Title
Robust Perception and Auto-teaching for Autonomous Robotic Systems

Author
Wang, Zining

Publication Date
2020

Peer reviewed|Thesis/dissertation

Robust Perception and Auto-teaching for Autonomous Robotic Systems

by

Zining Wang

A dissertation submitted in partial satisfaction of the

requirements for the degree of

Doctor of Philosophy

in

Mechanical Engineering

in the

Graduate Division

of the

University of California, Berkeley

Committee in charge:

Professor Masayoshi Tomizuka, Chair
Professor Roberto Horowitz
Professor Francis Christ

Fall 2020

Robust Perception and Auto-teaching for Autonomous Robotic Systems

Abstract

Robust Perception and Auto-teaching for Autonomous Robotic Systems

by

Zining Wang

Doctor of Philosophy in Mechanical Engineering

University of California, Berkeley

Professor Masayoshi Tomizuka, Chair

Modern autonomous robotic systems are equipped with perception subsystems to handle unexpected failure cases and to navigate more intelligently in the unstructured environment. Robots navigate in a cluttered environment full of noise and disturbance. Robust perception extracts target objects from visual observations while rejecting all the noise and disturbances. It also focuses on increasing the redundancy by fusing information from multiple sensors. However, the vision sensor is mounted on the robot system and is affected by the model uncertainty of the robot. Therefore, auto-teaching is proposed to handle the modeling error of the robot by calibrating the model parameters while estimating states of the target object. On the other hand, robustly detecting target objects is the prerequisite for auto-teaching without human intervention, which requires studying perception and auto-teaching simultaneously. In addition, perception is also an essential part for perceiving the complex environment, where deep learning methods are becoming the mainstream recently. However, robustness of learning-based perception algorithms is not well explored.

In this dissertation, robust perception is discussed for robots carrying vision sensors and auto-teaching is developed for robots to recover from failures. Robustness of the perception subsystem is considered by developing global methods to reject disturbances and sensor fusion to improve redundancy. Several methods are proposed in both classic computer vision and deep learning areas with applications to two kinds of autonomous robotic systems, namely the industrial manipulator and the autonomous vehicle.

For the industrial manipulator discussed in Part I of this dissertation, the name *hand-eye system* is conventionally used referring to a robot arm holding vision sensors. Chapter 2 models the system and builds the motion block. The kinematic model is used for visual-inertial sensor fusion and generating the calibration parameters for auto-teaching. Planning and tracking control of the system are necessary for auto-teaching and ensuring the quality of visual data captured by the hand-eye system.

Industrial manipulators and their target objects have rich geometric information and accurate known shape, which is more suitable for classic computer vision (CV) methods. Chapter 3 and 4 constructs the robust perception block of the hand-eye system. Chapter 3 proposes several global shape matching methods for two kinds of visual inputs, namely image and point clouds. We globally search all potential matches of deformed target objects to avoid local optimals caused by disturbances. Chapter 4 introduces probabilistic inference to increase the robustness against noise when matching detected objects temporally. The proposed probabilistic hierarchical registration algorithm outperforms the deterministic feature descriptor-based algorithm used in state-of-the-art SLAM methods.

Visual detection is not robust against model uncertainty of the system and only gives 2D location of the object. Auto-teaching simultaneously calibrates the parameters of the systems while estimating the state of detected objects. Chapter 5 introduces the auto-teaching framework directly using the perception results from Chapter 3 and Chapter 4. Visual-inertial sensor fusion is used to increase the calibration accuracy by taking the robot motion measurement into account. Chapter 6 proposes an active auto-teaching framework which closes the calibration loop of the hand-eye system by planning optimal measurement poses using the updated parameters.

Autonomous vehicles operate in a more versatile scenario where target objects are complex and unstructured. Deep learning-based methods have become the paradigm in this area in recent years, but robustness is the major concern for scaling up its application in the real world. In Part II of the dissertation, the robustness of learning-based detectors is discussed. Chapter 7 proposes two camera-LiDAR sensor fusion detection networks to increase the performance and redundancy of the detector. The proposed fusion layer is very efficient and back-propagatable which perfectly suits the learning framework. In Chapter 8, we further dive into the training and evaluation procedure of learning-based detectors. A probabilistic representation is proposed for labels in the dataset to handle the uncertainty of training data. A new evaluation metric is introduced for the proposed probabilistic representation to better measure the robustness of learning-based detectors.

To my family

Contents

List of Figures

List of Tables

Acknowledgments

The past five and a half years in UC Berkeley has been the most precious and unforgettable period in my life. I would like to give credits to all the awesome people providing enormous help and support throughout this journey.

First and foremost, I would like to express my greatest appreciation and deepest gratitude to my advisor, Professor Masayoshi Tomizuka, who drastically shaped both my life and career path with his tremendous knowledge and foresight in academia, as well as his enthusiasm and dedication in mentorship. He led me to the entrance of research, establishing my sense of innovation and guiding me through the initialization of projects. He has always been supportive and responsive anytime, anywhere and for any problem. I usually remember the day when he drove me to the South Bay in the first semester, as if it just happened yesterday. Many things changed after these years, but the energy and enthusiasm I felt from Professor stays the same, just like when he pushed the pedal. Besides, being the GSI in his ME233 class has taught me how to be a helpful teacher to students. This dissertation could not be completed without his guidance and patient help.

Special thanks to Professor Roberto Horowitz and Professor Michael Christ and Professor Anil Aswani in my qualifying and dissertation committee. Professor Roberto Horowitz was the chair of my qualifying exam committee. Many thanks to his kindness since we met before joining Berkeley and his great tolerance to an aggressive student like me in his ME232 and E231 classes. I am also grateful to Professor Michael Christ and Professor Anil Aswani for their insightful advice and comments.

I would like to thank Applied Materials for their generous financial support, as well as the mentors, Dr. Sanggyum Kim and Dr. Raechel Tan, during my internship. Their consistent help provides me rich experience and insight in the industry.

It is my honor to be a member of the Mechanical System Control (MSC) laboratory, working and building up friendship with so many brilliant colleges. In particular, I am grateful to Professor Cong Wang and Dr. Wei Zhan for their invaluable instructions, suggestions and encouragement. I want to thank Dr. Liting Sun, Dr. Yu Zhao and Dr. Xiaowen Yu for all the inspiring discussions. Besides, I would like to give my thanks to all current and previous lab members for their tremendous support throughout the years: Professor Changliu Liu, Dr. Chung-Yen Lin, Dr. Hsien-Chung Lin, Professor Minghui Zheng, Dr. Chen-Yu Chan, Dr. Kevin Haninger, Dr. Junkai Lu, Dr. Shiying Zhou, Dr. Cheng Peng, Dr. Yongxiang Fan, Dr. Daisuke Kaneishi, Dr. Shuyang Li, Dennis Wai, Dr. Jianyu Chen, Kiwoo Shin, Richard Lee, Saman Fahandezhsaadi, Yeping Hu, Yujiao Cheng, Zhuo Xu, Shiyu Jin, Chen Tang, Jiachen Li, Hengbo Ma, Jessica Leu, Yiyang Zhou, Lingfeng Sun, Changhao Wang, Xinghao Zhu, Ge Zhang, Huidong Gao, Ting Xu, Zheng Wu, Xiang Zhang, Wu-Te Yang, Jinning Li and Catherine Faulkner. My best wishes to all of you in your future endeavors.

Last but not least, I would like to express my love to my parents for their endless love. They offered their best to me, always supporting me to explore the world and chase the dream. I sincerely wish that my father could win the fight against the disease.

Chapter 1

Introduction

1.1 Background of Robust Detection and Auto-teaching

Robots are becoming more and more intelligent and are getting involved in the daily life of the society, especially for the emerging autonomous driving cars and robot manipulators which are being tested extensively. The general structure of an autonomous robotic system interacting with the environment is shown in Figure 1.1. The environment with which the robot interacts is complex, and a robust perception subsystem is necessary for extracting semantic and geometric information from the surrounding environment [1]. There are many concepts for robust perception. Some aim to extract more exclusive features and do global search of target objects to reject disturbances. Others improve the redundancy by fusing information from different sensors. These methods have been actively explored in the classic computer vision (CV) area for separating interested and non-interested objects in cluttered scenes, reconstructing 3D poses of target objects, tracking multiple objects that are similar to each other and keeping track of certain objects when some sensors are blocked.

Deep learning-based perception subsystems, however, are not well studied in terms of robustness although they have superior performance and generalization capability than classic CV methods in autonomous robots. It is difficult to fuse multiple sensors in a network. Moreover, failure of networks is also hard to be identified and tracked. Researchers are getting aware of the problem and various solutions are being explored. Fusion structures for networks are developed but the efficiency and training capability are the major concerns. Probabilistic detection networks try to estimate the uncertainty of their own predictions, but there is no evaluation mechanism for the predicted distributions.

The robot is sometimes affected by large disturbances which cannot be handled by the perception subsystem itself. For example, a manipulator may encounter severe failure which significantly changes the workspace and deteriorates the fine-tuned parameters. The perception subsystem may not be able provide accurate information anymore because the sensors are carried by the robot having model uncertainty. In this case, auto-teaching is proposed to

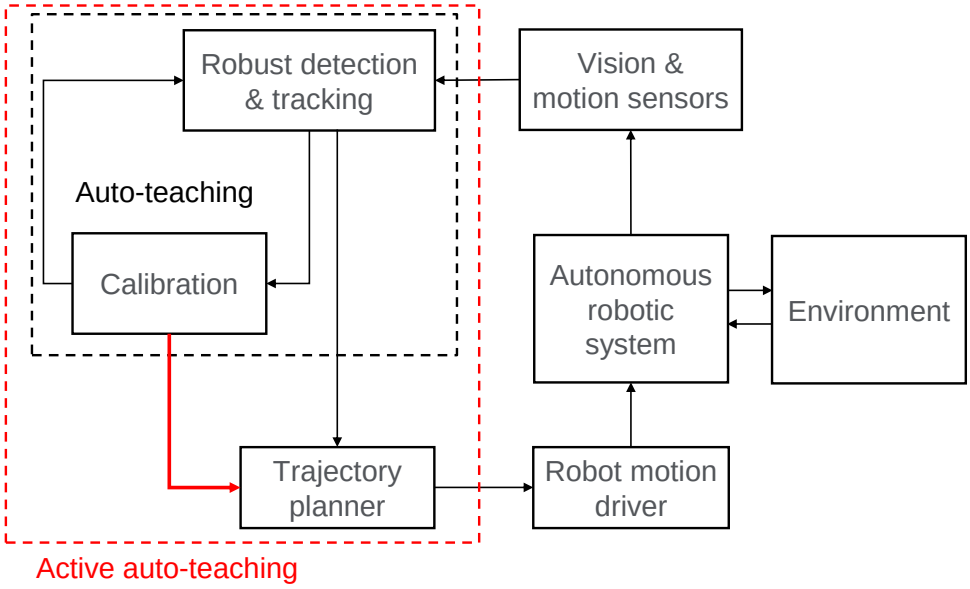


Figure 1.1: General structure of a robot with perception subsystem.

let the robot recover from failure on itself by calibrating the parameters without human intervention. It closes the perception loop to handle model uncertainty as shown in Figure 1.1. The auto-teaching is still somewhat an open-loop system. Although the control loop is closed, the calibration loop is open because the trajectory for calibration is not affected by the updated parameters and surrounding objects. Since the accuracy of auto-teaching depends heavily on measurement of the surrounding objects, designing the trajectory that better observes the object is important for improving the performance. Therefore, active auto-teaching is proposed which adds another feedback to the trajectory planner as shown in Figure 1.1.

1.2 Hand-eye System and Autonomous Driving System

Hand-eye System

Mobile robots and industrial robots usually interact with the surrounding object with a robot arm. Industrial robot manipulators, such as those from FANUC, ABB, Yaskawa, Kawasaki and KUKA as shown in Fig. 1.2, are designed to minimize compliance for better precision in their working environment. Some manipulators are equipped with visual sensors to improve position precision and accuracy. Other popular robots in various areas, such as those from Intuitive Surgical, Rethink robotics and Willow Garage as shown in Fig. 1.3, are less precise

but their tasks are more unstructured. Having a vision system is necessary for these robots to explore and understand the surrounding environment in order to conduct complex tasks.

The visual sensors, which are usually RGB or RGBD cameras, are mounted on the robot to obtain the view from the end effector as opposed to a global view by a camera away from the robot. The field of view of the local vision sensor is limited but can move with the robot. The robotic system with a camera mounted on the robot is referred to as a hand-eye system in many references such as [2, 3, 4, 5, 6, 7]. Chapter 2 through Chapter 5 mainly deal with problems of the hand-eye system.

(a) A FANUC robot. (b) An ABB robot. (c) A Yaskawa robot.

(d) A Kawasaki robot. (e) A KUKA robot.

Figure 1.2: Examples of industrial robot manipulators.

Autonomous Driving System

The emergence of autonomous driving systems nowadays depicts a bright future with improved road efficiency and safety, while also raising many challenging tasks. Autonomous driving vehicles interact with a highly dynamic and complex environment. As the first step of perceiving the surrounding environment, detection is the fundamental part of the perception system of driverless cars. The detection targets include static objects like traffic signals,

(a) Surgical robot from Intuitive surgical.

(b) Dual hand manipulator from Rethink Robotics.

(c) Dual hand manipulator from Willow Garage.

Figure 1.3: Examples of other robot manipulators.

road landmarks and trees, as well as dynamic objects like other vehicles, pedestrians and cyclists. The sensor configuration of a typical driverless car is shown by Fig. 1.4. 3D LiDAR and camera suite are the two types of perception related sensors. They are responsible for collecting 3D location and motion information of surrounding objects. These two types of sensors provide significantly different formats of data for perception and how to combine them is an essential problem in 3D detection.

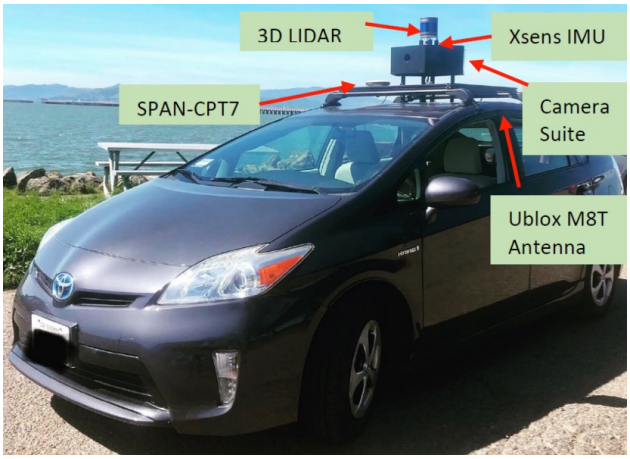


Figure 1.4: Sensor configuration of an autonomous driving vehicle [8].

For autonomous driving vehicles, we focus on 3D detection of dynamic objects with deep learning methods, since deep neural networks have become the standard framework for detecting unstructured objects. Sensor fusion can effectively improve the performance and

redundancy of the system. Several fusion methods of combining 2D camera and 3D LiDAR information are proposed in this work. The proposed fusion method is very efficient and provides an approach to fuse most state-of-the-art convolutional neural networks for different sensors. In addition to improving the robustness of learning-based methods, the evaluation of robustness is also important. Existing benchmark dataset only provide deterministic labels and evaluation metrics which do not capture the uncertainty of detection outputs. This problem is growing into an active research area where new datasets and evaluation methods are being developed.

1.3 Dissertation Outline

Several methodologies are developed for robust perception of two autonomous robotic systems: industrial manipulators and autonomous driving systems. Robust classic and deep learning-based CV methods are improved by introducing global methods and sensor fusion. Auto-teaching and active auto-teaching are proposed to further improve the robustness and performance of the perception subsystem. Classic CV methods and auto-teaching are implemented on an industrial manipulator. Fusion-based deep learning methods are utilized for the autonomous driving system and a new evaluation methodology is proposed for robust learning-based CV methods. The dissertation is organized as follows.

Industrial Manipulator System

In Part I of the dissertation, an industrial manipulator system, which is a wafer handling robot used in the semiconductor manufacturing industry, is studied. Chapter 2 models the hand-eye system which refers to the industrial manipulator carrying vision sensors on its end effector. Chapter 3 and 4 introduce several robust perception algorithms designed for identifying target objects in the robot workspace. Chapter 5 and 6 develop the auto-teaching structure to close the perception loop of the robot, improving the performance and robustness of the system.

Hand-eye System of Wafer Handling Robot

This chapter builds the basic geometric, kinematic and dynamic models of the hand-eye system which are fundamental for the perception subsystem of autonomous robots since the sensor moves together with the robot. On the one hand, Sensors fusion for robust perception relies on the geometric and kinematic model. On the other hand, auto-teaching relies on calibration parameters in the model of the hand-eye system. Motion planning and control methods are introduced where optimal trajectory planning is developed to guarantee safe and smooth navigation of the robot. Automatic tracking controller tuning minimizes the vibration of the hand to ensure the quality of visual data.

Robust Detection of Objects in the Robot Workspace

This chapter proposes robust detection of target objects from various data formats provided by vision sensors, including images and point clouds. The common idea behind these methods is that the shape models of target objects are known. Images and point clouds are rich in shape features but are also full of noise, distortion and disturbances. Global-matching is desired to find the best match to the target object. For images, basic geometric segments are available and Hough-based global voting is proposed for matching shapes consisting of basic segments. For point clouds, efficient global non-rigid point registration is proposed which searches for the best rigid transformation globally and allows non-rigid deformation to handle the noise and distortion. The robustness of proposed methods is evaluated with real objects in the robot workspace, including wafers and chamber openings.

Robust Multiple Object Tracking

Detection extracts target objects from each frame and detected objects are independent across different frames. It is tracking which associates objects in the time space. In Chapter 4, visual and motion readings from the robot are fused for robust tracking of objects returned by Chapter 3. A probabilistic hierarchical registration algorithm is proposed to handle the point-to-point matching noise, which outperforms the deterministic feature descriptor-based algorithm used in state-of-the-art SLAM methods.

Auto-teaching with Visual Inertial Sensor Fusion

Perception subsystem is coupled with the kinematic model of the hand-eye system since vision sensors are mounted on the robot. When model uncertainty is present, which often occurs when certain failure happens, perception performance will be deteriorated. Auto-teaching stabilizes the system against model uncertainty by automatically calibrating the parameters using the observed object and motion data. First, an Unscented Kalman Filter is introduced to leverage the perception output from SLAM and motion output from robot forward kinematics. Later on, visual and inertial data is further fused to form a joint optimization in the proposed SLAM-MOT method. It is shown that calibration parameters converge quickly and precise 3D reconstruction of target objects is achieved.

Active Auto-teaching of Wafer Handling Robot

The calibration loop of auto-teaching depends on the motion of the robot which is related the visibility of objects. In order to maximize the performance of auto-teaching, active auto-teaching is taken to close the calibration loop by providing designed measurement pose feedback to the motion block. We use optimal experimental design to measure the optimality of the measurement pose for calibration and to guide the planning of the next observation. The uncertainty of estimated parameters is reduced and the accuracy of the estimated 3D object location is increased compared to the auto-teaching method.

Autonomous Driving System

The autonomous driving system is considered in Part II of this dissertation where deep learning methods are more popular. For learning-based perception methods, robustness is not well studied. In Chapter 7, fusion layers are developed for the sensor fusion of learning-based detection networks. In Chapter 8, we further dive into the training and evaluation procedure of learning-based detectors. Uncertainty models and evaluation metrics for detection datasets are proposed.

Deep 3D Detection with Camera-LiDAR Sensor Fusion

Sensor fusion for deep learning-based CV methods are developed in Chapter 5. Specifically, we focus on fusing the image and point cloud from camera and LiDAR equipped on the autonomous driving vehicle. Fusion does not only improve the performance, but also introduces robustness to deep neural networks by adding redundancy and taking more information into account. Two fusion structures, namely a sequential one and a parallel one, are proposed for efficient end-to-end fusion of single sensor networks. The proposed sparse pooling layer design for parallel fusion shows significant advantage against existing state-of-art structures. It allows efficient transformation of the multi-view features at any stage of the network with fast inference speed and high accuracy. Moreover, the proposed layer is not restricted to detection only, it can be directly used in other perception tasks, such as semantic segmentation and tracking, to enable end-to-end deep sensor fusion.

Robustness Evaluation of Learning-based Detection

The availability of real-world datasets is the prerequisite for developing object detection methods for autonomous driving. While ambiguity exists in object labels due to error-prone annotation process or sensor observation noises, current object detection datasets only provide deterministic annotations without considering their uncertainty. This precludes an in-depth evaluation among different object detection methods, especially for those that explicitly model predictive probability. In Chapter 8, we propose a generative model to estimate bounding box label uncertainties from LiDAR point clouds, and define a new representation of the probabilistic bounding box through spatial distribution. Comprehensive experiments show that the proposed model represents uncertainties commonly seen in driving scenarios. Based on the spatial distribution, we further propose an extension of IoU, called the Jaccard IoU (JIoU), as a new evaluation metric that incorporates label uncertainty. Experiments on the KITTI and the Waymo Open Datasets show that JIoU is superior to IoU when evaluating probabilistic object detectors.

Part I

Industrial Manipulator System

Chapter 2

Hand-eye System of Wafer Handling Robot

2.1 Introduction

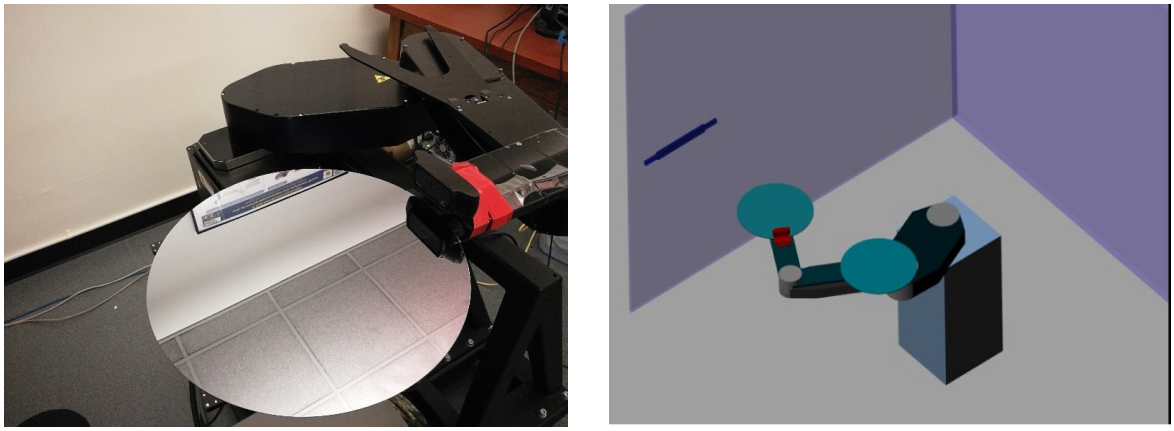
This chapter studies the modeling, planning and control of this system which are low-level parts that enable the other high-level parts studied in the following chapters. The robot system we work on in this paper is a wafer handling robot that works in the factory interface (FI) of the semiconductor industry. A typical working environment is shown in Fig. 2.1. The factory interface (FI) connects consecutive wafer processing procedures in different tools during the semiconductor manufacturing process. The clean environment and high precision requirement in FIs prevent human contact, therefore wafer handling robots work in FIs to transfer wafers. The typical work of a wafer handling robot is to pick up wafers from the wafer station (or FOUPs) and put them into the processing chamber through a narrow opening called load lock, or transfer wafers between workstations.

The wafer handling robot is able to move around in the FI automatically. The robot is not allowed to touch any tools in the FI, but a no-touch sensor such as a laser sensor and camera can be carried by the robot. The camera is configured to be mounted on the arm of the robot as shown by Fig. 2.2.

2.2 Kinematics and Dynamics Modeling of Hand-eye system

The robots are driven by motors at its joints and the vision sensor moves together with the robot. Therefore, the kinematics and dynamics of the robot are required. This section will first introduce the coordinates of the hand-eye system and introduce some general transformation. Then the detailed parameterized model of the hand-eye system studied in this thesis will be formulated.

Figure 2.1: A photo of the factory interface with a wafer handling robot.



(a) A photo of the hand-eye system.

(b) Simulation of the hand-eye system.

Figure 2.2: Demonstration of the hand-eye system of the wafer handling robot.

The Coordinate System and Transformations

There are several important frames in the hand-eye system, which are shown in Fig. 2.3. The world frame is the coordinate system of the workspace and it is usually static and fixed. The base frame is the coordinate of the base of the robot and it may be mobile. The end effector frame is where the robot physically interacts with the environment and conducts various tasks. The end effector is usually at the tip of the robot arm. The camera frame is the coordinate of the camera mounted on the robot. The coordinate systems are linked by rigid transformations that describe the motion of the robot. The world to base trans-

formation, T_{wb}, is from the world frame to the base frame and describes the motion of the robot base. The base to end effector transformation, T_{be}, describes the motion of the robot arm which corresponds to the so called forward kinematics of the robot. The end effector to camera transformation, T_{ec}, describes the geometric relationship between the perception sensor and the robot manipulator and is important for visual-guided manipulation. The world to camera transformation, T_{wc}, is used in detecting target objects. These transformations $T_{wb}, T_{be}, T_{ec}, T_{wc}$ are conventionally denoted as $Z^{-1}, B^{-1}, X, A^{-1}$ in hand-eye calibration papers [7].

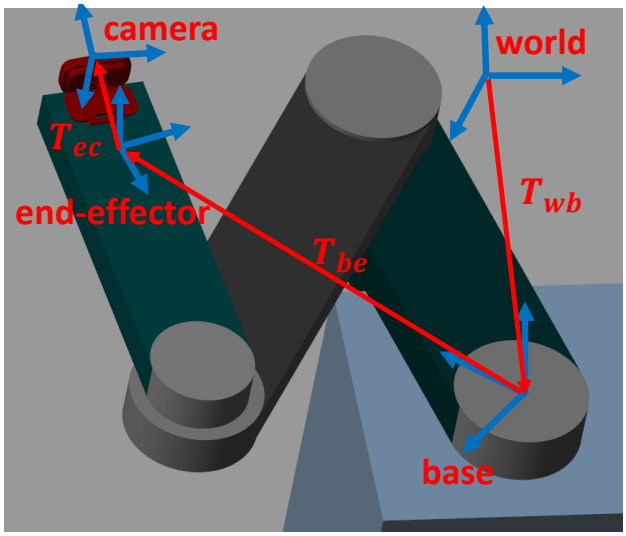


Figure 2.3: An illustration of the hand-eye system with important coordinates and transformations.

These transformations are usually represented as homogeneous transformations containing rotation and translation. For example, T_{wb} can be written as a 4×4 matrix in (2.1)

$$T_{wb} = \begin{bmatrix} R_{wb} & t_{wb} \\ 0_{1\times 3} & 1 \end{bmatrix}, \tag{2.1}$$

where $R_{wb} \in SO(3)$ is the rotation matrix from world frame to base frame and t_{wb} is the position of the origin of the base coordinate system expressed in coordinates of the world-centered coordinate system. The homogeneous transformations can be combined with matrix multiplication which forms a chain of rigid transformations. For example, the world-camera transformation can be derived as

$$T_{wc} = T_{wb}T_{be}T_{ec}, \tag{2.2}$$

and a point (x_c, y_c, z_c) in the camera frame has position (x, y, z) calculated in the world frame as

$$\begin{bmatrix} x \\ y \\ z \\ 1 \end{bmatrix} = T_{wc} \begin{bmatrix} x_c \\ y_c \\ z_c \\ 1 \end{bmatrix}. \tag{2.3}$$

The inverse of T_{wc} is also called the extrinsics of the camera $E := T_{wc}^{-1}$.

Kinematics of the Wafer Handling Robot with Camera

The robot used in this thesis is an atmosphere wafer handling robot. The robot works in the semiconductor industry. It picks wafers from the wafer carriers (FOUPs) in the FI for processing and puts the processed wafer back to another FOUP for further transferring. The geometry of the robot consists of cylinders and extrusion of polygons that approximates the shape of the robot. The actual shape of the robot is very close to the modeling used here. This robot transfers 300mm wafers represented by thin cylinders. Fig. 2.4 shows the solids used for modeling of the robot and the names of each joint which are JT2, JT4, JT6 and JT7. There is another joint JT3 not marked in the figure which drives the Z axis of the robot.

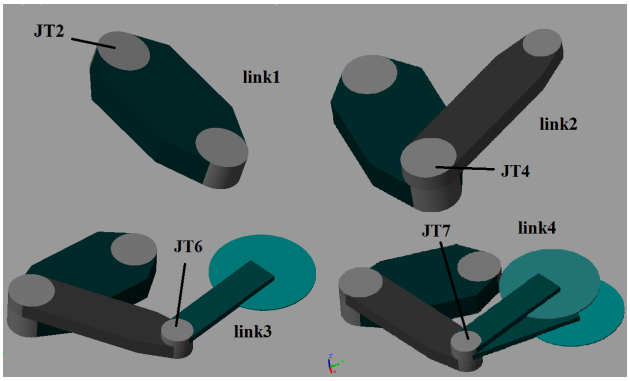


Figure 2.4: An illustration of the geometry of the wafer handling robot.

The parameters of its kinematic chain are shown in Fig. 2.5. The angles of JT2, JT4, JT6, JT7 are $\theta_1, \theta_2, \theta_3, \theta_4$, respectively. The height of the robot is determined by the position h of JT3. These four angles and the height determine the positions and orientations of all robot parts in the XY plane. l_1, l_2, l_3, l_4 are the lengths of links. h_{01} is the initial height of link4 and α, β are the orientations of two blades. Then T_{be} from base to blade 1 can be

derived as

$$T_{be_1} = \begin{bmatrix} \cos\alpha & 0 & \sin\alpha & l_1\cos\theta_1 + l_2\cos\theta_{12} + l_3\cos\theta_{123} \\ 0 & 1 & 0 & l_1\sin\theta_1 + l_2\sin\theta_{12} + l_3\sin\theta_{123} \\ -\sin\alpha & 0 & \cos\alpha & h + h_{01} \\ 0 & 0 & 0 & 1 \end{bmatrix}, \tag{2.4}$$

where $\theta_{12} = \theta_1 + \theta_2$ and $\theta_{123} = \theta_1 + \theta_2 + \theta_3$. The Jacobian for velocity calculation is given in (2.5) by derivation. Acceleration is given by (2.6). Sometimes the acceleration at the wafer center needs to be considered.

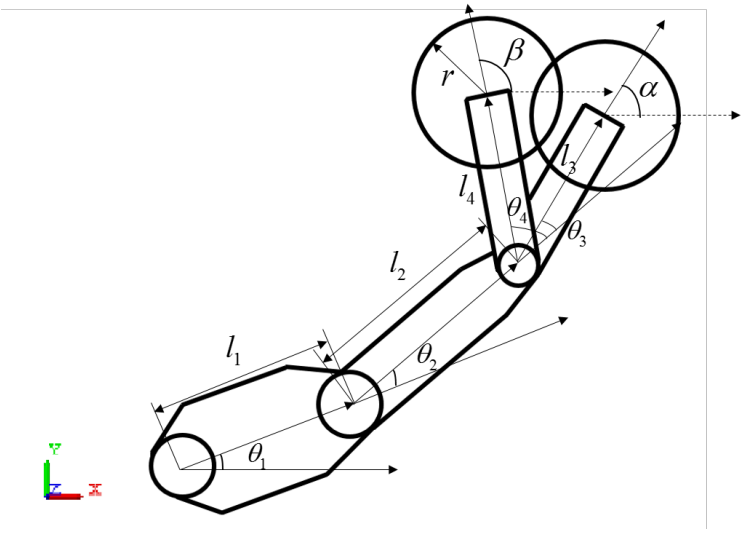


Figure 2.5: An illustration of the geometric parameters of the kinematics of the wafer handling robot.

$$\begin{bmatrix} v_x \\ v_y \\ v_z \\ \omega_\alpha \end{bmatrix} = \begin{bmatrix} -l_1\sin\theta_1 & -l2\sin\theta_{12} & -l3\sin\theta_{123} & 0 \\ l_1\cos\theta_1 & l_2\cos\theta_{12} & l_3\cos\theta_{123} & 0 \\ 0 & 0 & 0 & 1 \\ 0 & 0 & 1 & 0 \end{bmatrix} \begin{bmatrix} \dot\theta_1 \\ \dot\theta_{12} \\ \dot\theta_{123} \\ \dot h \end{bmatrix},$$
$$\Rightarrow J := \frac{d(v_x, v_y, v_z)}{d(\dot\theta_1, \dot\theta_2, \dot\theta_3, \dot h)} = \begin{bmatrix} -l_1\sin\theta_1 & -l2\sin\theta_{12} & -l3\sin\theta_{123} & 0 \\ l_1\cos\theta_1 & l_2\cos\theta_{12} & l_3\cos\theta_{123} & 0 \\ 0 & 0 & 0 & 1 \end{bmatrix} \begin{bmatrix} 1 & 1 & 1 & 0 \\ 0 & 1 & 1 & 0 \\ 0 & 0 & 1 & 0 \\ 0 & 0 & 0 & 1 \end{bmatrix}, \tag{2.5}$$

$$\begin{bmatrix} a_x \\ a_y \\ a_z \\ \gamma_\alpha \end{bmatrix} = \begin{bmatrix} -l_1\sin\theta_1 & -l_2\sin\theta_{12} & -l_3\sin\theta_{123} & 0 \\ l_1\cos\theta_1 & l_2\cos\theta_{12} & l_3\cos\theta_{123} & 0 \\ 0 & 0 & 0 & 1 \\ 0 & 0 & 1 & 0 \end{bmatrix} \begin{bmatrix} \ddot{\theta}_1 \\ \ddot{\theta}_{12} \\ \ddot{\theta}_{123} \\ \ddot{h} \end{bmatrix} + \begin{bmatrix} -l_1\cos\theta_1 & -l_2\cos\theta_{12} & -l_3\cos\theta_{123} & 0 \\ -l_1\sin\theta_1 & -l_2\sin\theta_{12} & -l_3\sin\theta_{123} & 0 \\ 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 \end{bmatrix} \begin{bmatrix} \dddot{\theta}_1 \\ \dddot{\theta}_{12} \\ \dddot{\theta}_{123} \\ \dddot{h} \end{bmatrix}. \tag{2.6}$$

The positioning of the camera, as demonstrated in Fig. 2.2, is right behind the tip of the robot arm at blade 2. The wafer points to the same direction of the blade or tilts downward by angle β_c. By using the conventional coordinate system of the camera (Z axis of camera points forward), the end effector to camera transformation is then

$$T_{e2c} = \begin{bmatrix} 0 & -1 & 0 & 0 \\ \sin\beta_c & 0 & -\cos\beta_c & d\sin\beta_c + h\cos\beta_c \\ \cos\beta_c & 0 & \sin\beta_c & d\cos\beta_c - h\sin\beta_c \\ 0 & 0 & 0 & 1 \end{bmatrix}. \tag{2.7}$$

The dynamics of the robot is derived by Euler-Lagrangian equation as (2.8)

$$\begin{bmatrix} 1 & 1 & 1 & 0 & 0 \\ 0 & 1 & 1 & 0 & 0 \\ 0 & 0 & 1 & 0 & 0 \\ 0 & 0 & 0 & 1 & 0 \\ 0 & 0 & 0 & 0 & 1 \end{bmatrix} \left[\frac{d}{dt} \left(\frac{\delta L}{\delta \dot{q}'} - \frac{\delta L}{\delta q'} \right) \right] = \tau, \tag{2.8}$$

where $q' := \begin{bmatrix} \theta_1 & \theta_{12} & \theta_{123} & \theta_{124} & h \end{bmatrix}^T$ is the generalized coordinates transformed from $q' := \begin{bmatrix} \theta_1 & \theta_2 & \theta_3 & \theta_4 & h \end{bmatrix}^T$ and τ is the vector of torques applied to JT2, JT4, JT6, JT7 and JT3 of the robot.

$$\begin{aligned} L :=& V_1\left(\dot{\theta}_1\right) + V_2\left(\theta_1, \theta_{12}, \dot{\theta}_1, \dot{\theta}_{12}\right) + V_3\left(\theta_1, \theta_{12}, \theta_{123}, \dot{\theta}_1, \dot{\theta}_{12}, \dot{\theta}_{123}\right) + \\ & V_4\left(\theta_1, \theta_{12}, \theta_{124}, \dot{\theta}_1, \dot{\theta}_{12}, \dot{\theta}_{124}\right) + V_w\left(q', \dot{q'}\right) + V_{cam}\left(q', \dot{q'}\right), \end{aligned} \tag{2.9}$$

where V_1, V_2, V_3, V_4 are the kinetic energy of the four links. And V_w, V_{cam} are the kinetic energy of wafers and the camera. The detailed dynamic equation is skipped here because it is too complicated. During the implementation, it is derived by the symbolic math toolbox in MATLAB. The torque calculated from the dynamic equation can be used to obtain the torque requirement of a trajectory given the joint space angle, velocity and acceleration reference. It is also used for the feedforward control of the trajectory to get better tracking performance.

2.3 Path Planning of Wafer Handling Robot

In order to navigate the workspace safely, the robot must plan trajectories without colliding with other objects. In order to navigate the workspace efficiently, trajectories should be optimized. However, finding a collision-free and time-optimal trajectory at the same time is very difficult. Optimization-based trajectory planning methods highly depend on the initial condition [9, 10, 11]. It is likely that some undesirable situations may trap the robot in an infeasible solution. Many trajectory planning methods that start from scratch [12, 13, 14] take the two-stage framework which separates finding the feasible path and optimizing the final trajectory to reduce the complexity of trajectory planning problem. In this section, we introduce the path planning method which is widely used for finding the feasible path.

Search-based Path Planning in Discretized Joint Space

The joint space, or configuration space [15], contains all possible geometric states of the robot. One approach to solve the path planning problem is by discretizing the configuration space and searching for a feasible path among a finite number of paths in the discretized space. The advantage of search-based method is that the computational load is bounded and predictable once the resolution is decided. The main disadvantage is the so-called curse of dimensionality. The mesh points required for an n-dimensional space is exponential to the dimensionality. Discretization in a high dimensional space is intractable due to the exponentially growing worst case number of points needed to examine. Therefore, it is only practical to implement it in the 4 to 5 dimensional path planning in joint space of the robot. The path planning problem of the wafer handling robot is at the margin of using the discretization.

The A* algorithm [16] is a typical search-based and is used in this work. The common procedure of the algorithm is first determining from which child the tree should expand and then to which direction. The most important components of the A* are the closed and open sets and heuristic function. The open set stores all the children in the tree that can be expanded. On the other hand, closed set stores the nodes in the tree that should not be examined. The closed and open sets ensure no loop or revisit is made during the search. The heuristic function assigns a value to all the discrete points in the configuration space, which guides the choice of child and direction of expansion. The heuristic function reflects the prior knowledge from the planning problem from the user. When there is no obstacle and h and cost are both assigned as Euclidean distance, then A* will give a straight line immediately. In fact, when the heuristic function coincides with the true cost from each node to the goal point, then A* will return the shortest path with highest efficiency. The more user knows about the problem, the more efficient A* will be. An admissible heuristic function will make A* return the shortest path. When the cost is Euclidean distance, a Euclidean distance heuristic function is always admissible.

Experimental Results using A* Path Planning Algorithm

The A* is implemented in the joint space with the cost and heuristic functions assigned as the Euclidean distance in the joint space. If the computation time is limited to 2-3 minutes in MATLAB, the path planning is guaranteed to finish with a 1-degree resolution when only 3 joints are considered. Planning in the 5-dimensional space requires a much coarse mesh of around 5-degree resolution. The feasibility of each node w.r.t. collision can be pre-calculated by checking the collision at each configuration. The initial open set is composed of only the start point. The initial closed set contains all the joint angle combinations where the robot interferes with obstacles. In order to increase the efficiency of the algorithm, the open set and closed set are also stored as properties in each node. When the algorithm needs to know whether a node is in the open set, the closed set or neither of them, it refers to the matrix that stores the node's state instead of searching through the open set and closed set.

The A* algorithm is performed for the path planning between different configurations of the robot listed in Fig. 2.6. The figure demonstrates the four configurations used for A* path planning. Position is the fully retracted state. Their positions in the configuration space are shown in Fig. 2.7. The plot is of the 3-dimensional projection of the joint space with JT2, JT4 and JT6 coordinates. The surface in blue is the boundary of the free space.

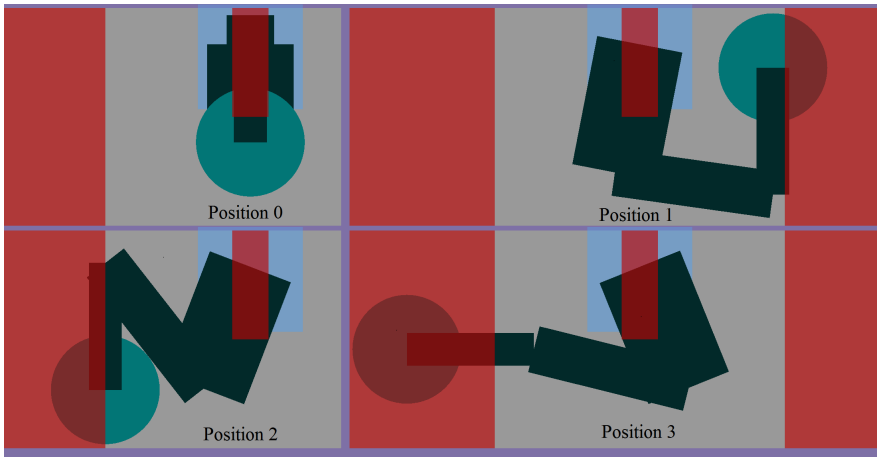


Figure 2.6: An illustration of the start and end points chosen for evaluation planning algorithms.

The curse of dimensionality is the largest problem faced by planning methods in discrete space. Some recent effort [17] has been made concerning the A* algorithm. Despite the shortcoming of planning in high dimensional space, smoothness is also a problem. The path output of the discrete planning method can be smoothed by fitting it with a spline curve. However, the sharp turns in the path will still have large curvature when fitted by a continuous curve, which causes more effort in robot dynamics.

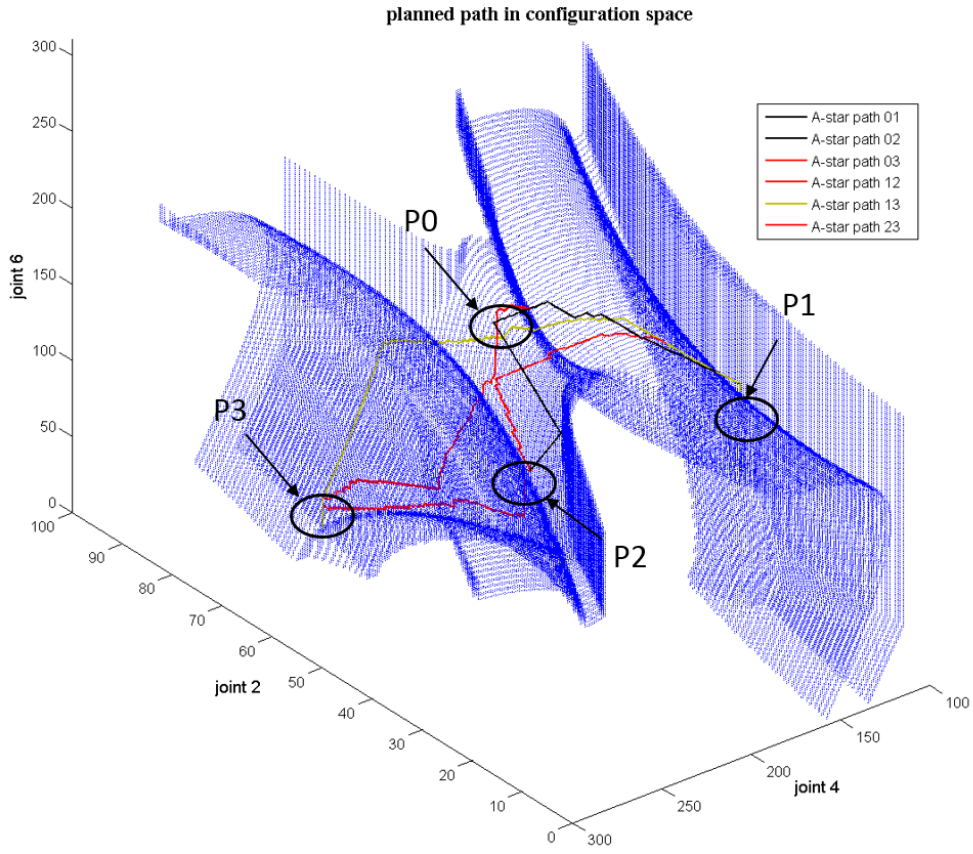


Figure 2.7: An illustration of the configuration space with A* planning results.

Another problem is the difference between optimality in discrete space and continuous space. Fig. 2.8 shows the shortest path in a discrete rectangular space with uniform mesh. Regardless of the planning algorithm, the discrete shortest path has a large deviation from the true shortest straight-line path in continuous space. This deviation usually happens when the moving distance in two coordinates differs a lot. The deviation is observed in the experimental results of the wafer handling robot where the planned paths are usually very close to the obstacle, causing safety concern although the path is theoretically feasible.

Sampling-based Path Planning in Continuous Joint Space

The sampling-based method samples for feasible points randomly and links the sampled point as a new node to expand the existing tree. The sampling step is very fast since it only needs to generate random values and check the feasibility, which easily generalizes to the high dimensional space. Moreover, the sampling is not restricted to a discretized space. The disadvantage of sampling-based methods is that there is no guarantee in finding a path and

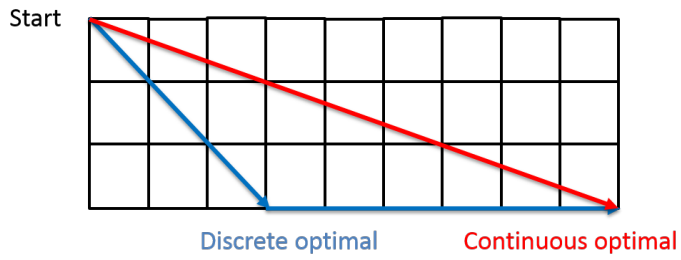


Figure 2.8: Difference between discrete shortest and continuous shortest path.

it does not find the optimal path in a finite time period. In this work the path planning in 5-dimensional joint space is done by the RRT* algorithm [18] because of their advantage in computational time in high dimensional space compared to the A* algorithm.

RRT* is a modified version of RRT which allows optimal planning. Optimal planning is realized by introducing a cost function to all nodes and an additional step after expanding the random tree. The edges are changed locally near the new sampled node if any node in the neighborhood can achieve a lower cost by linking from the new sampled node. The cost function can be the length of the path from the starting point. RRT* converges to a global optimal path in possibility, which means the possibility that the returned path has global minimum cost approaches one as the number of samples approaches infinity. RRT* does not stop when it finds a feasible path, but stops when the cost modification converges. RRT* is widely used in path planning and has the potential to be modified to become an optimal trajectory planning method [19, 20].

2.4 Time-optimal Trajectory Optimization of Wafer Handling Robot

The path planning result from Section 2.3 cannot be directly implemented to the robot since the time information is not included. Even if we can always parameterize the path with time, the generated geometric path does not serve as a good candidate for time-optimal trajectory. The path may have too large curvature which requires large acceleration. The robot will need to slow down when there is a sharp turn. Moreover, having all joints accelerating together in the same direction will result in a large equivalent inertia. This will result in a large torque input to maintain the acceleration which violates the torque limit. In order to get a trajectory that can be deployed to the robot, not only geometry, but also the kinematics and dynamics of the robot must be considered. The kinematic constraints are usually the velocity and acceleration bound. The dynamic constraints limit the control signals (usually torque).

Representation of trajectory

Some trajectory optimization methods solve the trajectory directly where the kinematic constraints are easier to be evaluated and satisfied, such as Trajopt [9] and CHOMP [11]. However, the control signal constraints involve the inverse dynamic function which is difficult to calculate. Other methods solve the control signals instead of the trajectory function so that dynamic constraints can be easily satisfied, such as the ILQR [21, 22]. However, the kinematic constraints require the forward simulation with the dynamic function which is complicated and not robust. In this work, we take the first strategy because there are more kinematic constraints and our objective is to minimize the execution time of running the trajectory.

Trajectories in this work are regarded as curves in joint space that are parameterized w.r.t. time. Piece-wise polynomials, specifically, B-spline curves are used to represent the trajectory. Compared to using control sequences [9, 10], this requires much less memory. The reference at arbitrary time can be easily and precisely calculated from polynomial functions with only tens of values stored. More specifically, the B-spline is used to represent trajectories.

Splines, which are piece-wise polynomial functions, are widely used in representing the trajectory because of their convenience for curve fitting. The popular s-curve and trapezoidal trajectory [23] used for the motion of manipulators are usually defined by a cubic or quantic spline. The continuity is controlled by the order of the polynomial. Since the space of a certain order of polynomials can have multiple basis, different types of splines with different basis functions bring various parameterization to the same group of trajectories. B-spline is a spline function defined by control points that do not lie on the curve as shown in Fig. 2.9. The parameters of a B-spline are the control points $P_1, P_2, \cdots P_n$ and knots $t_1, t_2, \cdots t_{n+k+1}$. k is the degree of the spline. Knots recursively defines the basis functions of B-spline by (2.10)

$$N_{i,k}(t) = \frac{t - t_i}{t_{i+k} - t_i} N_{i,k-1}(t) + \frac{t_{i+k+1} - t}{t_{i+k+1} - t_{i+1}} N_{i+1,k-1}(t), t \in [t_i, t_{i+k+1}]. \tag{2.10}$$

Control points serve as weights for the sum of basis functions which fully define a B-spline.

$$B(t) = \sum_{i=1}^{n} P_i \cdot N_{i,k}(t), \tag{2.11}$$

where n is the number of control points. A B-spline with n control points has segments $n-k$ which are polynomials of order k. When knots are not equal to each other, which is the common case, the order of continuity of a B-spline is equal to $k-1$. For example, a cubic B-spline is second order continuously differentiable. This means if time is chosen as knots of the cubic B-spline, then the acceleration will be continuous.

The most important property that makes B-spline suitable in this work is the local support property of its basis functions. The fact that basis functions have local support

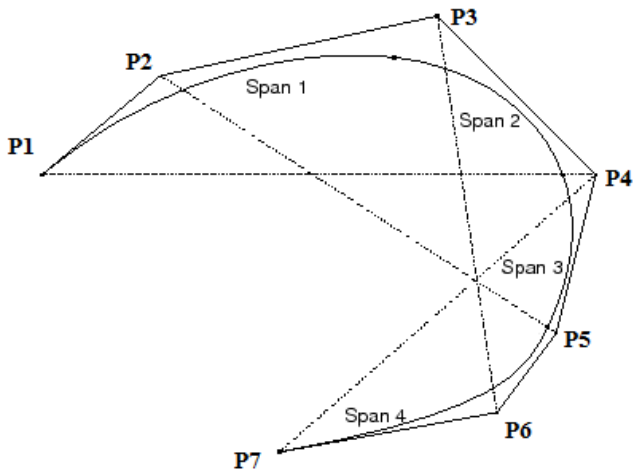


Figure 2.9: Illustration of a clamped cubic B-spline [24].

means that changing the parameters, namely control points and knots, will only change the value and derivatives of the spline in a small interval. For example, changing P_i in a cubic spline will only affect the values and derivatives in (t_i, t_{i+4}). This feature is highly preferred in the optimization as it creates a sparse gradient and hessian matrix.

The derivative of a B-spline is still a B-spline, but with one less degree. Therefore, the properties will all be preserved in the derivative function. When a trajectory is represented by a B-spline, the velocity, acceleration and jerk functions will be (2.12)

$$\begin{aligned} B'(t) &= \sum_{i=1}^{n-1} V_i \cdot N_{i,k-1}(t), V_i = \frac{k}{t_{i+k+1} - t_{i+1}} (P_{i+1} - P_i), \\ B''(t) &= \sum_{i=1}^{n-2} A_i \cdot N_{i,k-2}(t), A_i = \frac{k-1}{t_{i+k+1} - t_{i+2}} (V_{i+1} - V_i), \\ B'''(t) &= \sum_{i=1}^{n-3} R_i \cdot N_{i,k-3}(t), R_i = \frac{k-2}{t_{i+k+1} - t_{i+3}} (A_{i+1} - A_i). \end{aligned} \tag{2.12}$$

This means we can compute the "control points" for velocity, acceleration and jerk functions, which are $V - i, A_i, R_i$ respectively. The value of the function is bounded by these control points because of the convex hull property of B-spline. The convex hull property states that the curve will stay in the convex hull containing all control points. Note if cubic B-spline is used to represent the trajectory, the acceleration function will be a 1-degree B-spline, which

is a linear interpolation in (2.13) between the control points for acceleration

$$B''(t) = \sum_{i=1}^{n-3} \left[A_i (t - t_{i+3}) + A_{i+1} (t_{i+4} - t)\right] \cdot \mathbf{1}_{[t_{i+3}, t_{i+4}]}(t), A_i = \frac{k-1}{t_{i+k+1} - t_{i+2}} (V_{i+1} - V_i), \tag{2.13}$$

where $\mathbf{1}_{[a,b]}(t)$ is an indicator function.

We always want to fix the start and end points in a trajectory and sometimes we also want to specify the departure and arrival velocity and acceleration. However, general B-splines with strictly increasing knots always have 0 initial and end value. To take control of the initial and end value, some knots are fixed. Take cubic spline as an example, the following setting of knots in (2.14) will clamped the initial position, velocity and acceleration to (2.15). An exemplary clamped cubic B-spline is shown in Fig. 2.9.

$$\begin{aligned} t_1 &= t_2 = t_3 = t_4, \\ t_{n+1} &= t_{n+2} = t_{n+3} = t_{n+4}. \end{aligned} \tag{2.14}$$

$$\begin{aligned} B(0) &= P_1, B(T) = P_n, \\ B'(0) &= V_1 = \frac{3(P_2 - P_1)}{t_5 - t_1}, B'(T) = V_{n-1} = \frac{3(P_n - P_{n-1})}{t_{n+4} - t_n}, \\ B''(0) &= A_1 = \frac{2(V_2 - V_1)}{t_4 - t_2}, B''(T) = A_{n-2} = \frac{2(V_{n-1} - V_{n-2})}{t_{n+3} - t_{n+1}}. \end{aligned} \tag{2.15}$$

Formulating time objective trajectory optimization

The time optimal trajectory planning problem can be formulated to a nonlinear optimization problem. Here the optimization refers to optimizing both the shape and time scale of the trajectory instead of just parameterizing a path with fixed geometry. Many existing works fix the shape of the trajectory when doing optimization [12, 14], but the geometrically fixed initial trajectory does not provide a good candidate for time-optimal trajectory. Others [9, 10, 11] optimize the shape and time scale simultaneously but start with a geometrically infeasible trajectory. Optimization-based methods are easily affected by the initial condition, which may result in an infeasible local optimal output. Therefore, choosing a feasible initial condition for optimization is important. In this thesis, the initial condition is provided by the proposed path planning method so as to prevent the trajectory from being trapped by infeasible solutions.

The optimization problem is formulated as in (2.16), where g denotes the nonlinear inequality constraint and h denotes the nonlinear equality constraint. The dimension of the joint space is $D = 5$, the number of control points is n and the order of the B-spline is k. All the parameters that define a B-spline function should be optimization variables since we want to have free control of both the geometric shape and time consumption of the

trajectory. The difference between adjacent knots are defined as $\Delta t_i := t_i - t_{i-1}$. Using Δt_i can ensure the monotonicity of knots is always satisfied during optimization.

$$
\begin{aligned}
\text{minimize} & \sum_{i=1}^{n+4} \Delta t_i \\
\text{over } x = & \left(P_1^{(1)}, \cdots P_n^{(1)}, \cdots P_1^{(D)}, \cdots P_n^{(D)}, \Delta t_1, \Delta t_2 \cdots, \Delta t_{n+k+1} \right)^T \in \mathbb{R}^{K \times n+n+k+1} \\
\text{subject to } & g_{collision}(x) \leq 0 \\
& g_{acc}(x) \leq 0 \\
& g_{jerk}(x) \leq 0 \\
& g_{torque}(x) \leq 0 \\
& h_{acc}(x) = 0 \\
& \mathbf{A}_{velocity} \cdot x = (\mathbf{v}_{start}, \mathbf{v}_{end}) \\
& \Delta t_2 = \Delta t_3 = \Delta t_4 = 0 \\
& \Delta t_{n+k-1} = \Delta t_{n+k} = \Delta t_{n+k+1} = 0 \\
& \mathbf{P}_1 = \mathbf{P}_{start} \\
& \mathbf{P}_n = \mathbf{P}_{end} \\
& \Delta t_i \geq 0, i = 2, \cdots, n+k+1 \\
& \mathbf{A}_{velocity} x \leq \mathbf{v}_{max}.
\end{aligned}
\tag{2.16}
$$

The linear equality constraints with matrix $\mathbf{A}_{velocity}$ is to guarantee that the departure and arrival velocity are the desire ones $\mathbf{v}_{start}$ and $\mathbf{v}_{end}$. The nonlinear constraint $g_{collide}$ uses the signed distance defined in [10] between each part of the robot arm and each obstacle.

$$
g_{collide} := d_{safe} - min\left\{ d_{signed}(A, B) \,|A \in \text{robot arm}, B \in \text{obstacles} \right\}, \tag{2.17}
$$

where d_{safe} is the clearance specified to be kept between the robot and obstacles. g_{acc}, g_{torque} uses the second and third derivative of B-spline curves defined in (2.12) to set the limit of acceleration and jerk to $\mathbf{a}_{max}$ and $\mathbf{j}_{max}$. h_{acc} uses the second derivative of B-spline curves to set the start and end acceleration to $\mathbf{a}_{start}$ and $\mathbf{a}_{end}$. The computationally heavy work is in calculating the gradients of these nonlinear constraints and it is done by the symbolic math toolbox.

Experimental Results and Discussion

The optimization is solved by the MATLAB Optimization toolbox with the non-linear optimization function "fmincon". The optimization time and the resulting optimal execution time of trajectories are listed in Table 2.1. the optimized trajectories in joint space are compared to the A* planned path in Fig. 2.10. The dashed curves are the paths planned by

A* algorithm and the solid curves are the trajectories after optimization. The time-optimal trajectories are much smoother in the joint space

Trajectory name	Optimization time(s)	control points	Trajectory time(s)
traj01	10	14	1.134
traj02	5	10	0.869
traj03	25	18	1.440
traj12	26	18	1.469
traj23	15	16	1.271

Table 2.1: Optimization time and resulting trajectory running time.

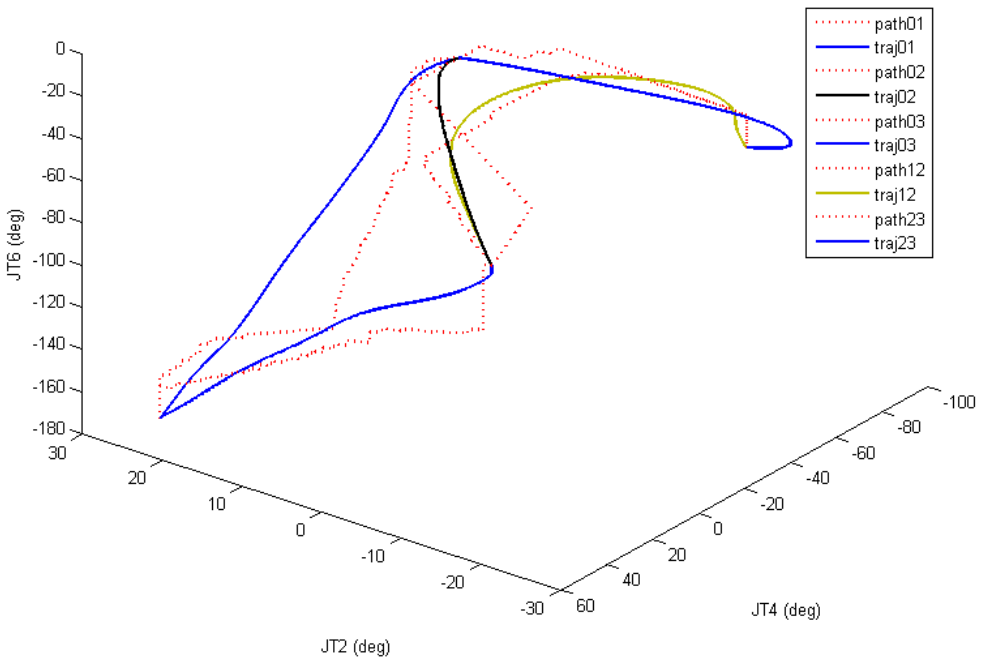


Figure 2.10: Trajectory optimization result in joint space.

The optimality is verified in the time plot of traj02 shown in Fig. 2.11. Note that the lower bound of the time consumption of a trajectory is the maximum time consumption of trapezoidal trajectories for each joint. If one joint is following a trapezoidal trajectories that saturates at each time step, then the trajectory is globally time-optimal.

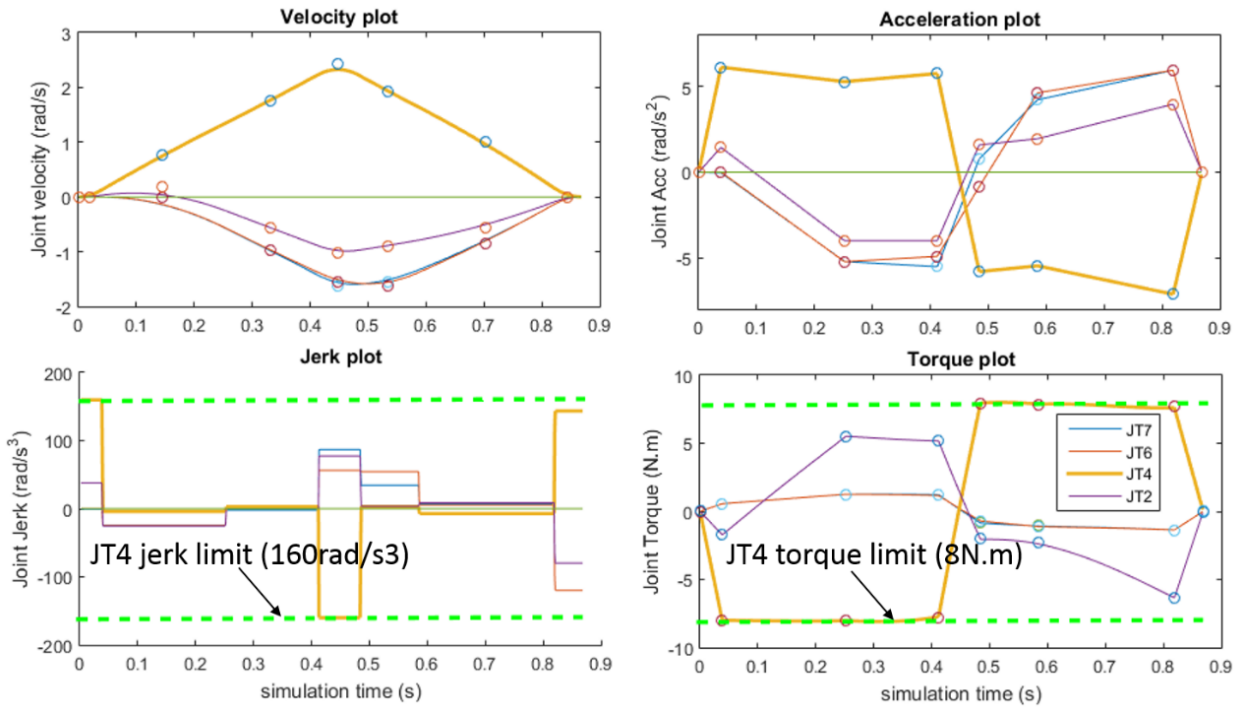


Figure 2.11: Optimal traj02 with zero end velocity

The Fig. 2.12 shows the result when traj02 is given a non-zero departure velocity at JT4. In this case, JT4 moves fast at the beginning so the robot very quickly leaves the region that is vulnerable to collision. Other joints are allowed to move at the beginning since JT4 is no longer the limiting factor of the trajectory. JT6/7 become the ones that possess the longest distance. Here they are following a trapezoidal trajectory of acceleration because torque requirement never reaches its limit. This is the global time-optimal trajectory in this case.

The time-optimal trajectory optimization is applied to optimize a wafer handling process which contains two fixed way-points between two FOUPs so that there are three separate trajectories to be designed. Although we cannot change the location of way-points, we can change the velocity at both way-points. We show that the total execution time of running these three trajectories can be reduced by optimizing the trajectories with different connecting velocity settings in Table 2.2. The first row is the original velocity setting. It can be seen from the table that the overall running time can be reduced with the proposed method.

The future work is to improve the global optimality of trajectory planning methods. Most optimization methods only converge to locally optimal solutions. Recent topological path

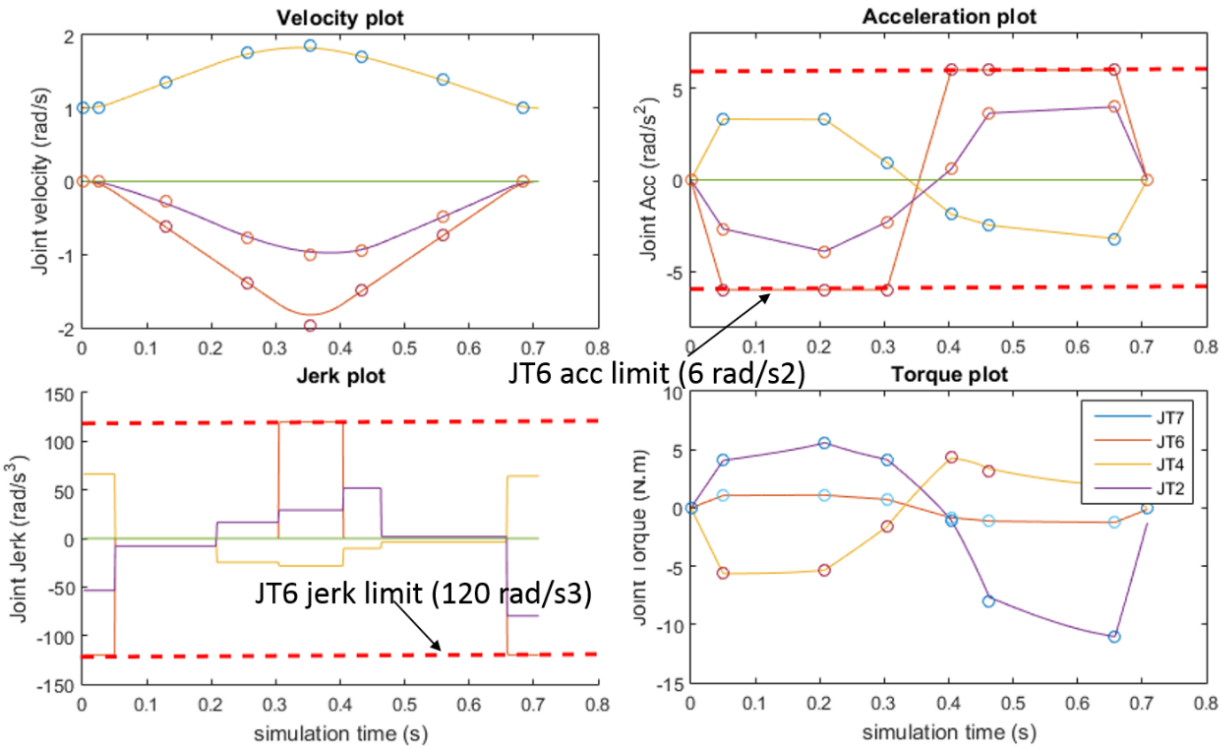


Figure 2.12: Trajectory optimization result in joint space.

planning methods [25] provides all the topologically inequivalent feasible paths that have the potential of becoming a global time optimal solution. Time optimal RRT* trajectory planning [19, 20] provides another approach for global optimal solution.

2.5 Tracking Controller Tuning for Hand Stabilization

The vibration of the robot hand is very important for the wafer handling robot, not only because the wafer should be transferred smoothly [26], but the camera should also be held steadily to ensure the image quality. Although the trajectory is designed in Section 2.4 to be smooth, significant vibration can still occur due to the motor controller. The low-level controller of the wafer handling robot is studied in this section.

Dynamic Models of Joints

There are 5 motors that actuate the robot as stated in Section 2.2. High Frequency response is the property that is directly related with vibration. However, since the robot dynamics is

Velocities at way-points	Length of middle trajectory(s)	Length of 3 trajectories(s)	Optimization time(s)
$v_1 = [0, 0, 0, 0, 0]$ $v_2 = [0, 0, 0, 0, 0]$	1.623	3.123	40
$v_1 = [0.39, 0.39, -0.55, 0.16, 0]$ $v_2 = [0, 0, 0, 0, 0]$	1.591	2.871	89
$v_1 = [0, 0, 0, 0, 0]$ $v_2 = [0.41, 0.41, -0.55, 0.14, 0]$	1.648	3.008	29
$v_1 = [0.39, 0.39, -0.55, 0.16, 0]$ $v_2 = [0.41, 0.41, -0.55, 0.14, 0]$	1.615	2.755	67

Table 2.2: Optimization time and resulting trajectory running time.

non-linear, the frequency responses of the motors will change at different robot poses. The low-level controllers, which need online calculation, are required to be fixed linear filters in the robot. Therefore, the dynamics of the joints are modeled as multi-modal linear transfer functions and the fixed structure controller of a joint is designed to work for all transfer functions. Each transfer function is identified at a robot pose using sinesweep and the data is collected at multiple poses to cover the range of different dynamics. Three joints, namely JT2 and JT4 are identified and the frequency responses are shown in Fig. 2.13. For each frequency response, we fit a transfer function that captures all the resonances with the "tfest" function in MATLAB.

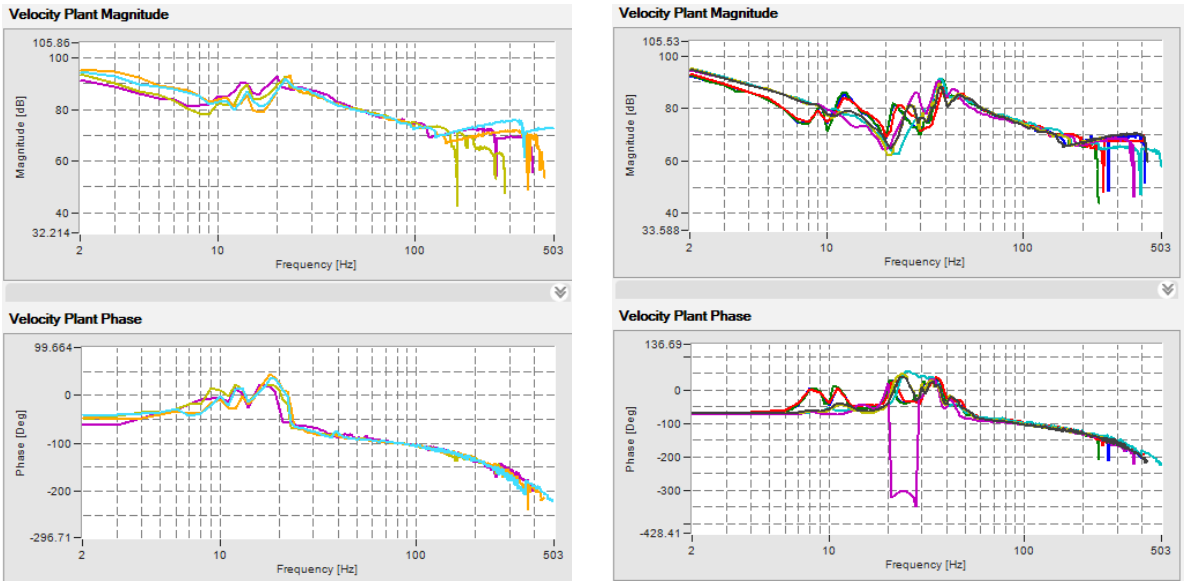
Controller Tuning Algorithm

The structure of the joint motor controller is a linear filter whose order is not restricted followed by a PID position controller. We need to design the parameters of the linear filter and the PID controller together. The controller tuning framework is shown by Fig. 2.14. With the fitted transfer functions, an initial guess is made by the classic loop-shape method using the "hinfsyn" function in MATLAB. In the loop shaping, we try to shape the open loop response to a desire transfer function:

$$G_0(s) = \frac{w_c}{s(\frac{1}{w_1}s + 1)(\frac{1}{w_2}s + 1)^2}, \tag{2.18}$$

where w_c is the desired crossover frequency, w_1, w_2 are the desired rolloff frequencies. They are all empirical parameters but are very intuitive and easy to be determined during implementation.

After deriving the initial guess of the controller, the nonsmooth H_{inf} [27] optimization [28] is used to optimize the linear filter and PID gains. This method can take multi-modal transfer



(a) JT2. (b) JT4.

Figure 2.13: Frequency responses of JT2 and JT4.

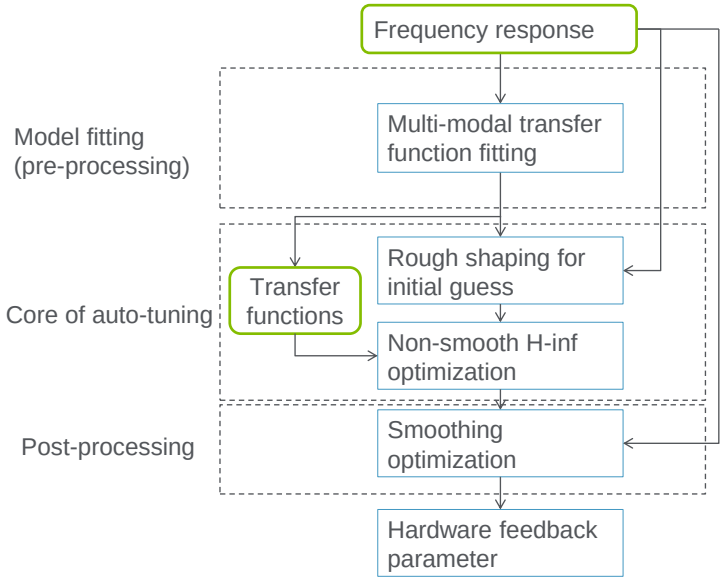


Figure 2.14: Illustration of controller tuning framework.

function as input and let the filter satisfy all the constraints while optimizing the control

performance in frequency space, which suits our purpose of suppressing the resonances of vibration. The optimization objective is

$$|S_i| < |S_d|, i = 1, 2, \cdots, N, \tag{2.19}$$

where S_i's are the closed-loop sensitivity functions corresponding to the fitted N open-loop transfer functions and S_d is the desired sensitivity function shown in Fig. 2.15. The constraints are listed as below. The first constraints guarantee the robustness and the other three constraints force the optimization to reduce the resonances of the closed loop system.

1. The gain margin is at least 12dB and phase margin is at least 50 degrees.
2. The dampings of closed-loop poles are at least 0.1.
3. The maximum open-loop gain is upper bounded by M between frequency range $[f_0, f_1]$.
4. The open-loop response decays by second order after frequency f_2.

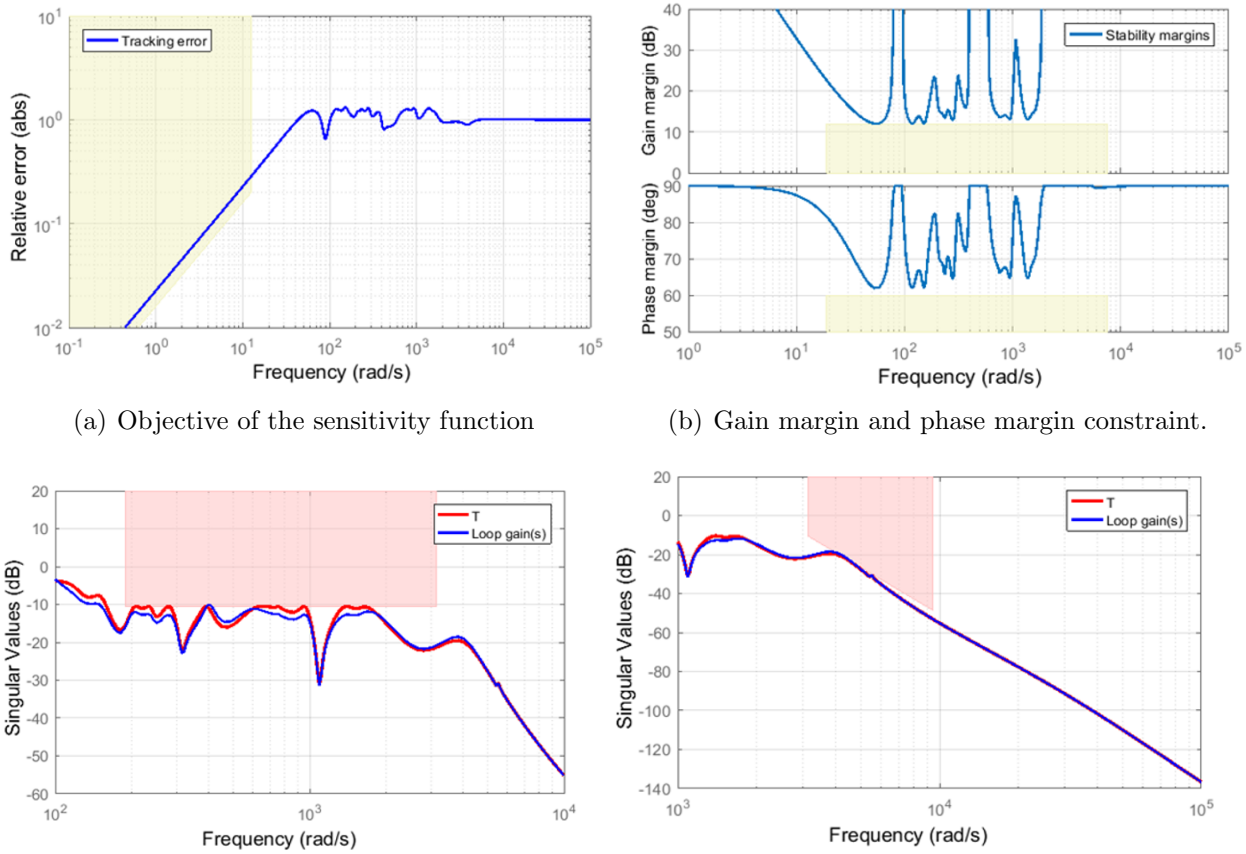
The resulting objective and constraints are shown in Fig. 2.15, together with the optimization results.

The controller designed by the proposed method is compared with the controller used in the factory which is tuned by the engineer. The robustness result is shown in Table 2.3. Notice that the bandwidth (the larger the better) of both joints are increased significantly with a small sacrifice of stability margins (the larger the better). JT4 even has a larger stability margin than the original one, which means both the tracking performance and robustness are improved with the proposed controller tuning method. The final vibration

JT2	Bandwidth(Hz) of	Phase margin(deg)	Gain margin(dB)	filter order
original	2.7	**62.9**	**19.6**	2
proposed	**8.4**	60.0	13.0	4
JT4	Bandwidth(Hz) of	Phase margin(deg)	Gain margin(dB)	filter order
original	10.5	51.3	9.8	2
proposed	**19.4**	**54.5**	**10.0**	6

Table 2.3: Crossover frequency and robustness results of proposed controller compared with the factory setting.

measurement at the robot hand is shown in Table 2.4. The comparison is also made with the factory setting used in production. The root mean square of accelerations at all directions are reduced which means the vibration is suppressed with the proposed controller.



(a) Objective of the sensitivity function

(b) Gain margin and phase margin constraint.

(c) Maximum open-loop gain constraint.

(d) Open-loop decay rate constraint.

Figure 2.15: Optimization requirements of JT2.

2.6 Chapter Summary

This chapter introduces the kinematic and dynamic model of a hand-eye system, which refers to a visual sensor mounted on a robot arm. In order to form a closed-loop structure for the hand-eye system, the planning and control modules are designed based on the model. The two-stage planner uses path planning to generate collision-free path and trajectory optimization to optimize the execution time. The proposed tracking controller stabilizes and minimizes the vibration of the hand. The trajectory planner and controller together enable the hand-eye system to navigate smoothly in the workspace. The proposed motion block is essential for precise perception and calibration of the hand-eye system.

RMS of acc	Vertical(Z) of	horizontal(Y)	side-to-side(X)
original	1.93	1.34	1.52
original	**1.82**	**1.30**	**1.50**

Table 2.4: Vibration results measured at the hand center of the robot. The acceleration is measured by the IMU mounted on the hand and the root mean square (RMS) of acceleration is used as the evaluation metric. Note smaller RMS of acceleration means less vibration.

Chapter 3

Robust Detection of Objects in the Robot Workspace

The robot detects objects in the image or point cloud captured by the vision sensors to perceive the surrounding environment. Detection returns the segmentation and location, either in image or 3D space, of target objects. The detection problem is a classic and fundamental problem in computer vision [29]. Feature descriptor and matching methods have been developed and used for a long period. Recently, learning-based detection techniques has become realistic in many robotic systems for generic object detection in recent years [30]. Different from the general object detection in computer vision which mostly focuses on extracting semantic meanings, industrial applications focus more on precision and robustness. Target objects in the industrial workspace usually have known shapes or meshes but their locations are unknown. It is desired to guarantee a almost 100% accuracy and recover the location of the target object through precise geometric calculation. In this case, recent learning-based methods are less preferred.

Consideration of robustness in detection:

- Coverage(thoroughness): Global search or matching
- Outlier rejection (both hard and soft)
- Redundancy: Evaluation in post-processing, multi-sensors, multi-frame. If we want to compare the difference with a controller, model uncertainty is usually not what we considered.

In this chapter, we focus on developing algorithms for robust detection of objects with known shape or mesh. Target objects are selected to be typical targets in the semiconductor industry but proposed methods can be extended to other objects with known CAD model. Section 3.1 reviews classic computer vision methods related to accurate feature matching and shape matching. Section 3.2 introduce a feature matching method for wafer detection which is based on elliptic segment detection. Section 3.3 introduces a shape matching algorithm

for the detection of objects which are the assembly of basic shape features. This algorithm is developed for the openings of chambers in FI. Section 3.4 introduces a robust global voting algorithm for matching-based methods like point registration or the one introduced in Section 3.3.

3.1 Introduction

Feature Detection in Images

Extracting interesting features from the image is a fundamental and broad research area in visual detection. Features range from the basic point (pixel) to a local region (patch) of the images [31] and there are separate research areas for each kind of features [32]. The pixel-level features are extracted by the edge detector [33] which calculates gradients of the points and extracts edgels. It provides point feature such as pixel location and gradient direction but does not contain shape information. The path level feature is a set of pixels that contains certain geometry or texture information and is extracted by contour detectors [34] or local feature descriptors [35]. Local feature descriptors are usually used for clustering general features and group them into different categories. Contour detectors focus more on finding contours which belong to a certain shape or template that we are interested in. In this work, we focus on the contour detection because the target objects in industrial applications are already known. In particular, objects that we are interested in are wafers and chamber openings in the semiconductor industry. Their shapes are precisely manufactured and the CAD models are composed of basic geometric elements such as lines and ellipses. For this kind of contour detection problem, two steps are taken. Firstly, line, corner and ellipse detection are performed to extract these basic elements that assemble the shape. Secondly, these elements are matched with the template shape of the target object so as to segment out the contour of the object in the image.

Wafers are thin cylinders as shown in Fig. 3.2 and the boundary of a wafer becomes ellipses in the image. Therefore, detection of wafers mainly relies on detecting elliptic shapes. 2D Ellipse detection is still an ongoing topic in computer vision as it serves as the basis for detecting circular objects [36, 37]. Given an edge image, there is no fast and exhaustive search method for ellipses as opposed to circles and lines. An ellipse or elliptic segment has a 5-dimensional parametric space, and thus it cannot be easily detected by the classical Hough transform or RANSAC [38, 39]. In arc-based ellipse detection methods [37, 40], it is assumed that most points in the edge images are connected and can be assembled into smooth arcs. This significantly reduces the complexity as the basic element becomes the arc which consists of hundreds of points. All the previous work, however, requires the ellipse to be relatively complete which means most parts of the ellipse should be visible. This is not the case for our problem in wafer detection where only less than half of the wafer is visible at most times. Observing less parts sets a higher requirement for detectors because more outliers might be included. We modified the ellipse detection framework in Section 3.2 to

form an elliptic segment detector that extracts and assembles elliptic-shape curves belonging to the same wafer. The experimental results are concluded in Section 3.5.[1]

Shape matching is a widely studied solution for detecting contours with known or parameterized shape. It matches the geometric shape of objects with the observation and numerous algorithms were developed [42]. Detection of all the objects in the image we discussed in this chapter falls into this category. After basic elements of the object are detected, we need to match them with the given shape. Several problems occur in finding the correspondence including

- Metric definition: Since we are matching instances, a distance function should be defined describing the similarity.
- Optimization: With the defined metric, we should formulate the optimization problem and efficiently solve the correspondence.
- Decision: Outlier is a critical issue and we should decide whether to accept or reject the matching.

Distance functions between shapes are proposed in many papers and the most popular one is the geometry-based shape context [43], Hausdroff distance and the voting-based shape consensus [44]. Some work finds matching that minimizes the distance by searching [45] or sampling [46] globally in the parameter space. Some work solve the optimization problem by gradient-base solver or EM algorithm which is rather local [43, 47]. In this chapter, voting-based scheme is taken to assemble the basic features in Section 3.3.

Point Registration for Point Clouds

Some robot systems are equipped with RGBD cameras or LiDAR scanners which provide 3D point clouds besides the image. Even without the direct 3D measurement, an RGB camera may produce point clouds using multi-view reconstruction or edge detection. In addition, some target objects are represented by mesh or a set of points instead of a continuous manifold. In this case, point registration method is preferred rather than feature detection in images. This section considers the point registration for shape matching problem in the industrial scenarios, where there could be multiple interested objects and non-uniformly distributed outliers presented in one scan, which is also called point registration in cluttered scenes in some references [48].

Point registration segments the points belong to the target object by predicting a binary mask on all measured points and locates the object by solving the transformation between the target object model and the detected object. The transformation aligning two sets of points can be rigid or nonrigid. The nonrigid transform is applied when the object is deformable, usually as an assembly of parts. It is pointed out that in most engineering applications, the rigid and non-rigid registration share the same idea of minimizing deviations

[1] This work includes materials from the author's previously published paper [41].

from isometry [49]. However, the non-rigid registration is considered to be more complex since the parameterization of the transformation is unknown and hard to model. Over the past few years, different approaches have been developed to address both transformations. A sophisticated survey of methods for general registration problems before 2013 is [50]. More recent works focus mainly on two branches faced by practical applications nowadays. One is the computational complexity and robustness discussed in [51, 52, 53, 54]. The other is the initialization of correspondence which is important for part-in-whole problems and handling noise and outliers, including [55, 56, 57, 58].

We propose a robust global initialization algorithm for locally optimal point registration methods in Section 3.4. The initialization is based on the rigid transform voting of all point locations and norm between two sets of points. A discretization-based clustering algorithm is developed for efficient clustering on the pairings of points, addressing both rotation and translation. Different from point feature-based initialization, the global voting considers the overall geometry consensus instead of relying on unique local structures which is uncommon in industrial cases. The problem originates from detection in a 2D edge image and generalizes to the detection in 3D point cloud reconstructed from depth image.

3.2 Elliptic Segment Detection of Wafers

2D Ellipse detection is a popular topic in computer vision as it serves as the basis for detecting circular objects [36, 37]. Given an edge image, there is no fast and exhaustive search for ellipses as opposed to circles and lines. Although ellipse is a basic shape in geometry, it has a 5-dimensional parametric space compared to circles and lines whose parametric space is 2-dimensional, and thus cannot be easily detected by the classical Hough transform or RANSAC [38, 39]. Most state-of-the-art real-time methods are arc-based [37, 40] which assume most points in the edge images are connected and can be assembled into smooth arcs. This significantly reduces the complexity as the basic element becomes an arc consisting of hundreds of points. Arc-based methods commonly have three stages. The first stage extracts arcs from the edge image and filters out short and non-convex arcs. The second stage rejects outlier-arcs by the geometry constraint of ellipses and finds potential combinations of arcs that form an ellipse. Then parameters of ellipses are estimated from combinations of arcs using various fitting algorithms [36]. The third stage validates estimated ellipses and merge arcs that belong to the same ellipse. The framework is shown in Fig. 3.1. The detection of elliptic-shape wafers is slightly different from the scenario faced by papers introduced above. From Fig. 3.2, it can be seen that only less than half of the boundary is visible for a wafer in the FOUP. Therefore, the edge is rather called an elliptic segment instead of a whole ellipse. As a result, algorithms designed for the second stage of arc-based methods do not apply to wafers. Being elliptic segment also means the estimation error is more prone to noise and stricter outlier rejection is required. On the other hand, with prior knowledge on the size and rough initial pose, better algorithms can be designed for robust wafer detection. The framework of this paper is based on [40]. New algorithms, however,

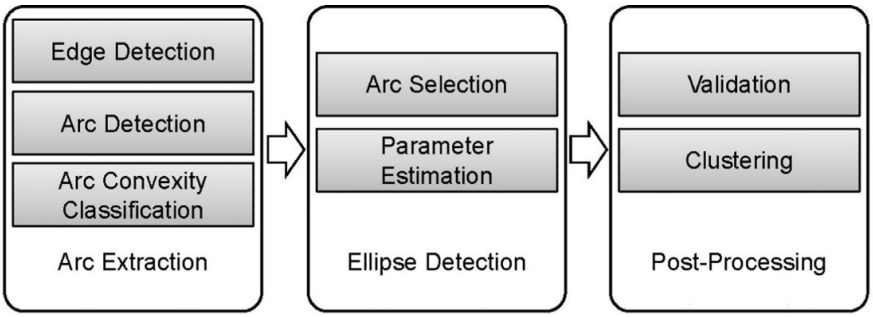


Figure 3.1: Ellipse detection framework in references [37, 40].

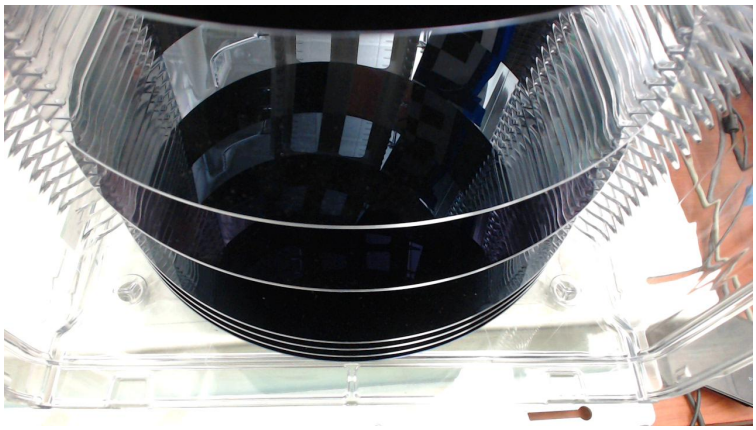


Figure 3.2: Wafers in the carrier (FOUP). The surface of the wafer is reflective. The bright boundary of the wafer forms two edges: a upper one corresponding to the upper surface and lower one corresponding to the lower surface

are developed and introduced in the following sections, including curvature estimation for outlier rejection and arc merging for elliptic segment validation. The process is summarized below and demonstrated in Fig. 3.3. We only discuss our modified parts in detail. Other parts that are not the contribution of this paper will only be given the reference.

Problem Statement

There are several expressions for an ellipse in 2D plane. Consider the ellipse shown in Fig. 3.4 below, one expression is the implicit form (3.1). A second expression is the explicit form of points (3.2). Furthermore, another expression is the quadratic form in the homogeneous coordinate of the general conic curve (3.3).

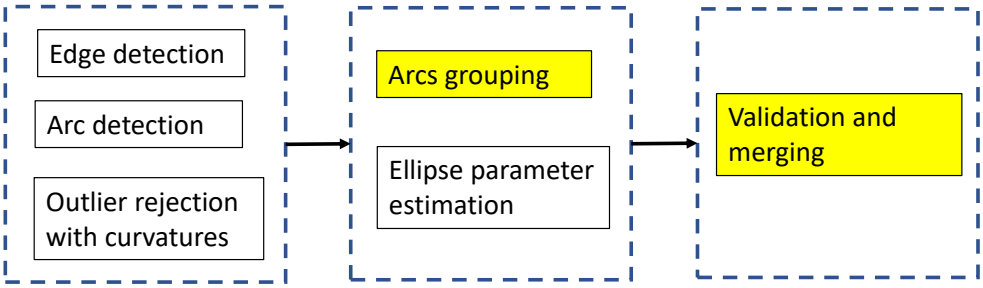


Figure 3.3: Framework of robust wafer detection method. Yellow frames are the modified part.

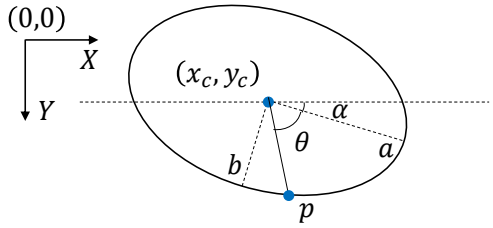


Figure 3.4: Illustration of 2D ellipse in XY plane .

$$\frac{[(x-x_c)\cos\alpha+(y-y_c)\sin\alpha]^2}{a^2}+\frac{[(x-x_c)\sin\alpha-(y-y_c)\cos\alpha]^2}{b^2}=1, \tag{3.1}$$

$$p:=\begin{bmatrix} x \\ y \end{bmatrix}=\begin{bmatrix} x_c \\ y_c \end{bmatrix}+\begin{bmatrix} \cos\alpha & -\sin\alpha \\ \sin\alpha & \cos\alpha \end{bmatrix}\begin{bmatrix} a\cos\theta \\ b\sin\theta \end{bmatrix}, \tag{3.2}$$

$$\begin{bmatrix} x \\ y \\ 1 \end{bmatrix}^T \underbrace{\begin{bmatrix} 1 & 0 & -x_c \\ 0 & 1 & -y_c \\ 0 & 0 & 1 \end{bmatrix}^T \begin{bmatrix} Q^T & 0 \\ 0 & 1 \end{bmatrix}\begin{bmatrix} a^{-2} & 0 & 0 \\ 0 & b^{-2} & 0 \\ 0 & 0 & -1 \end{bmatrix}\begin{bmatrix} Q & 0 \\ 0 & 1 \end{bmatrix}\begin{bmatrix} 1 & 0 & -x_c \\ 0 & 1 & -y_c \\ 0 & 0 & 1 \end{bmatrix}}_{C}\begin{bmatrix} x \\ y \\ 1 \end{bmatrix}=0. \tag{3.3}$$

For either case, the expression has five degrees of freedom in unknown parameters. Wafer detection is to first find clusters of points that belong to the same ellipse. then to estimate five parameters of the ellipse for each cluster. In practice, outliers are inevitable so there should be an additional cluster for points that do not belong to any ellipse. Arcs inherit from the edge image produced by the camera and edge detector. An arc is a curve of connected points. Arcs form the observation of wafers. Therefore, arcs naturally become the basic clusters of points. The problem then becomes clustering arcs and fitting ellipses from the

cluster of arcs instead of for individual points. Valid arcs are split into four classes called quadrants in [37, 40] according to two properties, namely gradients and difference between areas over and under the arc. Four quadrants numbered by I, II, III and IV. Existing ellipse detectors require selecting valid combinations of three arcs from different classes to form all possible ellipses as shown in Fig. 3.5. This is an exhaustive search of complexity $O(N^3)$ where N is the number of arcs. Invalid combinations are excluded by the checking geometry constraints described in Section 3.3 of [37]. The arc selection for ellipse detection, however,

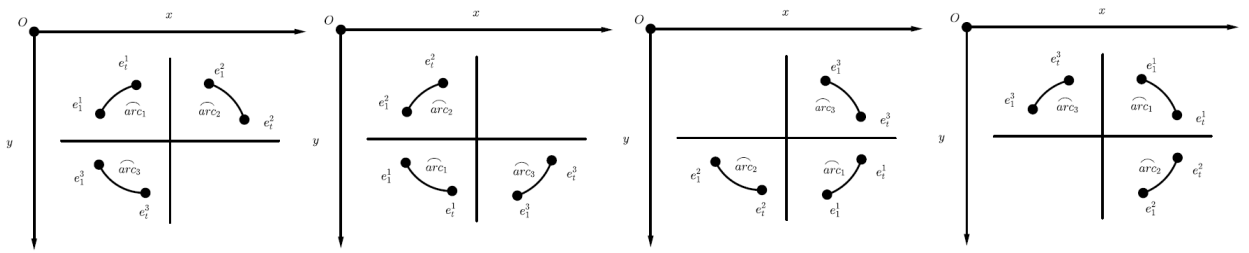


Figure 3.5: Valid combinations of arcs .

is not suitable for wafer detection as the three-quadrant combination is not visible for most cases. The arc selection algorithm is modified in the following Section 3.2 for wafer detection.

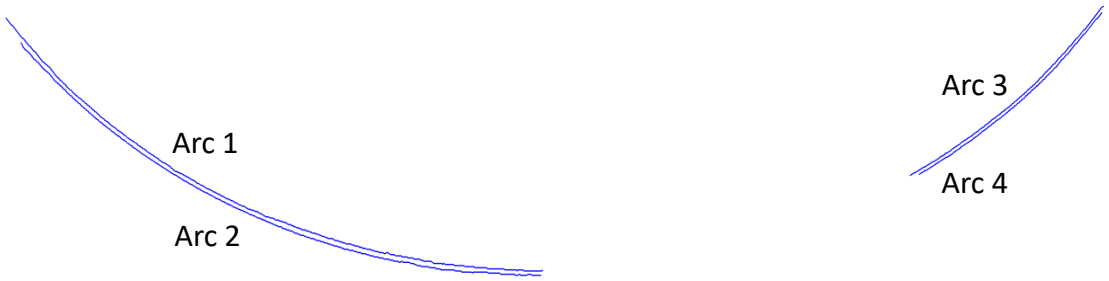
Validation is necessary because the ellipse fitted after arc selection only uses three arcs. There may be duplicate detections from the same ellipse. Meanwhile, arcs may not lie on the fitted ellipse due to false detection. Validation provides a quality measure for each fitted ellipse, as well as merging detections that belong to the same ellipse. The most common quality measure is the residual returned by the ellipse fitting algorithm or the number of points off the ellipse. In [37, 40], the measure is the percentage of points larger than 0.1 off the implicit ellipse expression (3.1). A constant is set so that fitted ellipses whose measure larger than the threshold is rejected. For merging fitted ellipses, another threshold is set. Ellipses are merged if their mutual distance in parameter space is smaller than the threshold. This threshold setting for validation, however, does not work well for the wafer detection case because one wafer has two elliptical edges close to each other. These two edges will cause false positives. The problem is also discussed and addressed in the following Section 3.2.

Robust Elliptic Segments Detection Modifications

We propose to use two quadrant combinations for arc selection. The number of combinations is much smaller than the three-quadrant case. In this work, instead of the fast geometry constraint check for valid combinations, we directly use the residual of ellipse fitting result to reject invalid arc combinations. Combination of arcs are rejected if either condition is true:

- Arcs overlap if they are projected onto the ellipse boundary.
- Residual of fitting result is larger than a threshold.
- Arcs are of the same quadrant.

The residual $r(i, j)$ of two arcs is an important value and will also be used in the validation stage. Note the residual cannot be directly used to check whether the fitted ellipse is valid. An example is shown in Fig. 3.6. Two elliptic segments, corresponding to the upper edge and lower edge of the same wafer, are broken into four arcs. Four valid combinations are found and their residuals are all small. However, only two combinations are true. Residuals of any combination is about 0.3 pixel. Setting a threshold or just finding 2 combinations with minimum residuals will not produce the correct solution. A bipartite matching solution



Residual (pixel)	Arc 3	Arc 4
Arc 1	0.304	0.336
Arc 2	0.361	0.348

Figure 3.6: Four arcs and the residuals are their potential combinations. (Arc1, Arc3) and (Arc2, Arc4) should be the correct combination, but the residual of (Arc1, Arc4) is smaller than (Arc2, Arc4). Using residual as the acceptance threshold is not suitable.

is proposed in this section to find optimal pairs of arcs to merge. The bipartite matching is applied to both arc selection stage and ellipse merging stage. Several assumptions are made to formulate the matching model:

- Each arc can at most belong to one elliptic segment.
- Arcs belong to the same elliptic segment must not overlap.

The bipartite matching model for find valid fitted ellipses is formulated as follows. First, we construct a graph $(V; E)$ denoting the potential grouping of arcs, where vertices V is the set of all arcs and $e(i, j) \in E$ means arcs v_i, v_j form a valid combination. The bipartite graph is constructed as $(S := V, T := V; E)$. A cost $r(i, j)$ is defined for each edge, denoting the

fitting error. $r(i,j)$ is the mean distance error to the fitted ellipse using points from v_i, v_j. Finally, Hungarian algorithm is performed on each bipartite graph to derive the optimal combination of arcs. The decomposition into connected components reduces the complexity because solving the bipartite matching is $O(N^3)$. The algorithm returns a mapping $\theta : V \rightarrow V \cup \{v_0\} \theta(v_i) = v_j \in V$ that assigns each arc to another arc or a dummy arc v_0. The set of valid elliptic segments is

$$S_0 = \{v_i \cup v_j | \theta(i) = j, j \neq 0\}. \tag{3.4}$$

The idea behind the bipartite matching solution is that one true positive ellipse may not have the minimum residual compared to another false positive ellipse which share the same arc, but the overall residual of all true positive ellipses should be the minimum among all possible combinations. Pseudo-code of the algorithm is written in Algorithm 1.

Algorithm 1 Arc merging algorithm

```
V // set of all arcs
S ← V
for all v_i ∈ S do
    for all v_j ∈ V do
        r(i,j) ← FitEllipse(v_i, v_j)
        e(i,j) ← r(i,j) < thresold?
    end for
end for
for all {S_k, V_k; E_k, R_k}←ConnectComp({S, V; E, R}) do
    θ ← BipartiteMatch({S_k, V_k; E_k, R_k})
    //update S and V
    for all s ∈ S do
        s ← s ∪ θ(s)
        V ← V − {θ(s)}
    end for
end for
S_0 ← S // merged arcs
V_0 ← V // remaining arcs
return S_0
```

Similarly, the merging of elliptic segments can also be done with bipartite matching by constructing $(S{:=}S_k, T{:=}S_k; E_k)$ where S_k is the set of valid elliptic segments at kth iteration. Cost is the residual of ellipse fitting using two elliptic segments. The algorithm is shown below in Algorithm 2.

Algorithm 2 Ellipse merging algorithm

```
S_0 // valid elliptic segments
V_0 // remaining arcs
iter ← 0
while S_iter not converge do
    S_{iter+1}, V_{iter+1} ← φ
    for all v_i ∈ S_iter do
        for all v_j ∈ V_iter do
            r(i,j) ← FitEllipse(v_i, v_j)
            e(i,j) ← r(i,j) < threshold OR not overlap(v_i, v_j)
        end for
    end for
    θ ← BipartiteMatch({S_iter, V_iter; E, R})
    for all v_i ∈ S_iter do
        if θ(v_i) ≠ v_0 then
            v_i ← v_i ∪ v_j
            S_{iter+1} ← S_{iter+1} ∪ {v_i}
            V_{iter+1} ← V_{iter+1} ∪ {v_i}
        else
            V_{iter+1} ← V_{iter+1} ∪ {v_i}
        end if
    end for
    iter ← iter + 1
end while
```

3.3 Matching-based Detection of Known Objects

Shape matching algorithms are taken in this section for the detection of objects with known shapes. Voting-based matching is applied to objects in the image space where edges and basic geometric segments are extracted. On the other hand, point registration is applied to objects observed as point clouds, where the observed points are registered to the known mesh.

The voting-based method is inspired by the generalized Hough transform [59]. The shape matching problem of pixels can be viewed as an unconstrained optimization. Suppose the features to be matched are $\{x_1, x_2, \cdots, x_n\} \subset X$, the shape is parameterized as $\theta \in \mathbb{R}^m$ where m is the dimension of the parameter space (or Hough Space). The distance d between the shape represented by θ and a feature x is defined as:

$$d : X \times \mathbb{R}^m \to \mathbb{R}^+, d(x, \theta). \tag{3.5}$$

And the shape is a set of features defined as $S(\theta) = \{x \in X | d(x, \theta) = 0\}$. The distance function has the important property that $d(x, \theta) = 0$ has infinite solutions θ with a given

x. This means the shape matching with one feature is very under-constrained. Many features need to cooperate together to provide enough constraints for a shape, which is better expressed as an optimization problem. The optimization problem of shape matching is

$$\min_{\theta} J(\theta) = \sum_{i=1}^{n} \left(1 - e^{-\alpha \cdot d^2(x_i, \theta)}\right), \tag{3.6}$$

where $\alpha > 0$ is denoted as the "inverse effective distance" of cost function. When $\alpha \to 0$, the optimization becomes a least square optimization which minimizes the distance of all points to the shape. However, for a shape matching problem with much noise, the optimal cost is still high because the problem is over-constrained with all the features. A shape cannot pass all the noise features, resulting in the optimal shape not passing meaningful features because it needs to compromise to the noise features.

When $\alpha \to \infty$, the cost function becomes the cost function for Hough Transform based detection, where all points only have local effect on the shape, which is the case of Hough Transform. The problem is that there are many local optimal solutions because any combination of noise features, which occasionally form a shape, contributes to a local optimal. In this case, the global optimal solution is solved by discretization and search. The tensor of cost function in the parameter space becomes the sum of tensors $[\delta_{x_i}(\theta_j)]$ transformed from each feature x_i

$$\begin{aligned} [J(\theta_j)] &= \sum_{i=1}^{n} [\delta_{x_i}(\theta_j)], \\ \delta_{x_i}(\theta_j) &= \begin{cases} 1, d(x_i, \theta_j) > \varepsilon \\ 0, d(x_i, \theta_j) \leq \varepsilon \end{cases} \end{aligned} \tag{3.7}$$

The computational complexity of HT depends on the dimension of the parameter space and the sparsity of the tensor $[\delta_{x_i}(\theta_j)]$. If the dimension is low, which is the common case for line and circle detection [60, 61], the sparsity can be ignored. Single pixel is usually picked as a feature and the algorithm is very efficient. When the dimension is high, like the case of ellipse and other complex or deformable shapes, the problem is still solvable if the tensor is very sparse. This means each feature should provide more constraint so the dimension of the parameter manifold that passes the feature is small. Pairs and arc segments [39, 37, 62], are usually used as features instead of individual pixels.

Detection of Chamber Openings in RGB image

The shape of the chamber opening is a cross-shape polygon simplified from its CAD model. The opening is captured by the camera on the robot from different angles and the shape in the image will be different. Fortunately, with the knowledge of the position and orientation of the surface it lies on, a planar perspective transformation (3.8) can be used to transform

points in the image to points in the reference plane

$$w \begin{bmatrix} u_0 \\ v_0 \\ d \\ 1 \end{bmatrix} = H^{-1} \begin{bmatrix} u \\ v \\ 1 \\ 0 \end{bmatrix}, \tag{3.8}$$

where H is the Homography matrix of the surface, (u, v) is the pixel in the actual camera and (u_0, v_0) is the pixel viewed in the reference plane. The intuition of the polygon shape detection using line features is demonstrated in Fig. 3.7. For a line segment, the polygon can slide along the line segment, resulting in the red dots indicating potential centers. Suppose

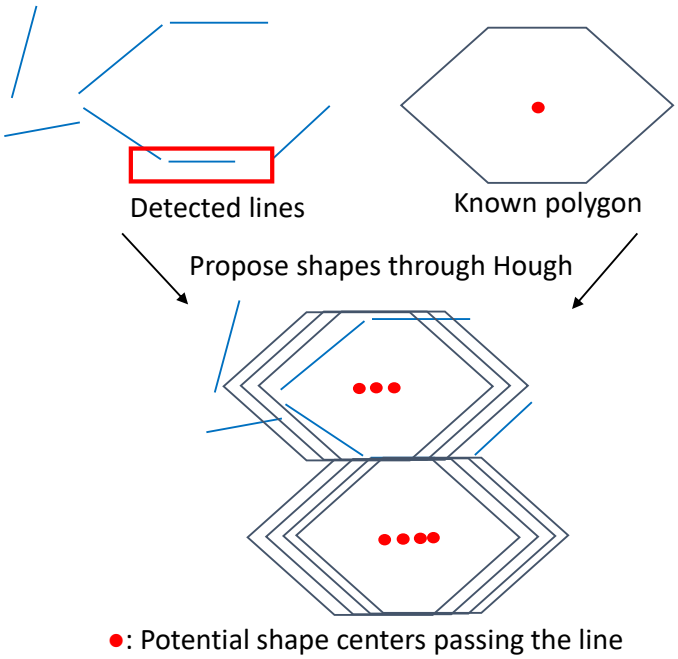


Figure 3.7: Intuition of Polygon detection.

there is no deformation, the shape can be parameterized by its center position and rotation so the dimension of Hough space is 3. However, the rotation is determined by the line feature so it can be treated as if there were only 2 dimensions. Suppose the end points of a line segment are (x_1, y_1), (x_2, y_2) and the corresponding edge length of the polygon is l. The potential centers form a line segment where end points (x_c^1, y_c^1), (x_c^2, y_c^2) are reached when the vertex reaches the end point of the blue line segment shown in Fig. 3.8.

$$\begin{aligned} \left(x_c^1, y_c^1\right) &= (x_2, y_2) - \frac{(1+\varepsilon)\, l}{l_1} \left[(x_2, y_2) - (x_1, y_1)\right] + (x_{ce}, y_{ce}), \\ \left(x_c^2, y_c^2\right) &= (x_1, y_1) + \frac{\varepsilon l}{l_1} \left[(x_2, y_2) - (x_1, y_1)\right] + (x_{ce}, y_{ce}). \end{aligned} \tag{3.9}$$

In practice, a mismatch of ϵl is allowed. If $\epsilon = 0$, then line segments even slightly longer than

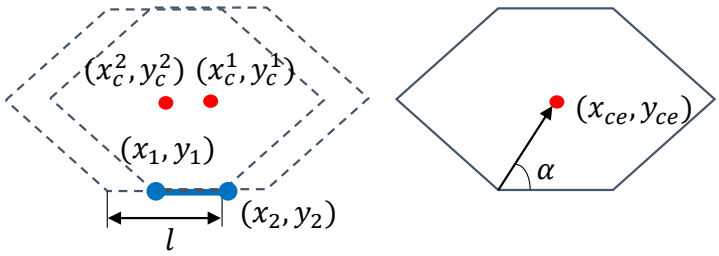


Figure 3.8: Demonstration of potential center calculation.

the edge will be rejected. In this work, the ϵ is chosen to be $\varepsilon = 0.1\sqrt{x_{ce}^2 + y_{ce}^2}\cos\alpha$. Votes of line segments are added to the discretized Hough space. The shape with the most votes is returned as the detected opening. The accurate position is then calculated by fitting the shape using iterative reweighted linear least square (IRWLS) [63] to exclude the influence of the remaining outliers. Supposed the line features fitted are (a_j, b_j, c_j)'s of $a_j x + b_j y + c = 0$ and the location has position parameter (x, y, θ, ϵ) where (x, y) is the center position, θ is the rotation of the shape w.r.t. nominal rotation and ϵ is the uniform expansion of the shape (shape size is $1+\epsilon$ of the nominal size). The regression problem is written as

$$\min_{x,y,\theta,\varepsilon} \sum_j \left[\left(a_j x + b_j y + c_j - d_j^0 - d_j^0 \varepsilon\right)^2 + \left(\theta - \arctan\frac{a_j}{b_j} + \arctan\frac{a_j^0}{b_j^0}\right)\right]. \tag{3.10}$$

Detection of Objects in Point Clouds

Some target objects are represented by a set of points or mesh that is not regularly gridded. In addition, some sensors, like LiDAR or RGBD camera, also return the observation as unstructured point clouds. Point registration is applied in this thesis to handle the shape matching problem of point cloud data. Point registration deal with the problem of finding the correspondence and coordinate transform between two sets of points, usually 2D or 3D point cloud in the Euclidean space. The two sets of points contain some common or similar parts but the correspondence of points in the two sets are unknown. Point registration finds the transformation of point coordinates from one set to the other so that the common parts are aligned. For shape matching problem, the purpose is to find the subset of observed points corresponding to the target object as detection, as well as the geometric parameters that locate the detected points.

The notations of the point registration problem are defined here. $X = \{x_1, x_2, \cdots, x_N\} \subset \mathbb{R}^D$ and $Y = \{y_1, y_2, \cdots, y_M\} \subset \mathbb{R}^D$ are two point sets with cardinality of M and N. Each point is of dimension D. The correspondence of two points between two sets is $a_{ij} = \{x_i, y_j, p_{ij}\}$. For binary correspondence, $p_{ij} \in \{0, 1\}$. $p_{ij} = 1$ means $x_i \in X$ is

exactly associated with point $y_j \in Y$. Another kind is called soft assignment or fuzzy correspondence [64] where $p_{ij} \in [0, 1]$. In this case, the correspondence can be viewed as a probability distribution.

Besides correspondence, point registration also finds the transformation to align two sets of points. Without loss of generalization, suppose the transformation is applied to Y in order to fit the point coordinates in X. The transformation is often denoted as a continuous function $f : \mathbb{R}^D \to \mathbb{R}^D$ parameterized by correspondence

$$x = f(y, X, Y, a), \tag{3.11}$$

where $x, y \in \mathbb{R}^d, X \subset \mathbb{R}^d, Y \subset \mathbb{R}^d$ and x, y are not necessary points in X, Y. An example is shown in Fig. 3.9. We choose to use the coherent point drift (CPD) [65] as the baseline

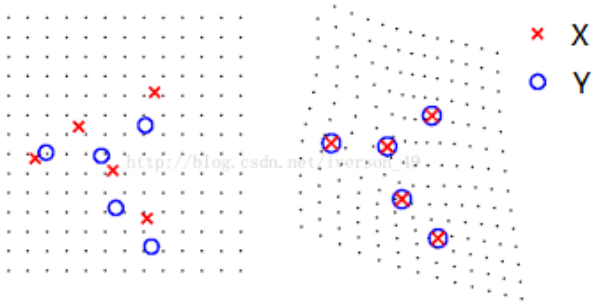


Figure 3.9: Set of points and the continuous transformation of the space.

algorithm for our point registration problem, which uses the expectation maximization (EM) algorithm to solve the posterior distribution of a Gaussian Mixture Model (GMM). CPD is computationally efficient on small point sets and relatively robust against deformation, noise and small rigid transformation. The main procedure of CPD is demonstrated in 3.10. The E-step, which corresponds to the expectation function update step of the EM algorithm, estimates the fuzzy correspondence p_{ij} of two sets of points. Given updated correspondences, the M-step optimizes a least square obtained from maximum likelihood estimation of GMM, solving the transformation function. One E-step and M-step form one iteration and it is often not sufficient to finish alignment. Then a loop of iterations is formed by recursively updating the coordinates of Y with solved transformation until convergence. Suppose the correspondence between $x_i \in X$ and $y_j \in Y$ is represented by the posterior probability p_{ij} of a categorical latent random variable

$$p_{ij} = \frac{p(y_j)p(x_i|y_j)}{p(x_i)}. \tag{3.12}$$

The Gaussian Mixture Model suppose that the prior distribution $p(x_i)$ of x_i is a GMM composed of Gaussians $p(x|y_j)$ centered at y_j with variance σ^2 and $p(y_j)$ is initialized as

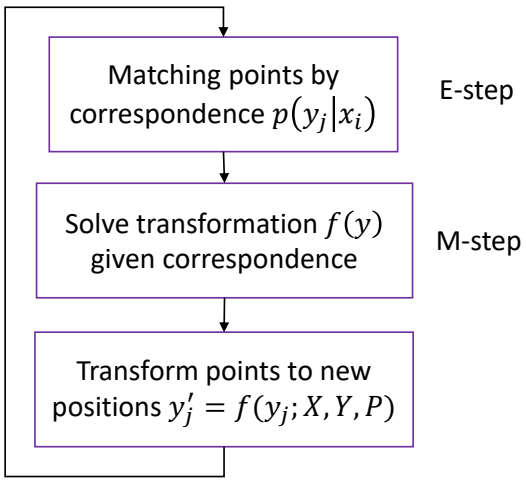


Figure 3.10: An illustration of the process of EM used in point registration.

constant $\frac{1}{m}$

$$\begin{aligned} p(x = x_i) &= \sum_{j=1}^{M} p(y_j) p(x = x_i | y_j), \\ p(x|y_j) &= \frac{1}{2\sigma} e^{-\frac{\|x - y_i\|^2}{2\sigma^2}}. \end{aligned} \tag{3.13}$$

In the M-step, the EM-algorithm transforms maximizing the likelihood function into maximizing weighted sum of individual log likelihoods.

$$Q = -\sum_{i=1}^{N} \sum_{j=1}^{M} p(y_j|x_i) \log(p(x_i|y_j) p(y_j)). \tag{3.14}$$

By fixing $p(y_j|x_i), p(y_j)$, and $p(x_i|y_j)$ being Gaussian distribution, maximizing Q in the M-step with respect to the transformation function becomes maximizing least squares.

$$\begin{aligned} \max_{f(y),\sigma} Q &= \max_{f(y)} \sum_{i=1}^{N} \sum_{j=1}^{M} p(y_j|x_i) \log(p(x_i|f(y_j))) \\ &= \max_{f(y),\sigma} \sum_{i=1}^{N} \sum_{j=1}^{M} \left(\frac{p(y_j|x_i) \|x_i - f(y_j)\|^2}{2\sigma^2} \right) + \log \sigma \cdot \sum_{i=1}^{N} \sum_{j=1}^{M} p(y_j|x_i). \end{aligned} \tag{3.15}$$

The optimization is still ill-posed because there is no cost on moving points or deforming the object shape. The solved transformation can fit points by destroying the original topology. Therefore, parameterization and regularization are needed. CPD introduces the norm in

Reproducing Kernel Hilbert Space (RKHS) as the regularization function. The regularization is defined as

$$\|f\|_{\mathcal{H}}^2 = \int_{\mathbb{R}^D} \frac{\left\|\hat{f}(s)\right\|^2}{\hat{G}(s)} ds = \int_{\mathbb{R}^D} \sum_{d=1}^{D} \frac{\left|\hat{f}_d(s)\right|^2}{\hat{G}(s)} ds, \tag{3.16}$$

where the RKHS $\mathcal{H}$ is a Hilbert space of continuous transformation functions f equipped with D dimensional Fourier transform $\mathcal{F}$ and $\mathcal{F}(f) = \hat{f} : \mathbb{C}^D \to \mathbb{C}^D$. G is the D dimensional Gaussian. It is proved that the solution of the M-step with regularization term becomes

$$\begin{aligned} f_d^*(y) &= \sum_{j=1}^{M} \omega_{jd} G(y - y_j), \\ \|f\|_{\mathcal{H}}^2 &= \sum_d \sum_k \sum_j \omega_{jd} G(y_j - y_k) \omega_{kd} = trace(\mathbf{W}^T \mathbf{G} \mathbf{W}), \end{aligned} \tag{3.17}$$

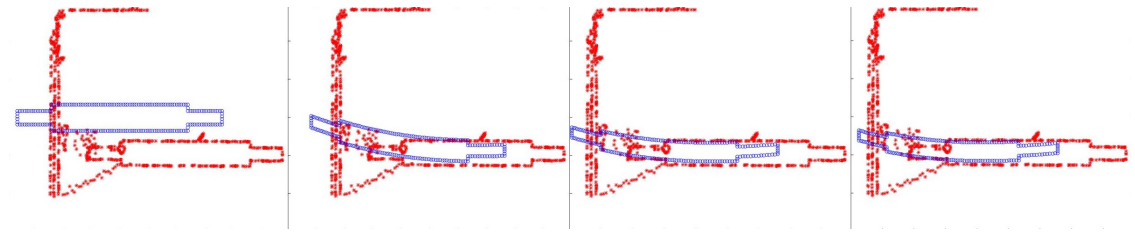
where the norm is in quadratic form and the optimization problem becomes solving a linear least square on ω_{jd} and σ. Denote $\mathbf{Y}, \mathbf{X}, \mathbf{P}$ be the stacked matrices of y_{jd}, x_{id}, p_{ij}, and $\mathbf{T} := \mathbf{Y} + \mathbf{G}\mathbf{W}$ be the updated matrix of points $f(y)$'s of Y, then

$$\begin{aligned} \mathbf{W} &= \left(\mathbf{G} + \lambda\sigma^2 diag(\mathbf{P1})^{-1}\right)^{-1} \left(diag(\mathbf{P1})^{-1}\mathbf{P}\mathbf{X} - \mathbf{Y}\right), \\ \sigma^2 &= \frac{1}{\mathbf{1}^T\mathbf{P1} \cdot D} \left[\mathrm{tr}\left(\mathbf{X}^T diag\left(\mathbf{P1}\right)\mathbf{X}\right) - 2\mathrm{tr}\left((\mathbf{P}\mathbf{X})^T\mathbf{T}\right) + \mathrm{tr}\left(\mathbf{T}^T diag\left(\mathbf{P1}\right)\mathbf{T}\right)\right]. \end{aligned} \tag{3.18}$$

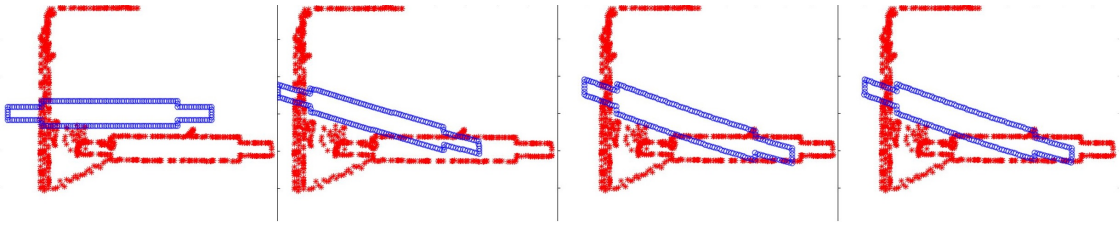
The CPD method is not robust against rotation and non-uniform outliers. It is the intrinsic disadvantage of CPD. If we add the update of transformation in M-step to E-step, it forms a highly non-convex or even concave optimization problem for the correspondence finding problem in E-step. Therefore, it requires a global correspondence finding method to at least help for initialization, which are what its variants [66, 67] pursues. The CPD and one of its variants called structure preserving point registration (SPR) [68] is implemented first on the 2D image in our experimental scenario. However, they do not perform well when registering a simple chevron shape to a filtered edge-detection image. Results of two algorithms applied on experimental data are shown in Fig. 3.11. The methods implemented both converge to outliers instead of the desired shape, even with fine-tuned parameters. Changing the initialized pose of the reference shape Y results in different converged shapes. This means the CPD and its variant both find the local minimum and are sensitive to the initial pose.

3.4 Efficient Global Voting for Non-rigid Shape Matching

Initialization of the correspondence matrix is very common in point registration problems especially for iterative methods utilizing the whole point set. Initialization is often done



(a) CPD



(b) SPR

Figure 3.11: Bad point registration results with CPD and its variant.

by finding representative features of the shape. Local shape features are the most efficient on [48]. Local shape features significantly down-sample the points by choosing unique features and reduces the potential correspondences by clustering features in the feature space. However, the cluttered scenes in industrial applications have few unique features to represent the shape. For example, corners of objects, which are extensively detected by most feature descriptors, are usually chamfered or rounded. Most surfaces are smooth instead of having sharp edges. Features are hard to select and separate in this case so an efficient clustering method on large amounts of correspondences is required.

This section incorporates a global voting framework to the coherent point drift (CPD) algorithm by an efficient geometric clustering algorithm. It will iteratively search the space of rigid transformation globally based on the non-rigid transformation solution of the current step. The whole method is called the global voting assisted CPD. It keeps track of several locally best matched shapes with minimum deformations, so that the point registration can handle large misalignment in cluttered scenes.

Rigid Transformation-based Global Voting

We assume that feature correspondences must be filtered by some regulations like rigid transformation. For example, if there are K final corresponding pairs of features (x_{i_1}, y_{j_1}), $\cdots$, (x_{i_K}, y_{j_K}), they must preserve the Euclidean distance by minimizing the change of

mutual distances

$$\min \sum_{p=1}^{K} \sum_{q=1}^{K} \left(\left\| x_{i_p} - x_{i_q} \right\|_2 - \left\| y_{j_p} - y_{j_q} \right\|_2 \right)^2. \tag{3.19}$$

Note it does not mean that non-rigid transformation cannot be solved from the correspondences, because certain soft margins of deviation from isometry are allowed. Evaluating the deviation is $O(K^2)$. And it requires a lot of iterations to find the optimal correspondence. In order to reduce the complexity, it is assumed that the local feature descriptors used here return a relatively correct local reference frame (LRF) of the surface in the neighborhood. Then the rigid transformation can be determined from one pair of points (x_i, y_j) and an efficient global algorithm can be developed. In fact, many state-of-the-art robust local descriptors returns LRF, such as 3D-SURF [69], SHOT [70], RoPS [71] and TriSI [72]. Suppose the LRF calculated by the descriptor is denoted as $[R, T]$ where $R \in \mathbb{R}^{3\times3}$ is the rotation matrix and $T \in \mathbb{R}^3$ is the translation vector, the rigid transformation solved from $[R_i, T_i]$ and $[R_j, T_j]$ is

$$R_{ij} = R_i R_j^T,\ T_{ij} = T_i - R_{ij} T_j. \tag{3.20}$$

Note rigid transformation can be parameterized by 6 variables $[\theta, T]$ with 3 Euler angles θ and 3 translations T. The $[R_{ij}, T_{ij}]$'s generated by corresponding pairs in two point sets result in $M_1 \times N_1$ points in a 6-dimensional space, where M_1 points are selected from X of M points and N_1 are selected from Y of N points. The point in the $SE(3)$ space which has most $[R_{ij}, T_{ij}]$ in the neighborhood is the global optimal transformation matching two shapes.

If the rigid transformation space is discretized, it becomes a voting problem like the Hough Transform. It can also be viewed as a clustering problem without discretizing the space. Another formulation is applied here using the graph representation, which is finding maximal cliques of an intersection graph of boxes [73]. The graph G is defined in (3.21)

$$\begin{aligned} &\text{Nodes: } V = \{v_{ij} = [\theta_{ij}, T_{ij}], i = 1, \cdots, M_1, j = 1, \cdots, N_1\}, \\ &\text{Edges: } E = \{(v_{ij}, v_{kl}) \,|I(v_{ij}) \cap I(v_{kl}) \neq \emptyset\}, \end{aligned} \tag{3.21}$$

where $I : V \to \mathbb{R}^6$ is a function of the intersection graph, mapping nodes to boxes in the 6-dimensional space. The rotation angle ranges in $[-\pi, \pi]$ and requires some special treatment near the boundary $-\pi, \pi$ but it does not affect the overall complexity. The sizes of intervals are set manually for each dimension but are uniform among all nodes.

$$\begin{aligned} I(v_{ij}) = &[\theta_{ij}^1 - \sigma_\theta^1, \theta_{ij}^1 + \sigma_\theta^1] \times [\theta_{ij}^2 - \sigma_\theta^2, \theta_{ij}^2 + \sigma_\theta^2] \\ &\times [\theta_{ij}^3 - \sigma_\theta^3, \theta_{ij}^3 + \sigma_\theta^3] \times [T_{ij}^1 - \sigma_T^1, T_{ij}^1 + \sigma_T^1] \\ &\times [T_{ij}^2 - \sigma_T^2, T_{ij}^2 + \sigma_T^2] \times [T_{ij}^3 - \sigma_T^3, T_{ij}^3 + \sigma_T^3] \end{aligned}. \tag{3.22}$$

In the rigid transformation graph, a maximal clique is very similar to a cluster, which is locally the largest subset of $[\theta_{ij}, T_{ij}]$'s where all rigid transformations are close to each other. Therefore, finding a maximum clique is equal to finding the rigid transformation that has most votes around it.

Efficient Tree Structure for Vote Counting

Finding the maximum clique in dimension 1 and 2, namely interval graph and rectangular graph, are both of $O(n \log n)$ complexity where $n = |V|$ is the number of nodes. For general intersection graph of boxes with dimension D, there are at most n^D maximal cliques and the complexity of finding a maximum clique is $O(n^{D-1} \log n)$. The rigid transformation space is of dimension 6 with $n = M_1 N_1$ so the complexity is not suitable. Approximation is needed to efficiently find clusters. Typical clustering algorithms used in rigid point registration methods [71, 74, 75] are based *kd*-tree and depth-first search. They are of $O(DM_1N_1 \log M_1N_1$ complexity but no guarantee of finding the maximum clique. RANSAC is random search and has larger expected complexity if we want to guarantee global maximum. A more sophisticated review on the complexity of solving global correspondence is [76].

An approximation algorithm of complexity $O(DM_1N_1 + DK)$ based on discretization is developed for efficient global voting, where K is the number of cells in one dimension and is at the same scale as M_1, N_1. The most important is that the complexity does not grow exponentially with respect to the number of dimensions, so the algorithm is very efficient in the 6-dimensional rigid transformation space. In terms of approximation rate, Euclidean clustering algorithms with margin σ return clusters which are complete subgraphs of diameter (or margin for intersection) 2σ. The proposed algorithm also generates complete subgraphs of the same diameter by using σ as the unit for discretization.

Efforts for robustness are made in the proposed method. we further modified the size counting of clusters, aka voting, to address density mismatches between two sets of points. Typically, each pair (x_i, y_j) in a cluster counts. This could cause problems when point sets have different distributions. For example, Fig. 3.12 shows two down-sampled point clouds in 2D by only keeping one point in each cell of a grid. The dense outliers in X generates a large distraction for typical clustering methods. The largest cluster corresponds to an undesired rigid transformation and thus wrong registration result due to the large number of votes contributed by the outliers matching with one edge of the model, even if we are using a classic global voting algorithm. The problem above is caused by counting all $(x_i, y_{j_1}), (x_i, y_{j_2}), \cdots, (x_i, y_{j_r})$ sharing the same x_i as r votes. However, to initialize a good rigid transformation for solving the non-rigid transform, the margin of the cluster must be set large enough to get enough votes. It is very common to see one-to-many correspondences. A solution to the distribution mismatch is to eliminate redundant votes that share the same point in Algorithm 3.

Two contributions of the proposed voting algorithm are summarized as follows. First, the complexity of clustering is reduced. The discretization is performed recursively on each dimension without the curse of dimensionality. When the size of discretized space is relatively small, the complexity can be reduced from $O(DM_1N_1 \log M_1N_1)$ to $O(DM_1N_1)$. It also gets rid of the additional *kd*-tree structure on the space of rigid transformations. Second, the best algorithm is more robust by modifying the size counting of clusters. The robustness of fitting the sets of points in Fig. 3.12 is shown below in Fig. 3.13. The discretized vote map is plotted on the translation domain. The dimension of rotation is reduced by choosing

Algorithm 3 Global Voting of Correspondences (Hypothesis Generation)

$\sigma^{(d)}$ // margin at dth dimension
$C_0 \leftarrow \{V\}$ // set of clusters
Discretize $\overline{V} \leftarrow \{\overline{v}_{ij} = [\frac{v_{ij}}{\sigma}] | v_{ij} \in V\}$
// Find boundary of each dimension
$K_M^{(d)} \leftarrow \max_{i,j} \{\overline{v}_{ij}^{(d)}\}$, $K_m^{(d)} \leftarrow \min_{i,j} \{\overline{v}_{ij}^{(d)}\}$

// Solve all rigid transformations
for all $x_i \in X_1$ **do**
  for all $y_j \in Y_1$ **do**
    $R_{ij} \leftarrow R_i R_j^T$
    $T_{ij} \leftarrow T_i - R_{ij} T_j$
    $v_{ij} \leftarrow [\theta_{ij}, T_{ij}]$
  end for
end for
for all $d \in \{1, 2, 3, 4, 5, 6\}$ **do**
  Initialize list of clusters $C_d \leftarrow \{\}$
  Initialize empty list $\hat{C}_d$ of size $K_M^{(d)} - K_m^{(d)}$
  // Complexity $O(M_1 N_1)$
  for all cluster $c_{d-1} \in C_{d-1}$ **do**
    Initialize Delete list $E \leftarrow \{\}$
    for all $v_{ij} \in c_{d-1}$ **do**
      Assign v_{ij} to $\hat{C}_d[\overline{v}_{ij}^{(d)}]$
      $E \leftarrow E \cup \overline{v}_{ij}^{(d)}$
    end for
    for all $k \in E$ **do**
      if $\hat{C}_d[k] \neq \emptyset$ **then**
        Count unique i index in $\hat{C}_d[k]$
        $C_d = C_d \cup \hat{C}_d[k]$
        $\hat{C}_d[k] \leftarrow \emptyset$
      end if
    end for
  end for
end for

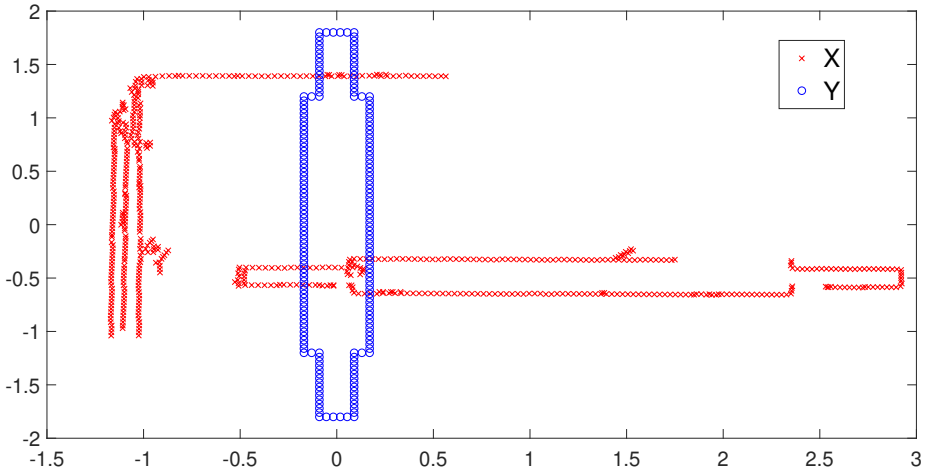


Figure 3.12: Down-sampled point sets with different densities.

the largest for each cell of (T_x, T_y). The original pairs-counting algorithm generates lots of high votes and the global maximum translation is not the ideal translation we are seeking. Meanwhile, the proposed counting algorithm captures the correct global maximum. All of the other local maximal translations have less than 50% votes against the ideal translation.

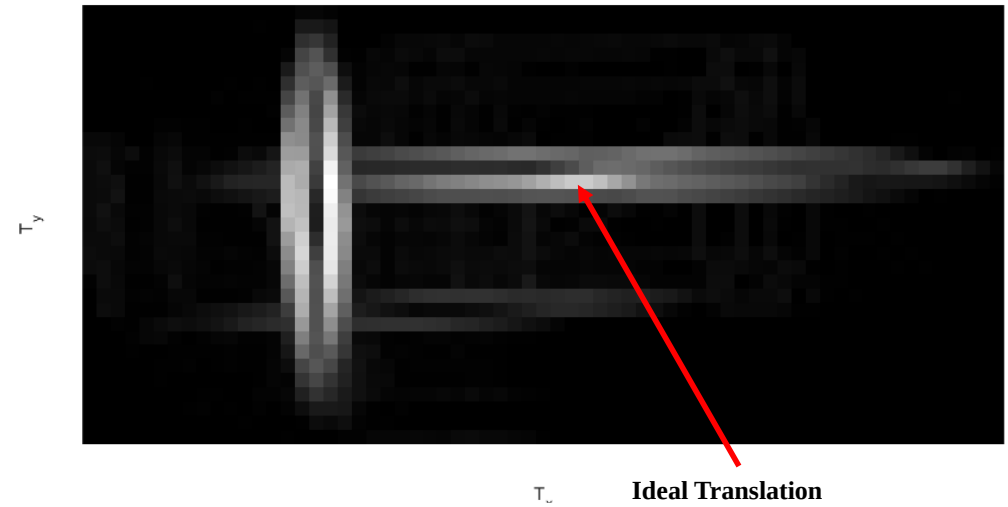
The Modified Global-CPD Framework

We propose a modification on the original "local" CPD algorithm in Fig. 3.14. Since a global search is conducted and there is no guarantee that the problem is convex, there can be multiple local maximal votes. In order to keep track of those local maximal transformations, top K of global transformations v_g^k are kept and their corresponding local deformations v_l^k are solved using CPD. Then only one with the least local deformation energy is kept, following the prior knowledge that the deformation should not be too large for the objects encountered in practical cases.

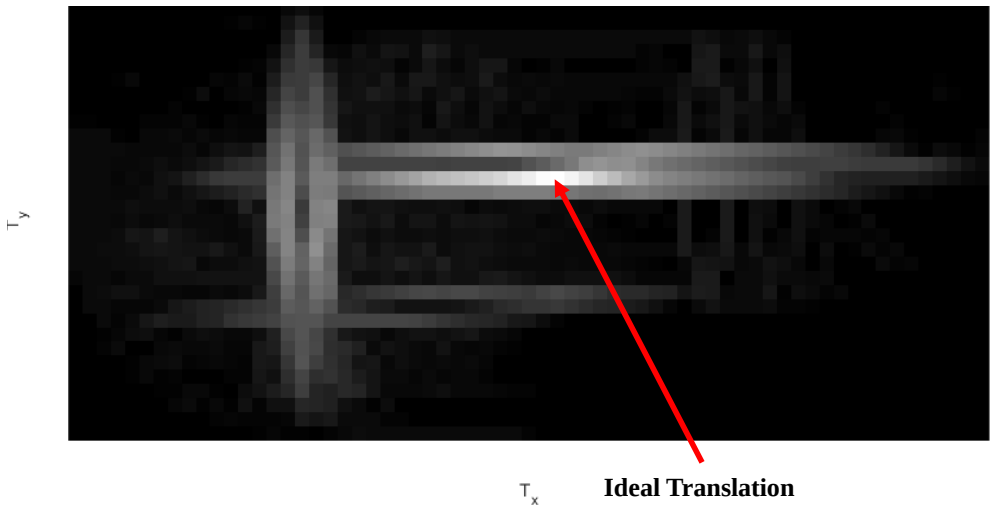
3.5 Experimental Results

Robust Wafer Detection Result

We fix the camera and wafer to verify the robustness of the wafer detection algorithm. The geometry of the system is shown by Fig. 3.15. The camera is held by a gripper for convenience of adjustment. All resulting poses are expressed as in the world frame coordinate. There are 26 slots in the carrier, numbered from 1 to 26 denoting slots from the bottom to the top.



(a) Vote map counting all pairs belonging to the same cell.



(b) Vote map using proposed counting method.

Figure 3.13: 2D vote maps using pairs-count and proposed Y-count.

Slots are parallel to each other with a machined distance of 10 ± 0.5mm. In order to give a more straight-forward result, the world frame is defined at slot 1.

Different disturbances, including FOUP load variation and light change, are added to the environment. The first test is about carrier occupancy. We place the target wafer in slot 5 and another disturbance wafer in different slots changing from 6 to 25, as shown in Fig. 3.16. The wafer detection algorithm is implemented to detect the target wafer. Due to the reflection from the disturbance wafer, the appearance of the target wafer changes in different images. Since the target wafer and the camera are static, we can aggregate the

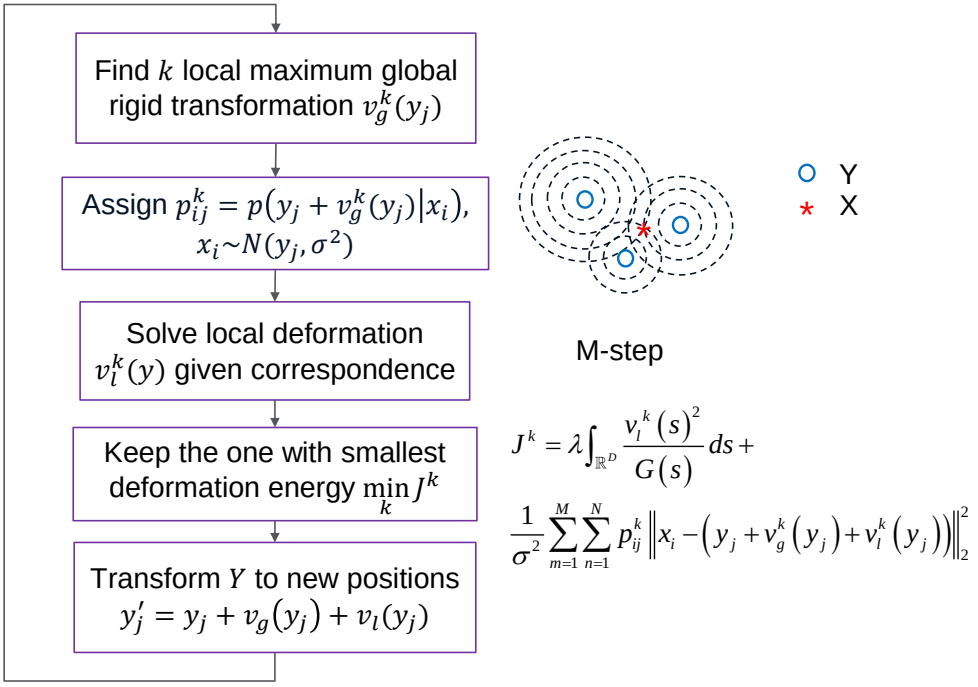


Figure 3.14: The proposed global-CPD framework.

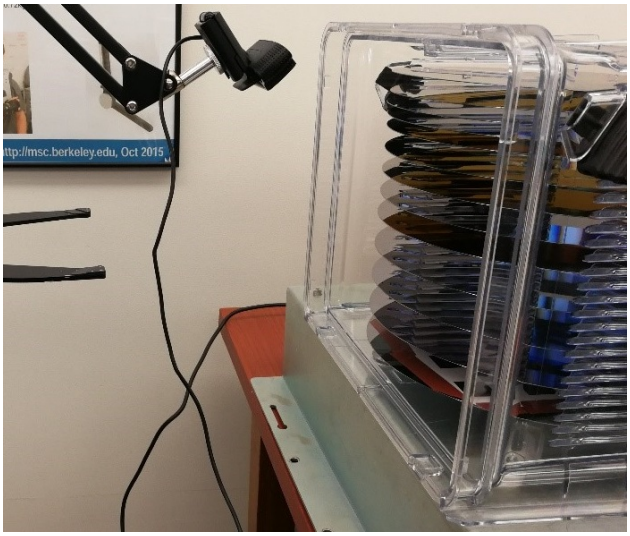
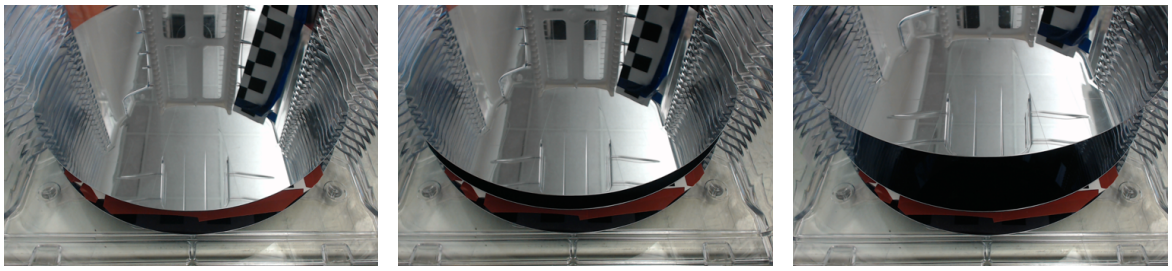


Figure 3.15: Experimental setup for wafer detection.

detected ellipses in the 2D image and verify whether they coincide with each other. It is shown in Fig. 3.17 that the different settings of the target wafer are detected within 1 pixel difference in the image.



(a) (b) (c)

Figure 3.16: photo of the first detection robustness test. (a) target wafer only; (b) add a disturbance wafer 2 slots above the target wafer; (c) add a disturbance wafer 7 slots above the target wafer.

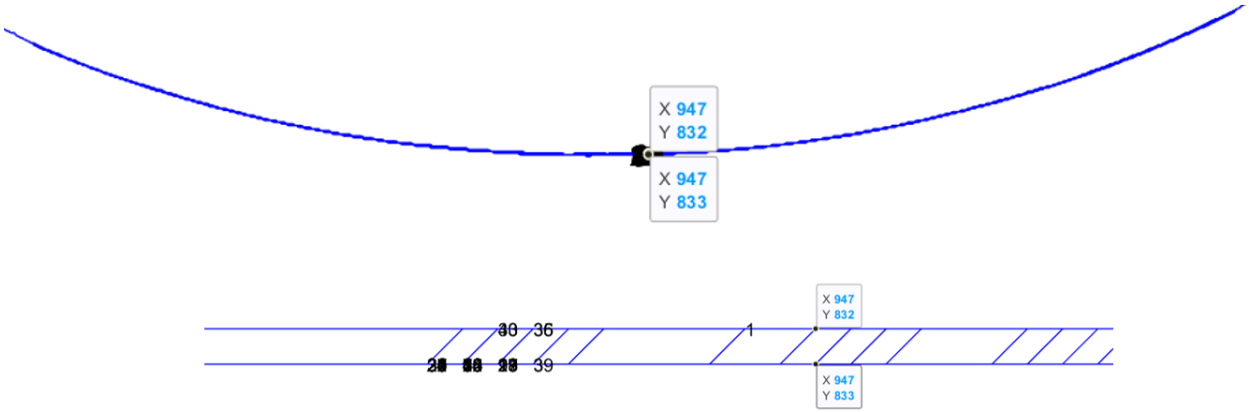


Figure 3.17: Fitted ellipses of all cases coincide with each other within 1 pixel deviation.

The second test is about light change. The edges of silicon wafers are strongly affected by the light condition of the environment. This experiment places the light source at different locations with different angles, brightness and colors, including some extreme light conditions. Some photos are listed in Fig. 3.18. sampled from a total of 105 photos. The success rate of detection is verified by checking the target wafer. Ground truth are manually labeled using the edge image as shown in Fig. 3.19. The target wafer does not produce long enough edges in some cases due to bad light angle or too low brightness. Detection accuracy is 100% for the target wafer.

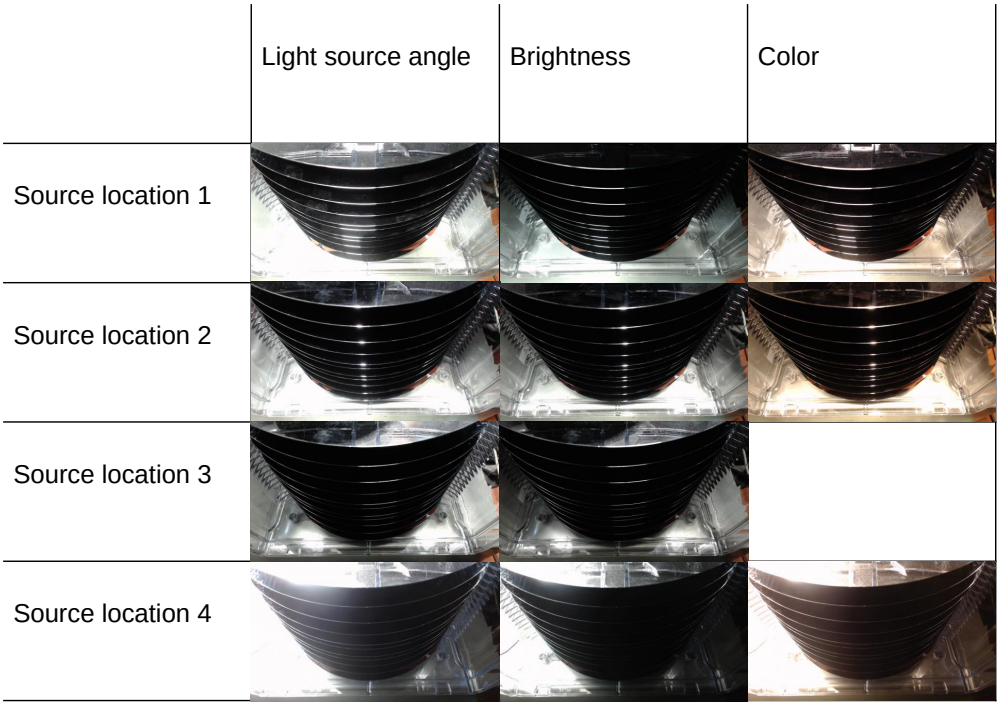


Figure 3.18: Photos of the second wafer detection robustness test.

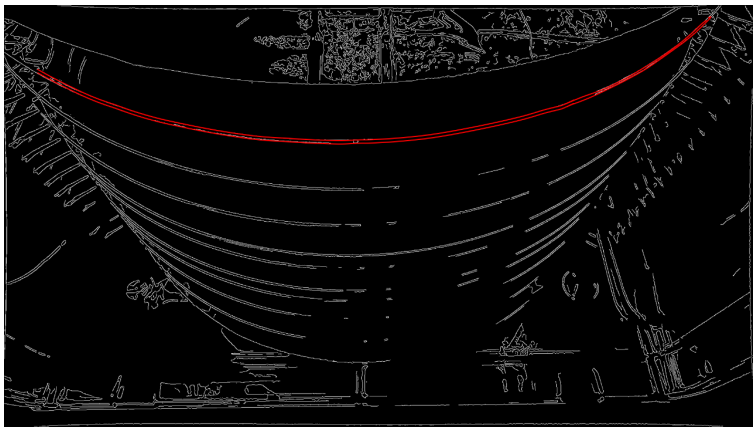


Figure 3.19: An exemplary edge image of the wafer used for robust detection verification. Red edges belong to the target wafer used to validation detection performance.

Robust Chamber Opening Detection Result

Two proposed approaches, which HT-based shape matching for image in Section 3.3 and Global-voting for point cloud in Section 3.4, are used for the detection of a chamber opening

in the semiconductor industry. We evaluate the detection algorithm on a manually created opening that mimics the actual chamber opening.

We add disturbance by drawing outliers and shapes that look like the openning to test the robustness of the detection algorithm in the image, as shown in Fig. 3.20. Blue lines are line segments detected by line HT. The end points of lines are transformed to the reference plane below. The right figure shows the Hough space of the shape. Note there is a bright pixel which means it has the most vote. After choosing the shape with the highest vote, all line segments that vote for the shape are returned as components for regression. The HT does not give a precise center position because of discretization. However, most of the noise is excluded after shape detection so shape fitting by least squares is now applicable. The red lines in the reference plane are the detected segments of the target shape and the red point is the regressed center of the shape. It shows that the algorithm is robust against various disturbances, including

- Occlusion.
- Extra noise lines.
- Similar distracting shapes.

We evaluate the global voting method on the point cloud with outliers and compare the results with CPD [65] and SPR [68]. The boundary of the target shape is discretized as a set of points and detection is performed on the point cloud generated by the RGBD camera. We show the intermediate and final non-rigid point registration results with different initial conditions in Fig. 3.21 and Fig. 3.22. Note our proposed method is not only global but only multi-modal. In this experiment we set the number of models as 3 for our method which means it will detect the top-3 likely objects.

It is shown in Fig. 3.21 that with a small drift in the initial position, the CPD and SPR are distracted by the noise points and do not converge to the correct shape. The proposed method is able to find the correct shape and fits the target object well. In Fig. 3.22, the initial condition is rotated by 90 degrees, making it far from the target object. The proposed method is still able to detect the correct object as the top-1 likely object. It also detects other objects which correspond to the local optimal returned by CPD.

3.6 Chapter Summary

This chapter introduces several robust detectors working on input format from various visual sensors, including image and point clouds, for the hand-eye system. The detectors are able to extract target objects globally, while rejecting the disturbances and outliers in the cluttered semiconductor workspace. For images, a robust elliptic segment detector is proposed for the detection of wafers and a Hough-transform-based shape matching algorithm is proposed for the detection of chamber opening. For point clouds, an efficient global voting algorithm is proposed for the initialization of non-rigid point registration. Experiments performed with

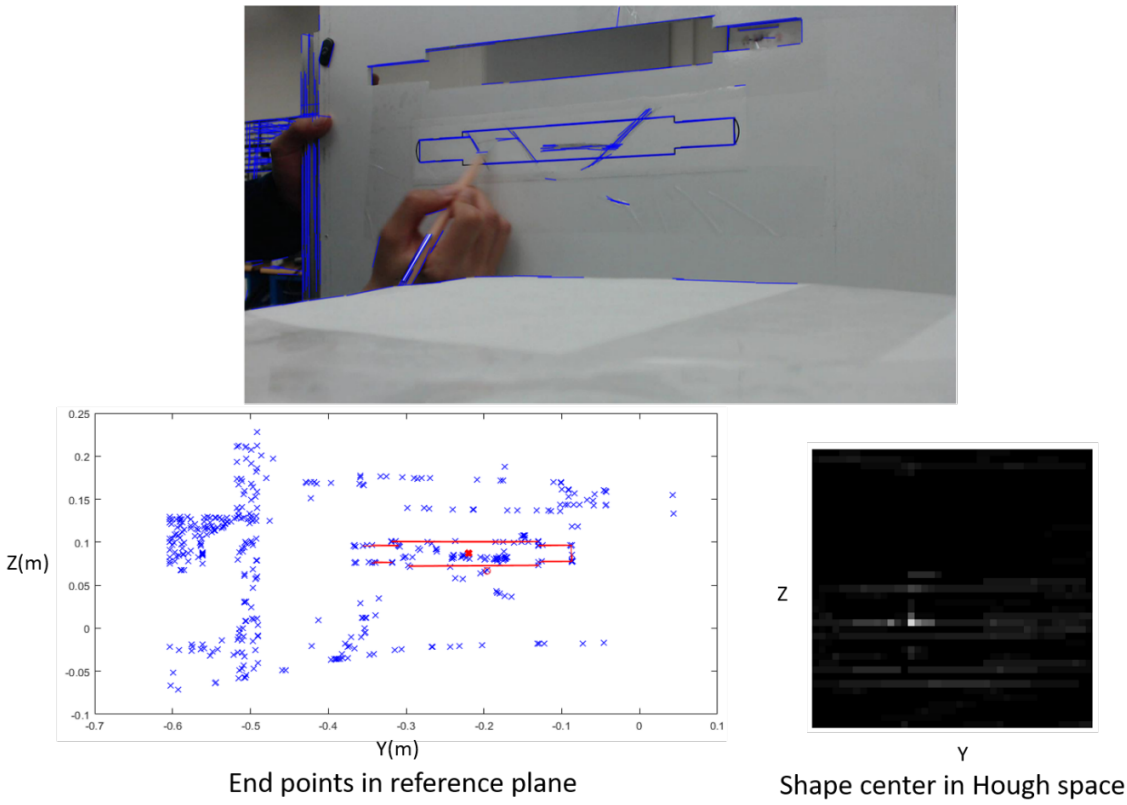
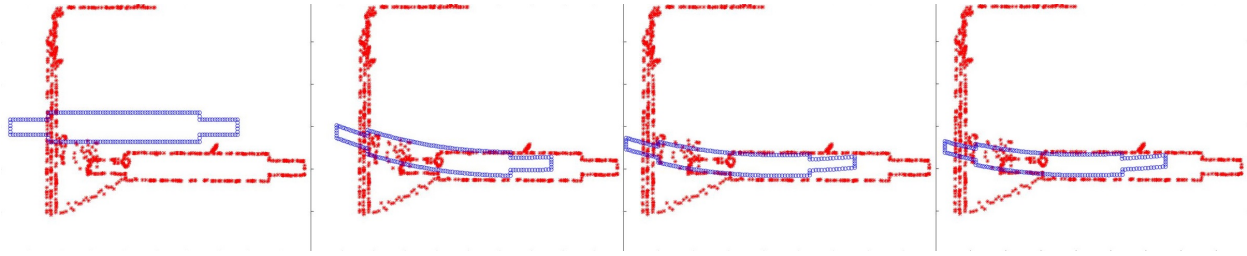
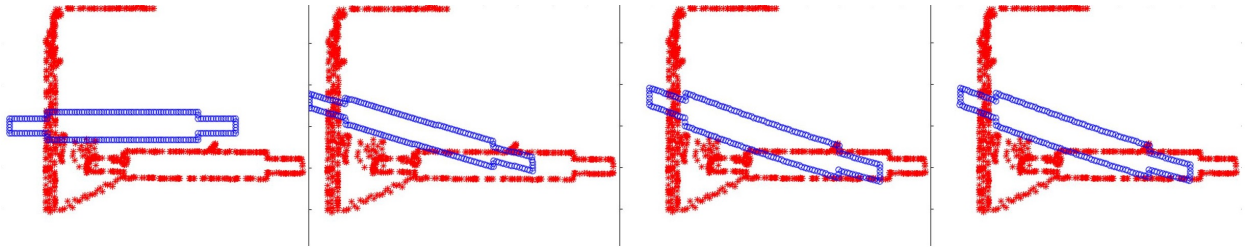


Figure 3.20: Robustness evaluation on HT-based polygon detection in the image.

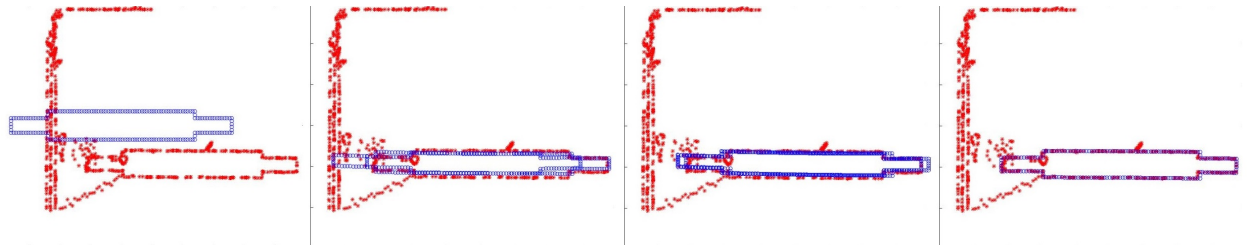
RGB and RGBD cameras show that the proposed methods accurately detect the target objects and outperform other existing detectors.



(a) CPD

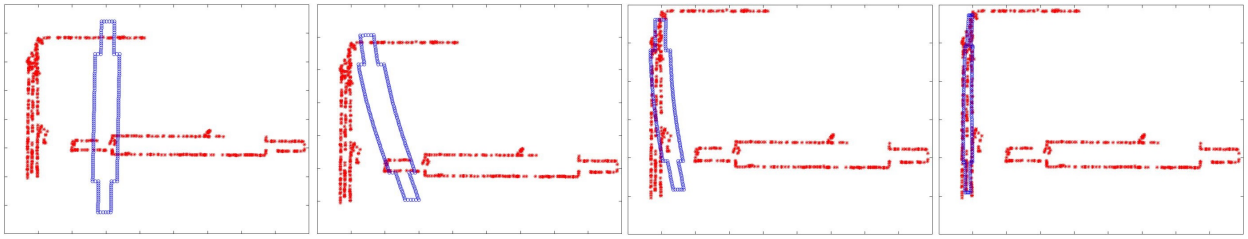


(b) SPR

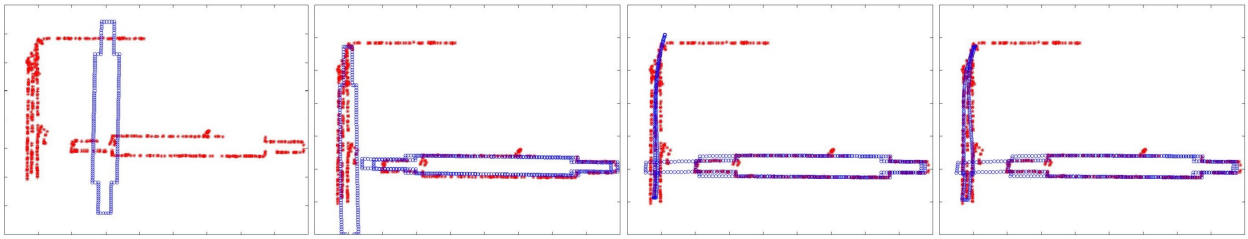


(c) Proposed

Figure 3.21: Point registration result of the opening with initial transnational difference.



(a) CPD



(b) Proposed

Figure 3.22: Point registration result of the opening with initial rotational difference..

Chapter 4

Robust Multiple Object Tracking

4.1 Introduction

Object tracking has many applications in all kinds of sequential observations including videos, radar measurement and LiDAR point clouds. Different from object detection introduced in Chapter 3 and Chapter 7, tracking focuses on aggregating the temporal data and associating objects in different time steps. It is also an essential block in the perception system of robots because localization, mapping and planning all require temporal information, such as motion, of the surrounding objects. There are usually many objects in the workspace of a robot which results in a multiple object tracking (MOT) problem. MOT is still a open problem and there are many aspects being explored in terms of both the solution and evaluation. There are some comprehensive surveys [77, 78] for the classic MOT problem which mostly discusses the process model of objects, optimization of temporal association and evaluation metrics. Similar to object detection, deep learning methods [79, 80, 81] are taking place in MOT and outperforming classic methods especially in the appearance model and motion prediction of objects. However, MOT relies more on the probabilistic model for robustness concern which is not well handled by current deep learning methods. For robustness and precision issue, classic tracking method is preferred in this work for tracking of objects in the robot workspace.

Wafers are represented by elliptic segments in our work. Elliptic segment tracking assigns the segments in different frames to wafer objects. Unlike tracking in the scenario faced by SLAM, local features of wafers are not unique, so it is impossible to associate segments by matching feature points. Ellipse tracking is the most related work known to our problem. The tracking of ellipses belonging to the same 3D object is discussed in [82, 83] and [84], but they only associate spherical objects without location prediction. [85] and [86] implement filtering methods such as Kalman filter for single ellipse tracking. The methods are not extendable to multiple objects because object association is not discussed. Authors of [87, 88] propose a particle filter for multiple ellipses tracking. However, points are partitioned by clustering instead of object detection, and the association of two ellipses close to each other

is problematic. We adopt the tracking-by-detection framework in multiple object tracking (MOT) for the tracking of multiple wafers in this thesis. Wafers are assumed to be detected in Section 3.2 with high accuracy.

Section 4.2 constructs a tracking-by-detection framework for multiple wafer tracking. The innovation of this work is the design of the robust association module discussed in Section 4.3. An EM-based association procedure is proposed taking the geometric model of wafers into account. The resulting association module has two levels where the first level softly associates individual points with loose constraint. The second level optimizes the hard association between segments which provides the decision for assigning predicted wafers to observed wafers. Noise model is included and outliers can be identified.[1]

4.2 Tracking by Detection Framework for Wafers

The framework is illustrated in Fig. 4.1. Wafers are indexed by k=$1, 2, \cdots$ and the wafer detection algorithm extracts elliptic segments $z_{l,t}$'s from the image at frame t. Each segment $z_{l,t}$ is a set of pixels. $z_{k,t-1}$ is the detected segment at frame $t-1$ and is already registered to wafer k. $z_{l,t}$ is the segment l detected at t and need to be registered. Section 4.2 introduces

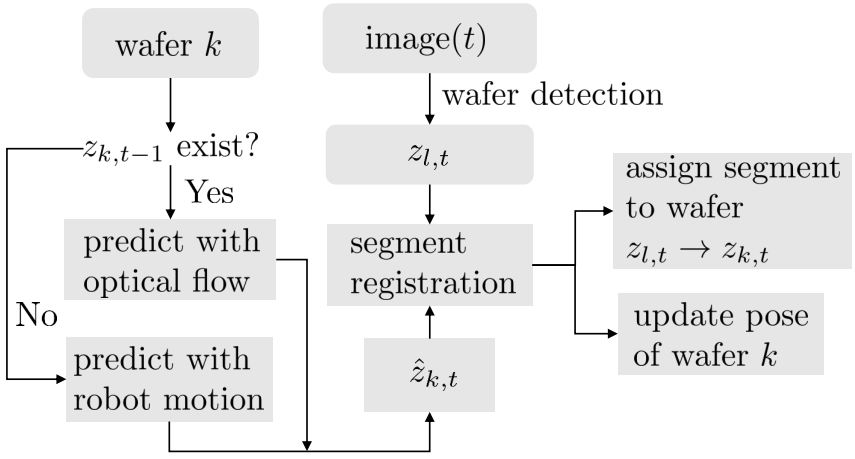


Figure 4.1: The tracking-by-detection framework for wafer tracking.

the method used for predicting the states of the wafer and its position in the coming image. Section 4.2 introduces the algorithm for updating the state of multiple wafers in the overall tracking algorithm.

[1]This work includes materials from the author's previously published paper [41].

Prediction of Elliptic Segments

Prediction is important for associating wafers in different frames, especially when there is no specific feature to distinguish elliptic segments of different wafers. For predicting a segment $z_{k,t}$ of index k at frame t, there are two cases. The first case is when the segment $\hat{z}_{k,t-1}$ of the last frame $t-1$ is visible. The second case is when the last visible segment is at frame $\tau < t-1$. Two prediction strategies are designed for these two cases. Optical flow is found to be more accurate for consecutive prediction and is used for the first case. Deepflow [89] implemented in OpenCV extracts the flow at the elliptic segment $z_{k,t-1}$ and the prediction $\hat{z}_{k,t}$ is simply made by adding the flow to the segment. The short-term optical flow prediction within several frames is found to be quite accurate.

When a segment is not continuously visible, it is impossible to use optical flow. In this case, $\hat{z}_{k,t}$ is predicted from projecting the estimated 3D wafer w_k to the image plane. w_k takes the location-quaternion expression where the first three elements are the XYZ location of the wafer center, and the last four elements are the quaternion of its rotation relative to the world frame. The projection is written as 4.1)

$$\begin{bmatrix} uw \\ vw \\ w \end{bmatrix} = K_I \left(r R_E R_w \begin{bmatrix} \cos\theta \\ \sin\theta \\ 0 \end{bmatrix} + R_E T_w + T_E \right), \tag{4.1}$$

where $[R_E(r_t), T_E(r_t)]$ is the camera extrinsic matrix as a function of robot pose r_t, r is the radius of the wafer, K_I is the intrinsic of the camera, R_w, T_w are the rotation and translation matrices of the wafer in the world frame. Note both w_k and the robot pose r_t are required, while these two variables can only be estimated. As a result, the accuracy of prediction is coupled with the calibration performance of the whole system, which is not robust in the long term. Therefore, the prediction based on robot motion is not preferred in the long term. If a wafer is invisible for a long time, it is discarded to avoid inaccuracy of the prediction.

Track Management

Algorithm 4 gives the pseudo-code of wafer tracking. Several sets are used to store the data and correspondence. Z_t stores all elliptic segments detected at frame t. The indexes of segments in Z_t does not contain tracking information. One *track* keeps all the segments of the same wafer. The matrix A maps the index k of a track to the index $A(k,t)$ of a segment. Z_{track} stores the last visible segments of tracks. Segments in Z_{track} are used for prediction.

There are two edges of one wafer. The distance between the upper and lower edges of a wafer is much smaller than distances to edges of other wafers. Therefore, it is simple to determine whether two elliptic segments belong to the same wafer and which segment is the upper or lower edge.

Algorithm 4 Wafer Tracking Algorithm

Z_0 // set of segments at frame 0
$Z_{track} \leftarrow \emptyset$ // tracks (set of latest segment on track)
θ // registration of segments
// Initializes tracks and registration
for all $z_{k,0} \in Z_0$ **do**
  $Z_{track} \leftarrow Z_{track} \cup z_{k,0}$
  $\theta(|Z_{track}|, 0) \leftarrow k$
end for
// Tracking
for all $t \in \{1, 2, \cdots, T\}$ **do**
  Detect elliptic segments Z_t
  $\widehat{Z}_{track} \leftarrow$ Prediction(Z_{track})
  $\theta \leftarrow$ Segment Registration$(\widehat{Z}_{track}, Z_t)$
  //update tracks and registration
  for all $k \in \{1, \cdots, |Z_t|\}$ **do**
    $z_{k,t} \leftarrow Z_t(k)$
    if $\sum_l \theta(k, l) = 1$ **then**
      $Z_{track} \leftarrow Z_{track} \cup z_{k,t}$
      $\theta\left(|Z_{track}|, t\right) \leftarrow k$
    else
      find l s.t. $\theta(k, l) = 0$
      $Z_{track}(l) \leftarrow z_{k,t}$
      $\theta\left(l, t\right) \leftarrow k$
    end if
  end for
end for

4.3 Hierarchical Segment Registration Using GMM

Data association becomes difficult when unique features of an object cannot be extracted, which is the case of our wafer tracking problem. Greedy association strategy is taken in [90] to register prediction to the closest observation but it does not work for our problem in the report. Point registration, instead, does not require features and is based on the Bayesian inference of all observed points and is more robust. There are many kinds of point registration methods including filtering [91], probability density [54], Gaussian mixture model (GMM) [65] and etc. This report uses the GMM type to associate elliptic segments between consecutive frames. Different from general point registration which considers one object, there are multiple objects, aka elliptic segments, in our case. The contribution of this section is to develop a GMM model of points that can also output association of multiple objects. The idea is to introduce a latent variable of object association, which originates

from [64].

Suppose X is the set of points $x_i \in \mathbb{R}^d, i = 1, \cdots, M$ in frame t. Each point x_i belongs to an elliptic segment $z_{k,t}, k = 1, \cdots, K$ detected by a mapping $k = f(i)$. $\hat{X}$ is the set of points $\hat{x}_j \in \mathbb{R}^d, j = 1, \cdots, N$ predicted for frame t. Each point $\hat{x}_j$ belongs to a predicted elliptic segment $\hat{z}_{l,t}, l = 1 \cdots, L$. The GMM defined in (4.2) assumes X is generated from $\hat{X}$ by a mixture of Gaussians centered at $\hat{x}_j$'s

$$p(x_i|s_i = j; \theta) \sim \mathcal{N}(\hat{x}_j, \sigma^2), \tag{4.2}$$

where s_i is a latent variable. $s_i = j$ means a point $x_i \in X$ is generated by $\hat{x}_j$. θ denotes the parameters of the model. In point registration, θ is the transformation that transforms predicted points to the actual observed points. The goal is to find the optimal probability of point association Q^*_{ij} and the optimal parameters θ^* given prediction and observations [92]. This is done by maximizing the likelihood (MLE) in (4.3)

$$\begin{aligned} \theta^* &= \arg\max_{\theta} \sum_i \log p(x_i; \theta), \\ P^*_{ij} &= p\left(s_i = j | X, \theta^*\right). \end{aligned} \tag{4.3}$$

The typical solution to MLE is expectation maximization (EM). Another solution is the variational inference (VI) which naturally brings in the prior distribution $p(\theta)$ by modeling θ as a Gaussian random field. Two solutions have the same form of alternating from E-step and M-step. E step is a Bayesian update of P_{ij} in (4.4)

$$P^{(k+1)}_{ij} = p(s_i = j|x_i; \theta^{(k)}) = \frac{p\left(x_i|s_i = j; \theta^{(k)}\right)}{\sum_j p\left(x_i|s_i = j; \theta^{(k)}\right) p\left(s_i = j\right)}, \tag{4.4}$$

where $p(x_i|s_i = j; \theta)$ is the Gaussian distribution in (4.2), $p(s_i = j)$ is the prior distribution of point associations which are usually assumed to be constant. $\theta^{(k)}$ is the estimated θ at kth iteration. M-step is an optimization that updates the variable θ. If θ is modeled as a random field, its prior distribution is included and the M-step is in (4.5)

$$\theta^{(k)} = \arg\max_{\theta} \sum_{i=1}^{M} \sum_{j=1}^{N} P^{(k)}_{ij} \log \frac{p\left(x_i, s_i = j; \theta\right)}{P^{(k)}_{ij}} + \log p\left(\theta\right). \tag{4.5}$$

In point registration methods, θ is the transformation of points. In the wafer tracking case, there are multiple elliptic segments, and θ is the assignment of segments. The GMM in this case is modified from (4.2) to (4.6)

$$p(x_i|s_i = j; \theta) \sim \begin{cases} \text{uniform}, & \theta\left(k, l\right) = 0 \\ N\left(\hat{x}_j, \sigma^2\right), & \theta\left(k, l\right) = 1 \end{cases}. \tag{4.6}$$

Note that s still exists which is the point association. s being a latent variable means points association is soft. A predicted point can be associated with multiple observed points.

$\theta \in \mathbb{R}^{K\times L}$ is a binary-valued matrix which denotes the assignment of segments. It is a hard association as the desired output is a one-to-one registration of elliptic segments. $\theta(k,l) = 1$ if and only if segment $z_{k,t}$ is assigned to predicted segment $\hat{z}_{l,t}$. Note k, l are defined as $k := f(i), l := g(j)$. If $\theta(k,l) = 0$ but point x_i is assigned to $\hat{x}_j$, the distribution is considered as uniform which means it is a noise point. σ^2 is a parameter denoting the covariance of the Gaussian component. The update of E-step introduced in (4.4) becomes (4.7) assuming the prior $p(s_i = j) = \frac{1}{N}$

$$p(s_i = j|x_i;\theta) = \frac{\omega + \theta(k,l)\left(\exp\left(-\frac{||x_i-\hat{x}_j||^2}{2\sigma^2}\right) - \omega\right)}{N\omega + \sum_j \theta(k,l)\left(\exp\left(-\frac{||x_i-\hat{x}_j||^2}{2\sigma^2}\right) - \omega\right)}, \tag{4.7}$$

where ω, an empirical parameter, represents the density of the uniform distribution. The uniform distribution of x_i is limited on the support region which is the image area. No probability is given to points outside the image range. The output of the E-step is also denoted as a matrix $Q(i,j) := p(s_i = j|x_i;\theta)$ for simplicity. The M-step introduced in (4.5) becomes (4.8) which solves the segment registration parameter θ.

$$\begin{aligned}
\theta &= \arg\max \sum_{i=1}^{M}\sum_{j=1}^{N} Q(i,j) \log \frac{p(x_i, s_i = j;\theta)}{Q(i,j)} \\
&= \arg\max \sum_{i,j} Q(i,j) \log p(x_i|s_i = j;\theta)\, p(s_i = j) \\
&= \arg\max \sum_{i,j} Q(i,j)\, \theta(k,l) \left(c_1 - 2\sigma^2 \log\omega - \|x_i - \hat{x}_j\|^2\right),
\end{aligned} \tag{4.8}$$

where $c_1 = 2d\sigma^2 \log(\sqrt{2\pi}\sigma)$. In this work, $d = 2$ because the registration is in an image plane. In this problem, it is desired to have a one-to-one registration between segments, and this adds a constraint to the problem which requires the sum of each column and each row of θ to be less or equal to 1. The optimization problem constructed for solving the M-step is then

$$\begin{aligned}
&\max_{\theta} \sum_{k,l} C(k,l)\,\theta(k,l) \\
&\text{s.t.} \sum_{k=1}^{K} \theta(k,l)\, 1, \forall l \in \{1,2,\cdots,L\} \\
&\qquad \sum_{l=1}^{L} \theta(k,l)\, 1, \forall k \in \{1,2,\cdots,K\} \\
&\qquad \theta(k,l) \in \{0,1\}, \forall k, l,
\end{aligned} \tag{4.9}$$

where

$$C(k,l) := \sum_{I(k),J(l)} Q(i,j)\left(c_1 - 2\sigma^2 \log\omega - \|x_i - \hat{x}_j\|^2\right)$$
$$I(k) := \{i|\, f(i) = k\}, J(l) := \{j|\, g(j) = l\}. \tag{4.10}$$

The optimization problem of (4.9) is equivalent to a bipartite matching problem and can also be solved analytically by the Munkres algorithm [93]. If $\theta(k,l) = 0$ for all l, then the observed segment k is not registered to any predicted segment, meaning it is a new object. A new track is initialized in this case. If $\theta(k,l) = 0$ for all k, then the predicted segment l is not registered to any observed segment, meaning the wafer is not visible in this frame. The overall procedure of the proposed hierarchical assignment method is summarized as pseudo-code in Algorithm 5

Algorithm 5 Hierarchical Segment Registration Algorithm

$Z_t, \hat{Z}_t$ // sets of all detected and predicted elliptic segments
$X_t := x_i | x_i \in v_k, \forall v_k \in Z_t$ // set of all detected edge points of segments
θ // registration of elliptic segments
$C \in \mathbb{R}^{|Z_t| \times |\hat{Z}_t|}$
for all $\hat{z}_l \in \hat{Z}_t$ **do**
  for all $\hat{x}_j \in \hat{z}_l$ **do**
    $X_{\text{kNN}} \leftarrow \text{kNN}(X_t, \hat{x}_j, K)$ // find kNN of $\hat{x}_j$
    for all $x_i \in X_{\text{kNN}}$ **do**
      calculate $Q(i,j)$ // E-step
      find k s.t. $x_i \in z_k \in Z_t$
      $C(k,l) += Q(i,j)(c - 2\sigma^2 \log\omega - \|x_i - \hat{x}_j\|^2$
    end for
  end for
end for
$\theta \leftarrow \text{Hungarian}(C)$
return θ

In conclusion, the modified GMM allows soft association between points and binary association between segments. Noise model is also included as a uniform distribution. The solution is a one-to-one mapping of elliptic segments between predicted and observed ones.

4.4 Chapter Summary

We introduce the tracking-by-detection framework in this chapter to tracking multiple objects in the robot workspace making use of the detected objects. A hierarchical registration algorithm is proposed for the association module in the tracking framework. It uses soft point-to-object association in the first level and optimizes the hard object-to-object assignment in the second level. The proposed method takes advantage of the detector and the prior

shape information of objects. It is more robust than the popular feature descriptor-based algorithm in the multiple wafer tracking scenario which has non-unique features.

Chapter 5

Auto-teaching with Visual Inertial Sensor Fusion

5.1 Introduction

Auto-teaching, which is also called auto-calibration, and mapping of processing tools is an important efficiency feature for wafer processing. The super-clean work environment in the semiconductor industry prevents human interference. When failure occurs, such as misplacement of wafers, the robot needs to re-initialize the work environment to avoid collision. Auto-teaching emphasis on precisely calibrating the robot without touching or entering the workspace. After auto-teaching, the coordinate of checkpoints such as carriers and chamber openings relative to the robot frame is precisely known. The robot is able to navigate and transfer wafer safely again in the FI after a much shorter shut down time.

This Chapter aims to estimate 3D poses of objects as well as calibrate parameters of the hand-eye system. It requires solving the motion of the robot and objects together with the sensor readings from the camera and encoders. Estimation of self-motion and object poses in a video is usually done through tracking and 3D reconstruction. Simultaneous localization and tracking (SLAM) is a standard solution for this general problem [94]. SLAM deals with uncertainties in the image and forward kinematics because of noise and calibration error. Filtering, such as Kalman filter and particle filter, forms one category of SLAM [95, 96, 97]. Another category is the more modern SLAM formulation based on probabilistic graphical model (PGM), or called graph-based SLAM [98]. Graph is a better interpretation of the SLAM problem than Markov process used in filtering-based methods, which represents the sparse probabilistic dependencies between states and measurements. With the PGM constructed, efficient solvers [99, 100] are available to optimize motion states and point locations locally step by step, as well as optimize globally to minimize the effect of nonlinearity.

The hand-eye system in Chap 2 designed for the wafer handling robot is capable of measuring the surrounding objects. In addition to vision sensors, the hand-eye system also obtains very precise motion data from its encoders and it is important to utilize this

information. The auto-teaching framework for the hand-eye system is illustrated in Fig. 5.1.

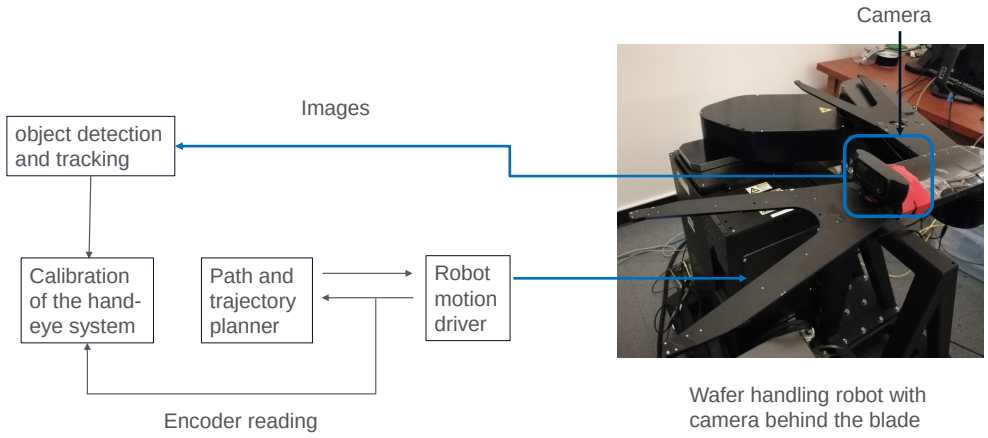


Figure 5.1: The auto-teaching framework which takes the camera image and motor angles as input and calibrate parameters of the hand-eye system.

Section 5.2 designs an Unscented Kalman Filter (UKF) to fuse the trajectory information regressed from vision sensor and motor readings. In addition to filtering-based methods, calibration of motion parameters and estimation of object poses in a video is usually done through 3D reconstruction. It is found that general SLAM methods [101, 102] do not perform well in the wafer processing scenario discussed in this work. Feature matching and tracking performances are deteriorated by large a number of outliers due to the lack of texture. Moreover, 3D reconstruction using point-based bundle adjustment does not consider the prior knowledge of the accurately known shape of objects. Section 5.3 proposes a SLAM-MOT framework which incorporates the MOT proposed in Section 5.3 and takes advantage of the shape prior of objects in the workspace.[1]

5.2 Unscented Kalman Filter Design for Visual-Inertial Sensor Fusion

Motivation

Recovering the accurate trajectory of the camera and reconstructing objects in 3D is an essential step for auto-teaching. Several publicly available SLAM packages including LSD-SLAM [30], ORB-SLAM2 [31] and DSO [32] are implemented for a test scenario of auto-teaching in Fig. 5.2. The result of the most state-of-the-art ORB-SLAM2 is picked, which

[1]This work includes materials from the author's previously published paper [41].

has best performance and is robust among many types of sensors including RGB-D, stereo and monocular cameras. A 50s length of video is recorded at 20Hz, resulting in a total of 1000 frames. The 3D reconstruction result is shown in Fig. 5.3 and Fig. 5.4. The result with the first 400 frames in Fig. 5.3 looks acceptable although the reconstructed wafers are very noisy. The result with all of the 1000 frames in Fig. 5.4 looks terrible. Although SLAM algorithms claim long-term correspondence, the reconstructed point cloud still shows large misalignment. In particular, the board is reconstructed into two pieces with different slopes.

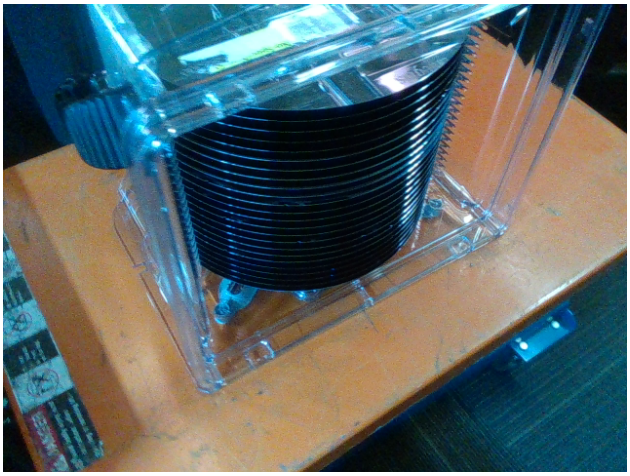


Figure 5.2: The wafer carrier used for testing the visual SLAM algorithm.

The long-term reconstructed point clouds above shown the estimated trajectory is drifted if only visual SLAM is used for localization. There is another observation of the camera pose calculated from robot encoder readings. However, the robot is assumed to be not calibrated, so robot encoder and forward kinematics do not provide accurate camera pose. Therefore, it is beneficial to combine both sensors for both calibration parameter estimation and camera localization. For the nonlinear system, an Unscented Kalman Filter is designed.

Unscented Kalman Filter Design

The UKF is based on the intuition that visual SLAM output, as an incremental measurement, is more accurate in the short term but drifts in the long term, while the camera pose calculated from encoder readings has consistent error. It estimates the joint zeroing error θ_0, which belongs to the calibration parameters and camera poses $X = [Tx, Ty, Tz, x, y, z, \omega]^T$, where (Tx, Ty, Tz) is the translation and (x, y, z, ω) is the quaternion form of camera rotation. SLAM update is used for state transition so as to handle its additive process noise. Robot encoder reading with inverse kinematics $f_k^{-1}(X)$ is used as the measurement function.

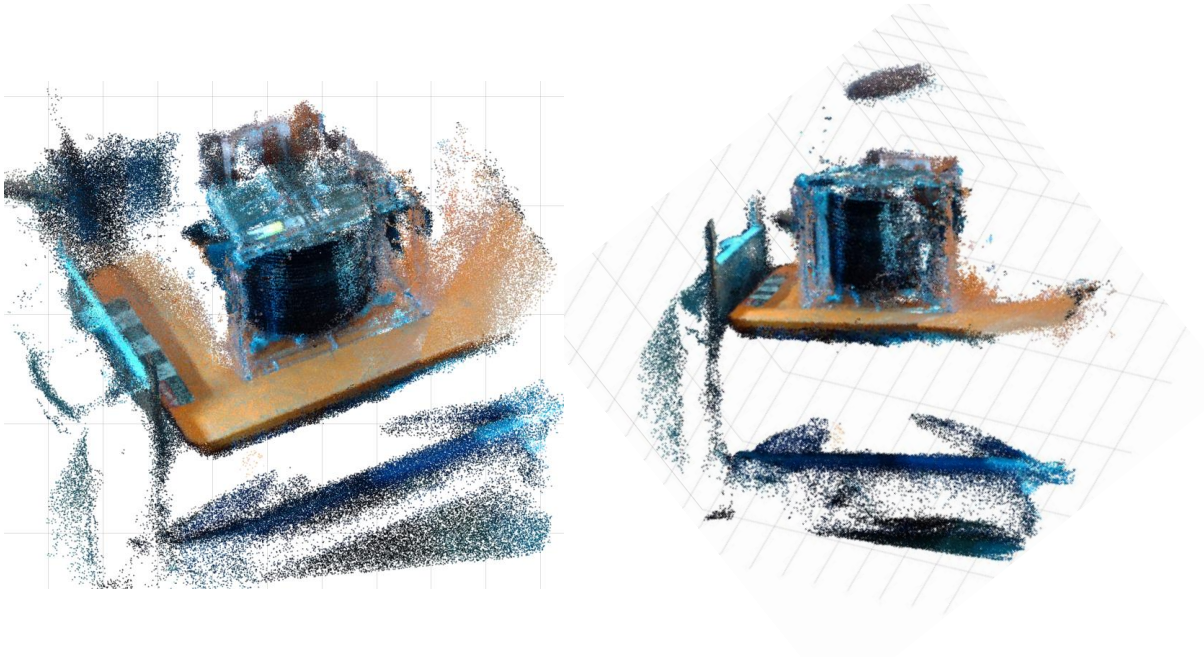


Figure 5.3: 3D reconstructed point cloud of ORB2-SLAM using the first 20s (400 frames) of the video.

The system function is (5.1)

$$X(k+1) = \begin{bmatrix} T(k) \\ {R_E}^{-1}(R_E(E(k))\, u_R(k)) \\ \theta_0(k) \end{bmatrix} + \begin{bmatrix} u_T(k) \\ 0 \\ 0 \end{bmatrix} + \begin{bmatrix} w_T \\ w_E \\ 0 \end{bmatrix}, \tag{5.1}$$
$$y(k+1) = {f_k}^{-1}(T(k+1) + v_T, E(k+1) + v_E) - \theta_0(k+1) + v_\theta,$$

where $X(k) = [T(k); E(k); \theta_0(k)]$ is a 9-degree vector consisting of camera translation, Euler angle representation of rotation and joint zeroing error. Inputs are $u_T(k)$ for change of translation, $u_R(k)$ for incremental rotation matrix and 0 for zeroing error. $R_E(\cdot)$ is the transformation from Euler angles to the rotation matrix. Note there is no processing noise on the joint zeroing error. The state transition updates camera transformation using inputs u_T, u_R from SLAM algorithm. The observation function calculates joint angles from current state with measurement noise w. The system has two nonlinear functions which are the rotation update and inverse kinematics $f_k^{(}-1)(\cdot)$, so Kalman Filter cannot be directly applied for state estimation. The unscented form is implemented to address the nonlinearity.

Since we attempt to fuse measurement from two sensors with UKF, the covariances of two sensors are important to leverage the confidence. Unfortunately, it is hard to get the actual covariances of noises in the real world. Therefore, covariances are tuned by trial and

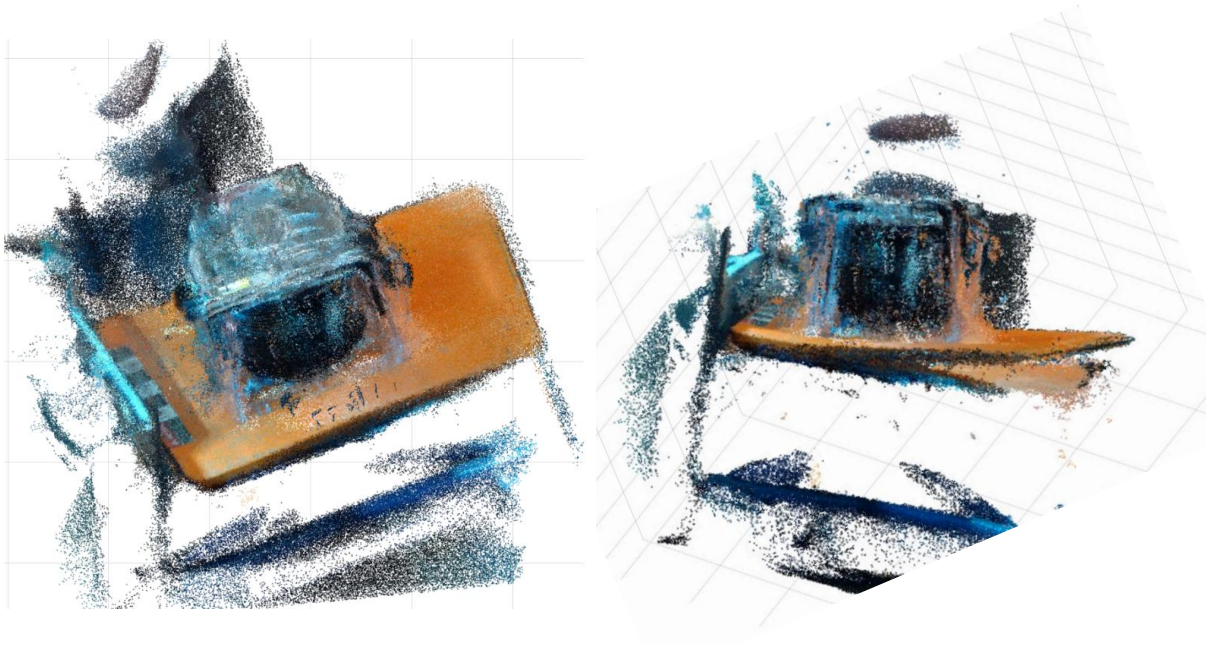


Figure 5.4: 3D reconstructed point cloud of ORB2-SLAM using the first 50s (1000 frames) of the video.

error in experiment. The resulting covariances are listed below

$$
\begin{aligned}
Var\{w_T\}^{\frac{1}{2}} &= \begin{bmatrix} 30 & 0 & 0 \\ 0 & 30 & 0 \\ 0 & 0 & 1.5 \end{bmatrix} (\text{mm}), Var\{w_E\}^{\frac{1}{2}} = \begin{bmatrix} 0.2 & 0 & 0 \\ 0 & 0.2 & 0 \\ 0 & 0 & 0.2 \end{bmatrix} (\text{deg}), \\
Var\{v_T\}^{\frac{1}{2}} &= \begin{bmatrix} 2 & 0 & 0 \\ 0 & 2 & 0 \\ 0 & 0 & 0.5 \end{bmatrix} (\text{mm}), Var\{v_E\}^{\frac{1}{2}} = \begin{bmatrix} 1.0 & 0 & 0 \\ 0 & 1.0 & 0 \\ 0 & 0 & 1.0 \end{bmatrix} (\text{deg}), \\
Var\{v_\theta\}^{\frac{1}{2}} &= \begin{bmatrix} 0.05 & 0 & 0 \\ 0 & 0.05 & 0 \\ 0 & 0 & 0.05 \end{bmatrix} (\text{deg}).
\end{aligned} \tag{5.2}
$$

It is demonstrated that with a careful tuning of noise covariance, the performance is improved in the experiment in Section 5.4.

5.3 Visual Intertial SLAM-MOT with Shape Information

The SLAM-MOT structure, which is a modification of standard SLAM, is taken in this report. SLAM-MOT is short for "SLAM with multiple object tracking". Standard SLAM has two types of states: self-motion and landmark points, while SLAM-MOT has three types by adding the state of surrounding objects. The original purpose of SLAM-MOT is to separate dynamic objects from the static map to increase accuracy as a modification of visual SLAM [103]. Objects we deal with in this chapter are static so there is no concern of dynamic objects, but the framework of SLAM-MOT helps in points listed below:

- It uses the detection and tracking result of objects [104, 105]
- Optimizations of object poses can be decoupled [94]
- By optimizing objects instead of individual points, shape constraints of wafers can be used.

The front-end of SLAM-MOT, which involves extracting the objects and associating them in the time space, are discussed in Chapter 3 and Chapter 4. This section focuses on setting up the back-end which involves building the PGM and solving the optimization problem of graph inference. In addition, a visual-inertial SLAM-MOT is constructed where the motion of the robot is considered to make use of all sensor measurements. We apply the proposed method to the wafer inspection and calibration scenario of the wafer handling robot and show that it outperforms the state-of-the-art visual SLAM method.

Factor Graph Construction

The optimization is formulated as a Bayesian inference problem of the probabilistic graphical model (PGM) of the system. The PGM is represented as a factor graph [106] which is popular in recent bundle adjustment methods [98]. The factor graph representation is especially effective when there is revisit to the same position in the trajectory called loop closure. And it leverages the uncertainty of different sensors. Since the repeatability of the wafer robot is very good, detecting loop closure is quite easy. This section constructs the factor graph for the visual-inertial SLAM-MOT used in this work. Poses of the robot and poses of objects are linked by sensor measurements from encoders and the camera. Their joint distribution is required to calculate the posterior of poses from measurements. The expression of the joint distribution may be complex, but it can be simplified by the dependence of variables. PGM consists of a collection of distributions that factorizes the joint distribution [106]. Then the joint distribution can be written as multiplication of simpler distributions. Denote x_t as the state of the robot at time t, w_k is the pose of object k, γ is the static parameters such as calibration parameters, u_t is the encoder measurement or control input and z_t is the

observation of object such as edges of wafers, then the factorization is (5.3)

$$P\left(x_{1,\cdots,T}, w_{1,\cdots,K}, z_{1,\cdots,T}, \gamma, u_{1,\cdots,T}\right) = \prod_{C\in\mathcal{C}} \phi_C\left(v_C\right), \tag{5.3}$$

where $\mathcal{C}$ is the set of all maximal cliques of the graph and v_C are the random variables in the maximal clique C. However, finding all maximal cliques is not straight forward and another commonly used structure of PGM is the *factor graph* which emphasizes on the factorization of joint distribution. In practical cases, it is sometimes more straight-forward to write out the factor graph form rather than the undirected graph form. For example, in our method, observations are assumed to be independent with each other, then each observation, either from vision or from encoder, becomes a factor of the joint distribution as in (5.4)

$$P\left(x_{1,\cdots,T}, w_{1,\cdots,K}, z_{1,\cdots,T}, \gamma, u_{1,\cdots,T}\right) = \prod_{C\in\mathcal{C}_1} \phi_C\left(x_C, z_C, w_C, \gamma\right) \prod_{C\in\mathcal{C}_2} \psi_C\left(x_C, u_C, \gamma\right), \tag{5.4}$$

where ϕ_C is related to the camera observation model, and ψ_C is related to the robot kinematic model. In SLAM-MOT the distributions of each factor are assumed to be Gaussian with nonlinear mean as function of robot and object. The Gaussian assumption results in a joint nonlinear least square.

The factor graph of the wafer calibration problem is modeled as shown in Fig. 5.5. There are two kinds of nodes for inference x_t, w_k, namely camera pose at time t and wafer pose of wafer k. Observations are $z_{k,t}, u_t$, namely elliptic segment and robot motion. t denotes the frame number and k denotes index of the wafer. Each observation corresponds to a factorization of the joint distribution. Wafer pose w_k and camera pose x_t are linked to

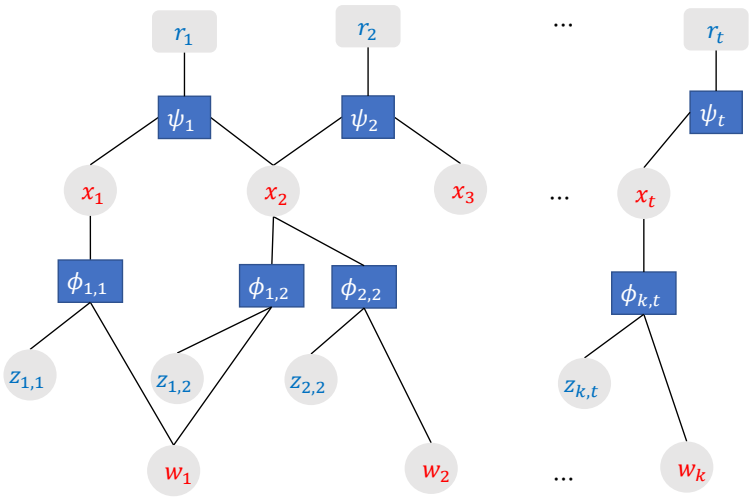


Figure 5.5: Illustration of the factor graph of the proposed visual-inertial SLAM-MOT.

the camera observation factor if wafer k is visible at frame t. The joint distribution of

the factorization $\phi_{k,t}$ is modeled through the projection error model $e_p(x_t, w_k, z_{k,t})$ which is supposed to be a Gaussian distribution

$$\phi_{k,t}(x_t, w_k, z_{k,t}) = \frac{1}{2\pi\sqrt{\left|\Omega^z_{k,t}\right|}} \exp\left(-\frac{1}{2} e_p^T \Omega^z_{k,t} e_p\right), \tag{5.5}$$

where $\Omega^z_{k,t} \in \mathbb{R}^{2\times 2}$ is the information matrix. $e_p \in \mathbb{R}^2$ is the projection error in the image. Camera poses x_t, x_{t+1} of two consecutive frames $t, t+1$ are linked to robot motion factor. The joint distribution of the factorization ψ_t is modeled through the robot motion error model $e_r(x_t, x_{t+1}, u_t)$ and is supposed to be Gaussian

$$\psi_t(x_t, x_{t+1}, u_t) = \frac{1}{\sqrt{(2\pi)^7 \left|\Omega^r_t\right|}} \exp\left(-\frac{1}{2} e_r^T \Omega^r_t e_r\right), \tag{5.6}$$

where $\Omega^r_t \in \mathbb{R}^{7\times 7}$ is the information matrix and obtained from the prior knowledge of the precision of the robot. e_r is the pose error of the end effector of the robot which holds the camera. It is a 7-dimensional vector containing the 3-dimensional position error and the 4-dimensional quaternion error, which expresses the full pose error in $SO(3)$. The inference of x_t, w_k is done by maximizing the likelihood of observation. According to the property of the factor graph, it is equal to the optimization

$$\max_{x_t, w_k} \left\{ \sum_{k,t} \log\left(\phi_{k,t}\left(x_t, w_k, z_{k,t}\right)\right) + \sum_t \log\left(\psi_t\left(x_t, x_{t+1}, u_t\right)\right) \right\}. \tag{5.7}$$

By the Gaussian assumptions, the optimization becomes a least square

$$\max_{x_t, w_k} \left\{ \sum_{k,t} \left\| e_p\left(x_t, w_k, z_{k,t}\right) \right\|^2_{\Omega^z_{k,t}} + \sum_t \left\| e_r\left(x_t, x_{t+1}, u_t\right) \right\|^2_{\Omega^r_t} \right\}, \tag{5.8}$$

where $\dot{|||}_\Omega$ denotes the 2-norm with kernel Ω. The information matrices $\Omega^z_{k,t}, \Omega^r_t$ become weights of each term. Factors of their eigenvalues are the ratio of confidence between different kinds of error in different directions. The value information matrices are determined by the inherent noise level of images and robot encoder.

Parameterization and Optimization for Wafers

The optimization introduced above falls into the standard bundle adjustment case in SLAM. However, the goal of this work is the calibration of wafers and the robot-camera system instead of camera poses and point cloud reconstruction. Therefore, there are two major differences introduced, namely camera pose parameterization and error functions, compared to standard bundle adjustment. The camera pose x_t is parameterized by the calibration

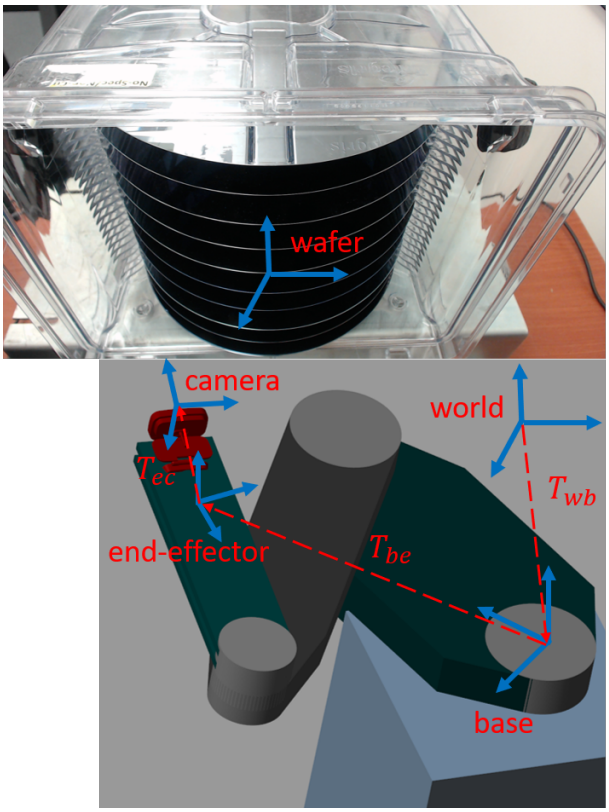


Figure 5.6: Illustration of the coordinate system.

parameters of the robot-camera system. The coordinate system including the wafers and the hand-eye system is shown in Fig. 5.6. According to (2.2), transformation matrix from world to the camera is written as a function of time

$$T_{wc}(u_t) = T_{wb}T_{be}(u_t)T_{ec}, \tag{5.9}$$

where T_{wb} and T_{wb} are fixed because the wafer handling robot is fixed, and the camera is mounted on the end effector. The relative robot motion r_t is defined as

$$r_t := T_{wc}(t)T_{wc}(t+1)^{-1} = T_{wb}T_{be}(t)T_{be}(t+1)^{-1}T_{wb}^{-1}, \tag{5.10}$$

where $r(t)$ is the change of the extrinsic matrix because extrinsic is the inverse of transformation matrix. T_{be} is determined by the robot forward kinematics with calibration parameters γ, so $r(t)$ is parameterized as $r(u_t; \gamma, T_{wc})$. Unknown parameters γ, T_{wb} are standard calibration parameters of robot kinematics and modeling of them are discussed in details in a lot of references[107, 108, 109, 110, 111] and Section 2.2. T_{be} is cancelled out and does not appear in the optimization. In the experiment, it is assumed that the forward kinematics of

the robot is accurate and thus we do not need to calibrate γ. However, the method is able to be generalized to optimize γ when forward kinematics is inaccurate.

The robot motion error $e_r(\cdot)$ is the difference between camera motion predicted from camera poses x_t, x_{t+1} and from robot forward kinematics r_t

$$e_r\left(x_t, x_{t+1}, r_t\right) = ||x_t \oplus r_t \ominus x_{t+1}||. \tag{5.11}$$

The operators are defined for $SO(3)$. Definition and calculation of Jacobians are adopted from [112]. Substitute (5.10), the parameterization of r_t into e_r gives a function of optimization variables

$$e'_r\left(x_t, x_{t+1}, \gamma, T_{wb}\right) := e_r\left(x_t, x_{t+1}, r_t\right). \tag{5.12}$$

The projection error function $e_p(\cdot)$ in (5.13) is the deviation from the conic curve equation instead of Euclidean distance. Instead of the point-wise landmark reprojection error used in other SLAM methods, e_p only penalizes the projection error of points off the surface of the object. As long as the projection is on the surface, the loss is zero which means point-to-point correspondence required in SLAM is not needed here. The point-to-object correspondence results in a less strict loss function, but since point-to-point correspondence usually leads to lots of outliers, our proposed formulation actually improves the final precision in experiment.

$$\begin{aligned} e_p\left(x_t, w_k, z_{k,t}\right) &:= p^T C p = p^T \left(KE(x_k, w_k)\right)^{-T} C_0 \left(KE(x_k, w_k)\right)^{-1} p, \\ p &= \begin{bmatrix} u & v & 1 \end{bmatrix}^T, \end{aligned} \tag{5.13}$$

where p is the homogeneous coordinate of observed edge points on the image plane, K is the camera intrinsic matrix and E is the extrinsic matrix from wafer frame to camera frame. Note E is not defined relative to the world frame so it is only a function of the camera pose x_t and wafer pose w_k. The quadratic kernel C_0 of the conic form is defined in wafer coordinates. The variables are solved over a standard sliding-window optimization as stated in [102].

5.4 Experimental Results

Experiments are conducted with a wafer handling robot and wafers in a standard carrier of the semiconductor manufacturing industry. A camera is held by the end effector of the robot and a video is recorded while the robot moves around the carrier. The experiment aims to verify the accuracy of calibrated wafer 3D poses and robot kinematic parameters.

UKF Result

Some example images for the UKF experiment are shown in Fig. 5.7 with an Intel RealSense RGBD camera. The intrinsic of the camera is pre-calibrated while the joint zeroing error of the robot is not calibrated. ORB-SLAM2 [101] is used to calculate the SLAM update for UKF. Two channels of trajectory inputs from two sensors are shown in Fig. 5.8. The red one

is the trajectory calculated by the SLAM algorithm from the camera and the blue one is the trajectory calculated by the forward kinematics from the motor encoders. Two trajectories start from the same beginning but deviate by a large extent in the long term. The fused trajectory calculated by the proposed UKF method is shown in yellow.

(a) (b) (c)

Figure 5.7: Example images captured for the UKF experiment. The hand-eye system moves around the wafer carrier filled with wafers.

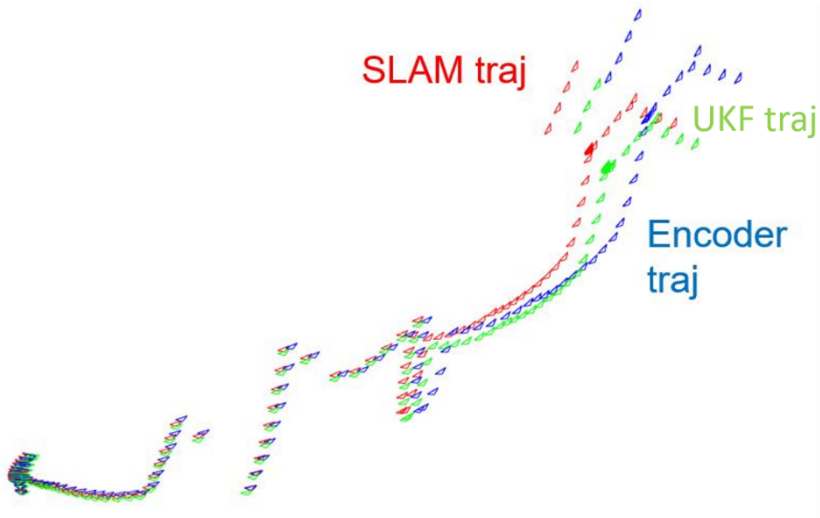
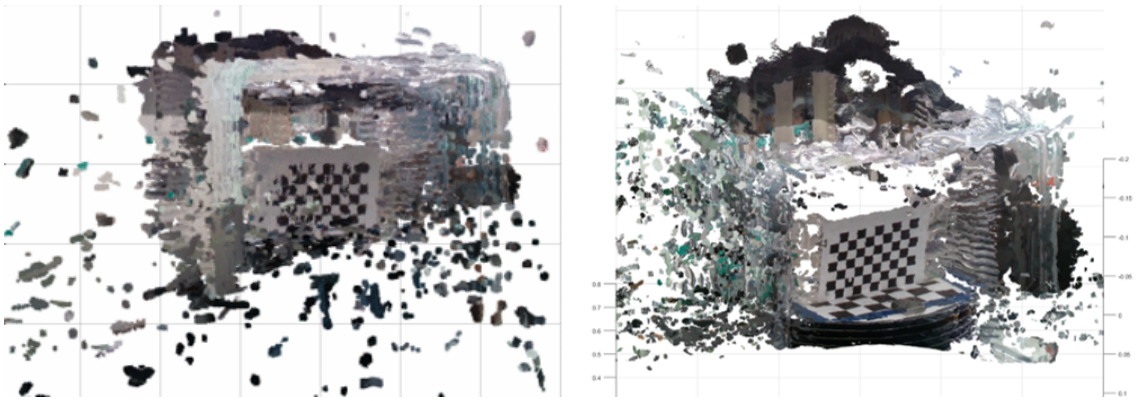


Figure 5.8: Trajectories calculated from two sensors and the trajectory fused by UKF. Each point represents a bird's eye view of the camera pose.

The comparison of reconstructed point clouds before and after UKF estimation is shown in Fig. 5.9. The carrier is not reconstructed clearly due to its transparency. Two checkerboards are both reconstructed correctly, as well as the gauge block on the wafer. The reconstructed point clouds after UKF are of much higher quality than those reconstructed just using SLAM. The wafers, though seem to be reconstructed correctly, still produce a lot

of scatters in the 3D space. This is because the IR depth sensor relies heavily on reflection, and there are always areas of the wafer which completely reflects the infrared light, resulting in wrong depth measurement. The reflection areas are different in different views so the whole wafer is reconstructed but it also generates scatters in each view. This kind of error inherits from the properties of the depth sensor and cannot be addressed by the UKF algorithm. The reconstruction problem of wafers is resolved by the proposed SLAM-MOT method and shown in the following experiment section.



(a) Point clouds before UKF using SLAM.

(b) Point clouds after UKF, fusing SLAM and encoder reading.

Figure 5.9: Example images captured for the UKF experiment. The hand-eye system moves around the wafer carrier filled with wafers.

Visual-inertial SLAM-MOT Result

A photo of the actual initial setting of the system is shown in Fig. 5.10. There are 26 slots in the carrier, numbered from 1 to 26 denoting slots from the bottom to the top. Slots are parallel to each other with a machined distance of 10 mm $\pm$ 0.5 mm. The world frame is defined to be at wafer 1 at the bottom in slot 4. Wafer 1 is accurately placed to represent the base of the carrier. The surface of this wafer is the XY plane of the world coordinate and its norm is the Z axis. All resulting poses are expressed relative to wafer 1.

The robot is a dual-blade wafer handling robot with 5 degrees of freedom. Only 4 joints are enabled in this experiment to provide a full X, Y, Z and yaw angle reachability of the end effector. The monocular camera is mounted on the end effector of the robot. The size of the sliding window for optimization is 5.

Twelve wafers are inserted into the carrier in slots listed in Table 5.1. Wafer 12 is tilted on purpose to simulate misplacements. The robot is commended to move around. Wafers are tracked, and their 3D poses are estimated by the proposed optimization. It is worth noticing that we only have the ground truth value ($\pm$0.1 deg) for rotations. Therefore, for

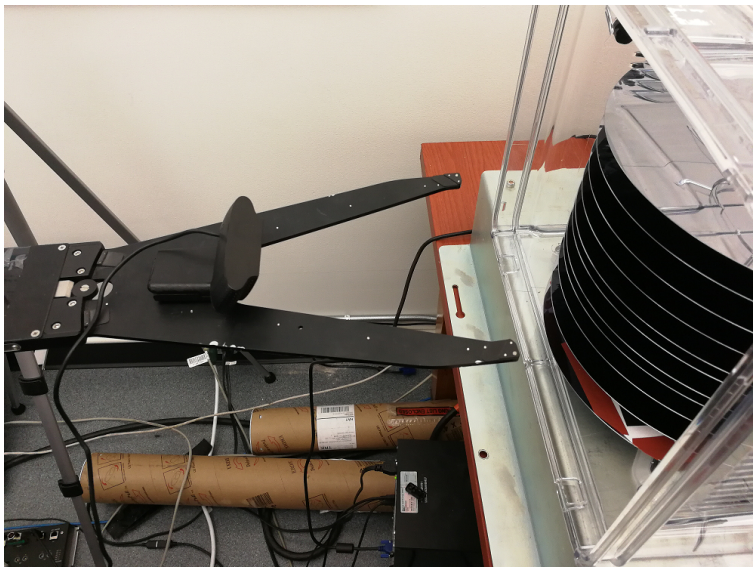


Figure 5.10: Photo of the initial poses

translations, only standard and max deviation are discussed in Table 5.1. It is shown that the rotation estimation accuracy is 0.5 deg and most errors are within 0.2 deg.

Table 5.1: Table of optimized wafer translation and rotation errors. Z axis is plotted in Fig. 5.12

wafer	slotNum	standard(max) deviation X,Y translation (mm)	reconstruction error X,Y rotation (deg)
1	4	[0.0(0.0), 0.0(0.0)]	[+0.0, +0.0]
2	7	[0.1(0.2), 0.1(0.1)]	[+0.2, +0.0]
3	10	[0.1(0.1), 0.1(0.2)]	[−0.2, +0.4]
4	13	[0.2(0.3), 0.1(0.2)]	[−0.2, +0.3]
5	15	[0.1(0.2), 0.1(0.2)]	[+0.2, +0.5]
6	17	[0.3(0.8), 0.1(0.2)]	[−0.3, +0.0]
7	19	[0.3(0.7), 0.1(0.2)]	[−0.2, +0.6]
8	21	[0.4(0.7), 0.1(0.3)]	[−0.2, −0.2]
9	22	[0.1(0.3), 0.0(0.1)]	[−0.2, −0.1]
10	23	[0.3(0.7), 0.1(0.2)]	[−0.2, −0.4]
11	24	[0.0(0.4), 0.1(0.1)]	[+0.0, +0.1]
12	25	[0.1(0.1), 0.0(0.0)]	[−0.1, −0.2]

The estimated calibration parameters are shown in Fig. 5.11. Parameters of T_{wb} converge as more frames are taken into account. Fig. 5.12 shows the trend of estimated Y, Z translation

Table 5.2: Wafer arrangement and theoretical resolutions.

wafer	slotNum	Resolution w.r.t. translation (pixel/mm)	Resolution w.r.t. rotation (pixel/deg)
1	4	$[0.32, 0.25]$	$[0.44, 0.39]$
2	7	$[0.31, 0.24]$	$[0.39, 0.35]$
3	10	$[0.31, 0.22]$	$[0.30, 0.31]$
4	13	$[0.26, 0.19]$	$[0.24, 0.26]$
5	15	$[0.23, 0.16]$	$[0.20, 0.20]$
6	17	[0.19.0.13]	$[0.16, 0.18]$
7	19	$[0.18, 0.12]$	$[0.14, 0.05]$

of wafers. The trend of X axis is similar to Y and its statistics is listed in Table 5.1. It is shown that the variations of estimation in Y axis of all wafers are bounded by 0.2 mm, and X axis has larger variation up to 0.8 mm.

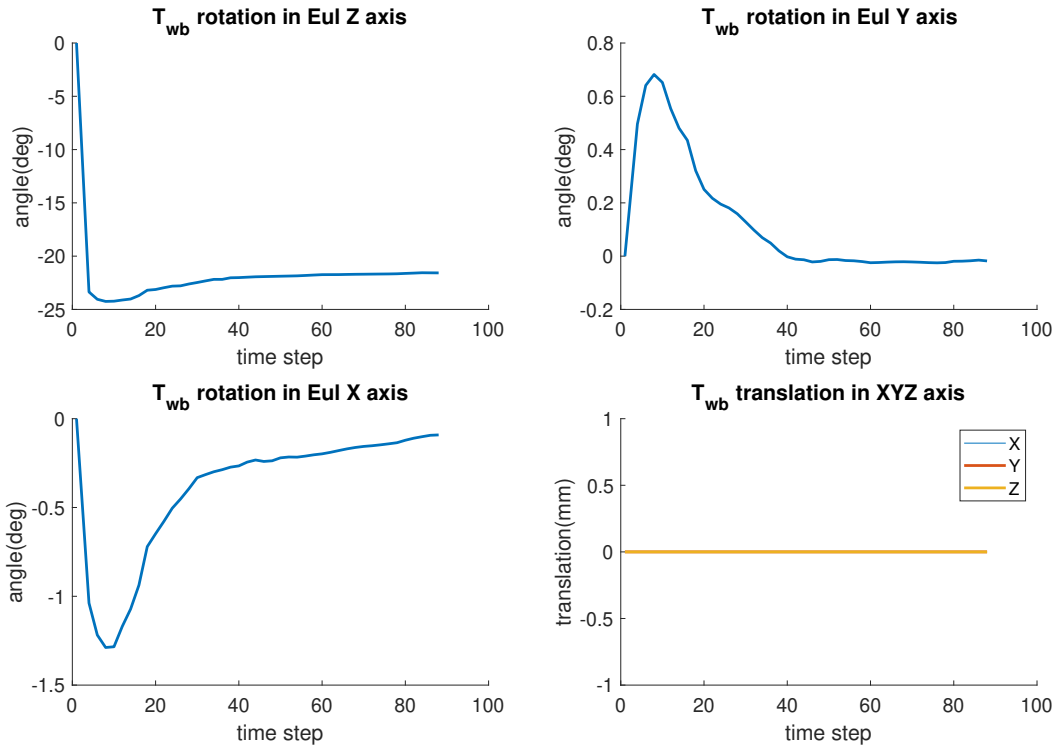


Figure 5.11: Estimated calibration parameters T_{wb} of the system

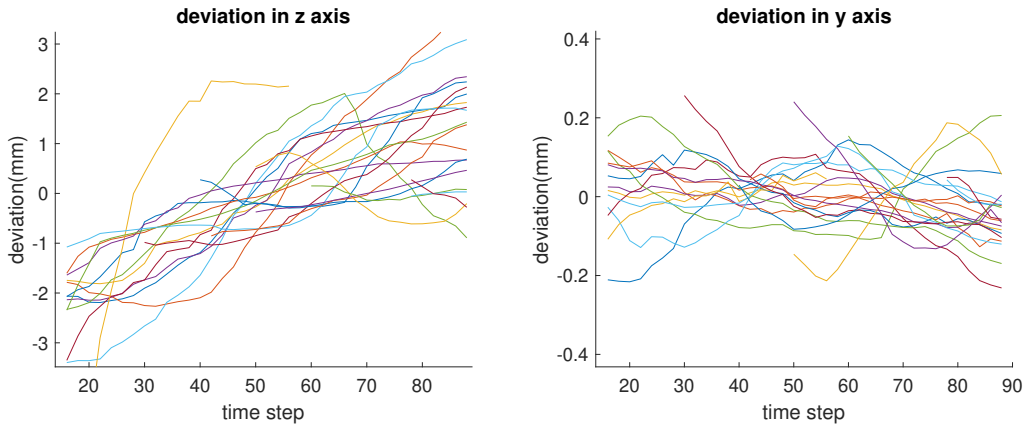


Figure 5.12: Estimated translation in Y and Z axis of all 12 wafers. The trend in X axis is similar to Y, but with higher variation.

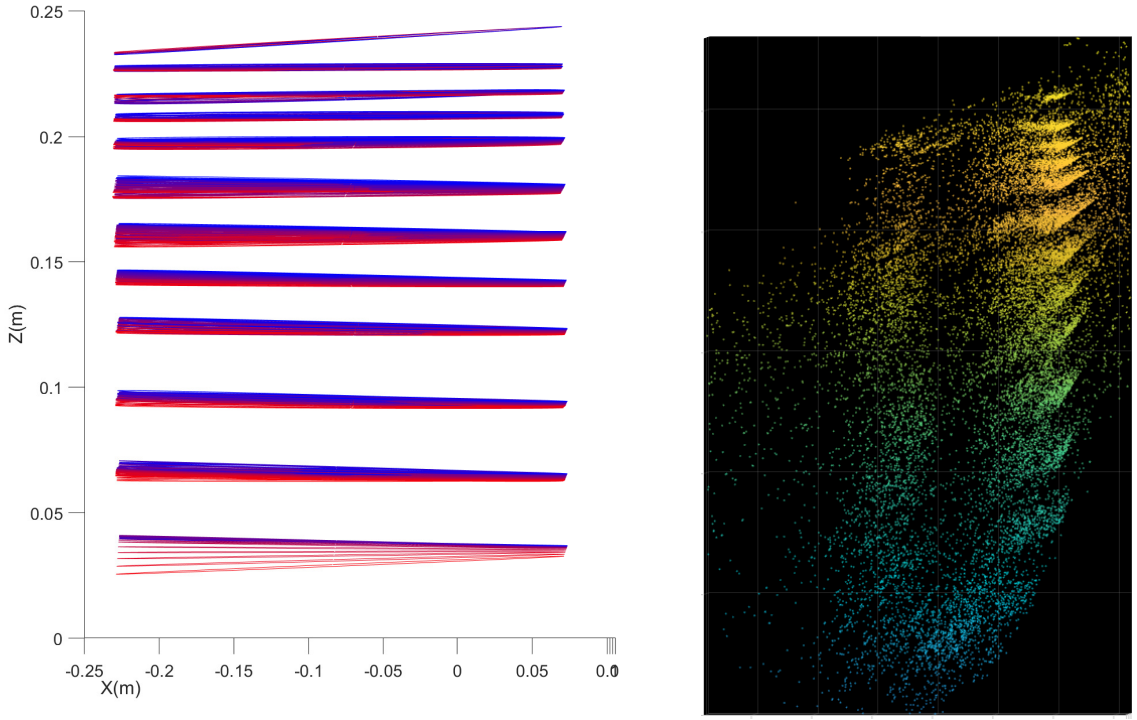
The reconstructed 3D map of wafers is shown in Fig. 5.13(a). Red ones are the intermediate results and blue ones are the converged results. Compared with the reconstructed 3D map by ORB-SLAM2[101] in Fig. 5.13(b), which is very noisy. Front view of the reconstructed point clouds by two methods are compared in Fig. 5.14. The precision of the proposed method apparently outperforms the regular SLAM method.

It is observed in the 3D calibration result that the final accuracy at positions of X and Z axes are not as good as those of Y axis and rotation angles. The calibration error has a trend of increasing at these two axes. Here we derive the theoretical resolution of the 3D calibration experiment in this paper. Another experiment is designed to explain and compensate for the bias. The resolution is simply calculated through the derivative of the projection function mapping 3D pose of the wafer to the ellipse in the image, which is given in (5.14) of Jacobians. By substituting the nominal poses of wafers and cameras, the theoretical resolutions of each wafer are given in Table 5.2 using (5.14).

$$\begin{aligned} \frac{du}{dT} &= \frac{E^T K_1^T \overline{x}^T w - u E^T K_3^T \overline{x}^T}{w^2}, \\ \frac{dv}{dT} &= \frac{E^T K_2^T \overline{x}^T w - v E^T K_3^T \overline{x}^T}{w^2}, \end{aligned} \tag{5.14}$$

where K_1, K_2, K_3 are the three rows of the intrinsic matrix K_I, E is the extrinsic matrix of the camera. T is the transformation matrix of the wafer. w is the scaling factor of the homogeneous coordinate after projective transformation and $\overline{x}$ is the homogeneous coordinate in the wafer frame. They are defined below

$$\begin{aligned} w &:= K_3 E T J_T \overline{x}, \\ \overline{x} &:= \begin{bmatrix} r\cos\theta & r\sin\theta & \pm h/2 & 1 \end{bmatrix}^T, \end{aligned}$$



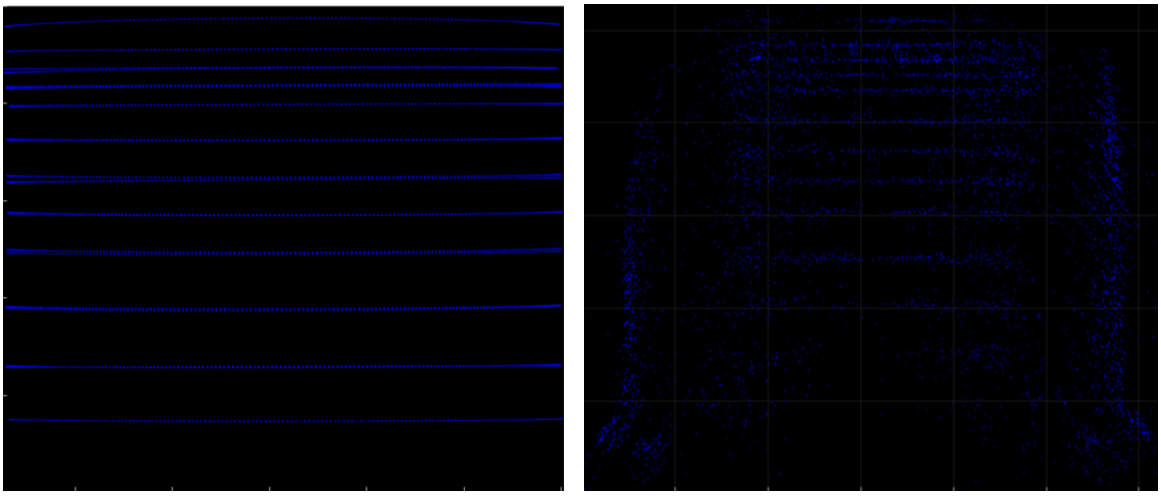
(a) Proposed method. The color changes from red to blue as optimization proceeds.

(b) ORB-SLAM2 [101].

Figure 5.13: 3D maps of wafers reconstructed by two methods. The proposed method reconstructs wafers in an object level and is more precise. The ORB-SLAM2 reconstructs each pont and the result is very noisy.

where $r = 150\text{mm}, h = 1\text{mm}$ are the radius and thickness of the wafer, θ varies from 0 to 2π representing different points on the wafer boundary. According to the detection robustness test in Section 3.5, the pose of the wafer is unlikely to be affected by the arrangement of wafers. The light condition in the 3D calibration experiment is consistent and much better than the detection experiment. Even in the worst light case of the robustness test, the bias of the detected ellipse is about 2-3 pixels, so there should be other factors that cause the error in the 3D calibration.

A calibration board is used for decoupling the factors that contributes to the bias. It is stuck on the surface of a wafer and inserted in a slot as shown in Fig. 5.15. The calibration board, being Lambertian, is not affected by the light condition. The robot runs the same trajectory but 3D location of corner points of the calibration board is estimated instead of the wafers. The estimated positions are plotted in Fig. 5.16. The slopes in X axis of these calibration board corners extrapolate to the slope in X axis of estimated wafer centers.



(a) Proposed. (b) ORB-SLAM2

Figure 5.14: Front view of reconstructed 3D point clouds of the wafers.

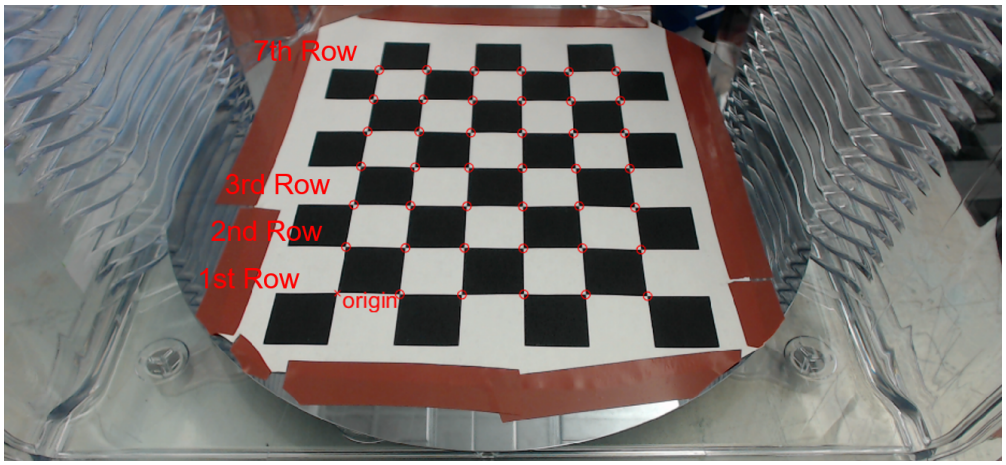


Figure 5.15: Factor experiment with calibration wafer.

More detailed analysis on the result of the light change test gives distribution of each tracked point on the wafer. An example is shown by Fig. 5.17. The blue points show the distribution of tracked wafer points under different light conditions. Coordinates of the points are normalized with respect to the nominal position so the y-axis are deviations. It shows the light color variation will introduce bias on the vertical axis of the image by 2 pixels. It will decrease the accuracy, but the variance (or precision) are affected too much because we already modeled a $\mathcal{N}(0, 1)$ Gaussian noise.

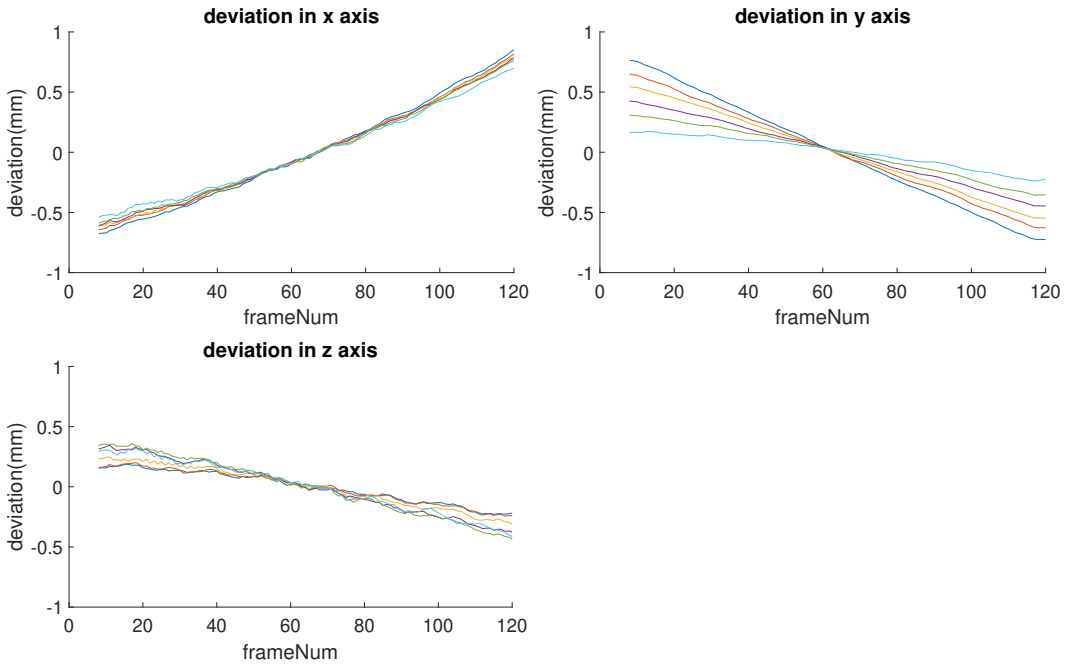


Figure 5.16: Estimated position of calibrated corners in the first row.

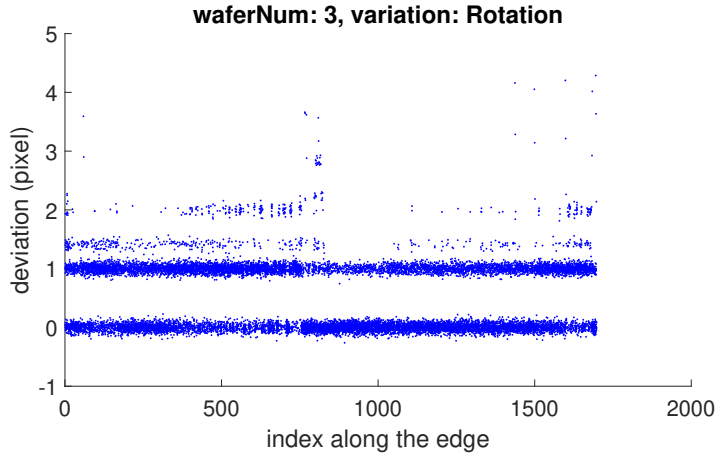


Figure 5.17: Distribution of tracked wafer points under different light conditions.

5.5 Chapter Summary

This chapter introduces auto-teaching of the hand-eye system which calibrates the parameters of the system and reconstructs the objects in 3D space through multi-view perception. Auto-teaching fuses the vision and motion information for precise calibration. Two approaches are proposed. First, an UKF is designed for fusing the trajectories from visual and motion sensors and joint zeroing errors of the system can be calibrated. Second, a SLAM-MOT framework is proposed for simultaneous calibration of parameters and reconstruction of objects. The experiments conducted with wafers in the workspace show that the proposed methods effectively fuse information of two sensors. Qualitative visualization and quantitative comparison show that they are more precise and robust than the state-of-the-art visual SLAM method.

Chapter 6

Active Auto-teaching of Wafer Handling Robot

6.1 Introduction

This chapter aims to further close the loop of the system, which is called active auto-teaching by optimizing the poses where the robot captures images for calibration. Measurement poses are configurations for a hand-eye system equipped with visual inspection devices [2, 113, 114, 115]. *Active calibration* is the term that refers to optimizing the measurement poses for calibration performance. Finding the set of optimal measurement poses corresponds to an optimization problem with unclear objective function, since there is no best scalar valued mapping from estimated parameters to calibration performance. Many criterions have been proposed in robotics, such as [116], [117], [118], [119], [120] and [109], following the principle of optimal experimental design (OEM) in statistics [121]. People have also been evaluating the performance of different criterions on robot systems [122, 123]. The physical constraints of the robot, such as reachable space, dynamic constraints and obstacles produce non-linear constraints.

To efficiently solve the optimization problem, the most commonly used method is to select poses from a finite pre-defined pose set, which will result in a convex optimization problem [124]. The DETMAX and its modifications are robust solvers and have been used for a very long period [125, 126, 127]. Other optimization methods are summarized in Table 6.1. The improvement of choosing optimal measurement poses for robot systems is significant. In terms of kinematics, a table of references showing the improvement of calibration from random selected pose [128, 109] does not provide mean value without standard deviation. However, the accuracy with OEM is always much higher.

The proposed active auto-teaching framework is shown in Fig. 6.1 which closes the loop by outputting the next measurement pose when calibration is updated. Different from the scenario used in 5, this chapter focuses on the scenario with a chamber opening where only one target object is present. The perception problem is less complex and it is easier to

Reference	Robot	Metric	Baseline poses	Optimized poses
[6]	PUMA-560	3D error	6.9 ± 2.6mm	0.3 ± 0.3mm
[129]	Prototype	Planar error	0.2 ± 0.1mm	0.1 ± 0.01mm
[128]	SLIMA	3D error	1.7 ± 0.0mm	0.9 ± 0.0mm
[130]	Gough	3D error	0.4 ± 0.4mm	0.02 ± 0.01mm
[109]	Surgical	3D error	2.7 ± 0.0mm	1.5 ± 0.0mm

Table 6.1: Effect of active calibration on robot kinematics, in terms of end effector position.

evaluate the benefit of introducing the active auto-teaching technique. Section 6.2 introduces the most related area of research called optimal experimental design (OEM) and reviews the related works. Section 6.3 introduces the estimation system being dealt with and the concrete design method which returns way points of trajectory. Section 6.4 gives the experiments conducted on the wafer handling robot navigating in front of a chamber opening and shows the proposed active auto-teaching improves the accuracy of the system.

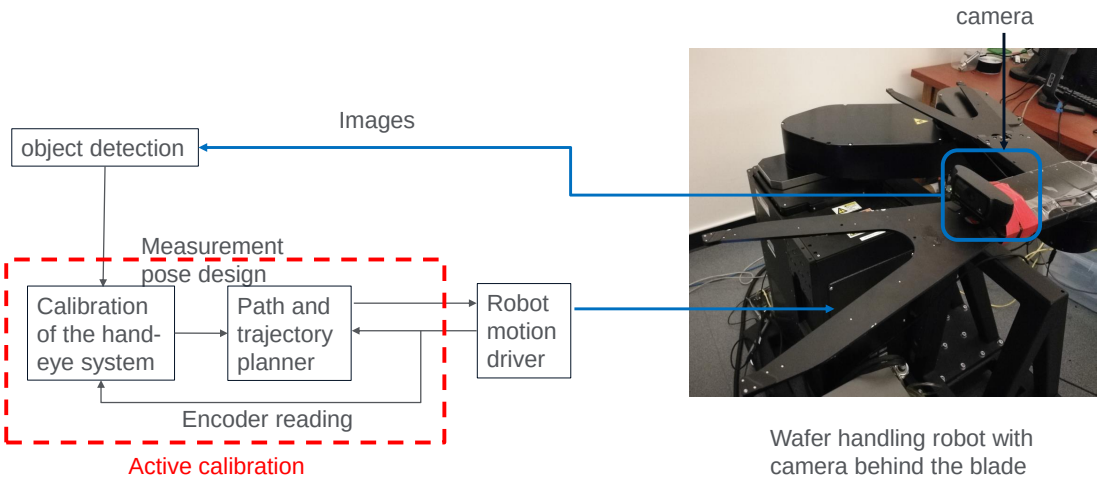


Figure 6.1: The active auto-teaching framework where the calibration module outputs the measurement poses.

6.2 Optimal Experimental Design (OEM)

Optimal experimental design, also called design of experiments and active calibration in robotics, starts from [131] for fitting univariate polynomials. A more historic example can be

the Chebyshev points for polynomial fitting that excludes instability. Modern works extend to optimizing parameter identification for systems whose observations can be controlled. Optimal experimental design is an active area in statistics [132] and applied to robotics for calibration. Both vision-based and non-vision-based calibrations of robotic systems use this technique, including kinematics, dynamics and target objects summarized in Table 6.2 and Table 6.3.

Fisher Information Matrix

The mathematical definition [132] uses measure in real analysis. Suppose measurements are free of choice from space X called design region, a design ξ of the experiment is a measure over X. In practice, the measure is usually discrete and we can denote ξ as

$$\xi = \left\{ \begin{array}{cc} x_1 & x_2 \cdots x_n \\ \omega_1 & \omega_2 \cdots \omega_n \end{array} \right\}, \tag{6.1}$$

which means the design has n supports at observations $x_1, x_2, \cdots, x_n$ and they are assigned weights of $\omega_1, \omega_2, \cdots, \omega_n$. Denote the output of observation as y, the observation equation returns the posterior probability $p(y|x, \theta)$ of y given x and system parameters θ. To estimate unknown parameters θ, maximum likelihood is often chosen, which is (6.2)

$$\theta(\xi) = \arg\min_{\theta \in \Omega} \int_X \log p(y|x, \theta)\, d\xi. \tag{6.2}$$

Sensitivity is of the most concern for the estimation which is the gradient of the likelihood function with respect to θ. The sensitivity is called Fisher score vector as (6.3)

$$f(y|x, \theta) = \frac{\partial}{\partial \theta} \log p(y|x, \theta). \tag{6.3}$$

Then the Fisher Information Matrix (FIM), which constitutes an indicator of sensitivity, is defined as the expectation of its covariance

$$M(\xi, \theta_0) = \int_X E\left\{ f(y|x, \theta_0)\, f^T(y|x, \theta_0) \right\} d\xi, \tag{6.4}$$

where θ_0 is the true parameter of the system. For nonlinear system with additive Gaussian measurement noise denoted as (6.5), its FIM has closed form as (6.6)

$$y_i = h(x_i, \theta) + v_i, \qquad v_i \sim N\left(0, \sigma_i^2\right), \tag{6.5}$$

$$M(\xi, \theta_0) = \sum_{i=1}^{n} \left(\frac{\omega_i}{\sigma_i^2} \cdot \frac{\partial h(x_i, \theta)}{\partial \theta} \frac{\partial h(x_i, \theta)}{\partial \theta}^T \right). \tag{6.6}$$

For linear systems, the FIM is the same as the Hessian matrix used for the recursive least square. Sometimes the inverse of M is used which is called the variance-covariance matrix D. Typically, the smaller the eigenvalues of D is, the better design we get.

Optimality Criterion

The FIM is a matrix so it is impossible to directly optimize on it. The optimal experimental design problem involves choosing a scalar valued function ψ as cost function, which forms the optimization problem in (6.7)

$$\xi^* = \arg\min_{\xi \in \Xi(X)} \psi\left[M\left(\xi\right)\right], \tag{6.7}$$

where $\Xi\left(X\right)$ is the feasible region. In OEM, there are A-optimality, D-optimality and E-optimality which are the trace of D, determinant of M and minimum singular value of M, respectively. In the area of robot calibration, five popular criterions are used called observability indexes. Another new index called O_6 is proposed in a very recent work [109]. Formulas are listed in Table 6.2, and they correspond to different optimality. The singular values $\sigma_1 \geq \sigma_2 \geq \cdots \geq \sigma_m$ are supposed to be sorted. Since they are all using the singular values of FIM, they have theoretical relationships with each other and some are proved to be equivalent under certain circumstances [122].

References	Observability Index	Optimality	$\psi[M(\xi)]$
[127, 58, 133, 134]	O_1	D-optimality	$(\sigma_1\sigma_2\cdots\sigma_m)^{1/m}$
[135, 136]	O_2	K-optimality	σ_m/σ_1
[129]	O_3	E-optimality	σ_m
[137]	O_4	-	σ_m^2/σ_1
[138]	O_5	-	$(\sigma_1^{-1}+\sigma_2^{-1}+\cdots+\sigma_m^{-1})^{-1}$
[109]	O_6	-	$(\sigma_1^{-P}+\sigma_2^{-P}+\cdots+\sigma_m^{-P})^{-1}$

Table 6.2: Review of observalibity indices.

Optimization Approaches

All optimality criterions require calculation of the singular value, which is definitely non-linear and non-convex in most cases. If the space of observations is discrete and finite, then it can be convexified to a selection problem. In other cases, gradient (first and second order) and sampling-based methods are used. Table 6.3 is a summary of optimization methods used in robot calibration works.

Gradient-based method is the fastest for high dimensional design space but it usually converges to an unsatisfactory local minimum in complex problems. The derivative $\phi(x,\xi)$ of the cost function ψ in direction $\overline{\xi}$ can be estimated as

$$\phi(x,\xi) = \lim_{a\to 0^+} \frac{1}{a}\{\psi[(1-a)M(\xi) + aM(\overline{\xi})] - \psi[M(\xi)]\}. \tag{6.8}$$

References	Calibration Model	Solver category	Optimization Method
[135, 136]	Dynamics	Sampling-based	Genetic
[139, 140]	Arbitrary	Sampling-based	Stochastic
[109]	Kinematics	Sampling-based	Particle Filter
[116]	Kinematics	Sampling-based	Particle Swarm
[127, 129]	Kinematics	Finite Set	DETMAX
[136, 133, 134, 141]	Dynamics	Gradient-based	-
[142, 143, 130]	Kinematics	Gradient-based	-

Table 6.3: Review of optimization approaches applied on robotic systems.

For gradient method, the optimality condition is stated by the General Equivalence Theorem for all the optimality criterions:

Theorem 6.2.1 (The General Equivalence Theorem)**.** *The following statements are equivalent for optimal design ξ^*.*

- *The design ξ^* minimizes $\psi[M(\xi)]$.*
- *The design ξ^* maximizes the minimum over X of $\varphi(x, \xi)$.*
- *$\varphi(x, \xi^*) = 0$ for all $x \in supp(\xi^*)$.*
- *$\varphi(x, \xi^*) \geq 0$ for all $x \in X$.*

Sampling-based method is not trapped by the local minimum but usually takes tens of minutes to several hours for a robot calibration problem. It is worth noticing that the classic DETMAX method provides a good strategy for sampling by iteratively adding several best samples and subtracting several worst samples in the design. For sampling-based method, a useful theorem is the theorem claiming finite support even for constrained cases:

Theorem 6.2.2. *Let $\varsigma(x)$ be the cost of observation taken at point x. For a constrained experiment optimization problem*

$$\begin{aligned} &\min_{\xi \in \Xi(X)} \psi\left[M\left(\xi\right)\right] \\ &s.t. \sum_{i=1}^{n} \omega_i \varsigma\left(x_i\right) \leq C. \end{aligned} \tag{6.9}$$

Suppose there are l constraints and M is of order m, then there exists at least one optimal design containing no more than $n = l + m(m+1)/2$ supporting points.

When the potential arrangement of measurements is already fixed and choices are finite, say $|X| < \infty$, the optimization can be reduced to an integer optimization problem in (6.10) by designing the ψ to be additive. $\psi[M(\delta(x_i))]$ is pre-calculated as the contribution of each single measurement $x_i \in X$.

$$\xi^* = \underset{\omega_1,\ldots,\omega_{|X|}\in\{0,1\}}{\arg\min} \sum_{i=1}^{|X|} \omega_i \cdot \psi\left[M\left(\delta\left(x_i\right)\right)\right]. \tag{6.10}$$

6.3 Greedy Measurement Poses Planning

OEM is utilized specially for the measurement poses design of the hand-eye system in this work. During the test of various kinds of planners, it is found that the vanilla approaches introduced in Section 6.2 do not produce good results. Therefore, a modified planner is designed based on the characteristics of robot hand-eye systems. This derives the calibration model and the optimization problem using dynamic programming. Some special properties are found for the OEM of the hand-eye system. We introduce a greedy planner based on an one-step horizon of dynamic programming, and we show that it has significant advantages over other methods.

Calibration Model of the Hand-eye System

For a point X in the 3D world frame and its corresponding pixel x in the 2D image plane, the projection equation depends on the extrinsic matrix $[R, t]$ of the camera. In this work, the calibration parameters γ we want to estimate include the object rigid transformation $[R_s, t_s]$, end effector to camera rigid transformation R_c, t_c and joint zeroing error $\delta\theta$. Their deviations from the nominal value are $[\delta R_s, \delta t_s]$ and $[\delta R_c, \delta t_c]$. The observation error model can be written as (6.11)

$$\delta x = \frac{-\left(K_{12} - xK_3\right) Q_1 \left[\begin{array}{cccc} R\left[X\right] & R & I & J + \frac{dRX}{d\theta} \end{array}\right]}{K_3 Q_1 \left(RX + T\right)} \left[\begin{array}{c} \delta R_s \\ \delta t_s \\ \delta t_c \\ \delta\theta \end{array}\right] + \frac{K_{12} Q_1 \left(RX + T\right)}{K_3 Q_1 \left(RX + T\right)}, \tag{6.11}$$

where K_{12} is the first two rows, K_3 is the third row of the intrinsic matrix K, J is the Jacobian matrix in (2.5) and $[X]$ denotes the matrix for the cross product of 3D vectors which is

$$[X] := \left[\begin{array}{ccc} 0 & -X_3 & X_2 \\ X_3 & 0 & -X_1 \\ -X_2 & X_1 & 0 \end{array}\right]. \tag{6.12}$$

Here δR_c is assumed to be pre-calibrated by the method stated in the previous work as Q_1 so it is not included in the error model. The effect of joint angle to rotation is

$$\frac{dRX}{d\theta} = \left[\frac{d\left(R\left(\theta\right)X\right)}{d\theta_1}, \frac{d\left(R\left(\theta\right)X\right)}{d\theta_2}, \cdots \frac{d\left(R\left(\theta\right)X\right)}{d\theta_q}\right] \in \mathbb{R}^{3\times q}, RX \in \mathbb{R}^3, \tag{6.13}$$

where q is the number of joint angles. By multiplying the norm $\hat{a}$ of the surface at point x in image, the observation function of the robot visual calibration system is thus linearized as (6.14)

$$\begin{aligned} y &= C\left[\begin{array}{cccc} \delta {R_s}^T & \delta {t_s}^T & \delta {t_c}^T & \delta \theta^T \end{array}\right]^T + y_0, \\ C &= \frac{-\widehat{a} \cdot \left(K_{12} - xK_3\right) Q_1 \left[\begin{array}{cccc} R\left[X\right] & R & I & J + \frac{dRX}{d\theta} \end{array}\right]}{K_3 Q_1 \left(RX + T\right)}, \\ y_0 &= \widehat{a} \cdot \frac{K_{12} Q_1 \left(RX + T\right)}{K_3 Q_1 \left(RX + T\right)}. \end{aligned} \tag{6.14}$$

One-step Horizon Planner

We suppose that the Kalman Filter is utilized for the auto-teaching of the hand-eye system. According to the derivation of FIM, it is equal to the posterior covariance matrix of the Kalman Filter without process noise, which means $M(k) = P(k)$ with the following dynamic system in (6.15)

$$\begin{aligned} x\left(k+1\right) &= x\left(k\right) \\ y\left(k\right) &= C\left(k\right) x\left(k\right) + v \\ P^{-1}\left(k+1\right) &= P(k)^{\text{-}\,1} + C^T\left(k\right) V^{-1} C\left(k\right). \end{aligned} \tag{6.15}$$

The posterior covariance matrix is a function of system variables including intrinsic matrix K, extrinsic matrix $[R, T]$, and object location X. More specifically, $P(k)$ is a function of joint angles θ, forward kinematics f_k, camera mounting parameter R_c, T_c and object location X. Among all variables, we are able to control θ during the auto-teaching. Then the optimization problem is formulated as (6.16)

$$\min_{\theta} \psi\left[P\left(n\right)\right] = \min_{\theta} \psi \left[\left(\sum_{k=1}^{n} C^T\left(k, \theta\right) V(k)^{-1} C\left(k, \theta\right)\right)^{-1}\right] \tag{6.16}$$

. The optimization above is unconstrained. However, for the camera system, the target object should always be kept visible in the image which brings in the visibility constraint in (6.17)

$$\left[\begin{array}{c} 0 \\ 0 \end{array}\right] \leq g_1\left(\theta\right) = \frac{K_{12}\left(R\left(\theta\right) X + T\left(\theta\right)\right)}{K_3\left(R\left(\theta\right) X + T\left(\theta\right)\right)} \leq \left[\begin{array}{c} W_{\max} \\ H_{\max} \end{array}\right]. \tag{6.17}$$

Some are empirical constraints restricting object size in the image in (6.18) and the view angle in (6.19)

$$S_{\min} \leq g_2\left(\theta\right) = \max_j \left\{\frac{K_{12}\left(R\left(\theta\right) X_j + T\left(\theta\right)\right)}{K_3\left(R\left(\theta\right) X_j + T\left(\theta\right)\right)}\right\} - \min_j \left\{\frac{K_{12}\left(R\left(\theta\right) X_j + T\left(\theta\right)\right)}{K_3\left(R\left(\theta\right) X_j + T\left(\theta\right)\right)}\right\} \leq S_{\max}, \tag{6.18}$$

$$\left|g_3\left(\theta\right)\right| \leq \alpha_{\max}, \tag{6.19}$$

where $g_3(\cdot)$ is the function calculating the view angle and $\alpha_max = 60deg$ is the maximum view angle. Size limit of the object $[S_{min}, S_{max}]$ is set to be $[300, 2000]$ For the robot system, there is also constraint for collision as (6.20).

$$g_{\text{collide}}(\theta) \leq d_{\max}, \tag{6.20}$$

where the function $g_{collide}(\cdot)$ calculated distances of robot arms to the boundary of the workspace. As a result, the optimal experimental design of measurement poses is a non-linear constrained optimization (6.21

$$\begin{aligned} \min_{\theta} & \psi\left[P(n)\right] = \min_{\theta} \psi \left[\left(\sum_{k=1}^{n} C^T(k,\theta) V(k)^{-1} C(k,\theta) \right)^{-1} \right] \\ \text{s.t. } & [0,0]^T \leq g_1(\theta) \leq [W_{\max}, H_{\max}]^T \\ & S_{\min} \leq g_2(\theta) \leq S_{\max} \\ & -\alpha_{\max} \leq g_3(\theta) \leq \alpha_{\max} \\ & g_{\text{collide}}(\theta) \leq d_{\max}. \end{aligned} \tag{6.21}$$

Choosing the estimation variable $x = [\delta R_s, \delta T_s, \delta T_c]$, optimization methods including gradient-based sequential quadratic programming (SQP) and sampling-based genetic programming are used for measurement poses design. However, it turns out the gradient-based method does not converge to a reasonable result. Only the sampling-based method gives performance improvement. The sampling-based optimization, however, requires optimizing the whole set of way points of the trajectory simultaneously which results in a very high dimension of search space. In terms of optimality criterion, D-optimality, also called O_1 criterion is considered. First we derive the dynamic programming formulation in (6.22), attempting to decouple the design of way points. J_k^0 is the optimal cost to go function using the D-optimality

$$\begin{aligned} J_k^0\left(P(k-1)\right) :&= \log\left(|M_n|\right) - \log\left(|M_{k-1}|\right) \\ &= \max_{C_k,\cdots,C_n} \left\{ \log\left|1 + {H_k}^T P(k-1) H_k\right| \right\} \\ &= \max_{C_k} \left\{ J_{k+1}^0\left(P(k)\right) + \log\left|1 + C(k)^T V(k)^{-\frac{1}{2}} P(k-1) V(k)^{-\frac{1}{2}} C(k)\right| \right\}, \end{aligned} \tag{6.22}$$

where M_n and H_k are defined as

$$\begin{aligned} M_n :&= \sum_{k=1}^{n} C(k)^T V^{-1}(k) C(k), \\ H_k :&= \left[C(k)^T V(k)^{-\frac{1}{2}}, C(k+1)^T V(k+1)^{-\frac{1}{2}}, \cdots C(n)^T V(n)^{-\frac{1}{2}} \right]. \end{aligned} \tag{6.23}$$

Unfortunately, there is no closed form solution of J_k^0, but using the property of imaging system and decomposition of determinant, the search space can be reduced in the image

plane. Further decoupling the optimization by only optimizing one step each time on

$$\log \left| 1 + C(k)^T V(k)^{-\frac{1}{2}} P\left(k-1\right) V(k)^{-\frac{1}{2}} C\left(k\right) \right|$$

gives the one-step horizon planner design. The resulting trajectory is not the optimal solution of the dynamic programming problem but it is found to have good performance in practice. One similar work with one-step horizon design is presented in [56] but it uses the gradient-based method with heuristic initializations. The detailed design steps are given below:

1. Given current posterior covariance P(k-1), formulate one-step horizon optimization $\max_{C(k)} \log \left| 1 + C(k)^T V(k)^{-\frac{1}{2}} P\left(k-1\right) V(k)^{-\frac{1}{2}} C\left(k\right) \right|$.
2. Reduce search dimension in image plane by placing the position of object in the next view point at the corner of the image.
3. Linear search for the optimal yaw angle of the camera.
4. Given object position in image and view angle of camera, sample on the potential end effector position of the robot for optimal solution.
5. Use the inverse kinematics for solving the joint angle and generating the next observation, then update the covariance matrix $P(k)$.

The proof optimality of principles 2, 3, 4 are given in the following section.

Optimality and Comparison to Other Approaches

This section aims to first explain the principles of the design introduced above. Second, it justifies the optimality of the one-step horizon design under unconstrained cases. For the constrained cases, it is compared to the OEM problem. Several assumptions are necessary for the principle to be optimal. First, there is no process noise in the Kalman Filter model. Second, the feasible design space of $C(k)$ is a subset of an elliptic surface. When the design space is an elliptic surface, it can be regularized to a sphere so that all potential vectors of $C(k)$ are of the same norm. Without loss of generality, we assume that we are designing $C(k)$ on a subset of the sphere. Third, joint zeroing error should not be included in the estimation together with other terms. The second and third terms can be ignored when it is not necessary to further reduce the search space dimension of the one-step horizon design. In this case, steps 2, 3, 4 are no longer optimal but the efficiency of one-step horizon design is improved. For the one-step horizon design, at step k, it is to solve $C(k)$ as a function of joint angles θ to maximize

$$\log \left| 1 + C(k)^T V(k)^{-\frac{1}{2}} P\left(k-1\right) V(k)^{-\frac{1}{2}} C\left(k\right) \right| .$$

The optimal value is achieved when $C(k)$ is closest to the smallest eigenvector c_k of $\sqrt{V(k)}P(k-1)\sqrt{V(k)}$. Note

$$\begin{aligned} C \cdot c_k &= \frac{-\widehat{a} \cdot (K_{12} - x(\theta) K_3) Q_1 \left[\begin{array}{cccc} R(\theta)[X] & R(\theta) & I & J(\theta) + \frac{dRX}{d\theta} \end{array} \right]}{K_3 Q_1 (R(\theta) X + T(\theta))} \cdot c_k \\ &= (\widehat{a} \cdot \mathbf{f}_{12}(\theta) \cdot c_k - (\widehat{a} \cdot x(\theta)) \mathbf{f}_3(\theta) \cdot c_k) \\ &= f_0(\theta) - (\widehat{a} \cdot x(\theta)) f_3(\theta), \end{aligned} \tag{6.24}$$

where f_0, f_3 are scalar functions of θ. $\max_\theta C(k) \cdot c_k$ is not decoupled when third assumption is not satisfied due to the existence of $J(\theta)$ the Jacobian matrix. When joint zeroing error $\delta\theta$ is not included,

$$\begin{aligned} C \cdot c_k &= \frac{-\widehat{a} \cdot (K_{12} - x(\theta) K_3) Q_1 \left[\begin{array}{ccc} R(\theta)[X] & R(\theta) & I \end{array} \right]}{K_3 Q_1 (R(\theta) X + T(\theta))} \cdot c_k \\ &= (\widehat{a} \cdot \mathbf{f}_{12}(R(\theta)) \cdot c_k - (\widehat{a} \cdot x(\theta)) \mathbf{f}_3(R(\theta)) \cdot c_k) \\ &= f_0(R(\theta)) - (\widehat{a} \cdot x(\theta)) f_3(R(\theta)). \end{aligned} \tag{6.25}$$

In this case f_0, f_3 are only functions of the rotation matrix R of the camera. Though x, R are both functions of θ, the position x of the object in the image plane can still vary in a certain area of the image plane when R is fixed. Now $f_0 - (\hat{a} \cdot x) f_3$ is easily maximized by following the direction of $\hat{a}$ on the boundary when the search region of x is convex in image. Generally, it will result in the corner of the image.

When x is derived, $R(\theta)$ can be chosen separately. The wafer handling robot only has rotation freedom at the yaw angle; hence a line search can be performed on the $C(k)\dot{c}_k$ for maximum value. If there are more than one dimension of freedom for the robot, R can be optimized using [144].

6.4 Experimental Results

The experiment is done with a wafer handling robot and a manually made chamber opening in the lab. The robot holds a wafer on the upper blade and a camera is mounted on the blade. The intrinsic matrix of the camera is already calibrated ahead. The communication loop used in the experiment is shown in Fig. 6.2. The image captured by the camera is received and processed by the host PC. All the auto-teaching related calibration is done in the host PC. The estimated object position, expressed as offsets to the nominal position, is fed to the target PC at 20Hz frequency using UDP communication. The host PC also receives the encoder readings at the time when the image is captured. All the motor-level control is done in the host PC which sends control commands to the robot driver at 1kHz frequency.

To avoid the effect of vibration and camera delay, the robot is commanded to move step by step. Images are taken when the robot is fully stabilized. The robot is moved to a random initial pose at each trial. During each trial, the robot

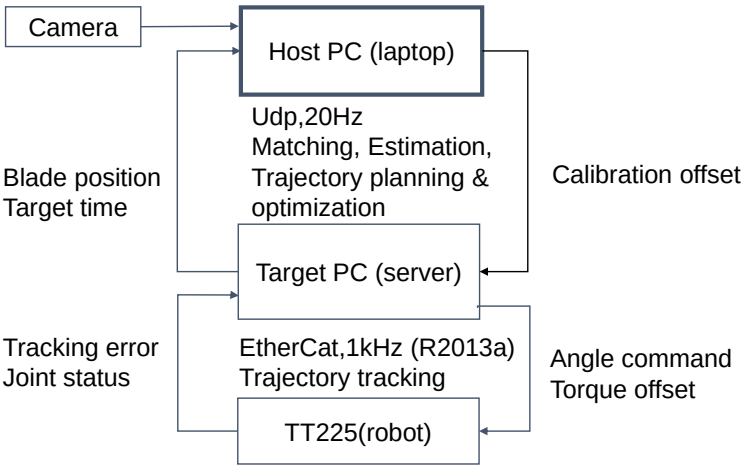


Figure 6.2: Communication loop between robot controller, camera and host PC.

1. detects and estimates the shape
2. designs the measurement poses for optimal calibration
3. updates the error terms using the Kalman filter
4. updates the target position
5. gets next pose of measurement from trajectory planner and updates the pose designed in 2) using the updated target position

The number of measurements, or the number of steps of the movement, are fixed because the trajectory planner is an exact design, which means the optimality refers to a fixed number of measurements. However, the set of measurements can be extended by setting the current pose and estimated covariance as initial conditions and conducting another optimal design with the initial condition. 10 trials are executed with different initial poses to get the statistics of the auto-teaching result. The convergence of the calibration errors are shown in Fig. 6.3. The coupled estimation is the sum of camera translation and center translation error. When the blade approaches the opening and the orientation does not change in the final steps, the contribution of center and camera translation cannot be decoupled. The coupled estimation converges means that the opening calculated from the calibration result coincides with the detected opening in terms of the image plane.

The statistics of the final step is calculated and shown in Table 6.4. Error terms in the last step of the 10 trials are extracted and statistics of the 10 trials is expressed as

$$\pm\text{standard deviation (peak deviation)}$$

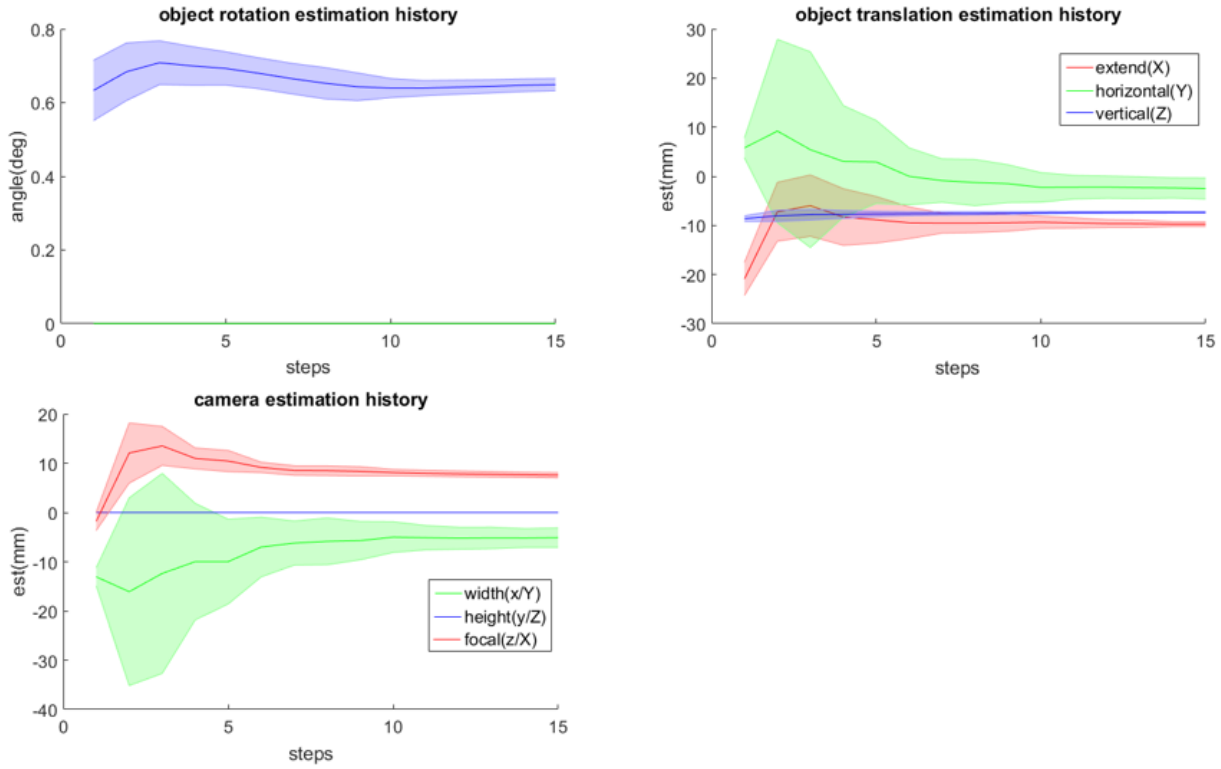


Figure 6.3: Convergence of estimated rotation and translation errors.

It is observed that the Z axis (vertical or height) has a much smaller standard deviation than the X and Y axis. It is mainly because there are two very long straight-line features in the shape of the chamber opening. It makes the eigenvalue along the Z axis the highest and thus making it have the highest accuracy. The optimality criterions (observability indices) of the proposed active auto-teaching method is compared with the auto-teaching experiment with an adaptive controller in Fig. 6.4. The shaded area represents the standard deviation of observability indices in 10 trials. It can be seen that the optimality of proposed methods is always higher than the default one without active calibration.

6.5 Chapter Summary

This chapter introduces active auto-teaching which closes the loop of calibration process of the hand-eye system. An optimal measurement pose planner is proposed to plan waypoints of the trajectory for auto-teaching using the feedback from the calibration module. The OEM-based planner designs robot poses that maximize the eigenvalues of the Fisher information matrix for calibration, which minimizes the posterior uncertainty of estimated variables.

Table 6.4: Statistics of the final estimation in 10 trials.

Translation error		X(extend)	Y(horizontal)	Z(vertical)
Opening	w/o active calibration	±33(90)	±25(65)	±0.2(0.5)
	proposed	±0.5(0.9)	±2.0(4.0)	±0.1(0.3)
Camera	w/o active calibration	±33(86)	±26(66)	±0.2(0.5)
	proposed	±0.4(0.6)	±1.9(3.4)	-
Coupled	w/o active calibration	±0.9(1.2)	±1.1(0.7)	±0.4(0.5)
	proposed	±0.5(1.0)	±0.3(0.6)	±0.1(0.3)

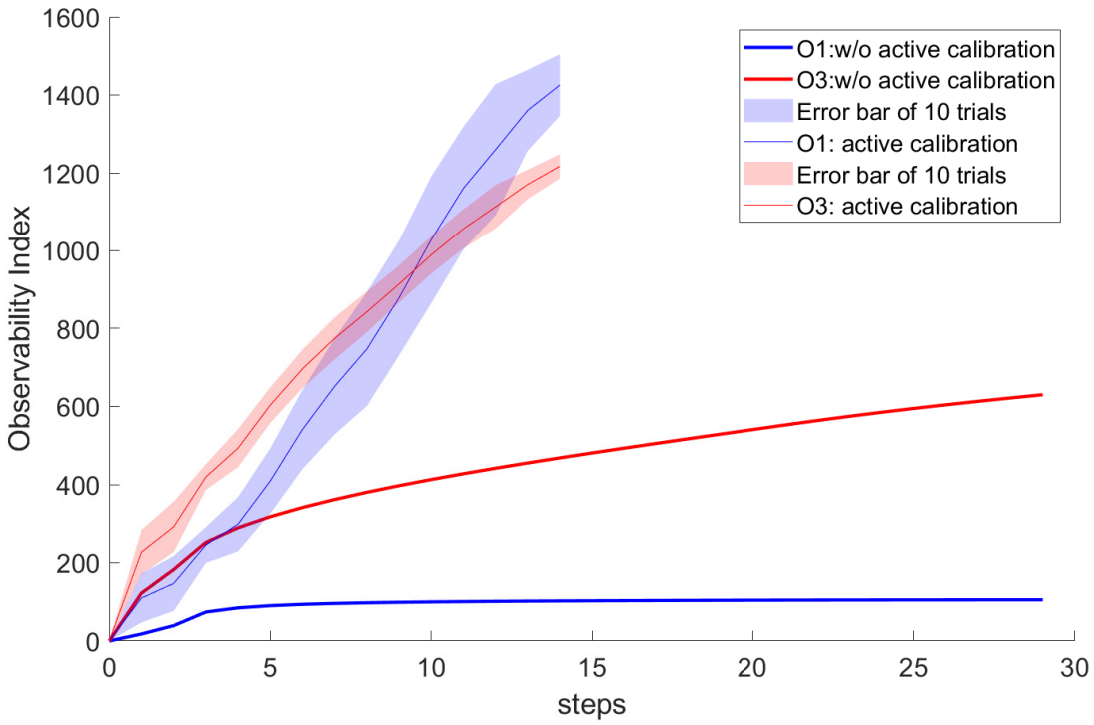


Figure 6.4: Comparison of observability indices between optimal designed trajectories and adaptive control trajectories generated by the previous auto-teaching experiment.

Experimental results in the workspace with chamber opening show that active auto-teaching improves both the accuracy and robustness of the estimated hand-eye system parameters and object locations.

Part II

Autonomous Driving System

Chapter 7

Deep 3D Detection with Camera-LiDAR Sensor Fusion

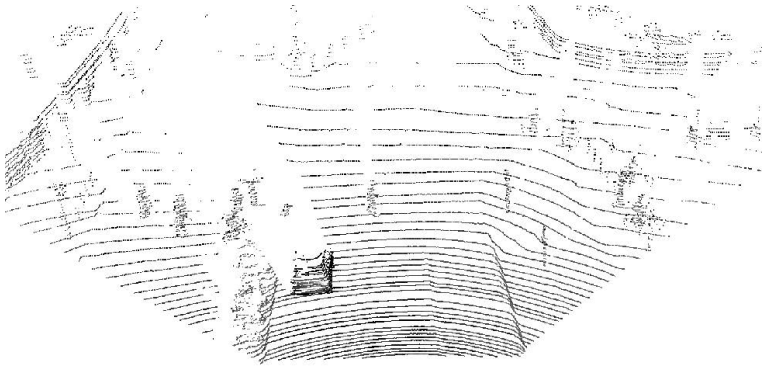
7.1 Introduction

The camera suite on the driver-less car generates 2D images with different view angles. The most frequently used is the front camera. The 3D LiDAR is a laser scanner and it returns sparse 3D point clouds with accurate distance information. An example of the returned data of these two types of sensors is shown in Fig. 7.1. 2D images from the camera suite provide rich texture descriptions of the surrounding, while accurate depth is hard to obtain. On the other hand, 3D point clouds from LiDAR can provide accurate depth and reflection intensity, but the resolution is comparatively low. Therefore, 2D images and 3D point clouds are potentially supplementary to each other to accomplish accurate and robust perception, which is a prerequisite for autonomous driving.

Recent works on camera-LiDAR fusion can be categorized into two main frameworks as illustrated in Fig. 7.2. Serial fusion in Fig. 7.2(a) processes the camera data and LiDAR data sequentially such as F-PointNet [146], RoarNet [147] and others [148, 149, 150]. This structure does not conduct interaction between sensors and it extracts different detection results from the two sensors due to their unique property. The rich texture information of images provided by the camera can recognize objects effectively. Therefore this framework first generates potential 2D detections from camera data only, and then refine them with the 3D LiDAR data and network. The image detection module does not receive information from LiDAR. Some earlier work [151, 152, 153] switch the sequence and process LiDAR data first. The serial fusion structure modularizes the design and tuning of algorithms for the camera and LiDAR, but limits the upper bound of the performance because it lacks deep fusion of multi-modal information and usually disables the joint training of networks.

Parallel fusion in Fig. 7.2(b) uses one big network for processing both camera and LiDAR data. Famous network architectures of this category include MV3D [154], AVOD [155], MMF [156] and 3D-CVF [157]. The parallel fusion integrates multi-modal information inside

(a) Front-view image from camera suite.

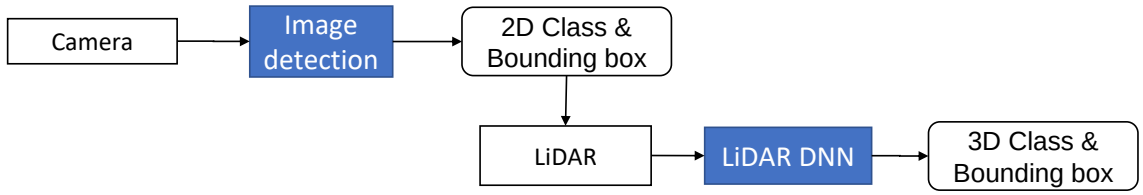


(b) Point-cloud from 64-line 3D LiDAR

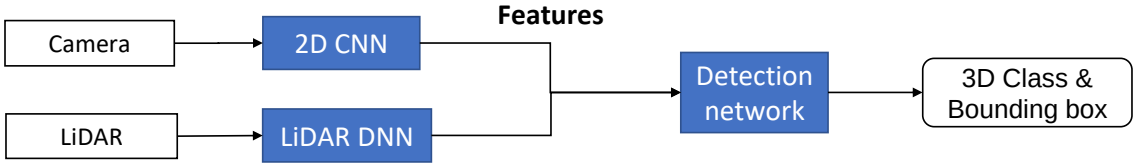
Figure 7.1: Visualization of data returned by the camera and LiDAR [145].

the network and uses joint training to make sure that the detector is optimized for taking both sensors into account when producing detection outputs. The challenge of parallel fusion lies in how to design the fusion mechanism because camera and LiDAR data are of different spaces and have quite different representations. Unlike serial fusion, the performance and efficiency of parallel fusion mainly depends on the design of the fusion mechanism. Earlier work [158, 159] implement simple fusion by transforming the raw data between sensors. Later work, including our proposed method [160], implement more complex transformation at feature level in the middle of the network. However, finding the most effective fusion mechanism is still an open question and people are looking for in-depth understanding of multi-modal fusion.

In this chapter, we propose two fusion methods using the correspondence of point cloud and pixels for serial and parallel camera-LiDAR fusion in deep convolutional neural networks (CNN). The proposed methods are able to conduct fusion in arbitrary layers of the network. They are very efficient and add little overhead to the inference time. The proposed Sparse Non-homogeneous Pooling Layer (SHPL) for parallel fusion is independent of the CNNs, which means it can be applied to any state-of-the-art single sensor networks and integrate them to construct a fusion network. This important property reduces the effort on designing the overall architecture of the network and takes the advantage of the latest development in



(a) Serial fusion.



(b) Parallel fusion.

Figure 7.2: Illustration of two camera-LiDAR fusion frameworks.

single-sensor detection.

The chapter is organized as follows. In Section 7.2, detection networks with single sensor input are reviewed which are used as the backbone of our sensor fusion networks. In Section 7.3, the fusion networks are reviewed in detail to provide an overall knowledge of the most recent development in this area. In Section 7.4, a serial fusion structure is proposed by segmenting the depth information from LiDAR data with a camera proposal network. In Section 7.5, the innovative sparse non-homogeneous pooling layer (SHPL) is proposed for general parallel fusion between camera and LiDAR detection networks. In Section 7.6, several camera-LiDAR fusion networks are constructed with SHPLs and the performance evaluated on the benchmark dataset for autonomous driving.[1]

7.2 Detection Networks with Single Sensor Input for Autonomous Driving

Deep neural networks (DNNs) based on a single sensor have been developed in many other areas. Networks for images are very successful and their extensions to LiDAR are growing rapidly. Their CNN backbones serve as good candidates for the data processing units of a fusion-based network.

The camera-based detection receives the most attention from researchers. The competition is very intense on KITTI dataset [161] with hundreds of proposed structures. Most well-performed camera-based networks including MS-CNN [162], RRC [163] and modified Faster-RCNN [164] only produce high quality bounding boxes in 2D front view images. There

[1]This work includes materials from the author's previously published paper [160].

are surprisingly some single-camera-based networks capable of 3D detection. By adding the prior dimension knowledge of cars and cyclists, the earlier works such as 3DVP [153] and Mono3D are able to produce 3D bounding boxes. The more recent works such as Monocular 3D [165], Deep3DBox [166] and DeepMANTA [167] produce even more accurate 3D boxes from monocular vision than networks using stereo vision such as 3DOP [168].

Unfortunately, the nature of the sensor limits the upper bound of the 3D detection accuracy of camera-based networks. As the development of 3D convolution on unstructured point cloud data accelerates after 2017, it is very hard for camera-based networks to catch the performance of LiDAR-based networks. Despite the new development of providing more semantic information and more precise segmentation in camera-based networks, only few camera-based 3D detection networks are published nowadays such as Pseudo-LiDAR [169] and MonoPair [170]. Meanwhile, they are not even comparable to other LiDAR-based networks.

LiDAR-based detection networks has been explored extensively in the last three years, and their architecture has been changing drastically. The most intuitive thought of extending the successful 2D-CNN from images to the LiDAR point cloud is to use 3D-CNN with voxel representation. Unfortunately, the high computational complexity in the large outdoor scene makes it intractable. LiDAR data collected in autonomous driving scenarios has its own inherent sparse property. Vote3Deep [145] conducts sparse 3D convolution to reduce computational load, but it does not apply to GPUs with parallel acceleration. VoxelNet [171] implements 3D convolution on voxel representation that is coarsely divided in height. It achieves highest scores in 3D detection with acceptable speed in 2017 and has become the symbol of LiDAR-based network with voxel representation. The main drawbacks of this category are the loss of information due to voxelization and low speed. Its successor SECOND [172] proposed sparse 3D GPU-convolution to increase the efficiency. PointPillar [173] reduces to 2D convolution after a 3D feature encoder for faster speed. Center3DNet [174] uses the anchor-free technology to reduce the number of 3D proposals.

Another main stream is to use multilayer perceptron (MLP) with point representation and the most typical work is PointNet [175]. Point representation preserves the unstructured data format from the LiDAR sensor which minimizes the information loss. As a result, available operators on the point representation is quite limited and not suitable for large-scale scenarios in autonomous driving. PointNet++ [176] groups and separates the point cloud by only processing the neighborhood but calculating the neighborhood topology is expensive which slows down the method. More recent networks, including 3DSSD [177], SERCNN [178] and STD [179], focus on reducing the complexity with various new point-based structures.

VeloFCN [180] and the SqueezeSeg [181, 182, 183] series project point cloud to the front view with cells gridded by LiDAR rotation and then applies normal 2D CNN for classification and segmentation. This architecture is simple and very fast in inference time, which is most suitable for on-line implementation on the vehicle. However, the front view representation of point clouds shares the same multi-scale problem as the camera, because the sizes of objects change as distance varies. Therefore, its performance is not as good as the other

networks with true 3D representation. It is worth noticing that all these types of networks keep generating higher detection accuracy as a consequence of the development of the general deep learning area.

Many new categories of architectures emerged in the last year, producing higher accuracy and shorter inference time. Hybrid methods with both voxel and point representation, including Point R-CNN [184], PV-RCNN [185, 186], SA-SSD [187] and HVNet [188], achieves the highest accuracy in current benchmark. RangeRCNN [189] integrates the point representation and front view projection to leverage the speed and accuracy. New convolution layer called graph convolutional network (GCN) is also being explored in PointRGCN [190] and Point-GNN [191].

7.3 Detection Networks with Camera-LiDAR Fusion

The network based on camera and LiDAR fusion, especially for pedestrian detection, has not been sufficiently investigated when our work is proposed. Before our proposed work, fusion networks were slow and limited. The fusion must either happen at the early stage of raw data level or late stage of object level shown by Fig. 7.3. No network is able to perform fusion at the middle of its backbone. Our proposed work enables using the fast one-stage detection structure in fusion-based networks to increase speed. Moreover, it is a new middle-stage fusion highlighted in Fig. 7.3.

Prior to our proposed work, some works [158, 152] apply early fusion by projecting point clouds to the image plane and augment the image channels after upsampling. Such structure fuses camera and LiDAR data through the whole network but only on the image plane. It means that the accurate 3D information of LiDAR point clouds is almost lost. The localization banks on that the regression can magically retrieve the 3D measurement of point cloud, which is not the case according to their low 3D detection score. Pose-RCNN [151] adapts the Fast-RCNN structure where the region proposal is done by classic selective search in LiDAR voxel representation. The fusion structure does not feed LiDAR data into the deep convolutional network which means the classification relies merely on camera data. The state-of-the-art MV3D network applies a region proposal network (RPN) on the point cloud projected to the bird's eye view plane. It preserves the 3D measurement in the region proposal stage. Then it uses the Faster-RCNN and Deep-fused Net [192] structure in the second stage. Note that the region proposal stage only takes the bird's eye view LiDAR data. The quality of proposals is limited by using a single sensor. The speed is also limited because the fusion must happen at the second stage.

Right at the same time and after when we proposed our method, a lot of middle-stage parallel fusion methods were also introduced by other researchers. MMF [156] and ContFuse [156] by Uber share very similar ideas in terms of grid-based feature fusion layer. PointFusion [150] introduces point-wise feature fusion. However, the performance of middle-stage parallel fusion has been increasing slowly in the last few years while all other fusion methods have been iterating quickly over time. This is because the middle-stage method uses

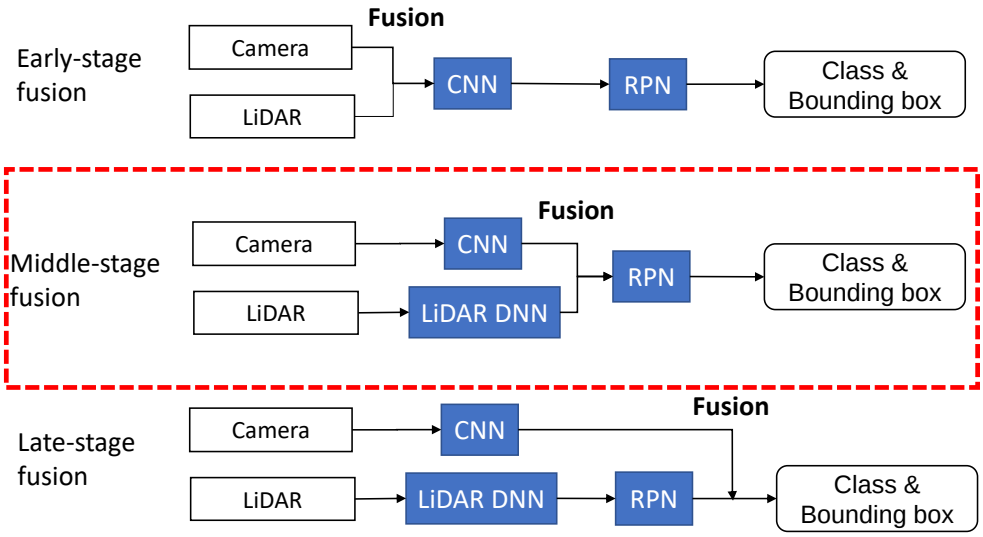


Figure 7.3: Three different structures of the parallel fusion method.

features at the middle of the network which are the least controllable and interpretable. The detection system becomes a huge black box which is hard to probe and tune despite its theoretical upper limit of potential. The most recent 3D-CVF [157] uses both the middle-stage and late-stage parallel fusion and adds a refinement network which achieves high ranking in the public benchmark dataset.

The late-stage parallel fusion is similar to serial fusion methods. They share the same intuition of proposal-refinement framework. F-PointNet [146] is most famous for this. In this framework, the potential objects are proposed from one (serial fusion) or both (parallel fusion) sensors, and then the class and pose of these potential objects are checked and refined by the following network using either the other sensor (serial fusion) or both sensors (late-stage parallel fusion). The refinement is mainly dominated by the LiDAR sensor because it provides very precise 3D measurement. RoarNet [147] and F-Conv [149] are other serial fusion networks with state-of-the-art (SOTA) performance when they were published. MLOD [193], PI-RCNN [194] and CLOCs [195] are recent works of the late-stage parallel fusion category where they use both sensors to generate proposals and fuse multi-modal features in each proposal for further refinement. Unlike middle-stage counterparts, proposals have clear physical meaning which means they can be supervised by labels, making the training of the network more modularized and much easier. The proposal-refinement intuition is so successful that it also affects most SOTA LiDAR-detection networks and there are even works dedicated to designing sub-networks merely for refinement like epBRM [148] and Patch-EMP [196].

7.4 Serial Attenion-based Fusion Layer

This section proposes a serial fusion structure which uses camera-based network to help segment LiDAR point clouds. The idea is that the 3D position of objects are accurately measured by the LiDAR point clouds. Since there is the calibration matrix that aligns images with point clouds in the front view, we can extract corresponding point clouds from the segmentation of the camera as shown by Fig. 7.4. However, the existing camera segmentation network requires first down-sampling and then up-sampling which is very slow. In this work, we propose to segment 3D positions from down-sampled features. A non-zero max-min pooling layer is used to keep the extreme 3D locations of the point cloud when doing down-sampling. Finally, the 3D position of the object detected from the camera is interpolated from the down-sampled extreme 3D locations of point clouds.

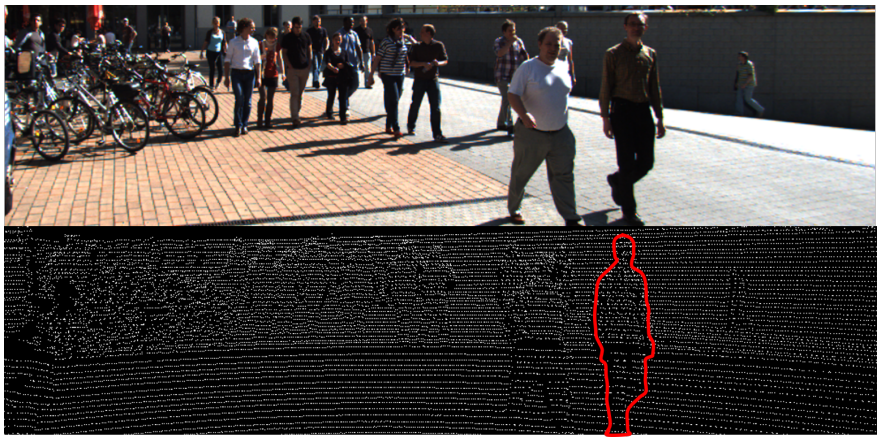


Figure 7.4: Segmented front-view pedestrian point cloud from camera.

As a serial fusion method, this work depends heavily on the camera network to segment and classify objects. The advantage is that we can use a 2D detection network during the training. The LiDAR point cloud is only used for regressing the 3D position of objects while the orientation cannot be well estimated.

Downsampling Extreme Values of LiDAR Point Cloud

The 3D position of an object is bounded by the x, y, z location of its point clouds measured by the LiDAR sensor. In order to keep the upper and lower bounds of point clouds in a down-sampling network in GPU, we need to design a specific layer. The max pooling layer is widely used in convolutional neural networks which returns the extreme value in every $k \times k$ region. The projected LiDAR point cloud on the image plane, however, has undefined points because point clouds are much sparser than camera image pixels. In the projected front-view point-cloud in Fig. 7.4, the write pixels are corresponding LiDAR points and the

black pixels are undefined which are set to the default 0. In order to extract the extreme values while not touching the undefined points, we constructed a non-zero max-min pooling layer as shown by Fig. 7.5. This pooling layer conducts a similar operation as the max pooling layer but ignores the point with value 0. A 2×2 non-zero max-min pooling layer extracts the maximum and minimum value of the $H \times W \times C$ input layer in each channel and outputs a $N/2 \times M/2 \times 2C$ output layer. If all the points are undefined like the red square in Fig. 7.5, then the output will be set to the default 0.

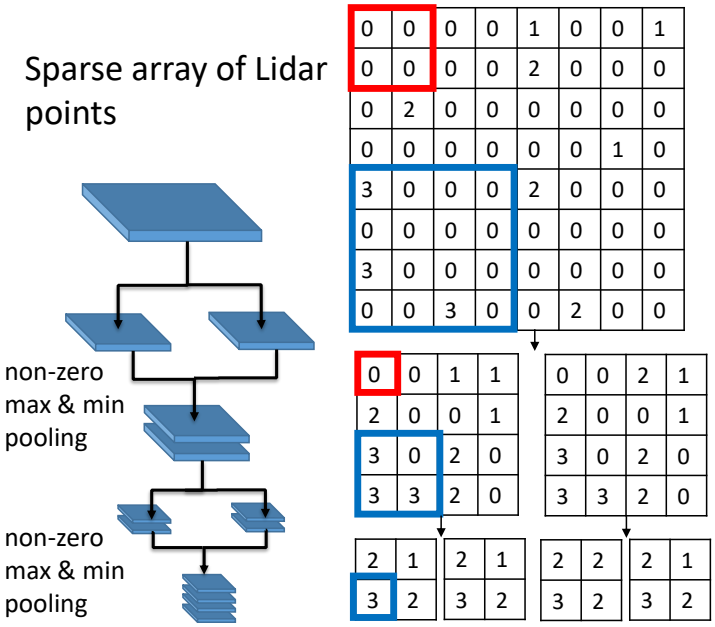


Figure 7.5: A demonstration of the non-zero max-min pooling layer which down-samples the projected front-view LiDAR point cloud. After each operation, the input feature map will be down-sampled into two output feature maps, the max-values on the left and the min-value on the right, respectively.

One may think the proposed layer just keeps the boundaries of point clouds in a local region. However, after multiple down-sampling operations, the non-zero max-min pooling is doing more than just keeping the boundary, because it keeps all combinations of max's and min's. A example is illustrated by Fig. 7.6 where the value of other edges are extracted except the extremes 3 and -9 in the output $1 \times 1 \times 8$ output layer. The segmentation network can then determine whether to take $3, 2, 1, -1, -2$ or -9 as the object center instead of just choosing from 3 and -9.

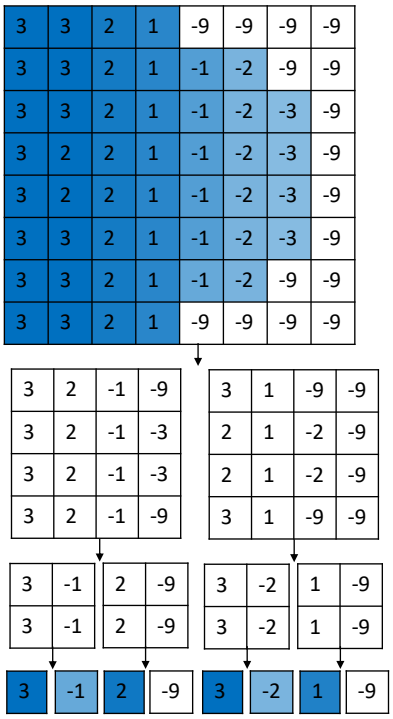


Figure 7.6: An example of a $8 \times 8 \times 1$ input layer after 3 times of pooling. The values simulate two clusters and a background of value -9. The max-min pooling will not only result in the 8×8 extremes 3 and -9, but also the edges $2, 1, -1, -2$ of local clusters.

Soft Attention with 2D Heatmap

The attention mechanism is widely used for improving both the performance and the interpretability of deep neural networks. Here we use the attention layer from the camera-based network to interpolate the 3D positions we down-sampled from LiDAR point clouds. The structure is illustrated below in Fig. 7.7. First, we pick the front-view image features and projected point clouds that are down-sampled by the same number of times, so that they have a one-by-one correspondence. Suppose that they are down-sample by 3 times and the camera feature map is $H \times W \times C$ and the LiDAR point map is $H \times W \times 8$. Then a 2D convolutional layer is applied to produce a $H \times W \times 8$ heap map followed by a softmax layer which normalizes the heat map to a $H \times W \times 8$ probability map that sums up to 1 pixel-wise. Then the down-sampled point cloud dot products with the probability map, so that a position map of $H \times W$ is generated. Each pixel is the estimated position of the object containing this pixel.

The overall network structure with the fusion structure is implemented in Fig. 7.8. The MS-CNN, which is the best 2D detection network when this structure is proposed, is used as the camera-based network. The inputs are front-view RGB image from camera and

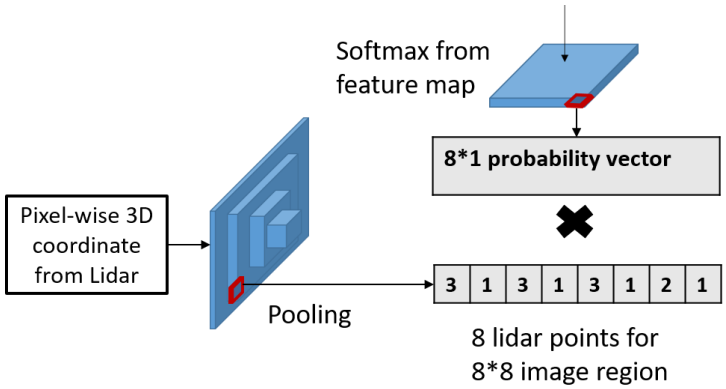


Figure 7.7: The fusion structure that interpolates the attention probability from camera-based network to interpolate LiDAR point clouds.

projected point cloud image from LiDAR which are aligned by the calibration between two sensors. During the training phase, the camera-based network has the same loss function for 2D detection training, and the 3D location loss function is implemented at the end of the LiDAR stream. The 3D loss function is the huber loss between pixel-wise estimated object position and ground truth object position. When doing inference, the layers of MS-CNN after the one that generates the heatmap, which are conv5-1, conv5-2, conv5-3, conv6-1, deconv and RPN, can be discarded to improve the speed.

Experimental Results

The result is evaluated on the KITTI detection dataset. The KITTI dataset has 7481 frames of camera images and LiDAR scans in the training dataset with about 30K objects. It labels all objects with a 3D bounding box. Since this proposed network is unable to predict the 3D orientation, we evaluate the 3D localization performance of detected objects following the standard defined in SubCNN [197]. A detection becomes a true positive (TP) if the predicted 3D location of the object is within a certain distance of the ground truth 3D location. Otherwise it is a false negative (FP). A ground truth label becomes a false negative (FN) if there is no predicted object within its ball of a certain distance. Then the average precision (AP) [198] is used as a metric to measure the performance.

3D localization performance on the KITTI validation set is shown by the precision-recall curve in Fig. 7.9 and Fig. 7.10. We use two thresholds, 1.5m and 2m for calculating the AP. The KITTI dataset has three difficulty levels, namely Easy, Moderate and Hard, so there are three APs for three difficulty levels. The comparison is done with the SubCNN published at the similar time. We achieve higher localization accuracy in Table. 7.1 on pedestrians and cyclists which are harder to predict than cars.

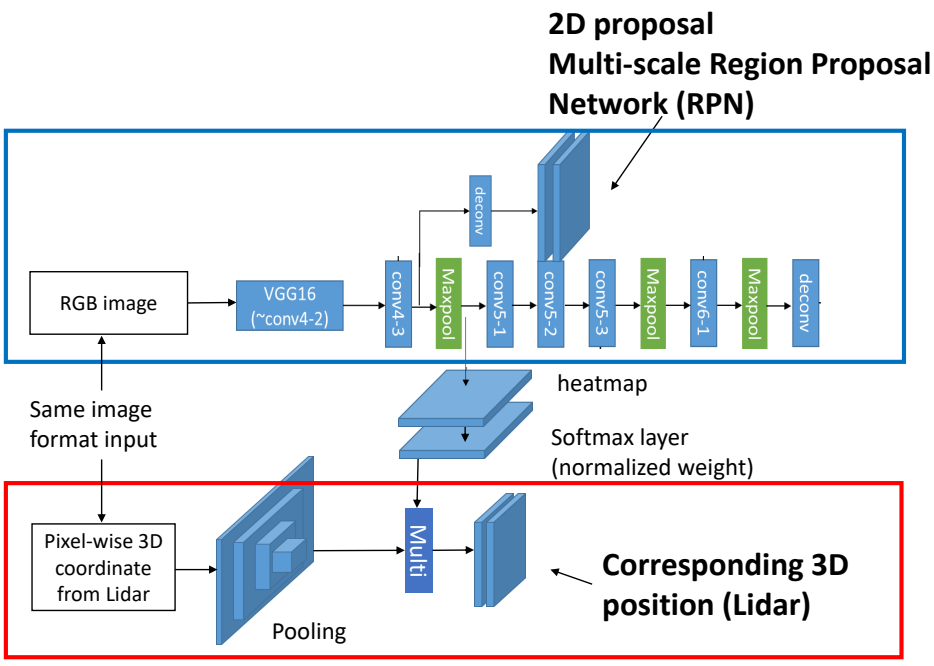
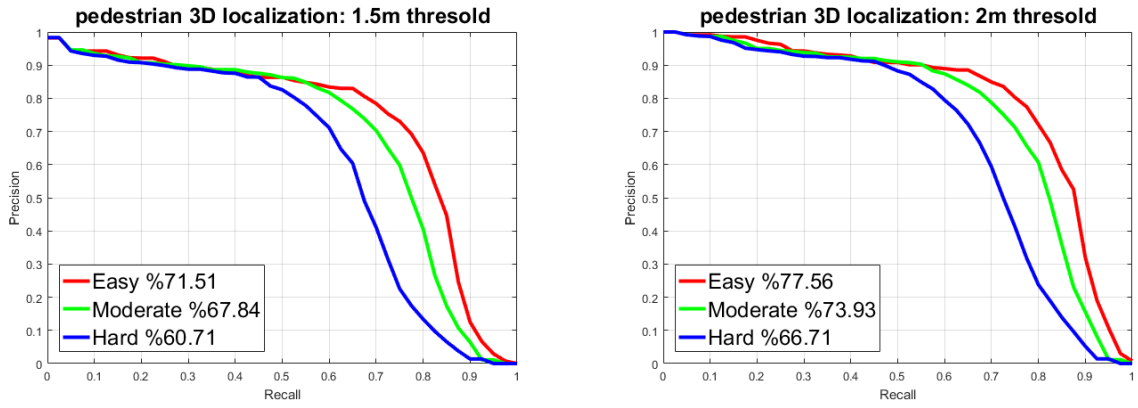
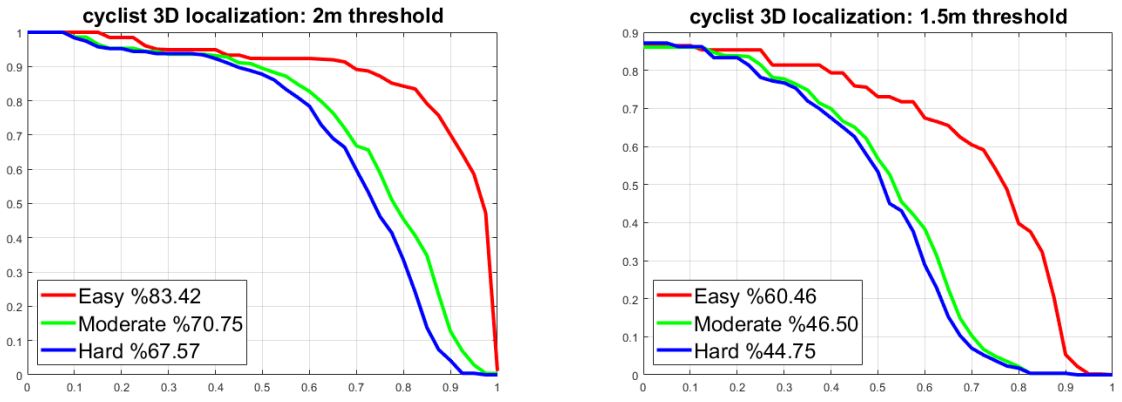


Figure 7.8: The serial soft-attention-based fusion network constructed with MS-CNN [162] and the proposed non-zero max-min pooling.



(a) (b)

Figure 7.9: 3D localization performance on pedestrians with distance thresholds 1.5m and 2m. Results are on the KITTI validation set.



(a) (b)

Figure 7.10: 3D localization performance on cyclists with distance threshold 1.5m and 2m. Results are on the KITTI validation set.

Table 7.1: 3D localization performance compared with other networks on KITTI validation set with distance threshold as 2m.

KITTI validation	Cars			Pedestrians			Cyclists		
Average precision	Easy	Moderate	Hard	Easy	Moderate	Hard	Easy	Moderate	Hard
SubCNN [197]	70.52	56.20	47.03	-	-	-	-	-	-
3DVP [153]	66.56	51.52	42.39	-	-	-	-	-	-
Ours	-	-	-	77.56	73.93	66.71	60.46	46.50	44.75

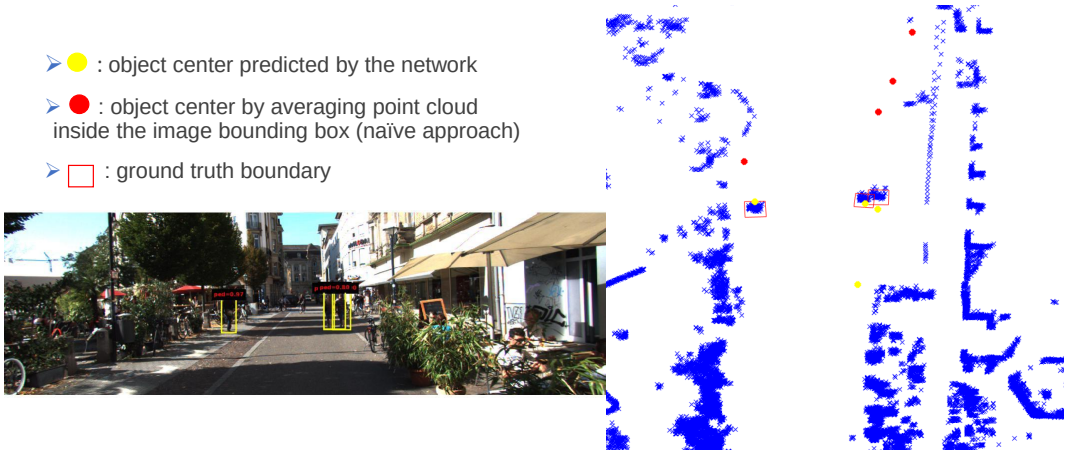
We also show the 2D detection performance compared to MS-CNN used as our backbone network in Table. 7.2. Since we are also adding point cloud information to the MS-CNN, the 2D detection detection performance becomes better as more information is fused. The inference time of our network, with 2D and 3D parts together, is 0.12s while the inference time of original MS-CNN is 0.10s.

Table 7.2: 2D detection performance compared with other networks on KITTI validation set.

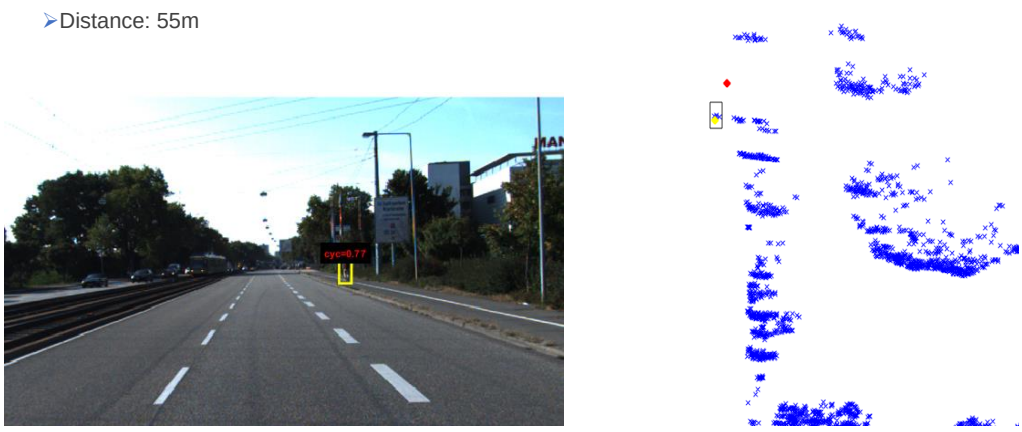
KITTI validation	Cars			Pedestrians			Cyclists		
Average precision	Easy	Moderate	Hard	Easy	Moderate	Hard	Easy	Moderate	Hard
SubCNN [197]	95.77	86.84	74.07	86.43	69.95	64.03	74.92	59.13	55.03
MS-CNN [162]	94.08	89.12	75.54	77.74	72.49	64.43	-	-	-
Ours (uses MS-CNN)	-	-	-	77.56	**73.93**	**66.71**	60.46	46.50	44.75

Some exemplary detection results are shown in Fig. 7.11. The predicted 3D position marked by yellow circle is compared to a naive approach of estimating position from 2D

bounding boxes marked by red. The naive approach calculates the mean of all projected LiDAR points cloud inside the detected 2D bounding box in the image plane, which may include outlier points not belonging to the point. Our predicted 3D position is much better than that of the naive approach with the same predicted 2D bounding boxes. This means our method is able to accept the correct points belonging to the true object and reject the outlier points.



(a) Example 1 of pedestrians.



(b) Example 2 of cyclist.

Figure 7.11: 2D detectoin and 3D localization result on KITTI validation set. Left is the camera image. Right is the bird's eye view LiDAR point cloud.

7.5 The Sparse Non-homogeneous Pooling Layer

We propose a new layer for parallel fusion networks. It aims to transform feature maps between CNNs of different sensors in an efficient way. Its idea originates from the spatial transformer network proposed in [199] which introduces parallel computation of rigid transformation and scaling transformation on feature maps as convolutions. These transformations, however, do not apply to the relationship between LiDAR and camera sensors because their corresponding transformations are not affine. In terms of convolution, we name the convolutions for rigid and scaling transformation as "homogeneous convolution". For example, the general convolution and pooling used in CNNs are homogeneous since

$$f(x,y) = \sum_{u,v} k(x-u, y-v)\, g(u,v). \tag{7.1}$$

The kernel $k(u,v)$ has the same finite support on (u,v), such as $[-1,1]\times[-1,1]$ with the kernel size of 3, which is independent of (x,y). The widely used convolution and pooling layers, including the layer proposed in [199] all belong to this category. Consider the projective transformation between the front view camera image and 3D point clouds

$$\begin{bmatrix} uw \\ vw \\ w \end{bmatrix} = P \begin{bmatrix} x \\ y \\ z \\ 1 \end{bmatrix}, \tag{7.2}$$

where P is the projection matrix derived from camera-LiDAR calibration, (u,v) is the pixel in image and (x,y,z) is the coordinate in 3D. If written as a convolution operation, the equation is then

$$f(x,y,z) = \sum_{u,v} k_{x,y,z}(u,v)\, g(u,v), \tag{7.3}$$

and the support of kernel is $\sup(k_{x,y}) = \left\{(u,v) \,\middle|\, [u,v]^T = w^{-1}P_{12}X, w \in \mathbb{R}^+\right\}$ where $X = [x,y,z,1]^T$ and P_{12} contains the first two rows of P. Both the elements and measure of the support depend on (x,y,z). If the transformation was done as above, the computation would be heavy and non-parallel because a large but various number of $g(u,v)'s$ were involved for each (x,y). The proposed sparse non-homogeneous pooling uses the point cloud to reduce the support region. It also formulates the non-homogeneous convolution and pooling as parallel operations with sparse matrix multiplication so that it can be done in GPU with auto-back-propagation for network training.

Non-homogeneous Pooling as Sparse Matrix Multiplication

The transformation between the front view image feature map and LiDAR feature map can be sparsified by the point cloud. Instead of matching one pixel (u,v) in front view with a full

homogeneous line $(\lambda x, \lambda y, \lambda y)$ in 3D, only (u, v) and (x, y, z) that share the same point in the point cloud are paired. The transformation is still non-homogeneous as the pairing does not guarantee one-to-n (n fixed) mapping, but the computation is sparse as the number of points in point cloud is of only 10^4 scale compared to the number of 3D cells at 10^6 scale. Suppose the front view feature map is of size $H_f \times W_f$ and the 3D feature map is $L_b \times W_b \times H_b$. The LiDAR point cloud is $\{(x_i, y_i, z_i) \,|i = 1, 2, \cdots N\}$, the transformation kernel is sparsified as (7.4)

$$k_{x,y,z}\left(u, v\right) = \delta_{(x,y,z)(x_i,y_i,z_i)} k_{x,y,z}\left(u, v\right) \delta_{(u,v)(u_j,v_j)}, i, j = 1, 2 \cdots, N, \tag{7.4}$$

because only $(u, v), (x, y, z)$ corresponding to the same LiDAR point are transformed. The transformation becomes (7.5)

$$\begin{aligned} f\left(x_i, y_i, z_i\right) &= \sum_{u,v} k_{x_i,y_i,z_i}\left(u, v\right) \delta_{(u,v)(u_j,v_j)} g\left(u, v\right) \\ &= \sum_{j} k_{x_i,y_i,z_i}\left(u_j, v_j\right) g\left(u_j, v_j\right) \\ &= \sum_{j} k_{x_i,y_i,z_i}\left(u_j, v_j\right) \delta_{(x_i,y_i,z_i)(x_j,y_j,z_j)} g\left(u_j, v_j\right), \end{aligned} \tag{7.5}$$

where

$$\delta_{ab} = \begin{cases} 1, a \sim b \\ 0, \text{otherwise} \end{cases}. \tag{7.6}$$

So the transformation kernel, instead of having a small support like convolution, is a sparse $L_b W_b H_b \times H_f W_f$ matrix of $\{(x, y)\} \times \{(u, v)\}$, or of $\mathbb{Z}_N / \sim \times \mathbb{Z}_N$ where $\sim$ is the equality and $[\cdot]$ means round to integer.

$$(x_i, y_i, z_i) \sim (x_j, y_j, z_j) \Leftrightarrow ([x_i], [y_i], [z_i]) = ([x_j], [y_j], [z_j]). \tag{7.7}$$

There are at most N non-zero elements in the sparse matrix regardless of the size of feature maps. The kernel is normalized by the number of points in one cell as formulated in (7.8)

$$k_{x_i,y_i}(u_j, v_j) = \left|\{k \in \{1, 2, \cdots N\} \,|(x_i, y_i) \sim (x_k, y_k)\}\right|^{-1}. \tag{7.8}$$

Note this can be extended to more general interpolation methods like bilinear pooling by associating the LiDAR point with not only the one pixel it projects to, but also its neighbors and normalize the matrix row-wise to have sum 1.

SHPL Implementation as a Network Layer

The sparse matrix multiplication is decomposed into three parts when implemented in the GPU, which corresponds to the following three matrices P, K, Q in (7.9)

$$f(x_i, y_i, z_i) = \sum_{k=1}^{n} Q_{ik} \sum_{l=1}^{C} K_k(l) \sum_{j=1}^{N} P_{kj} g(u_j, v_j), \tag{7.9}$$

and their operations are shown in Fig. 7.12. The P is a gather operation in GPU which extracts the features (cells) in 2D feature map corresponding to each projected point cloud. Assume there are n points in the point cloud and each 2D feature is of D_{2D} dimension. The gather operation stores the extracted features as a dense $n \times 1 \times D_{2D}$ tensor. Moreover, instead of just gathering the only one cell for each point, we can gather its neighborhood with C features for each point which will result in a $n \times C \times D_{2D}$ tensor. Then the K denotes a general convolution operation applied to the dense tensor. This convolution is only conducted on a small subset of features in the feature. Assume the features in 3D feature map are of dimension D_{3D}, then the K can generate a $n \times 1 \times D_{3D}$ tensor. Finally, the Q, which is a scattering operation, redistributes these features to the 3D feature map whose cells correspond to the n LiDAR points.

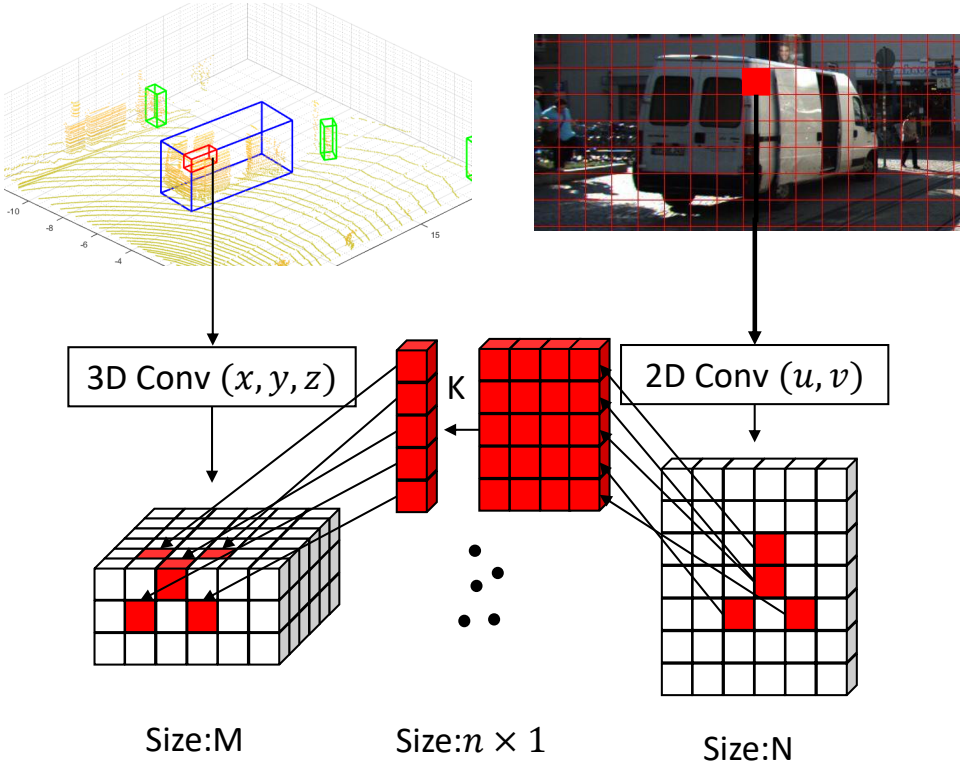


Figure 7.12: The illustration of doing sparse pooling from 2D image feature map to 3D feature map with an example of n point clouds. First the $C \times n$ neighborhood cells in 2D corresponding to n LiDAR points are gathered and stored as a $n \times C \times D_g$ tensor. Then we can do row-wise convolution K to the tensor. Finally the features are scattered to the correponding cells in the 3D feature map.

7.6 Parallel Fusion Detection Networks with SHPL

The fusion structure introduced in Section 7.5 allows one-stage detector because the fusion is done in the first stage across different views, unlike [200] where fusion is done only after RoI pooling in the second stage. In this section we introduce a one-stage detector that takes front view camera image and bird's eye view LiDAR point cloud as input and produces 3D bounding box from bird's eye view without RoI pooling. The detection scheme is adapted from [201] but those of [202, 203] are also compatible.

One-stage Object Detection Network with SHPL

The resolution of image and bird's eye view point cloud captured in the autonomous driving scenario is much larger than the benchmark datasets used to evaluate general detection networks. The test time per frame is important for practical application. One-stage detectors are much faster than two-stage detectors because they do not have the RoI pooling, non-minimum suppression and fully connected operation in the second stage. One-stage detectors directly predict bounding boxes from all the proposals produced by RPN whose number is usually of 100K scale, which means the proposals should be of very high quality to extract the very few true positives. In this section we propose a one-stage detection network with the SHPL for fusion.

The vanilla structure is shown in Fig. 7.13. There are two fully convolutional backbones, namely the image unit and LiDAR unit. The sparse non-homogeneous pooling layer serves as the cross-bridge of two units to exchange information between sensors. The image unit uses the same RPN structure as recent camera-based one-stage detectors. Although the region proposal is not used during the test, an auxiliary loss is still applied on the image CNN so that image features get gradients from the supervision in front view in addition to gradients from the 3D proposal. It serves the similar functionality as the auxiliary loss in [200].

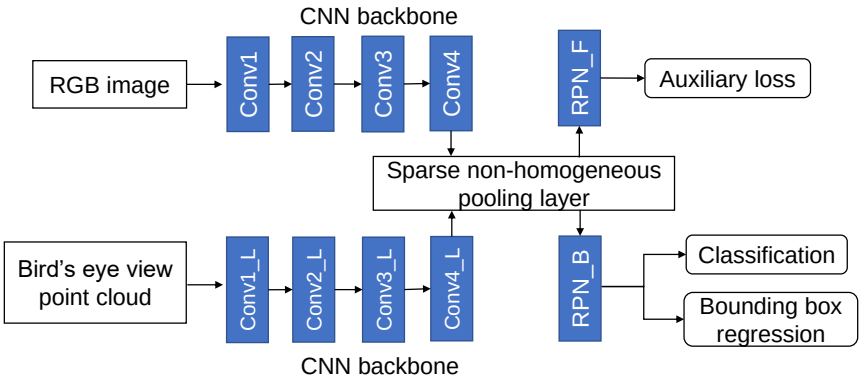


Figure 7.13: The vanilla fusion-based one-stage object detection network.

Since the region proposal is in the bird's eye view, objects of different distances are of the similar size if they are in the same category. The vehicles in KITTI dataset have bounding box size (l, w, h) of (4.0, 1.6, 1.6)m $\pm$ (0.6, 0.1, 0.2)m and pedestrians are of (0.9, 0.6, 1.6)m $\pm$ (0.2, 0.1, 0.2)m in the bird's eye view plane. There is no multi-scale issue hence the multi-scale structure in [200] and [204] are not needed.

One-stage detection networks must be more careful with the class imbalance problem. The class imbalance problem in detectors based on selective search [205] and its solution in DNNs is summarized in [201]. For two-stage detectors [206], the imbalance problem is addressed by RoI pooling as it keeps a positive:negative$\approx$1:3 ratio in the second stage. Actually, other second-stage detectors like [200] also use bootstrapping and weighted class loss in the first stage when the class imbalance is huge for small objects.

Class imbalance problem is because in addition to the prediction of probability, classification also involves binary (or discrete) decision [207]. A utility function or decision cost is imposed to the classification result where many true negatives, whose probabilities are below some threshold, contribute almost no cost. The so-called hard negative mining solutions share the same idea of lifting the loss of hard negatives, because there is a mismatch between the used cross-entropy loss and the decision loss that actually measures the performance. This work uses the focal loss [201] which is a weighted sum of cross-entropy

$$\mathrm{FL}\left(\mathbf{p}, y\right) = \alpha_y (1 - p_y)^{\gamma} \mathrm{CE}\left(\mathbf{p}, y\right), \tag{7.10}$$

where $\mathbf{p}$ is the vector of probability of the sample among all classes. y is the label of the sample. $p_y = \mathbf{p}(y)$ is the probability at class y and α_y is an empirical weight for each class. $\mathrm{CE}(\cdot)$ is the cross-entropy. Focal loss is very efficient and is demonstrated to be effective on image-based one-stage object detectors by [201].

Experiments Using the Proposed One-stage Detector

The network based on sparse non-homogeneous pooling is evaluated on the KITTI dataset which has calibrated camera and LiDAR data and ground truth 3D bounding boxes. The KITTI 3D object and bird's eye view evaluation are used. We focus on the pedestrian category. When the network is published, the maximum average precision (mAP) on KITTI is still very low among all fusion-based networks where average precision (AP) is only about 26%. It is shown that the fusion structure helps a lot on the 3D proposal of pedestrians from bird's eye view since there is enough fused information.

Implementation Details

The vanilla version in Fig. 7.13 uses VGG for both LiDAR and camera. The VGG16 is full VGG pretrained on ImageNet in contrast to the reduced VGG used in [200]. The input is the 1280×384 camera image and the $600 \times 600 \times 9$ gridded LiDAR bird's eye view representation with 0.1m resolution on the ground. The feature map produced by the backbone is downsampled by 4 times in bird's eye view and 8 times in front view. In addition, another

network is constructed with the state-of-the art single sensor networks in Fig. 7.14, called SOTA version. It uses MS-CNN[162] for camera and VoxelNet[171] for LiDAR. The input is the 1280×384 camera image and LiDAR voxels following the same setting as VoxelNet. The feature map produced by the backbone is down-sampled by 2 times in bird's eye view and 8 times in front view. For both kinds, camera inputs are augmented by globally scaling randomly between $[0.95, 1.05]$ and shifting randomly between $[-10, 10]$. LiDAR inputs are augmented following the instruction of VoxelNet. The calibration matrices for sparse pooling are changed accordingly. The network is implemented using TensorFlow and one TITAN X GPU as hardware.

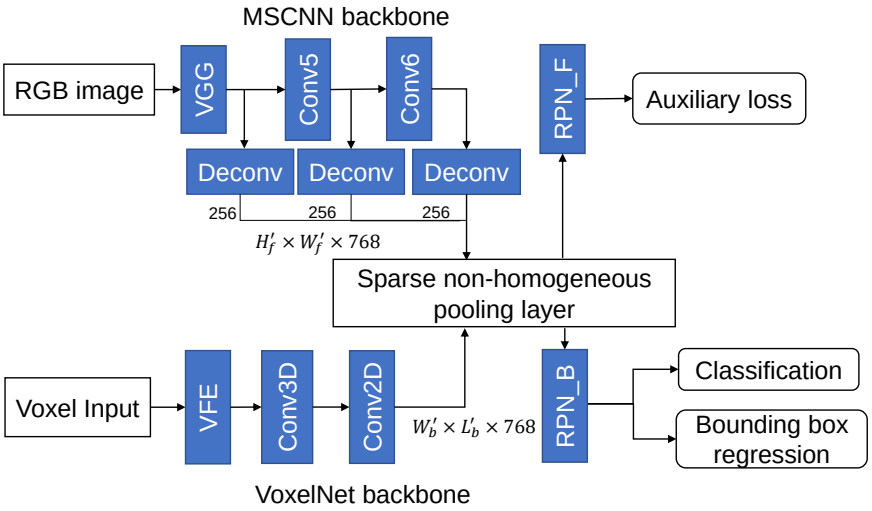


Figure 7.14: The SOTA version fusion-based one-stage detection network with state-of-the-art single-sensor networks.

The focal loss is applied to all anchors to deal with the class imbalance. Different from [201], it is observed that using the focal loss for the non-object class only and cross-entropy for object classes performs better than using focal loss for all classes. Moreover, instead of using biased initial weights to prevent the instability of training in [201], an adaptive weight of losses is used. For non-object class, the loss is formulated in (7.11) as

$$loss_{neg} = (1 - \alpha) \times \mathrm{CE}_{neg} + \alpha \times \mathrm{FL}_{neg}, \tag{7.11}$$

where CE_{neg} is the common cross entropy loss and FL_{neg} is the focal loss on all non-object anchors . $\alpha = 0$ for the first 10% of iterations and $\alpha = \mathrm{recall}_{pos}$ which is the average recall rate of objects updated every 500 iterations by exponential average with a decay rate of 0.998. As the recall rate grows higher during training, the weight of focal loss becomes higher, bringing higher weight on hard negatives. This is similar to that used in [162] where random sampling is used on all negative anchors first and bootstrapping is used later. The same smooth l_1 loss as [208] is used for bounding box regression.

Evaluation Speed

The SHPL is tested to be efficient. The pooling of raw inputs (camera image and bird's eye view LiDAR above) takes 20ms. The pooling of features in vanilla version ($155 \times 46 \times 512$ for camera and $75 \times 75 \times 512$ for LiDAR) takes 14ms. With the efficient sparse non-homogeneous pooling fusion and one-stage detection structure, the test time is 0.11s per frame compared to the 0.7s per frame in MV3D. For the SOTA version, the inference time is longer because there is no available official version of VoxelNet. The implementation is modified from an unofficial publicly available reproduction by Jeasine Ma and does not reach the speed claimed by the paper [171]. The test time consists of the sparse matrix construction, network inference and bounding box post-processing (NMS) but the file I/O is excluded.

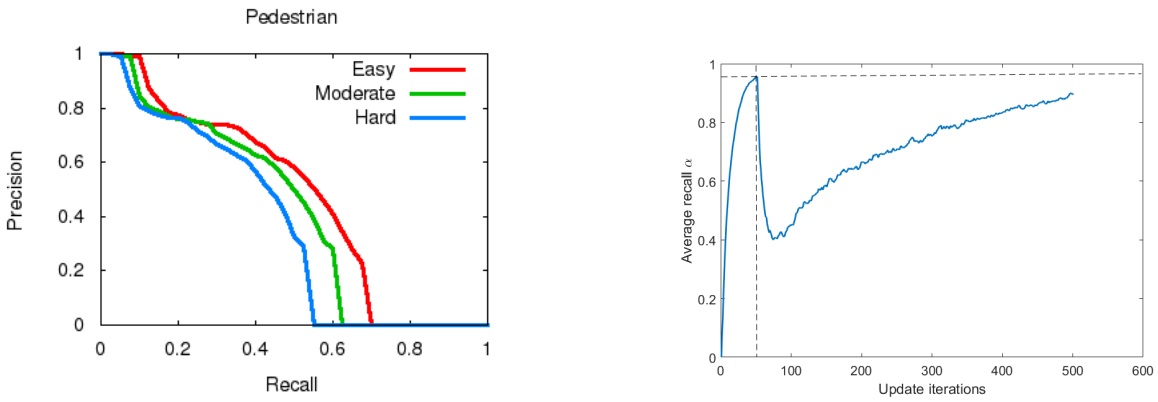
Detection Results and Discussion

To verify the fusion structure is effectively augmenting features in the RPN, the recall and precision of the vanilla version is compared with that of the MV3D. To make a fair comparison, the cross-entropy loss is used in RPN instead of focal loss and IoU threshold is set to 0.5. The MV3D implemented in this paper is unofficial but reproduced according to the paper. Table 7.3 shows, for vehicles, the precisions are similar but for pedestrians, the fusion-based RPN has much higher precision. This is because the pedestrian is hard to distinguish with just LiDAR data, while vehicles are distinct from bird's eye view. The proposals for pedestrians require more information and the fusion structure provides the front view image feature, making proposal quality of pedestrians similar to that of vehicles.

Network	Vehicle		Pedestrian	
Type	Recall	Precision	Recall	Precision
MV3D	99.4	17.3	97.8	4.2
ours	99.2	19.5	96.6	17.3

Table 7.3: RPN quality of RPN on pedestrians and vehicles on the validation set. All difficulty levels are included

The pedestrian detection result is evaluated on the validation set and shown in Fig. 7.15. Due to the limit of hardware, the network is not trained to the same number of epochs as in MV3D and VoxelNet which are around 150 epochs. The network is trained for 30 epochs on the train split with 3712 samples and evaluated on the validation split with 3769 samples. The learning rate is 0.0005 for the first 50% iterations and reduces linearly to 0 for the last 50%. The CNN backbone of LiDAR part is initialized randomly and the camera part loads the pretrained weights from VGG and MS-CNN, for vanilla version and SOTA version respectively.



(a) The precision-recall-curve on the validation set.

(b) The weight of focal loss within 30 epochs.

Figure 7.15: The detection results of SOTA version.

Table 7.4 shows the performance comparison of 3D detection networks presented on the KITTI dataset. The VoxelNet, due to its superior LiDAR input representation, has much higher score than other networks. It is even better than our vanilla version fusion-based network because the vanilla version uses the same hand-crafted LiDAR input as MV3D. Using the SOTA version, we are able to achieve the highest performance on the validation set. Fig. 7.15(b) shows the curve of average recall rate during training. The recall drops when the focal loss is activated and is still increasing at the end of the training process.

Network	Pedestrian		
Name	Easy	Moderate	Hard
3dssd	27.4	24.0	22.4
AVOD	34.4	26.1	24.2
VoxelNet(Official)	46.1	40.7	38.1
Vanilla(Proposed)	34.0	31.4	29.3
VoxelNet(Reproduced)	46.9	38.1	33.9
SOTA(Proposed)	**51.3**	**45.0**	**40.2**

Table 7.4: Comparison with the SOTA on KITTI. Note the last three rows are on the validation set with 30 epochs and the first three rows are on the testing set with 150 epochs.

Two-stage Object Detection Network with SHPL

This section aims to show that the proposed SHPL is very general and can be applied to other state-of-the-art network structures to improve performance. AVOD [155] is a two-stage fusion-based detection network that has the top performance on KITTI dataset. Its structure is shown in Fig. 7.16(a). As we have stated in Section 7.3, it has to fuse features object-wise after the region proposal. No information between sensors is exchanged before proposing potential objects. We introduce SHPL to AVOD and modify its structure as shown in Fig. 7.16(b). The backbone is broken somewhere in the middle and features are transformed to the other backbone using SHPL.

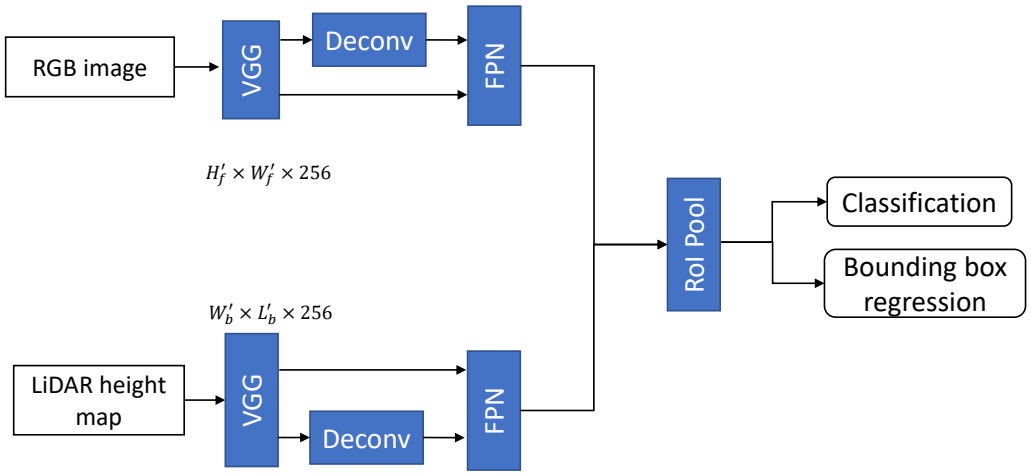
All the other configurations are kept the same as the original AVOD except the fusion with SHPL part. The features after the conv4 layer of VGG are used for fusion. The feature map from the camera network is a $H'_f \times W'_f \times 256$ tensor and the feature map from LiDAR is $W'_b \times L'_b \times 256$. After pooling, each feature map from VGG is concatenated with the feature map transformed from the other sensor, resulting in a feature map of 512 channels.

Experiments Using the Proposed Two-stage Detector

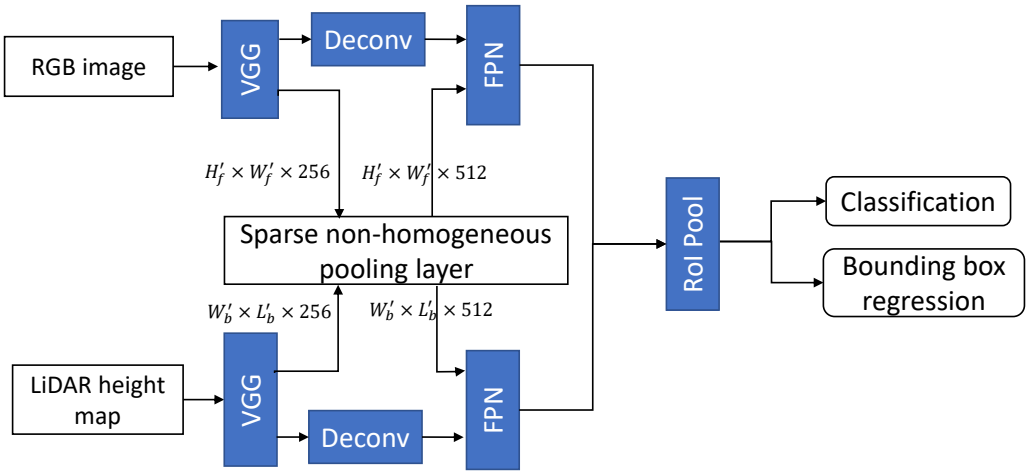
The proposed two-stage detector serves as a very good candidate for ablation study to demonstrate the effectiveness of SHPL. Both the original AVOD and our modified AVOD with SHPL are trained end-to-end on the KITTI training set and evaluated on the validation set detecting pedestrians and cyclists. The average precision of their detection results are shown in Table 7.5. We examine three fusion schemes with SHPL. The first only pools 2D camera features to the LiDAR network and augments the 3D LiDAR feature. The second one only does the reverse way. The third one does the full fusion as shown by Fig. 7.16(b). Since AVOD uses 2D and 3D features equally for the final 3D detection, augmenting either sensor channel increases the performance. The third fusion scheme has the largest improvement for pedestrians. The average inference time of original AVOD is 132ms while the inference time of modified AVOD with SHPL is 136ms. There is only a 4ms overhead added which means our proposed fusion layer is very efficient.

Network	Pedestrian			Cyclist		
Name	Easy	Moderate	Hard	Easy	Moderate	Hard
AVOD (original)	50.7	44.7	38.0	59.0	39.6	33.6
AVOD+SHPL (3D$\leftarrow$2D)	+1.0	+2.5	+3.5	+9.2	+2.7	+8.2
AVOD+SHPL (3D$\rightarrow$2D)	+1.0	+2.0	+3.3	+8.2	+8.3	+6.0
AVOD+SHPL (3D$\leftrightarrow$2D)	+7.6	+7.4	+8.4	+5.3	+0.5	+5.2

Table 7.5: Comparison with the SOTA on KITTI. Note the last three rows are on the validation set with 30 epochs and the first three rows are on the testing set with 150 epochs.



(a) The structure of the original AVOD network which does fusion after the region proposal.



(b) The structure of the modified AVOD network with SHPL which fused whole feature map.

Figure 7.16: Modification of AVOD with SHPL which enables sensor fusion before RPN.

7.7 Chapter Summary

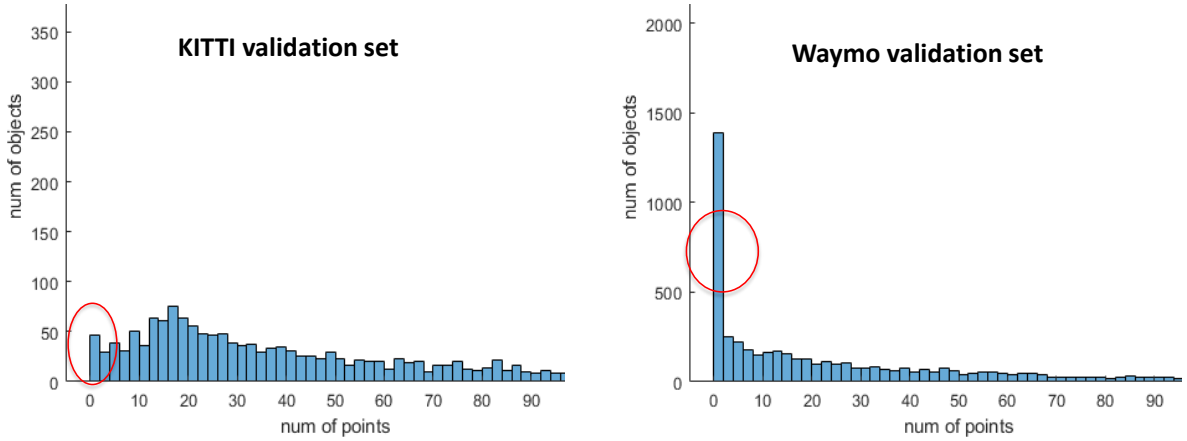
We explored different fusion structures for 3D detection networks. The focus of our methods is to provide an efficient end-to-end fusion layer for various kinds of CNN backbones. The sparse non-homogeneous pooling layer (SHPL), especially, is very general and can fuse different SOTA single-sensor detection networks. It can even be used to enhance other fusion-based detectors with very small overhead. In addition, by adding SHPL to the backbone, we enable the potential of using fusion networks to do other perception tasks like semantic

segmentation. Without our full feature map fusion method, other fusion methods developed for detection are not extendable to other tasks because they require fusion at the object-level.

Chapter 8

Robustness Evaluation of Learning-based Detection

Deep learning based methods rely heavily on the data and human labels. In Chapter 7, labels from the public KITTI dataset are trusted and used with equal weights. However, it is shown in Fig. 8.1 that there are lots of labels with few or even no LiDAR observations, making them bad labels. Training with these labels will confuse the object detector. On the other hand, the probabilistic distribution of detection outputs is essential for the following prediction and planning modulesTo further improve the robustness of learning-based object detectors and to better evaluate the information generated by detectors, we study the spatial distributions of objects and propose a new evaluation metric for learning-based detection methods in this chapter.



(a) KITTI validation set.

(b) Waymo validation set.

Figure 8.1: Histogram of LiDAR observations (number of points clouds) of all objects.

8.1 Introduction

The availability of real-world driving datasets such as KITTI [209] and Waymo [210] is one of the key reasons behind the advancement of object detection algorithms in autonomous driving. However, the data labelling process can be error-prone due to human subjectivity and resource constraints. Ambiguity or uncertainty may also inherently exist in an object label. Think about the 3D object detection task, where annotators need to estimate the object positions only based on the surface information from cameras or LiDARs. Fig 8.2 illustrates several bounding box labels of "Car" objects and their associated LiDAR point clouds in Bird's Eye View (BEV). The areas with dense LiDAR points (typically "L-shape" areas) are easier to be labeled (e.g. object 2), whereas the back side of the object has higher labeling uncertainty due to insufficient observations (e.g. object 3). Ignoring such label uncertainty during training may degrade the generalization capability of an object detector since the model is forced to fit each training data sample equally, even the ones with remarkable noises. Significantly polluted data with noises can also deteriorate the detection performance [211]. Therefore, modelling label uncertainty in a dataset is indispensable for building robust, accurate object detectors for autonomous driving. Previous works have been focused on modelling class label noises in image classification problem [212, 213, 214, 215]. To the best of our knowledge, modelling label uncertainties in object detection problems has not been widely studied, especially for bounding box labels.

Uncertainties should be comprehensively considered, not only for the labels, but also for the evaluation metrics. Intersection over Union (IoU), defined as the geometric overlap ratio between two bounding boxes, is the most common metric to measure localization accuracy in object detection. Based on IoU, several metrics for detection accuracy are proposed, such as Average Precision (AP) [216] and Localization Recall Precision (LRP) [217]. However, those metrics are designed only for deterministic object detection: they can not be used to evaluate probabilistic object detectors [218] which provide additional uncertainty estimation. The Probability-based Detection Quality (PDQ) metric [219] is designed specifically for probabilistic object detection. However, PDQ and the other aforementioned metrics only compare predictions with bounding box labels without considering the uncertainty (or ambiguity) in the labeling process. As a result, existing evaluation metrics may not fully reflect the performance of an object detector.

In this work, we explicitly model the uncertainty of bounding box parameters, which is inherent in labels ("label uncertainty") for object detection datasets with LiDAR point clouds. The label uncertainty is inferred through a generative model of LiDAR points. In this way, we can easily incorporate prior knowledge of sensor observation noises and annotation ambiguity into our model. Then we propose the "spatial distribution" to visualize and represent the label uncertainty in 3D space or the LiDAR Bird's Eye View (BEV). We show that it reflects not only the typical L-shape observations in LiDAR point clouds, but also the quality of bounding box labels in a dataset. Based on the spatial distributions of bounding boxes, we further propose a probabilistic IoU, namely, Jaccard IoU (JIoU), as a new evaluation metric for object detection. The metric treats each bounding box label differently

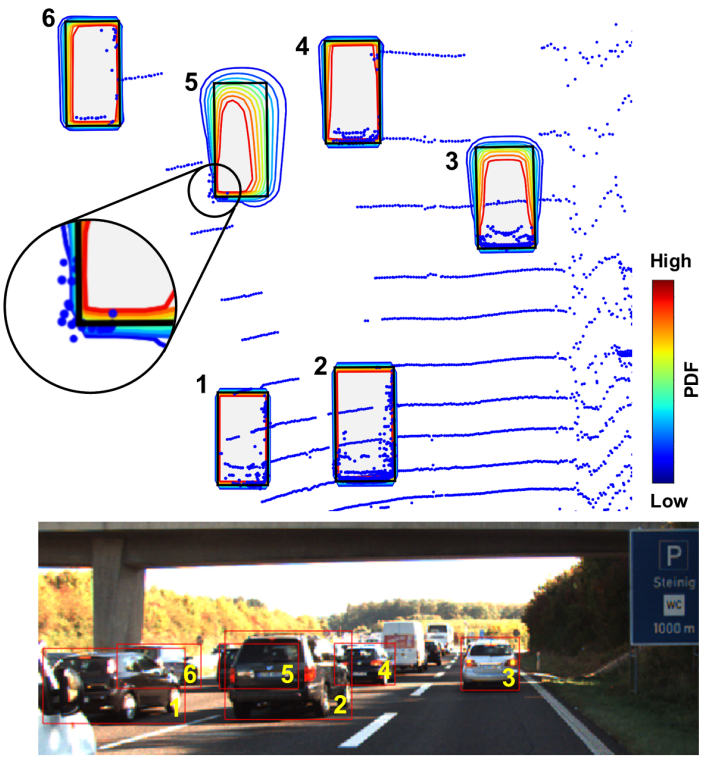


Figure 8.2: A demonstration of our proposed spatial uncertainty for bounding box labels in the KITTI dataset [209]. Objects are shown in the LiDAR Bird's Eye View (BEV). There exist errors (or uncertainty) inherent in labels. For object 3, estimating its length is difficult because the surface information is only available on the side facing towards the ego-vehicle. For object 5, the bounding box label does not even fully cover the LiDAR reflections near the bottom-left corner. Original data labels are deterministic, and they do not provide information on label wellness. In this work, we infer label uncertainty via a generative model of LiDAR points.

according to its uncertainty, and provides richer localization information than IoU. Using our proposed metric, we study the quality of uncertainty estimation from a state-of-the-art probabilistic object detector in KITTI [209] and Waymo [210] datasets.

In summary, our **contributions** are three-fold:

- We infer the inherent uncertainty using a generative model in bounding box labels for object detection, and systematically analyze its parameters.
- We propose a new evaluation metric called JIoU for the object localization task, which considers label uncertainty and provides richer information than IoU when analyzing

probabilistic object detectors.

- We conduct comprehensive experiments with real-world datasets to justify the proposed method.

Modelling Label Noises

Explicitly modelling label noises (or uncertainty) has been an active research field [220]. Whereas it is common to assume independent Gaussian noises for target regression, class label noises are much more complex to model, and incorrect assumptions may deteriorate the model performance (e.g. flipping class labels reverses the prediction results). Therefore, almost all proposed methods in the literature focus on modelling class label noises, especially in the image classification task [212, 213, 214, 215]. For instance, Sukhbaatar *et al.* [212] assumes noisy class labels are conditionally independent of input images, and uses an output layer to directly predict noisy labels. Lawrence *et al.* [213] and Xiao *et al.* [214] characterize input samples, noisy labels and the latent clean labels via direct graphical model, and Vahdat [215] uses Conditional Random Fields (CRF). They infer the latent clean labels by Expectation Maximum (EM) and improve the classification accuracy. Only [221] and [222] are closely related to our work, which use simple heuristics to model uncertainties in bounding box labels for object detection. [221] approximates the uncertainty with the IoU value between the label and the convex hull of aggregated LiDAR points. [222] assumes Laplacian noises within bounding box labels. They interpret the Huber loss as the KL divergence between label uncertainty and predictive uncertainty and select hyper-parameters based on intuitive understanding of label uncertainties, in order to train object detection networks.

Our methodology can be linked to the measurement models used in the LiDAR-based 3D single target object tracking [223, 224, 225]. We assume a (generative) measurement model which generates noisy LiDAR measurements given a latent object state. However, instead of inferring a probability distribution over the latent object state from measurements over time and a prediction prior of a previous time-step, we infer a distribution over the latent object label from human annotators, given measurements of a single time-step and the assumption of known mean.

Evaluation Metrics for Object Detection

Intersection over Union (IoU) has been de fecto the standard metric to measure localization accuracy. It is often used to determine true positives and false positives among predictions, given a certain IoU threshold (e.g. in KITTI car detection benchmark, IoU is set to be 0.7 [209]). Furthermore, it has been extended as auxiliary losses during training to improve the detection performance [226, 227, 228, 229]. Based on the fixed IoU threshold, Average Precision (AP) is derived as the standard metric to measure detection accuracy [216]. However, AP does not fully reflect the localization performance of a detection algorithm, since all predictions higher than the IoU threshold are treated equally. Observing this fact, the

MS COCO benchmark [230] calculates AP averaged over several IoU thresholds to show the difference. Oksuz *et al.* [217] proposes a new evaluation metric called Localization Recall Precision (LRP) to incorporate the IoU score for each detection. The NuScenes object detection benchmark [231] defines several new geometric metrics such as Average Translation Error (ATE), Average Scale Error (ASE) and Average Orientation Error (AOE) to specifically measure the bounding box localization performance. While those metrics only show the performance of deterministic object detection, Hall *et al.* [219] focus on probabilistic object detection and propose the Probability-based Detection Quality (PDQ) metric, which jointly evaluates semantic probability (by the "Label Quality" term) and spatial probability (by the "Spatial Quality" term). The optimal PDQ is obtained when the predicted probability matches the absolute prediction error (e.g. a smaller IoU between a predicted bounding box and the ground truth indicates higher spatial uncertainty).

Probabilistic Object Detection Networks

There are two types of uncertainties we can model in an object detection network: the epistemic uncertainty and aleatoric uncertainty [232]. The epistemic uncertainty shows the model's capability to describe the observed data, while the aleatoric uncertainty reflects observation noises inherent in environments or sensors. Previous studies model epistemic uncertainty using Monte-Carlo Dropout [233] to improve detections in the open-set conditions [234], or reduce the labeling efforts in the active learning framework [235]. Other works model aleatoric uncertainty by directly assuming a probability distribution in the network's outputs (e.g. Gaussian distribution), and optimizing the network with attenuated loss [232]. They discover that aleatoric uncertainty, especially in the bounding box regression task, can largely improve the detection accuracy [236, 237, 238] and reduce the number of false positive detections [239, 240]. Furthermore, [241, 242] study the impact of merging strategy such as Non Maximum Suppression (NMS) on the uncertainty estimation, and [243, 244] identify miscalibrated uncertainties in object detection networks.

Problem Statement

Labels in standard object detection datasets for autonomous driving (such as KITTI) usually include object classes cls, and deterministic object 3D poses and sizes y in the form of bounding boxes. Denote x_{all} as the set of all LiDAR points in a scan. Our target is to estimate the posterior distribution of bounding box labels, and extend IoU to a new metric called Jaccard IoU (JIoU) that incorporates spatial uncertainty. In this work, We demonstrate our method in vehicle objects.

We usually have the prior knowledge of object shape given its class cls. Therefore, it is possible to infer the posterior distribution of y using the prior knowledge and observation x_{all}, i.e. $p(y|x_{all}, cls)$. For this purpose, we assume:

- Labeling of class cls is accurate.

- Segmentation is accurate. The set of points x belonging to the object segmented out from x_{all} by human annotation has few outliers.
- Human-labeled bounding box parameters $\overline{y}$ is the accurate mean of y. We only care about the spread of y.

Under these assumptions, the position, rotation and size of an object y is only conditioned on the observation of points belonging to this object, denoted as $x = \{x_1, \cdots, x_K\}$. The posterior distribution $p(y|x_{all}, cls)$ is denoted as $p(y|x)$ for notation simplicity. We are especially interested in $p(y|x)$ because the LiDAR observations provided to the human annotators are usually not enough for determining the full size and center of the object, due to occlusion and sensor noises.

An object in detection task is parameterized by its center location c_1, c_2, c_3, 3D extents l, w, h and orientation r_y, i.e. $y = [c_1, c_2, c_3, l, w, h, r_y]$. This provides a universal affine transformation from the point $v_0 \in [-0.5, 0.5]^3$ of the unit bounding box $B(y^*)$ to the actual point $v(y)$ on the object surface in the 3D space:

$$v(y) = R_y \begin{bmatrix} l & 0 & 0 \\ 0 & w & 0 \\ 0 & 0 & h \end{bmatrix} v_0 + \begin{bmatrix} c_1 \\ c_2 \\ c_3 \end{bmatrix}, \tag{8.1}$$

with R_y being the rotation matrix from r_y, and v_0 depending on the prior knowledge of the shape, e.g. from the CAD model, bounding box or point cloud rendering method [245]. The graphical model is then illustrated in Fig. 8.3. In this paper, we uniformly sample v_0's from the bounding box boundary to generate the object surface for simplicity.

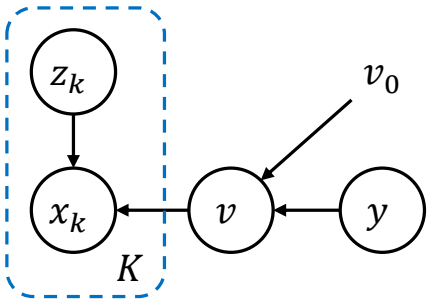


Figure 8.3: Probabilistic graphical model for the inference of y. z_k is the latent variable associated with each observation, which contains the semantic meaning such as the part of the object that the point x_k belongs to.

The distribution of object $p(y|x)$ is in the parameter space of $y \in \mathbb{R}^7$ is difficult to visualize and represent the uncertainty of a bounding box. Therefore, Section 8.2 proposes to transform the distribution $p(y|x)$ into a distribution $p(u)$ in the 3D space $u \in \mathbb{R}^3$ or BEV

space $u \in \mathbb{R}^2$. Such label spatial distribution allows us to extend IoU to a metric (JIoU) that measures the probabilistic object localization performance, described in Section 8.3. Section 8.4 introduces how we derive the posterior distribution $p(y|x)$ via a variational Bayes method.[1]

8.2 Probabilistic Representation of Objects with Uncertainty

A proper representation of the probabilistic bounding box is desired for evaluating the probabilistic detection [243, 219, 247]. Prior to our work, the uncertainty of 3D bounding boxes are represented by the error margin of their sizes as used by the introduction figure of [248]. It does not show the full uncertainty of bounding boxes such as position and rotation uncertainty. The PDQ proposed in [219] defines a random field in 2D image but is not practical in 3D. We propose a generative model that generates a spatial distribution of the bounding box in 3D or BEV. It provides a visualization of uncertainty and is later shown that it supports the extension of the commonly used IoU.

The idea of probabilistic box representation is proposed in PDQ [219] for 2D axis-aligned bounding box of image. The resulting spatial distribution $P(u) \in [0, 1]$ denotes the probability of pixel $u \in \mathbb{Z}^2$ belonging to the object. A natural generalization of $P(u)$ to $u \in \mathbb{R}^3$ in 3D space or $u \in \mathbb{R}^2$ in BEV for rotated bounding box $B(y)$ is:

$$P_{PDQ}(u) := \int_{\{y|u\in B(y)\}} p_{\widehat{Y}}(y|x)dy, \tag{8.2}$$

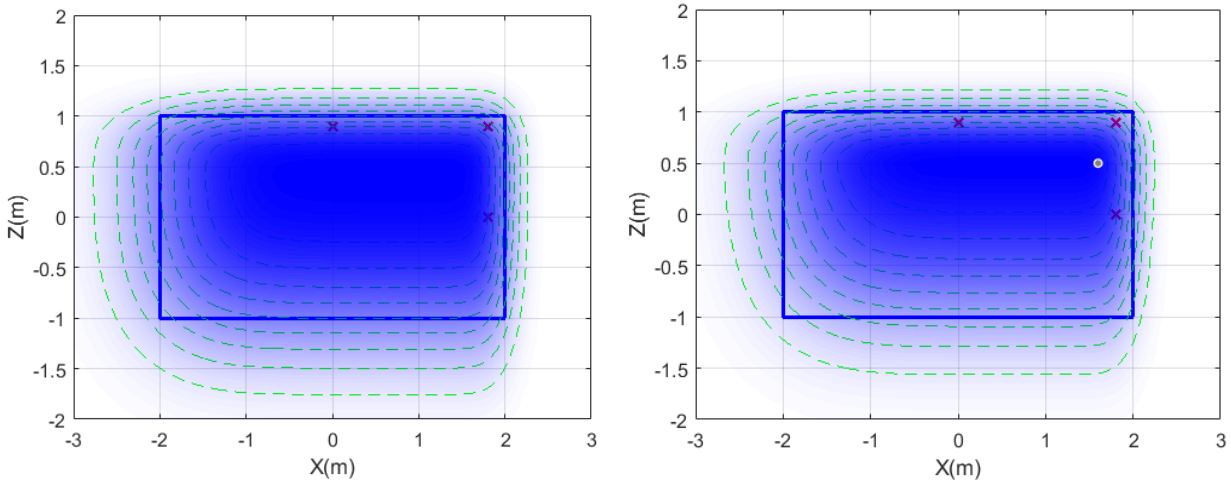
where $P(u)$ is the probability that u is a point of the object. The subscript $\widehat{Y}$ (or Y) is the random variable of the detection (label). (8.2) is easy to calculate for axis-aligned bounding boxes but hard for rotated boxes because it has to integrate over the space of y, which is 7 dimensional for 3D. A transformation in the integral gives another expression as a probabilistic density fumction (PDF):

$$\begin{aligned} p_G(u) &:= \int_{v_0\in B(y^*)} p_{V\left(v_0,\widehat{Y}\right)}(u)\, dv_0 \\ &= \int_{\{y|u\in B(y)\}} \frac{1}{A(y)} p_{\widehat{Y}}(y|x)dy, \end{aligned} \tag{8.3}$$

where $V(v_0, Y)$ is defined in (8.1), and V and Y are random variables. v_0 is added as an argument because we need to integrate over v_0. Given the probabilistic density of Y, e.g. from Section 8.4, it is not difficult to get the density of $V(v_0, Y)$ as (8.1) is quite simple. Details of proof and calculation of $V(v_0, Y)$ is left to be included in Appendix A.1.

[1] This work includes materials from the author's previously published paper [246].

The proposed definition of spatial distribution p_G is slightly different from P_{PDQ} by scaling the density with the size $A(y)$ of the bounding box $B(y)$. The scaling factor enables the transformation to a integral over distributions generated by points v_0 inside the unit box $B(y^*)$, i.e. $A(y^*)=1$. Therefore, it has a significant advantage of reducing the dimension of the integral from 7 to 3 for 3D. Besides, $\int_u p_G(u)=1$ if it is integrated on spatial points u, but $P_{PDQ}(u)$ is not normalized. The shapes of their distribution differ only when the object size is uncertain and the proposed $p_G(u)$ tend to be more concentrated, as shown in Fig. 8.4.



(a) $P_{PDQ}(u)$ defined by (8.2) [219]

(b) $P_G(u)$ defined by (8.3)

Figure 8.4: Spatial distributions of the BEV bounding box with the label uncertainty calculated by (8.10).

8.3 Jaccard IoU: A New Metric on Probabilistic Bounding Boxes

IoU is one of the most commonly used metrics for detection. It has an intuitive geometry definition measuring the overlap between the predicted and ground truth bounding boxes. Despite its popularity, IoU only applies to deterministic predictions and labels. In this section, we define a metric over $p(u)$ of probabilistic bounding box following the definition of the probabilistic Jaccard index [249]:

$$JIoU := \int_{R_1 \cap R_2} \frac{du}{\int_{R_1 \cup R_2} \max\left(\frac{p_1(v)}{p_1(u)}, \frac{p_2(v)}{p_2(u)}\right) dv}, \tag{8.4}$$

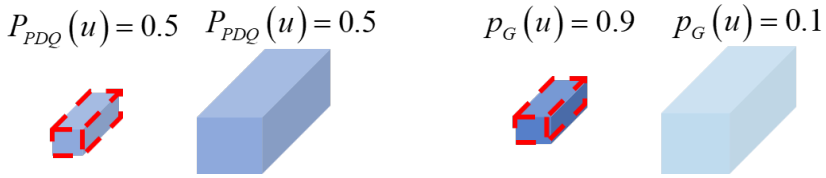
p_1, p_2 are the spatial distributions of two boxes, as introduced in (8.3). u, v are points in the 3D or BEV space. R_1, R_2 are the supports of p_1, p_2, respectively. Note that JIoU degenerates to IoU when two boxes are deterministic, i.e., $p(y|x)$ is a delta function, where R_1, R_2 become bounding boxes and p_1, p_2 become uniform inside their boxes. Additionally, it can be proved that JIoU has the following property if either box is deterministic

$$\mathbb{E}_{Y,\hat{Y}}\left[\text{IoU}\left(y,\hat{y}\right)\right] \leq \text{JIoU} \leq \frac{\mathbb{E}\left[Area\left(B(y)\cap B(\hat{y})\right)\right]}{\mathbb{E}\left[\max(A(y),A(\hat{y}))\right]}, \tag{8.5}$$

which means that JIoU is no less than the mean of IoU over all possible box parameters $y, \hat{y}$ but less than the expected intersection over the largest box. Some properties of JIoU can be concluded as:

- JIoU = IoU if two boxes are both deterministic
- The computational complexity is $O(NlogN)$ using Eq.3 of [249] if N points are sampled from $R_1 \cup R_2$.

The proposed $p_G(u)$ is more reasonable than $P_{PDQ}(u)$ when JIoU is used as a metric. Consider when label Y is a discrete random variable with two values of equal probability, i.e. two possible bounding boxes, and when the prediction $\hat{Y}$ only fits one of the box as shown in Fig. 8.5. Then $P_{PDQ}(u) = 0.5$ for the label and JIoU$(Y, \hat{Y})\approx 0$ because one box is much smaller. On the contrary, JIoU$(Y, \hat{Y})$=0.5 under p_G, no matter what the size difference is. Here it is desired that JIoU = 0.5 because the prediction has matched one of the two possible ground truths.



(a) JIoU=0.1 using P_{PDQ} (b) JIoU=0.5 using p_G

Figure 8.5: Spatial distributions of discrete Y with two possible values in blue and predicted box $\hat{Y} = \hat{y}$ in dashed red line.

8.4 Inferring Spatial-uncertainty of Labels

We start to elaborate our method with a simple example given in Fig. 8.6 in the bird's eye view (BEV) with point clouds as observation $x = \{x_1, x_2, x_3\}$. Three points are segmented

out inside the bounding box as red cross markers. The posterior is solved by Bayes rule:

$$p(y|x) = \frac{p(x|y)p(y)}{p(x)}, \tag{8.6}$$

assuming that $p(x|y) = \prod_{k=1}^{K} p(x_k|y; v_k)$, K=3, where each point $x_k \in \mathbb{R}^2$ is independently generated by the nearest point $v_k(y) \in \mathbb{R}^2$ on the boundary of the ground truth bounding box with Gaussian noise:

$$p(x_k|y; v_k) \sim \mathcal{N}\left(v_k(y), \sigma^2\right), \quad \sigma{=}0.2m.$$

The label parameters are center and size $y{=}[c_1, c_2, l, w]$. The posterior $p(y|x)$ is then Gaussian given Gaussian prior $p(y) \sim \mathcal{N}([0, 0, 4, 2], 100^2)$:

$$p(y|x) = \prod_{k=1}^{3} p(x_k|y; v_k)p(y) \sim \mathcal{N}(\cdot, 0.01 \times \begin{bmatrix} 4 & 0 & -4 & 0 \\ 0 & 4 & 0 & -4 \\ -4 & 0 & 6 & 0 \\ 0 & -4 & 0 & 6 \end{bmatrix}).$$

Different from others [221, 236] who calculate the uncertainty of predictions and labels as independent variables, the uncertainty derived here is a joint distribution with correlation. A singular value decomposition (SVD) of the covariance matrix shows that two edges of the bounding box which have point cloud observations, namely $X = c_1 + l/2$ and $Z = c_2 + w/2$, have the smallest standard deviations $0.09m$ and $0.09m$. The other two edges without observation have the largest standard deviations $0.43m$ and $0.43m$. Fig. 8.6 illustrates the confidence interval within one standard deviation by green dashed bounding boxes. The advantage of producing a joint distribution is significant. For those who produce disjoint distributions on size and pose, the confidence intervals or variances are the same between face and back, and between left and right side of the car. Meanwhile, it is the common sense that surfaces with more observations should have less variance, as shown by Fig. 8.6.

A more general model is by assuming that $p(x|y)$ is some mixture model with categorical latent variable z, e.g. a Gaussian Mixture Model (GMM) of center points $v_j(y)$'s:

$$p(x_k|y) = \sum_{j=1}^{M} p\left(z_k{=}j\right) \mathcal{N}\left(v_j(y), \sigma_j^2 I\right), \tag{8.7}$$

Each $v_j(y)$ is created from a unique point inside the unit bounding box $B(y^*) := [-0.5, 0.5]^3$, by using (8.1). σ_j^2 is an empirical covariance related to the sensor noise and the confidence of the rendering method. We use the surface of the boundary of the box to create v_0 and $\sigma_j{=}\sigma$ for all j, but it does not exclude the potential of more complex rendering methods.

Suppose $p(y|z) = p(y)$ and let $p(z = z_j|x) = \frac{1}{M}$, a numerical solution of the posterior exists by calculating the naive Bayes in (8.8). The corresponding joint distribution of c_2, w is shown in Fig. 8.4(a). It can be seen that $c_2 + w/2$ has small variance while $c_2 - w/2$ has large variance which is the same as the previous solution.

$$p(y|x) = \int_z \frac{p(x|y, z)p(y)}{p(x|z)} p(z|x). \tag{8.8}$$

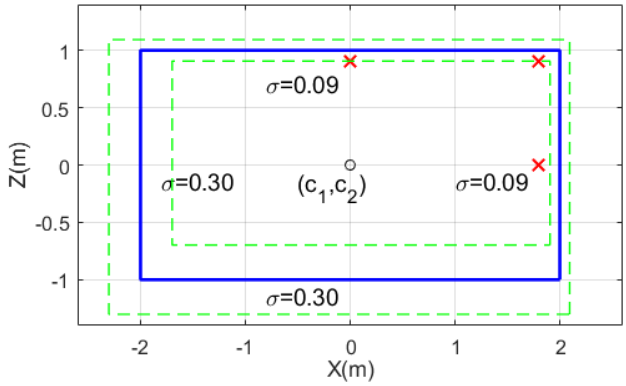


Figure 8.6: A simple demonstration of calculating the posterior with deterministic latent variables. x_1=$(1.8, 0)$, x_2=$(1.8, 0.9)$, x_3=$(0, 0.9)$ and corresponding v's are v_1=$(c_1$+$0.5l, 0)$, v_2=$(c_1$+$0.5l, c_2$+$0.5w)$, v_3=$(0, c_2$+$0.5w)$

(8.8), however, is usually intractable in non-trivial cases. An approach to solve the posterior of GMM is by variational Bayes (VB), assuming y, z are independent. A good practice of VB is the point registration method. The problem then becomes solving $q(y)$, $q(z)$ that minimize the KL-divergence between the assumed class of distribution and the actual posterior:

$$D_{KL}\left(q||p\right) = \int_z q(y,z) \log \frac{q(y,z)}{p(y,z|x)}, \tag{8.9}$$

and the solution is given below by mean field method [250]:

$$\begin{aligned} q(y) &\propto exp\Big\{ \log p(y) + \mathbb{E}_z \left[\log p(x|z,y)\right] \Big\} \\ &\propto exp\Big\{ -\sum_{j=1}^{M} \frac{1}{2\sigma_j^2} \sum_{k=1}^{K} \varphi_{jk} ||x_k - v_j(y)||^2 \Big\}, \end{aligned} \tag{8.10}$$

with $\varphi_{jk} := p(z_k = j|x_k)$ being the probability of registering x_k to v_j and it can be calculated using the nominal value $\overline{y}$ of the ground truth bounding box:

$$\varphi_{jk} = \frac{\exp\left(-\frac{1}{2\sigma_j^2}||x_k - v_j(\overline{y})||^2\right)}{\sum_{j=1}^{M} \exp\left(-\frac{1}{2\sigma_j^2}||x_k - v_j(\overline{y})||^2\right)}, \tag{8.11}$$

where $v_j(y)$ is a linear function of y in our example used for demonstration and the resulting posterior $p(y|x)$ is Gaussian as shown in Fig. 8.7.

The proposed variational Bayes method (8.10) is used to calculate the label uncertainty of vehicles in terms of detection based on point cloud. The generative model is generated by the ground truth bounding box. For extensions to pedestrians and other objects, a more dedicated rendering model is desired and it is left for future development.

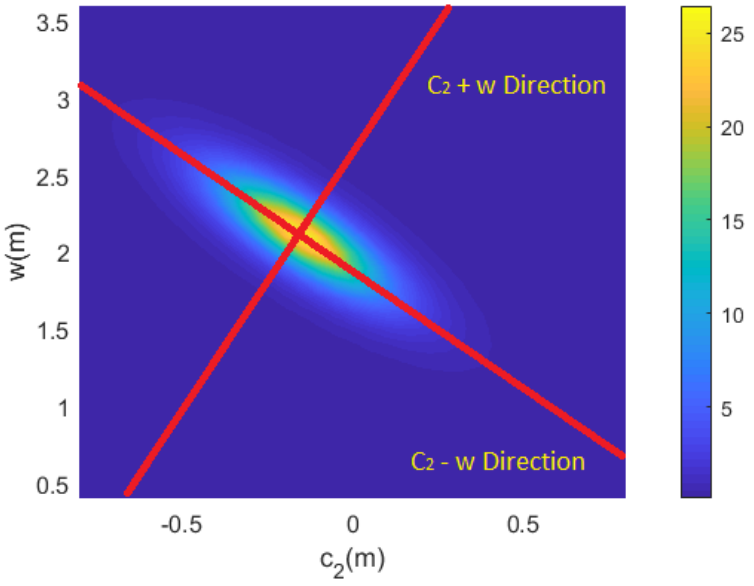


Figure 8.7: Joint distribution of $p(c_2, w|x)$ with the same observation as in Fig. 8.6 calculated by variational Bayes.

8.5 Experimental Results

In this work, we propose (1). a new spatial distribution to visualize and represent the uncertainty of bounding boxes, and (2). JIoU as an extension of IoU to evaluate probabilistic object detectors, and (3). a generative model to infer the uncertainty of bounding box labels for LiDAR point clouds. In the following, we design three experiments to verify our proposed methods. First, we study how the model parameters affect the label uncertainty, including the LiDAR observation noises and the prior knowledge of human annotators (Section 8.5). Second, we applied the visualization to the inferred uncertain labels and probabilistic detection networks to justify our methods. We show that the spatial uncertainty reflects the typical "L"-shape behaviours in LiDAR point clouds, which are also observed in state-of-the-art object detection networks (Section 8.5). In addition, we show our methods reflect the quality of bounding box labels (Section 8.5). Third, we use JIoU as a new evaluation metric to explore predictions from probabilistic object detection networks (Section 8.5).

We evaluate the proposed method on the KITTI dataset [209] and the recently released Waymo open dataset [210]. Both provide LiDAR observations and 3D object labels from human annotators. We use the "Car" category on the KITTI training dataset, with 7481 data frames and nearly 30K objects. As for the Waymo dataset, we select the training data drives recorded in San Francisco, and down-sample the frames by a factor of 10. The original "Vehicle" class in the Waymo dataset does not distinguish among objects such as motorcycles, cars or trucks, making it difficult to directly compare with the KITTI dataset.

Therefore, we extract the "Car" objects from the "Vehicle" class by thresholding the vehicle length within 3m−6.5m. We use such modified Waymo database with 7545 frames and over 150K objects. When evaluating our proposed label uncertainty with the help of object detection networks, We report the standard Average Precision (AP) from the BEV (AP_{BEV}), with IoU=0.7 threshold. Objects in the KITTI dataset are categorized into "Easy", "Moderate" and "Hard" settings [209], while objects in the Waymo dataset are categorized by thresholding the LiDAR range up to 30m, 50m and 70m.

Choice of Parameters for Label Uncertainty

The inference of label uncertainty $p(y|x)$ introduced in Section 8.4 allows us to incorporate prior knowledge of LiDAR observation noises $p(x|y,z)$. Furthermore, it can incorporate models for human annotation uncertainty in the priors $p(y)$. This section explores the impact of parameters on the label uncertainty. More specifically, we examine the standard deviation of LiDAR observation noise σ, and the covariance of $p(y)$. We empirically define the covariance matrix in BEV of $y = [c_1, c_2, l, w, r_y]^T$, referring to the variance of vehicle size for all objects in the KITTI dataset: $1/w \times \text{diag}([0.44^2, 0.11^2, 0.25^2, 0.25^2, 0.17^2])$, where w is the weight of the prior knowledge. Larger w means smaller variance of prior distribution and more confidence in human annotators. Fig. 8.8 shows the evolution of spatial distribution $p(u)$ and JIoU score for a bounding box label with increasing σ and w. There are only a few LiDAR observations on the front surface of the vehicle. This results in high uncertainty, or low density, at the opposite side of the bounding box label, if no prior distribution is added. As more prior knowledge is incorporated (e.g.. increasing weights w for human annotations, or decreasing observation noise σ), the posterior uncertainty decreases, resulting in higher JIoU score. Furthermore, we observe that labels with lots of observed LiDAR points are almost never affected by the choice of parameters. They have an uniform spatial distribution inside bounding boxes, even without prior distribution.

Instead of empirically choosing the value of σ, it can be estimated by the EM algorithm in [213] using

$$\sigma^2 = \frac{1}{Md} \sum_{j=1}^{M} \sum_{k=1}^{K} \varphi_{jk} ||x_k - z_j(\overline{y})||^2, \tag{8.12}$$

with iterative updates. The resulting σ is 0.2m for KITTI and 0.3m for Waymo. (8.12) implies that these results are very close to the root mean square of distances between LiDAR points and bounding box labels. The chosen value of σ^2 includes both the measurement noise of the LiDAR sensor and the approximation error of a bounding box to the actual surface of a car.

Justification of uncertainty model

The uncertainty model proposed in this paper suggests that different points of the bounding box have different variances: points close to the LiDAR are likely to have smaller variances.

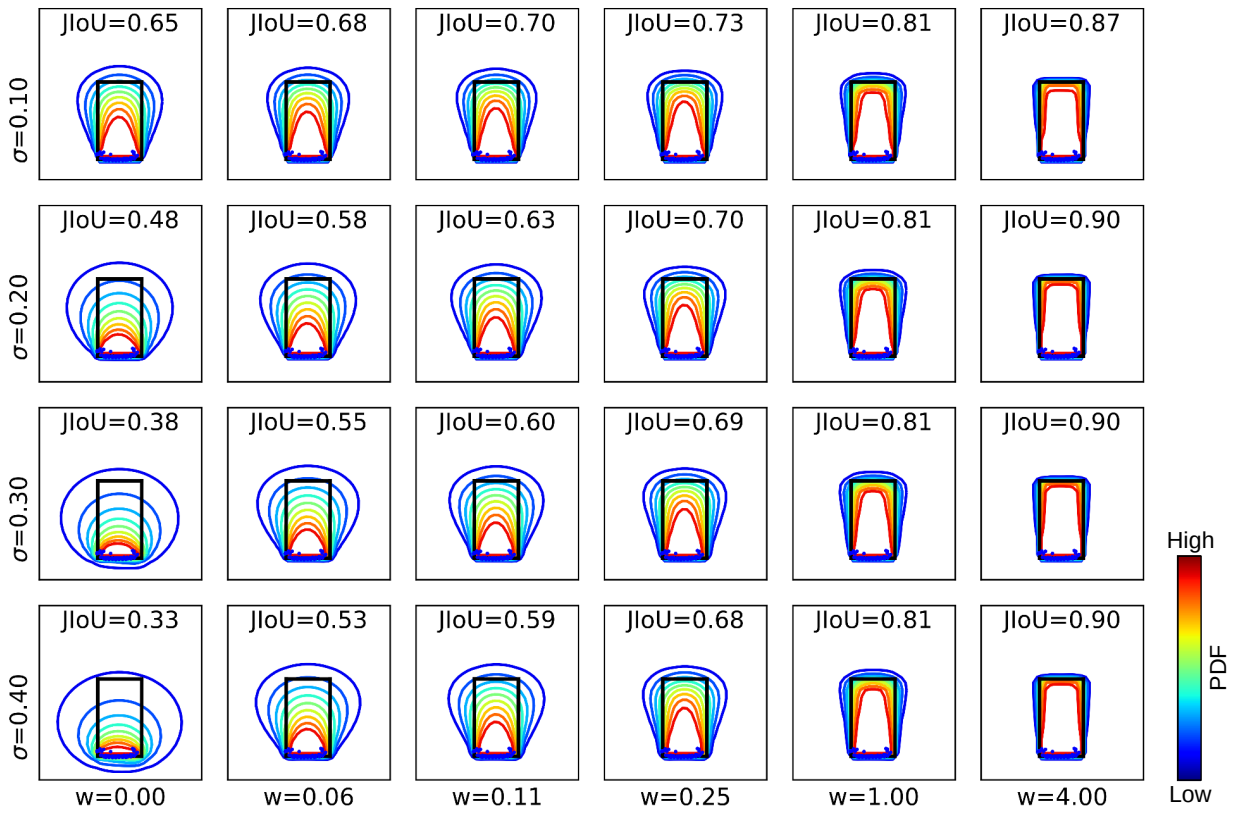


Figure 8.8: Influence of LiDAR measurement noise and prior distribution on spatial distribution. The sample is drawn from frame 1287, object 11 in KITTI.

The importance of modeling the variance of different points is also revealed in a probabilistic detection paper [247]. Fig. 8.9 measures the average total variances (TV) of each of the four corners in the BEV bounding boxes in the KITTI and Waymo datasets (C1-C4, sorted by their distance to the ego-vehicle in the ascending order). The TV scores are calculated by our proposed spatial distribution. We observe that the nearest corner C1, which usually has dense laser points are more reliable than the center of the box and the distant, occluded corders, by showing smaller TV scores. This observation corresponds to the intuition of L-shapes [251] widely used for vehicle detection and tracking. The nearest corner of the L-shape is determined more confidently by our method while the size of the vehicle is more uncertain. Considering the mechanism of LiDAR perception, we expect that using object detectors to predict the nearest corner for location and orientation should be less prone to label noise than predicting the center. Table 8.1 verifies this result and shows how a small modification of "corner" and "center" directly benefits the AP of multiple networks.

In the table, We modify the outputs from several state-of-the-art LiDAR-based object

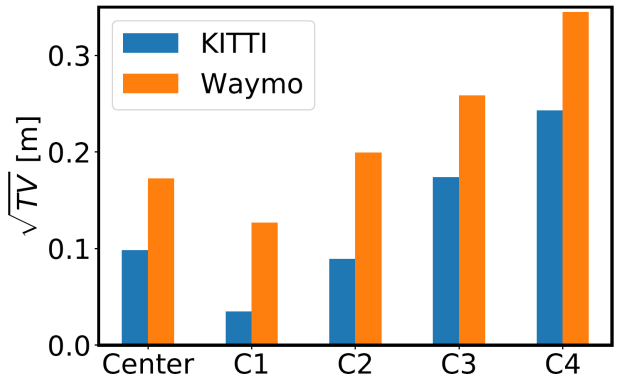


Figure 8.9: The average total variances [236] of centers and corners of all "Car" objects in the KITTI dataset. Variances are calculated by the proposed uncertainty model. Corners (C1-C4) are sorted by their distances to the ego-vehilce.

detectors, and show how the location and orientation accuracy is governed by L-shape. We use STD [179], SECOND [172], AVOD [252], VoxelNet [253] and PointRCNN [254] in the KITTI dataset, and PIXOR [255] in the Waymo dataset. Following the nuScenes [231] evaluation criterion which uses partial ground truth information to adjust predictions in order to separate location and size accuracy, we evaluate networks by replacing size predictions with ground truth labels, while keeping the center predictions or the nearest corner predictions unchanged.

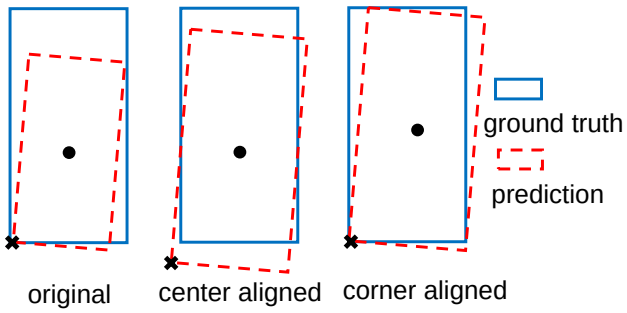


Figure 8.10: Column "origin" in Table 8.1 does no modification to the predicted bounding box. "center aligned" assembles the bounding boxes using center predictions and ground truth sizes. "corner aligned" using nearest corner predictions and ground truth sizes.

The predicted bounding boxes are reassembled as shown in Fig 8.10. "corner aligned" consistently demonstrates much better AP_{BEV} improvement than "center aligned" among

all networks and datasets. The result justifies that the networks have better predictions on corners than centers. Our proposed label uncertainty naturally captures this behaviour, and has the potential to provide rich information when evaluating object detection performance.

To conclude, our proposed methods reflect the typical "L"-shape in LiDAR observations, which is also observed in LiDAR-based object detection networks.

Table 8.1: Change of $AP_{BEV}(\%)$ of two types of predicted bounding boxes compared to their original values on the KITTI *val* set.

KITTI	Origin (Easy, Moderate, Hard)	Center aligned	Corner aligned
STD [179]	90.0, 88.1, 87.7	$+0.0, +0.1, +0.2$	$+0.1, +0.4, +0.4$
SECOND [172]	89.9, 87.9, 86.8	$+0.0, +0.2, +0.1$	$+0.1, +0.5, +0.5$
AVOD [252]	88.9, 79.6, 78.9	$-0.2, +0.0, +0.1$	$+0.0, +6.3, +0.4$
Voxel [253]	81.6, 70.7, 65.8	$+1.2, +1.1, +1.0$	$+3.0, +4.4, +8.0$
PointRCNN [254]	88.8, 86.3, 86.0	$+0.0, +0.4, +0.7$	$+0.2, +0.7, +1.2$
Waymo	Origin (< 30m, < 50m, < 70m)	Center aligned	Corner aligned
PIXOR [255]	$62.2, 54.3, 48.5$	$+3.8, +5.5, +4.0$	$+4.3, +6.7, +6.9$

Label Quality Analysis

In this section, we study how the proposed spatial uncertainty captures the quality of bounding box labels in the KITTI dataset. We evaluate the spatial uncertainty by the JIoU score between the (deterministic) object label and its spatial distribution from the our proposed generative model ("jiou-gt"). It is compared with "cvx-hull-iou" proposed by Meyer *et al.* [221], which approximates the spatial distribution by measuring the IoU value between the bounding box label of an object and its convex hull of aggregated LiDAR points. We also calculate the number of points within an object as a simple heuristic ("num-points"). All three methods range between 0 and 1, with larger scores indicating better label quality.

First, we study the relationships among three methods. Fig. 8.11 illustrates their distributions for all objects in the KITTI *val* set. We observe that the relationship between "cvx-hull-iou" and "num-points" is relatively small (Fig. 8.11(a)). This is because they capture different aspects of spatial uncertainty. "cvx-hull-iou" assumes that if larger parts of an object are observed, the uncertainty is smaller. In contrast "num-points" focuses on the observation density. "jiou-gt" is highly related both with "cvx-hull-iou" and "num-points" (Fig. 8.11(b) and Fig. 8.11(c)), indicating that our proposed method naturally considers both the size and density of observed region when inferring spatial uncertainty.

Then, we study how three spatial uncertainty methods behave on objects which tend to have large label noises. Ignoring errors inherent in labels can not only limit or even deteriorate the performance of an object detector during training, but also fail to reflect its full detection performance during evaluation, as each testing data point is treated equally,

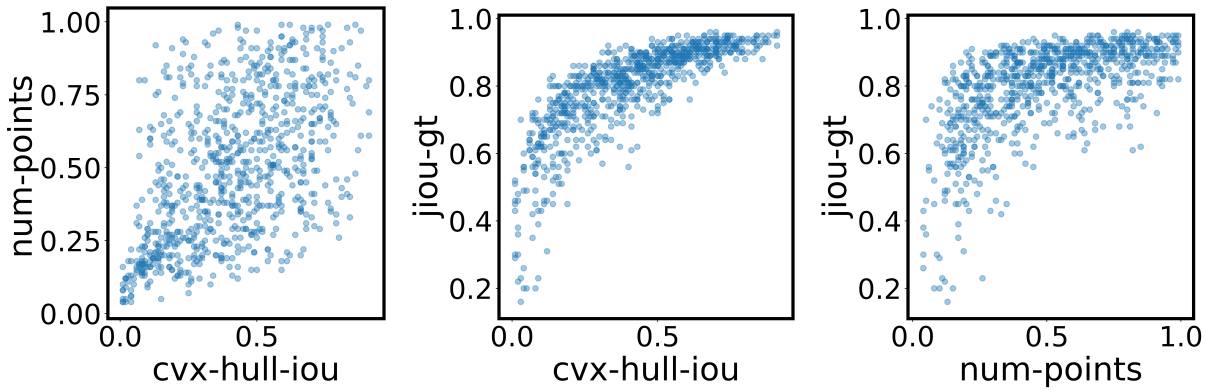


Figure 8.11: Distributions of different label uncertainty measures for all objects in the KITTI *val* set. Our method ("jiou-gt") is compared with convex hull method ("cvx-hull-iou") proposed by [221], as well as the simple heuristic that calculates the number of LiDAR observations within an object ("num-points"). All three uncertainties range among $[0, 1]$.

regardless the quality of the corresponding label (e.g. object 5 in Fig. 8.2 shows high label error). Ideally, the modelled spatial uncertainty should depict higher values in bad labels than the labels with high quality. However, it is difficult to directly study the label noises within object detection datasets, as they do not provide "ground truths" of label uncertainty. Instead, we leverage the common false negatives from several state-of-the-art detectors, with the assumption that objects with bad labels are also difficult to be detected. In this regard, we collect the common false negatives from detectors listed in Tab. 8.1 as bad examples, and the rest of the objects as good labels. We threshold spatial uncertainty scores to discriminate good labels from the bad ones. From the ROC curves in Fig. 8.12, we observe that "jiou-gt" performs better than "covx-hull-iou", showing that it captures more reliable spatial uncertainty. "num-points" outperforms the other two methods, because we select bad labels based on predictions from LiDAR-based object detectors, whose classification performances highly depend on the number of LiDAR observations. We leave it as an interesting future work to do evaluation with "ground truths" of label uncertainty, which can be generated by querying human annotators or by simulation.

To conclude, our proposed spatial uncertainty captures the quality of bounding box labels.

JIoU on Different Networks

The proposed JIoU, as an extension of IoU, allows us to evaluate the prediction or label uncertainties under the same evaluation framework of deterministic object detection. In this section, we demonstrate how to get insights of probabilistic detections with the help of JIoU. We employ a state-of-the-art probabilistic object detector called "ProbPIXOR" from [243], which models data-dependent (aleatoric) uncertainty on PIXOR [255] - a deter-

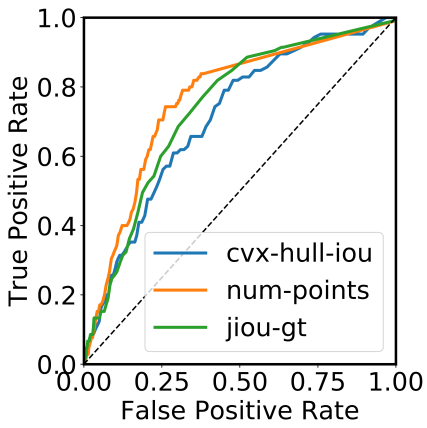


Figure 8.12: ROC curves for detecting bad labels in the KITTI *val* set, by thresholding spatial uncertainty scores. We compare among three spatial uncertainty methods, including "jiou-gt", "cvx-hull-iou", and "num-points".

ministic LiDAR-based object detection network. ProbPIXOR assumes the regression variables are Gaussian-distributed with diagonal covariance matrix. Since ProbPIXOR is shown to predict over-confident or under-confident uncertainties in [243], we further calibrate its uncertainty estimation based on temperature scaling proposed in [243] and call the new network "CalibProbPIXOR". The only difference between CalibProbPIXOR and ProbPIXOR is the scale of variances in the Gaussian distribution. We calculate the JIoU between the predicted and the label bounding boxes for all three networks (PIXOR, ProbPIXOR and CalibProPIXOR), as well as the ground truth JIoU following the method in Section 8.4. The aleatoric uncertainties of ProbPIXOR and CalibProbPIXOR are used to construct the distribution of predicted boxes, which include the box center position, length, width and heading. As for PIXOR, JIoU is directly applied to the deterministic box predictions. Note that all three networks are not optimized for JIoU metric, and ProbPIXOR and CalibProbPIXOR only produce symmetric probability distributions due to the bounding box encoding.

We first explore how JIoU behaves on predictions of ProbPIXOR and CalibProbPIXOR in the KITTI dataset. We observe that JIoU scores are in general consistent with IoU scores, with higher IoU corresponding to larger JIoU, as illustrated in Fig. 8.13. However, JIoU provides us with additional uncertainty information to evaluate detections. For example, Fig. 8.13(a-2) shows a bad detection with only IoU=0.03. However, the ground truth JIoU is already small with JIoU=0.39 (Fig. 8.13(a-1)) due to sparse LiDAR observations, indicating that over-emphasizing the detection performance for this object is unnecessary. Another example are detections in Fig. 8.13(b-2) and Fig. 8.13(c-2). Both are measured with the same IoU but different JIoU scores, because of different label and predictive probability distributions. Furthermore, we observe in the third row of Fig. 8.13 that the predictive probability distributions become wider after calibration, because most predictions are over-

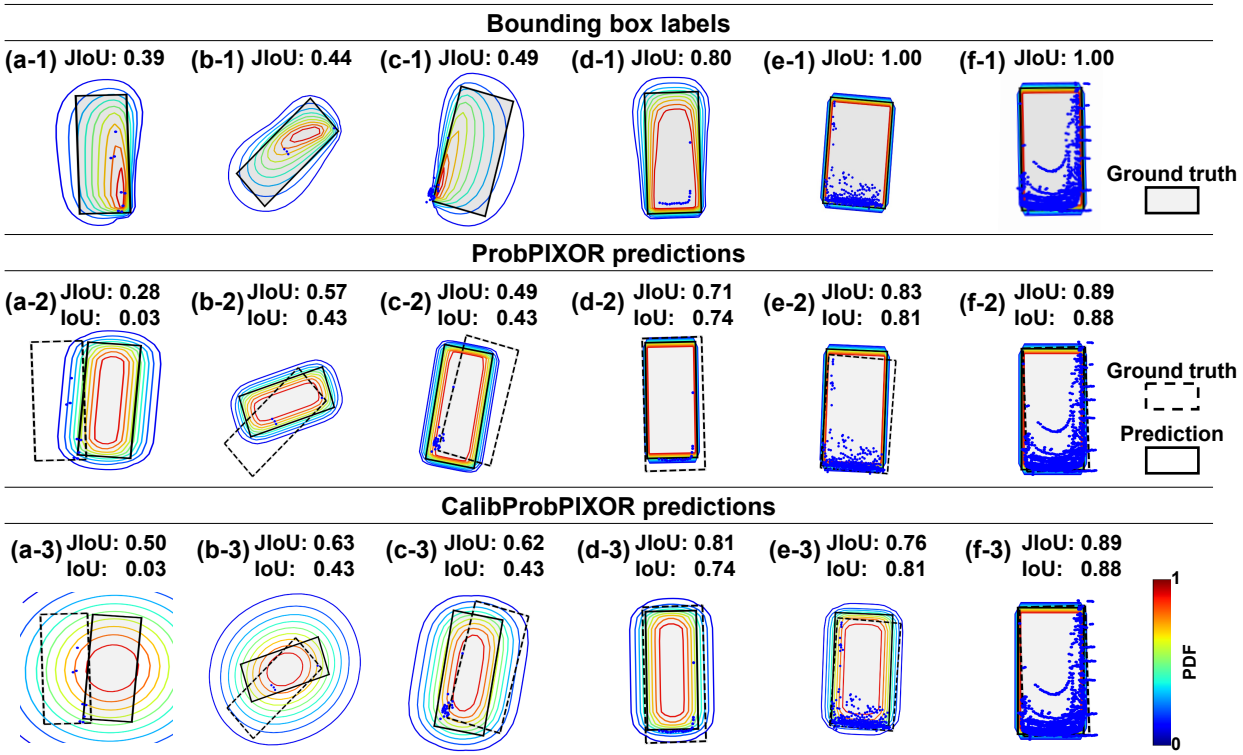
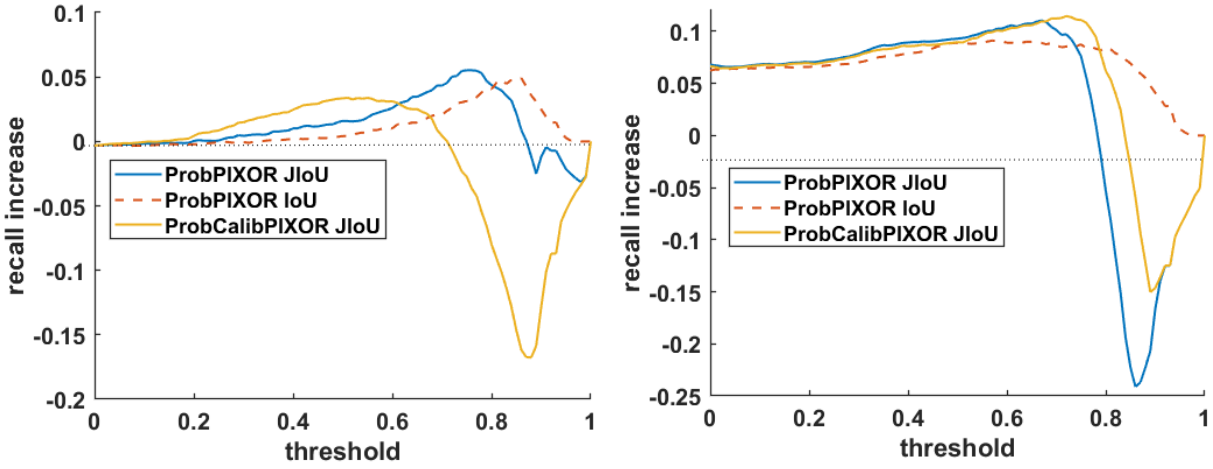


Figure 8.13: Some detection examples and their JIoU scores in the KITTI dataset. We visualize the spatial distribution for the bounding box labels in the first row, predictions from ProbPIXOR in the second row, and predictions from CalibProbPIXOR in the third row.

confident (similar to [243]). This results in improved JIoU scores in most cases compared to ProbPIXOR. However, temperature scaling only improves the uncertainty on the whole dataset, and does not guarantee that each detection is better-calibrated [243]. Therefore, we observe a worse JIoU after calibration in Fig. 8.13(e-3).

Next, we use JIoU to quantitatively analyze how modelling uncertainties affects detections. Fig. 8.14 shows the relative recall gain of probabilistic object detectors (ProbPIXOR and CalibProbPIXOR) compared with the original PIXOR network, by thresholding detections based on IoU or JIoU metrics. In both KITTI and Waymo datasets, we observe consistent recall improvement by modelling uncertainty based on the IoU metric (see "ProbPIXOR IoU" in Fig. 8.14), indicating higher Average Precision performance (as shown in [236]). When using JIoU, however, both ProbPIXOR and ProbCalibPIXOR do not perform well at high threshold (e.g. JIoU>0.8), because they only produce symmetric probability distribution (as discussed above), and cannot well calibrate our proposed spatial distribution. The result shows that JIoU effectively penalize the distribution mismatch between two probabilis-



(a) On the KITTI *val* set.

(b) On Waymo dataset.

Figure 8.14: Increase of recall from ProbPIXOR and CalibProbPIXOR compared with PIXOR, by thresholding detections based on IoU or JIoU.

tic boxes especially at high values, and indicates the potential improvements for probabilistic modelling in object detection networks (e.g. better bounding box encodings or assuming correlation between regression variables [242]).

To conclude, JIoU provides us richer information than IoU to evaluate probabilistic object detection networks.

8.6 Chapter Summary

In this work, we propose a generative model to infer the uncertainty inherent in bounding box labels of object detection datasets with LiDAR point clouds. The label uncertainty includes object shape and measurement information, and can represent non-diagonal correlation of label parameters. We further propose a new spatial distribution to visualize and represent the uncertainty of 2D or 3D bounding boxes. Finally, we propose JIoU, as an extension of IoU, to evaluate probabilistic object detection. Comprehensive experiments on KITTI and Waymo datasets verify our proposed method.

Our work can be extended in several ways. For example, it is possible to incorporate the proposed label uncertainties to train an object detector, in order to improve its robustness against noisy data. Using JIoU to evaluate uncertainty estimation performance among different probabilistic object detectors (e.g. [236, 242]) is another interesting future work. Finally, as our work is similar to the measurement models in extended object tracking, multi-target approaches like random finite sets could be used to devise a model for simultaneously

inferring multiple object labels, e.g. [224, 225], lifting the restriction on correct association of LiDAR points.

Chapter 9

Conclusion and Future Work

9.1 Summary

In this dissertation, robust perception is discussed for robots carrying vision sensors and auto-teaching is developed for robots to handle model uncertainty. Robustness of the perception subsystem is considered by developing global methods to reject disturbances and sensor fusion to improve redundancy. Several methods are proposed in both classic computer vision and deep learning areas with applications to two autonomous robotic systems, namely the industrial manipulator and the autonomous vehicle.

Industrial Manipulator System

The name "hand-eye system" is used denoting the industrial manipulator equipped with vision sensors on its arm. Chapter 2 models the system and builds the motion block. The kinematic model is used for visual-inertia sensor fusion and generating the calibration parameters for auto-teaching. Planning and tracking control of the system are necessary for closing the loop of auto-teaching and ensuring the visual data captured by the hand-eye system.

Robots and target objects have rich geometric information and accurate known shape, which is more suitable for classic computer vision (CV) methods. Chapter 3 and 4 constructs the robust perception block of the hand-eye system. Chapter 3 proposes several global shape matching methods for two kinds of visual inputs, namely image and point clouds. We globally search all potential matches of deformed target objects to avoid local optimals caused by disturbances. Chapter 4 introduces probabilistic inference to match detected objects in order to increase the robustness against noise. The proposed probabilistic hierarchical registration algorithm outperforms the deterministic feature descriptor-based algorithm used in state-of-the-art SLAM methods.

Visual detection is not robust against model uncertainty of the system and only gives 2D location of the object. Auto-teaching simultaneously calibrates the parameters of the systems while estimating the state of detected objects. Chapter 5 introduces the auto-

teaching framework using the perception results from Chapter 3 and Chapter 4. Visual-inertial sensor fusion is used to increase the calibration accuracy by taking the robot motion measurement into account. It is shown that the proposed method outperforms the existing visual SLAM method in the wafer calibration experiment. Chapter 6 proposes an active auto-teaching framework which closes the calibration loop of the hand-eye system by planning optimal measurement poses. The uncertainty of estimated parameters is reduced compared to auto-teaching methods in the chamber opening calibration experiment.

Autonomous Driving System

Deep learning-based methods are preferred in this area due their superior accuracy and generalizability, but robustness is the major concern for scaling up its application in the real world. In Part II of the dissertation, the robustness of learning-based detectors is discussed. Chapter 7 proposes two camera-LiDAR sensor fusion frameworks to increase the performance and redundancy of the 3D object detector. A sparse non-homogeneous pooling layer is developed for efficient and end-to-end fusion of detection networks of different sensors. The proposed fusion layer is able to fuse feature maps at any stage of the network and can be naturally extended to other fields of perception such as semantic segmentation and object tracking. In Chapter 8, we further dive into the training and evaluation procedure of learning-based detectors. The probabilistic representation is proposed for labels in the dataset to handle the uncertainty of training data. A new evaluation metric, called JIoU, is introduced for the proposed probabilistic representation to better measure the robustness of learning-based detectors.

9.2 Future Work

Cross-modal Perception Networks

Current deep sensor fusion networks take all the sensors into account by combining their features. The interpretability of such networks is low and predicted features are coupled which is hard to be used in other tasks. Cross-modal perception is an interesting topic which tries to decouple the common information and unique information between sensors. The decoupling makes the network more interpretable where failures can be traced by to specific sensors using the unique features. The common features make the output more stable so it can improve the robustness of the network when part of the sensors are not working. It also enables transferring knowledge between tasks with different sensor configurations.

Auto-calibration of Learning-based Methods

Auto-teaching works well for classic CV methods but it is hard to identify and calibrate parameters for learning-based methods. Networks are usually trained on a specific configuration

of the system and model uncertainty is definitely holding back the scaling of applications in the real world. Data augmentation is a common technique to improve the robustness and adaptive networks are proposed in the control area, but they require extensive manual tuning. More insight is desired to formulate a meaningful set of parameters for networks. Besides, we need a more flexible calibration framework for deployed networks that are already trained.

Appendix A

Probabilistic Representation of Detection Results

A.1 Proof of Spatial Distribution

This section give the equality of (8.4) based on the relationship between v and y given in (8.1).

Proof. Denote

$$Y := [C_1, C_2, C_3, L, W, H, r_Y]^T, C{:=}[C_1, C_2, C_3]^T, S{:=}[L, W, H], v_0{:=}[v_1, v_2, v_3]^T$$

and R_Y as the rotation matrix of r_Y and ,Then from (8.1), we have

$$R_Y^T(V-C) = \left[\left[\begin{matrix} l \\ w \\ h \end{matrix}\right]\right] \circ v_0, \tag{A.1}$$

where $\circ$ denotes the element-wise multiplication (Hadmard product) of matrices. This equation means when conditioned on $r_Y{=}r_y$ and $C = [c_1, c_2, c_3]$, the mapping between V and S is a diffeomorphism and the probability density function of $V|r_Y, C$ is

$$p_{V|r_y,c}(u) = \left|\frac{1}{v_1 v_2 v_3}\right| p_{S|r_y,c}\left(R_y^T(u-c) \circ v_0^{-1}\right), \tag{A.2}$$

where v_0^{-1} is the element-wise inverse of v_0. Then

$$\begin{aligned}
&\int_{v_0 \in B(y^*)} p_V(u)\, dv_0 \\
&= \int_{v_0 \in B(y^*)} \left(\int p_{V|r_y,c}(u) p_{r_Y,C}(r_y, c) dr_y dc \right) dv_0 \\
&= \int p_{r_Y,C} dr_y dc \int_{v_0 \in B(y^*)} \left| \frac{1}{v_1 v_2 v_3} \right| p_{S|r_y,c}\left(R_y^T (u - c) \circ v_0^{-1} \right) dv_0. \\
&= \int p_{r_Y,C} dr_y dc \int_{\{s|u \in B(c,s,r_y)\}} \left| \frac{1}{lwh} \right| p_{S|r_y,c}(s)\, dl dw dh \\
&= \int_{\{c,s,r_y|u \in B(y), y=[c,s,r_y]\}} \left| \frac{1}{lwh} \right| p_Y(c, s, r_y) dc ds dr_y
\end{aligned} \tag{A.3}$$

where $s{=}[l, w, h]^T$ are the length, width and height of the bounding box. Note that $\|lwh\| = A(y)$, hence (8.4) is proved. □

How to Calculate Spatial Distribution

The distribution $p_{V(v_0,Y)}(u)$ of $V(v_0, Y)$ is approximated by Gaussian distribution to reduce the computational load. Several possible situations may occur when calculating the spatial distribution defined by (8.4):

1. The distributions of some points of $V(v_0, Y)$, such as corners of the bounding box [219, 238], are predicted.
2. The distribution of Y is predicted but the distribution of V is unknown, which are the cases of [236, 248] and Section 8.4

The moments of $V(v_0, Y)$ is related to Y by (8.1). Remember that (8.1) can be rewritten in the homogeneous coordinate of v_0 as

$$v(v_0, y) = \Phi(y)^T w, \tag{A.4}$$

where w is the homogeneous coordinate of v_0 and

$$\Phi(y)^T = \begin{bmatrix} l\cos(r_y) & 0 & -w\sin(r_y) & c_1 \\ 0 & h & 0 & c_2 \\ l\sin(r_y) & 0 & w\cos(r_y) & c_3 \end{bmatrix}, \tag{A.5}$$

is the feature matrix. Considering the assumption that we approximate the distribution of $V(v_0, Y)$ as Gaussian, we only need to calculate

$$\begin{aligned}
E[V] &= E[\Phi(Y)]^T w \\
E[V^2] &= E[\Phi(Y)^T w w^T \Phi(Y)],
\end{aligned} \tag{A.6}$$

for all w's corresponding to the points v_0 in the unit bounding box. For 3D, $ww^T \in \mathbb{R}^4$ is a symmetric matrix. Since the base of the set $Sym_4(\mathbb{R})$ of symmetric matrices is 10, at most 10 points is enough for representing all possible ww^T's. This means there exist a set $w_1, w_2, \cdots, w_{10}$ such that for any homogeneous coordinate w, there exist coefficients $\alpha_1, \alpha_2, \cdots, \alpha_{10}$ such that

$$\sum_{i=1}^{10} \alpha_i w_i w_i^T = ww^T. \tag{A.7}$$

Note that w is homogeneous coordinate which means the 4th colum of ww^T is w, so there is also

$$\sum_{i=1}^{10} \alpha_i w_i = w. \tag{A.8}$$

When we want to recover the uncertainty of parameters, specific w's can be chosen. For the center (c_1, c_2, c_3), choose $w_c = [0, 0, 0, 1]^T$, so that

$$E[\Phi(Y) w_c w_c^T \Phi(Y)] = E\left[\begin{bmatrix} c_1^2 & 0 & 0 \\ 0 & c_2^2 & 0 \\ 0 & 0 & c_3^2 \end{bmatrix}\right], \tag{A.9}$$

gives the variance of the center parameters. For the length l, choose $w_l = [1, 0, 0, 0]^T$, so that

$$\begin{aligned} E[l^2] &= trace\left(E[\Phi(Y) w_l w_l^T \Phi(Y)]\right) \\ &= trace\left(E\left[\begin{bmatrix} l^2\cos^2(r_y) & 0 & 0 \\ 0 & 0 & 0 \\ 0 & 0 & l^2\sin^2(r_y) \end{bmatrix}\right]\right), \end{aligned} \tag{A.10}$$

and for the width w, choose $w_w = [0, 0, 1, 0]^T$ so that

$$\begin{aligned} E[w^2] &= trace\left(E[\Phi(Y) w_w w_w^T \Phi(Y)]\right) \\ &= trace\left(E\left[\begin{bmatrix} w^2\sin^2(r_y) & 0 & 0 \\ 0 & 0 & 0 \\ 0 & 0 & w^2\cos^2(r_y) \end{bmatrix}\right]\right), \end{aligned} \tag{A.11}$$

and for the height h, choose $w_h = [0, 1, 0, 0]^T$ so that

$$\begin{aligned} E[h^2] &= trace\left(E[\Phi(Y) w_h w_h^T \Phi(Y)]\right) \\ &= trace\left(E\left[\begin{bmatrix} 0 & 0 & 0 \\ 0 & h^2 & 0 \\ 0 & 0 & 0 \end{bmatrix}\right]\right), \end{aligned} \tag{A.12}$$

Bibliography

[1] Cesar Cadena et al. "Past, present, and future of simultaneous localization and mapping: Toward the robust-perception age". In: *IEEE Transactions on robotics* 32.6 (2016), pp. 1309–1332.

[2] Yuichi Motai and Akio Kosaka. "Hand-eye calibration applied to viewpoint selection for robotic vision". In: *IEEE Transactions on Industrial Electronics* 55.10 (2008), pp. 3731–3741.

[3] Ping Liang, Yuh-Lin Chang, and Susan Hackwood. "Adaptive self-calibration of vision-based robot systems". In: *IEEE transactions on systems, man, and cybernetics* 19.4 (1989), pp. 811–824.

[4] Agostino Martinelli, Davide Scaramuzza, and Roland Siegwart. "Automatic self calibration of a vision system during robot motion". In: *2006 IEEE International Conference on Robotics and Automation (ICRA)*. IEEE. 2006, pp. 43–48.

[5] Hanqi Zhuang, Kuanchih Wang, and Zvi S Roth. "Simultaneous calibration of a robot and a hand-mounted camera". In: *IEEE Transactions on Robotics and Automation* 11.5 (1995), pp. 649–660.

[6] Yan Meng and Hanqi Zhuang. "Autonomous robot calibration using vision technology". In: *Robotics and Computer-Integrated Manufacturing* 23.4 (2007), pp. 436–446.

[7] Wei Li et al. "Simultaneous robot–world and hand–eye calibration without a calibration object". In: *Sensors* 18.11 (2018), p. 3949.

[8] Mechanical Systems Control (MSC) lab. *Data-collection vehicle*. `https://msc.berkeley.edu/research/autonomous-vehicle.html`. (Visited on 2020).

[9] John Schulman et al. "Finding Locally Optimal, Collision-Free Trajectories with Sequential Convex Optimization." In: *Robotics: science and systems*. Vol. 9. 1. Citeseer. 2013, pp. 1–10.

[10] Yan Duan et al. "Planning locally optimal, curvature-constrained trajectories in 3D using sequential convex optimization". In: *2014 IEEE International Conference on Robotics and Automation (ICRA)*. IEEE. 2014, pp. 5889–5895.

[11] Nathan Ratliff et al. "CHOMP: Gradient optimization techniques for efficient motion planning". In: *2009 IEEE International Conference on Robotics and Automation*. IEEE. 2009, pp. 489–494.

[12] Zvi Shiller and Steven Dubowsky. "Global time optimal motions of robotic manipulators in the presence of obstacles". In: *Proceedings. 1988 IEEE International Conference on Robotics and Automation*. IEEE. 1988, pp. 370–375.

[13] Kang Shin and Neil McKay. "Minimum-time control of robotic manipulators with geometric path constraints". In: *IEEE Transactions on Automatic Control* 30.6 (1985), pp. 531–541.

[14] Xiaowen Yu et al. "Trajectory planning for robot manipulators considering kinematic constraints using probabilistic roadmap approach". In: *Journal of Dynamic Systems, Measurement, and Control* 139.2 (2017).

[15] Steven M LaValle. *Planning algorithms: Chapter 2.2*. Cambridge university press, 2006.

[16] Steven M LaValle. *Planning algorithms: Chapter 2.2*. Cambridge university press, 2006.

[17] Sandip Aine et al. "Multi-heuristic a". In: *The International Journal of Robotics Research* 35.1-3 (2016), pp. 224–243.

[18] Sertac Karaman and Emilio Frazzoli. "Incremental sampling-based algorithms for optimal motion planning". In: *Robotics Science and Systems VI* 104.2 (2010).

[19] Dustin J Webb and Jur Van Den Berg. "Kinodynamic RRT*: Asymptotically optimal motion planning for robots with linear dynamics". In: *2013 IEEE International Conference on Robotics and Automation*. IEEE. 2013, pp. 5054–5061.

[20] Jeong hwan Jeon, Sertac Karaman, and Emilio Frazzoli. "Anytime computation of time-optimal off-road vehicle maneuvers using the RRT". In: *2011 50th IEEE Conference on Decision and Control and European Control Conference*. IEEE. 2011, pp. 3276–3282.

[21] Weiwei Li and Emanuel Todorov. "Iterative linear quadratic regulator design for nonlinear biological movement systems." In: *ICINCO (1)*. 2004, pp. 222–229.

[22] Emanuel Todorov and Weiwei Li. "A generalized iterative LQG method for locally-optimal feedback control of constrained nonlinear stochastic systems". In: *Proceedings of the 2005, American Control Conference, 2005.* IEEE. 2005, pp. 300–306.

[23] Sonja Macfarlane and Elizabeth A Croft. "Jerk-bounded manipulator trajectory planning: design for real-time applications". In: *IEEE Transactions on robotics and automation* 19.1 (2003), pp. 42–52.

[24] Nicholas M Patrikalakis and Takashi Maekawa. *Shape interrogation for computer aided design and manufacturing*. Springer Science & Business Media, 2009.

[25] Florian T Pokorny et al. "High-dimensional winding-augmented motion planning with 2d topological task projections and persistent homology". In: *2016 IEEE International Conference on Robotics and Automation (ICRA)*. IEEE. 2016, pp. 24–31.

[26] Zining Wang, Cong Wang, and Masayoshi Tomizuka. "Vibration cancellation of semiconductor manufacturing robots". In: *Manufacturing Letters* 4 (2015), pp. 6–9.

[27] NA Bruinsma and M Steinbuch. "A fast algorithm to compute the H_∞-norm of a transfer function matrix". In: *Systems & Control Letters* 14.4 (1990), pp. 287–293.

[28] Pierre Apkarian and Dominikus Noll. "Nonsmooth H_∞ synthesis". In: *IEEE Transactions on Automatic Control* 51.1 (2006), pp. 71–86.

[29] Richard Szeliski. *Computer vision: algorithms and applications*. Springer Science & Business Media, 2010.

[30] Li Liu et al. "Deep learning for generic object detection: A survey". In: *International journal of computer vision* 128.2 (2020), pp. 261–318.

[31] Richard Szeliski. "Computer vision: algorithms and applications". In: Springer Science & Business Media, 2010. Chap. Feature detection and matching, pp. 207–208.

[32] Yali Li et al. "A survey of recent advances in visual feature detection". In: *Neurocomputing* 149 (2015), pp. 736–751.

[33] Mitra Basu. "Gaussian-based edge-detection methods-a survey". In: *IEEE Transactions on Systems, Man, and Cybernetics, Part C (Applications and Reviews)* 32.3 (2002), pp. 252–260.

[34] Giuseppe Papari and Nicolai Petkov. "Edge and line oriented contour detection: State of the art". In: *Image and Vision Computing* 29.2-3 (2011), pp. 79–103.

[35] Krystian Mikolajczyk and Cordelia Schmid. "A performance evaluation of local descriptors". In: *IEEE transactions on pattern analysis and machine intelligence* 27.10 (2005), pp. 1615–1630.

[36] Dilip K Prasad, Maylor KH Leung, and Chai Quek. "ElliFit: An unconstrained, non-iterative, least squares based geometric Ellipse Fitting method". In: *Pattern Recognition* 46.5 (2013), pp. 1449–1465.

[37] Qi Jia et al. "A fast ellipse detector using projective invariant pruning". In: *IEEE Transactions on Image Processing* 26.8 (2017), pp. 3665–3679.

[38] Wei Lu and Jinglu Tan. "Detection of incomplete ellipse in images with strong noise by iterative randomized Hough transform (IRHT)". In: *Pattern Recognition* 41.4 (2008), pp. 1268–1279.

[39] Yonghong Xie and Qiang Ji. "A new efficient ellipse detection method". In: *Object recognition supported by user interaction for service robots*. Vol. 2. IEEE. 2002, pp. 957–960.

[40] Michele Fornaciari, Andrea Prati, and Rita Cucchiara. "A fast and effective ellipse detector for embedded vision applications". In: *Pattern Recognition* 47.11 (2014), pp. 3693–3708.

[41] Zining Wang and Masayoshi Tomizuka. "Precise 3D Calibration of Wafer Handling Robot by Visual Detection and Tracking of Elliptic-shape Wafers". In: *2020 IEEE International Conference on Robotics and Automation (ICRA)*. IEEE. 2020, pp. 4977–4982.

[42] Remco C Veltkamp and Michiel Hagedoorn. "State of the art in shape matching". In: *Principles of visual information retrieval*. Springer, 2001, pp. 87–119.

[43] Serge Belongie, Jitendra Malik, and Jan Puzicha. "Shape matching and object recognition using shape contexts". In: *IEEE transactions on pattern analysis and machine intelligence* 24.4 (2002), pp. 509–522.

[44] Dana H Ballard. "Generalizing the Hough transform to detect arbitrary shapes". In: *Pattern recognition* 13.2 (1981), pp. 111–122.

[45] Markus Ulrich, Carsten Steger, and Albert Baumgartner. "Real-time object recognition using a modified generalized Hough transform". In: *Pattern Recognition* 36.11 (2003), pp. 2557–2570.

[46] Martin A Fischler and Robert C Bolles. "Random sample consensus: a paradigm for model fitting with applications to image analysis and automated cartography". In: *Communications of the ACM* 24.6 (1981), pp. 381–395.

[47] Zhuowen Tu and Alan L Yuille. "Shape matching and recognition–using generative models and informative features". In: *European Conference on Computer Vision*. Springer. 2004, pp. 195–209.

[48] Yulan Guo et al. "3D object recognition in cluttered scenes with local surface features: a survey". In: *IEEE Transactions on Pattern Analysis and Machine Intelligence* 36.11 (2014), pp. 2270–2287.

[49] Yaron Lipman and Thomas Funkhouser. "Möbius voting for surface correspondence". In: *ACM Transactions on Graphics (TOG)* 28.3 (2009), p. 72.

[50] Gary KL Tam et al. "Registration of 3D point clouds and meshes: a survey from rigid to nonrigid." In: *IEEE transactions on visualization and computer graphics* 19.7 (2013), pp. 1199–1217.

[51] Jiayi Ma, Ji Zhao, and Alan L Yuille. "Non-rigid point set registration by preserving global and local structures". In: *IEEE Transactions on image Processing* 25.1 (2016), pp. 53–64.

[52] Lifei Bai, Xianqiang Yang, and Huijun Gao. "Nonrigid Point Set Registration by Preserving Local Connectivity". In: *IEEE transactions on cybernetics* 48.3 (2018), pp. 826–835.

[53] Jiayi Ma et al. "Robust L2E Estimation of Transformation for Non-Rigid Registration." In: *IEEE Trans. Signal Processing* 63.5 (2015), pp. 1115–1129.

[54] Liang Li et al. "Robust Point Set Registration Using Signature Quadratic Form Distance". In: *IEEE transactions on cybernetics* 99 (2018).

[55] Jiaqi Yang, Zhiguo Cao, and Qian Zhang. "A fast and robust local descriptor for 3D point cloud registration". In: *Information Sciences* 346 (2016), pp. 163–179.

[56] Anders Glent Buch et al. "Pose estimation using local structure-specific shape and appearance context". In: *Robotics and Automation (ICRA), 2013 IEEE International Conference on*. IEEE. 2013, pp. 2080–2087.

[57] Andy Zeng et al. "3dmatch: Learning local geometric descriptors from rgb-d reconstructions". In: *Computer Vision and Pattern Recognition (CVPR), 2017 IEEE Conference on*. IEEE. 2017, pp. 199–208.

[58] Yulan Guo et al. "A comprehensive performance evaluation of 3D local feature descriptors". In: *International Journal of Computer Vision* 116.1 (2016), pp. 66–89.

[59] Jana Košecká and Wei Zhang. "Extraction, matching, and pose recovery based on dominant rectangular structures". In: *Computer Vision and Image Understanding* 100.3 (2005), pp. 274–293.

[60] Jiri Matas, Charles Galambos, and Josef Kittler. "Robust detection of lines using the progressive probabilistic hough transform". In: *Computer vision and image understanding* 78.1 (2000), pp. 119–137.

[61] HK Yuen et al. "Comparative study of Hough transform methods for circle finding." In: *Image and vision computing* 8.1 (1990), pp. 71–77.

[62] Michele Fornaciari and Andrea Prati. "Very fast ellipse detection for embedded vision applications". In: *2012 Sixth International Conference on Distributed Smart Cameras (ICDSC)*. IEEE. 2012, pp. 1–6.

[63] James O Street, Raymond J Carroll, and David Ruppert. "A note on computing robust regression estimates via iteratively reweighted least squares". In: *The American Statistician* 42.2 (1988), pp. 152–154.

[64] Haili Chui. "Non-rigid point matching: algorithms, extensions and applications". In: (2001).

[65] Andriy Myronenko and Xubo Song. "Point set registration: Coherent point drift". In: *IEEE transactions on pattern analysis and machine intelligence* 32.12 (2010), pp. 2262–2275.

[66] Jiayi Ma, Ji Zhao, and Alan L Yuille. "Non-rigid point set registration by preserving global and local structures". In: *IEEE Transactions on image Processing* 25.1 (2015), pp. 53–64.

[67] Lifei Bai, Xianqiang Yang, and Huijun Gao. "Nonrigid point set registration by preserving local connectivity". In: *IEEE transactions on cybernetics* 48.3 (2017), pp. 826–835.

[68] Te Tang and Masayoshi Tomizuka. "Track deformable objects from point clouds with structure preserved registration". In: *The International Journal of Robotics Research* (2018), p. 0278364919841431.

[69] Jan Knopp et al. "Hough transform and 3D SURF for robust three dimensional classification". In: *European Conference on Computer Vision*. Springer. 2010, pp. 589–602.

[70] Federico Tombari, Samuele Salti, and Luigi Di Stefano. "Unique signatures of histograms for local surface description". In: *European conference on computer vision*. Springer. 2010, pp. 356–369.

[71] Yulan Guo et al. "Rotational projection statistics for 3D local surface description and object recognition". In: *International journal of computer vision* 105.1 (2013), pp. 63–86.

[72] Yulan Guo et al. "An accurate and robust range image registration algorithm for 3D object modeling". In: *ieee transactions on multimedia* 16.5 (2014), pp. 1377–1390.

[73] Terry A McKee and Fred R McMorris. *Topics in intersection graph theory*. SIAM, 1999.

[74] Bertram Drost et al. "Model globally, match locally: Efficient and robust 3D object recognition". In: *2010 IEEE computer society conference on computer vision and pattern recognition*. Ieee. 2010, pp. 998–1005.

[75] Ajmal Mian, Mohammed Bennamoun, and Robyn Owens. "On the repeatability and quality of keypoints for local feature-based 3d object retrieval from cluttered scenes". In: *International Journal of Computer Vision* 89.2-3 (2010), pp. 348–361.

[76] Oliver Van Kaick et al. "A survey on shape correspondence". In: *Computer Graphics Forum*. Vol. 30. 6. Wiley Online Library. 2011, pp. 1681–1707.

[77] Wenhan Luo et al. "Multiple object tracking: A literature review". In: *arXiv preprint arXiv:1409.7618* (2014).

[78] Litong Fan et al. "A survey on multiple object tracking algorithm". In: *2016 IEEE International Conference on Information and Automation (ICIA)*. IEEE. 2016, pp. 1855–1862.

[79] Yingkun Xu et al. "Deep learning for multiple object tracking: a survey". In: *IET Computer Vision* 13.4 (2019), pp. 355–368.

[80] Gioele Ciaparrone et al. "Deep learning in video multi-object tracking: A survey". In: *Neurocomputing* 381 (2020), pp. 61–88.

[81] Laura Leal-Taixé et al. "Tracking the trackers: an analysis of the state of the art in multiple object tracking". In: *arXiv preprint arXiv:1704.02781* (2017).

[82] Jürgen Biber et al. *Robust real-time tracking of ellipse arcs*. na, 2001.

[83] Ion Martinikorena et al. "Fast and robust ellipse detection algorithm for head-mounted eye tracking systems". In: *Machine Vision and Applications* 29.5 (2018), pp. 845–860.

[84] Nicola Greggio et al. "Real-time Ellipse Fitting, 3D Spherical Object Localization, and Tracking for the iCub Simulator." In: *ICINCO (2)*. 2011, pp. 248–256.

[85] Cai Meng, Zhan Hu, and HongChao Sun. "An ellipse feature tracking method based on the Kalman filter". In: *2015 8th International Congress on Image and Signal Processing (CISP)*. IEEE. 2015, pp. 882–887.

[86] Pablo G Tahoces et al. "Automatic estimation of the aortic lumen geometry by ellipse tracking". In: *International journal of computer assisted radiology and surgery* 14.2 (2019), pp. 345–355.

[87] Karl Granstrom and Umut Orguner. "A PHD filter for tracking multiple extended targets using random matrices". In: *IEEE Transactions on Signal Processing* 60.11 (2012), pp. 5657–5671.

[88] Shishan Yang, Marcus Baum, and Karl Granström. "Metrics for performance evaluation of elliptic extended object tracking methods". In: *2016 IEEE International Conference on Multisensor Fusion and Integration for Intelligent Systems (MFI)*. IEEE. 2016, pp. 523–528.

[89] Philippe Weinzaepfel et al. "DeepFlow: Large displacement optical flow with deep matching". In: *Proceedings of the IEEE international conference on computer vision*. 2013, pp. 1385–1392.

[90] Alex Teichman, Jesse Levinson, and Sebastian Thrun. "Towards 3D object recognition via classification of arbitrary object tracks". In: *2011 IEEE International Conference on Robotics and Automation*. IEEE. 2011, pp. 4034–4041.

[91] Liang Li et al. "Rigid point set registration based on cubature Kalman filter and its application in intelligent vehicles". In: *IEEE Transactions on Intelligent Transportation Systems* 19.6 (2017), pp. 1754–1765.

[92] Andriy Myronenko, Xubo Song, and Miguel A Carreira-Perpinán. "Non-rigid point set registration: Coherent point drift". In: *Advances in neural information processing systems*. 2007, pp. 1009–1016.

[93] François Bourgeois and Jean-Claude Lassalle. "An extension of the Munkres algorithm for the assignment problem to rectangular matrices". In: *Communications of the ACM* 14.12 (1971), pp. 802–804.

[94] Chieh-Chih Wang et al. "Simultaneous localization, mapping and moving object tracking". In: *The International Journal of Robotics Research* 26.9 (2007), pp. 889–916.

[95] Jose A Castellanos et al. "The SPmap: A probabilistic framework for simultaneous localization and map building". In: *IEEE Transactions on robotics and Automation* 15.5 (1999), pp. 948–952.

[96] Dirk Hahnel et al. "An efficient FastSLAM algorithm for generating maps of large-scale cyclic environments from raw laser range measurements". In: *Proceedings 2003 IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS 2003)*. Vol. 1. IEEE. 2003, pp. 206–211.

[97] Sebastian Thrun et al. "Simultaneous localization and mapping with sparse extended information filters". In: *The international journal of robotics research* 23.7-8 (2004), pp. 693–716.

[98] Giorgio Grisetti et al. "A tutorial on graph-based SLAM". In: *IEEE Intelligent Transportation Systems Magazine* 2.4 (2010), pp. 31–43.

[99] Giorgio Grisetti et al. "g2o: A general framework for (hyper) graph optimization". In: *Proceedings of the IEEE International Conference on Robotics and Automation (ICRA), Shanghai, China*. 2011, pp. 9–13.

[100] Sameer Agarwal, Keir Mierle, et al. "Ceres solver". In: (2012).

[101] Raul Mur-Artal and Juan D Tardós. "Orb-slam2: An open-source slam system for monocular, stereo, and rgb-d cameras". In: *IEEE Transactions on Robotics* 33.5 (2017), pp. 1255–1262.

[102] Stefan Leutenegger et al. "Keyframe-based visual–inertial odometry using nonlinear optimization". In: *The International Journal of Robotics Research* 34.3 (2015), pp. 314–334.

[103] Muhamad Risqi U Saputra, Andrew Markham, and Niki Trigoni. "Visual SLAM and structure from motion in dynamic environments: A survey". In: *ACM Computing Surveys (CSUR)* 51.2 (2018), pp. 1–36.

[104] Paschalis Panteleris and Antonis A Argyros. "Vision-based SLAM and moving objects tracking for the perceptual support of a smart walker platform". In: *European Conference on Computer Vision*. Springer. 2014, pp. 407–423.

[105] David Márquez-Gámez and Michel Devy. "Active visual-based detection and tracking of moving objects from clustering and classification methods". In: *International Conference on Advanced Concepts for Intelligent Vision Systems*. Springer. 2012, pp. 361–373.

[106] Martin J Wainwright and Michael Irwin Jordan. *Graphical models, exponential families, and variational inference*. Now Publishers Inc, 2008.

[107] Mike Tao Zhang and Ken Goldberg. "Calibration of wafer handling robots: A fixturing approach". In: *2007 IEEE International Conference on Automation Science and Engineering*. IEEE. 2007, pp. 255–260.

[108] Chengyi Yu, Xiaobo Chen, and Juntong Xi. "Determination of optimal measurement configurations for self-calibrating a robotic visual inspection system with multiple point constraints". In: *The International Journal of Advanced Manufacturing Technology* 96.9-12 (2018), pp. 3365–3375.

[109] Weidong Wang et al. "A universal index and an improved PSO algorithm for optimal pose selection in kinematic calibration of a novel surgical robot". In: *Robotics and computer-integrated manufacturing* 50 (2018), pp. 90–101.

[110] J Dupuis, C Holst, and H Kuhlmann. "Improving the kinematic calibration of a coordinate measuring arm using configuration analysis". In: *Precision Engineering* 50 (2017), pp. 171–182.

[111] Jose Maurıcio ST Motta, Guilherme C De Carvalho, and RS McMaster. "Robot calibration using a 3D vision-based measurement system with a single camera". In: *Robotics and Computer-Integrated Manufacturing* 17 (2001), pp. 487–497.

[112] Joan Sola. "Quaternion kinematics for the error-state Kalman filter". In: *arXiv preprint arXiv:1711.02508* (2017).

[113] Shengyong Chen et al. *Active sensor planning for multiview vision tasks*. Vol. 1. Springer, 2008.

[114] Bernard Schmidt and Lihui Wang. "Automatic work objects calibration via a global–local camera system". In: *Robotics and Computer-Integrated Manufacturing* 30 (2014), pp. 678–683.

[115] Albert Nubiola et al. "Comparison of two calibration methods for a small industrial robot based on an optical CMM and a laser tracker". In: *Robotica* 32.3 (2014), p. 447.

[116] Jin-Hwan Borm and Chia-Hsiang Meng. "Determination of optimal measurement configurations for robot calibration based on observability measure". In: *The International Journal of Robotics Research* 10.1 (1991), pp. 51–63.

[117] Morris R Driels and Uday S Pathre. "Significance of observation strategy on the design of robot calibration experiments". In: *Journal of robotic systems* 7.2 (1990), pp. 197–223.

[118] Ali Nahvi, John M Hollerbach, and Vincent Hayward. "Calibration of a parallel robot using multiple kinematic closed loops". In: *Proceedings of the 1994 IEEE International Conference on Robotics and Automation*. IEEE. 1994, pp. 407–412.

[119] Ali Nahvi and John M Hollerbach. "The noise amplification index for optimal pose selection in robot calibration". In: *Proceedings of IEEE international conference on robotics and automation*. Vol. 1. IEEE. 1996, pp. 647–654.

[120] Albert Nubiola and Ilian A Bonev. "Absolute calibration of an ABB IRB 1600 robot using a laser tracker". In: *Robotics and Computer-Integrated Manufacturing* 29.1 (2013), pp. 236–245.

[121] Valerii Fedorov. "Optimal experimental design". In: *Wiley Interdisciplinary Reviews: Computational Statistics* 2.5 (2010), pp. 581–589.

[122] Yu Sun and John M Hollerbach. "Observability index selection for robot calibration". In: *2008 IEEE international conference on robotics and automation*. IEEE. 2008, pp. 831–836.

[123] Ahmed Joubair and Ilian A Bonev. "Comparison of the efficiency of five observability indices for robot calibration". In: *Mechanism and Machine Theory* 70 (2013), pp. 254–265.

[124] Themistoklis C Xygkis, George N Korres, and Nikolaos M Manousakis. "Fisher information based meter placement in distribution grids via the D-optimal experimental design". In: *IEEE Transactions on Smart Grid* 9.2 (2016), pp. 1452–1461.

[125] Toby J Mitchell. "An algorithm for the construction of "D-optimal" experimental designs". In: *Technometrics* 42.1 (2000), pp. 48–54.

[126] William DuMouchel and Bradley Jones. "A simple Bayesian modification of D-optimal designs to reduce dependence on an assumed model". In: *Technometrics* 36.1 (1994), pp. 37–47.

[127] Yu Sun and John M Hollerbach. "Active robot calibration algorithm". In: *2008 IEEE International Conference on Robotics and Automation*. IEEE. 2008, pp. 1276–1281.

[128] Tian Li et al. "Optimal measurement configurations for kinematic calibration of six-DOF serial robot". In: *Journal of Central South University of Technology* 18.3 (2011), pp. 618–626.

[129] Pierre Renaud et al. "Optimal pose selection for vision-based kinematic calibration of parallel mechanisms". In: *Proceedings 2003 IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS 2003)(Cat. No. 03CH37453)*. Vol. 3. IEEE. 2003, pp. 2223–2228.

[130] David Daney, Yves Papegay, and Blaise Madeline. "Choosing measurement poses for robot calibration with the local convergence method and Tabu search". In: *The International Journal of Robotics Research* 24.6 (2005), pp. 501–518.

[131] Kirstine Smith. "On the standard deviations of adjusted and interpolated values of an observed polynomial function and its constants and the guidance they give towards a proper choice of the distribution of observations". In: *Biometrika* 12.1/2 (1918), pp. 1–85.

[132] Anthony Atkinson, Alexander Donev, Randall Tobias, et al. *Optimum experimental designs, with SAS*. Vol. 34. Oxford University Press, 2007.

[133] WW Shang and Shuang Cong. "Exciting trajectories design for the dynamic identification of parallel manipulators". In: *IFAC Proceedings Volumes* 44.1 (2011), pp. 10287–10292.

[134] Jingfu Jin and Nicholas Gans. "Parameter identification for industrial robots with a fast and robust trajectory design approach". In: *Robotics and Computer-Integrated Manufacturing* 31 (2015), pp. 21–29.

[135] G Calafiore, Marina Indri, and Basilio Bona. "Robot dynamic calibration: Optimal excitation trajectories and experimental parameter estimation". In: *Journal of robotic systems* 18.2 (2001), pp. 55–68.

[136] Jun Wu, Jinsong Wang, and Zheng You. "An overview of dynamic parameter identification of robots". In: *Robotics and computer-integrated manufacturing* 26.5 (2010), pp. 414–419.

[137] Jian Zhou, Hee-Jun Kang, and Young-Shick Ro. "Comparison of the observability indices for robot calibration considering joint stiffness parameters". In: *International Conference on Intelligent Computing*. Springer. 2010, pp. 372–380.

[138] Alexandr Klimchik et al. "Optimal selection of measurement configurations for stiffness model calibration of anthropomorphic manipulators". In: *Applied Mechanics and Materials*. Vol. 162. Trans Tech Publ. 2012, pp. 161–170.

[139] Lyudmila Mihaylova et al. "A comparison of decision making criteria and optimization methods for active robotic sensing". In: *International Conference on Numerical Methods and Applications*. Springer. 2002, pp. 316–324.

[140] Hui Yu, Hong Yue, and Peter Halling. "A two-loop optimization strategy for multi-objective optimal experimental design". In: *IFAC-PapersOnLine* 49.7 (2016), pp. 803–808.

[141] Lyudmila Mihaylova, Joris De Schutter, and Herman Bruyninckx. "A multisine approach for trajectory optimization based on information gain". In: *Robotics and Autonomous Systems* 43.4 (2003), pp. 231–243.

[142] Graziano Chesi and Yeung Sam Hung. "Global path-planning for constrained and optimal visual servoing". In: *IEEE Transactions on Robotics* 23.5 (2007), pp. 1050–1060.

[143] Kostantinos Tarabanis, Roger Y Tsai, and Peter K Allen. "Automated sensor planning for robotic vision tasks". In: (1991).

[144] Traian Abrudan, Jan Eriksson, and Visa Koivunen. "Conjugate gradient algorithm for optimization under unitary matrix constraint". In: *Signal Processing* 89.9 (2009), pp. 1704–1714.

[145] Martin Engelcke et al. "Vote3deep: Fast object detection in 3d point clouds using efficient convolutional neural networks". In: *2017 IEEE International Conference on Robotics and Automation (ICRA)*. IEEE. 2017, pp. 1355–1361.

[146] Charles R Qi et al. "Frustum pointnets for 3d object detection from rgb-d data". In: *Proceedings of the IEEE conference on computer vision and pattern recognition*. 2018, pp. 918–927.

[147] Kiwoo Shin, Youngwook Paul Kwon, and Masayoshi Tomizuka. "Roarnet: A robust 3d object detection based on region approximation refinement". In: *2019 IEEE Intelligent Vehicles Symposium (IV)*. IEEE. 2019, pp. 2510–2515.

[148] Kiwoo Shin and Masayoshi Tomizuka. "Improving a Quality of 3D Object Detection by Spatial Transformation Mechanism". In: *arXiv preprint arXiv:1910.04853* (2019).

[149] Zhixin Wang and Kui Jia. "Frustum convnet: Sliding frustums to aggregate local point-wise features for amodal. In 2019 IEEE". In: *RSJ International Conference on Intelligent Robots and Systems (IROS)*, pp. 1742–1749.

[150] Danfei Xu, Dragomir Anguelov, and Ashesh Jain. "Pointfusion: Deep sensor fusion for 3d bounding box estimation". In: *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*. 2018, pp. 244–253.

[151] Markus Braun et al. "Pose-rcnn: Joint object detection and pose estimation using 3d object proposals". In: *2016 IEEE 19th International Conference on Intelligent Transportation Systems (ITSC)*. IEEE. 2016, pp. 1546–1551.

[152] Stefan Lange, Fritz Ulbrich, and Daniel Goehring. "Online vehicle detection using deep neural networks and lidar based preselected image patches". In: *2016 IEEE Intelligent Vehicles Symposium (IV)*. IEEE. 2016, pp. 954–959.

[153] Yu Xiang et al. "Data-driven 3d voxel patterns for object category recognition". In: *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*. 2015, pp. 1903–1911.

[154] Xiaozhi Chen et al. "Multi-view 3d object detection network for autonomous driving". In: *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*. 2017, pp. 1907–1915.

[155] Jason Ku et al. "Joint 3d proposal generation and object detection from view aggregation". In: *2018 IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS)*. IEEE. 2018, pp. 1–8.

[156] Ming Liang et al. "Multi-task multi-sensor fusion for 3d object detection". In: *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*. 2019, pp. 7345–7353.

[157] Jin Hyeok Yoo et al. "3D-CVF: Generating Joint Camera and LiDAR Features Using Cross-View Spatial Feature Fusion for 3D Object Detection". In: *arXiv preprint arXiv:2004.12636* (2020).

[158] Taewan Kim and Joydeep Ghosh. "Robust detection of non-motorized road users using deep learning on optical and lidar data". In: *2016 IEEE 19th International Conference on Intelligent Transportation Systems (ITSC)*. IEEE. 2016, pp. 271–276.

[159] Joel Schlosser, Christopher K Chow, and Zsolt Kira. "Fusing lidar and images for pedestrian detection using convolutional neural networks". In: *2016 IEEE International Conference on Robotics and Automation (ICRA)*. IEEE. 2016, pp. 2198–2205.

[160] Zining Wang, Wei Zhan, and Masayoshi Tomizuka. "Fusing bird's eye view lidar point cloud and front view camera image for 3d object detection". In: *2018 IEEE Intelligent Vehicles Symposium (IV)*. IEEE. 2018, pp. 1–6.

[161] Andreas Geiger, Philip Lenz, and Raquel Urtasun. "Are we ready for autonomous driving? the kitti vision benchmark suite". In: *Computer Vision and Pattern Recognition (CVPR), 2012 IEEE Conference on*. IEEE. 2012, pp. 3354–3361.

[162] Zhaowei Cai et al. "A unified multi-scale deep convolutional neural network for fast object detection". In: *European Conference on Computer Vision*. Springer. 2016, pp. 354–370.

[163] Jimmy Ren et al. "Accurate single stage detector using recurrent rolling convolution". In: *Proceedings of the IEEE conference on computer vision and pattern recognition*. 2017, pp. 5420–5428.

[164] Fan Yang, Wongun Choi, and Yuanqing Lin. "Exploit all the layers: Fast and accurate cnn object detector with scale dependent pooling and cascaded rejection classifiers". In: *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*. 2016, pp. 2129–2137.

[165] Xiaozhi Chen et al. "Monocular 3d object detection for autonomous driving". In: *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*. 2016, pp. 2147–2156.

[166] Arsalan Mousavian et al. "3D Bounding Box Estimation Using Deep Learning and Geometry". In: *arXiv preprint arXiv:1612.00496* (2016).

[167] Florian Chabot et al. "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*. 2017, pp. 2040–2049.

[168] Xiaozhi Chen et al. "3d object proposals for accurate object class detection". In: *Advances in Neural Information Processing Systems*. 2015, pp. 424–432.

[169] Yan Wang et al. "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*. 2019, pp. 8445–8453.

[170] Yongjian Chen et al. "MonoPair: Monocular 3D Object Detection Using Pairwise Spatial Relationships". In: *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*. 2020, pp. 12093–12102.

[171] Yin Zhou and Oncel Tuzel. "Voxelnet: End-to-end learning for point cloud based 3d object detection". In: *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*. 2018, pp. 4490–4499.

[172] Yan Yan, Yuxing Mao, and Bo Li. "Second: Sparsely embedded convolutional detection". In: *Sensors* 18.10 (2018), p. 3337.

[173] Alex H Lang et al. "Pointpillars: Fast encoders for object detection from point clouds". In: *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*. 2019, pp. 12697–12705.

[174] Guojun Wang et al. "CenterNet3D: An Anchor free Object Detector for Autonomous Driving". In: *arXiv preprint arXiv:2007.07214* (2020).

[175] A. Garcia-Garcia et al. "PointNet: A 3D Convolutional Neural Network for real-time object class recognition". In: *International Joint Conference on Neural Networks*. 2016, pp. 1578–1584.

[176] Charles Ruizhongtai Qi et al. "Pointnet++: Deep hierarchical feature learning on point sets in a metric space". In: *Advances in neural information processing systems*. 2017, pp. 5099–5108.

[177] Zetong Yang et al. "3dssd: Point-based 3d single stage object detector". In: *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*. 2020, pp. 11040–11048.

[178] Dingfu Zhou et al. "Joint 3D Instance Segmentation and Object Detection for Autonomous Driving". In: *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*. 2020, pp. 1839–1849.

[179] Zetong Yang et al. "STD: Sparse-to-dense 3d object detector for point cloud". In: *Proc. IEEE Conf. Computer Vision and Pattern Recognition*. 2019, pp. 1951–1960.

[180] Bo Li, Tianlei Zhang, and Tian Xia. "Vehicle detection from 3d lidar using fully convolutional network". In: *arXiv preprint arXiv:1608.07916* (2016).

[181] Bichen Wu et al. "SqueezeSeg: Convolutional Neural Nets with Recurrent CRF for Real-Time Road-Object Segmentation from 3D LiDAR Point Cloud". In: *arXiv preprint arXiv:1710.07368* (2017).

[182] Bichen Wu et al. "Squeezesegv2: Improved model structure and unsupervised domain adaptation for road-object segmentation from a lidar point cloud". In: *2019 International Conference on Robotics and Automation (ICRA)*. IEEE. 2019, pp. 4376–4382.

[183] Chenfeng Xu et al. "Squeezesegv3: Spatially-adaptive convolution for efficient point-cloud segmentation". In: *arXiv preprint arXiv:2004.01803* (2020).

[184] Yilun Chen et al. "Fast point r-cnn". In: *Proceedings of the IEEE International Conference on Computer Vision*. 2019, pp. 9775–9784.

[185] Shaoshuai Shi et al. "Pv-rcnn: Point-voxel feature set abstraction for 3d object detection". In: *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*. 2020, pp. 10529–10538.

[186] Prarthana Bhattacharyya and Krzysztof Czarnecki. "Deformable PV-RCNN: Improving 3D Object Detection with Learned Deformations". In: *arXiv preprint arXiv:2008.08766* (2020).

[187] Chenhang He et al. "Structure Aware Single-stage 3D Object Detection from Point Cloud". In: *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*. 2020, pp. 11873–11882.

[188] Maosheng Ye, Shuangjie Xu, and Tongyi Cao. "HVNet: Hybrid Voxel Network for LiDAR Based 3D Object Detection". In: *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*. 2020, pp. 1631–1640.

[189] Zhidong Liang et al. "RangeRCNN: Towards Fast and Accurate 3D Object Detection with Range Image Representation". In: *arXiv preprint arXiv:2009.00206* (2020).

[190] Jesus Zarzar, Silvio Giancola, and Bernard Ghanem. "PointRGCN: Graph convolution networks for 3D vehicles detection refinement". In: *arXiv preprint arXiv:1911.12236* (2019).

[191] Weijing Shi and Raj Rajkumar. "Point-gnn: Graph neural network for 3d object detection in a point cloud". In: *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*. 2020, pp. 1711–1719.

[192] Jingdong Wang et al. "Deeply-fused nets". In: *arXiv preprint arXiv:1605.07716* (2016).

[193] Jian Deng and Krzysztof Czarnecki. "MLOD: A multi-view 3D object detection based on robust feature fusion method". In: *2019 IEEE Intelligent Transportation Systems Conference (ITSC)*. IEEE. 2019, pp. 279–284.

[194] Liang Xie et al. "PI-RCNN: An Efficient Multi-Sensor 3D Object Detector with Point-Based Attentive Cont-Conv Fusion Module." In: *AAAI*. 2020, pp. 12460–12467.

[195] Su Pang, Daniel Morris, and Hayder Radha. "CLOCs: Camera-LiDAR Object Candidates Fusion for 3D Object Detection". In: *arXiv preprint arXiv:2009.00784* (2020).

[196] Johannes Lehner et al. "Patch Refinement–Localized 3D Object Detection". In: *arXiv preprint arXiv:1910.04093* (2019).

[197] Yu Xiang et al. "Subcategory-aware convolutional neural networks for object proposals and detection". In: *2017 IEEE winter conference on applications of computer vision (WACV)*. IEEE. 2017, pp. 924–933.

[198] Mark Everingham et al. "The pascal visual object classes challenge: A retrospective". In: *International journal of computer vision* 111.1 (2015), pp. 98–136.

[199] Max Jaderberg, Karen Simonyan, Andrew Zisserman, et al. "Spatial transformer networks". In: *Advances in neural information processing systems*. 2015, pp. 2017–2025.

[200] Xiaozhi Chen et al. "Multi-view 3d object detection network for autonomous driving". In: *arXiv preprint arXiv:1611.07759* (2016).

[201] Tsung-Yi Lin et al. "Focal loss for dense object detection". In: *Proceedings of the IEEE international conference on computer vision*. 2017, pp. 2980–2988.

[202] Wei Liu et al. "Ssd: Single shot multibox detector". In: *European conference on computer vision*. Springer. 2016, pp. 21–37.

[203] Joseph Redmon and Ali Farhadi. "YOLO9000: better, faster, stronger". In: *arXiv preprint arXiv:1612.08242* (2016).

[204] Tsung-Yi Lin et al. "Feature pyramid networks for object detection". In: *arXiv preprint arXiv:1612.03144* (2016).

[205] Jasper RR Uijlings et al. "Selective search for object recognition". In: *International journal of computer vision* 104.2 (2013), pp. 154–171.

[206] Shaoqing Ren et al. "Faster R-CNN: Towards real-time object detection with region proposal networks". In: *Advances in neural information processing systems*. 2015, pp. 91–99.

[207] Frank E Harrell Jr. "Regression modeling strategies: with applications to linear models, logistic and ordinal regression, and survival analysis". In: Springer, 2015. Chap. Prediction vs. Classification, pp. 4–5.

[208] Ross Girshick. "Fast r-cnn". In: *Proceedings of the IEEE international conference on computer vision*. 2015, pp. 1440–1448.

[209] Andreas Geiger, Philip Lenz, and Raquel Urtasun. "Are we ready for Autonomous Driving? The KITTI Vision Benchmark Suite". In: *Proc. IEEE Conf. Computer Vision and Pattern Recognition*. 2012.

[210] Pei Sun et al. "Scalability in Perception for Autonomous Driving: An Open Dataset Benchmark". In: *arXiv preprint arXiv:1912.04838* (2019).

[211] C. Haase-Schütz, H. Hertlein, and W. Wiesbeck. "Estimating Labeling Quality with Deep Object Detectors". In: *IEEE Intelligent Vehicles Symp.* 2019, pp. 33–38.

[212] Sainbayar Sukhbaatar et al. "Training convolutional networks with noisy labels". In: *arXiv preprint arXiv:1406.2080* (2014).

[213] Neil D Lawrence and Bernhard Schölkopf. "Estimating a kernel fisher discriminant in the presence of label noise". In: *International Conference on Machine Learning*. Vol. 1. Citeseer. 2001, pp. 306–313.

[214] Tong Xiao et al. "Learning from massive noisy labeled data for image classification". In: *Proc. IEEE Conf. Computer Vision and Pattern Recognition*. 2015, pp. 2691–2699.

[215] Arash Vahdat. "Toward robustness against label noise in training deep discriminative neural networks". In: *Advances in Neural Information Processing Systems*. 2017, pp. 5596–5605.

[216] Mark Everingham et al. "The pascal visual object classes (VOC) challenge". In: *International journal of computer vision* 88.2 (2010), pp. 303–338.

[217] Kemal Oksuz et al. "Localization recall precision (LRP): A new performance metric for object detection". In: *Proc. Eur. Conf. Computer Vision*. 2018, pp. 504–519.

[218] Di Feng, Lars Rosenbaum, and Klaus Dietmayer. "Towards safe autonomous driving: Capture uncertainty in the deep neural network for lidar 3d vehicle detection". In: *2018 21st International Conference on Intelligent Transportation Systems (ITSC)*. IEEE. 2018, pp. 3266–3273.

[219] David Hall et al. "Probabilistic object detection: Definition and evaluation". In: *arXiv preprint arXiv:1811.10800* (2018).

[220] Benoit Frénay and Michel Verleysen. "Classification in the presence of label noise: a survey". In: *IEEE transactions on neural networks and learning systems* 25.5 (2013), pp. 845–869.

[221] Gregory P Meyer and Niranjan Thakurdesai. "Learning an Uncertainty-Aware Object Detector for Autonomous Driving". In: *arXiv preprint arXiv:1910.11375* (2019).

[222] Gregory P Meyer. "An Alternative Probabilistic Interpretation of the Huber Loss". In: *arXiv preprint arXiv:1911.02088* (2019).

[223] Karl Granström, Christian Lundquist, and Umut Orguner. "Tracking rectangular and elliptical extended targets using laser measurements". In: *Proc. IEEE Int. Conf. on Information Fusion*. IEEE. 2011, pp. 1–8.

[224] Alexander Scheel and Klaus Dietmayer. "Tracking multiple vehicles using a variational radar model". In: *IEEE Transactions on Intelligent Transportation Systems* 20.10 (2018), pp. 3721–3736.

[225] T. Hirscher et al. "Multiple extended object tracking using Gaussian processes". In: *Proc. IEEE Int. Conf. on Information Fusion (FUSION)*. 2016, pp. 868–875.

[226] Hamid Rezatofighi et al. "Generalized intersection over union: A metric and a loss for bounding box regression". In: *Proc. IEEE Conf. Computer Vision and Pattern Recognition*. 2019, pp. 658–666.

[227] Dingfu Zhou et al. "IOU Loss for 2D/3D Object Detection". In: *2019 International Conference on 3D Vision (3DV)*. IEEE. 2019, pp. 85–94.

[228] Zhaohui Zheng et al. "Distance-IoU Loss: Faster and Better Learning for Bounding Box Regression". In: *The AAAI Conference on Artificial Intelligence (AAAI)*. 2020.

[229] Borui Jiang et al. "Acquisition of localization confidence for accurate object detection". In: *Proceedings of the European Conference on Computer Vision (ECCV)*. 2018, pp. 784–799.

[230] Tsung-Yi Lin et al. "Microsoft COCO: Common objects in context". In: *Proc. Eur. Conf. Computer Vision*. Springer. 2014, pp. 740–755.

[231] Holger Caesar et al. "nuScenes: A multimodal dataset for autonomous driving". In: *arXiv preprint arXiv:1903.11027* (2019).

[232] Alex Kendall and Yarin Gal. "What Uncertainties Do We Need in Bayesian Deep Learning for Computer Vision?" In: *Advances in Neural Information Processing Systems*. 2017, pp. 5574–5584.

[233] Yarin Gal. "Uncertainty in Deep Learning". PhD thesis. University of Cambridge, 2016.

[234] Dimity Miller et al. "Dropout Sampling for Robust Object Detection in Open-Set Conditions". In: *IEEE Int. Conf. Robotics and Automation*. 2018.

[235] Di Feng et al. "Deep Active Learning for Efficient Training of a LiDAR 3D Object Detector". In: *IEEE Intelligent Vehicles Symp.* 2019.

[236] Di Feng et al. "Leveraging Heteroscedastic Aleatoric Uncertainties for Robust Real-Time LiDAR 3D Object Detection". In: *IEEE Intelligent Vehicles Symp.* 2019.

[237] Yihui He et al. "Bounding box regression with uncertainty for accurate object detection". In: *Proc. IEEE Conf. Computer Vision and Pattern Recognition*. 2019, pp. 2888–2897.

[238] Gregory P Meyer et al. "LaserNet: An efficient probabilistic 3D object detector for autonomous driving". In: *Proc. IEEE Conf. Computer Vision and Pattern Recognition*. 2019, pp. 12677–12686.

[239] Michael Truong Le et al. "Uncertainty estimation for deep neural object detectors in safety-critical applications". In: *IEEE 21st Int. Conf. Intelligent Transportation Systems*. IEEE. 2018, pp. 3873–3878.

[240] Jiwoong Choi et al. "Gaussian YOLOv3: An Accurate and Fast Object Detector Using Localization Uncertainty for Autonomous Driving". In: *Proc. IEEE Conf. Computer Vision*. 2019.

[241] Dimity Miller et al. "Evaluating merging strategies for sampling-based uncertainty techniques in object detection". In: *IEEE Int. Conf. Robotics and Automation*. IEEE. 2019, pp. 2348–2354.

[242] Ali Harakeh, Michael Smart, and Steven L Waslander. "BayesOD: A Bayesian Approach for Uncertainty Estimation in Deep Object Detectors". In: *arXiv preprint arXiv:1903.03838* (2019).

[243] Di Feng et al. "Can We Trust You? On Calibration of a Probabilistic Object Detector for Autonomous Driving". In: *arXiv:1909.12358 [cs.RO]* (2019).

[244] Dan Levi et al. "Evaluating and Calibrating Uncertainty Prediction in Regression Tasks". In: *arXiv preprint arXiv:1905.11659* (2019).

[245] Jin Fang et al. "Simulating LIDAR point cloud for autonomous driving using real-world scenes and traffic flows". In: *arXiv preprint arXiv:1811.07112* 1 (2018).

[246] Zining Wang et al. "Inferring Spatial Uncertainty in Object Detection". In: *arXiv preprint arXiv:2003.03644* (2020).

[247] Hujie Pan et al. "Towards Better Performance and More Explainable Uncertainty for 3D Object Detection of Autonomous Vehicles". In: *arXiv preprint arXiv:2006.12015* (2020).

[248] Sascha Wirges et al. "Capturing object detection uncertainty in multi-layer grid maps". In: *2019 IEEE Intelligent Vehicles Symposium (IV)*. IEEE. 2019, pp. 1520–1526.

[249] Ryan Moulton and Yunjiang Jiang. "Maximally Consistent Sampling and the Jaccard Index of Probability Distributions". In: *arXiv preprint arXiv:1809.04052* (2018).

[250] David M Blei, Alp Kucukelbir, and Jon D McAuliffe. "Variational inference: A review for statisticians". In: *Journal of the American statistical Association* 112.518 (2017), pp. 859–877.

[251] Xiao Zhang et al. "Efficient L-shape fitting for vehicle detection using laser scanners". In: *IEEE Intelligent Vehicles Symp.* IEEE. 2017, pp. 54–59.

[252] Jason Ku et al. "Joint 3D proposal generation and object detection from view aggregation". In: *2018 IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS)*. IEEE. 2018, pp. 1–8.

[253] Yin Zhou and Oncel Tuzel. "VoxelNet: End-to-End Learning for Point Cloud Based 3D Object Detection". In: *Proc. IEEE Conf. Computer Vision and Pattern Recognition*. 2018.

[254] Shaoshuai Shi, Xiaogang Wang, and Hongsheng Li. "PointRCNN: 3D Object Proposal Generation and Detection From Point Cloud". In: *Proc. IEEE Conf. Computer Vision and Pattern Recognition*. 2019.

[255] Bin Yang, Wenjie Luo, and Raquel Urtasun. "Pixor: Real-time 3d object detection from point clouds". In: *Proc. IEEE Conf. Computer Vision and Pattern Recognition*. 2018, pp. 7652–7660.