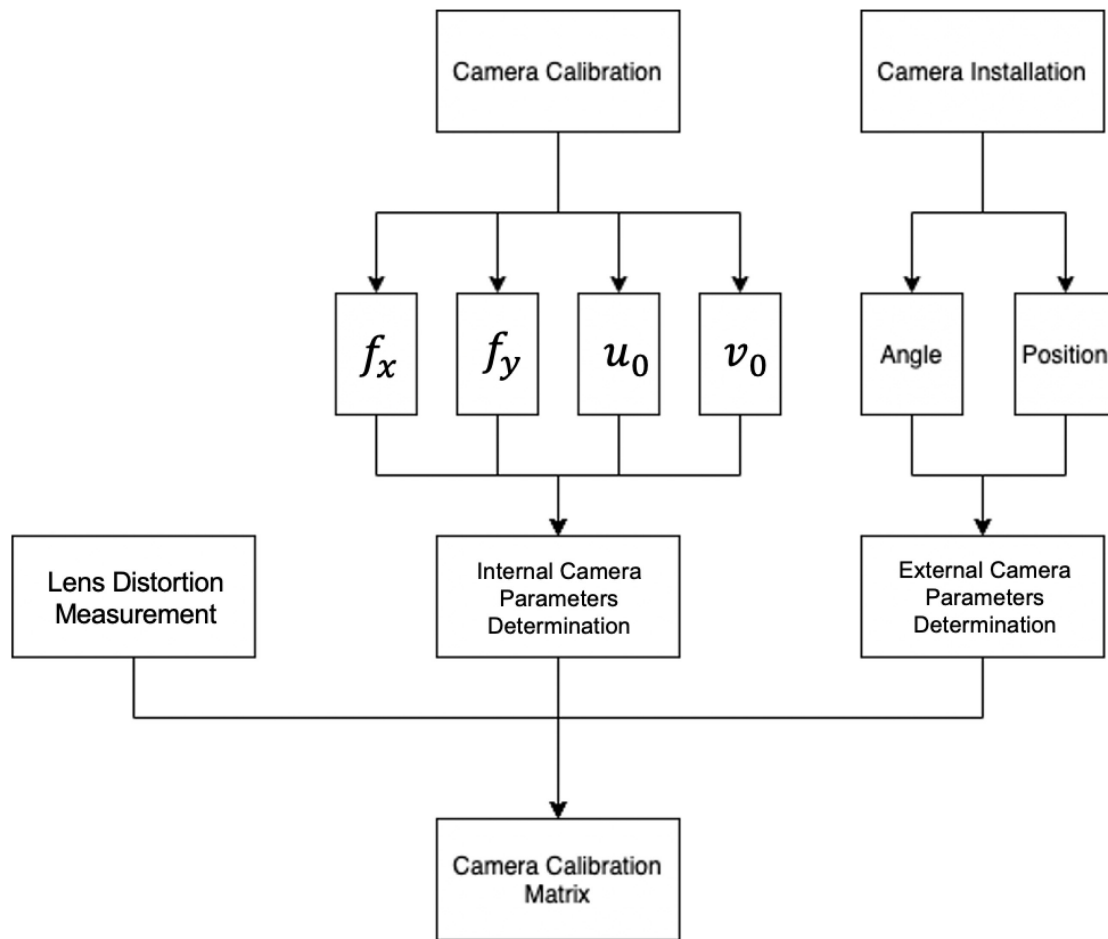


Figure 3-5 Structure of 2D to 3D Projection

External Camera Parameter Calibration

The external camera parameters calibration is a process to map a point in the real-world coordinate system to the camera coordinate system, which needs to calculate two matrices, i.e. rotate matrix (R) and offset matrix (T). Figure 3-6 demonstrates the relationship between the camera coordinate system and the real-world coordinate system.

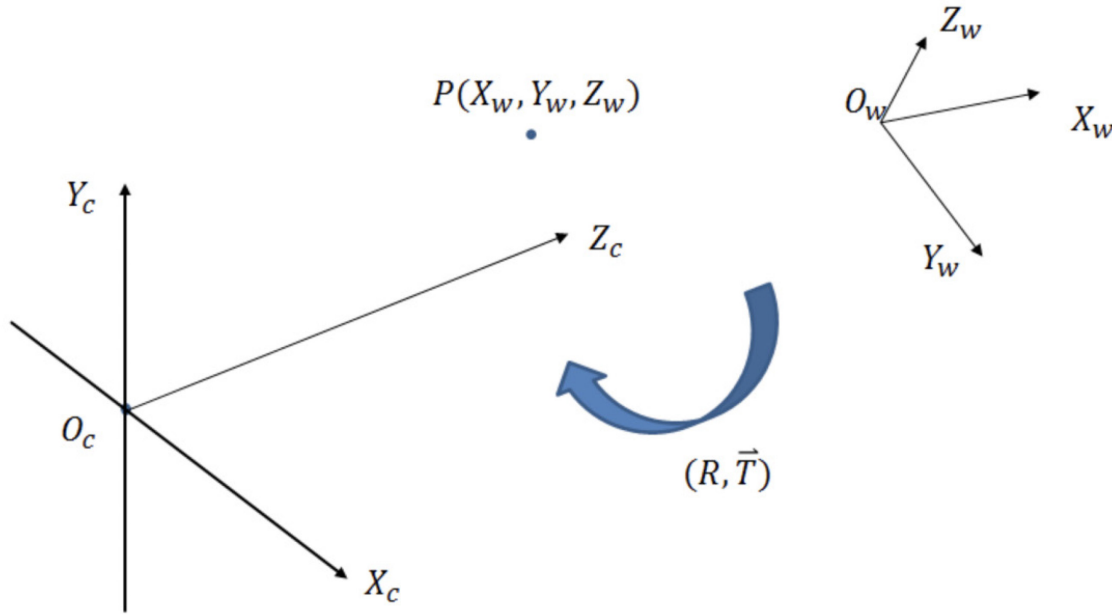


Figure 3-6 Convert between the real-world coordinate system and the camera coordinate system

Since the two coordinate systems have three dimensions, there are three possible rotations between the two coordinate systems. The rotation in one of the dimensions can be calculated using the following equation:

$$\begin{bmatrix} x \\ y \\ z \end{bmatrix} = \begin{bmatrix} \cos \theta & -\sin \theta & 0 \\ \sin \theta & \cos \theta & 0 \\ 0 & 0 & 1 \end{bmatrix} \begin{bmatrix} x' \\ y' \\ z' \end{bmatrix} = R_1 \begin{bmatrix} x' \\ y' \\ z' \end{bmatrix} \quad (3-3)$$

Where, θ is the included angle, and R_1 is the rotation matrix in the desired dimension.

Following the same way, the rotation matrices (R_2, R_3) of the other two dimensions can be calculated and the rotation matrix $R = R_1 R_2 R_3$. As for the offset matrix, T , it just measures the offset of the distance of the two original points of the two coordinate systems. Thus, the relationship between the two-coordinate system can be written in a vector form as the following equation

$$\begin{bmatrix} X_c \\ Y_c \\ Z_c \\ 1 \end{bmatrix} = \begin{bmatrix} R & T \\ \mathbf{0} & 1 \end{bmatrix} \begin{bmatrix} X_w \\ Y_w \\ Z_w \\ 1 \end{bmatrix} \quad (3-4)$$

Internal Camera Parameters Calibration

The internal camera parameter calibration targets to convert the camera-based coordinate to an image coordinate. Figure 3-7 shows the relationship between a point in the camera coordinate system $P(X_c, Y_c, Z_c)$ and a corresponding point in the image $p(x, y)$.

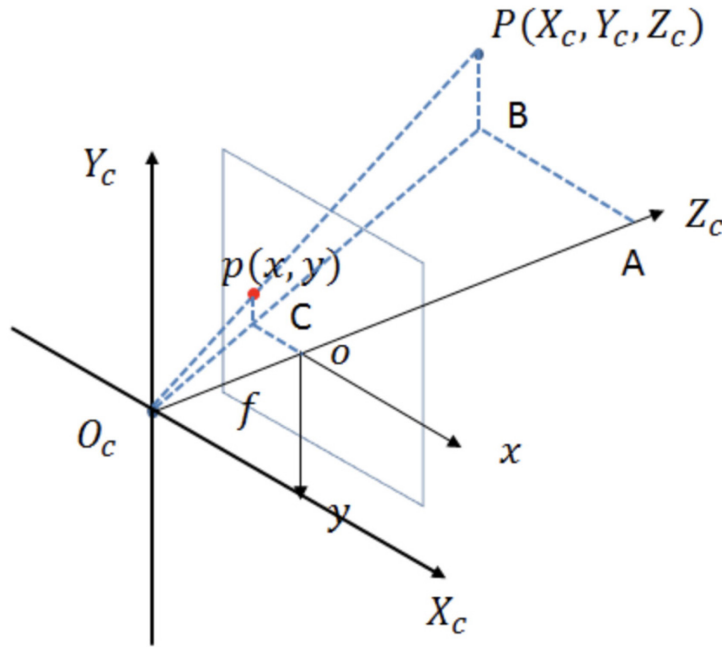


Figure 3-7 Convert between the camera coordinate system and the image coordinate system.

The mapping from the camera coordinate system to the image coordinate system is a mapping with dimension reduction from 3D to 2D, where the process can be characterized as

$$Z_c \begin{bmatrix} x \\ y \\ 1 \end{bmatrix} = \begin{bmatrix} f & 0 & 0 & 0 \\ 0 & f & 0 & 0 \\ 0 & 0 & 1 & 0 \end{bmatrix} \begin{bmatrix} X_c \\ Y_c \\ Z_c \\ 1 \end{bmatrix} \quad (3-5)$$

Where, f is the focal length.

Since the pixel coordinate has bias to the image coordinate, as shown in Figure 3-8. A minor offset adjustment is needed as follows

$$\begin{bmatrix} u \\ v \\ 1 \end{bmatrix} = \begin{bmatrix} \frac{1}{dx} & 0 & u_0 \\ 0 & \frac{1}{dy} & v_0 \\ 0 & 0 & 1 \end{bmatrix} \begin{bmatrix} x \\ y \\ 1 \end{bmatrix} \quad (3-6)$$

Where, (u_0, v_0) is the center of the image, and dx, dy are the numbers of pixels in each row and column in the image.

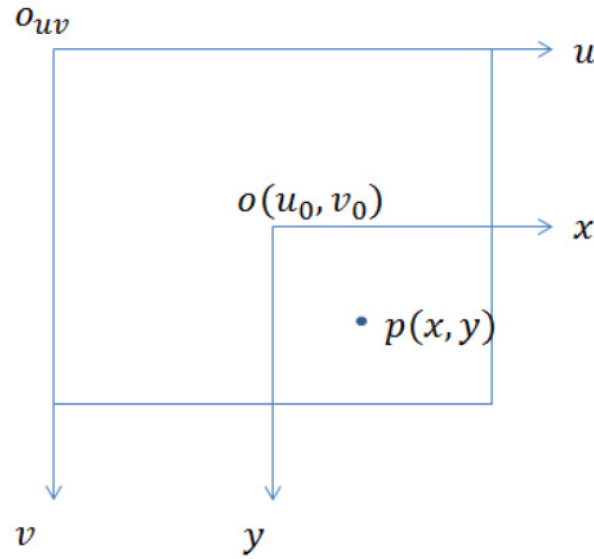


Figure 3-8 Bias between image coordinate and pixel coordinate.

2D Pixel Coordinate to 3D real-world Coordinate

By combining the converting/mapping process, the relationship between the coordinate in the real-world and the coordinate in the image can be described as follows:

$$Z_c \begin{bmatrix} u \\ v \\ 1 \end{bmatrix} = \begin{bmatrix} \frac{1}{dx} & 0 & u_0 \\ 0 & \frac{1}{dy} & v_0 \\ 0 & 0 & 1 \end{bmatrix} \begin{bmatrix} f & 0 & 0 & 0 \\ 0 & f & 0 & 0 \\ 0 & 0 & 1 & 0 \end{bmatrix} \begin{bmatrix} X_c \\ Y_c \\ Z_c \\ 1 \end{bmatrix} = \begin{bmatrix} f_x & 0 & 0 & 0 \\ 0 & f_y & 0 & 0 \\ 0 & 0 & 1 & 0 \end{bmatrix} \begin{bmatrix} R & T \\ \mathbf{0} & 1 \end{bmatrix} \begin{bmatrix} X_w \\ Y_w \\ Z_w \\ 1 \end{bmatrix} \quad (3-7)$$

Where, u, v are the pixel location in the image.

In this way, given a pixel (u, v) , the coordinate of the object in the real-world coordinate system can be calculated. One problem still exists that how to determine the Z_c , which can be considered as the depth information of the detected object with respect to the camera. In this project, the sensing environment is

fixed, and our desired vehicle location is exactly located on the ground. Thus, the Z_w equals to zero. The parameters (f_x, f_y, R, T) can be acquired in the calibration process. In this way, if we can find a point with $(X_w, X_y, X_z = 0)$ on the ground in the real-world scenario, the value of Z_c can be calculated. With Z_c calculated, it is very easy to estimate the coordinate of any point on the ground (X_w, X_y) given any image pixel (u, v) .

Camera Parameters Solution

To solve all the necessary parameters in the system, the report rewrites the previous equations in the following format:

Rewrite Eq. (3-4) for real-world coordinates to camera coordinates:

$$\begin{bmatrix} X_c \\ Y_c \\ Z_c \end{bmatrix} = \begin{bmatrix} r_{xx} & r_{xy} & r_{xz} \\ r_{yx} & r_{yy} & r_{yz} \\ r_{zx} & r_{zy} & r_{zz} \end{bmatrix} \begin{bmatrix} X_w \\ Y_w \\ Z_w \end{bmatrix} + \begin{bmatrix} T_x \\ T_y \\ T_z \end{bmatrix}$$

$$X_c = r_{xx}X_w + r_{xy}Y_w + r_{xz}Z_w + T_x$$

$$Y_c = r_{yx}X_w + r_{yy}Y_w + r_{yz}Z_w + T_y$$

$$Z_c = r_{zx}X_w + r_{zy}Y_w + r_{zz}Z_w + T_z$$

(3-8)

Rewrite Eq. (3-5) for camera coordinates to image coordinates:

$$\begin{bmatrix} X \\ Y \end{bmatrix} = \frac{f}{Z_c} \begin{bmatrix} X_c \\ Y_c \end{bmatrix}$$

$$X = f \cdot \frac{r_{xx}X_w + r_{xy}Y_w + r_{xz}Z_w + T_x}{r_{zx}X_w + r_{zy}Y_w + r_{zz}Z_w + T_z}$$

$$Y = f \cdot \frac{r_{yx}X_w + r_{yy}Y_w + r_{yz}Z_w + T_y}{r_{zx}X_w + r_{zy}Y_w + r_{zz}Z_w + T_z}$$

(3-9)

Rewrite Eq. (3-6) for image coordinates to pixel numbers:

$$\begin{bmatrix} u \\ v \end{bmatrix} = \begin{bmatrix} s_x \cdot X + u_o \\ s_y \cdot Y + v_o \end{bmatrix}$$

$$s_x X = u - u_o$$

$$Y = v - v_o$$

(3-10)

If we select a set of points $P = \{p_1, p_2, \dots, p_n\}$ in the image. For each point p_i , with $X_{wi}, Y_{wi}, Z_{wi}, u_i, v_i, u_o, v_c$ known, we can get $s_x X_i$ and Y_i from Eq. (3-9). Then, relationships between $s_x X_i$ and Y_i and X_{wi}, Y_{wi}, Z_{wi} can be established as follows:

$$s_x X_i = s_x f \cdot \frac{r_{xx}X_{wi} + r_{xy}Y_{wi} + r_{xz}Z_{wi} + T_x}{r_{zx}X_{wi} + r_{zy}Y_{wi} + r_{zz}Z_{wi} + T_z}$$

(3-11)

$$Y_i = f \cdot \frac{r_{yx}X_{wi} + r_{yy}Y_{wi} + r_{yz}Z_{wi} + T_y}{r_{zx}X_{wi} + r_{zy}Y_{wi} + r_{zz}Z_{wi} + T_z}$$

(3-12)

Dividing the Eq. (3-11) and Eq. (3-12), we can get:

$$\frac{s_x X_i}{Y_i} = s_x \cdot \frac{r_{xx}X_{wi} + r_{xy}Y_{wi} + r_{xz}Z_{wi} + T_x}{r_{yx}X_{wi} + r_{yy}Y_{wi} + r_{yz}Z_{wi} + T_y}$$

(3-13)

Assume $X'_i = s_x X_i, Y'_i = Y_i$, Eq. (3-13) can be expressed as:

$$\frac{X'_i}{Y'_i} = s_x \cdot \frac{r_{xx}X_{wi} + r_{xy}Y_{wi} + r_{xz}Z_{wi} + T_x}{r_{yx}X_{wi} + r_{yy}Y_{wi} + r_{yz}Z_{wi} + T_y}$$

(3-14)

To simplify the Eq. (3-14), we set the origin of the object coordinate planes on the ground. T_y is set to be nonzero by putting the origin of the object world coordinate away from the Y_c axis of the camera coordinate system. Thus, rearranging Eq. (8) yields:

$$Y'_i X_{wi} \frac{s_x r_{xx}}{T_y} + Y'_i Y_{wi} \frac{s_x r_{xy}}{T_y} + Y'_i Z_{wi} \frac{s_x r_{xz}}{T_y} + Y'_i \frac{s_x T_x}{T_y} - X'_i X_{wi} \frac{r_{yx}}{T_y} - X'_i Y_{wi} \frac{r_{yy}}{T_y} - X'_i Z_{wi} \frac{r_{yz}}{T_y} = X'_i$$

(3-15)

Therefore, the seven unknown parameters in Eq. (3-15) are: $\frac{s_x r_{xx}}{T_y}$, $\frac{s_x r_{xy}}{T_y}$, $\frac{s_x r_{xz}}{T_y}$, $\frac{s_x T_x}{T_y}$, $\frac{r_{yx}}{T_y}$, $\frac{r_{yy}}{T_y}$, and $\frac{r_{yz}}{T_y}$.

The establishment of seven equations from the seven non-collinear points gives us a platform on which to solve these unknowns. According to Eq. (3-15), for each calibration point i with (X_{wi}, Y_{wi}, Z_{wi}) as the 3D world coordinate and (X'_i, Y'_i) as the image coordinate, we can set up the following linear equation:

$$X'_i = [Y'_i X_{wi}, Y'_i Y_{wi}, Y'_i Z_{wi}, Y'_i, -X'_i X_{wi}, -X'_i Y_{wi}, -X'_i Z_{wi}] \begin{bmatrix} \frac{s_x r_{xx}}{T_y} \\ \frac{s_x r_{xy}}{T_y} \\ \frac{s_x r_{xz}}{T_y} \\ \frac{s_x T_x}{T_y} \\ \frac{r_{yx}}{T_y} \\ \frac{r_{yy}}{T_y} \\ \frac{r_{yz}}{T_y} \end{bmatrix}$$

(3-16)

Then, when we apply this equation to a real scenario, a linear equation set can be set up and the seven coefficients can be determined for 3D vehicle localization. Therefore, based on the 12 car keypoints we detected in the previous steps, there are still five freedom degrees to solve out all seven necessary parameters.

Section 4 Prototype Development

Subsection 4.1 MUST Sensor Prototype Design

In the project, an IoT Edge Computing device, Mobile Unit for Sensing Traffic (MUST) sensor, will be developed and deployed in practice to realize 3D sensing. MUST is an advanced traffic sensor developed by the research team at the University of Washington (UW) Smart Transportation Applications and Research Laboratory (STAR Lab). MUST can work as the intelligent control center in the area with comprehensive sensing capabilities (camera, Wi-Fi/Bluetooth, environmental sensor, etc.), real-time data analysis/edge computing, and communication functions.

In the project, the team takes advantage of MUST sensor, an advanced traffic sensing unit installed on the roadside for the traffic detection, to implement the 3D vehicle detection. Therefore, to build up the prototype system, a customized MUST sensor is necessary. Therefore, in the project, the team has modified the structures of MUST sensors to ensure it can fit the demands of the project better. Figure 4-1 shows the customized design of the MUST sensor. Figure 4-2 shows the MUST sensor rendering and Figure 4-3 shows the produced MUST sensor.

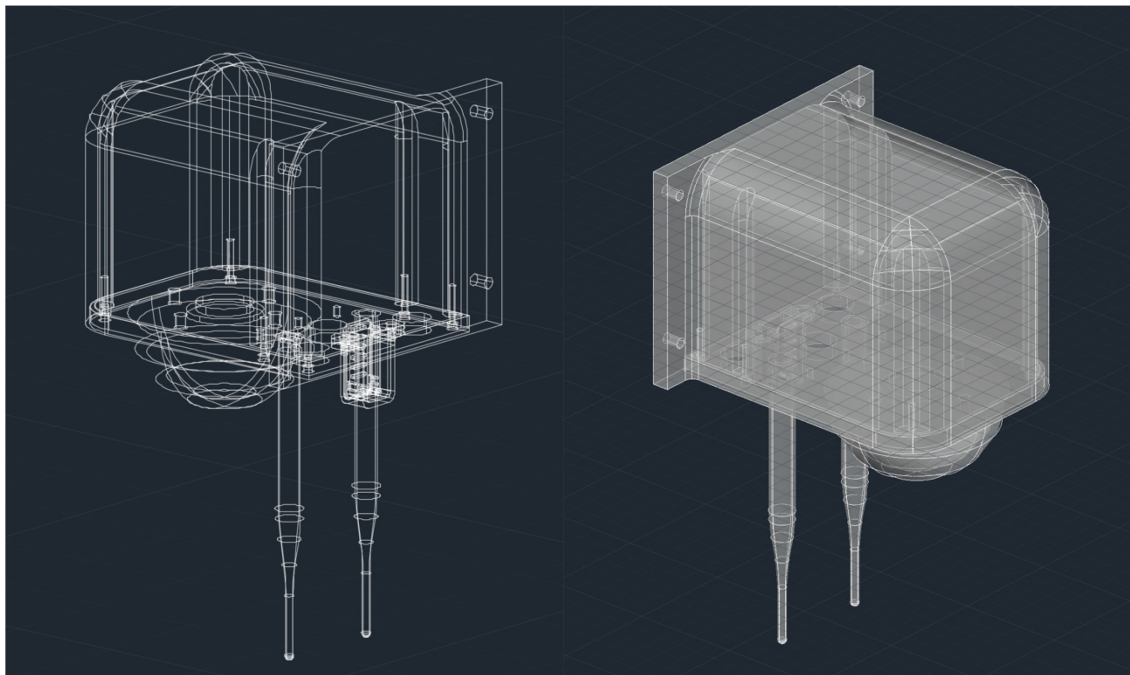


Figure 4-1 Customized MUST Sensor Design



Figure 4-2 Customized MUST Sensor Rendering



Figure 4-3 Produced Customized MUST sensor

Subsection 4.2 Data Stream Design

Data stream is the most significant part in the prototype system. In the prototype system, the image data is collected by the camera, and then it will be transmitted to the edge device for processing. The analysis results, detection results, and original image data will be transmitted to the server for further analysis and storage. In the process, even if only one component is stuck, the stream will be broken. Therefore, the team has contributed a lot of efforts on the data stream buildup and improvements to ensure its reliability. Finally, the completed data stream is shown in Figure 4-4, and the following paragraphs go over the details in the data stream.

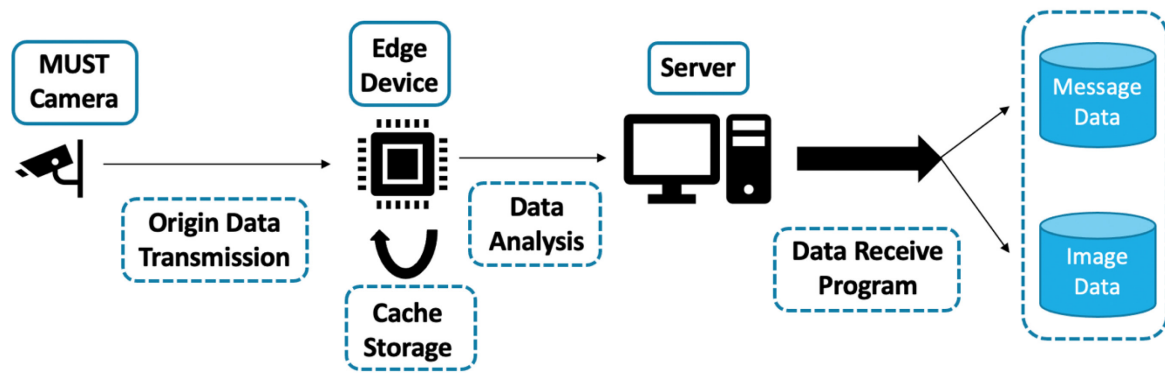


Figure 4-4 MUST Prototype System DataStream

Firstly, the communication between MUST camera and Edge Device is based on local area network (Appenzeller et al., 2004). The image captured by the camera will be divided into multiple frames (based on the camera setting) for data transmission. If the size of the frame exceeds the capacity of the buffer capacity (Villamizar et al., 1995) (Avrachenkov et al., 2002), an algorithm will be implemented to the frame and divide it into pixels (Weng et al., 1992). The data receiving program in the edge device will receive the data and reorganize the data. Then the 3D object detection algorithm in the edge device can do the analysis based on the image data. The analysis results and the origin image data can be stored in the edge device for 6 hours. The new data will cover the old data because of the limit of the storage.

Secondly, the communication between Edge Device and Server is based on the UDP protocol (Gong et al. 2014) (Ha et al., 2008). There are two kinds of data that will be transmitted from Edge Device to the Server: 1) Analysis Results; 2) Image Data. The analysis results will be stored as a string and sent to the server (i.e., the Message Data). The data receive program on the server side can receive the data string and reorganize them as a dataset (Raina et al., 2005) for storage. At present, a new csv file will be generated in the designed folder every hour to store the message data. The image data will transmit just like what we do between the MUST camera and Edge Device. The data receive program will store the image data in the designed folder and name it with the current timestamp.

Finally, the validation algorithm on the server will analyze the image data sent from the edge device to do the 3D object detection. The detection results will be used to validate the analysis results from the edge device.

Subsection 4.3 Prototype Preliminary Tests

After producing the MUST sensor, the team has conducted a series of tests to ensure its quality can fit the requirements of the project. During the period, we communicated with the factory about the quality of the

product and reproduced the MUST sensor several times. Finally, all the tests shown in Table 1 have passed. And some test figures are shown in Figure 4-5.

Table 3-1 MUST sensor quality tests

NO.	Tests	Status	Results
1	Tests of Mechanical Features	Done	Passed
2	Tests of Water-proof Capability	Done	Passed
3	Design Fitness Verification	Done	Passed
4	Temperature Control Capability Tests	Done	Passed
5	Surface Light Condition Tests	Done	Passed
6	Camera Performance Tests	Done	Passed
7	Power Switch Tests	Done	Passed
8	MUST installation Tests	Done	Passed
9	Communication Tests	Done	Passed



(a) Waterproof Tests



(b) Low Temperature Tests



(c) Electric Surges Tests

Figure 4-5 MUST Quality Tests

In the next part, the team tested some basic functions of the MUST sensor including the data streaming and vehicle detection algorithm. In Task 5, we have done some pioneer tests in the city of Bellevue to verify the performance of the algorithm. The MUST test version is installed in a curbside scenario, the most complex scenario used in the urban area. Figure 4-6 shows the vehicle detection algorithm's results. The accuracy of the detection can reach 95% in the curbside scenario, which may achieve higher accuracy in other scenarios. However, it is easy to find that in the low light conditions (e.g., night), the detection accuracy drops a lot because it is hard for CNN to extract the features. To address the problem, the team introduced a new IP

camera with night vision in the MUST sensor. The vision comparison can be shown in Figure 4-7. We haven't installed the new IP camera to the test bed, but we did some basic tests in our lab and the sample results are shown in Figure 4-7. Figure 4-8 shows the detection results in complete darkness.



Figure 4-6 Sample Practical Test Result in City of Bellevue for Vehicle Detection in Different Conditions (Daylight, Night Vision, Rainy, and Snowy)

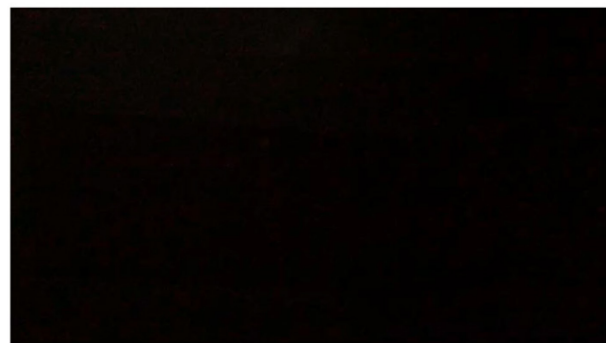


Figure 4-7 Comparison of Night Vision between IP camera (left) and USB camera (right) in the same dark room

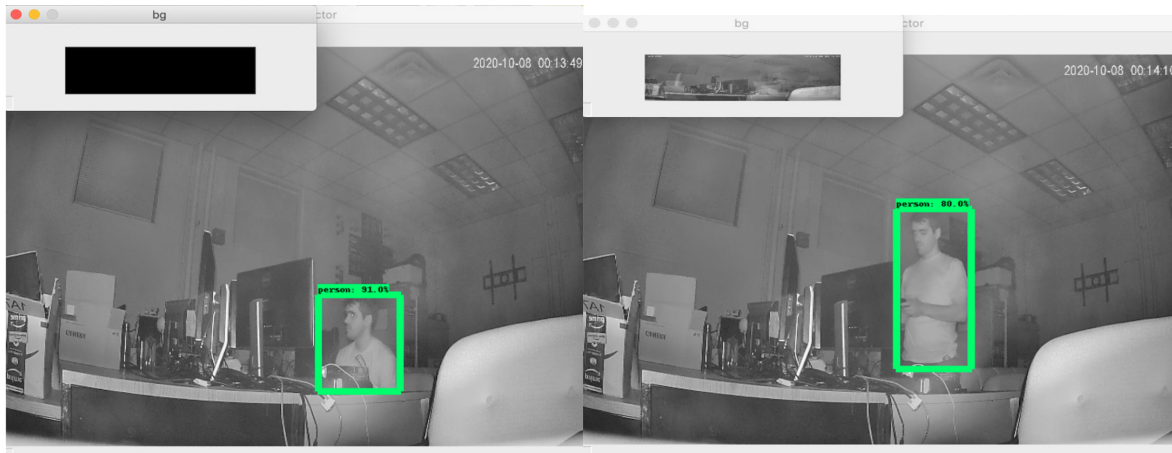


Figure 4-8 Object Detection in Complete Darkness

Section 5 Field Test

Subsection 5.1 In-lab Tests

In the section, the report demonstrates the ability of the system to realize 3D structural information sensing with the MUST sensor. Firstly, we did some preparation and In-lab tests to understand the system better. Secondly, the report introduces the experiment design and deployment in Section 5.2. Then Section 5.3 introduces the datasets we used in the system to train and test the algorithm. Fourthly, we present the performance of the algorithm on car keypoints detection in Section 5.4. Additionally, a basis analysis will be summarized based on the detection results. Finally, in Section 5.5, the projection from 2D to 3D is applied in a real-world scenario. The ground truth data is used to validate the results.

Subsection 5.1.1 First In-lab Tests

Before filed test, because of the breakout of Covid-19, the team cannot realize the field test on time. Alternatively, the team implemented an in-lab test to prepare the forthcoming field test as a preparation.

In the lab environment, the team developed a basic test bed (see Figure 5-1) based on the traditional chess board camera calibration method.

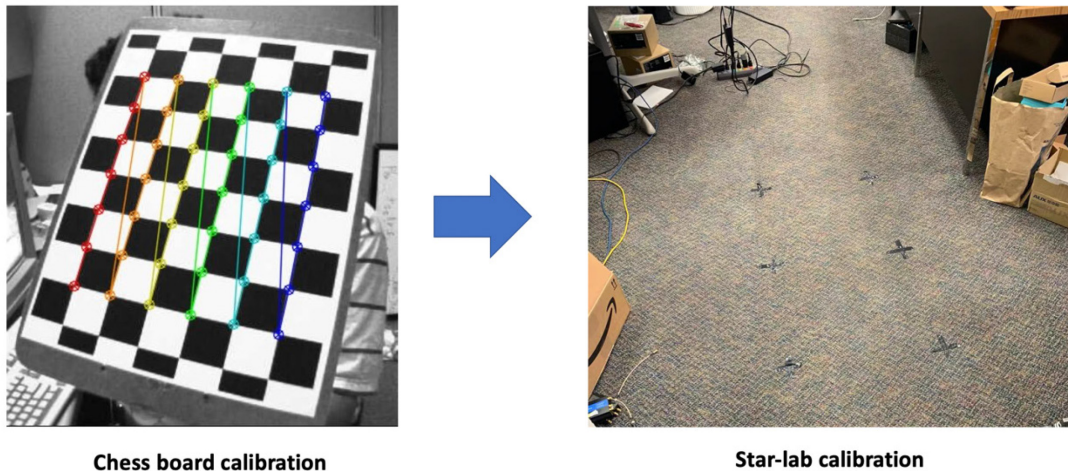


Figure 5-1 In-lab Camera Calibration Tests

The team attached 6 crosses on the floor in star lab work as the input to solve out the seven parameters. In the previous discussion, to simplify Eq. (3-13), the team set the origin of the object coordinate planes on the ground (see Figure 5-1). The seven points we select here are shown on Figure 5-2. The reason why we select a point out of the ground (p_7) is to ensure the rows' independence in the parameter matrix. We will discuss

the linear dependency of rows in the parameter matrix in the next section. In the section, we apply Eq. (3-15) to the in-lab scenario, a linear equation set can be set up as follow Eq. (5-1):

$$\begin{bmatrix} 0, 0, Y'_1 h, Y'_1, 0, 0, -X'_1 h \\ Y'_1 d, 0, 0, Y'_1, -X'_1 d, 0, 0 \\ 2Y'_1 d, 0, 0, Y'_1, -2X'_1 d, 0, 0 \\ Y'_1 d, Y'_1 d, 0, Y'_1, -X'_1 d, -X'_1 d, 0 \\ 2Y'_1 d, Y'_1 d, 0, Y'_1, -2X'_1 d, -X'_1 d, 0 \\ Y'_1 d, 2Y'_1 d, 0, Y'_1, -X'_1 d, -2X'_1 d, 0 \\ 2Y'_1 d, 2Y'_1 d, 0, Y'_1, -2X'_1 d, -2X'_1 d, 0 \end{bmatrix} \begin{bmatrix} \frac{s_x r_{xx}}{T_y} \\ \frac{s_x r_{xy}}{T_y} \\ \frac{s_x r_{xz}}{T_y} \\ \frac{T_y}{s_x T_x} \\ \frac{T_y}{r_{yx}} \\ \frac{T_y}{r_{yy}} \\ \frac{T_y}{r_{yz}} \\ \frac{T_y}{T_y} \end{bmatrix} = \begin{bmatrix} X'_1 \\ X'_2 \\ X'_3 \\ X'_4 \\ X'_5 \\ X'_6 \\ X'_7 \end{bmatrix}$$

(5-1)

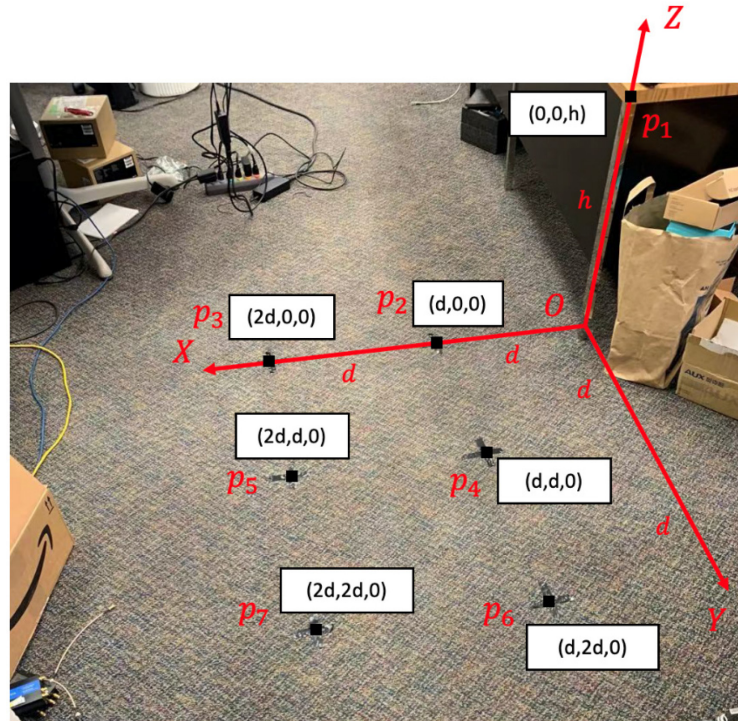


Figure 5-2 In-lab Camera Calibration Coordinates

The seven parameters can be easily solved out by Eq. (5-1) because there are six points are located on the ground, which indicates their height $z = 0$. The results can be shown in Table 5-1.

Table 5-1 Camera Calibration Results in First In-lab Tests

Parameters	Camera Calibration Results	Ground Truth
Rotation Matrix (R)	$\begin{bmatrix} 0.9966 & -0.3991 & 0.0246 \\ -0.1564 & -0.4949 & -0.8714 \\ 0.3432 & 0.8510 & -0.4609 \end{bmatrix}$	$\begin{bmatrix} 0.100 & -0.400 & 0.0250 \\ -0.1500 & -0.5000 & -0.9000 \\ 0.3500 & 0.8500 & -0.4500 \end{bmatrix}$
Translation Matrix (T)	$[0.9935 \ 1.4521 \ 1.4781]^T$	$[1.0000 \ 1.5000 \ 1.500]^T$
Scale Factor (s_x)	0.9862	1.000
Focal Length (f)	240.3520	240.0000

The results indicate the accuracy of camera calibration can reach about 90% accuracy. After a careful investigation, the team identified the human labelled points (the black points drawn on Figure 5-1 and Figure 5-2) can be the main cause for the accuracy drop. In camera calibration, the label points should reach pixel level accuracy, however, in the In-lab test, limited by the tape width, the cross sticked to the ground cannot reach such level accuracy. As a result, the camera calibration algorithm cannot estimate the camera parameters in high accuracy.

Subsection 5.1.2 Second In-lab Tests

Therefore, to improve the understanding of the algorithm, the team takes advantage of the data collected in other projects to test the algorithm in a real-world scenario. The test bed is provided by the City of Bellevue for curbside monitoring. However, because the shutter speed of the camera is low, we cannot get an image of moving vehicles at high resolution, which may cause a drop in camera calibration accuracy. In the test, the team used a standard shipping container passing by the camera (shown in Figure 5-3). Standard shipping container is a rectangle of dimension (height × width × length) that must fall in one of the four standard sizes provided by DOT. The distance between the underside of a shipping container and ground surface is assumed to be d , which is often known for specific truck types. In addition, because the origin panel must be placed on the ground, the origin O is assumed to be under p_1 .

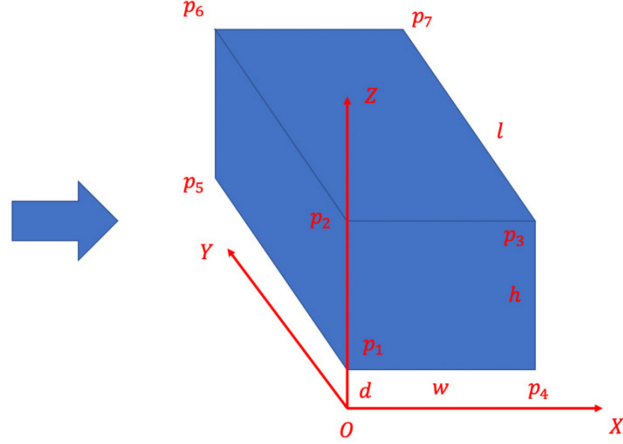


Figure 5-3 Camera Calibration Coordinates in Second In-lab Tests

When we apply Eq. (3-15) to the shipping container, the linear equation set can be written in Eq. (5-2).

$$\begin{bmatrix}
 0, 0, Y'_1 d, Y'_1, 0, 0, -X'_1 d \\
 0, 0, (h + d)Y'_2, Y'_2, 0, 0, -(h + d)X'_2 \\
 wY'_3, 0, (h + d)Y'_3, Y'_3, -wX'_3, 0, -(h + d)X'_3 \\
 Y'_4 w, 0, Y'_4 d, Y'_4, -X'_4 w, 0, -X'_4 d \\
 0, Y'_5 l, Y'_5 d, Y'_5, 0, -X'_5 l, -X'_5 d \\
 0, Y'_6 l, Y'_6 (h + d), Y'_6, 0, -X'_6 l, -X'_6 (h + d) \\
 wY'_7, lY'_7, (h + d)Y'_7, Y'_7, -wY'_7, -lY'_7, -(h + d)Y'_7
 \end{bmatrix}
 \begin{bmatrix}
 \frac{s_x r_{xx}}{T_y} \\
 \frac{s_x r_{xy}}{T_y} \\
 \frac{s_x r_{xz}}{T_y} \\
 \frac{s_x T_x}{T_y} \\
 \frac{r_{yx}}{T_y} \\
 \frac{r_{yy}}{T_y} \\
 \frac{r_{yz}}{T_y}
 \end{bmatrix}
 =
 \begin{bmatrix}
 X'_1 \\
 X'_2 \\
 X'_3 \\
 X'_4 \\
 X'_5 \\
 X'_6 \\
 X'_7
 \end{bmatrix}
 \quad (5-2)$$

It is obvious that the parameter matrix is much more complicated than Eq. (5-1). As a result, the team needs to ensure the linear independence of the rows and then set up a suitable coordinate before solving the matrix.

Firstly, we should emphasize that among the seven corner points, no three points are collinear on the image plane. During an object's motion through the view range of a video camera, there are a large number of snapshots that can be collected in different frames. In order to get an available snapshot for calibration, it is reasonable to assume that on the image plane, any two points among the seven corner points are not collinear with the origin, and no point is on X axis or Y axis.

Because the proof for the linear independence of the rows in the coefficient matrix is a pure math process, we only present the calculation for the field tests. The same method can be applied to the in-lab tests.

Let K be the coefficient matrix in Eq. (5-2), then K can be written as:

$$K = \begin{bmatrix} 0, 0, Y'_1 d, Y'_1, 0, 0, -X'_1 d \\ 0, 0, (h + d)Y'_2, Y'_2, 0, 0, -(h + d)X'_2 \\ wY'_3, 0, (h + d)Y'_3, Y'_3, -wX'_3, 0, -(h + d)X'_3 \\ Y'_4 w, 0, Y'_4 d, Y'_4, -X'_4 w, 0, -X'_4 d \\ 0, Y'_5 l, Y'_5 d, Y'_5, 0, -X'_5 l, -X'_5 d \\ 0, Y'_6 l, Y'_6 (h + d), Y'_6, 0, -X'_6 l, -X'_6 (h + d) \\ wY'_7, lY'_7, (h + d)Y'_7, Y'_7, -wY'_7, -lY'_7, -(h + d)Y'_7 \end{bmatrix} \quad (5-3)$$

Let M_i be the i^{th} row of K , it is to be shown that the necessary and sufficient condition for $\sum_{i=1}^7 a_i M_i = 0$ is that $a_i = 0$ ($i = 1, 2, \dots, 7$). The sufficiency is obvious in this case, so we move on to the necessity.

From column 2 and 6, we can get:

$$\begin{cases} a_5 lY'_5 + a_6 lY'_6 + a_7 lY'_7 = 0 \\ a_5 lX'_5 + a_6 lX'_6 + a_7 lX'_7 = 0 \end{cases} \quad (5-4)$$

Because all the values are positive, we can get:

$$a_5 l(X'_5 + Y'_5) + a_6 l(X'_6 + Y'_6) + a_7 l(X'_7 + Y'_7) = 0 \quad (5-5)$$

Because p_5, p_6, p_7 are not collinear, at least one of a_5, a_6, a_7 is zero. Due to symmetry, it suffices to take $a_5 = 0$. Now, if $a_6 \neq 0$ and $a_7 \neq 0$, it is also impossible since we assumed no two points have a linear relationship with the origin of the image coordinate. Hence, due to symmetry again, we take $a_6 = 0$, then it is obvious $a_7 = 0$.

Similarly, from column 1 and 5, we can get $a_3 = 0$ and $a_4 = 0$. And from column 3 and 4, we can get $a_1 = 0$ and $a_2 = 0$.

As a result, all the $a_i = 0$ ($i = 1, 2, \dots, 7$), which implies the linear independence of the rows of the coefficient matrix in Eq. (5-3).




































Class 1 Motorcycles		Class 7 Four or more axle, single unit	
Class 2 Passenger cars		Class 8 Four or less axle, single trailer	
			
			
			
Class 3 Four tire, single unit		Class 9 5-Axle tractor semitrailer	
			
			
Class 4 Buses		Class 10 Six or more axle, single trailer	
		Class 11 Five or less axle, multi trailer	
			
Class 5 Two axle, six tire, single unit			
		Class 12 Six axle, multi-trailer	
		Class 13 Seven or more axle, multi-trailer	
Class 6 Three axle, single unit			
			
			
			

Figure 5-4 FHWA 13 Vehicle Category Classification¹

Based on the 13-vehicle category classification chart shown in Figure 5-4, the detected standard truck should belong to classification 6. A rough estimate for d is 3.5' here, while it has only a little influence on Matrix (T_x, T_y, T_z) and reconstruction. As shown in Figure 5-5, we manually collect the seven corner points' pixel numbers and run the model. Table 5-2 shows the calculated parameters in the model.

¹ https://www.fhwa.dot.gov/policyinformation/tmguidetmg_2013/vehicle-types.cfm



Figure 5-5 Identify seven points and obtain their pixels number for calibration

Table 5-2 Camera Calibration Results in Second Lab Tests

Parameters	Camera Calibration Results
Rotation Matrix (R)	$\begin{bmatrix} 0.9966 & -0.3991 & 0.0246 \\ -0.1564 & -0.4949 & -0.8714 \\ 0.3432 & 0.8510 & -0.4609 \end{bmatrix}$
Translation Matrix (T)	$[8.0751 \ 13.8752 \ 57.1012]^T$
Scale Factor (s_x)	0.9475
Focal Length (f)	371.7290

From the results, it can be seen that there is no ground truth available in the test because it is impossible to get it from an installed surveillance camera. As a result, to validate the results obtained from the tests, the team proposed two methods to validate the results.

1) Verification Using Rotation Matrix

We haven't completely made use of the property 3) of matrix R in solving camera parameters. For property 1) and 2) of matrix R , it is clear that not all the rows or columns were taken to compute the unknowns. With the properties that have been used before, the determinant of matrix R still cannot be around the true value if it is not solved properly. Hence, the results shown in this work are convincing since the determinant of matrix R turns out to be 1.0283, which is very close to 1.

2) Verification Using the Same Shipping Container in a Different Snapshot

In the ideal situation, point 1, 2, 3 and 4 have the same Y_w coordinates, as do Point 5, 6 and 7; Point 1, 2, 5 and 6 have the same X_w coordinates, as do Point 3, 4 and 7. As we can see, Table 5-2 presents quite reasonable reconstruction results according to this principle.

In addition, as we know the true distance between every two container corner points, we can choose any two corner points and compute their reconstructed distance to see if the result is reasonable compared to the true value. For example, from the results in Table 5-2, the reconstructed distance between Point 1 and 5 is estimated to be 16.1', which is very close to its true value 16'.

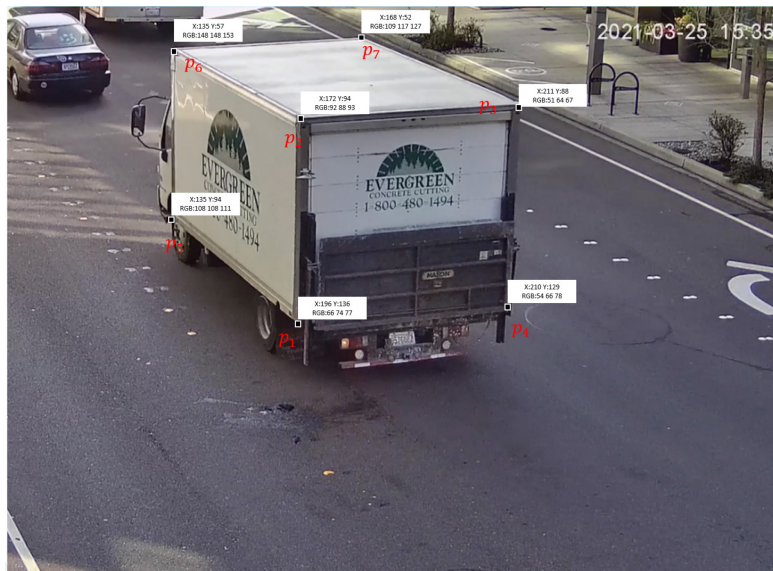


Figure 5-6 Same Truck in Another Snapshot

Subsection 5.2 Field Test Design

3D-SISS is a complex system which requires a comprehensive experiment to test its performance. Therefore, in the research, cooperating with the City of Bellevue, we installed MUST sensors (see Figure 5-7) in the City of Bellevue to test the 3D-SISS system. The location and the installation height were pre-measured in the

process for final results validation. Also, the team can adjust the focus length and the angle of the surveillance camera through the control panel. Therefore, we can get the ground truth information about the camera on both internal and external sides for camera parameter estimation performance test.



Figure 5-7 MUST Sensor Installed in Bellevue Test Bed

On the vehicle side, we prepared three vehicles with GPS devices to drive through the test to test the 3D projection performance. To simplify the result validation process, the team uses the mean location of detected 12 keypoints to represent the location of the vehicle.

Subsection 5.2 Data Collection and Training

In the project, three kinds of datasets are used in the system.

- **Car-render dataset:** In the research, we use the vehicles from ShapeNet (Chang et al., 2015) and 3D annotated by (li et al., 2017). In the previous research work (Reddy et al., 2019), 36 keypoints are predefined for accurate 3D vehicle reconstruction. As mentioned in Section 3.2, in the project only 12 keypoints are selected to represent the vehicle, which can provide enough structural information with much lower computation loads. We use 300 synthetic CAD models for training, 72 for validation and 100 for testing. We project the 3D keypoint annotations of the CAD model with visibility. We trace a ray toward the object from a pixel and check if the first intersection is close to the ground truth location to determine visibility.
- **CarFusion dataset:** CarFusion dataset is provided by (Reddy et al., 2019) (Reddy et al., 2018). The dataset contains 2.5 million images from 5 crowded traffic intersections. The two papers sampled about 53000 images at uniform intervals from each video. Approximately, 100000 cars detected in

these images were annotated with 12 keypoints each. Each annotation contains the visible and occluded keypoint locations on the car. We selected four annotated intersections to train the network while using one intersection to test it, which split the annotation data into 36000 images for training and 17000 for testing. We further compute a 90-10 train validation split on the training data to validate our training algorithm.

- **Self-collected dataset:** As described in Section 3.1, in the experiment, the team collects three kinds of data: 1) Video data: The camera installed on the roadside is used to collect video and image data about the passing vehicles in one week. 2) Camera parameters: During the week, the team adjusted the camera angle and focus length multiple times to test if 3D-SISS can estimate the new parameters of the camera correctly. 3) GPS data: the vehicle with GPS device driving through the road and collecting GPS records to test if 3D-SISS can estimate the location of the vehicle correctly.

Subsection 5.3 Car Keypoints Detection Results

Some sample results of the car keypoints detection can be viewed in Figure 5-8. We can see that all the visible keypoints can be detected very clearly. However, for invisible keypoints prediction, some mismatches can be observed. To quantitatively evaluate the keypoints detection result, the paper introduces the Percentage of Correct Keypoints (PCK) metric. PCK metric is a well-known metric which is used for human pose detection result evaluation. The definition of PCK is described as "Detected joint is considered correct if the distance between the predicted and the true joint is within a certain threshold (threshold varies)." In our application scenario, we replant PCK into car keypoints evaluation. According to the PCK metric, a keypoint is considered correct if it lies within the radius αL of the ground truth. Here L is defined as the maximum of length and width of the bounding box and $0 < \alpha < 1$. Figure 5-9 shows the quantitative results of the car keypoints detection.

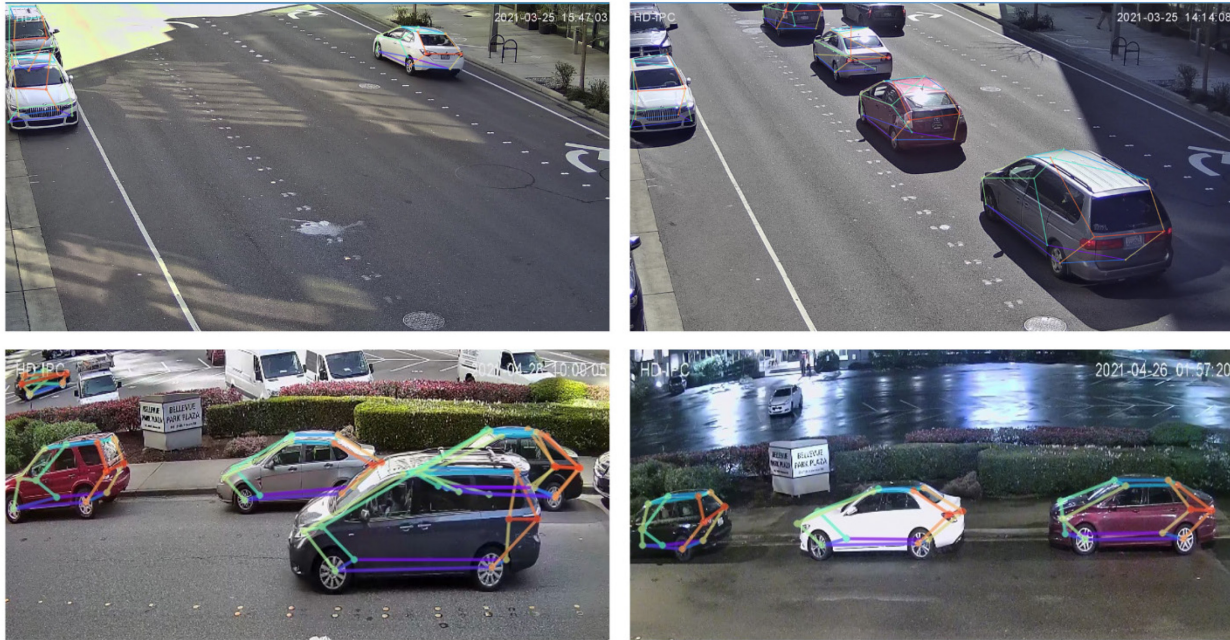


Figure 5-8 Example Results of 3D-SISS on Self-collected Data in Bellevue Test Bed

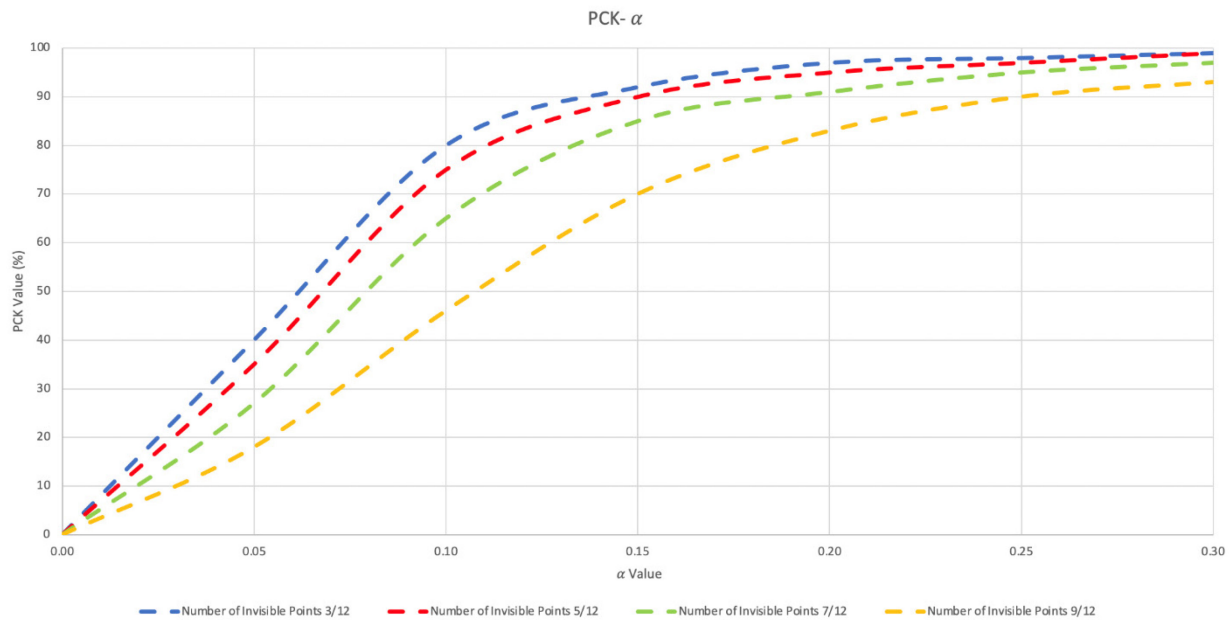


Figure 5-9 Accuracy vs α with Various Number of Invisible Car Keypoints

From Figure 5-9, the four lines indicate various invisible keypoints of the vehicle. x axis indicates the value of α and the y axis indicates the value of PCK. We can easily observe that when the number of invisible keypoints is 3, 90% detected keypoints located in the 0.15 error range. However, when the number of

invisible keypoints increases to 9, which indicates that only 3 keypoints are observed, only 70% detected keypoints located in the 0.15 error range. Therefore, the occlusion has significant impacts on the car keypoints detection. In summary, the overall car keypoints detection accuracy is acceptable, which is high enough to estimate the camera parameters and realize 3D projection.

Subsection 5.4 Camera Calibration Results

In the section, just like we have done in the Second In-lab Test, we will project the detected 2D car keypoints to 3D real-world coordinates and calculate the accurate location of the vehicle. The experiment can be divided into two parts: 1) camera parameters estimation; 2) vehicle localization.

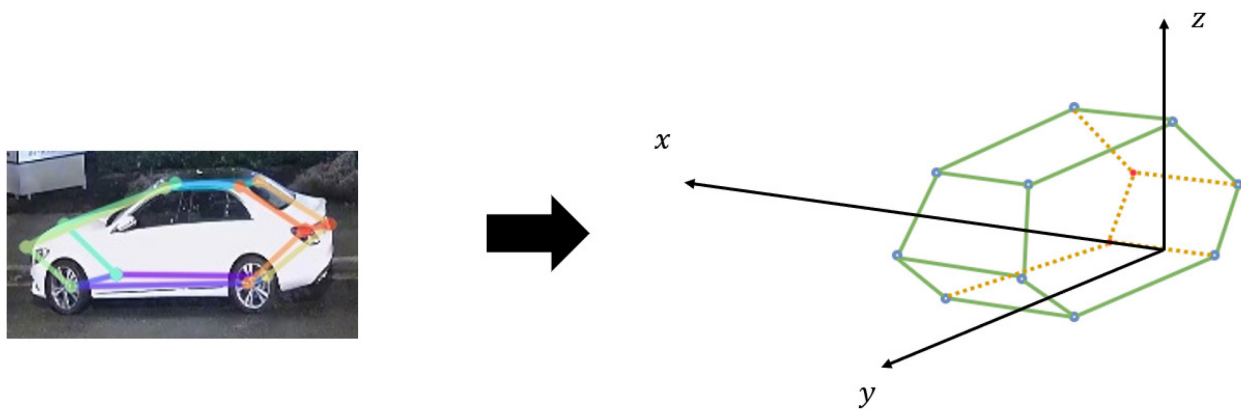


Figure 5-10 Camera Calibration Coordinates in Field Test

To simplify the process of solving the coefficient matrix, the team set the origin points on the center of rear wheels (see Figure 5-10). As a result, z value of each keypoints can be presented by their height to the ground. We can select any seven points combinations in the figure as the input to Eq. (3-15) to get a solution for the camera parameters. To demonstrate the process, the report picked up seven keypoints randomly and presented the calculation in the following paragraphs.

Firstly, to match the detected keypoints with the vehicle dimension, it is necessary to present the keypoints in 3D coordinate (see Figure 5-11). Therefore, the team divided the 12 keypoints into three planes p_1 , p_2 , and p_3 , and there are four keypoints in each plane. The size of three parallel planes can be denoted by (w, d) , where w indicates the width in x dimension and d indicates the length in y dimension. The distances between the three planes are denoted as h_{12} and h_{23} . Dihedral angles γ and θ are used to determine the related positions of the three planes. As a result, each keypoints in each plane can be represented by the parameters predefined.

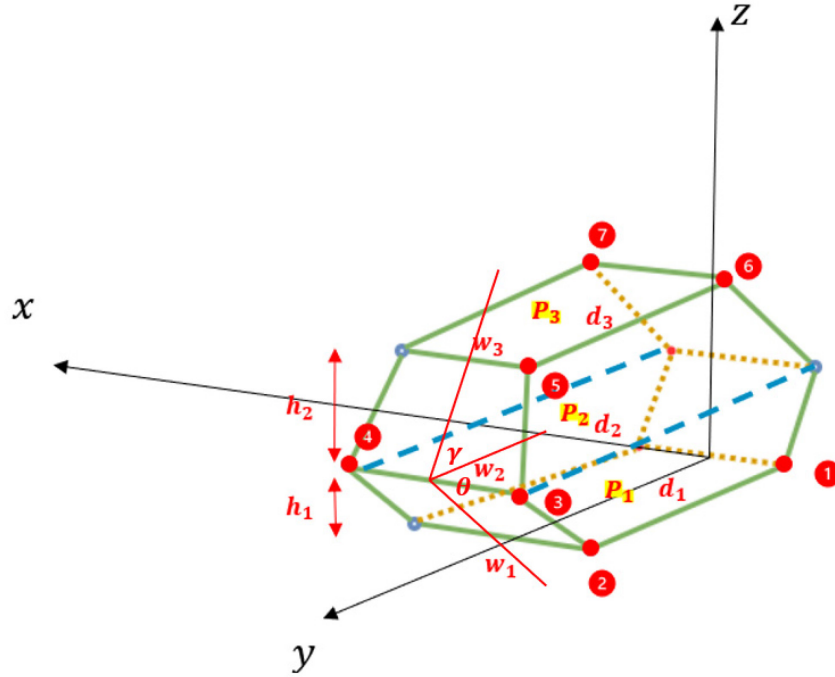


Figure 5-11 Vehicle Dimension with Picked Seven Keypoints

The seven keypoints randomly picked in the example are denoted as big red circles in Figure 5-11. Based on the analysis before, it is easy to get the coordinate of the picked seven points:

$$\begin{cases} P_1: \left(-\frac{w_1}{2}, 0, 0\right) \\ P_2: \left(-\frac{w_1}{2}, d_1, 0\right) \\ P_3: \left(-\frac{w_2}{2}, d_1 + \frac{h_1}{\tan(\theta)}, h_1\right) \\ P_4: \left(\frac{w_2}{2}, d_1 + \frac{h_1}{\tan(\theta)}, h_1\right) \\ P_5: \left(-\frac{w_3}{2}, d_1 + \frac{h_1}{\tan(\theta)} - \frac{h_2}{\tan(\gamma)}, h_1 + h_2\right) \\ P_6: \left(-\frac{w_3}{2}, d_1 + \frac{h_1}{\tan(\theta)} - \frac{h_2}{\tan(\gamma)} - d_3, h_1 + h_2\right) \\ P_7: \left(\frac{w_3}{2}, d_1 + \frac{h_1}{\tan(\theta)} - \frac{h_2}{\tan(\gamma)} - d_3, h_1 + h_2\right) \end{cases} \quad (5-6)$$

Just like what we have done in Section 5.1.2, the team input the seven points into the Eq. (3-15) and solve the coefficient matrix. The results are shown in Table 5.3.

Table 5-3 Sample Results of Camera Calibration

Parameters	Camera Calibration Sample Results
Rotation Matrix (R)	$\begin{bmatrix} 0.9166 & -0.3991 & 0.0246 \\ -0.1564 & -0.4649 & -0.8714 \\ 0.3432 & 0.8540 & -0.4629 \end{bmatrix}$
Translation Matrix (T)	$[8.0751 \ 13.8752 \ 57.1012]^T$
Scale Factor (s_x)	0.9475
Focal Length (f)	369.7290

In the car keypoints detection, 12 keypoints have been detected, while only 7 keypoints are needed to calculate the camera calibration matrix. Therefore, in statistics, there are 792 combinations in total. To take advantage of the five freedom degrees, the team investigated the shape of vehicles and tried to figure out the most accurate solution. Finally, the team identified the eight keypoints on plane 1 and plane 2 should be more reliable in camera calibration calculation, because their dimensions are much more stable in different vehicle types. Then the team ranked the keypoints based on PCK value and then calculated the calibration results based on the 8 keypoints with the highest accuracy. Finally, the final calibration result is the mean of the results we get from eight combinations of keypoints.

Based on the scale of standard cars, the team projects the 2D keypoints to the 3D coordinates. As a result, the location dependency of the points can be measured on both image coordinate and 3D coordinate. Based on the relationship, the team can estimate both internal and external parameters of the camera. In the experiment, without human interaction, the parameters of the camera can keep stable during the test period. As a result, every vehicle passing by the camera can provide input for the system to update the parameters. Finally, the estimated parameters can be limited in a small error range. Table 5-4 shows the estimated parameters after 100 runs (i.e., parameters updated by 100 passing vehicles).

Table 5-4 Camera Calibration Results after 100 Runs

Parameters	Calibration Results After 100 Runs	Calibration Results in First Run
Rotation Matrix (R)	$\begin{bmatrix} 0.9262 & -0.3587 & 0.0224 \\ -0.1462 & -0.4519 & -0.8628 \\ 0.3601 & 0.8442 & -0.4716 \end{bmatrix}$	$\begin{bmatrix} 0.9166 & -0.3991 & 0.0246 \\ -0.1564 & -0.4649 & -0.8714 \\ 0.3432 & 0.8540 & -0.4629 \end{bmatrix}$
Translation Matrix (T)	$[8.1035 \ 13.7832 \ 57.1008]^T$	$[8.0751 \ 13.8752 \ 57.1012]^T$
Scale Factor (s_x)	0.9865	0.9475
Focal Length (f)	371.0268	369.7290

Subsection 5.5 Results Validation

Because MUST sensor is attached to the intelligent camera, it is possible for the team to validate the camera calibration results with some ground truth data. The location and the installation height are pre-measured in the process for final results validation. Also, the team can adjust the focus length and the angle of the surveillance camera through the control panel. Therefore, we can get the ground truth information about the camera on both internal and external sides for camera parameter estimation performance test.

On the vehicle side, we prepared three vehicles with GPS devices to drive through the test to test the 3D projection performance. To simplify the result validation process, the team uses the mean location of detected 12 keypoints to represent the location of the vehicle.

Therefore, the report proposes the following three methods to validate the final results:

- **Verification Using Rotation Matrix**

We haven't completely made use of the property 3) of matrix R in solving camera parameters. For property 1) and 2) of matrix R, it is clear that not all the rows or columns were taken to compute the unknowns. With the properties that have been used before, the determinant of matrix R still cannot be around the true value if it is not solved properly. Hence, the results in first run show that the determinant of matrix R turns to be 1.1521. However, after 100 runs, the determinant of matrix R turns to be 1.0293, which is much closer to 1. Therefore, the results show the work is convincing and the improvements after 100 runs can reach 80%. Although we can foresee the improvements rate will drop with the number of runs increasing, while the accuracy is still high enough to realize vehicle 3D sensing.

- **Verification Using the Pre-measured Ground Truth Data**

Based on the pre-measured camera data including installation height, angle, position, and focal length, the team calculates the ground truth data (shown in Table 5-5). The comparison between the calibration results and ground truth data are presented in Table 5-5. It shows that the accuracy after 100 runs of the system can reach about 95%.

Table 5-5 Camera Calibration Results Validation with Ground Truth Data

Parameters	Calibration Results After 100 Runs	Ground Truth Data
Rotation Matrix (R)	$\begin{bmatrix} 0.9262 & -0.3587 & 0.0224 \\ -0.1462 & -0.4519 & -0.8628 \\ 0.3601 & 0.8442 & -0.4716 \end{bmatrix}$	$\begin{bmatrix} 0.9386 & -0.4035 & 0.0205 \\ -0.1520 & -0.4608 & -0.8681 \\ 0.3623 & 0.8336 & -0.4705 \end{bmatrix}$
Translation Matrix (T)	$[8.1035 \ 13.7832 \ 57.1008]^T$	$[8.1023 \ 13.7912 \ 57.0995]^T$
Scale Factor (s_x)	0.9865	1.0000
Focal Length (f)	371.0268	371.0000

- **Verification Using Self-collected GPS Data**

The team installed GPS devices on the vehicle and derived through the detection area to test if the system can extract the accurate location of the vehicle. Figure 5-12 shows the vehicle validation results based on GPS data. The icon “camera” indicates the location where we installed the MUST sensor. The blue bullets indicate the GPS records collected by the on-board GPS device, and the red bullets denote the detection results of the system. The area in the red circle indicates MUST sensor field of view. The results shown in Figure 5-12 and Table 5-6 indicate the accuracy of vehicle localization can reach about 95% accuracy.

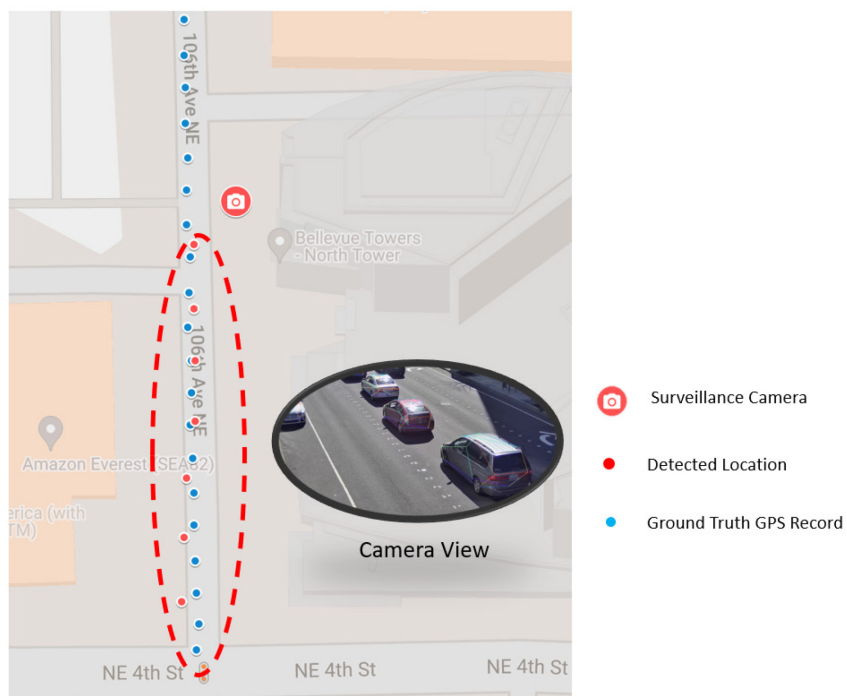


Figure 5-12 Vehicle Localization Results Validation

Table 5-6 Vehicle Localization Results Validation

Detected Location	
Latitude	Longitude
47.6144917	-122.1989245
47.614398	-122.1989232
47.6143227	-122.1989218
47.6142327	-122.1989205
47.6141498	-122.1989393
47.6140637	-122.198946
47.61397	-122.19895
Ground Truth GPS Records	
Latitude	Longitude
47.6149218	-122.1989482
47.6148691	-122.1989466
47.6148218	-122.1989462
47.6147682	-122.1989445
47.6147209	-122.1989442
47.6146682	-122.1989425
47.6146182	-122.1989382
47.6145709	-122.1989405
47.6145209	-122.1989415
47.6144736	-122.1989331
47.6144209	-122.1989341
47.61437	-122.1989364
47.6143218	-122.1989294
47.6142745	-122.1989304
47.6142272	-122.1989327
47.614179	-122.1989257
47.614129	-122.198924
47.6140808	-122.198925
47.614029	-122.198922
47.6139818	-122.1989176
47.6139372	-122.1989146
47.613898	-122.1989196

Section 6 Outreach and Technical Transfer

The proposed research can help advance cooperative perception algorithms of the road-side and on-board units for driving assistance. Particularly, this project developed a prototype system to improve the effectiveness and accuracy of the driving assistance system for roadway traffic and parking assistance.

The research outcomes are disseminated via the following ways:

- Cooperative perception algorithms were summarized into technical papers for presentations at the Transportation Research Board (TRB) Annual Meeting for presentation.
- The manuscript was revised and resubmitted to IEEE transaction on Intelligent Transportation
- Results and findings of the prototype system were disseminated to transportation agencies including WSDOT, City of Bellevue, and PacTrans through webinar.
- The prototype system had been tested in the City of Bellevue testbed.

Section 7 Conclusion and Discussion

Subsection 7.1 Research Conclusion

The research developed a 3D global sensing system based on MUST sensor, which can realize single monocular camera-based traffic scene monitoring and perception. The key accomplishment of the project can be summarized into three points:

- The team developed a customized Mobile Unit for Sensing Traffic (MUST) sensor based on the demands of the project. Because of the outbreak of Covid-19, we cannot implement the sensors in the testbed provided by C2Smart. However, alternatively, the UW team cooperated with the City of Bellevue and installed two customized MUST sensors in their testbed for field tests. The results presented in the report show that the MUST sensors work very well on traffic scene perception even in the heavy snow weather conditions.
- The research project proposed a cooperative perception method that realizes accurate 3D vehicle localization based on a single monocular camera for driving or parking assistance. The customized Computer Vision (CV) algorithm based on Mask-RCNN is implemented on the IoT device for 2D car keypoints detection (X and Y coordinates) based on the image or video data collected by the monocular cameras. Cooperating with the Car Keypoints Prediction (CKP) algorithm, the Depth Detection Algorithm (DDA) developed by the team can extract the third-dimension information (Z coordinates) through the detected car keypoints. Finally, the fusion algorithm integrates the Mask-RCNN Model and DDA results to provide more reliable localization services to all the vehicles in the sensing range. As a result, the SRSU 3D Vehicle Localization System can 1) provide more reliable vehicle localization information; 2) increase calculation efficiency to reduce the information delay; 3) serve more vehicles in a more extensive sensing range.
- Based on the MUST sensor and 3D-SISS algorithm, a communication system has been built up for data transmission among MUST sensors, back servers and other devices. The raw data collected by the camera can be transmitted to MUST sensors for analysis. Then both results and raw data will be transmitted to the back server for data storage. Finally, the warning message will be broadcasted to the devices in the communication range.

Subsection 7.1 Discussion and Future Work

The developed MUST-based system has completed the tasks assigned by the project, however, there are still some areas to improve. In addition, during the field test, certain problems were exposed. Therefore, the report summarizes the following two points as the directions for future work:

- **3D sensing algorithm improvements:**

The existing 3D-SISS system can detect the key points of cars and match them with their 3D dimension to get the camera calibration results. However, at present, because of the lack of labeled data, the team cannot consider the camera lens distortion in the algorithm which may result in some tiny errors. The IP camera we used in MUST sensor doesn't have such problems, however, if we would like to apply the algorithm to other cameras like fisheye cameras, the dissertation may result in serious errors. Therefore, in the future, the team would like to add the lens distortion into the algorithm to increase the flexibility of the system.

- **Communication improvements:**

At present, limited by the communication strategy, we can only realize message broadcasting in the communication area. Therefore, users cannot get the personalized message based on their location or travel mode. Also, because it is still one-way broadcasting, receivers cannot respond or send feedback to the MUST-based system. As a result, in the future, the team will integrate the 3D sensing algorithm with the communication to send out the personalized messages to every user in the detection area. Also, if we can identify their device based on their properties such as location, the two-way communication can be realized.

References

- [1]. Aoki, S., Higuchi, T., & Altintas, O. (2020). Cooperative Perception with Deep Reinforcement Learning for Connected Vehicles. *arXiv preprint arXiv:2004.10927*.
- [2]. A. Angelov, A. Robertson, R. Murray-Smith, and F. Fioranelli. Practical classification of different moving targets using automotive radar and deep neural networks. *IET Radar, Sonar Navigation*, 12(10):1082–1089, 2018. 3, 6, 7
- [3]. Arnold, E., Dianati, M., & de Temple, R. (2019). Cooperative Perception for 3D Object Detection in Driving Scenarios using Infrastructure Sensors. *arXiv preprint arXiv:1912.12147*.
- [4]. Bagnell, J. A., Bradley, D., Silver, D., Sofman, B., & Stentz, A. (2010). Learning for autonomous navigation. *IEEE Robotics & Automation Magazine*, 17(2), 74-84.
- [5]. B. Yang, W. Luo, and R. Urtasun. Pixor: Real-time 3d object detection from point clouds[C]//Proceedings of the IEEE conference on Computer Vision and Pattern Recognition. 2018: 7652-7660.
- [6]. B. Behroozpour, P. A. M. Sandborn, and M. C. Wu. Lidar system architectures and circuits[J]. *IEEE Communications Magazine*, 2017, 55(10): 135-142.
- [7]. Barabanau, I., Artemov, A., Burnaev, E., & Murashkin, V. (2019). Monocular 3d object detection via geometric reasoning on keypoints. *arXiv preprint arXiv:1905.05618*.
- [8]. Chabot, F., Chaouch, M., Rabarisoa, J., Teuliere, C., & Chateau, T. (2017). Deep manta: A coarse-to-fine many-task network for joint 2d and 3d vehicle analysis from monocular image. In *Proceedings of the IEEE conference on computer vision and pattern recognition* (pp. 2040-2049).
- [9]. Chen, Q., Ma, X., Tang, S., Guo, J., Yang, Q., & Fu, S. (2019b). F-cooper: feature based cooperative perception for autonomous vehicle edge computing system using 3D point clouds. In *Proceedings of the 4th ACM/IEEE Symposium on Edge Computing* (pp. 88-100).
- [10]. Chen, X., Ma, H., Wan, J., Li, B., & Xia, T. (2017). Multi-view 3d object detection network for autonomous driving. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition* (pp. 1907-1915).
- [11]. Chen, Q., Tang, S., Yang, Q., & Fu, S. (2019a). Cooper: Cooperative perception for connected autonomous vehicles based on 3d point clouds. In *2019 IEEE 39th International Conference on Distributed Computing Systems (ICDCS)* (pp. 514-524). IEEE.
- [12]. Chen, X., Kundu, K., Zhang, Z., Ma, H., Fidler, S., & Urtasun, R. (2016). Monocular 3d object detection for autonomous driving. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition* (pp. 2147-2156).
- [13]. C. R. Qi, W. Liu, and C. Wu. Frustum pointnets for 3d object detection from rgb-d data[C]//Proceedings of the IEEE conference on computer vision and pattern recognition. 2018: 918-927.
- [14]. Chen, Y., Tai, L., Sun, K., & Li, M. (2020). MonoPair: Monocular 3D Object Detection Using Pairwise Spatial Relationships. *arXiv preprint arXiv:2003.00504*.
- [15]. G. Appenzeller, I. Keslassy, N. McKeown, "Sizing Router Buffers," Proc. of Sigcomm 2004, August 30 - Sept. 3, 2004, Portland, Oregon, USA.
- [16]. Gong Y., Rossi D., Testa C., Valenti S., Täht M.D. "Fighting the bufferbloat: on the coexistence of AQM and low priority congestion control," *Comput. Netw.*, 65 (2014), pp. 255-267
- [17]. Hurl, B., Cohen, R., Czarnecki, K., & Waslander, S. (2019). TruPercept: Trust Modelling for Autonomous Vehicle Cooperative Perception from Synthetic Data. *arXiv preprint arXiv:1909.07867*.
- [18]. Ha S., Rhee I., Xu L., "CUBIC: a new TCP-friendly high-speed TCP variant," *ACM SIGOPS Oper. Syst. Rev.*, 42 (5) (2008), pp. 64-74

- [19]. Jayaweera, N., Marasinghe, D., Rajatheva, N., & Latva-Aho, M. (2020, March). Factory Automation: Resource Allocation of an Elevated LiDAR System with URLLC Requirements. In *2020 2nd 6G Wireless Summit (6G SUMMIT)* (pp. 1-5). IEEE.
- [20]. J. Weng, P. Cohen, and M. Herniou, "Camera calibration with distortion models and accuracy evaluation," *IEEE Trans. Pattern Anal. Mach. Intell.*, Vol. 14, pp. 965-980, Oct. 1992.
- [21]. Jayaweera, N., Rajatheva, N., & Latva-aho, M. (2019, April). Autonomous driving without a burden: View from outside with elevated lidar. In *2019 IEEE 89th Vehicular Technology Conference (VTC2019-Spring)* (pp. 1-7). IEEE.
- [22]. J. Hecht. Lidar for self-driving cars[J]. *Optics and Photonics News*, 2018, 29(1): 26-33.
- [23]. J. V. Brummelen, M. O'Brien, and D. Gruyer. Autonomous vehicle perception: The technology of today and tomorrow[J]. *Transportation research part C: emerging technologies*, 2018, 89: 384-406.
- [24]. Kim, S. W., Chong, Z. J., Qin, B., Shen, X., Cheng, Z., Liu, W., & Ang, M. H. (2013, November). Cooperative perception for autonomous vehicle control on the road: Motivation and experimental results. In *2013 IEEE/RSJ International Conference on Intelligent Robots and Systems* (pp. 5059-5066). IEEE.
- [25]. Kim, S. W., Qin, B., Chong, Z. J., Shen, X., Liu, W., Ang, M. H., ... & Rus, D. (2014). Multivehicle cooperative driving using cooperative perception: Design and experimental validation. *IEEE Transactions on Intelligent Transportation Systems*, 16(2), 663-680.
- [26]. K. Avrachenkov, U. Ayesta, E. Altman, P. Nain and C. Barakat, "The effect of router buffer size on the TCP performance", in *Proc. of the LONIS Workshop on Telecommunication Networks and Teletraffic Theory*, pages 116-121, St. Petersburg, Russia, January 2002.
- [27]. Kong, P. Y. (2020). Computation and Sensor Offloading for Cloud-Based Infrastructure-Assisted Autonomous Vehicles. *IEEE Systems Journal*.
- [28]. K. Simonyan and A. Zisserman. Very deep convolutional networks for large-scale image recognition. *arXiv preprint arXiv:1409.1556*, 2014. 3
- [29]. Kundu, A., Li, Y., & Rehg, J. M. (2018). 3d-rcnn: Instance-level 3d object reconstruction via render-and-compare. In *Proceedings of the IEEE conference on computer vision and pattern recognition* (pp. 3559-3568).
- [30]. Lang, A. H., Vora, S., Caesar, H., Zhou, L., Yang, J., & Beijbom, O. (2019). Pointpillars: Fast encoders for object detection from point clouds. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition* (pp. 12697-12705).
- [31]. Li, P., Zhao, H., Liu, P., & Cao, F. (2020). RTM3D: Real-time Monocular 3D Detection from Object Keypoints for Autonomous Driving. *arXiv preprint arXiv:2001.03343*.
- [32]. Liu, W., Kim, S. W., Chong, Z. J., Shen, X. T., & Ang, M. H. (2013, November). Motion planning using cooperative perception on urban road. In *2013 6th IEEE Conference on Robotics, Automation and Mechatronics (RAM)* (pp. 130-137). IEEE.
- [33]. Liu, Z., Wu, Z., & Tóth, R. (2020). SMOKE: Single-Stage Monocular 3D Object Detection via Keypoint Estimation. *arXiv preprint arXiv:2002.10111*.
- [34]. Lucic, M. C., Ghazzai, H., Alsharoa, A., & Massoud, Y. (2020). A Latency-Aware Task Offloading in Mobile Edge Computing Network for Distributed Elevated LiDAR. *arXiv preprint arXiv:2003.09741*.
- [35]. Ma, X., Wang, Z., Li, H., Zhang, P., Ouyang, W., & Fan, X. (2019). Accurate monocular 3d object detection via color-embedded 3d reconstruction for autonomous driving. In *Proceedings of the IEEE International Conference on Computer Vision* (pp. 6851-6860).

- [36]. Manhardt, F., Kehl, W., & Gaidon, A. (2019). Roi-10d: Monocular lifting of 2d detection to 6d pose and metric shape. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition* (pp. 2069-2078).
- [37]. M. Liang, B. Yang, Y. Chen, R. Hu, and R. Urtasun. Multi-task multi-sensor fusion for 3d object detection. In CVPR, 2019. 1
- [38]. M. A. Richards. Fundamentals of radar signal processing. Tata McGraw-Hill Education, 2005. 3, 4
- [39]. Mousavian, A., Anguelov, D., Flynn, J., & Kosecka, J. (2017). 3d bounding box estimation using deep learning and geometry. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition* (pp. 7074-7082).
- [40]. P. Dong, and Q. Chen. LiDAR remote sensing and applications[M]. CRC Press, 2017.
- [41]. Qi, C. R., Liu, W., Wu, C., Su, H., & Guibas, L. J. (2018). Frustum pointnets for 3d object detection from rgb-d data. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition* (pp. 918-927).
- [42]. Qi, C. R., Su, H., Mo, K., & Guibas, L. J. (2017). Pointnet: Deep learning on point sets for 3d classification and segmentation. In *Proceedings of the IEEE conference on computer vision and pattern recognition* (pp. 652-660).
- [43]. Qin, Z., Wang, J., & Lu, Y. (2019, July). Monogrnet: A geometric reasoning network for monocular 3d object localization. In *Proceedings of the AAAI Conference on Artificial Intelligence* (Vol. 33, pp. 8851-8858).
- [44]. Roddick, T., Kendall, A., & Cipolla, R. (2018). Orthographic feature transform for monocular 3d object detection. *arXiv preprint arXiv:1811.08188*.
- [45]. Raina G., Wischik D. "Buffer sizes for large multiplexers: TCP queueing theory and instability analysis," Next Generation Internet Networks, 2005, IEEE (2005), pp. 173-180
- [46]. Shi, S., Wang, X., & Li, H. (2019). Pointtrcnn: 3d object proposal generation and detection from point cloud. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition* (pp. 770-779).
- [47]. S. Heuel and H. Rohling. Two-stage pedestrian classification in automotive radar systems. In 2011 12th International Radar Symposium (IRS), pages 477–484, Sep. 2011. 3
- [48]. Villamizar, C. and Song C. (1995), "High Performance TCP in ANSNET", ACM Computer Communication Review, vol. 24, no. 5, pp. 45-60.
- [49]. Wang, Y., Chao, W. L., Garg, D., Hariharan, B., Campbell, M., & Weinberger, K. Q. (2019). Pseudo-lidar from visual depth estimation: Bridging the gap in 3d object detection for autonomous driving. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition* (pp. 8445-8453).
- [50]. Xiang, Y., Choi, W., Lin, Y., & Savarese, S. (2017, March). Subcategory-aware convolutional neural networks for object proposals and detection. In *2017 IEEE winter conference on applications of computer vision (WACV)* (pp. 924-933). IEEE.
- [51]. Xu, B., & Chen, Z. (2018). Multi-level fusion based 3d object detection from monocular images. In *Proceedings of the IEEE conference on computer vision and pattern recognition* (pp. 2345-2353).
- [52]. X. Du, M. H Ang Jr, S. Karaman, and Daniela Rus. A general pipeline for 3d detection of vehicles. In ICRA, 2018. 1, 8
- [53]. X. Chen, H. Ma, and J. Wan. Multi-view 3d object detection network for autonomous driving[C]//Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition. 2017: 1907-1915.
- [54]. X. Gao, G. Xing, S. Roy, and H. Liu. Experiments with mmwave automotive radar test-bed. In Asilomar Conference on Signals, Systems, and Computers, 2019. 3, 6, 7

- [55]. Yan, Y., Mao, Y., & Li, B. (2018). Second: Sparsely embedded convolutional detection. *Sensors*, 18(10), 3337.
- [56]. Yang, Z., Sun, Y., Liu, S., Shen, X., & Jia, J. (2019). Std: Sparse-to-dense 3d object detector for point cloud. In *Proceedings of the IEEE International Conference on Computer Vision* (pp. 1951-1960).
- [57]. Zhou, Y., & Tuzel, O. (2018). Voxelnet: End-to-end learning for point cloud based 3d object detection. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition* (pp. 4490-4499).