Visual Servoing Control of Robotic System for Off-shore Oil and Gas Pipeline Inspection

A dissertation submitted to the Department of Mechanical Engineering,

Cullen College of Engineering

in partial fulfillment of the requirements for the degree of

Doctor of Philosophy
in Mechanical Engineering

by
Xiongfeng Yi

Chair of Committee: Dr. Zheng Chen

Committee Member: Dr. Hien Nguyen

Committee Member: Dr. Karolos Grigoriadis

Committee Member: Dr. Matthew Franchek

Committee Member: Dr. Gangbing Song

University of Houston
December 2021

Copyright 2021, Xiongfeng Yi

ACKNOWLEDGMENTS

I would like to express my sincere gratitude to those who have helped me with this dissertation and supported me during my four-year career journey.

First of all, I would like to appreciate my advisor, Dr. Zheng Chen, for his support, encouragement, guidance, and patience. In the academy, he provides many valuable suggestions in theory and application with his solid knowledge and experience. In my daily life, he has offered great support mentally and physically and helped me pass through this Covid pandemics. I would like to appreciate my committee members, Dr. Hien Nguyen, Dr. Karolos Grigoriadis, Dr. Matthew Franchek, Dr. Gangbing Song, for their great lectures, the knowledge they taught me, and their suggestions on my Ph.D. research and future career.

I would like to thank my fellow labmates, Alicia Keow, Wenyu Zuo, Theophilus Kaaya, Shengbin Wang, Yejin Wi, Xiang Shen, for their support, discussion, inspiration, and all the fun we have had. This work would not be finished without you guys.

A special thanks to my parents, who always back me up in all different situations. They never speak it out, but their endless love always reminds me that I am not alone during hard times.

ABSTRACT

Subsea pipelines are essential in off-shore oil and gas transportation. However, recent news has been reported that the leaking problem in Subsea pipelines has caused intensive losses in both the economy and the environment. This research is aimed to integrate the-state-of-art technologies in machine learning, visual servoing, Smart Touch inspection, underwater vehicles, and controls to enable automatic pipeline inspection. More specifically, this research introduces: 1) an optimal visual tracking and prediction algorithm which is robust in the complex environment with obstacles and light reflections; 2) a collision avoidance control for under-actuated robotic fish; 3) an integrated robotic system with a 4-degree-of-freedom (4-DOF) robotic arm that can automatically grab a flange and perform bolt looseness inspection using a pair of lead zirconate titanate (PZT) transducers; 4) a novel angle-based pipeline tracking control for a remotely operated vehicle (ROV) to track an underwater pipeline. In this research, visual detection and prediction, and collision avoidance are used for vision tracking and servoing control of robotic fish. The robotic arm, which can also be regarded as the manipulator for underwater vehicles, is guided by an on-board visual servoing system to achieve automatic Smart Touch pipeline inspection in which deep learning convolutional neural networks and stereo camera systems provide the 3D position of a targeted flange from images. Finally, the visual servoing control for pipeline tracking has been tested on a Blue ROV in a 10 m by 5 m swimming pool. Experimental results have demonstrated that the vehicle can move along the oil pipeline and monitor any failure and leaking problems.

TABLE OF CONTENTS

LIST OF TABLES

LIST OF FIGURES

Chapter 1

Introduction

1.1 Background and Motivation

With the development of the oil and gas industry, more accurate and intelligent inspection and fast responsive systems are increasingly needed for incidence, rescue, and effective damage control. With the implementation of robotics, the assistance and autonomous operation contribute to the safety and cost-effective production of oil and gas industry. In the offshore oil platform, remotely operated vehicles (ROVs) have been widely used for underwater detection and monitoring. With the manipulator or robotic arms, the ROV can accomplish more complex missions such as bolt looseness inspection. The ground vehicles help engineer with the maintenance and basic operation of the oil platform. Computer vision is one of the most important methods to acquire feedback information for the whole system. The autonomous control of the vehicles and robots with visual sensors is called visual servoing [1].

With the increasing demands of energy for modern industrialization, offshore oil has been a new source of fossil fuels. However, oil spill has become a critical problem for both the industry and the environment. For example, recently, a catastrophic oil spill on the southern California coast, which is presented in Fig. 1.1, has killed fish, mired birds, and destroyed wetlands. Advanced ROVs, Autonomous Underwater Vehicles (AUVs), and Unmanned Aerial Vehicles (UAVs) were sent to do remote operations and monitoring for the

Figure 1.1: Oil Leakage.
Source: https://www.latimes.com/california/story/2021-10-02/coast-guard-rushes-to-contain-newport-beach-oil-slick

subsurface analysis of oil spill pollution [2, 3]. In the awareness of those terrible oil spill crisis, Europe has funded several research projects to develop intelligent robots and autonomous vehicles for oil spill management [4, 5]. Another reason for applying the robots on the oil platform is due to the limitation of human capability working in extreme conditions. With the unknown environment and potential danger, the assistance from autonomous vehicles is intensively valuable [6, 7].

Subsea pipelines are crucial to the transmission of fuel and gas. For example, in the Gulf of Mexica, the US has installed over $72,000$ km of oil pipelines from 1952 to 2017 [8]. Subsea pipelines can be extended into a very deep area. According to Curtis and Maria [9], the deepest pipelines installation has been done at 2900 m in the Gulf of Mexica. Subsea pipelines operation has various potential hazards because of the extreme physical conditions and unknown environments. The damage of the pipeline cannot be treated lightly since it can lead to catastrophic oil leaking which causes damage to the economy and environment. According to the US Department of Transportation(DOT), the failure in oil and gas pipelines can be classified into five categories [10, 11, 12]: 1) Mechanical failure, which is due to manufacturing or installation issues; 2) Operation failure; 3) Corrosion, which may be internal or external; 4) Natural hazards, like earthquakes, wave currents, or

storm; 5) Third party mistake, which usually happens in the shallow water, where the ship vessels can cause damage to it while anchoring.

The pipeline inspection consists of the use of special tools and methods to inspect the elements and detect the potential damages. For example, inspection through vision was used to check the corrosion and fractures on the surface of the pipelines. Ledezma *et al.* has concluded several types of non-destructive testing (NDT) [13]: 1) Ultrasonic testing (UT) [14]; 2) Magnetic P-particle inspection (MPI) [15]; 3) Magnectic flux leakage [16]; 4) Eddy-current testing [17]; 5) Guided wave pipeline inspection [18]; 6) Cathodic protection measurement [19].

Many companies have developed their own underwater vehicles with inspection functions. In Fig. 1.2, Saab Seaeye introduced their Tiger ROV, which can be operated at the depths of 1000 m. Fugro Subsea Services Ltd. developed ROV FCV 2000D. This ROV can work in the environment at a depth of 2000 m and inspect the pipeline structure of 25 km per day. BlueRobotics designed BlueROV2, which is claimed to be the world's most affordable high-performance ROV. It is open-source in both electronics and software. In the later chapter, we will use this ROV to test the pipeline following algorithm [20].

Figure 1.2: Saab Seaeye Tiger ROV.
Source: https://www.saabseaeye.com/solutions/underwater-vehicles/tiger

1.2 Literature Review and Technical Challenges

1.2.1 Early Work on Visual Servoing

Visual servoing is a vision-based robot control, which uses visual feedback to control the motion of robots. In 1979, Gerald J Agin from SRI International Labs discussed this topic in the book "Real Time Control of a Robot with a Mobile Camera" [1], which is regarded as the earliest paper about visual servoing. The image based features which include points, lines, corners, and other patterns are used to build up the environment within the camera view. Because of that, computer vision is the sensing mechanism which provides position, velocity, and acceleration based on the images captured by cameras. Visual servoing has been studied in multiple forms for more than four decades. With the boost of the computation hardware

and the image processing algorithm, visual servoing technique has been applied to complex missions in an unpredictable environment. For example, an ROV can automatically explore the deep ocean with visual servoing; a heavy quadcopter can deliver the package to the customer; a self-driving vehicle can finish the driving task without any drivers.

Visual servoing has been a method for estimating the states through computer vision systems. The basic function of the visual servoing is to guide the motion of the robot relative to the target object or locations. An error function or task function [21] is defined as

$$e(t) = s(t) - s_d, \tag{1.1}$$

where $s(t)$ and s_d denote the states vectors of output and desired features, respectively. The goal of the control system is to regulate $e(t)$ to zero. Visual servoing is such a kind of control process where computer vision provides all or part of the real output vector $s(t)$ in real-time.

A closed-loop visual servoing control system consists of two processes: 1) Tracking (visual); 2) Control (servoing). In the real-time control, the tracking process provides a continuous state estimation and feature update while the robot/vehicle is moving. In the control unit, the output of the sensor is regarded as an input to the controller, where a control signal is generated to move the whole system such that the control error can be reduced to zero. For the basic visual servoing method, the system requires a manual initialization to start the tracking process. Usually, it starts with the ground information or object segmentation. However, some automation systems have applied the self-search function to initialize the visual tracking process by themselves [22].

The image processing and the control of the robot consist of two tasks. In a typical visual servoing procedure, the computer vision method is performed and followed by a control sequence. An example is to recognize an object by matching image features to a geometrical template and calculating the position, and posing relative to the camera coordinate [23]. The Cartesian-space pose information is used to generate the control signal for the robot to move to the desired position and pose. The model of the object is identified

in order to estimate the pose of the object. To extract the visual information from camera frames, the camera system has to be calibrated before it is included in the visual servoing system. For the manipulation robot like robotic arm, the forward and inverse kinematic models are derived to transfer the information among Cartesian space, joint space, and the camera system space.

Recently, visual servoing has attracted significant attention since computational resources and algorithms support made it possible to deal with complex computer vision processes in a much shorter period such that the real-time process is applicable. Besides, robust control methods for the complex scenarios in the real world have moved the achievement from the experiment into our daily life. Visual servoing is used in various applications in the real world, such as lane tracking of autonomous driving cars, objects manipulation in the factory, and indoor robots for smart home and wholesale management.

In 1979, Hill *et al.* discussed real-time visual servoing with a mobile camera [1]. Sanderson *et al.* introduced the "Dynamic look-and-move systems" and "Direct visual servo system" [24]. In Dynamic look-and-move systems, there are two stages of the control: 1) the visual system provides the feedback information to the controller; 2) The controller uses the joint to stabilize the robot.

Behind the fast development of the visual servoing technique, researchers solved millions of problems in this domain. They introduced new algorithms to implement new functions, verified the robustness of the method in different real world situations, and made tutorials and literature reviews. Hutchinson *et al.* made a visual servo control tutorial on robotics manipulators [23]. Based on Hutchinson's work, Kragic *et al.* summarized the visual servoing techniques from 1979 to 2002 [22]. Kragic pointed out that the lack of terminologies and taxonomy for the different applications and the approaches became a problem with the increasing number of contributions to visual servoing. In addition, there was a lack of a survey of the vast literature in visual seroving at that time. However, after another ten years, tutorials and surveys have been produced to present visual servoing with different

subdivisions. Chaumette *et al.* divided the visual servoing knowledge into basic and advanced approaches, which includes the object detection through computer vision approach and servoing control stability analysis [25, 26]. Kazemi *et al.* reviewed the research related to the path-planning for visual servoing from 1980 to 2010. He summarized the previous works in visual servoing and moved into the path-planning domain [27].

Visual servoing involves many other research domains, which include modeling (dynamics, kinematics, geometry), real-time control systems, controller design, computer vision, mapping and localization, path and motion planning, and systems integration. In 1994, Peter I Corke [28] pointed out that the visual servoing can be classified based on different aspects: the configuration of sensors, the number of cameras in the system, whether the control command is in 2D or 3D, the environment interpretation, and the computer vision that has been applied.

Current visual servoing techniques can be categorized into two classes: Position-based visual servoing (PBVS) and Image-based visual servoing (IBVS). PBVS are systems that build up 3d information about the environment where the position and the orientation estimation of the object are calculated for the world, joint, and camera coordinate. PBVS has been widely used in robotic arms and indoor operation vehicles control, where the camera has a clear vision of the targets. In PBVS, the camera is standard equipment that provides the target's 3D position and orientation. However, the simultaneous localization and mapping (SLAM) of the camera needs more power consumption due to the heavy computation cost for the complex 3D calibration and stereo camera localization algorithm. Many algorithms have been developed for visual servoing tasks. The relative reference between the camera and the ground is given for most ground vehicles. Thus the position and the orientation are calculated from the reference [29, 30, 31]. However, for underwater vehicles like ROVs or unmanned aerial vehicles (UAV), the vehicle does not share the same horizontal plane with the target objects. The height from the camera to the ground is unknown if the vehicle only equips a monocular camera. The direct control of the distance in the vertical direction is an important step before the horizontal plane control. For aerial

vehicles, the optimal height guarantees a beneficial trade-off between the loss and fading in the path following process. It is also the prerequisites for precise localization and stable motion for the UAVs [32, 33]. Buoyancy control is one of the most important functions of underwater vehicles. Without an optimal active control in the buoyancy force, the vehicles either sink to the bottom or float on the surface in a complex scenario. The high pressure and dusty seabed environment and the waves on the surface make the underwater missions impossible [34]. Keow *et al.* designed an optimal depth control unit with reversible PEM fuel cells, which saves a lot of energy for the equipment [35].

For a robotics system in 3D environments, u is the control input for the Euclidian motion of the robot in a task space τ, which is denoted as $\tau_G \in SE(3)$ [22]. If the robot is controllable in six degrees of freedom,

$$u = [V_x,\ V_y,\ V_Z,\ \omega_X,\ \omega_Y,\ \omega_Z]^T \in R(6). \tag{1.2}$$

The measurement y contains the pose or coordinate of the target, while $y*$ is the desired measurement of the control system. As we mentioned earlier, the goal of visual servoing task is to regulate the task function, or the control error function to zero [36]. The task function is also referred to as the kinematic error function for PBVS and the image error function for IBVS.

1.2.2 The-State-of-the-Art on Visual Detection and Localization

In visual servoing control, the feedback information from the computer vision is given from the image plane with correlation based methods, optical flow techniques, the difference between images, together with the camera parameters and other known features. This process depends on the number of cameras, camera structures, the calibration level and the pre-known references.

To determine the object's 3D pose in camera coordinate, the motion of the object in serials of images are derived based on the intrinsic parameters of the camera [37, 38, 39, 40]. In the traditional method, three match points between the image and the object provide

multiple solutions, while a unique solution is given with four match points [38, 41]. To explicitly measure the pose and the position of the object, the system with a single camera usually matches features (corners [42], edges [43], lines [44]).

For the monocular camera, the 3D to 2D project mapping is important from H. C. Longuet-Higgins's research [45]. The relationship between matched features and Euclidian structure was reconstructed from the matrix estimation [46]. To calibrate the camera, the fundamental matrix allows reconstruction and calibration without a known camera's intrinsic parameters [47]. One of the most commonly used methods is to calculate the estimation directly from the image plane. There are many famous algorithms, which include window based tracking, and feature based tracking [48, 49]. For the images with low resolution, window based techniques do not require large computing power and are flexible with respect to the content of the image. However, a large manual reconfiguration has to be done before this algorithm is transferred to a different object. In the feature match category, one of the most famous techniques is the Hough transformation [50, 51]. The Hough transformation is easy to be implemented but requires storage space and time, which increases exponentially with the number of features and the size of the image. Many researchers [52, 53, 54] used active contours to track moving objects. This algorithm tracked objects with arbitrary shapes and is robust to the situation with a blocked view. Using a camera for global localization is a common method in early visual servoing systems. One of the earliest visual servo systems used edges and lines to extract the pose, and the location of the object [55]. In 1992 Feddema *et al.* studied two single-camera configurations, which are eyes-in-hand and stand-along modes. Probability map has been widely used to localize the possible position of the object in the image. In 2005, John Ashburner used a probabilistic framework to realize a unified image segmentation [56]. Fang *et al.* introduced a boosting instance segmentation method via probability map [57].

Stereo vision, which extracts 3D views from multiple 2D images, provides feedback information for automation control. The 3D information can be obtained from a pair of stereo images(stereo pair) by estimating the relative depth of corresponding points in

stereo cameras [58]. A stereo camera consists of two or more lenses with separated image sensors. Camera calibration is a crucial step to extract information from 2D images and build the 3D coordinates. One can classify the techniques into two categories: 1) Single-camera calibration, which derives properties of each camera and restores the lens distortion problem [59]; 2) Stereo camera calibration, which is used to determine the relative location and intrinsic parameters of cameras for stereo rectification and 3D reconstruction [60]. A stereo camera system usually consists of two or more cameras. This arrangement is used to provide the 3D information of the environment. One of the common methods for the 3D reconstruction is to calculate the disparity of the view and then estimate the depth [61, 62, 63]. Some approaches are adopted to find the disparity by matching the corresponding feature between the images from cameras: 1) correlation region matching; 2) feature matching.

Camera calibration is the key to 3D computer vision. For single-camera calibration, based on a pinhole camera model, a mapping relationship between the image position and the real-world position can be given as

$$w[x_i, y_i, 1] = [X_w, Z_w, Y_w, 1] \begin{bmatrix} R \\ t \end{bmatrix} K, \tag{1.3}$$

where w is the scale factor, 2-D vector $[x_i, y_i]$ is the image position, 3-D vector $[X_w, Y_w, Z_w]$ is the real world position of the flange, P is the camera matrix. In the extrinsic matrix, R and t contribute to rotation and translation of positions, respectively. The intrinsic matrix K is defined as

$$K = \begin{bmatrix} f_x & 0 & 0 \\ s & f_y & 0 \\ c_x & c_y & 1 \end{bmatrix}, \tag{1.4}$$

where $[c_x, c_y]$ is the optical center in the unit of pixels, $[f_x, f_y]$ is the focal length in pixels and the skew coefficient s is a non-zero scale if the image axes are not perpendicular.

Different from the pinhole camera, the modern cameras have the distortion phenomenon

because of the lens: when light rays pass through the edge of a lens, they are bent more than at the optical center. Two main distortions for lens cameras are radial and tangential. Single-camera calibration for reducing lens distortion is necessary for more accurate stereo camera parameters. For an ideal stereo camera system where there are two parallel optical axes and co-planar image planes, the depth is derived from triangulation [64]. However, to achieve a reliable 3D position, a more complex calibration method is applied to the problem in general case [65].

$Z_w = f\frac{T_X}{d}$, where f, T_X and d represent the focal length, the x axis shift in real world coordinate from the main camera(left camera) to another one, and the disparity (horizontal pixel difference of the same object in two images), respectively. However, the real stereo system never has two cameras sharing the same baseline. A small angle or position difference leads to a huge error, especially for the objects far away from the system. Thus rectification is necessary before the stereo calculation. Rectification adjusts stereo images with determined extrinsic and intrinsic matrices so that they are on the same plane. We refer $[\hat{x}_{i1}, \hat{y}_{i1}]$, $[\hat{x}_{i2}, \hat{y}_{i2}]$ to rectify the image coordinates of the left and right camera, respectively. The real world positions of the target object can be determined from the following equations

$$\begin{bmatrix} X_w \\ Z_w \\ Y_w \\ 1 \end{bmatrix}^T = w \begin{bmatrix} \hat{x}_{i1} \\ \hat{y}_{i1} \\ \hat{x}_{i1} - \hat{x}_{i2} \\ 1 \end{bmatrix}^T \begin{bmatrix} 1 & 0 & 0 & 0 \\ 0 & 1 & 0 & 0 \\ 0 & 0 & 0 & \frac{-1}{T_x} \\ -c_x & -c_y & f & 0 \end{bmatrix}. \tag{1.5}$$

It is not until this decade when computers' hardware has been explosively upgraded, which provides a favorable environment for artificial intelligence. Convolutional neural networks (CNNs) are widely used in computer vision for automation control and pattern classification [66]. To provide the location information and classification, people also developed RCNN, and faster RCNN [67, 68]. Later, You only look once (YOLO) [69] were introduced to improve the learning performance of networks. It predicts all bounding boxes for all classes in an image simultaneously such that YOLO designs provide end-to-end training

and real-time speeds while maintaining high detection precision. Later, Liu *et al.* developed a single shot multibox detector for object localization, which is faster than the YOLO and significantly more accurate [70]. These days, deep learning has a significant effect on traffic, health care, manufacturing and increases the efficiency of society.

1.2.3 The-State-of-the-Art on Control and Collision Avoidance

The development of collision avoidance control is motivated by the increasing concern for vehicle safety. With the collision avoidance system, the vehicles can avoid unnecessary colliding with static and moving obstacles. In the motion planning of the mobile robot, obstacle avoidance is one of the fundamental functions. Motion planning can be classified as static (when the obstacles are stationary in the environment) or dynamic (when the position, shape, size, and orientation are changing) [71, 72]. The environment is either completely known when the trajectory of the obstacle is given along the time or partially known when the information of the object is not fully given. Early collision avoidance control mainly focuses on the static obstacles by sensing the environment around the vehicle [73], applying decision trees to deal with different corner cases [74], and using the path planning method to avoid obstacles [75]. However, in the real world, some obstacles are not static. Watanabe *et al.* [76] developed a computer vision-based collision avoidance control with optimal effort. Mujumdar *et al.* [77] demonstrated a collision avoidance method with nonlinear differential geometric guidance. [78] defined next-generation airborne collision avoidance system by dynamic programming. The collision avoidance control method can be further divided into two categories: non-cooperative approach and cooperative avoidance approach. Non-cooperative collision avoidance control does not have prior information about the obstacles, while the cooperative approaches have cooperative communication and share the information between each other. Yang *et al.* defined a 3D collision avoidance strategy in a non-cooperative environment [79]. In [80], Jenie *et al.* developed another non-cooperative avoidance technique under the assumptions that the distance, directions, and the speed constraint of the obstacle are known. In a multi-vehicle system, cooperative collision avoidance

controls are required [81, 82, 83]. Boivin *et al.* described cooperative predictive control for collision avoidance [84]. Hafner *et al.* discussed the cooperative collision avoidance control at intersections and used experiments to verify the algorithm [85].

For ground vehicles, Seriler *et al.* discussed the rear-end collision avoidance system, together with the collision warning/avoidance (CW/CA) algorithm [86, 87, 88]. The system with the CW/CA algorithm should meet three criteria: 1) The system warning should have a minimum load on the driver's attention; 2) The automatic brakes control does not affect the normal driving operation; 3) The system is robust in different driving conditions. In 2019, Huang *et al.* presented the existing work of collision avoidance control of multi unmanned aerial vehicles through classification based on algorithms, frameworks, and primary features [89]. They analyzed the concern about the current collision avoidance problem in practical application. Collision avoidance control is also crucial in marine affairs. Chirdchoo *et al.* introduced a Aloha-based protocol with collision avoidance control for underwater acoustic networks [90]. Spangelo provided trajectory planning and collision avoidance for underwater vehicles with optimal control.

1.2.4 The-State-of-the-Art on Underwater Robots

In the offshore oil and gas industry, underwater vehicles are important to search, rescue, and monitor. They also contribute to oceanographic research, underwater archaeology, and ocean creatures analysis. Many of these missions are operated by remotely operated vehicles (ROVs). ROVs are tethered and controlled from surface computers. Recently, autonomous operation on ROV has become increasingly essential. Autonomous underwater vehicles (AUVs) are now applied to multiple maritime domains.

ROVs are tethered underwater robots. They could be either powered by a physical link from the surface or the internal batteries. The control signal is from the ship-borne operators. The British and US navies contribute to the early ROV development during 1950s and 1960s [91]. In 1950s, British developed their ROV to perform deep-sea rescue

and recovery operations. The ROV named Cutlet was shown in Fig. 1.3. In 1973, the Cable-controlled Undersea Recovery Vehicle(CURV)-III finished the deepest underwater rescue in history. It rescued two men who were stranded $480m$ below the ocean surface for 76 hours. The CURV-III is in Fig. 1.4

Figure 1.3: The Cutlet ROV by the Royal Navy.
Source: https://en.wikipedia.org/wiki/Remotely-operated-underwater-vehicle

Figure 1.4: The US Navy CURV-III ROV during the Pisces III rescue operation. Source: https://en.wikipedia.org/wiki/CURV

In general, current industrial ROV operations are based on manual control. The operation performed depends on the experience and proficiency of the operator. Thus the efficiency of the operations is not guaranteed, and it needs large funding investment on training operators. The autonomous control in ROV increases the efficiency and thereby reduces the funding cost. This is one of the main reasons for the development of AUV [92]. Semi-autonomous control is a solution to the ROV operations, where the base work is done by the vehicle itself, and the operator only focuses on monitoring the behavior of the vehicles and the extreme cases. [93] discusses the technology components for autonomous control in ROV operations.

The dynamic positioning and control have been developed for more than twenty years [94, 95]. The progress in recent research is related to the stability analysis, coordinate transformation, and hydrodynamic parameter identification [96]. In 2004, Christ *et al.* discussed the ROV system in practical aspects [97]. The research of vehicle manipulator operation is summarized in [98]. Position keeping is also an important function for underwater vehicles.

Candeloro *et al.* applied the observer theory in 2012 [99]. Other related theoretical research and result about telerobotics also contribute to the development of the ROV autonomous operations [100].

1.3 Constraints in Visual Servoing Implementation

The stability and the convergence in both IBVS and PBVS are not guaranteed with some constraints [101]. The constraints can be divided into two categories: 1) The constraints of image or camera; 2) Robot or Physical constraints.

The sensor limit of the camera system is the main component of the constraints of the image or camera. The internal relationship is given as

$$\dot{s} = {}^{s}L_{x}\dot{x}, \tag{1.6}$$

where $\dot{s}$ is the optical flow (rate of change) of the features, $\dot{x}$ is the camera's Cartesian velocity, and ${}^{s}L_{x}$ is the image interaction matrix or Jacobian matrix [23]. These constraints include the field of view limits, local image minima, and singularities in the image Jacobian matrix.

The camera has certain limitations based on its manufacture and mechanisms. For example, an object is visible to a camera only if all 3D features' projects are within the boundary of the image. The limits of the image are reflected by the size of the rectangular frame. In IBVS control, the output is given defined in the image. There is a possibility that the target features are outside of the camera's field of view, in particular when the current position and the target position are close [101, 102].

In the IBVS context, the unpredicted image motions which are not included in the Jacobian matric ${}^{s}L_{x}$ will cause the image local minima. The determination of the image local minima is not easy without the specific target location in different coordinates, which causes an exhaustive search for local minima during the visual servoing task. A nominal value of image Jacobian estimation for the desired location help avoid local minima according to

[101]. But this method may violate the field of view constraint, especially in PBVS. Thus the PBVS methods are known to work with local minima-free images.

When there are singularities in the Jacobian matrix, some camera motions cannot be achieved in the image space. There are many cases that image singularities are considered: the vector of image features s consists of three co-linear points or three points on the same cylinder, which contains the camera optical center. Using more noncoplanar points will avoid such singularities, but the Jacobian matrix may still be singular. For example, a visual servoing task has a 180° rotation around the optical axis. Using lines instead of pure points features helps avoid the singularities. Motion perceptibility is proposed to measure the closeness to image singularities. [103]

The motion of the robot system is produced by the visual servoing control loop. This motion has some constraints, which are introduced by the robot and obstacles in the workspace. These constraints include: 1) Robot kinematics such as joint limits; 2) Robot dynamics; 3) The safety zoom to avoid the collision with obstacles or self-collision; 4) Occlusion in the 3D space.

The robotics community has been developed for more than four decades, and a lot of research has been devoted to planning the path to avoid constraints or collision in robot kinematics and dynamics. Some researchers proposed partitioned control strategies with a specific degree of freedom in IBVS and PBVS [104, 105, 106]. These partitioned approaches have their own advantages and disadvantages in different constraints. Gans *et al.* have made a comparison based on the performance of some of the partitioned approaches in IBVS [107]. Later, hybrid strategies have been proposed to enlarge the stability region of the classical visual servoing method and switch between a set of the unstable controller to make sure that the overall system is stable [105, 108].

1.4 Research Contributions

The object of this research is to discuss the visual servoing control of robotic systems, which contribute to the off-shore oil and gas pipeline inspection. For the first visual tracking

method, we aim at tracking the object in two-dimension by a fixed camera in the environment with noise and obstacles. We first introduce the method and algorithm we use to locate the object, including the view domain transformation, using minimum volume ellipse to find the vector to represent the object. We use the vector, the shape, and the location information to classify the pixels into four kinds and apply a different method to resolve or predict the position of the object. The Kalman filter [109, 110] is applied to the position and velocity data to reduce the effect of random noise and the sudden change of the velocity and position.

For the robotic fish collision avoidance algorithm, The servo motor produces sufficient mechanical power to move the fish forward while the voltage applied to the IPMC controls the heading direction of the robotic fish. Such a robotic fish has several advantages when compared to the robotic fish actuated by servo only or artificial muscle only: 1) The tail is light, and therefore using a slice of IPMC in the second joint reduces the mass and drag of the tail (when compared to adding an extra servo motor as the second joint); 2) The caudal fin of the fish is easily attached to the first joint of the fishtail by an IPMC slice; 3) The servo motor driven first joint generates a high swinging torque to make the tail flap at high frequency, and this can drive the robotic fish to swim at a relatively high speed; 4) The hybrid tail makes the fish turn smoothly, due to the actuation introduced by the IPMC; 5) Since the IPMC works in a wet environment, there is no need to waterproof the IPMC.

The goal for the robotic arms system is to detect the looseness of a flange using a robotic system with portable smart touch sensors. To achieve this goal, a lightweight robotic system, which consists of 4 degree-of-freedom (DOF) robotic arm, four stereo cameras, and piezoceramic smart touch sensors, is developed. The piezoceramic transducers are mounted on the two fingers of the robotic arm. The robotic arm automatically moves the piezoceramic sensors to touch on the flange for bolt looseness detection. The flange is recognized by the cameras using YOLO convolutional neural networks with 944 image samples and located with a feature-based method and bounding boxes. The 3-dimension (3D) world coordinates from the stereo camera system are sent to the robotic arm controller. When the fingers

are close to the flange and the cameras do not have a complete view of the flange, the controller is switched to a mode under which the robotic fingers move to the flange step by step until the fingers touch the flange. At each step, the robotic fingers try to grasp the flange, and the smart touch sensors provide touching feedback information. Once the robotic arm grabs the target, the energy waves from the piezo sensor on fingers reflect the bolt looseness of the flange. Several experiments have been conducted to validate the proposed approach. Experimental results have shown the potential of using this robotic system in autonomous robotics-assisted pipeline inspection. The major contribution of this research is that we develop a systematic approach that integrates computer vision, machine learning, and active sensing for flange bolt looseness detection, which enables autonomous pipeline inspection to prevent bolt failure and oil leaking.

The major novelties of the ROV pipeline tracking method are: 1) An angle based path planning method transfers a 2D path following problem into a 1D trajectory tracking problem; 2) An line pickup criteria rule out the noise and choose the proper target direction from multiple pipeline segments; 3) The angle projection mapping function derives the states information even when the length reference information (the width of the pipeline or the distance from ROV to the pipeline) is not given in the calculation; 4) A model-based nonlinear visual servoing controller tracks the reference angle, a Lyapunov stability analysis proves the stability of the controller, and the whole process is validated by simulations and experiments with the disturbance.

1.5 Dissertation Organization

The objective of the research focuses on the application of visual servoing, which includes the robust approach of computer vision and control algorithms. In Chapter 2, we introduce a robust visual tracking method in complex environments where there are occlusions from obstacles and light reflection noises. In Chapter 3, we introduce a cooperative collision avoidance control on two robotic fish. The fish is propelled by a two-joint-tail, a servo motor, and an ionic polymer-metal composite (IPMC). Chapter 4 discusses a novel robotic

system that can automatically grab the flange of the pipelines and perform bolt looseness inspection autonomously. The robot consists a stereo camera system and applies a region-based deep learning algorithm. Chapter 5 describes a novel pipeline tracking algorithm of an underwater vehicle with a monocular camera. The algorithm works under the disturbance when the outside length reference is unknown. Finally, in Chapter 6, the main findings and conclusions are summarized, and the research for the future plan is also listed.

Chapter 2

A Robust and Optimal Visual Tracking and Predicting Method for unmanned mobile systems

The chapter introduces an optimal algorithm for tracking the object in an environment with occlusion and noise. If the object is partially or completely blocked by the obstacles or water reflection, the method can predict its position based on the previous information and the Kalman filter. The motivation of this chapter is to find and predict the position of the object even when the visual sensor does not work so that the unmanned mobile systems can still receive the feedback information and control the object [111].

2.1 Visual Tracking and Predicting method

In this section, the main algorithms for tracking the vehicle on the surface are introduced. They are divided into three parts: 1)View domain transformation. This part calculates the matrix to transform the position data from the image to the real-life Cartesian coordinates. 2)Preparation, this part is the beginning step that resolves the object and gives the initial position data. 3)Object tracking, this part resolves the pixels and then figures out pixels

that represent the object being tracked. 4)Optimization, we apply the Kalman filter on the velocity result to reduce the experimental error [109, 110, 112, 113].

2.1.1 View Domain Transformation

The mechanism of projection on the image is to first transfer the 3D real-world coordinate(The coordinate which is given before the tracking) to the camera coordinate, then project on the film. The affine result of the project is demonstrated as pixels [114]. Eqn. 2.1 is a transformation from 2D to 3D

$$\begin{bmatrix} x \\ y \\ 1 \end{bmatrix} \sim \begin{bmatrix} h_{11} & h_{12} & h_{13} \\ h_{21} & h_{22} & h_{23} \\ h_{31} & h_{32} & h_{33} \end{bmatrix} \begin{bmatrix} c \\ r \\ 1 \end{bmatrix}. \tag{2.1}$$

In the relationship Eqn. 2.1, x and y are the coordinates of points in the real world. c and r represent the column and the row of the pixel in the image. To simplify the expression, we use 'R' to represent the real world domain and 'I' to represent the image pixels domain. The transformation matrix h is the result of combining the three projection steps from 'R' to 'I'. To solve the matrix with nine unknown variables, we need nine equations, which are four points with an extra condition on the matrix or more than five points by using the least square regression to have an approximation. In our algorithm, we first pick up four points. If h_{33} is not zero, we set it as one and calculate the rest eight variables with the given four points. The equations between the 'R' coordinate and the 'I' coordinate are given as

$$x = \frac{h_{11} \times c + h_{12} \times r + h_{13}}{h_{31} \times c + h_{32} \times r + h_{33}} \tag{2.2}$$

and

$$y = \frac{h_{21} \times c + h_{22} \times r + h_{23}}{h_{31} \times c + h_{32} \times r + h_{33}}. \tag{2.3}$$

2.1.2 Preparation

In the preparation part, we first pick up reference points of four corners whose coordinate in 'R' is known. Other points with specific coordinates are also the choice of the reference points in case any corner points are blocked from the camera view. After the tracking domain is fixed, pixels outside the domain are all set to zero to rule out any possible noise from the outside domain. We pick up the fifth point on the objects. A searching window is chosen around the object. To choose the size of the searching window, we first make a square in the 'R' and transform the four corners into the 'I'. We use the maximum and minimum of the value in x and y direction of the four corners to make a new searching window. Try a different value of the length of the square so that the object is the largest pixel group in the new searching window in 'I'. The pixels are resolved by the color feature. We set a different threshold of red, green, and blue and the ratio between them to differentiate the object from other noise.

The object tracking part includes the important algorithms used in each time step of the process. During the tracking process, the object may disappear from the camera due to interference from the noise and the obstacles. Two situations are analyzed in this research; one is when part of the object body is blocked, another is when the whole body is blocked. To solve the part blocked problem, we first find an ellipse with the minimum area containing enclosing the object [115], the part of the long axis inside the object is used as a vector to represent the object. The direction of the vector meets the condition that $\vec{v} \cdot \vec{p} > 0$ where $\vec{v}$ and $\vec{p}$ represent the velocity and the vector, respectively.

2.1.3 Minimum Volume Ellipse

The general expression of the ellipse is

$$\frac{(x\cos(\alpha) + y\sin(\alpha))^2}{a^2} + \frac{(x\sin(\alpha) - y\cos(\alpha))^2}{b^2} = 1. \tag{2.4}$$

In Eqn. 2.4, the center of the ellipse is at the origin. $x = x - c_x$, $y = y - c_y$ if the center is at the point c. The expression of the ellipse is also given as 2.6, where

$$E = \begin{bmatrix} \frac{\cos^2(\alpha)}{a^2} + \frac{\sin^2(\alpha)}{b^2} & \cos(\alpha)sin(\alpha)(\frac{1}{a^2} - \frac{1}{b^2}) \\ \cos(\alpha)\sin(\alpha)(\frac{1}{a^2} - \frac{1}{b^2}) & \frac{\sin^2(\alpha)}{a^2} + \frac{\cos^2(\alpha)}{b^2} \end{bmatrix}, \tag{2.5}$$

$$(x - c)^T E(x - c) = 1, \tag{2.6}$$

and x is the points on the ellipse, c is the center of the ellipse and α represent the rotation angle of the ellipse. The volume of the ellipse is written as $\frac{\pi}{\sqrt{|E|}}$. Nima Moshtagh's research in 2005 gave a solution to find the minimum determinant of E^{-1}, where all the given points are in the ellipse, where $(x - c)^T E(x - c) < 1$ [115].

2.1.4 Vector Prediction

In this method, we use a vector to represent the object. By doing this, there are three advantages: 1) we directly use the middle point of the vector as the location of the object to calculate the velocity and acceleration: as shown in the figure 2.2, we first use the color threshold to have the edge of the object and transfer it into 'R' space. After we find the minimum volume ellipse enclosing the pixels [115], the long diameter of the ellipse has at least two intersections with the boundary (the edge that the shrink factor is 0.5) of the pixels. We choose two intersections with the longest distance as the vector. The middle point of them is the location of the object. 2) The vector in the previous frame is used as a reference to find the object in the new frame: when a group of pixels is resolved, if the difference between the current vector and the reference is less than the tolerance in both length and the direction, this group of pixels passes the vector check. 3) When part of the object is blocked, the vector is used to predict the location of the object: When the blocked part of the object is small, the recognized shape gives a vector with the same direction but a shorter length. We use the direction from the given vector and the average length of all previous vectors to represent the object. As shown in the figure 2.1, although a part of the

object has already been blocked by the obstacle, the red vector shows how much the object was blocked. The location of the object, which is the middle point of the vector, is marked by the black cross. If the blocked part is very large compared to the rest, the signal which is sent back to the camera is limited. The controller of the object will not make any change if there is not enough input signal. Thus the object will remain the same status as the previous frame. We use the direction of the previous vector and the length of the average value of all previous vectors as our reference vector.

Figure 2.1: The location given by the vector prediction method. The red line is the vector representing the object, the black cross is the location of the object

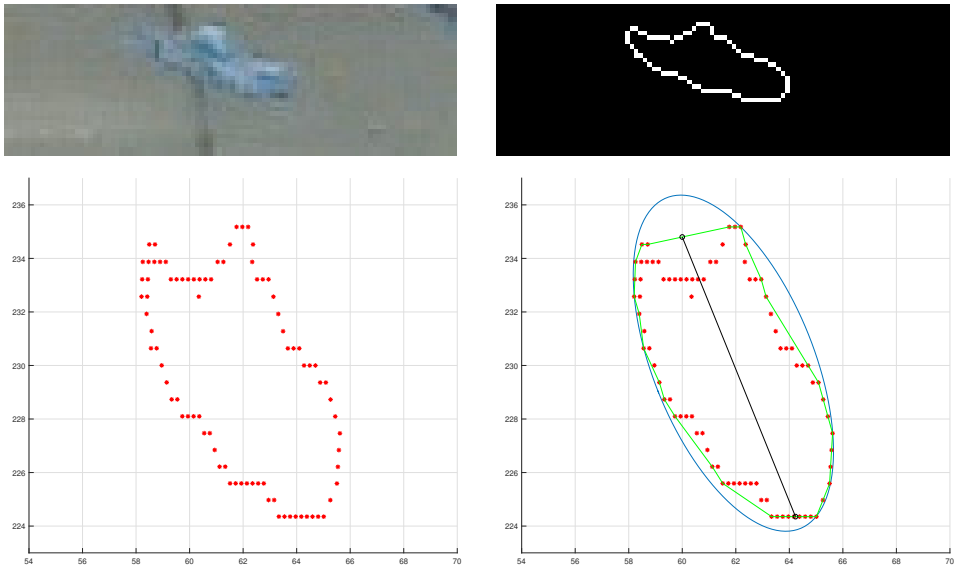

Figure 2.2: The process of locating the object. The first image is the original frame, the second one is the edge of the object, the third is the edge point in the 'R', the last one is the result of the minimum volume

2.1.5 Shape Recognition

Sometimes the vector prediction method does not provide a good result because the vector of the noise is coincidentally the same as the reference vector. In that case, we introduce the shape recognition method. Shape recognition is only applied in special cases where the tracking time interval is short, and the shape change is small between two neighbor images. Thus the shape of the last frame is used as the reference to figure out the shape of the object. In this research, we use Canny's algorithm [116] to have the edge. For the image $I(x, y)$, where all pixels are real numbers. The gradient of the image is given as

$$\nabla I(x, y) = (\frac{\partial I}{\partial x}(x, y), \frac{\partial I}{\partial y}(x, y)). \tag{2.7}$$

Direction of the gradient is

$$\nabla \hat{I} = \frac{\nabla I}{\|\nabla I\|}. \tag{2.8}$$

The pixel is an edge if

$$\|\nabla I(x, y)\| > \text{Threshold},$$

$$\left\|\nabla I(x,y)\right\| > \left\|\nabla I((x,y) + \nabla \hat{I}(x,y))\right\|,$$

$$\left\|\nabla I(x,y)\right\| > \left\|\nabla I((x,y) - \nabla \hat{I}(x,y))\right\|.$$

In Canny's method [116], there are two threshold. Pixels are the edge if $\left\|\nabla I(x,y)\right\| > Highthreshold$ or $\left\|\nabla I(x,y)\right\| > Lowthreshold$ and the pixel is next to an edge. If the overlap between the current edge and the reference is larger than the threshold percentage, the pixel group will be regarded as the object.

2.1.6 Tracking Steps

After the preparation, we first use color to resolve pixels in the normal track part. We create a searching window where the center is the current location of the object. The vector and the location of all pixel groups resolved in the searching window are tested. If the test is passed, the position in both 'R' and 'I', the vector body in 'R', and the edge are stored and passed to the next iteration. If anyone of the test is not passed, this situation will be treated as a special case, and the starting time is from the last resolved frame. In the special tracking method, the time interval is shortened to increase the accuracy of the prediction. As shown in table 2.1, '1' means passed the test, and '0' means did not pass. We first test the vector. As long as the object is not blocked, the length in 'R' is always the same. Thus when the length difference is less than the tolerance, this group of pixels is the object. When a small part of the object is blocked, the angle of the vector does not change much, but the length is short. In that case, we use the current direction and the length of the average value of previous vectors to predict the location of the object. When a large part of the object is blocked, there is a large difference between the reference and the current vector in length and angle. We use the shape and the location condition to test if the group of pixels is the part of the object. In the location prediction, we use the same velocity during the last time interval to predict the location in the next frame. If the distance between the group of pixels and the result of location prediction is less than the tolerance, those pixels pass the location check. If there both shape and location check is

passed, the pixels are regarded as part of the object. Other than that, the resolved pixels are noise. If all pixels are noise, it means the whole object has been blocked, and we use the result of the location prediction to represent the object.

Table 2.1: The classification of the different situation when tracking the object

Shape	Location	Length	Angle	Situation
		1		Normal
		0	1	Small part blocked
1	1	0	0	Large part blocked
	0	0	0	Noise
0		0	0	Noise

2.2 Optimization

In each frame, when we try to find the position of the object, the error due to the brightness change and the fluctuation of the water affect the accuracy of the result. Although the value of the error is small in position, since the time interval is small, the error is large as the result of velocity. When the object is about to enter the blind area, we use the smaller time interval to increase the accuracy of the prediction in position. However, the smaller time interval amplifies the error in the velocity domain. To reduce the effect of the error, a traditional method is to take several sets of the data and use the average value as a result. However, in the real-time monitor and control system, it is impossible to do that. Alternatively, since the relationship between the position and the velocity is linear, we introduce the Kalman filter to optimize our simulation result [109, 110, 112, 113]. The Kalman filter model assumes that the state at time t is derived from the time t-1 by the Eqn. 2.9, where F_t is the state transition model, B_t is the control-input model given as

$$\hat{x}_t^- = F_t x_{t-1} + B_t u_t + w_t. \tag{2.9}$$

The posteriori error covariance matrix P_t is calculated in Eqn. 2.10

$$\hat{P}_t = F_t P_{t-1} F_t^T + Q, \tag{2.10}$$

where Q is the covariance of the prediction model. The prediction update form is in the Eqn. 2.11, 2.12 and 2.13

$$K_t = \hat{P}_t H^T (H \hat{P}_t H^T + R)^{-1}, \tag{2.11}$$

$$\hat{x}_t = \hat{x_t}^- + K_t(z_t - H\hat{x}_t^-), \tag{2.12}$$

and

$$P_t = (I - K_t H)\hat{P}_t, \tag{2.13}$$

where K_t is the optimal Kalman gain at t state, H is the state observation model mapping the actual state space into the observation space, z_t is the observation value at t state and R is the variance of the observation data.

According to our assumption, $u_t = 0$, $w_t = 0$ and R is set to be 0.01 to reduce the change in position but optimize most in the velocity result.

2.3 Experimental Results and Discussion

The section demonstrates the result of the tracking. In the simulation, we used the algorithm to track a remote control submarine and fish in a $90cm * 240cm$ water tank. The experiment setup is shown in the figure 2.3. There were three pieces of glass above the tank, which blocked the view of the camera when the submarine passed through them. The Kalman filter was used to optimize the result of the position and the velocity.

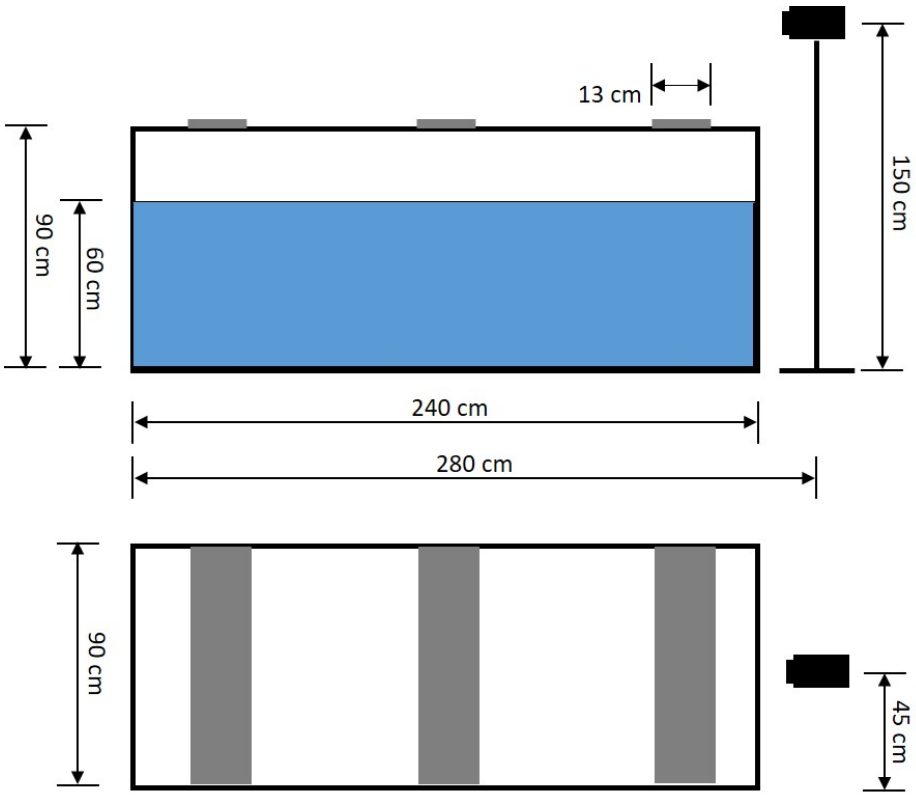

Figure 2.3: The setup of the experiment. The gray squares are the blur glass blocking the view of the camera. The blue square is the water.

2.3.1 Visual Tracking Results

Behind the blurred glass was the blind area. As shown in the figure 2.4, the submarine was outside of the blind area in the first frame. In the next three frames, when the submarine began to enter the blind area, part of it was blocked. In the last two frames, the submarine was blocked completely. In all six frames, the black cross was the result of the different methods in the different conditions classified by table 2.1

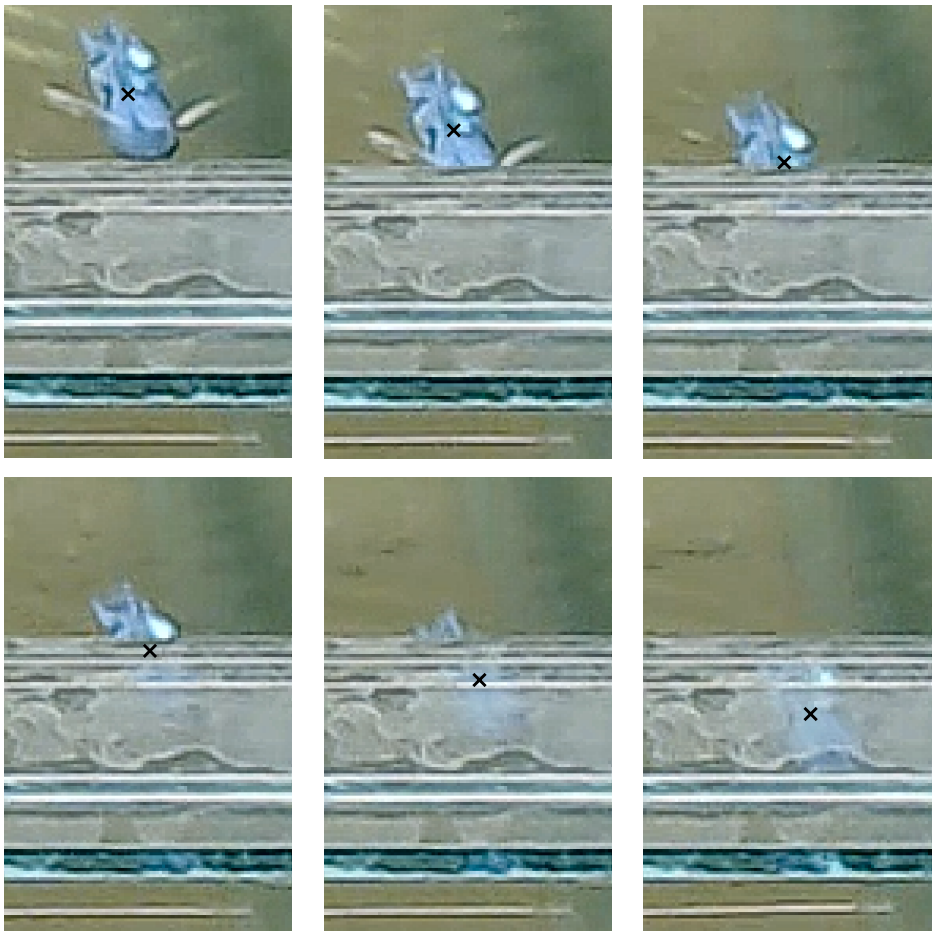

Figure 2.4: The tracking result of the blue submarine passing through the blocked area. The black cross represent the location of the object

In figure 2.4, we saw that even when part of the submarine was blocked by the glass, the cross was not at the center of the blue area(the part of the submarine which is outside of the blind area). Since we did not change anything on the remote controller, the submarine remained at the same velocity when passing through the blind area. When we used the same velocity to predict the position of the submarine, the result showed that the black cross was still near the center of the submarine. After we applied the tracking method to the simulation, we had the path and the velocity in x and y direction.

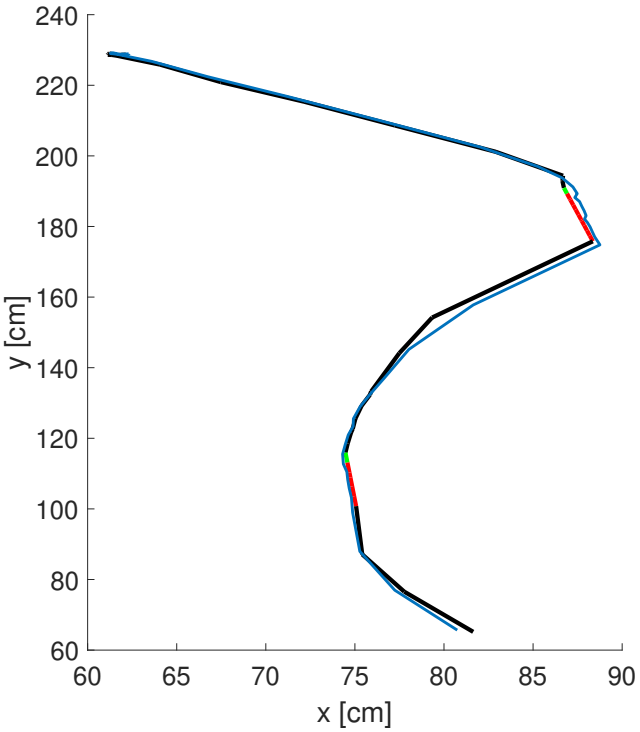

Figure 2.5: The path of the submarine in the simulation. The black line is the normal tracking result, the green line is the result given by the vector prediction, the red line is the result of location prediction with the constant velocity, the blue line is the result of Kalman filter

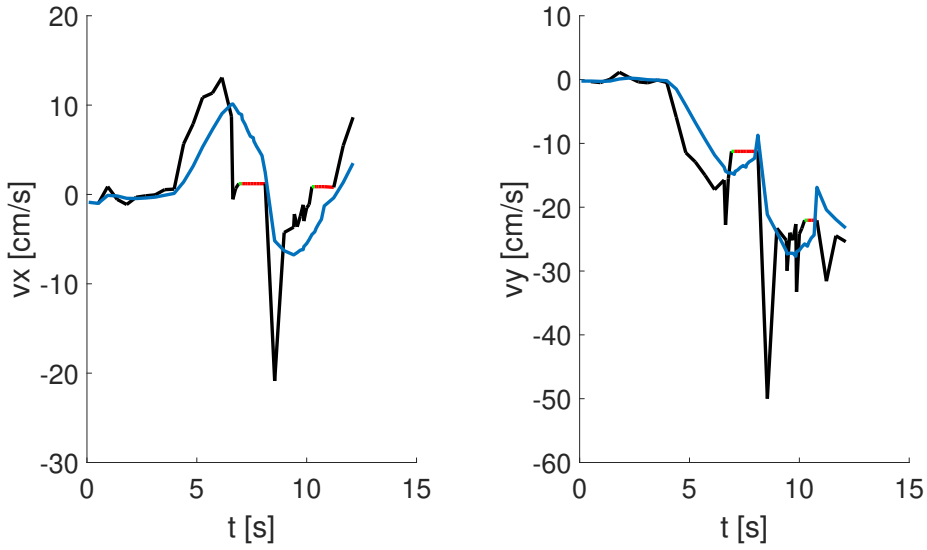

Figure 2.6: The left figure is the velocity of the submarine in x direction and the right figure is the velocity in y direction based on the tracking result.

As shown in the figure 2.5, the submarine first stayed at the same position. Due to the error in the image and the calculation process, it behaved like floating around a certain point. After it began for several seconds, it hit the glass, bounced back, and moved into the first blind zone. Since the hit occurred just after the submarine disappeared, it created a sudden velocity change in the blind zone, which did not satisfy our assumption of the same velocity in the blind zone. Because of that, the location prediction was not accurate in the blind area, which led to the large position change in a short time. After the submarine came outside of the blind area, the result was normal tracking before it entered the next blind area. In the second blind area, there was no velocity change, which gave a good location prediction result. According to the calculation of 8 sets of data, we chose the variance of position at the first time state, 0.01, and the result of Kalman filter in blue.

The velocity was plotted in figure 2.6. After four seconds, the velocity began to increase. Due to the drag force by the water, the acceleration was decreasing. When the submarine hit the boundary glass on the right side of the tank, the velocity on x direction has a large change while the change in y direction is not very large. After the vector prediction frames, the velocity became constant due to the constant location prediction in the blind area. When the submarine moved out of the blind area, there was a large change of location in a short time. When it moved in the second blind area, the velocity change was also large, compared with the constant velocity in the second blind area. After we applied the Kalman filter, the velocity was smoother than the original result. The fluctuation problem was solved, and the sudden drop in velocity became a gradual change.

In another simulation, we tracked the black auto fish. In figure 2.7, the red cross represents the position of the fish. The path of the fish in the water tank was shown in the figure 2.8, where the black color represented the normal tracking result, the brown was the vector prediction result when the small part of the object was blocked, the green was the vector prediction result when the large part of the object was blocked, the red was the location prediction when the whole object was blocked, and the blue line was the result after applying the Kalman filter.

Figure 2.7: The tracking result of the black auto fish passing through the blocked area. The red cross represent the location of the fish

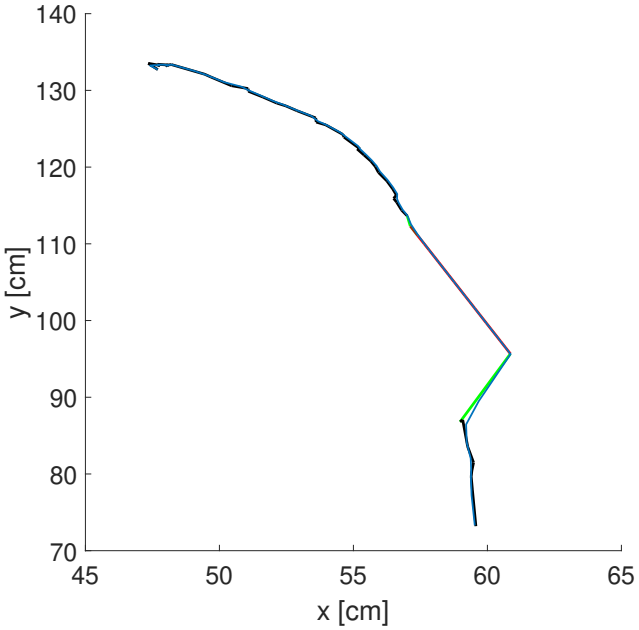

Figure 2.8: The path of the auto fish in the simulation. The black line the normal tracking result, the green line is the result given by the vector prediction, the red line is the result of location prediction with the constant velocity, the blue line is the result of Kalman filter

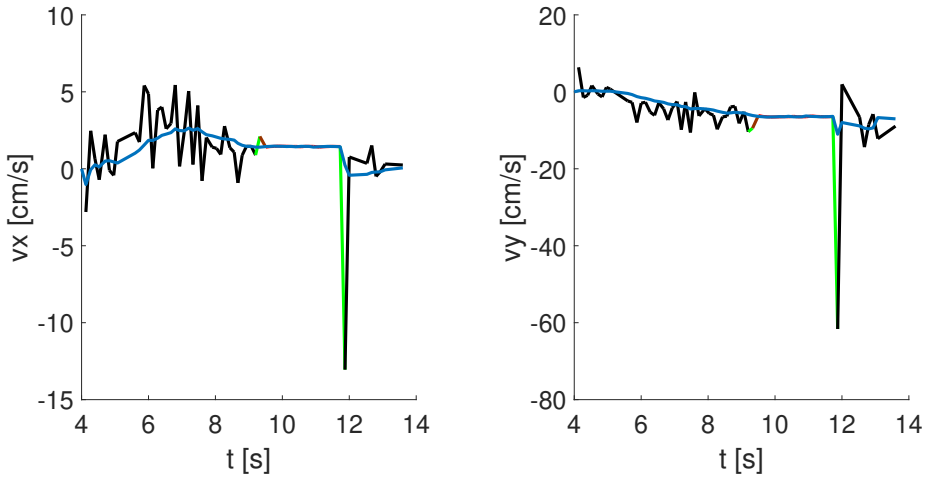

Figure 2.9: The left figure is the velocity of the auto fish in x direction and the right figure is the velocity in y direction based on the tracking result

2.3.2 Discussion

In the figure 2.5 and figure 2.7, the normal tracking gave an accurate result, but the prediction result introduced more error on the result. When a small part of the submarine was blocked, the direction of the current vector had a very small variation because of the change of shape. When a large part was blocked, the vector we used was the same as the one in the last frame. This prediction was valid only when the time was short between two frames and the change of the velocity was small. The result of this prediction had its own error plus the error from the previous frame. In the location prediction, we assumed the constant velocity, which was calculated in the last vector prediction result. The error of the location prediction was large because it accumulated all errors in the previous frames and prediction method, which caused a large location change when it changed back to the normal tracking method.

When we were trying to resolve pixels representing the object, due to the environment light change and the voltage fluctuation on the camera, the brightness of the object changed. When the brightness of the edge pixels decreased below the resolved threshold, the shape of the object was smaller. The shape variation caused the fluctuation in the length of the vector and the position of the object. The velocity fluctuation was amplified before the blind area. This was due to the smaller time steps and the same standard deviation on the position noise. Although the Kalman filter did not completely remove the experimental error, it avoided the vibration and the sudden change of the feedback signal and provided a more robust response in the control system.

Comparing two tracking results, in figure 2.4, the prediction had a better result than the result in the figure 2.7. In figure 2.7, when the fish was moving into the blind area, the prediction result did not mark the position precisely. This was because then the fish shook its body in order to swim forward. The vector representing the fish kept changing its direction, which caused the error in the vector prediction when it was moving into the blind area. This phenomenon is also reflected in the figure 2.9. In the normal tracking result, the path and the velocity in both x and y directions had a large fluctuation. In figure 2.8, the

vector prediction result shown as the red line deviated from the original direction, which also caused the location prediction to be less precise. In the signal time step, the fish did not have a stable moving direction. However, in the long-time tracking, it followed the order from the controller and moved to the given direction. The Kalman filter was applied to eliminate the fluctuation of the velocity and the path.

2.4 Chapter Summary

In this chapter, we develop a tracking method for an object on the water surface and derived the 2D position in world coordinate from the camera vision. The method is robust even there is an occlusion in the camera view.

We first generate the projection matrix, which can transfer the position of a pixel from image coordinates to the world coordinates. For the pixels that represent the objects, we classify them into four groups. We apply different methods for different groups. We use the minimum volume ellipse to have the vector to represent the shape and the heading angle of the object. When the object is partially blocked, the vector derived from the ellipse is used to predict the location of the object. If the object is blocked completely, we assume that there is no disturbance in the blind area and apply the Kalman filter to predict the position. This algorithm will be applied to real-time control for multiple objects.

Chapter 3

Cooperative Collision Avoidance Control of Servo/IPMC Driven Robotic Fish with Back-Relaxation Effect

In this chapter, a collision avoidance control strategy is developed for robotic fish, which has a two-joint fishtail. The first joint is a servo motor, which provides the forward thrust, while the second joint is an ionic polymer-metal composite (IPMC). Due to the complexity of the solid-fluid interaction, we identify an empirical model to represent the robotic fish with sufficient data. The empirical model is then integrated with a cone based collision avoidance control law to determine the IPMC voltages. We validate the algorithm by the experiment with two robotic fish [117].

3.1 Description of Servo/IPMC Driven Robotic Fish Back-Relaxation

This section first describes a robotic fish with an IPMC/Servo driven two joint fishtail. The back-relaxation of the IPMC, when subjected to different values of input voltage and immersed in water with different salinity levels, is then demonstrated and characterized. Finally, an intuitive explanation of the IPMC's bending mechanism is given based on the observation.

3.1.1 Robotic Fish Description

The robotic fish is shown in Fig. 3.1. The fish has two joints: the first joint is actuated by a servo motor, which provides the main propulsion. The second joint is an IPMC, which controls the direction of the fish's movement. The fish is divided into head and tail portions. The head portion contains the battery, circuits, H-bridge, regulator, and an Arduino microcontroller which communicates with the control center through WiFi. The tail portion contains the servo motor, the IPMC actuator, and a passive plastic fin. The dimensions of the fish are shown in Fig. 3.1.

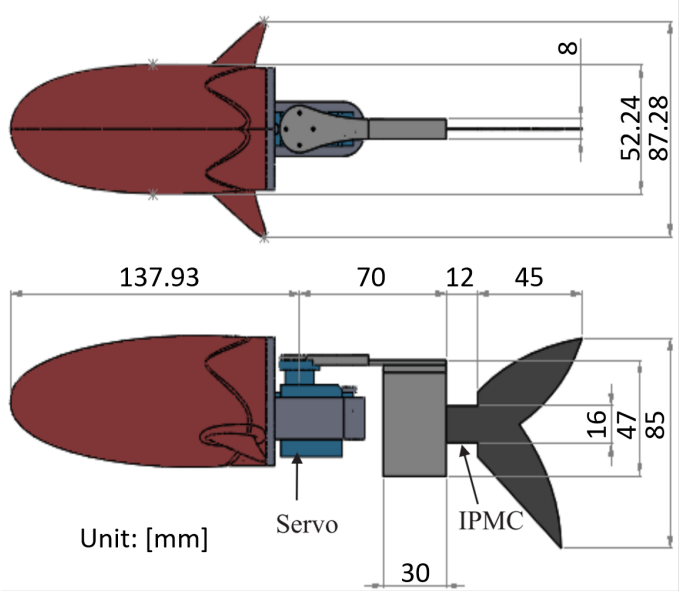

Figure 3.1: The servo-motor/IPMC-driven robotic fish design with dimensions.

3.1.2 IPMC Actuation Mechanism

IPMCs have been used as artificial muscles due to their lightweight, flexibility, and their shape changes under the influence of an electric field [118]. As shown in the upper figure of Fig. 3.2, when an IPMC is submerged in a wet environment, the positive ions of the IPMC attach to the water molecules and move freely. When a voltage is applied across the IPMC, in the first stage, the positive ions together with the water molecules move to the cathode side, and this induces a bending effect on the IPMC [119].

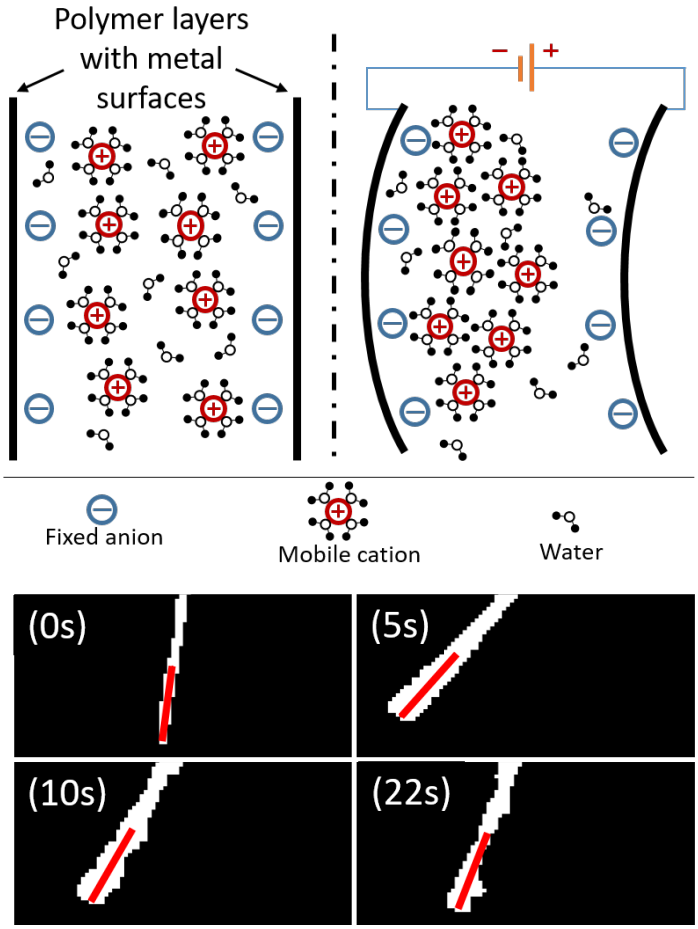

Figure 3.2: (a) The IPMC actuation mechanism; (b) Snapshots of IPMC bending, where the red line represents the direction at the end of the slice.

3.1.3 Experimental Observation of IPMC Back-relaxation

In order to capture the back-relaxation process in the IPMC, we fixed the fishtail to a frame and connected only the IPMC to the voltage from the microcontroller. The servo-motor was not powered. The voltage supplied to the IPMC was controlled by the value of the pulse width modulation (PWM) duty cycle, where analog values in the range 0-255 were scaled to 0-9 (this number is referred to as the PWM value in the rest of the paper). The mapping between the PWM values and the corresponding output voltages is given in table 3.1. In order to avoid burning the IPMC while performing long-duration tests, PWM values of 7 or higher are not used in these experiments.

Table 3.1: Output voltages for different values of PWM duty cycle.

PWM Value	1	2	3	4	5	6
Voltage (V)	2.80	3.22	3.75	4.10	4.33	4.85

In the experiments, the IPMC was submerged in water and subjected to a constant PWM value. The bending response of the IPMC was captured by a video camera (Logitech webcam c920). The captured video was then analyzed through an image processing algorithm to characterize the angle at the end of the IPMC slice. The IPMC was a gray/black metal slice. Given that the background was white paper, it was easy to determine a proper threshold for the color.

Fig. 3.2(b) shows the experimental results for a case when the PWM value is 6 (voltage is 4.85 V). As shown in this figure, four binary images of the IPMC's position are snapshots at different time instants of 0 s, 5 s, 10 s, and 22 s, respectively. The white pixels in each sub-figure were then fitted using a second-order polynomial. The red line in each sub-figure shows this fitting line at the end of the IPMC slice.

It is seen in Fig. 3.2(b) that the bending angle of the IPMC slice first reached its maximum value (towards the left side) around 10 seconds and then reduced in the "back-relaxation" process. The time interval over which the bending angle increased was shorter than that of the "back-relaxation" process. This demonstrates that the bending angle of

an IPMC (when subjected to a constant voltage) reaches its maximum value quickly in a short time and then decreases slowly. In order to characterize the back-relaxation under different DC voltages, three more tests were conducted with the PWM values set at $-6, 3$, and -3. The experimental results of these tests are shown in Fig. 3.3(a), wherein the black line (whose y-axis is on the left side) depicts the bending angle of the IPMC versus time. This angle was calculated from the arc-tangent of the direction at the end of the slice (red line in Fig. 3.2). The dashed line (whose y-axis is on the right side) represents the DC voltage applied to the IPMC. The applied voltages are 3.75 V, -3.75 V, 4.85 V and -4.85 V, corresponding to PWM values of $3, -3, 6$ and -6, respectively.

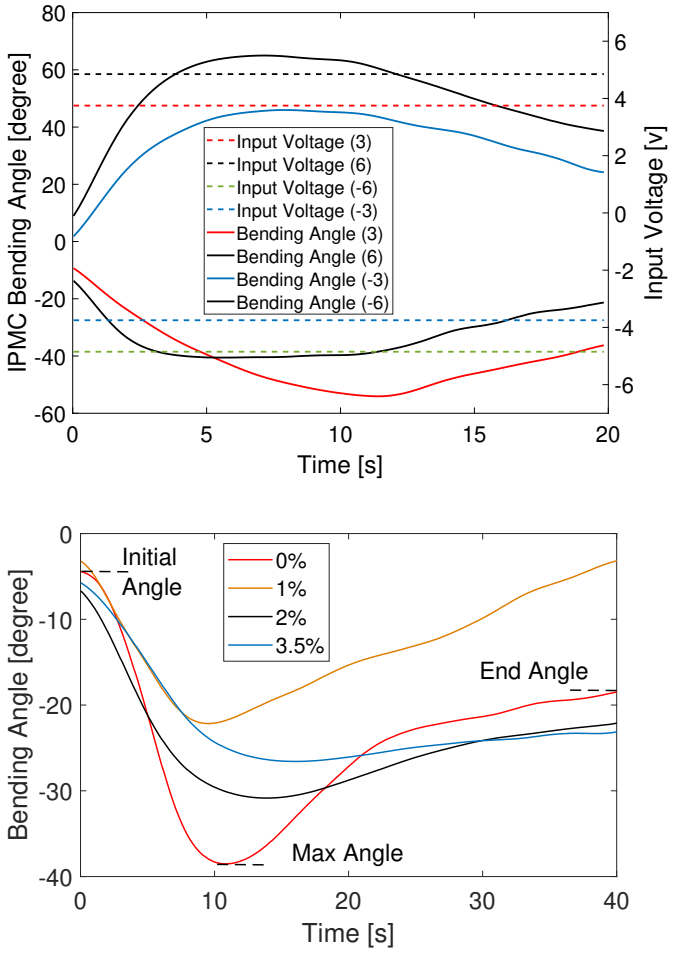

Figure 3.3: Experimental observation of IPMC back-relaxation. (a) IPMC bending angle versus time when a DC voltage is applied; (b) IPMC bending angle versus time under 3 PWM values with the water salinity levels at 0%, 1%, 2%, and 3.5%.

As shown in Fig. 3.3(a), for the PWM value of 3, it took 6 to 8 seconds for the IPMC slice to reach its maximum bending angle, after which it started the "back-relaxation" process. For the PWM value of 6, it took only 3 to 4 seconds for the IPMC to reach its maximum bending angle. This shows that the higher the voltage, the faster the IPMC bends to its peak angle value.

3.1.4 Salinity Level Effect on IPMC Back-Relaxation

We also tested the back-relaxation drop-down percentage with different water salinity levels. The Fig. 3.3(b) shows sample results of the IPMC bending angle for 0%, 1%, 2%, and 3.5% (seawater) salinity levels. The lines are smoothed by a Gaussian filter. The back-relaxation percentage is calculated as

$$\eta = 100\frac{\text{(Max\_angle - End\_angle)}}{\text{(Max\_angle - Initial\_angle)}}\%, \tag{3.1}$$

where the Initial, Max, and End angles are defined in Fig. 3.3(b). The mean back-relaxation percentage for each salinity level is shown in table 3.2, where the percentage of bending angle drop decreases as the salinity level increases. This shows that the salinity level of the environment affects the back-relaxation drop-down degree. Capturing the quantitative relationship between them needs more research and experiments, which will be conducted in our future work.

Table 3.2: Average IPMC back-relaxation drop down percentage under different salinity levels.

Salinity	0%	1%	2%	3.5%
Degree	99.62%	59.46%	36.03%	17.21%

3.1.5 Intuitive Explanation for IPMC Back-Relaxation

An intuitive explanation for the "back-relaxation" phenomenon is that the entry of water from outside to the anode side neutralizes the unbalanced water concentration in the

IPMC thus reducing the bending effect. According to prior research on IPMC [119], most water molecules move along with the positive ions to the cathode side, thereby causing the right side of the IPMC to lose water, and this leads to the bending of the IPMC to the anode side. Given that the IPMC is immersed in water, the outside water is absorbed by the IPMC from its anode side, and this gradually reduces the bending effect and causes the "back-relaxation". When these two processes reach an equilibrium, the bending angle reaches its steady state. Based on this interpretation, when the electric field is removed after the steady-state is reached, positive ions and water molecules move back to their original position. The water, which is absorbed by the anode side of the slice, gradually moves out. Although the voltage of the electric force is reduced to zero suddenly [120, 121], the molecular movement due to the voltage change is faster than the concentration change. The slice first bends to the opposite side and then slowly returns back to its original position. For the same reason, if the voltage is reversed after the steady-state, the maximum bending angle is larger than the bending angle when the voltage jumps from zero volts.

3.1.6 Back-relaxation Effect on Turning Dynamics of the Fish

When the IPMC is used to generate flapping motion of the tail [122], the back-relaxation effect does not have a significant effect since the voltage applied on the IPMC keeps changing all the time. However, in this paper, since the IPMC joint is used to generate a bending displacement for turning motion, the IPMC voltage might be kept for a long period. The influence of the IPMC back-relaxation has a significant effect on the turning dynamics of the robotic fish. Thus this effect needs to be fully investigated and modeled. Because the multi-physics involved in back-relaxation and the fluid-structure-interaction of the fishtail are fully coupled, it is too complicated to obtain an accurate and practical physics-based model to capture the turning dynamics introduced by the back-relaxed IPMC joint. This paper adopts a data-driven modeling approach to capture those dynamics, and these are discussed in the next section.

3.2 Data-driven Modeling of Robotic Fish

In this section, we discuss how the experiments were setup, and data was collected and processed to identify the data-driven model of the fish's turning dynamics.

3.2.1 Experimental Setup

In order to collect sufficient experimental data for the data-driven model, we conducted several free-swimming robotic fish tests. The objective of these tests was to determine the relationship between the applied IPMC voltage and the turning response of the robotic fish. The experimental setup is shown in Fig. 3.4. The experiment pool was 136 cm $\times$ 165 cm and a Logitech webcam $c920$ was placed 40 cm away from one side of the pool. Power was provided by a 9 V battery. Control commands calculated on a laptop were sent to a $ESP8266$ micro-controller through WiFi to generate the voltages applied to the servo-motor and the IPMC, respectively. The micro-controller and the battery were placed in the fish head. The IPMC connected with the fishtail was fixed on the rotor controlled by the servo motor. The surface of the whole fish was covered with a waterproof coat. A Para-film strip was used to connect the head to the fish body. Vaseline was used to cover all the electrical connecting parts to prevent any possible electrical leakages. In one of these experiments, a PWM value of 3 (3.75 V) was applied to the IPMC, while the flapping frequency and flapping amplitude of the servo motor were set at 0.5 Hz and 28 degrees, respectively. With these parameters, a test video was filmed at 30 Hz for 60 seconds.

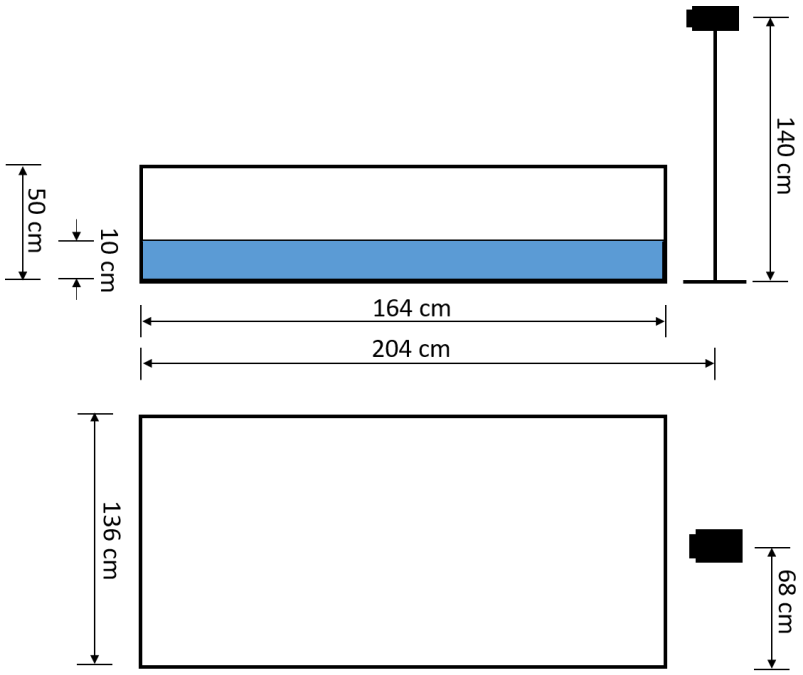

Figure 3.4: Experimental setup.

3.2.2 Data Collection using Visual Tracking Algorithm

To obtain the data-driven model of robotic fish, the relative position and velocity were extracted from the frames of the video. The green color of the fish is easily recognized in the image tracking algorithm [123]. The image tracking algorithm comprises of several steps: 1) Get the first frame of the video and pick 4 reference points on the water surface with known positions in the real world coordinates, 2) Set the searching windows around the object being tracked (IPMC fish), 3) In each searching window, use a color threshold to pick up the main body of the object, 4) Use the minimum volume ellipse surrounding the pixels of the object and pick the intersections between the long diameters of the minimum volume ellipse and the edge of the pixels as the vector to represent the object, 5) Apply a Kalman filter [113, 124] to smooth the position and get an estimate of velocity, 6) After obtaining the velocity from the Kalman filter, we take another time derivative of the velocity and obtain the normal acceleration based on Eqn (3.2). This algorithm reduces the localization error coming from the water fluctuation and reflection noise. The system control updating time is 0.2 s, which meets the computational requirement of this algorithm. To observe how the

IPMC's bending influences the direction of the fish's motion, we extracted the information of the fish's position, angular velocity, and the acceleration normal to the instantaneous velocity vector, from the captured video. The position was extracted directly from the video, the velocity was the time derivative of the position along the x and y directions, and the normal acceleration was calculated based on the velocity vector of the fish '$\overrightarrow{V}$'

$$\overrightarrow{a}(t) = \frac{d\overrightarrow{V}}{dt} - \frac{d\|\overrightarrow{V}\|}{dt}\frac{\overrightarrow{V}}{\|\overrightarrow{V}\|}. \tag{3.2}$$

3.2.3 Data-Driven Model Identification and Verification

In the experiment, step voltages of different amplitudes were applied to the IPMC, and the fish's trajectories were recorded by the camera. The remaining conditions were the same as in previous experiments. The PWM values for the tests were 5, 4, 3, and 1, respectively. Fig. 3.5 shows one set of data when the PWM value was 3 (the voltage was 3.75 V). Fig. 3.5(a) shows the trajectory of the robotic fish, as it moved upwards from the bottom.

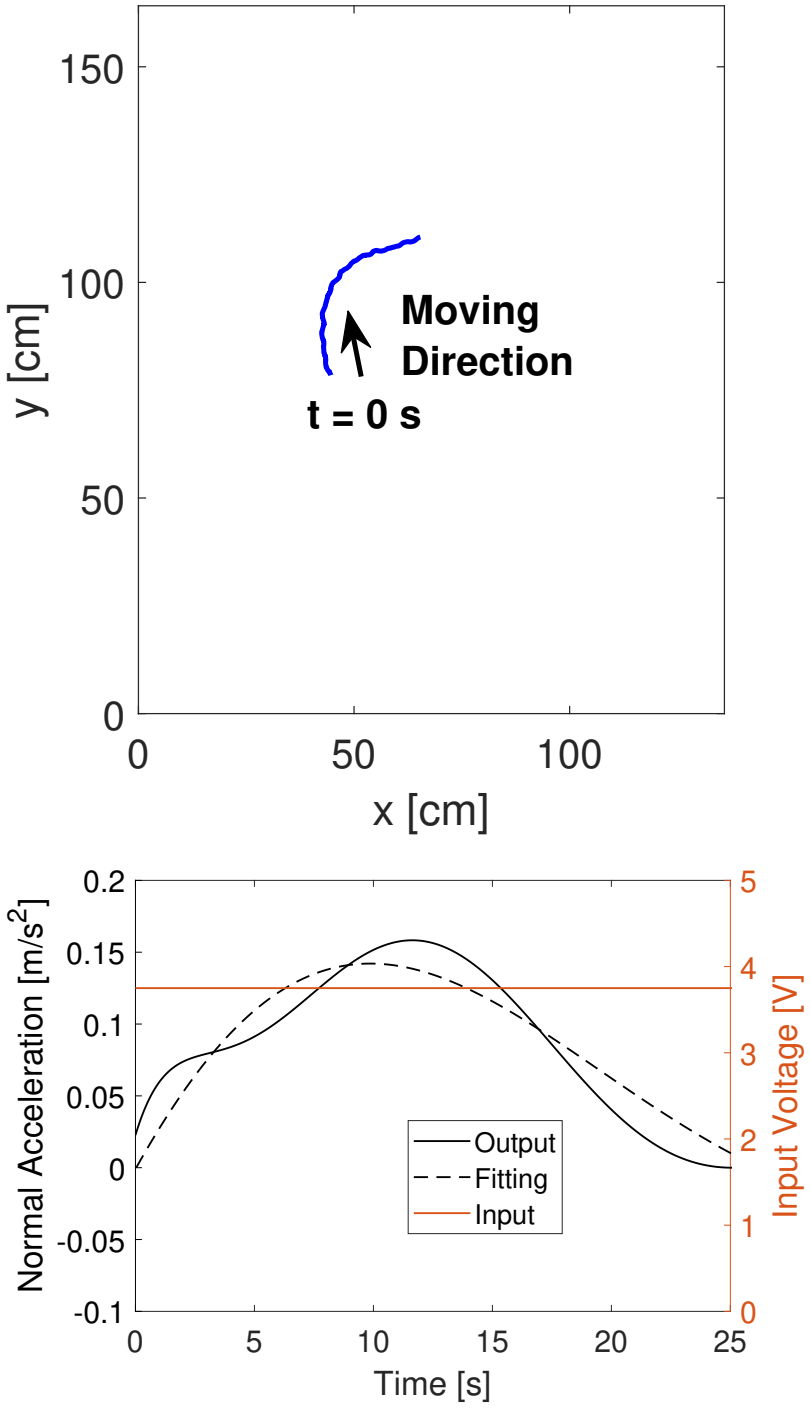

Figure 3.5: Experimental identification of data driven modeling: (a) Trajectory of the robotic fish with a 3.5 V voltage applied to the IPMC joint; (b) Normal acceleration of the robotic fish.

Fig. 3.5 (b) demonstrates the normal acceleration obtained in the experiment (black line)

when a 3.5 V voltage was applied to the IPMC. The normal acceleration value increases during the first 10 seconds, after which it subsequently decreases. This result coordinates with the IPMC test in Subsec. 3.1.2 which shows that under a unit step voltage, the IPMC bending angle increases fast in the first few seconds and then slowly decreases. This back-relaxation behavior is captured by a second-order transfer function. In our previous work, we developed a physics-based dynamic model of IPMC/Servo robotic fish [118]. However, the back-relaxation phenomenon was not considered in that model, and that model was also not practical for control design purposes. In this paper, we use an empirical model to capture the turning dynamics of the robotic fish when it swims at a constant speed, and a long-duration voltage is applied to IPMC.

To obtain such an empirical model, the first step is to identify the model structure. Based on the experimental evidence of Fig. 3.5, we infer that the transfer function that relates the IPMC voltage $V_0(s)$ to the magnitude of the normal acceleration has the following structure

$$\frac{\|\overrightarrow{a}(s)\|}{V_o(s)} = \frac{Ks}{s^2 + 2\zeta\omega_n s + \omega_n^2}. \tag{3.3}$$

A nonlinear fitting toolbox in Matlab was used to identify the parameters in the transfer function. Since the original acceleration data had some fluctuations, a Gaussian filter was used prior to fitting the data. The fitted normal acceleration is shown in Fig. 3.5(b) (blue line).

Since the fish dynamics vary under different conditions, such as salinity level and condition of IPMC, the parameters of the empirical model can change. To obtain a nominal model for control design, we collected sixteen sets of data and obtained the statistical characteristic of the identified parameters in terms of mean value and standard deviation. The identified parameters are shown in Fig. 3.6, where K has a mean value of 0.00716, the second coefficient $2\xi\omega_n$ has a mean value of 0.09526, and ω_n^2 has a mean value of 0.01634. The mean values of the parameters were used for the control algorithms in the remaining experiments. The standard deviations will be used to perform a robustness analysis of the control system in our future work.

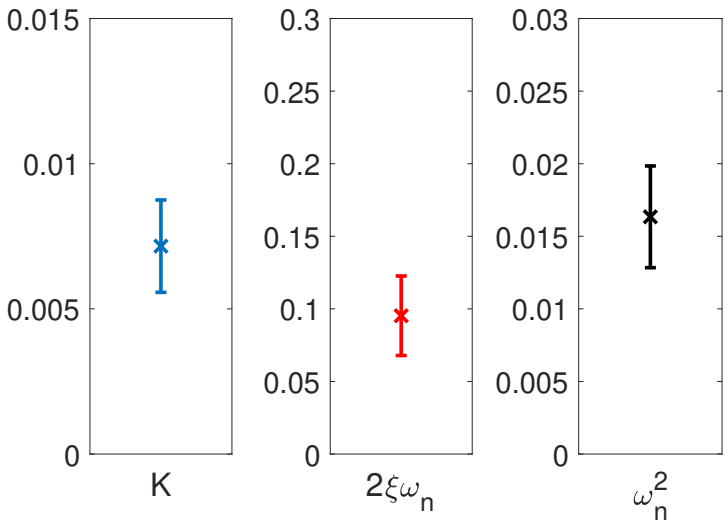

Figure 3.6: The mean value and the standard deviation of the identified parameters.

3.3 Collision Avoidance Control Design

To achieve collision avoidance, which accounts for the back-relaxation of the IPMC driving the robotic fish, we develop a kinematic model-based collision avoidance control. In this section, we determine analytical nonlinear guidance laws by virtue of which the robotic fish can employ its IPMC actuator to achieve collision avoidance. Consider two circular objects A and B, moving with speeds V_A and V_B, and heading angles α and β, respectively, as shown in Fig. 3.7. Let P_1 and P_2 represent the centers of these circles, and let R_A and R_B be their respective radii. The kinematics of the engagement between A and B are characterized by the following equations

$$\dot{r} = V_r, \tag{3.4}$$

$$\dot{\theta} = V_\theta / r, \tag{3.5}$$

$$\dot{V}_\theta = -V_\theta V_r / r - a_A \sin(\delta_A - \theta) + a_B \sin(\delta_B - \theta), \tag{3.6}$$

and

(3.7)

$$\dot{V}_r = V_\theta^2/r - a_A \cos(\delta_A - \theta) + a_B \cos(\delta_B - \theta). \tag{3.8}$$

Here, r and θ represent, respectively, the range and the bearing angle of the line P_1P_2. V_θ and V_r represent, respectively, the relative velocity components (of B with respect to A) that are normal to, and along, P_1P_2. a_A and a_B represent, respectively, the magnitudes of the accelerations of A and B, while δ_A and δ_B represent the angles at which these accelerations are applied. We assume that these accelerations act normal to the respective velocity vectors, that is, $\delta_A = \alpha + \frac{\pi}{2}$ and $\delta_B = \beta + \frac{\pi}{2}$.

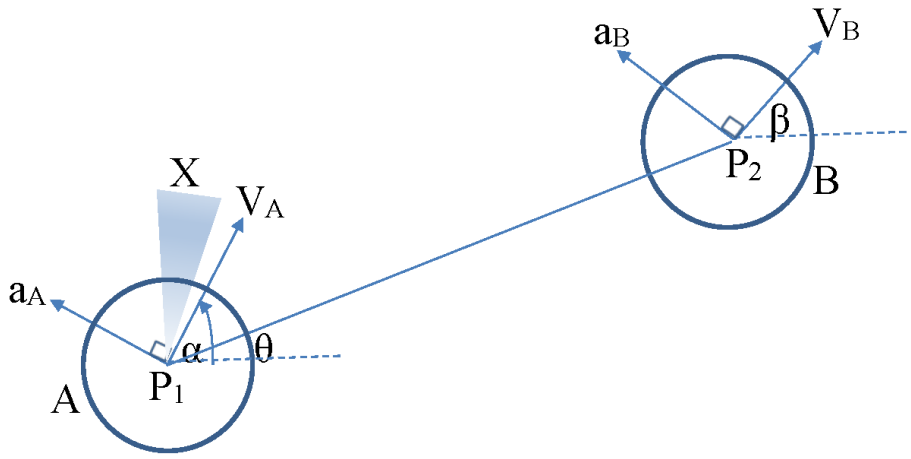

Figure 3.7: Engagement between two robotic fish A and B.

As shown in [125], the collision cone between A and B is defined as the instantaneous cone of heading angles of A that will cause A to lie on a collision course with B. For circular objects, the collision cone is encapsulated in a scalar quantity of y, defined as follows

$$y = \frac{r^2 V_\theta^2}{V_r^2 + V_\theta^2} - (R_A + R_B)^2. \tag{3.9}$$

The collision cone is then defined as

$$\mathcal{X} = \{\alpha : y < 0 \ \cap \ V_r < 0\}. \tag{3.10}$$

In Fig. 3.7, a schematic representation of the collision cone $\mathcal{X}$ is provided. The condition $y < 0, V_r < 0$ corresponds to the physical scenario of the relative velocity vector being inside the collision cone. When $y = 0, V_r < 0$, this corresponds to the physical scenario wherein the relative velocity vector is aligned with the boundary of the collision cone.

Recall the transfer function (3.3), which relates the IPMC voltage to the normal acceleration of the robotic fish. Let z_1^A and z_2^A represent the two states (associated with this transfer function) for robotic fish A, and z_1^B, z_2^B represent the corresponding two states for robotic fish B. Note that z_2^A and z_2^B are the same as the normal accelerations of robotic fish A and B, respectively. We combine these states (obtained from 3.3) with the kinematic equations (3.8) to obtain the following equations for the engagement between two IPMC-actuated robotic fish (whose shapes are modeled by circular objects) as follows

$$\begin{aligned}
\dot{r} &= V_r \\
\dot{\theta} &= V_\theta / r \\
\dot{V}_\theta &= -V_\theta V_r / r - z_2^A \cos(\alpha - \theta) + z_2^B \cos(\beta - \theta) \\
\dot{V}_r &= V_\theta^2 / r + z_2^A \sin(\alpha - \theta) - z_B^2 \sin(\beta - \theta) \\
\dot{\alpha} &= z_2^A / V_A \\
\dot{\beta} &= z_2^B / V_B \\
\dot{z}_1^A &= z_2^A \\
\dot{z}_2^A &= -2\zeta\omega_n z_2^A - \omega_n^2 z_1^A + KVo_A \\
\dot{z}_1^B &= z_2^B \\
\dot{z}_2^B &= -2\zeta\omega_n z_2^B - \omega_n^2 z_1^B + KVo_B,
\end{aligned} \tag{3.11}$$

where, Vo_A and Vo_B represent the IPMC voltages applied to robotic fish A and B, respectively. The voltages (Vo_A, Vo_B) need to be such that they drive y in (3.9) to zero before the time of closest approach, as this is equivalent to A and B having achieved successful collision avoidance. By differentiating (3.9) along the trajectories of the system of Eqn. 3.11,

we get the following

$$\begin{aligned}\dot{y} &= \Big[\frac{\partial y}{\partial r}\dot{r} + \frac{\partial y}{\partial V_\theta}\dot{V}_\theta + \frac{\partial y}{\partial V_r}\dot{V}_r\Big] \\ &= z_2^A\Big[\frac{\partial y}{\partial V_r}\sin(\alpha-\theta) - \frac{\partial y}{\partial V_\theta}\cos(\alpha-\theta)\Big] \\ &- z_2^B\Big[\frac{\partial y}{\partial V_r}\sin(\beta-\theta) - \frac{\partial y}{\partial V_\theta}\cos(\beta-\theta)\Big]. \end{aligned} \tag{3.12}$$

Defining N_1 and N_2 as follows

$$\begin{aligned} N_1 &= \frac{\partial y}{\partial V_r}\sin(\alpha-\theta) - \frac{\partial y}{\partial V_\theta}\cos(\alpha-\theta) \\ N_2 &= \frac{\partial y}{\partial V_r}\sin(\beta-\theta) - \frac{\partial y}{\partial V_\theta}\cos(\beta-\theta). \end{aligned} \tag{3.13}$$

We rewrite (3.12) as follows

$$\dot{y} = z_2^A N_1 - z_2^B N_2. \tag{3.14}$$

Differentiating the above again with respect to time, we get

$$\begin{aligned}\ddot{y} &= z_2^A\dot{N}_1 - z_2^B\dot{N}_2 - (2\zeta\omega_n z_2^A + \omega_n^2 z_1^A)N_1 \\ &+ (2\zeta\omega_n z_2^B + \omega_n^2 z_1^B)N_2 + KVo_A N_1 \\ &- KVo_B N_2. \end{aligned} \tag{3.15}$$

We then see that, by using a dynamic inversion control design, we enforce y to follow the dynamics

$$\ddot{y} + K_1\dot{y} + K_2 y = 0, \tag{3.16}$$

if we choose the IPMC voltage pairs (Vo_A, Vo_B) as follows

$$\begin{aligned} K(N_1 Vo_A - N_2 Vo_B) &= -K_1\dot{y} - K_2 y - z_2^A\dot{N}_1 + z_2^B\dot{N}_2 \\ &+ (2\zeta\omega_n z_2^A + \omega_n^2 z_1^A)N_1 \\ &- (2\zeta\omega_n z_2^B + \omega_n^2 z_1^B)N_2. \end{aligned} \tag{3.17}$$

In the above equation, if we now choose $K_1 > 0$ and $K_2 > 0$, it is guaranteed that y will decay to zero. Furthermore, these values need to be chosen such that y decays to zero before the time of closest approach of A and B. In (3.17), the variables $\dot{N}_1, \dot{N}_2$ are defined as follows

$$\begin{aligned}
\dot{N}_1 &= (\frac{z_2^A}{V_A} - \frac{V_\theta}{r})\Big[\frac{\partial y}{\partial V_r}\cos(\alpha-\theta) + \frac{\partial y}{\partial V_\theta}\sin(\alpha-\theta)\Big] \\
&+ XJT_A \\
\dot{N}_2 &= (\frac{z_2^B}{V_B} - \frac{V_\theta}{r})\Big[\frac{\partial y}{\partial V_r}\cos(\beta-\theta) + \frac{\partial y}{\partial V_\theta}\sin(\beta-\theta)\Big] \\
&+ XJT_B. \qquad (3.18)
\end{aligned}$$

The intermediate terms J, T and T_A that are used in $\dot{N}_1$ and $\dot{N}_2$ are the following

$$\begin{aligned}
J &= \begin{bmatrix} \frac{\partial^2 y}{\partial r^2} & \frac{\partial^2 y}{\partial r V_\theta} & \frac{\partial^2 y}{\partial r V_r} \\ \frac{\partial^2 y}{\partial r V_\theta} & \frac{\partial^2 y}{\partial V_\theta^2} & \frac{\partial^2 y}{\partial V_r V_\theta} \\ \frac{\partial^2 y}{\partial r V_r} & \frac{\partial^2 y}{\partial V_r V_\theta} & \frac{\partial^2 y}{\partial V_r^2} \end{bmatrix} \\
X^T &= \begin{bmatrix} V_r \\ -V_\theta V_r/r - z_2^A\cos(\alpha-\theta) + z_2^B\cos(\beta-\theta) \\ V_\theta^2/r + z_2^A\sin(\alpha-\theta) - z_2^B\sin(\beta-\theta) \end{bmatrix} \\
T_A &= \begin{bmatrix} 0 \\ -\cos(\alpha-\theta) \\ \sin(\alpha-\theta) \end{bmatrix} \\
T_B &= \begin{bmatrix} 0 \\ -\cos(\beta-\theta) \\ \sin(\beta-\theta) \end{bmatrix}. \qquad (3.19)
\end{aligned}$$

Eqn (3.17) thus represents the set of IPMC voltage pairs (Vo_A, Vo_B) with which the robotic fish A and B cooperatively achieve collision avoidance with one another.

The closed loop system is stable, and this can be seen as follows. We note that the

voltage inputs employed for performing collision avoidance maneuvers, are chosen such that the quantity $y(t)$ defined in Eqn (3.9) follows the stable dynamics shown in Eqn (3.16). Therefore, $y(t)$ and $\dot{y}$ are stable variables, and they remain bounded. Since $y(t)$ and $\dot{y}$ depend on $r(t)$,$V_\theta(t)$,$V_r(t)$ and their derivatives, boundedness of $y(t)$ and $\dot{y}$ automatically implies boundedness of $r(t)$, $V_\theta(t)$, $V_r(t)$ and their derivatives.

We note that (3.17) represents a single equation in two unknowns Vo_A and Vo_B. We assume $Vo_A = Vo_B$ and solve this equation for a single unknown. Performing such an implementation would correspond to a centralized controller, and each robotic fish would need to know the internal IPMC state of the other robotic fish. Alternatively, we also implement (3.17) in a decentralized fashion, by writing it as two separate equations as follows

$$\begin{aligned} KN_1Vo_A = &-K_1 z_2^A \Big[\frac{\partial y}{\partial V_r}\sin(\alpha-\theta) - \frac{\partial y}{\partial V_\theta}\cos(\alpha-\theta)\Big] \\ &- K_2\frac{y}{2} - z_2^A\dot{N}_1 + (2\zeta\omega_n z_2^A + \omega_n^2 z_1^A)N_1 \end{aligned} \tag{3.20}$$

and

$$\begin{aligned} KN_2Vo_B = &-K_1 z_2^B \Big[\frac{\partial y}{\partial V_r}\sin(\beta-\theta) - \frac{\partial y}{\partial V_\theta}\cos(\beta-\theta)\Big] \\ &+ K_2\frac{y}{2} - z_2^B\dot{N}_2 + (2\zeta\omega_n z_2^B + \omega_n^2 z_1^B)N_2. \end{aligned} \tag{3.21}$$

In (3.20)-(3.21), Vo_A does not depend on the internal IPMC state of fish B and Vo_B does not depend on the internal IPMC state of fish A. Assuming each fish has an on-board vision sensor and an IMU, it can use the image information (fused with its IMU readings) to determine r, θ, V_r, V_θ and the normal acceleration of the other fish. When the robotic fish possesses such a sensor suite, decentralized collision avoidance using (3.20)-(3.21) is performed.

3.4 Experimental Validation of Collision Avoidance Control

In this section, we perform an experimental validation of the collision avoidance control laws. We fabricated two servo motor/IPMC actuated robotic fish. The flapping frequency and the amplitude are the same as the fish in Sec. 3.2. A block diagram is shown in Fig. 3.8. In Fig. 3.8, the miss distance is represented by y, which is calculated using Eqn (3.9). The desired miss distance is set to zero, and this corresponds to the two circles grazing each other. The difference between the desired miss distance and the actual miss distance is regarded as the control error. The parameters in (3.9) are obtained from camera data. The collision avoidance controller generates the PWM value from the error and the voltage to the IPMC of each robotic fish. The IPMC voltages follow Eqn (3.17) and are such that the two circles will graze each other, and the quantity y follows the stable dynamics given in Eqn (3.16).

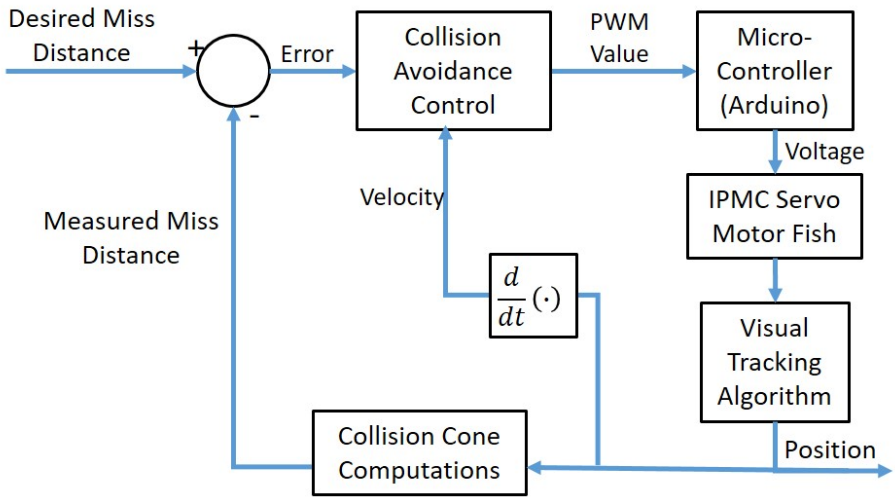

Figure 3.8: The block diagram of experiments.

Fig. 3.9 shows three snapshots from the video of an experiment, at 10 s (top figure), 14 s (middle figure) and 26 s (lower figure), respectively.

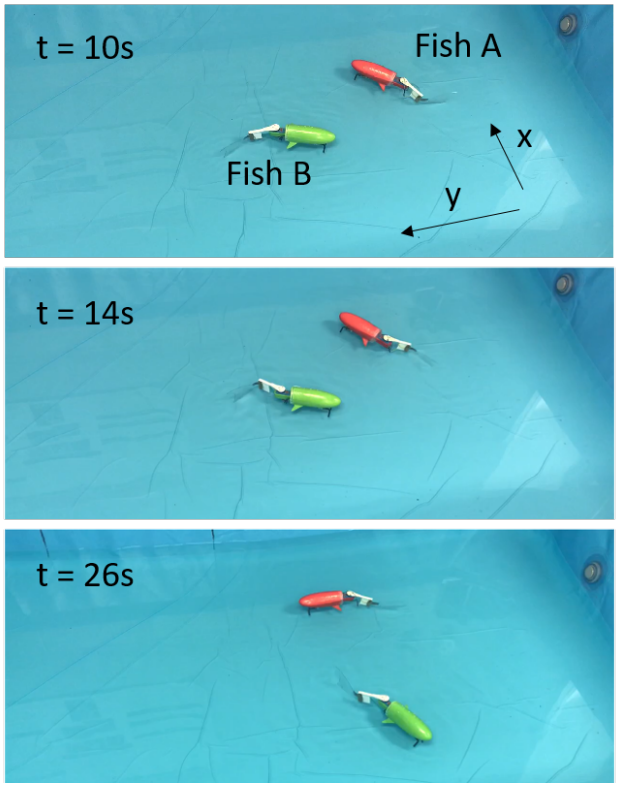

Figure 3.9: Snapshots of the experiment video. Threes frames were taken at 10 s, 14 s and 26 s, from top to bottom, respectively.

Fig. 3.10(a) shows the trajectories of the two robotic fish in the experiment. Here, the red line represents the smoothed trajectory of Fish A, the green line represents that of Fish B, and the radius of the bounding circle of each fish is $R = 8.0$ cm. Fig. 3.10(b) represents the time history of V_r in Eqn (3.8) (black solid line) and y in Eqn (3.9) (orange solid line). The PWM value is recorded at the lower right-hand-side figure. The swing of the fish heads causes some fluctuations in the PWM value. Since $V_{OA} = V_{OB}$, these two PWM values are the same. In order to protect the control circuits, a saturation limit of 5 was applied to the PWM. In Eqn (3.9), $y < 0$ reflects that the relative velocity vector is located inside the collision cone, and $V_r < 0$ indicates that the distance between the two fish is reducing. During the time interval $10 - 16$ s, y and V_r are both negative, which means the fish are on a collision course. The experiment validates that the collision avoidance control algorithm worked well on two fish and reveals how parameters in the algorithm reflect the control results. Most tracking errors came from water reflection and fluctuation. Besides that, the

original tracking path was not smooth due to the shake of the fish body. The minimum volume ellipse method [123] and the Kalman filter was used to reduce those errors.

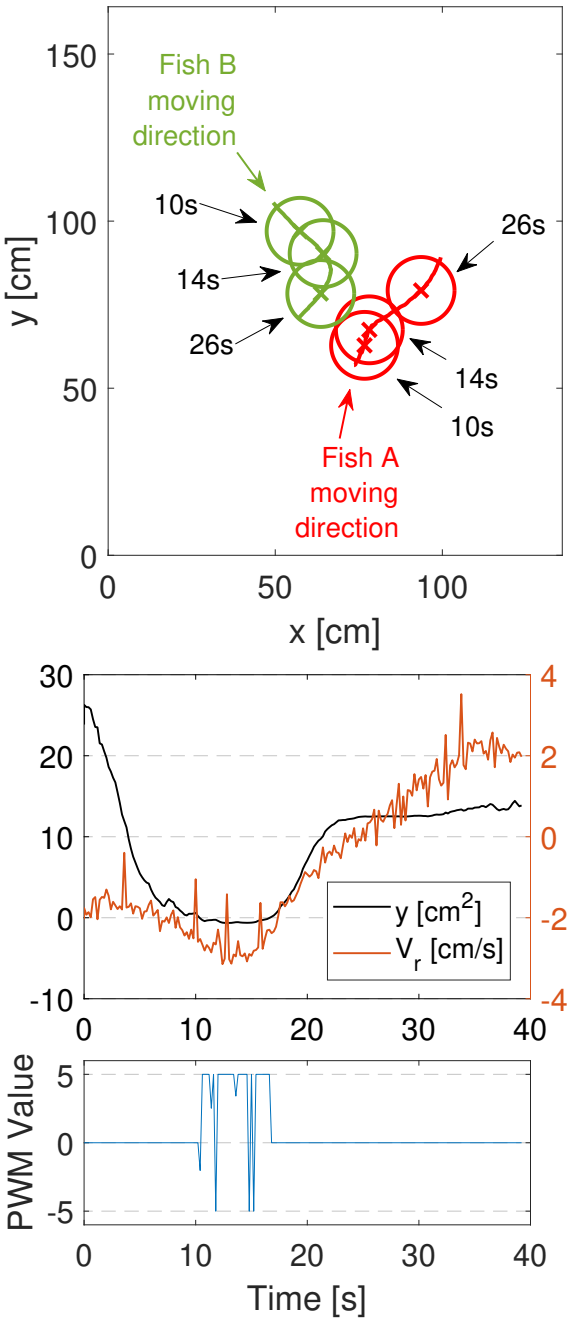

Figure 3.10: Experimental results: (a) The path of two fish under the influence of collision avoidance control, (b) the time histories of y, V_r and PWM.

3.5 Chapter Summary

In this chapter, the back relaxation characteristic is considered in an IPMC/Servo driven robotic fish, with the development of a model-based collision avoidance control. We first verify the IPMC "back-relaxation" phenomenon by experiments. Next, we test on robotic fish to find the turning behavior of the fish when the IPMC is under a constant voltage. From the experiment, a transfer function model relating the IPMC voltage input to the normal acceleration is identified. After that, we introduce a collision avoidance algorithm, which considers both the relative velocity kinematics of the fish to obstacles and the robotic fish system's dynamics. We apply the collision avoidance control to two robotics fish to validate the control through several experiments.

Chapter 4

Robotics Assisted Smart-Touch Pipeline Inspection

A timely inspection of pipelines is important to prevent oil & gas leaking. This chapter develops a novel robotic system that automatically grabs the flange and performs bolt looseness inspection autonomously. The robotic system has a four degree of freedom robotic arm with a stereo camera system and a pair of piezoceramic transducers. In the experiment, when the robotic arm grabs the flange, a stress wave signal is generated by one piezoceramic transducer and received by the other one, which indicates whether the flange is tightly bolted or not. The experiment results have validated this autonomous robotic inspection approach [102].

4.1 Visual Servoing System

This section discusses theories and preparation for the visual servoing detection, including the system description, the camera position allocation, the train and detection process of the YOLO networks, the localization according to the bounding box and flange feature.

4.1.1 System Description

In this research, the flange grasping system is a KINOVA ultra-lightweight robotic arm with 4 DC motor actuators on a four-wheel mobile platform. As shown in Fig. 4.1, there are four actuators. The hand with three fingers is jointed to the arm with actuator 4. In order to provide the location information for the flange, a 3D-printed frame for four cameras is mounted on the plastic ring of actuator 4. If the frame is mounted on actuator 1, 2, or 3, it is possible that obstacles in a complex environment block cameras. Besides, when cameras are close to the object, they are able to observe the texture or crack on the surface. The parameters of the robotic arm are listed in table 4.1, where D_1 represents the distance between base and actuator 2, D_2 is the distance from actuator 2 to actuator 3. D_3 shows the length from actuator 3 to the fingers. The KINOVA built-in sensor provides the direction and the angle of each actuator. The angles are defined as ϕ, θ_1 and θ_2 for actuator 1, 2 and 3, respectively. Based on the structure of the robotic arm, the cylinder coordinate is suitable to the system, where the rotation angle is ϕ, and the vertical and radial axis, z and r are derived from θ_1 and θ_2. The Cartesian coordinate at the robotic arm base transfer the vector from the stereo camera coordinate.

In Fig. 4.1, the mobile platform uses suspensions for the wheels. The maximum speed is 30-rpm for the gear motor of each tire, which generates a 0.39 $N \cdot m$ torque. With parallel sets of wheels, the mobile platform could steer the body by rotating wheels on both sides oppositely. The vehicle is manually controlled by a remote joystick controller. The communication is through an $nrf24$ radio transmitter. The micro-controller translates the command into a pulse width modulation (PWM) signal and sends the signal to two $L298n$ H-bridge motor drivers. Motors on the one side share the same signal. Two $14.8v$ Lipo batteries with voltage regulators provided power for circuits and motors.

Table 4.1: Parameters for the robotic arm

Parameters	Description	Length (m)
D_1	Base to shoulder	0.2755
D_2	Upper arm length (shoulder to elbow)	0.4100
D_3	Elbow to wrist	0.2037
D_4	wrist to figure center	0.1600

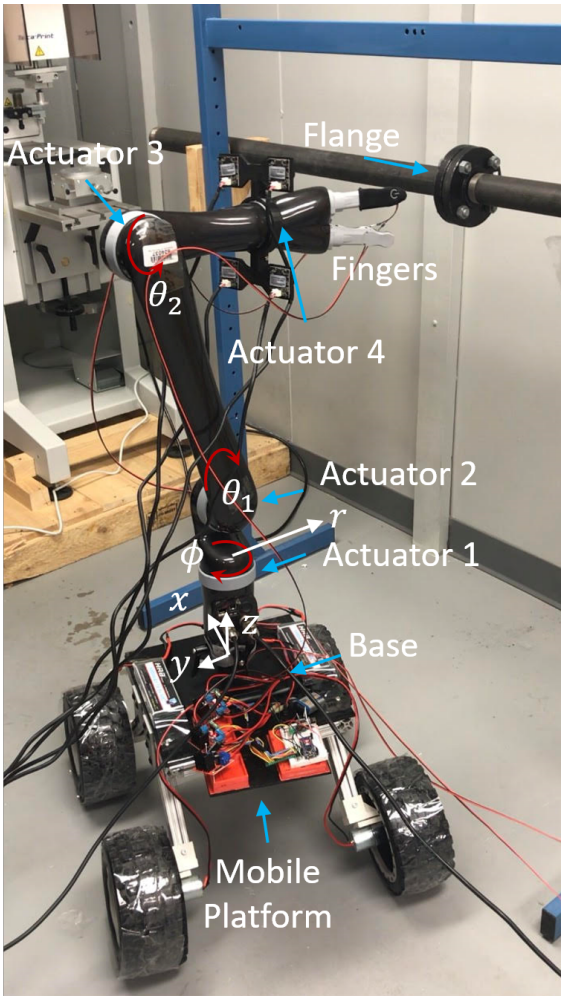

Figure 4.1: Robotic Arm with 4 cameras stereo systems.

4.1.2 Camera Position Allocation

In the research, there are four cameras for 3D localization. The installation of the camera follows Fig. 4.2. The frame of four cameras is on the wrist of the robotic arm. The section view from the back elbow to the front finger is shown in the left of Fig. 4.2. For a pair of cameras in a stereo system, the left one is always regarded as the main camera. For example, in the pair of left up (a) and right up (b) cameras in the red dashed line box, the origin of the relative coordinate is at the focus point of the left up camera. In application, the object position to the robotic arm base v_{bo} follows

$$\mathbf{v}_{bo} = \mathbf{v}_{bf} + \mathbf{v}_{fc} + \mathbf{v}_{co}, \tag{4.1}$$

where $\mathbf{v}_{bf}$, $\mathbf{v}_{fc}$, and $\mathbf{v}_{co}$ are the relative vector from the base to the finger center, the finger center to the camera, and the camera to the object, respectively. In Eqn. 4.1, $\mathbf{v}_{bf}$ is from the dynamic parameters of the robotic arm, and $\mathbf{v}_{co}$ is the detection result from stereo camera. $\mathbf{v}_{fc}$ is unknown when the baseline of the stereo camera (x'-axis) is not horizontal. A solution is to put cameras in symmetric positions so that the center of symmetric cameras is always in the wrist position, which is known from the dynamic parameters. Thus, Eqn. 4.2 becomes

$$\mathbf{v}_{bo} = \mathbf{v}_{bf} + \mathbf{v}_{fr} + \mathbf{v}_{ro}, \tag{4.2}$$

where all vectors are in the coordinates of Fig. 4.1. The module and the direction of $\mathbf{v}_{fr}$ are from table 4.1 the dynamic parameters of the robotic arm, and $\mathbf{v}_{ro}$ is the relative vector from the wrist to the object, which is the average value of the object's position from all camera pairs. Due to the manual frame assembly and camera frame deformation, the optical axis(z'-axis) is not parallel to the centerline of the forearm. This problem introduces errors to the $\mathbf{v}_{ro}$ calculation. The solution to this problem is to upgrade the traditional dual stereo camera to four cameras system so that there are twelve pairs of stereo cameras (permutation of four cameras). The average value of twelve 3D positions reduces the error in the whole localization process.

A four-camera system requires more computation cost compared with a stereo camera system. However, in object localization, which is the most time-consuming process, the YOLO network is faster than other traditional image processing methods. Besides, the grasping motion focuses on accuracy rather than time efficiency. Thus, it is worth updating the camera system.

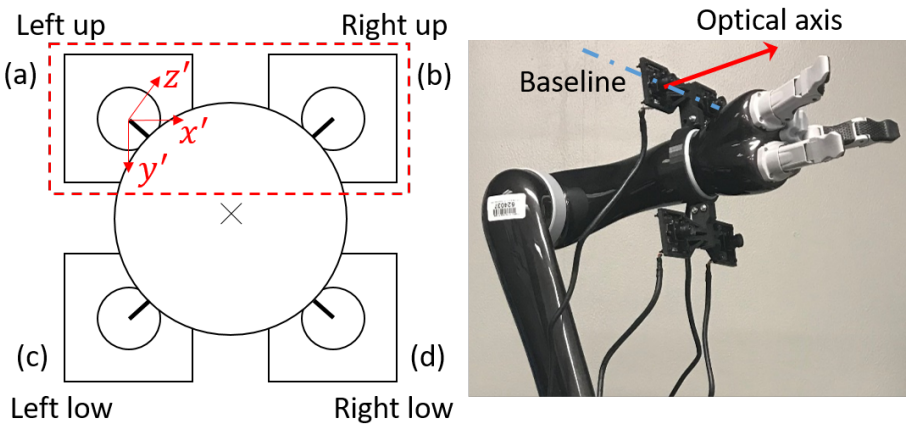

Figure 4.2: The installation of cameras.

4.1.3 YOLO Localization

To provide solid and consistent feedback information for robotic arm automatic control, we apply YOLO to classify and localize the flange. There are four major steps for the training process: 1) preparing samples; 2) building the network; 3) training the network; 4) validating the trained network. Training samples of the network require labeled images with the bounding box of the object. There are 944 flange images for the same flange with different distances, directions, brightness, and background view in the research. We manually draw bounding boxes for 944 flange images with the size (480, 640, 3).

The YOLO network layer in this research is derived from ResNet-50 CNN architecture [126]. The network consists of three convolution blocks (ConvBlock), a residual network block composed of convolution, batch normalization, and ReLU activation layers. Each convolution block is followed by an identified block(gray). The three identity blocks include 2, 3, 5 residual network blocks. Compared with the convolution block, the identity block

does not have one convolution and batch normalization block at the skipped connection. The activation function after the convolution block and identity block is followed by the YOLO convolution, YOLO batch normalization, and YOLO ReLU layers. The output layer computes the flange position with 0.72 intersection over union(IoU).

4.1.4 Localization in images

The position points in images are the key parameters to build the 3D coordinates. A traditional way is to regard the center of the bounding box as the position. However, due to different angles of the flange in the image and the size of the bounding boxes, the center of the bounding box may deviate from the real center of the flange. As shown in Fig. 4.3, the orange cross is the bounding box center. The orange cross is in the connection place of the flange, which is not the center position of the flange. Moreover, the bounding box center changes for different boxing boxes, which means that on the same flange, the positions based on the bounding box center are different. In the 3D view, this difference is amplified and introduces large errors to the stereo localization. To avoid this situation, this paper introduces an 'improved localization method', which relies on the features of the flange in localization. Since the flange is a structure where two pipelines are mounted by screws, there is little light in the narrow gap. When this is reflected in the flange image, the middle area of the flange is darker than the area around it. Thus, the first step of the 'improved localization method' is to find the center of the bounding box; the next step is to build a flat rectangle searching area, which takes the bounding box center as the center; the final step is to find the geometry center of dark pixels with the least $l-2$ norm RGB value in the searching area and set it as the position of the flange.

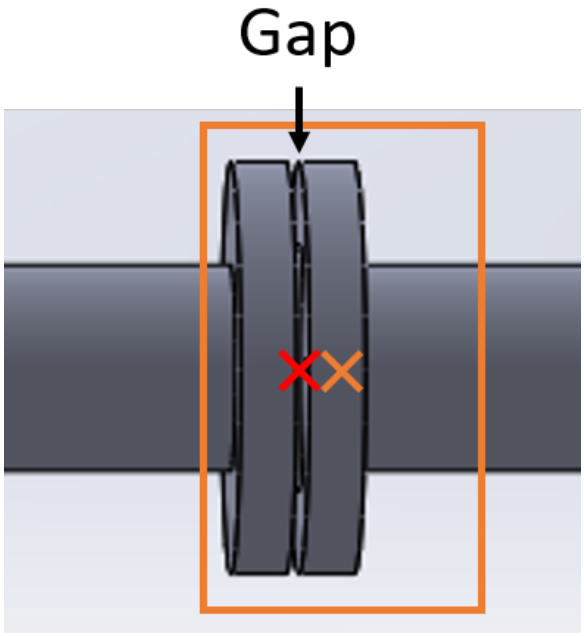

Figure 4.3: The detected position of the flange with different angle by bounding box center.

4.1.5 Training & Detection Result

To train the YOLO network, we apply stochastic gradient descent with a momentum (SGDM) update algorithm. The training process was finished on MATLAB with NVIDIA Quadro P2000 GPU. The training performance is shown in Fig. 4.4, where the mini-batch loss and the validation loss are the blue and orange lines for the left and right y-axis, respectively. The mini-batch size is 16. The max epochs number is 120. The overall iterations number is 4320. The learning rate is 10^{-3}. The final mini-batch loss and validation loss are less than 0.013 and 0.045. In Fig. 4.4, the mini-batch loss and the validation loss are close to zero.

The detection result of four cameras is in Fig. 4.5, where the number above the yellow bounding box is the detection confidence. The translucent white area in the middle of the bounding box is the flat box for picking the position of the flange. The yellow and the red crosses are the center of the bounding box and the position of the flange, respectively.

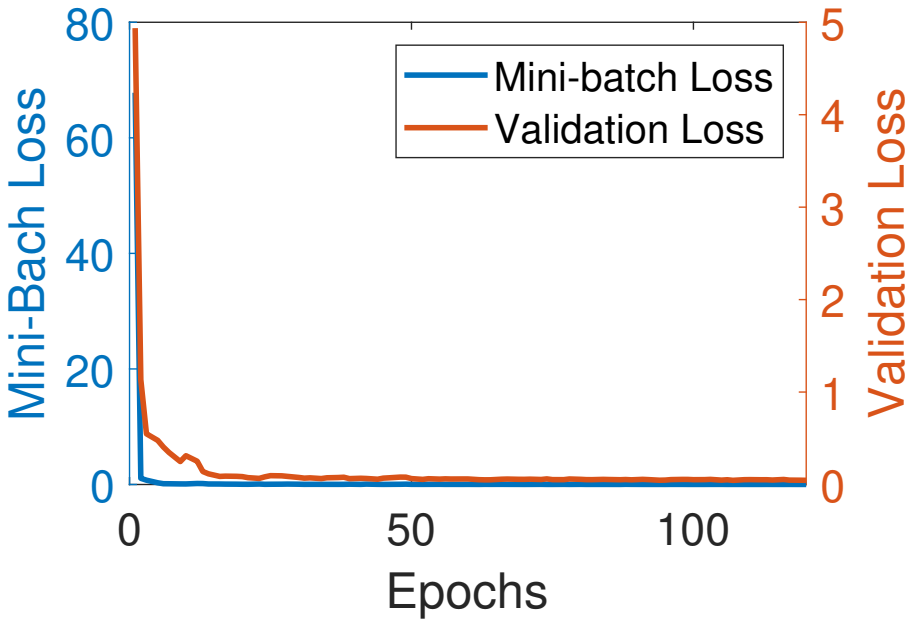

Figure 4.4: The training process of the network with mini-batch loss and validation loss.

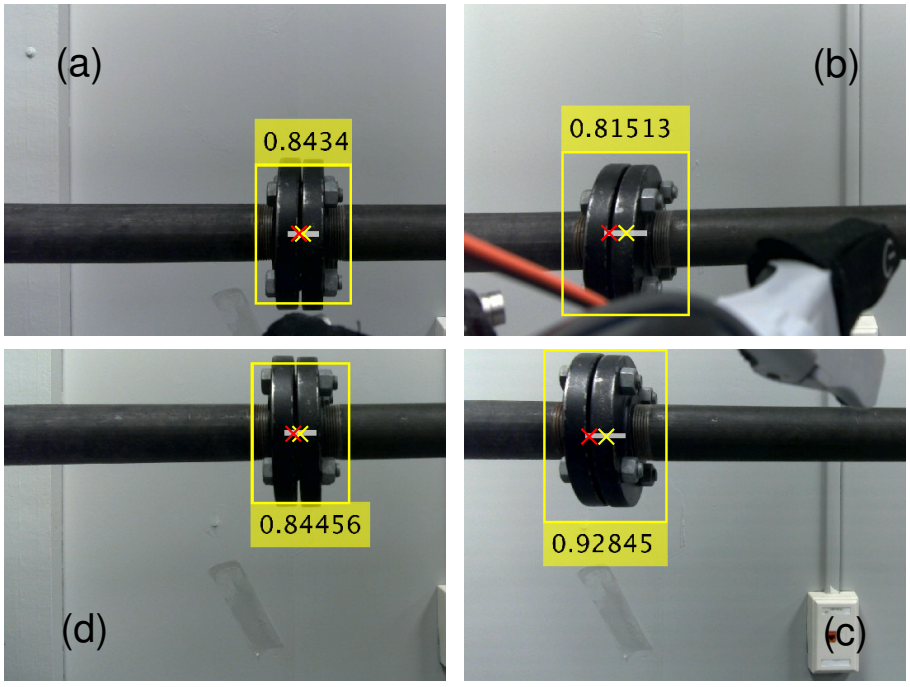

Figure 4.5: YOLO network detection result, where yellow rectangles are bounding boxes, the values above are the detect confident, the yellow cross are the center of the bounding box, the flat white box is the region for flange position, and the red cross is the position of the flange.

In Fig. 4.5, compared with the result from the traditional method(yellow cross), the revised position(red cross) is all in the middle area of the flange, which means they approximately represent the same point in the 3D view. The position range(white inside box) is flat is due to the stereo camera localization mechanism [60, 65], where the depth of the target is derived from the coordinates in the horizontal direction in both cameras. Based on the red cross position, the stereo camera computes the 3D localization as feedback information for the robotic system control, which will be discussed in the next section.

4.2 Visual Servoing Control

This section includes the kinematic model of the robotic arm, the control system, and the final grasping motion. The procedures are demonstrated in Fig. 4.6. When the camera system is able to detect the flange through the YOLO networks, the system follows the 'Control 1'. The output flange position in world coordination is the reference for 'Control 2', which is applied when the camera loses the detection of the flange in the image view.

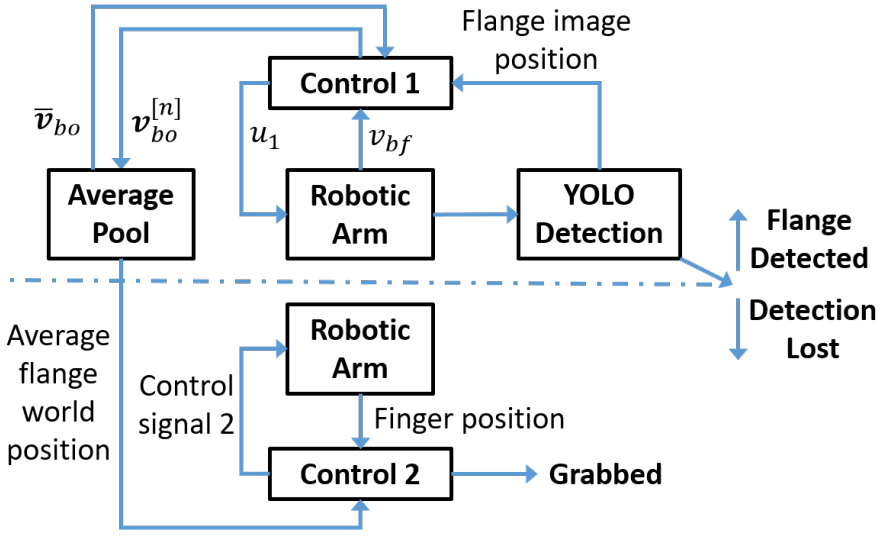

Figure 4.6: Control system diagram.

4.3 Flange Detected

The structure of 'Control 1' is in Fig. 4.7, where the boxes with the solid line are the components of the 'Control 1', and the dashed line boxes are corresponding parts in Fig. 4.7. The stereo camera parameters, which are introduced in Sec. 4.1, outputs the 3D position in camera coordinates for each stereo camera pair. The 'Average pool' is to calculate the 3D vector from the wrist to the flange in the camera's coordinates '$\mathbf{v}'_{ro}$' where the origin point is at the wrist. The calculation is that

$$\mathbf{v}'_{ro} = \sum_{i=1}^{n} (\mathbf{v}'_{coi} R_{\alpha i}), \tag{4.3}$$

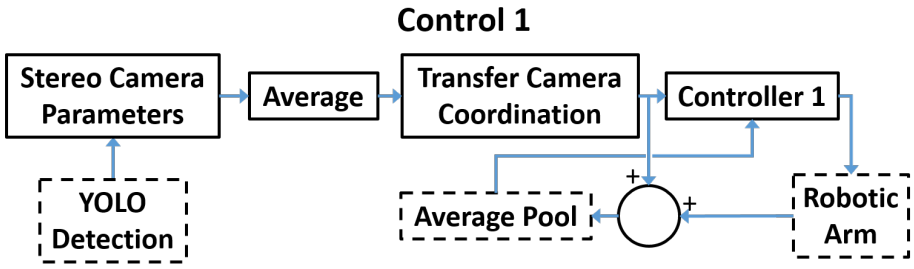

Figure 4.7: Control 1 structure

where $\mathbf{v}'_{coi}$ is the vector from the ith pair of stereo camera to the flange (ith pair in the permutation of camera (a)(b)(c)(d)), and $R_{\alpha i}$ is the rotation matrix from the ith pair of stereo camera coordinates to the default one (the pair (a)(b) in Fig. 4.2).

The 3-D world position from cameras is based on the focal plane and optical axis, while v_{ro} in Eqn. 4.2 is in the base coordinates. Thus, coordinate transformation follows

$$\mathbf{v}_{ro} = R_{cylinder} R_z(\phi) R_y(\theta_1 - \theta_2) R_I \mathbf{v}'_{ro}, \tag{4.4}$$

where $R_{cylinder}$ is the transform matrix from Cartesian to cylinder coordinates; $R_y(.)$ and $R_z(.)$ are rotation matrices for y and z axis, respectively; R_I is the replacement from camera

coordinates $x'y'z'$ to the world coordinates xyz. R_I is represented as

$$R_I = \begin{bmatrix} 0 & -1 & 0 \\ 0 & 0 & -1 \\ 1 & 0 & 0 \end{bmatrix}.$$

The robotic arm has inner-loop and outer-loop controllers. The inner-loop controller controls motors and moves joints to command positions, which is computed from the robotic arm dynamic system. Kinova robotics company embeds a proportional, integral, and derivative(PID) controller into the actuator systems. In the inner-loop, the reference is a desired position. The feedback information is the current position, which is measured by the angle sensor. The PID controller from the error generates the control signal for motors. In the application, the gains are $K_P = 2.0$, $K_I = 0.01$ and $K_D = 0.05$.

The outer-loop control is shown in Fig. 4.6. To control the displacement is equivalent to control the velocity in constant time. Thus the controller 1 is given as

$$u_1 = P(\bar{\mathbf{v}}_{bo} - \mathbf{v}_{bf}^{[n]}), \tag{4.5}$$

where u_1 is the control signal for controller 1, $\mathbf{v}_{bf}^{[n]}$ is the position vector from the base to the robotic arm finger center in the nth iteration, and P is the proportional gain. To guarantee the stability of the system and avoid overshoot, we set the range of p as (0, 1). The $\bar{\mathbf{v}}_{bo}$ in Eqn. 4.5 is computed as

$$\bar{\mathbf{v}}_{bo} = \frac{\sum_{i=1}^{n}(\mathbf{v}_{bo}^{[i]})}{n}, \tag{4.6}$$

where $\mathbf{v}_{bo}^{[i]}$ is the vector $\mathbf{v}_{bo}$ at ith iteration.

In application, v_{bo} is affected by errors in vision localization and camera assembly. The position of the flange is the summation of the finger center position and the object position from the stereo camera system in base coordination based on Eqn. 4.2. According to Eqn. 4.4, the error related to the camera is amplified by the 3-D distance between the object and the cameras. From the start position of the robotic arm and the final grasping

position, the most distance change is on r axis (cylinder coordination in Fig. 4.1), which is the direction towards the object. In the experiment, an average pool collects the position data and computes the average position of the object in every control iteration.

4.3.1 Flange Detection Lost

If the cameras on the robotic arm are near the object, the flange is outside the frame edge, and the cameras completely lose the object. In this situation, the automation system applies 'Control 2' in Fig. 4.8, which is:

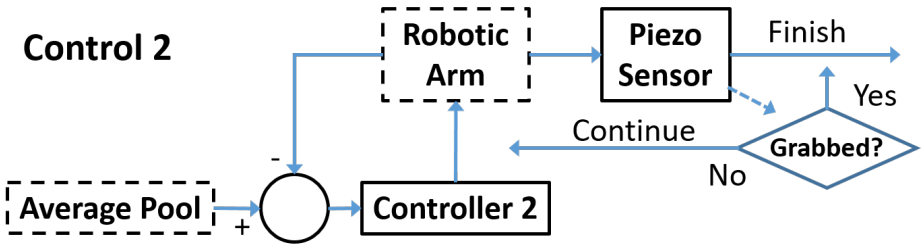

Figure 4.8: Control 2 structure

When images no longer provide useful information, the average pool stops receiving flange position, and the average value is the reference for controller 2. Since the robotic arm has a signal threshold, which means if the input voltage is too small, it does not move. Thus the controller 2 is given as

$$u_2 = \begin{cases} 0 & \text{Grabbed} \\ \max(P(\bar{\mathbf{v}}_{bo} - \mathbf{v}_{bf}^{[n]}), r_g - 2r_s), & \text{Not grabbed} \end{cases} \tag{4.7}$$

where u is the control signal, r_g is the radius of the region where the piezo-sensor gives clear feedback, r_s is the radius of the piezo-sensor. $r_g - 2r_s$ is the minimum control signal of u_2 before the finger touches the flange, which is to prevent overshooting and protect the piezo sensor. In every control iteration of this algorithm, we close the finger to grab the object. The signal from the piezo-sensor indicates whether the grasping process is finished or not. If the signal wave did not indicate successful grasping motion, the controller loop continues.

4.4 Piezoelectric Transducers

Monitoring the load on bolted joints can prevent the bolt from loosening and ensure the safety and reliability of structures. A method is to detect bolt looseness by load cells or strain gauges. Besides that, the piezoelectric transducer has been wildly used in bolted looseness monitoring. In this paper, the stress wave signal analysis is based on the theory in [127, 128, 129]. As shown in Fig. 4.9, when waves are propagating through the bolted joint surface, the energy dissipation is affected by interface integrity. Thus, the signal energy receives by the piezoelectric sensor varies when ultrasonic waves pass through the bolted joint interface under different pre-loads. The bolt looseness could be calculated by revealing the relationship between the response wave energy and the bolt axial load.

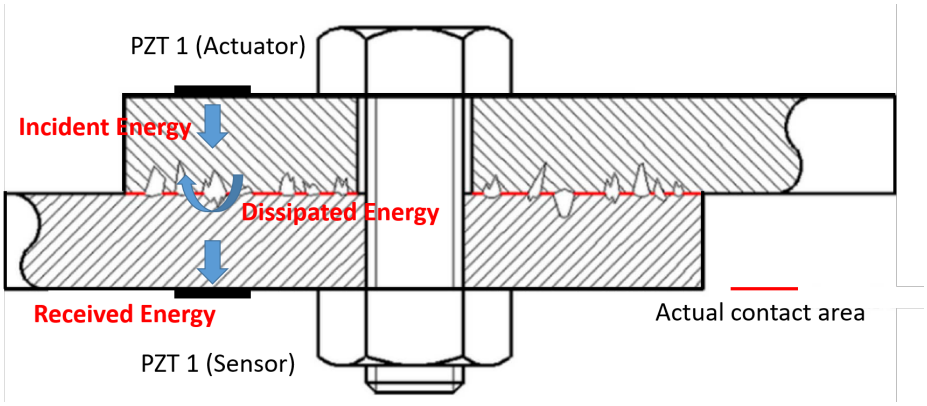

Figure 4.9: The micro contact state of bolted joint surface.

According to the theory in [127], the total response wave energy reflects the bolt looseness of the flange. To calculate the energy, first, we denote the received signal at piezoceramic transducers as X. Then, X can be decomposed into X_j (which has 2^n frequency bands) via a n-level wavelet packet as

$$X_j = [X_{j1}, X_{j2}, ..., X_{jm}], \tag{4.8}$$

where $j = 1, 2, 3, ..., 2^n$, and m is the number of sampling data of the decomposed signal.

The total energy E_j of the decomposed signal X_j is calculated as

$$E_j = \|X_j\|^2 = X_{j1}^2 + X_{j2}^2 + X_{j3}^2 + ... + X_{jm}^2. \tag{4.9}$$

Finally, the total response wave energy can be calculated by summarizing all decomposed signal energy E_j as,

$$E = \sum_{j=1}^{2^n} E_j. \tag{4.10}$$

The bolt looseness is reveal by comparing the summation of the total energy of all cases.

4.5 Experimental Results & Discussion

In this section, we present and discuss the experiment. The experiment is shown in Fig. 4.10, where the mobile platform manually delivers the robotic arm system to the front of the flange so that the robotic arm base is on the radial surface of the flange and the flange is within the operating range. Then, the robotic arm system detects, locates, and approaches the flange automatically. Once fingers touch the surfaces of the flange, the processed signal reflects the looseness of the flange. Finally, the inspection is completed, and the robotic arm leaves the flange.

Since the wrist is straight, if the robotic arm base is not on the radial surface, the piezoelectric transducers on the finger do touch the flange surface properly, and the signal from the sensor is not correct. The robotic arm's start position is where its four cameras can observe the flange. There are experiments for different control gain: $K_p = \frac{1}{3}$, $\frac{1}{4}$ and $\frac{1}{5}$. We did the same experiments for each control gain for 4 times and plotted the mean standard deviation error bars for trajectories of the robotic arm fingers and the average observed positions of the flange. In Fig. 4.11, the position is on r axis in cylinder coordinates. The x-axis of the figure is the number of control iterations. The dashed line is the position of the finger center, which is from the robotic arm dynamics monitors in section 4.2. The solid line is the average value of the current and all previous distance from the base to the

flange (the module of $\mathbf{v}_{bo}$). The shaded error bar for each line is the standard deviation. The red, blue and black lines represents the results with $K_p = \frac{1}{3}$, $\frac{1}{4}$ and $\frac{1}{5}$, respectively. In Fig. 4.11, dashed lines are longer than solid lines. This is because in the last few iterations, cameras no longer provide the flange location, and the last average location value is used to provide the control signal. Fig. 4.12 plots the distance between the finger center and the flange with different control gains, which is also regarded as control errors in Fig. 4.8. The place where the change of control errors become small or even negative is the iteration that cameras stop providing the position information.

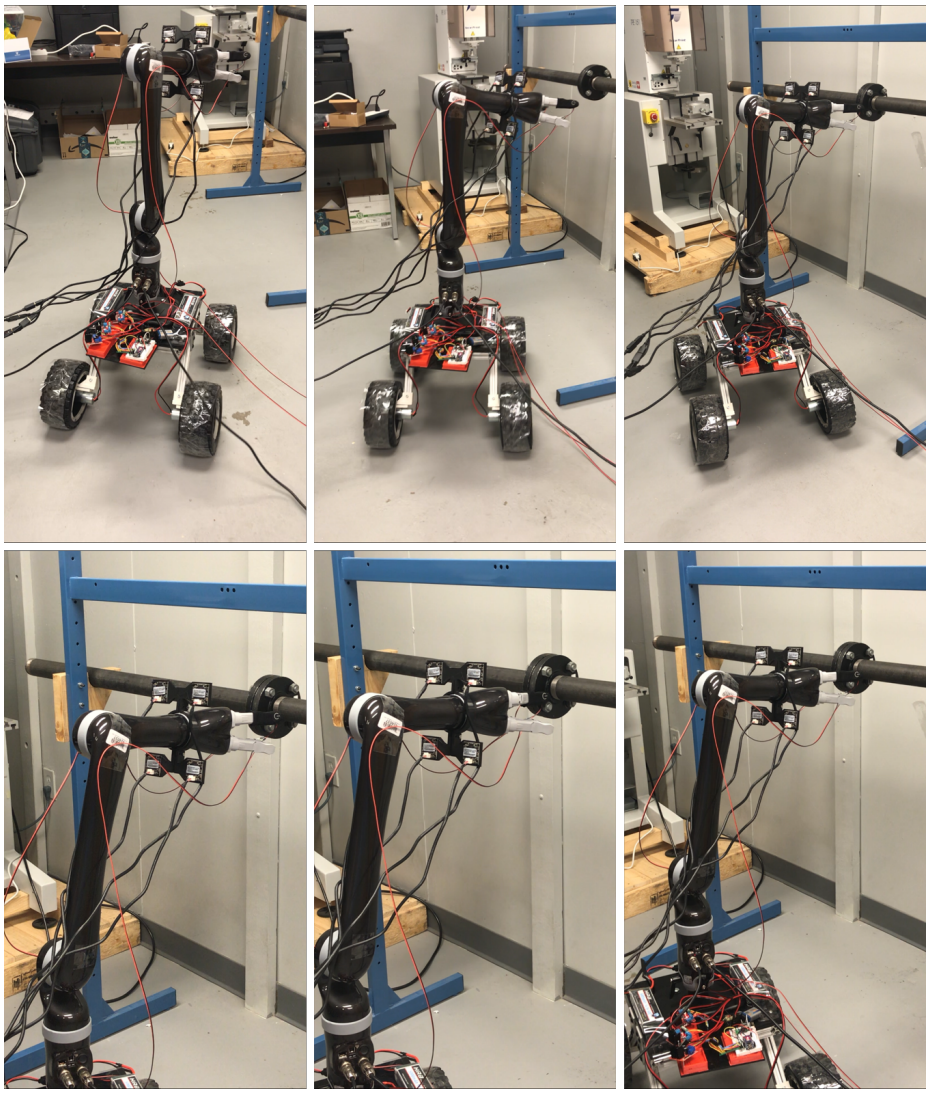

Figure 4.10: The grasping experiment of robotic arm system on a mobile platform.

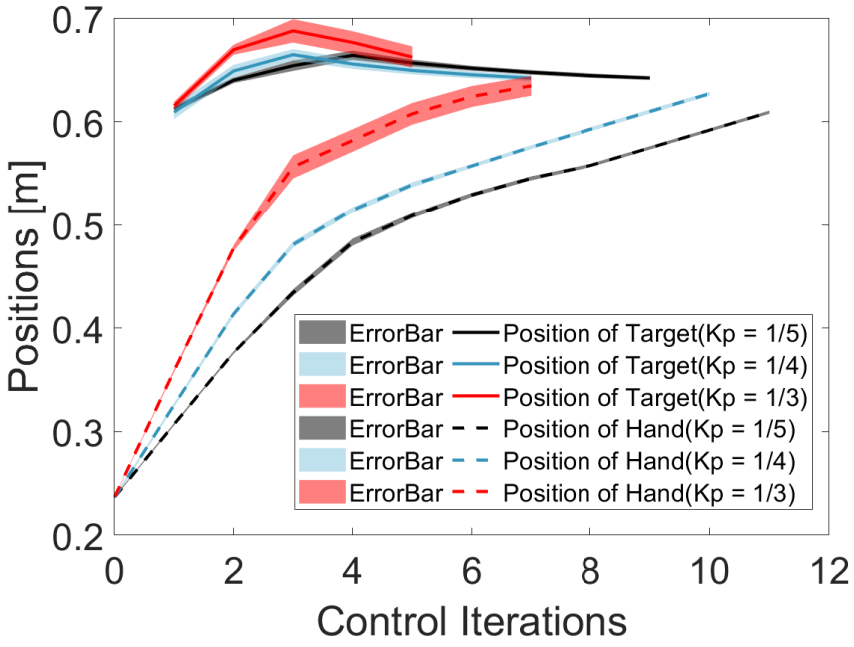

Figure 4.11: Trajectories of the robotic arm finger center and observed flange position on r axis in the cylinder coordinate.

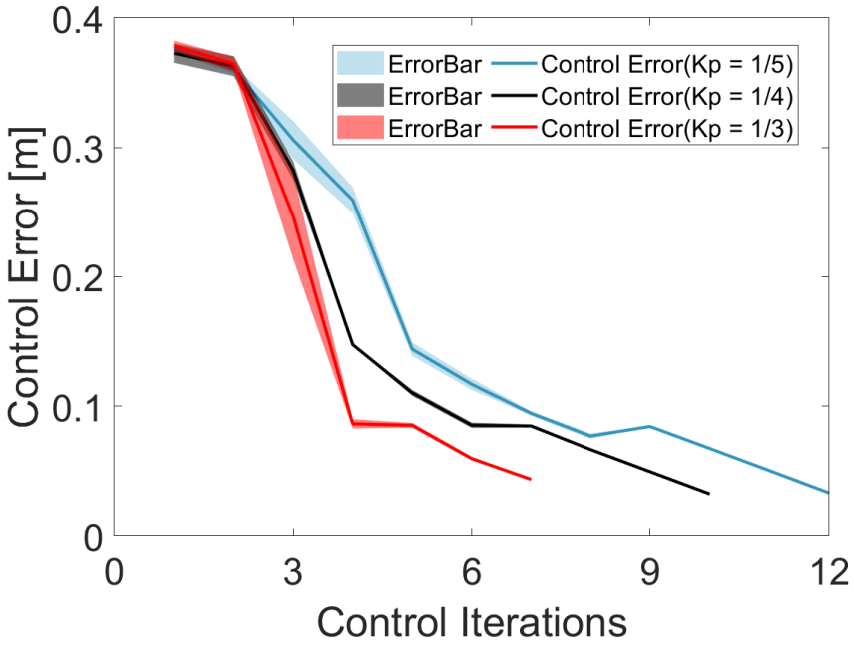

Figure 4.12: The control error of the automation system.

The gripper of the robotic arm wears a pair of the designed smart gloves(gloves with

piezoelectric transducers). One piezoelectric transducer sends a stress wave through the flange along the axial direction, which is received by another piezoelectric sensor. Since the stress wave is affected by the media, the flange bolts mounted with specific force lead to a signal with a unique shape in the sensor, which indicates whether the flange needs extra torque. As shown in Fig. 4.13, the grasping area on the flange is within the red dash-line oval. In experiment we denote the four M12 bolts on the flange as B_1, B_2, B_3, and B_4. The details about the arrangement of bolt looseness are given in table 4.2

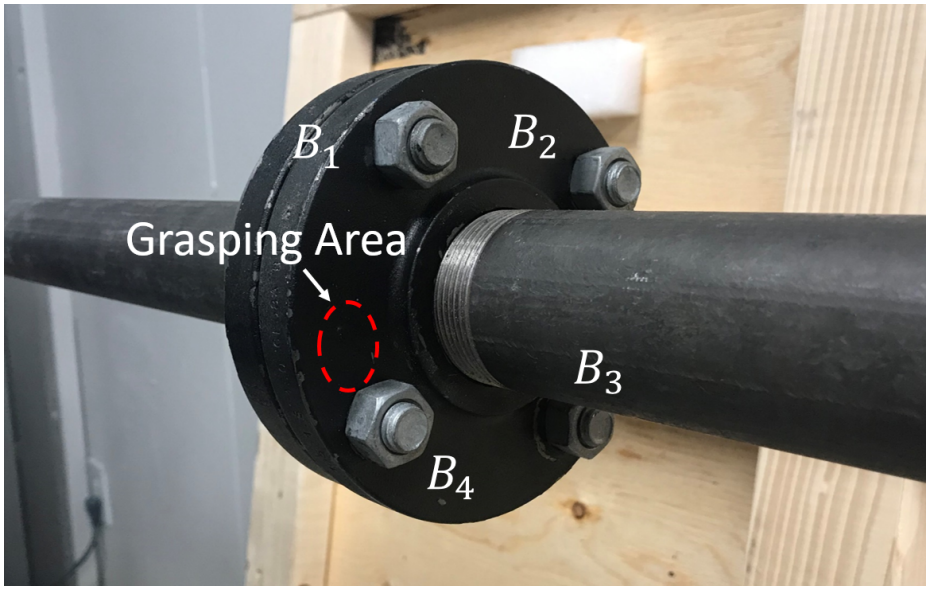

Figure 4.13: Received signal under different cases and corresponding energy.

Table 4.2: Arrangement of bolt looseness

Cases	B_1	B_2	B_3	B_4
1	Loose	Loose	Loose	Loose
2	Tighten	Loose	Loose	Loose
3	Tighten	Tighten	Loose	Loose
4	Tighten	Tighten	Tighten	Loose
5	Tighten	Tighten	Tighten	Tighten

The input and output of the active sensing method are stress wave (ultrasonic) signals,

which are generated and received by using an NI DAQ system (NI USB-6363). The duration of input is 0.1 seconds with a frequency range of 80-200 kHz, and the received signals are illustrated in Fig. 4.14. By calculating the signal energy (integral of square amplitude) as the indicator, one can achieve bolt looseness detection. According to Fig. 4.14, the signal energy decreases when more bolts are loosening, which verifies the effectiveness of the proposed method.

Besides the bolt looseness check, the signal from the piezoelectric sensor tells whether the robotic arm grabs the flange, which follows the control strategy of grasping the flange in Sec. 4.2 when cameras stop providing feedback information. In Fig. 4.14, case 1 represents the case that all four bolts are loose. If the signal energy is less than case 1, it means at least one finger does not fully touch the testing area.

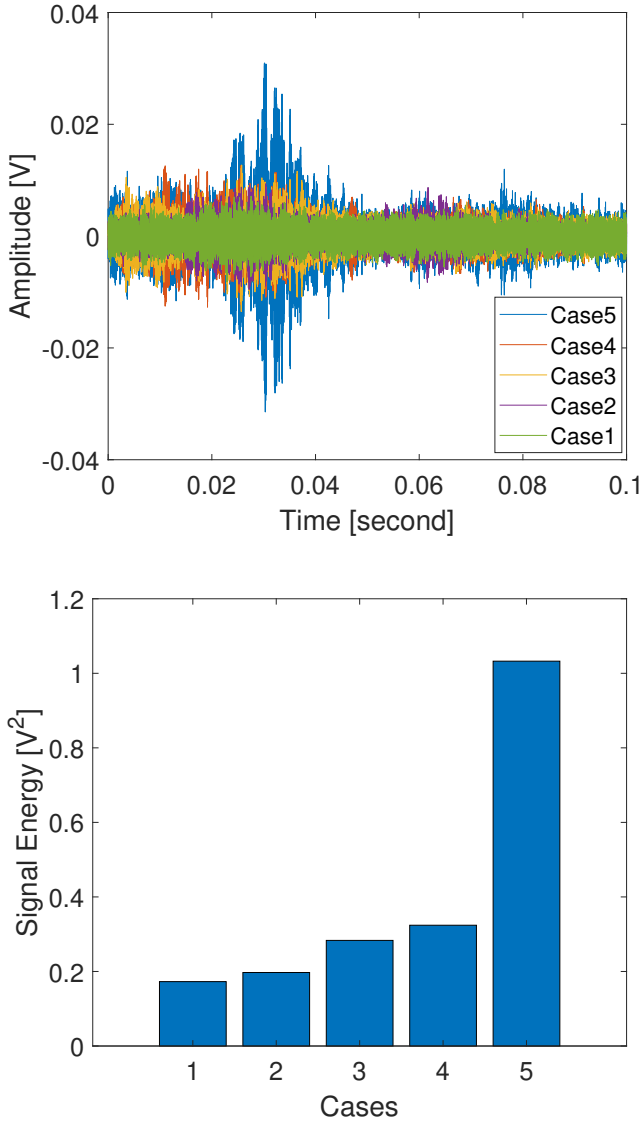

Figure 4.14: Received signal under different cases and corresponding energy.

4.6 Chapter Summary

In this chapter, we introduce a method to detect flange bolt looseness by operating a robotic arm. We control a four DOFrobotic arm to deliver piezo-sensors to a flange. The piezo sensors are mounted on the fingers of the robotic arm. During the automatic control process, a stereo camera system with four cameras, can detect and localize the flange in the image coordinates. With the calibrated stereo camera parameters, the 2D image coordinates

are transferred into 3D camera coordinates and then transferred into the robotic arm base coordinates with known robotic arm kinematics. The distance between the flange and the finger center is the control error. The next position of the robotic arm is computed by the proportion gain controller. When the camera is too close to the flange, the robotic arm creeps towards the flange by small steps. For each step, the fingers close to try to grab the flange until the piezo-sensor receives a signal of the correct grasping. In the experiment, we use a mobile platform to deliver the robotic arm, system and apply mathematical techniques to integrate the system with different coordinates and reduce experiment errors.

Chapter 5

Visual Servoing Control of Underwater Vehicle for Pipeline Tracking

This chapter introduced a novel pipeline tracking control of an ROV with an on-board camera. A geometry projection algorithm transfers this 2D path following problem into a 1D trajectory tracking problem. A special computer vision method derives the current heading angle and target angle, which are sent to a stable model-based controller. The controller performance is simulated in an OpenGL environment. The experiments compare the designed model-based controller and a PD controller. It validates this tracking method even the ROV was affected by the jet disturbance [130].

5.1 Angle Projection

The vehicle has a detect vector r_c along its moving direction as the red arrow in Fig. 5.1, where the 2D earth fixed position of the vehicle is (x_1, y_1), the current heading angle is ψ, the targeted heading direction is ψ_n (black dashed arrow), and the control error ψ_e [131],

where

$$\psi_e = \psi_n - \psi. \tag{5.1}$$

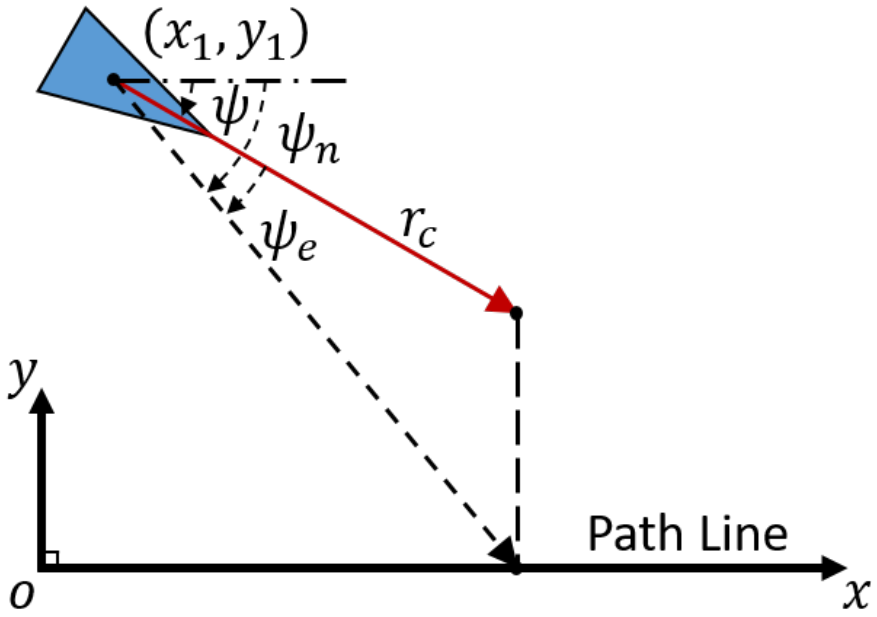

Figure 5.1: The path tracking algorithm description.

In a 3D environment, a monocular camera provide angle information for the $x' - y'$ plane (Body fixed coordinates), where x' axis is the vehicle's moving direction (along r_c in Fig. 5.1). Under the assumption that the distortion is neglected, the relationship between the heading and the pipeline is revealed in Fig. 5.2 where the camera detects the edge of the pipeline. Fig. 5.2(A) is the view of ROV on-board camera, (B) is the view from the right-hand-side of the camera, and Fig. 5.2(C) is observed from above. The notations with subscripts 'm' and 'w' represent the points in the image and world coordinates. In Fig. 5.2(A), the blue and red dash-dot line is the vanishing line of the camera view and the direct line of the vehicle, respectively.

For a well-calibrated vehicle camera, the vanishing line is a horizontal line passing through the vanishing point 'O_m', and the direct line is in the middle of the image. They are perpendicular to each other. The camera has a tilting angle 'α' from the horizontal direction about the y' axis. If $\alpha = 0$, the vanishing line will be in the middle of the image,

where the middle line (black dash-dot line) locates. The intersection of the direction line and the middle line is the origin point of the image, which is also the focal point of the camera. Later in the simulation, α is given directly in the program; The tilt angle of the camera is controlled by a gimbal in the experiment.

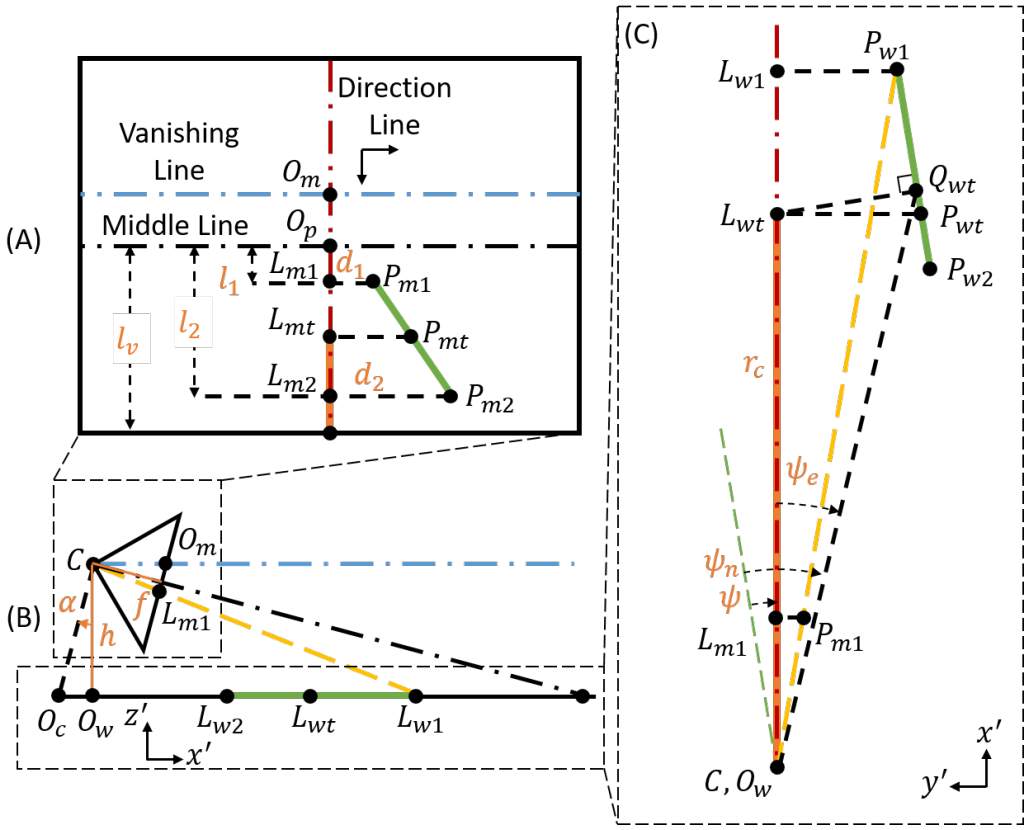

Figure 5.2: Description of angle projection relationship from real-world to image coordinates. (A) is the camera image view, (B) is the view from the right-hand-side of the camera, and (C) is observed from above.

In Fig. 5.2(A), P_{m1} and P_{m2} have their projections on the direction line as $L_{m.}$. In Fig. 5.2(B), the triangle represents the camera, where the vertical line represents the sensor plane; the black bottom line is the horizontal earth plane. CO_c is perpendicular to the optical axis of the camera, and O_c is on the ground. The projection of $P_{w.}$ on the direction line in the real-world coordinates is $L_{w.}$. The yellow dashed line is the light passing through P_{m1} and P_{w1}. In Fig. 5.2(C), the red dashed line is the direction line; the thick red line is r_c, the green dashed line is parallel to the target line (green line). The angles ψ, ψ_n, and

ψ_e defined in Fig. 5.1 are also shown in Fig. 5.2(C).

For easy implementation in following derivations and programs, in the '$column - row$' coordinates of Fig. 5.2(A), O_p, O_w are the origin points for the image and real-world, respectively. Define $P_{m1}(d_1, l_1)$, $P_{m2}(d_2, l_2)$, the length of CO_w as h, CO_w as f, where h is the vertical distance of the vehicle from the earth plane, f is the focal length of camera in pixels since image distance is approximately equal to focal length in the case of long object distance [132]. The coordinates of points on the pipeline in the real-world are $P_{w1}(P_{w1x'}, P_{w1y'})$, $P_{w2}(P_{w2x'}, P_{w2y'})$. Since two points have equal states in equations, an example of P_{w1} illustrates the relationship in Fig. 5.2

$$O_c L_{w1} /\!/ CO_m \,.$$

The equation related to the coordinates of P_{w1} is

$$\frac{\frac{f}{\cos\alpha}}{f\tan\alpha + l_1} = \frac{P_{w1x'} + h\tan\alpha}{\frac{h}{\cos\alpha}} \tag{5.2}$$

and

$$\frac{d_1}{-P_{w1y'}} = \frac{\frac{f}{\cos\alpha}}{P_{w1x'} + h\tan\alpha}. \tag{5.3}$$

Thus, the coordinates of the points are

$$P_{w1x'} = \frac{hf}{\cos^2\alpha(l_1 + f\tan\alpha)} - h\tan\alpha, \tag{5.4}$$

$$P_{w1y'} = -\frac{d_1 h}{\cos\alpha(l_1 + f\tan\alpha)}, \tag{5.5}$$

$$P_{w2x'} = \frac{hf}{\cos^2\alpha(l_2 + f\tan\alpha)} - h\tan\alpha, \tag{5.6}$$

and

$$P_{w2y'} = -\frac{d_2 h}{\cos\alpha(l_2 + f\tan\alpha)}. \tag{5.7}$$

Since $P_{w1}P_{w2}$ is the reference path line, the heading angle ψ in Fig. 5.1 is

$$\psi = -\arctan\frac{P_{w1y'} - P_{w2y'}}{P_{w1x'} - P_{w2x'}}\,, \tag{5.8}$$

where $\psi \in (-\frac{\pi}{2}, \frac{\pi}{2}]$.

In Fig. 5.2(A), L_{mt} is on the direction line (red dash-dot line) and in the middle between the middle line (black dash-dot line) and the bottom edge. The point P_{mt} is on the line $P_{w1}P_{w2}$, and $P_{mt}L_{mt} /\!\!/ P_{m1}L_{m1}$. The projection of L_{mt} and P_{mt} in (C) are L_{wt} and P_{mt}, respectively. The vector O_wL_{wt} is the detect vector r_c in (C) (thicker red solid line). Q_{wt} is on $P_{w1}P_{w2}$, $L_{wt}Q_{wt}$ is normal to $P_{w1}P_{w2}$. The target angle ψ_n is $\angle L_{wt}O_wQ_{wt}$. The positions of $P_{wt}(P_{wtx'}, P_{wty'})$ and $Q_{wt}(Q_{wtx'}, Q_{wty'})$ are computed as

$$P_{wtx'} = \frac{2hf}{(l_v + 2f\tan\alpha)\cos^2\alpha} - h\tan\alpha\,, \tag{5.9}$$

$$P_{wty'} = \frac{-P_{wtx'}(\frac{l_v}{2}(d_1 - d_2) + l_1 d_2 - l_2 d_1)}{\frac{f}{\cos\alpha}(l_1 - l_2)}\,, \tag{5.10}$$

$$Q_{wtx'} = \sin\psi\cos\psi P_{wty'} + P_{wtx'}\,, \tag{5.11}$$

and

$$Q_{wty'} = \cos^2\psi P_{wty'}\,, \tag{5.12}$$

where l_v is the distance from the middle to the bottom edge of Fig. 5.2(A), and all variables

are yellow letters in Fig. 5.2. The target direction angle for the vehicle is

$$\begin{aligned}\psi_n &= \psi + \psi_e\,, \\ &= \psi + \arctan\frac{Q_{wty'}}{Q_{wtx'}}\,, \\ &= \psi + \arctan\frac{\cos^2\psi\frac{l_v(d_1-d_2)+2l_1d_2-2l_2d_1}{(l_1-l_2)}}{\sin 2\psi\frac{l_v(d_1-d_2)+2l_1d_2-2l_2d_1}{2(l_1-l_2)} - \frac{2f}{\cos\alpha}}\,.\end{aligned} \tag{5.13}$$

In both Eqn. 5.13, 'h' is canceled in numerator and denominator, thus the height of the ROV is not necessary for the calculation of the target direction angle.

5.2 Line pickup

The Hough algorithm detected possible lines through the image segmentation method with binary mask [133]. The result contains the lines of the pipeline's edge and noise. An example is in Fig. 5.3 (A), where the orange lines are detection results from Hough algorithm. In world coordinates, all detected lines from the same pipeline are parallel to each other, and their differences in ρ from the Hough transformation are small, while the noise lines have no relations between each other. The Fig. 5.3 (B) shows the distribution of lines in $\psi - \rho$ plane, where the clusters with most points represent the target lines for the tracking control. Fig. 5.3 (C) is the 2D histogram of the distribution in (B), where brighter color represents more points in the block. The two blocks with the most histogram value are chosen regions. The average of lines within that region is the target line to be tracked by the ROV.

In Fig. 5.3, there are two target line candidates from the histogram. The rule to decide the current tracking line has to satisfy three requirements: First, the current line must be closer to the camera; second, if the current line is too short in the image, the target line switches; Finally, after the line switch, the target does not switch back to the previous line for any instance. In order to pick up the target line in the view, for the ith iteration, each

detected line segment has a weight $w^{[i]}$, which is given in Eqn. 5.14. The candidate with the largest weighted histogram value is the target line to be tracked. For one iteration, since the time interval is short, the angle change for the ROV is small. The angle difference is small between the detected lines of two neighbor iteration unless the length of the current line is too short in the image. Fig. 5.3 (D) presents the final result pipeline detection

$$w^{[i]} = \sqrt{(d_1^{[i]} - d_2^{[i]})^2 + (l_1^{[i]} - l_2^{[i]})^2} \frac{\pi}{|\psi^{[i]} - \psi^{[i-1]}| + \frac{\pi}{c_s}} \,. \tag{5.14}$$

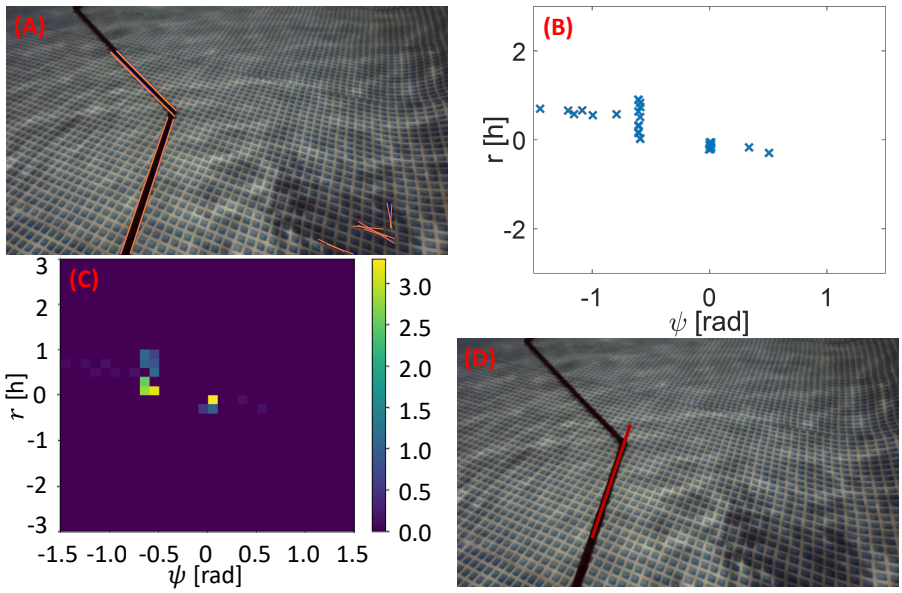

Figure 5.3: (A) is the detection result where orange lines are computed from Hough algorithm; (B) is the distribution of lines in world coordination, where x axis is the angle, and y axis is the distance from origin point to the lines; (C) is the histogram of (B); (D) shows the target line.

5.3 ROV model

A general underwater vehicle dynamic model in body-fixed coordinates [94] is

$$\mathbf{M}\dot{\boldsymbol{\nu}} + \mathbf{C}(\boldsymbol{\nu})\boldsymbol{\nu} + \mathbf{D}(\boldsymbol{\nu})\boldsymbol{\nu} + \mathbf{g}(\boldsymbol{\nu}) = \boldsymbol{\tau}, \tag{5.15}$$

where

$$\begin{aligned}
\mathbf{M} &= \text{inertia matrix (including added mass)} \\
\mathbf{C}(\boldsymbol{\nu}) &= \text{matrix of Coriolis and centripetal terms} \\
&\quad \text{(including added mass)} \\
\mathbf{D}(\boldsymbol{\nu}) &= \text{damping matrix} \\
\mathbf{g}(\boldsymbol{\nu}) &= \text{gravity forces and moments} \\
\tau &= \text{control inputs.}
\end{aligned} \tag{5.16}$$

In this research, the following assumptions are made before the modeling work:

- The ROV is at slow speed (at most 1m/s) during the pipeline inspection, thus lift forces is negligible;
- The ROV is assumed to have port-starboard symmetry and fore-aft symmetry, and the center of gravity (CG) is located in the symmetry planes, thus the motion in roll and pitch are neglected;
- The environment water is ideal fluid;
- The force from tether attached to the ROV is neglected;

The pipeline tracking mission is reduced to a 2D path following the problem under the assumptions above. In Eqn. 5.15, the CG is the origin point of the coordinates, and matrices are given

$$\mathbf{M} = \begin{bmatrix} m - X_{\dot{u}} & 0 & 0 \\ 0 & m - Y_{\dot{v}} & 0 \\ 0 & 0 & I_z - N_{\dot{r}} \end{bmatrix}, \tag{5.17}$$

$$\mathbf{C}(\boldsymbol{\nu}) = \begin{bmatrix} 0 & 0 & Y_{\dot{v}}v - mv \\ 0 & 0 & mu - X_{\dot{u}}u \\ mv - Y_{\dot{v}}v & X_{\dot{u}}u - mu & 0 \end{bmatrix}, \tag{5.18}$$

and,

$$\mathbf{D}(\boldsymbol{\nu}) = -\begin{bmatrix} X_u + X_{u|u|}|u| & 0 & 0 \\ 0 & Y_v + Y_{v|v|}|v| & 0 \\ 0 & 0 & N_r + N_{r|r|}|r| \end{bmatrix}, \tag{5.19}$$

where the values of all added mass and drag coefficient are negative, and,

$$\boldsymbol{\nu} = [u, v, r]^T. \tag{5.20}$$

$\boldsymbol{\nu}$ is the vector of velocities in body-fixed coordinates. For the states in earth fixed coordinates

$$\dot{\boldsymbol{\eta}} = \mathbf{J}_{\boldsymbol{\psi}}(\boldsymbol{\eta})\boldsymbol{\nu}, \tag{5.21}$$

where, $\boldsymbol{\eta}$ is the vector of position in earth coordinates,

$$\boldsymbol{\eta} = [x, y, \psi]. \tag{5.22}$$

In 3 degree of freedom (DOF) coordinates, the transformation $\mathbf{J}_{\boldsymbol{\psi}}$ is

$$\mathbf{J}_{\boldsymbol{\psi}} \overset{3DOF}{=} \boldsymbol{R}(\psi) = \begin{bmatrix} \cos\psi & -\sin\psi & 0 \\ \sin\psi & \cos\psi & 0 \\ 0 & 0 & 1 \end{bmatrix}. \tag{5.23}$$

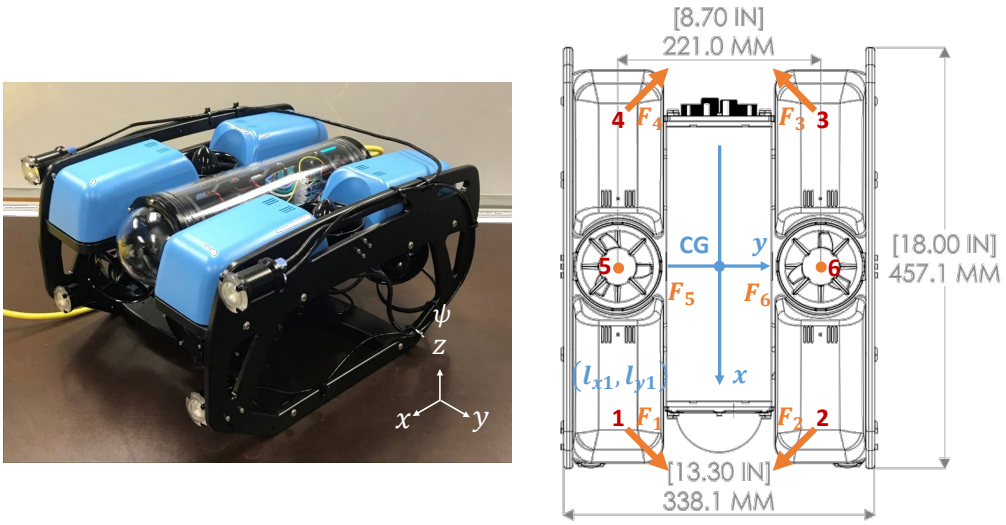

Figure 5.4: The BlueROV and Schematics of it (top view).

In Eqn. 5.15, the control inputs are the total forces and the torque in difference direction, which is

$$\boldsymbol{\tau} = [F_x, F_y, T_\psi]^T. \tag{5.24}$$

As shown in Fig. 5.4, the forces and the torque from thrusters $1-6$, which generate thrusts $F_1 - F_6$. Thrusters 5, 6 provide the vertical thrust, while $1-4$ control the motion in $x-y$ plane. The parameters are defined in the section view of the ROV in Fig. 5.4. The buoyancy control along z direction is important for the ROV control. However, in this research, we assumed that the vertical motion has been well controlled, and the depth is maintained.

From Fig. 5.4, the force and the torque from each thruster are

$$\boldsymbol{\tau}_i = \begin{bmatrix} F_{xi} \\ F_{yi} \\ T_{\psi i} \end{bmatrix} = \begin{bmatrix} F_i \cos\gamma_i \\ F_i \sin\gamma_i \\ F_{yi} l_{xi} - F_{xi} l_{yi} \end{bmatrix}, \tag{5.25}$$

where F_i, F_{xi}, F_{yi} and $T_{\psi i}$ are the thrust, its components in x, y direction and torque for thruster i $(i = 1, 2, 3, 4)$. γ_i is the angle of the thrust force, and (l_{xi}, l_{yi}) is the position of the thruster in the body fixed coordinates. An example of (l_{x1}, l_{y1}) is shown in Fig. 5.4.

The total force on the ROV is

$$\boldsymbol{\tau} = \sum_{i=1}^{4} \boldsymbol{\tau}_i. \tag{5.26}$$

Since a monocular on-board camera is not able to calculate the displacement information with no reference, the strategy is to set F_x to its maximum F_{xmax}, F_y to zero, and control the heading angle so that $\psi \to \psi_n$ in Fig. 5.1. Besides the equations in Eqn. 5.25 and 5.26, there is an additional restriction for this equation system, that is

$$F_i \in [-F_{max}, F_{max}], \tag{5.27}$$

where F_{max} is the maximum output force for each thruster. With the assumption of the symmetric distribution of thrusters $1-4$, according to Eqn. 5.25, 5.26, and 5.27, the relationship between F_x and T_ψ is

$$\frac{F_{xmax}}{\cos\gamma_1} + \frac{T_\psi}{l_{x1}\cos\gamma_1 - l_{y1}\sin\gamma_1} \in [-4F_{max}, 4F_{max}]. \tag{5.28}$$

Since F_{xmax} is positive, the restriction of the torque is

$$|T_\psi| \leq T_{max} = (4F_{max} - \frac{F_{xmax}}{\cos\gamma_1})(l_{x1}\cos\gamma_1 - l_{y1}\sin\gamma_1). \tag{5.29}$$

Eqn. 5.29 separates the forward thrust force and the yaw torque to the different regions. Thus, during the path following control in the next section, there are

$$F_x = F_{xmax},$$

and

$$|T| \leq T_{max}.$$

Although this allocation does not utilize the maximum power of thrusters, the control of torque does not affect the forward thrust.

5.4 Path Following

This section includes controller design, stability analysis, and angle restriction. With the designed model-based controller, there are two steps for the system to reach the steady state: 1) Prove the convergence of the angle, where $\psi \to \psi_n$; 2) Validate that the whole system is stable with the condition from 1).

For the first step, the controller is defined to guarantee stability in angle. According to Fig. 5.1 in the earth fixed coordinates, the reference angle has the equation that

$$\tan \psi_n = -\frac{y}{r_c \cos \psi}. \tag{5.30}$$

According to Eqn. 5.15, under the assumption that the added mass has $Y_{\dot{v}} = X_{\dot{u}}$, the control in ψ domain is:

$$(I_z - N_{\dot{r}})\dot{r} = T_\psi + N_r r + N_{r|r|}|r|r. \tag{5.31}$$

The control input signal is T_ψ, which is designed to achieve a stable system. To analyze the stability, the Lyapunov function V_1 is

$$V_1 = \frac{c(\psi - \psi_n)^2 + (I_z - N_{\dot{r}})(r - \dot{\psi}_n)^2}{2}, \tag{5.32}$$

where, c is a positive gain value. The derivative of the Lyapunov function is

$$\begin{aligned} \dot{V}_1 &= c(\psi - \psi_n)(r - \dot{\psi}_n) + (I_z - N_{\dot{r}})(r - \dot{\psi}_n)(\dot{r} - \ddot{\psi}_n) \\ &= c(\psi - \psi_n)(r - \dot{\psi}_n) \\ &\quad +(r - \dot{\psi}_n)(T_\psi + N_r r + N_{r|r|} r|r| - \ddot{\psi}_n(I_z - N_{\dot{r}})). \end{aligned} \tag{5.33}$$

The designed model-based controller is

$$T_\psi = -d(r - \dot{\psi}_n) - c(\psi - \psi_n) - N_r r - N_{r|r|}|r|r + \ddot{\psi}_n(I_z - N_{\dot{r}}), \tag{5.34}$$

where, d is a positive gain value. If we substitute Eqn. 5.34 into Eqn. 5.33, the derivative of the Lyapunov function is

$$\dot{V}_1 = -d(r - \dot{\psi}_n)^2. \tag{5.35}$$

Thus, the system is stable according to the Lyapunov Lasalle theorem [134] with the controller in Eqn. 5.34.

After the first step, where the condition that the steady state of the ROV system in ψ domain has been achieved, where $\psi \to \psi_n$, the ROV reaches the steady state in $x-y$ plane and converges to the target path. The second Lyapunov function is

$$V_2 = \frac{a\boldsymbol{\nu}_m^T \mathbf{M}\boldsymbol{\nu}_m + by^2 + c(\psi - \psi_n)^2 + (I_z - N_{\dot{r}})(r - \dot{\psi}_n)^2}{2}, \tag{5.36}$$

where $\boldsymbol{\nu}_m = \boldsymbol{\nu} - [u_{max}, 0, \psi_n]$, u_{max} is the maximum value of u. a and b are positive gain values. According to Eqn. 5.18, $\boldsymbol{\nu}_m^T \mathbf{C}\boldsymbol{\nu}_m \equiv 0$. The derivative of the second Lyapunov function is

$$\begin{aligned} \dot{V}_2 &= a((F_{xmax} + X_u u + X_{u|u|}|u|u)(u - u_{max}) + Y_v v^2 \\ &\quad + Y_{v|v|}|v|v^2) + b(y(u\sin\psi + v\cos\psi)) + \dot{V}_1. \end{aligned} \tag{5.37}$$

Since $\psi \to \psi_n$, based on Eqn. 5.30, there is

$$\begin{aligned} & b(y(u\sin\psi + v\cos\psi)) = by(-\frac{y}{r_c}u + \frac{\sqrt{r_c^2 - y^2}}{r_c}v) \\ = \; & \frac{-b}{ur_c}(yu - \frac{\sqrt{r_c^2 - y^2}v}{2})^2 + \frac{b(r_c^2 - y^2)v^2}{4ur_c}. \end{aligned} \tag{5.38}$$

Eqn. 5.37 is then transformed to

$$\begin{aligned} \dot{V}_2 &= a(F_{xmax} + X_u u + X_{u|u|}|u|u)(u - u_{max}) \\ &\quad + a(Y_v v^2 + Y_{v|v|}|v|v^2) - \frac{b}{ur_c}(yu - \frac{\sqrt{r_c^2 - y^2}v}{2})^2 \\ &\quad + \frac{b(r_c^2 - y^2)v^2}{4ur_c} + \dot{V}_1, \end{aligned} \tag{5.39}$$

For each part of Eqn. 5.39, there are

$$\dot{V}_1 < 0,$$

and

$$Y_{v|v|}|v|v^2 < 0,$$

since the thrust is only on the positive direction along 'u',

$$(F_{xmax} + X_u u + X_{u|u|}|u|u)(u - u_{max}) < 0,$$

and

$$-\frac{b}{ur_c}(yu - \frac{\sqrt{r_c^2 - y^2}v}{2})^2 < 0,$$

and

$$v^2(aY_v + \frac{b(r_c^2 - y^2)}{4ur_c}) < 0,$$

by setting the gain $a > \frac{-b(r_c^2-y^2)}{4ur_cY_v}$. Thus,

$$\dot{V}_2 < 0.$$

According to the Lyapunov Lasalle theorem, the ROV system is stable with designed model-based controller while following the specific path.

In application, if the target pipeline is outside the camera view, there is no feedback information, and the control is terminated, even if the vehicle is on a stable trajectory. Moreover, if $\psi \notin (-\frac{\pi}{2}, \frac{\pi}{2})$ due to an overshoot situation, the target direction is then opposite. Thus, the target angle ψ_n is assigned according to the calculation value of ψ_n and ψ from

Eqn. 5.8 and Eqn. 5.13 with the restriction that

$$\psi_n \coloneqq \begin{cases} -\psi_s & \psi \in [-\pi, -\frac{\pi}{2}] \\ -\psi_s & \psi_n \in (-\pi, -\psi_s] \psi \in (-\frac{\pi}{2}, \frac{\pi}{2}) \\ \psi_n & \psi_n \in (-\psi_s, \psi_s), \psi \in (-\frac{\pi}{2}, \frac{\pi}{2}) \\ \psi_s & \psi_n \in [\psi_s, \pi], \psi \in (-\frac{\pi}{2}, \frac{\pi}{2}) \\ \psi_s & \psi \in [\frac{\pi}{2}, \pi)\,, \end{cases} \tag{5.40}$$

where ψ_s is the safety angle. If there is no safety angle, the overshoot in ψ domain results in a sign change of the tracking direction. The tracking control is terminated due to the system divergence or the loss of reference.

In Eqn. 5.40. If the safety angle ψ_s is too small, the converge rate is slow when the vehicle is far away from the target line. However, it takes the risk of overshooting if ψ_s is close to $\frac{\pi}{2}$. Rather than setting a value to compromise between continuity and convergence, a better strategy is to set two-stage safety angles ψ_{s1} and ψ_{s2}. The switch occurs when the distance between the vehicle and the target line r_d is larger than r_c. According to Fig. 5.2 and equations in Sec. 5.1, r_d and r_c are

$$r_d = |\sqrt{Q_{wty'}^2 + Q_{wtx'}^2} \sin \psi_n| \tag{5.41}$$

and

$$r_c = P_{wty'}. \tag{5.42}$$

The switching condition for the safety angle ψ_s is

$$\psi_s \coloneqq \begin{cases} \psi_{s1} & r_d \leq r_c \\ \psi_{s2} & r_d > r_c\,, \end{cases} \tag{5.43}$$

where ψ_{s1} is the first stage safety angle and $\psi_{s1} < \psi_{s2}$.

5.5 Simulation and Experiment

This section demonstrates the simulation and the experiment result. In the simulation part, the results show how different parameters affect tracking performance. The picked solution from the simulation is then applied to the experiment. The control diagram is in Fig. 5.5. In each control iteration, the line detection block derives the current angle ψ and the target angle ψ_n from the camera snapshot. The control error ψ_e, is sent to the controller. After that, the controller output the PWM value which steers the vehicle and starts a new control iteration. In the experiment, the details inside the line detection block are in the lower figure of Fig. 5.5.

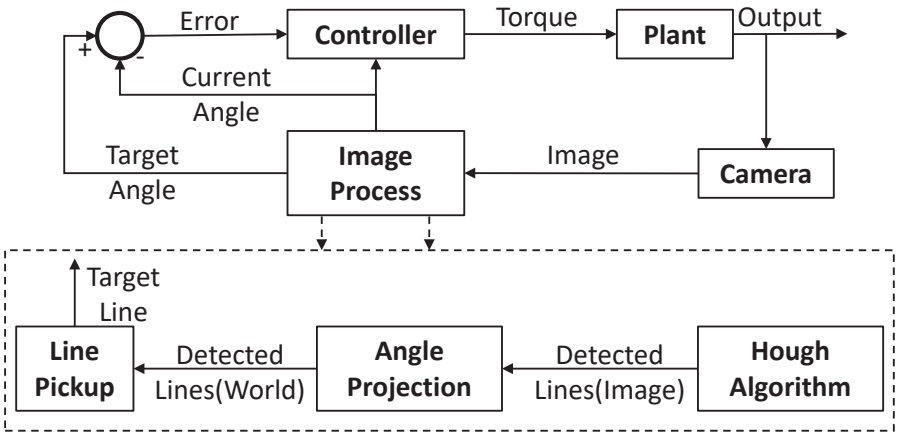

Figure 5.5: The control diagram, where below is the detail of image process block.

5.5.1 Simulation

The simulation is in the first-person perspective, and the 3D environment is built up with OpenGL and Pygame. Fig. 5.6 shows four on-board camera views of the OpenGL 3D environment with edges detected by the Hough algorithm and the line pickup criteria, where the red dashed line, blue dashed line, solid orange line, and solid green line are direction line, vanishing line, detect vector, and target line, respectively. The position of the vehicle

is listed in Fig. 5.6 for screenshots (1), (2), (3), and (4).

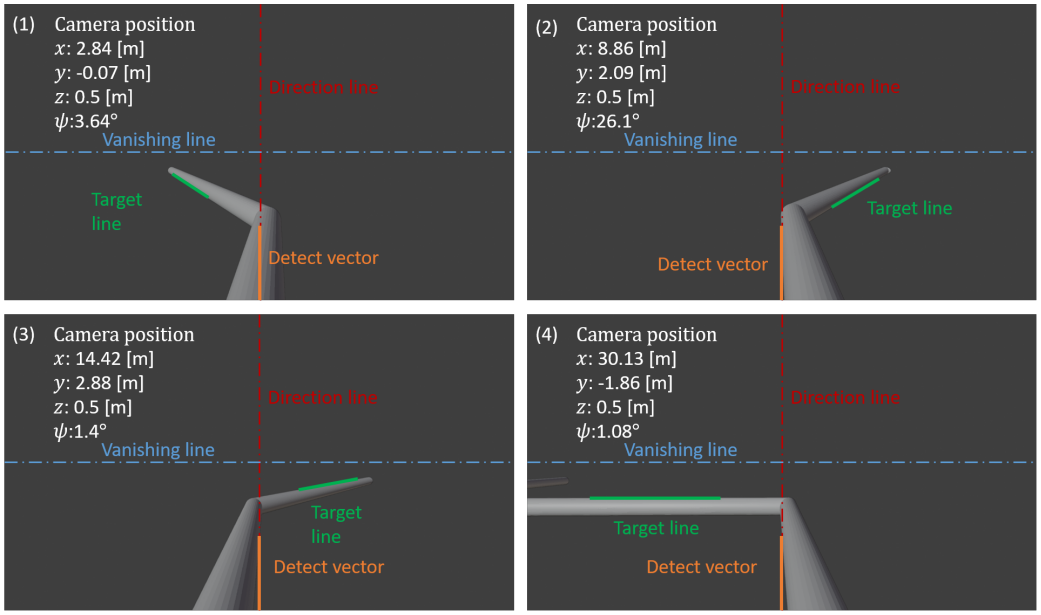

Figure 5.6: The 3D simulation environment.

All parameters for the ROV(BlueROV2) in simulations are all verified by experiments in [135, 136]

Table 5.1: Simulation parameters.

m	I_z	F_{max}
11.17 kg	0.24 kgm^2	$40N$
$X_{\dot{u}}$	$Y_{\dot{v}}$	$N_{\dot{r}}$
-5.5 kg	-5.5 kg	-0.12 kgm^2/rad
X_u	Y_v	N_r
-4.03 Ns/m	-6.22 Ns/m	-0.07 Ns/rad
$X_{u\|u\|}$	$Y_{v\|v\|}$	$N_{r\|r\|}$
-18.18 Ns^2/m^2	-21.66 Ns^2/m^2	-1.55 Ns^2/rad^2

5.5.2 Simulation Result

In this research, the control variable method is used to test each parameter's function to the ROV tracking performance. According to the control variable method, only one variable changes for each simulation. Fig. 5.7 includes cases with different initial positions and heading angles with a designed path. Nine initial conditions (ICs) are in table 5.2 (ICs in legends of Fig. 5.7). For the same path, the tracking results with different safety angle, detect vector length, switch coefficient, simulation control time delay (time interval for each iteration) are shown in Fig. 5.7, where the detect vector's amplify ratio is based on its actual length in world coordinates (the default vector is $O_w L_{wt}$ in Fig. 5.2 (C)).

Table 5.2: Initial conditions.

Name	x[m]	y[m]	z[m]	ψ [rad]
IC0	0.0	1.0	0.5	-0.45
IC1	0.0	1.0	0.5	-0.7
IC2	0.0	2.0	0.5	-0.45
IC3	0.0	2.0	0.5	-0.7
IC4	0.0	-1.0	0.5	0.35
IC5	0.0	-1.0	0.5	0.6
IC6	0.0	-2.0	0.5	0.35
IC7	0.0	-2.0	0.5	0.6
IC8	0.0	0.0	0.5	0.0

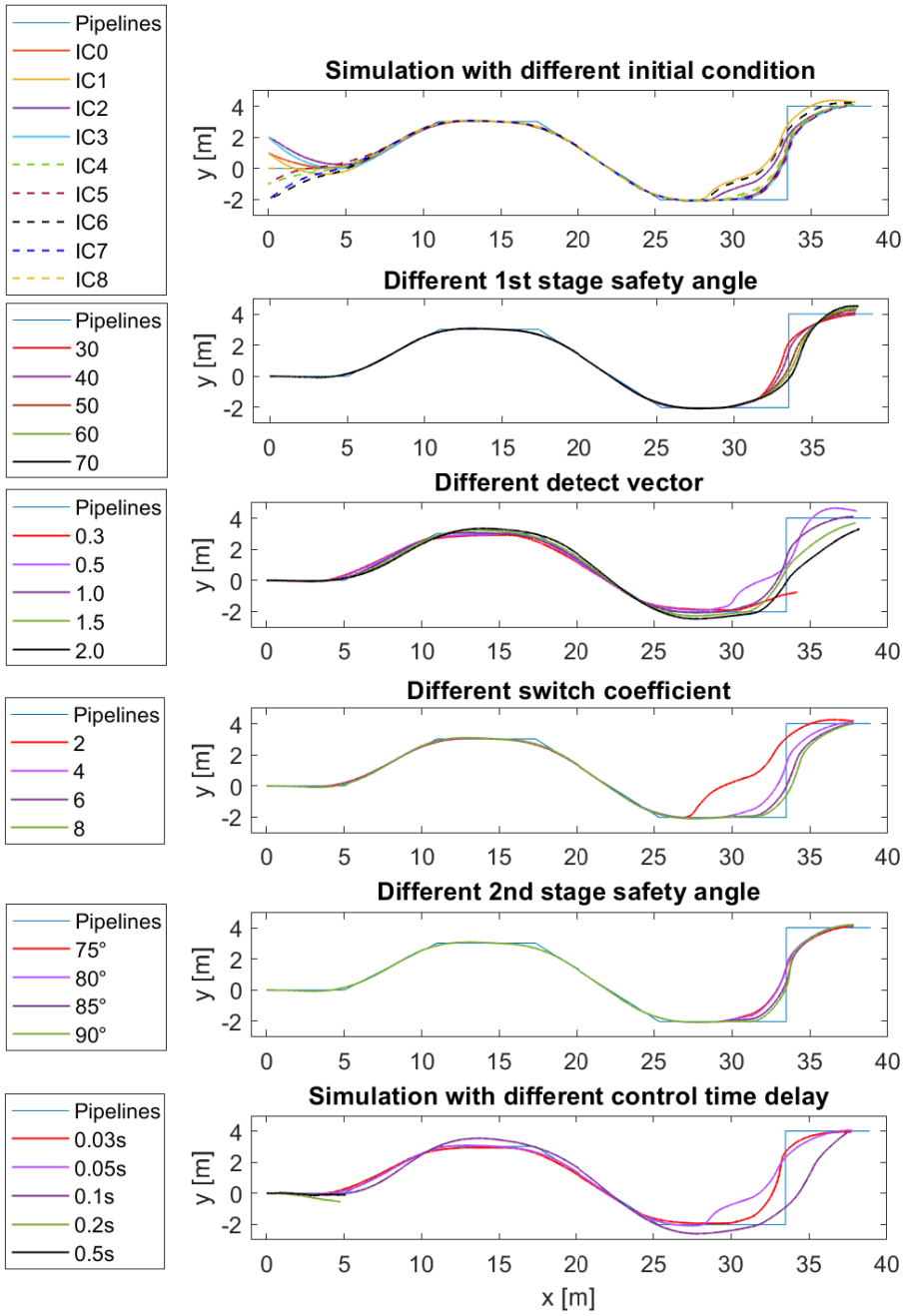

Figure 5.7: Tracking results with different initial condition and controller parameters and control time delay.

The first sub-figure of Fig. 5.7 shows the tracking result with nine different initial positions and heading angles. The difference among cases is reduced significantly in the second pipeline segment. As long as other parameters have proper values and the pipeline is in the camera view initially, the initial conditions do not affect the tracking performance too much.

When it is close to the right angle, the critical factor is the selection of safety angles based on Eqn. 5.40. In Fig. 5.7, the first stage safety angle does not change the tracking result for a small angle change. When the tracking angle changes $\frac{\pi}{2}$, and the distance between the vehicle and the line is short, the first safety angle only changes the target angle. When the only changing parameter is the detect vector amplify ratio, the tracking is terminated at the right angle area when it is 0.3 times the default length ($O_w L_{wt}$) because the pipeline is outside of the view. Though the ratio increases to 0.5, the fluctuation is high at the right angle area. In the simulation for different switch coefficients, c_s does not affect the performance when it is no less than 4. However, when it is 2 or even smaller, $\frac{\pi}{c_s}$ is large, and the influence of the angle difference between two neighbor sample times is low. According to the target line switch criteria in Eqn. 5.14, a new line is picked when the current target line is short enough in the image view. If the length of the line has the most impact ratio to decide the target line, the Hough algorithm will introduce uncertainty to the tracking performance. The second stage safety angle only takes effect when the vehicle is outside of the first stage area. The plots in Fig. 5.7 have differences only when the ROV is near the right angle lines. In Fig. 5.7, when the time delay increases from 0.05 s to 0.1 s, the deviation of the trajectory is not negligible. When the delay is 0.2 s and 0.5 s, the tracking terminates before the first turn.

Based on the comparison and the discussion of the simulation results, the test parameters and their optimal values are summarized in the Tab. 5.3,

Table 5.3: Tested values and the picked values for all parameters.

Parameters	Picked value	Test values
Initial position	$(0, 0)$	***IC*** 0-8
Initial heading direction	0	***IC*** 0-8
1st safety angle ψ_{s1} [degree]	40	30, 50, 60, 70
Detect vector amplify ratio	1	0.3, 0.5, 1.5, 2
Switch coefficient c_s	4	2, 6, 8
2nd safety angle ψ_{s2} [degree]	75	80, 85, 90
Control time delay [s]	0.05	0.03, 0.1, 0.2, 0.5

5.5.3 Experiment Setups

In the experiment, the testing vehicle is BlueROV2 from Blue Robotics Inc. The camera in the front of the ROV has $80 orizontal field of view and +/- 90$ ilt angle range. In Fig. 5.8, the pool's size is 10 m $\times$ 5 m $\times$ 1.2 m, and the water level is around 1 m. The black PVC pipelines are at the bottom of the pool, where the bending angles are $45 nd 90$ The jet locates at the top right place of the pool.

The control of the ROV is based on the ArduSub project, which is a fully-featured and open source solution for ROVs. The communication between the Ardusub and the vehicle is through a protocol, which is called MAVlink. In this research, the implementation of the protocol is in python. The thrusters and the ROV motion are controlled by pulse width modulation (PWM) value. The MAVlink pre-defined PWM output range is $[1100, 1900]$, where 1500 is the station value, 1900 is the positive maximum, and 1100 is the negative maximum. The PWM control function was integrated into the keyboard for manual control, a mode switch to automatic pipeline tracking, and emergency stop. The time interval for each control loop is 0.1 s, which is enough for image processing, controller calculation, and vehicle communication.

In the automatic control process, first, with the jet off, we compare the performance

between the model-based nonlinear controller in Eqn 5.34 and a PD controller. The control signal is

$$T_{PD} = -K_D(r - \dot{\psi}_n) - K_P(\psi - \psi_n).$$

The value of gains in this experiment are $K_D = d = 7$, $K_P = c = 1$. The PWM is mapped from the control torque as $\frac{180T_{\cdot}}{\pi}$, where $T_{\cdot}$ is T_{PD} or T_{ψ}. Then, we turn on the jet and test the tracking algorithm under the disturbance from the jet flow.

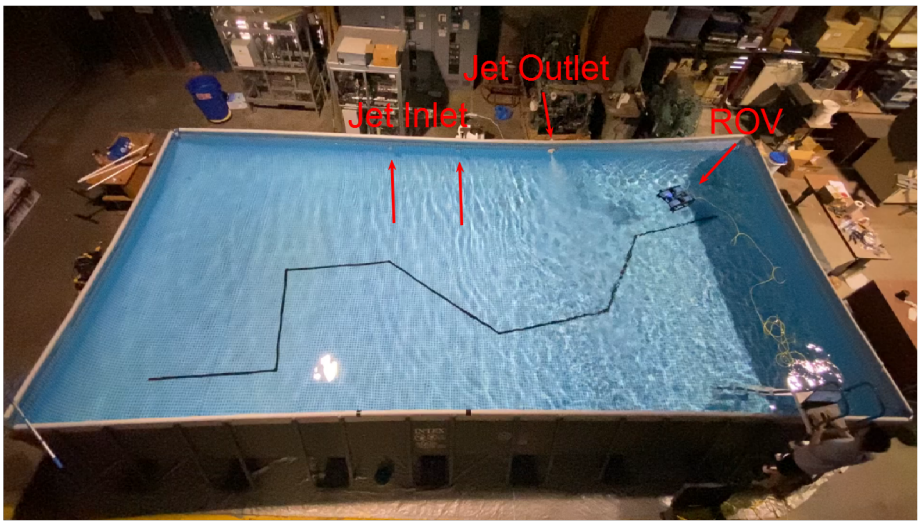

Figure 5.8: The above view of the ROV, the pool, and the pipeline.

5.5.4 Experimental Result

At the beginning of the experiment, the ROV moved to the initial position of the pipeline, which is at right-upper corner of the pool. After the ROV switched to the "automatic mode", it started to track the pipeline automatically.

The four figures in Fig. 5.9 are the screenshot of the ROV on-board camera, which are also the original image for the Hough line detection algorithm, where the time, ψ, ρ are at the left upper corner, h from Fig. 5.2 is unknown. The green line is the detected result for each image, which is also regarded as the target line. According to Sec. 5.1, the target line is the x-axis of the world coordinates. The right-upper corner is the current detected ψ and

ρ from the Hough transformation domain in the world coordinate.

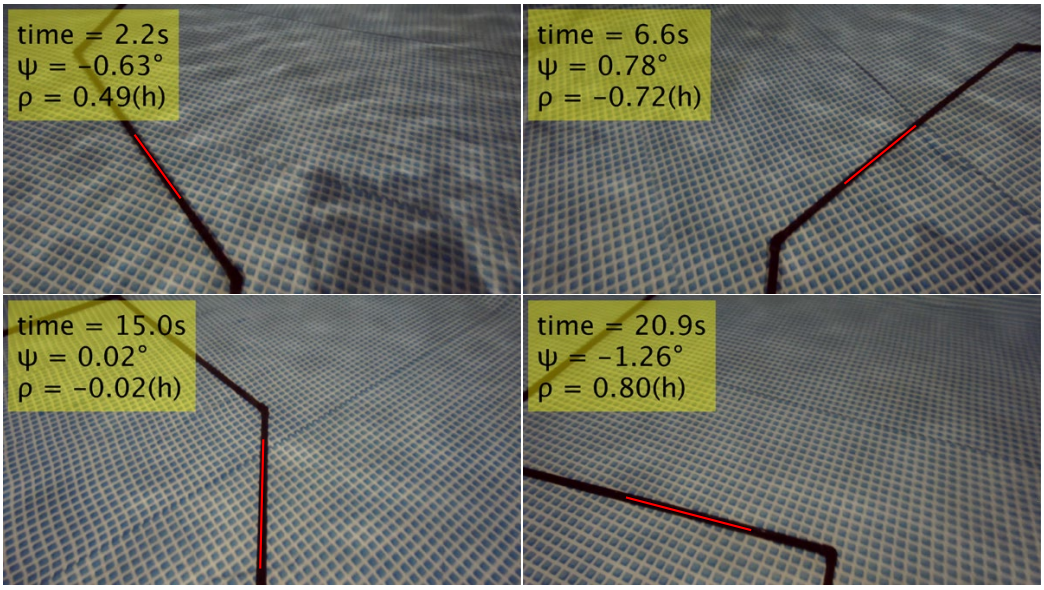

Figure 5.9: The snapshots from the on-board camera, where the red line is the target.

The states of ROV are shown in Fig. 5.10. In the above figure, the angle relative to the current target line ψ, the target angle ψ_n, control error ψ_e, and the ρ from the Hough algorithm in world coordinate are blue, red, yellow, and black lines, respectively. The control input (PWM value) is the blue line below. The black line has a boolean value, where '1' means the tracking target switches to the next pipeline segment. The solid line is the controller from Eqn. 5.34, and the dashed line is the result of PD controller, respectively.

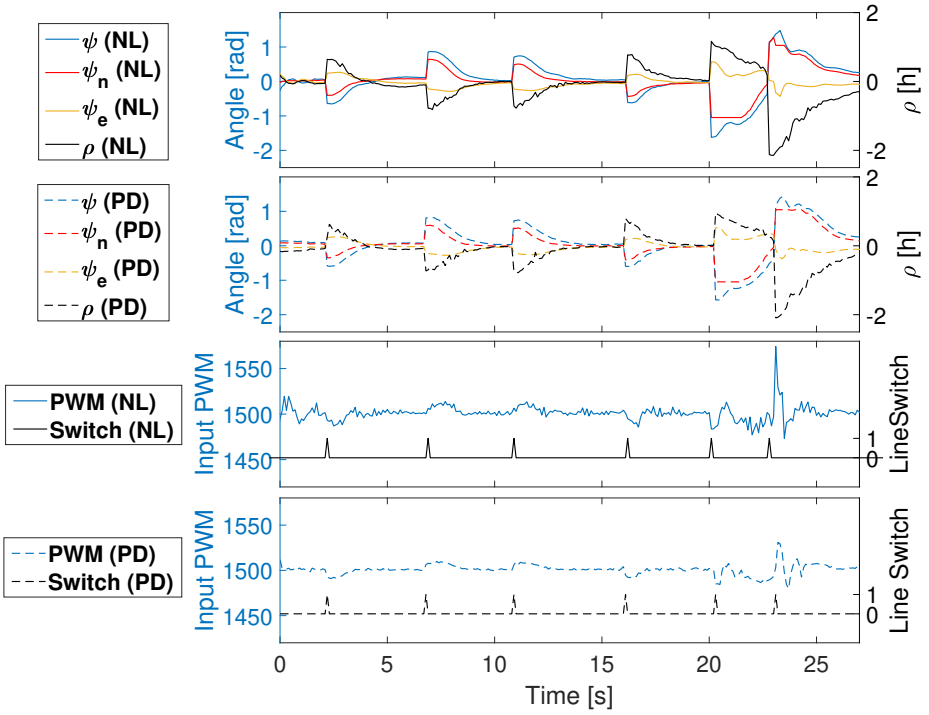

Figure 5.10: The states of ROV in the experiment with the model-based nonlinear (NL) and PD controller.

Fig. 5.11 shows the states with the model-based nonlinear under an unknown disturbance from a jet. The location of the jet is shown as Fig. 5.8.

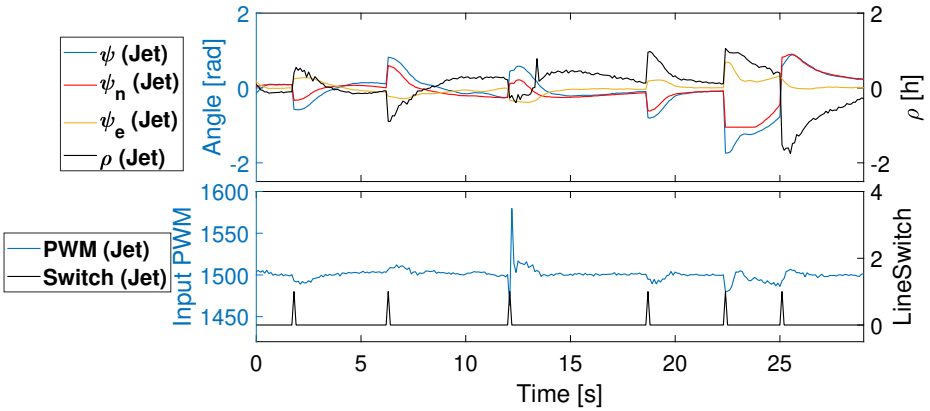

Figure 5.11: The states of ROV in the experiment with model-based nonlinear under an unknown disturbance from a jet.

5.5.5 Discussion

The experiment result is one from many repeated tests. Since the initial positions for those tests are not the same based on manual control, it is meaningless to do average on the tracking states even with the same parameters. However, the trend for each test is similar to Fig. 5.10. For each pipeline segment, the ROV converges to the line and remains steady states. When the target line switches, ψ, ψ_n, ψ_e, and ρ increase sharply and gradually return to the steady states. The input PWM value also changes with the new line segment. Its amplitude is related to the angle difference between the previous and current segments. The angle change are $\pm 45°$ and $\pm 90°$ from Fig. 5.8. For each line switch, the sudden change of the ψ is regarded as an impulse disturbance in ROV's heading angle while the ROV is tracking a straight line. Thus, as long as the target line is still in the camera view, the ROV is able to recover from the impulse change in the angle domain.

Since PD control is a model-free controller, the tracking result of PD controller with well-tuned gains is shown as dashed lines in Fig. 5.10. Based on the control gain from PD control, we plot the trajectory of the nonlinear controller in Eqn. 5.34 and compare its performance with the PD controller. For all states, the nonlinear controller has more fluctuation. One of the contributions comes from derivative in Eqn. 5.34. The angle errors from the line detection step are also amplified in r (the first order derivative of ψ) and $\ddot{\psi}$.

Compared with the previous experiment(jet off), the jet affects the ROV at the time between 10 s and 15 s in the experiment with the disturbance. At the third line switch place, the change of the PWM value is large at first and then gradually decreases. This is because the target angle is changed from 0o-45 and the jet starts to push the ROV at the same time. The controller generates a large control signal to maneuver the ROV against the disturbance from the jet and the angle change.

The biggest challenge of the experiments compared to the simulation is the environmental noise and errors inside the control loop. In this experiment, the camera loses a few frames due to the signal transmission overwhelming. Under this situation, the feedback information is the prediction based on the previously detected line, and Kalman filter [111].

During the tracking process, the noise comes from the shadow of the ROV itself and water waves on the surface. The brightness of the camera view changes as the relative position between the ROV and the lights varies. Since Hough transformation is based on the color, the brightness change introduces extra error to the line detection result. The histogram has ruled out these errors. In the image process step, the errors come from pixels allocation and image blur due to the sudden shake of the ROV. The control input is not smooth for both controllers because of this error. However, the ROV has an input dead zone, where the propeller does not move when the absolute PWM value is less than 10. Thus the fluctuation of the PWM value in Fig. 5.10 is acceptable, and the ROV accomplishes the tracking mission.

5.6 Chapter Summary

In this chapter, an image-based visual servoing control is developed for an ROV to track the pipeline with a single-lens on-boards camera, even when there is no length reference outside the camera. We develop an angle-based path planning method to transfer the 2D path following problem into a 1D trajectory tracking problem. Hough transformation criteria detect straight lines in the camera snapshot. We derive the reference and current angle states from the angle projection rules and designed a model based controller to track the reference angle with stability analysis. To verify this algorithm, we test the method in simulation and experiment. We also compare the model-based controller with a well-tuned PD controller under the disturbance jet. In the future, we will include the pipeline surface inspection with image processing method.

Chapter 6

Summary and Future Work

6.1 Summary

This research aims to provide algorithms for visual servoing control under different circumstances.

In Chapter 2, we develop a method to track the object on the water surface and provide the position even when the object is blocked by the obstacle. We first generate the projection matrix to transfer the data from the image coordinate to the real world coordinate. To differentiate the pixels of the object and the noise, we classify the pixel groups into four categories. With a different group, we apply a different method. We use the minimum volume ellipse to have the vector to represent the object. When the part of the object is blocked, the vector is used to predict the location of the object. If the object is blocked completely, we use the velocity of the last frame and the Kalman filter to predict the position of the object. The vector prediction and the constant velocity location prediction provide the position information if there is no extra velocity change in the blind area. This tracking algorithm provides position and velocity information for the intelligence feedback control even when the feedback information is partially/completely missing. However, there is an accumulation of error in the prediction of position and velocity in the blind area. The problem is complex when the obstacle is not stationary.

In Chapter 3, the back relaxation characteristic of the IPMC in an IPMC/Servo driven

robotic fish is considered in the development of a model-based collision avoidance control for robotic fish. First, we verify the IPMC "back-relaxation" phenomenon by experiments. Next, we assemble one fish to investigate the turning behavior of the fish when the IPMC is subjected to different constant voltages. Based on these experiments, a transfer function model relating the IPMC voltage input to the normal acceleration of the fish is identified. After that, a collision avoidance algorithm that considers the relative velocity kinematics of the fish to obstacles and the dynamics of the robotic fish system are developed. 'To validate the control, two robotic fish were fabricated and assembled. Experiments have demonstrated successful collision avoidance performance.

In Chapter 4, we develop a robotic system to monitor the tightness of bolts on each flange of pipelines. We control a 4 DOF robotic arm to deliver a pair of piezo-sensor and actuator to two sides of a flange. The piezo-sensor and actuator are mounted on the fingers of the robotic hand. When the robotic hand grabs the flange, the piezo-actuator generates a stress wave signal. The stress-wave signal measured from the piezo-sensor side can reflect whether the bolt is tightly bolted or not. During the control process, a stereo camera system, which consists of four cameras, is applied to localize the flange in images through the YOLO network and its feature. The localized 2D image coordinates are transferred into 3D real-world positions. When the camera is too close to the flange to provide any information, the robotic hand moves with small steps towards the flange. During this process, the robot tries to grab the flange by opening and closing its fingers. When the piezo-sensor receives the signal from the flange, the flange is grabbed and the bolt looseness detection is started. In the experiment, we use a mobile platform to deliver the robotic arm system and connect the system with different coordinates.

In Chapter 5, an image-based visual servoing control is developed for an ROV to track the pipeline using a single-lens on-board camera. An angle-based path planning method is used to transfer this 2D path following problem into a 1D trajectory tracking problem in the angle domain. Thus, the algorithm can work even there is no length reference outside the vehicle. A Hough transformation criteria is used to detect straight lines from images. A line

pickup criteria is used to select the target line. An angle projection rule is used to derive the target angle from the target line in the image. A model-based controller is designed with stability analysis. The simulations have been conducted in the OpenGL environment, where parameters in the method have been tested and tuned to reach a smooth tracking performance. The experiments compare the model-based controller with a well-tuned PD controller and verified this tracking algorithm with disturbance jet flows

6.2 Future Direction

In Chapter 2, 3, and 5, the image processing steps are based on the binary mask with a well calibrated threshold. However, the threshold with a fixed value does not perform well when the brightness changes or the contrast between the target object and the background is not obvious. One possible solution is to apply a more complex image detection algorithm. For example, in Chapter 4, a YOLO neural network is used to detect the target flange with different view angles and backgrounds. Thus, the deep learning algorithm provides a robust result for path prediction [137] and line detection [138].

In real world underwater operation, 3D control is necessary. In addition to the 2D visual servoing control in this dissertation, depth control has to be applied. However, besides the buoyancy control, an extra dimension means more complex algorithms in all applications. For example, the tilt angle has to be maintained to fix the camera view direction. The safety zone in the collision avoidance control can be a sphere or cylinder instead of a circle [79].

In the future plan, we will optimize the visual tracking algorithm to have less error and faster computing speed to make sure that the computer vision tracking algorithm is good enough for the robotic fish and other underwater vehicles. We will upgrade the models, sensors, and control algorithms from 2D to 3D, where the target objects are tracking in a sensor sphere with lidars, radars, and sonars [139, 140, 141].

We will put the on-board sensors on the robotic fish and implement the decentralized collision avoidance control. Since we have already verified the robustness of the robotic

arm grasping control with the smart sensor, the next step is to equip the manipulators for underwater vehicles. We will also apply a deep learning algorithm to inspect the texture and cracks on the oil pipelines.

References

[1] John Hill. Real time control of a robot with a mobile camera. In *9th Int. Symp. on Industrial Robots, 1979*, pages 233–246, 1979.

[2] Elizabeth B Kujawinski, Melissa C Kido Soule, David L Valentine, Angela K Boysen, Krista Longnecker, and Molly C Redmond. Fate of dispersants associated with the deepwater horizon oil spill. *Environmental science & technology*, 45(4):1298–1306, 2011.

[3] Richard Camilli, Christopher M Reddy, Dana R Yoerger, Benjamin AS Van Mooy, Michael V Jakuba, James C Kinsey, Cameron P McIntyre, Sean P Sylva, and James V Maloney. Tracking hydrocarbon plume transport and biodegradation at deepwater horizon. *Science*, 330(6001):201–204, 2010.

[4] Harilaos N Psaraftis and Nikolaos P Ventikos. An intelligent robot system to respond to oil spills: the eu-mop project. *Proceedings of the 6th. Interspill*, 2006.

[5] Dennis Fritsch, Kai Wegener, and Rolf Dieter Schraft. Control of a robotic swarm for the elimination of marine oil pollutions. In *2007 IEEE Swarm Intelligence Symposium*, pages 29–36. IEEE, 2007.

[6] Charlotte Skourup, John Pretlove, Neil Stembridge, and Mona Svenes. Enhanced awareness for offshore teleoperation. In *Intelligent Energy Conference and Exhibition*. OnePetro, 2008.

[7] Amit Shukla and Hamad Karki. Application of robotics in offshore oil and gas industry—a review part ii. *Robotics and Autonomous Systems*, 75:508–524, 2016.

[8] Mark J Kaiser. The global offshore pipeline construction service market 2017–part i. *Ships and Offshore Structures*, 13(1):65–95, 2018.

[9] Curtis Lohr and Maria Pena. Stones development: a pioneering management philosophy for enhancing project performance and safety. In *Offshore Technology Conference*. OnePetro, 2017.

[10] Qiang Bai and Yong Bai. *Subsea pipeline design, analysis, and installation*. Gulf Professional Publishing, 2014.

[11] Ahmed Senouci, Mohamed Elabbasy, Emad Elwakil, Bassem Abdrabou, and Tarek Zayed. A model for predicting failure of oil pipelines. *Structure and Infrastructure Engineering*, 10(3):375–387, 2014.

[12] Michael Ho, Sami El-Borgi, Devendra Patil, and Gangbing Song. Inspection and monitoring systems subsea pipelines: A review paper. *Structural Health Monitoring*, 19(2):606–645, 2020.

[13] F Diaz Ledezma, Ayman Amer, Fadl Abdellatif, Ali Outa, Hassane Trigui, Sahejad Patel, and Roba Binyahib. A market survey of offshore underwater robotic inspection technologies for the oil and gas industry. In *SPE Saudi Arabia Section Annual Technical Symposium and Exhibition*. OnePetro, 2015.

[14] Josef Krautkrämer and Herbert Krautkrämer. *Ultrasonic testing of materials*. Springer Science & Business Media, 2013.

[15] Maersk Drilling Norway. Magnetic particle examination. 2011.

[16] Yan Shi, Chao Zhang, Rui Li, Maolin Cai, and Guanwei Jia. Theory and application of magnetic flux leakage pipeline detection. *Sensors*, 15(12):31036–31055, 2015.

[17] Ali Sophian, Guiyun Tian, and Mengbao Fan. Pulsed eddy current non-destructive testing and evaluation: A review. *Chinese Journal of Mechanical Engineering*, 30(3):500–514, 2017.

[18] Francisco CR Marques and Alessandro Demma. Ultrasonic guided waves evaluation of trials for pipeline inspection. In *17th World Conference on Nondestructive Testing*, pages 25–28. Citeseer, 2008.

[19] Victor Ashworth and C JL Booker. Cathodic protection: Theory and practice. 1986.

[20] Blue Robotics. Bluerov2. *Datasheet, June*, 2016.

[21] Bernard Espiau, François Chaumette, and Patrick Rives. A new approach to visual servoing in robotics. *ieee Transactions on Robotics and Automation*, 8(3):313–326, 1992.

[22] Danica Kragic and Henrik I Christensen. Survey on visual servoing for manipulation. *Computational Vision and Active Perception Laboratory, Fiskartorpsv*, 15:2002, 2002.

[23] Seth Hutchinson, Gregory D Hager, and Peter I Corke. A tutorial on visual servo control. *IEEE transactions on robotics and automation*, 12(5):651–670, 1996.

[24] AC Sanderson. Image based visual servo control using relational graph error signal. In *Proc. Int. Conf. Cybernetics and Society, Cambridge, 1980*, 1980.

[25] François Chaumette and Seth Hutchinson. Visual servo control. i. basic approaches. *IEEE Robotics & Automation Magazine*, 13(4):82–90, 2006.

[26] François Chaumette and Seth Hutchinson. Visual servo control. ii. advanced approaches [tutorial]. *IEEE Robotics & Automation Magazine*, 14(1):109–118, 2007.

[27] Moslem Kazemi, Kamal Gupta, and Mehran Mehrandezh. Path-planning for visual servoing: A review and issues. *Visual Servoing via Advanced Numerical Methods*, pages 189–207, 2010.

[28] Peter I Corke. Visual control of robot manipulators–a review. In *Visual Servoing: Real-Time Control of Robot Manipulators Based on Visual Sensory Feedback*, pages 1–31. World Scientific, 1993.

[29] Andrea Cherubini, François Chaumette, and Giuseppe Oriolo. Visual servoing for path reaching with nonholonomic robots. *Robotica*, 29(7):1037–1048, 2011.

[30] Jin-Woo Lee, Sung-Uk Choi, Young-Jin Lee, and K. S. Lee. A study on recognition of road lane and movement of vehicles using vision system. In *SICE 2001. Proceedings of the 40th SICE Annual Conference. International Session Papers (IEEE Cat. No.01TH8603)*, pages 38–41, 2001.

[31] Mukhamad Aji Putra, Endra Pitowarno, and Anhar Risnumawan. Visual servoing line following robot: Camera-based line detecting and interpreting. In *2017 International Electronics Symposium on Engineering Technology and Applications (IES-ETA)*, pages 123–128. IEEE, 2017.

[32] Fenner H Holman, Andrew B Riche, Adam Michalski, March Castle, Martin J Wooster, and Malcolm J Hawkesford. High throughput field phenotyping of wheat plant height and growth rate in field plot trials using uav based remote sensing. *Remote Sensing*, 8(12):1031, 2016.

[33] Mohammad Mahdi Azari, Fernando Rosas, Kwang-Cheng Chen, and Sofie Pollin. Optimal uav positioning for terrestrial-aerial communication in presence of fading. In *2016 IEEE Global Communications Conference (GLOBECOM)*, pages 1–7. IEEE, 2016.

[34] Noel-Marie Gray, Kimberly Kainec, Sandra Madar, Lucas Tomko, and Scott Wolfe. Sink or swim? bone density as a mechanism for buoyancy control in early cetaceans. *The Anatomical Record: Advances in Integrative Anatomy and Evolutionary Biology: Advances in Integrative Anatomy and Evolutionary Biology*, 290(6):638–653, 2007.

[35] Alicia Keow, Zheng Chen, and Hilary Bart-Smith. Pida control of buoyancy device enabled by water electrolysis. *IEEE/ASME Transactions on Mechatronics*, 25(3):1202–1210, 2020.

[36] Francois Chaumette, Patrick Rives, and Bernard Espiau. Positioning of a robot with respect to an object, tracking it and estimating its velocity by visual servoing. In *ICRA*, pages 2248–2253, 1991.

[37] Lawrence G Roberts. *Machine perception of three-dimensional solids*. PhD thesis, Massachusetts Institute of Technology, 1963.

[38] Martin A Fischler and Robert C Bolles. Random sample consensus: a paradigm for model fitting with applications to image analysis and automated cartography. *Communications of the ACM*, 24(6):381–395, 1981.

[39] David G Lowe. Robust model-based motion tracking through the integration of search and estimation. *International Journal of Computer Vision*, 8(2):113–122, 1992.

[40] Daniel F DeMenthon and Larry S Davis. Model-based object pose in 25 lines of code. *International journal of computer vision*, 15(1-2):123–141, 1995.

[41] Sundaram Ganapathy. Decomposition of transformation matrices for robot vision. *Pattern Recognition Letters*, 2(6):401–412, 1984.

[42] Patrick Wunsch and Gerd Hirzinger. Real-time visual tracking of 3d objects with dynamic handling of occlusion. In *Proceedings of International Conference on Robotics and Automation*, volume 4, pages 2868–2873. IEEE, 1997.

[43] Tom Drummond and Roberto Cipolla. Real-time tracking of multiple articulated structures in multiple views. In *European Conference on Computer Vision*, pages 20–36. Springer, 2000.

[44] Andreas Ruf, Martin Tonko, Radu Horaud, and H-H Nagel. Visual tracking of an end-effector by adaptive kinematic prediction. In *Proceedings of the 1997 IEEE/RSJ*

International Conference on Intelligent Robot and Systems. Innovative Robotics for Real-World Applications. IROS'97, volume 2, pages 893–899. IEEE, 1997.

[45] H Christopher Longuet-Higgins. A computer algorithm for reconstructing a scene from two projections. *Nature*, 293(5828):133–135, 1981.

[46] Olivier Faugeras and OLIVIER AUTOR FAUGERAS. *Three-dimensional computer vision: a geometric viewpoint*. MIT press, 1993.

[47] Zhengyou Zhang, Rachid Deriche, Olivier Faugeras, and Quang-Tuan Luong. A robust technique for matching two uncalibrated images through the recovery of the unknown epipolar geometry. *Artificial intelligence*, 78(1-2):87–119, 1995.

[48] Scott A Brandt, Christopher E Smith, and Nikolaos P Papanikolopoulos. The minnesota robotic visual tracker: a flexible testbed for vision-guided robotic research. In *Proceedings of IEEE International Conference on Systems, Man and Cybernetics*, volume 2, pages 1363–1368. IEEE, 1994.

[49] Alfred A Rizzi and Daniel E Koditschek. An active visual estimator for dexterous manipulation. *IEEE Transactions on Robotics and Automation*, 12(5):697–713, 1996.

[50] Paul VC Hough. Method and means for recognizing complex patterns, December 18 1962. US Patent 3,069,654.

[51] Dana H Ballard. Generalizing the hough transform to detect arbitrary shapes. *Pattern recognition*, 13(2):111–122, 1981.

[52] Demetri Terzopoulos. On matching deformable models to images. In *Topical Meeting on Machine Vision Tech. Digest Series*, volume 12, pages 160–167, 1987.

[53] Andrew Blake and Michael Isard. *Active contours: the application of techniques from graphics, vision, control theory and statistics to visual tracking of shapes in motion*. Springer Science & Business Media, 2012.

[54] Michael Kass, Andrew Witkin, and Demetri Terzopoulos. Snakes: Active contour models. *International journal of computer vision*, 1(4):321–331, 1988.

[55] Yoshiaki Shirai and Hirochika Inoue. Guiding a robot by visual feedback in assembling tasks. *Pattern recognition*, 5(2):99–108, 1973.

[56] John Ashburner and Karl J Friston. Unified segmentation. *Neuroimage*, 26(3):839–851, 2005.

[57] Hao-Shu Fang, Jianhua Sun, Runzhong Wang, Minghao Gou, Yong-Lu Li, and Cewu Lu. Instaboost: Boosting instance segmentation via probability map guided copy-pasting. In *Proceedings of the IEEE/CVF International Conference on Computer Vision*, pages 682–691, 2019.

[58] Miguel Angel Garcia and Agusti Solanas. 3d simultaneous localization and modeling from stereo vision. In *IEEE International Conference on Robotics and Automation, 2004. Proceedings. ICRA'04. 2004*, volume 1, pages 847–853. IEEE, 2004.

[59] Janne Heikkila and Olli Silven. A four-step camera calibration procedure with implicit image correction. In *Proceedings of IEEE Computer Society Conference on Computer Vision and Pattern Recognition*, volume 97, page 1106. Citeseer, IEEE Comput. Soc, 1997.

[60] Z. Zhang. A flexible new technique for camera calibration. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 22(11):1330–1334, 2000.

[61] David Marr and Tomaso Poggio. Cooperative computation of stereo disparity. *Science*, 194(4262):283–287, 1976.

[62] John EW Mayhew and John P Frisby. Psychophysical and computational studies towards a theory of human stereopsis. *Artificial Intelligence*, 17(1-3):349–385, 1981.

[63] Kurt Konolige. Small vision systems: Hardware and implementation. In *Robotics research*, pages 203–212. Springer, 1998.

[64] Richard Hartley and Andrew Zisserman. *Multiple View Geometry in Computer Vision*. Cambridge university press, 2003.

[65] David A Forsyth and Jean Ponce. *Computer vision: a modern approach*. Prentice Hall Professional Technical Reference, 2002.

[66] Alex Krizhevsky, Ilya Sutskever, and Geoffrey E. Hinton. ImageNet classification with deep convolutional neural networks. *Communications of the ACM*, 60(6):84–90, May 2017.

[67] Ross Girshick. Fast r-CNN. In *2015 IEEE International Conference on Computer Vision (ICCV)*. IEEE, December 2015.

[68] Ross Girshick, Jeff Donahue, Trevor Darrell, and Jitendra Malik. Region-based convolutional networks for accurate object detection and segmentation. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 38(1):142–158, January 2016.

[69] Joseph Redmon, Santosh Divvala, Ross Girshick, and Ali Farhadi. You only look once: Unified, real-time object detection. In *2016 IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*. IEEE, June 2016.

[70] Wei Liu, Dragomir Anguelov, Dumitru Erhan, Christian Szegedy, Scott Reed, Cheng-Yang Fu, and Alexander C Berg. Ssd: Single shot multibox detector. In *European conference on computer vision*, pages 21–37. Springer, 2016.

[71] Paolo Fiorini and Zvi Shiller. Motion planning in dynamic environments using velocity obstacles. *The International Journal of Robotics Research*, 17(7):760–772, 1998.

[72] Animesh Chakravarthy and Debasish Ghose. Obstacle avoidance in a dynamic environment: A collision cone approach. *IEEE Transactions on Systems, Man, and Cybernetics-Part A: Systems and Humans*, 28(5):562–574, 1998.

[73] Tomas Lozano-Perez. Spatial planning: A configuration space approach. In *Autonomous robot vehicles*, pages 259–271. Springer, 1990.

[74] Javier Minguez and Luis Montano. Nearness diagram (nd) navigation: collision avoidance in troublesome scenarios. *IEEE Transactions on Robotics and Automation*, 20(1):45–59, 2004.

[75] John Bellingham, Michael Tillerson, Arthur Richards, and Jonathan P How. Multi-task allocation and path planning for cooperating uavs. In *Cooperative control: models, applications and algorithms*, pages 23–41. Springer, 2003.

[76] Yoko Watanabe, Anthony Calise, Eric Johnson, and Johnny Evers. Minimum-effort guidance for vision-based collision avoidance. In *AIAA atmospheric flight mechanics conference and exhibit*, page 6641, 2006.

[77] Anusha Mujumdar and Radhakant Padhi. Reactive collision avoidance of using non-linear geometric and differential geometric guidance. *Journal of guidance, control, and dynamics*, 34(1):303–311, 2011.

[78] Mykel J Kochenderfer, Jessica E Holland, and James P Chryssanthacopoulos. Next-generation airborne collision avoidance system. Technical report, Massachusetts Institute of Technology-Lincoln Laboratory Lexington United States, 2012.

[79] Xilin Yang, Luis Mejias Alvarez, and Troy Bruggemann. A 3d collision avoidance strategy for uavs in a non-cooperative environment. *Journal of Intelligent & Robotic Systems*, 70(1):315–327, 2013.

[80] Yazdi I Jenie, Erik-Jan van Kampen, Cornelis C de Visser, Joost Ellerbroek, and Jacco M Hoekstra. Three-dimensional velocity obstacle method for uncoordinated avoidance maneuvers of unmanned aerial vehicles. *Journal of Guidance, Control, and Dynamics*, 39(10):2312–2323, 2016.

[81] Dušan M Stipanović, Peter F Hokayem, Mark W Spong, and Dragoslav D Šiljak. Cooperative avoidance control for multiagent systems. 2007.

[82] Yasuhiro Kuriki and Toru Namerikawa. Consensus-based cooperative formation control with collision avoidance for a multi-uav system. In *2014 American Control Conference*, pages 2077–2082. IEee, 2014.

[83] Xiaohua Wang, Vivek Yadav, and SN Balakrishnan. Cooperative uav formation flying with obstacle/collision avoidance. *IEEE Transactions on control systems technology*, 15(4):672–679, 2007.

[84] Eric Boivin, André Desbiens, and Eric Gagnon. Uav collision avoidance using cooperative predictive control. In *2008 16th Mediterranean Conference on Control and Automation*, pages 682–688. IEEE, 2008.

[85] Michael R Hafner, Drew Cunningham, Lorenzo Caminiti, and Domitilla Del Vecchio. Cooperative collision avoidance at intersections: Algorithms and experiments. *IEEE Transactions on Intelligent Transportation Systems*, 14(3):1162–1175, 2013.

[86] Peter Seiler, Bongsob Song, and J Karl Hedrick. Development of a collision avoidance system. Technical report, SAE Technical Paper, 1998.

[87] Avraham D Horowitz and Thomas A Dingus. Warning signal design: A key human factors issue in an in-vehicle front-to-rear-end collision warning system. In *Proceedings of the Human Factors Society Annual Meeting*, volume 36, pages 1011–1013. SAGE Publications Sage CA: Los Angeles, CA, 1992.

[88] Yasuhiko Fujita, Kenji Akuzawa, and Makoto Sato. Radar brake system. *Jsae Review*, 1(16):113, 1995.

[89] Sunan Huang, Rodney Swee Huat Teo, and Kok Kiong Tan. Collision avoidance of multi unmanned aerial vehicles: A review. *Annual Reviews in Control*, 48:147–164, 2019.

[90] Nitthita Chirdchoo, W-S Soh, and Kee Chaing Chua. Aloha-based mac protocols

with collision avoidance for underwater acoustic networks. In *IEEE INFOCOM 2007-26th IEEE International Conference on Computer Communications*, pages 2271–2275. IEEE, 2007.

[91] Robert Bogue. Underwater robots: a review of technologies and applications. *Industrial Robot: An International Journal*, 2015.

[92] Giacomo Marani, Song K Choi, and Junku Yuh. Underwater autonomous manipulation for intervention missions auvs. *Ocean Engineering*, 36(1):15–23, 2009.

[93] Ingrid Schjølberg and Ingrid Bouwer Utne. Towards autonomy in rov operations. *IFAC-PapersOnLine*, 48(2):183–188, 2015.

[94] Thor I Fossen. *Handbook of marine craft hydrodynamics and motion control*. John Wiley & Sons, 2011.

[95] Geoff N Roberts and Robert Sutton. *Advances in unmanned marine vehicles*, volume 69. Iet, 2006.

[96] Gianluca Antonelli and G Antonelli. *Underwater robots*, volume 3. Springer, 2014.

[97] Robert D Christ and Robert L Wernli Sr. *The ROV manual: a user guide for remotely operated vehicles*. Butterworth-Heinemann, 2013.

[98] Pal Johan From, J Tommy Gravdahl, and K Ytterstad Pettersen. *Vehicle-manipulator systems*. Springer, 2016.

[99] Mauro Candeloro, Fabio Dezi, Asgeir J Sørensen, and Sauro Longhi. Analysis of a multi-objective observer for uuvs. *IFAC Proceedings Volumes*, 45(5):343–348, 2012.

[100] Günter Niemeyer, Carsten Preusche, Stefano Stramigioli, and Dongjun Lee. Telerobotics. In *Springer handbook of robotics*, pages 1085–1108. Springer, 2016.

[101] Francois Chaumette. Potential problems of stability and convergence in image-based and position-based visual servoing. In *The confluence of vision and control*, pages 66–78. Springer, 1998.

[102] Xiongfeng Yi, Furui Wang, Wenyu Zuo, Gangbing Song, and Zheng Chen. Robotics assisted smart-touch pipeline inspection. *International Journal of Intelligent Robotics and Applications*, 5(3):326–336, 2021.

[103] Rajeev Sharma and Seth Hutchinson. Motion perceptibility and its application to active vision-based servo control. *IEEE Transactions on Robotics and Automation*, 13(4):607–617, 1997.

[104] Peter I Corke and Seth A Hutchinson. A new partitioned approach to image-based visual servo control. *IEEE Transactions on Robotics and Automation*, 17(4):507–515, 2001.

[105] Nicholas R Gans and Seth A Hutchinson. Stable visual servoing through hybrid switched-system control. *IEEE Transactions on Robotics*, 23(3):530–540, 2007.

[106] Guillaume Morel, Thomas Liebezeit, Jérôome Szewczyk, Sylvie Boudet, and Jacques Pot. Explicit incorporation of 2d constraints in vision based control of robot manipulators. In *Experimental Robotics VI*, pages 99–108. Springer, 2000.

[107] Nicholas R Gans, Seth A Hutchinson, and Peter I Corke. Performance tests for visual servo control systems, with application to partitioned approaches to visual servo control. *The International Journal of Robotics Research*, 22(10-11):955–981, 2003.

[108] Koichi Hashimoto and Toshiro Noritsugu. Potential problems and switching control for visual servoing. In *Proceedings. 2000 IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS 2000)(Cat. No. 00CH37113)*, volume 1, pages 423–428. IEEE, 2000.

[109] Rudolph Emil Kalman. A new approach to linear filtering and prediction problems. *Journal of basic Engineering*, 82(1):35–45, 1960.

[110] Raman Mehra. On the identification of variances and adaptive kalman filtering. *IEEE Transactions on automatic control*, 15(2):175–184, 1970.

[111] Xiongfeng Yi and Zheng Chen. A robust visual tracking method for unmanned mobile systems. *Journal of Dynamic Systems, Measurement, and Control*, 141(7), 2019.

[112] Ramsey Faragher. Understanding the basis of the kalman filter via a simple and intuitive derivation [lecture notes]. *IEEE Signal processing magazine*, 29(5):128–132, 2012.

[113] Brian DO Anderson and John B Moore. Optimal filtering. *Englewood Cliffs*, 21:22–95, 1979.

[114] Emanuele Trucco and Alessandro Verri. *Introductory Techniques for 3-D Computer Vision*. Prentice Hall PTR, Upper Saddle River, NJ, USA, 1998.

[115] Nima Moshtagh. Minimum volume enclosing ellipsoid. *Convex optimization*, 111(January):1–9, 2005.

[116] John Canny. A computational approach to edge detection. In *Readings in Computer Vision*, pages 184–203. Elsevier, 1987.

[117] Xiongfeng Yi, Animesh Chakarvarthy, and Zheng Chen. Cooperative collision avoidance control of servo/ipmc driven robotic fish with back-relaxation effect. *IEEE Robotics and Automation Letters*, 6(2):1816–1823, 2021.

[118] Zheng Chen, Piqi Hou, and Zhihang Ye. Robotic fish propelled by a servo motor and ionic polymer-metal composite hybrid tail. *Journal of Dynamic Systems, Measurement, and Control*, 141(7), 2019.

[119] Zheng Chen and Xiaobo Tan. A control-oriented and physics-based model for ionic polymer–metal composite actuators. *IEEE/ASME Transactions On Mechatronics*, 13(5):519–529, 2008.

[120] Veiko Vunder, Andres Punning, and Alvo Aabloo. Mechanical interpretation of back-relaxation of ionic electroactive polymer actuators. *Smart Materials and Structures*, 21(11):115023, 2012.

[121] Xiaoqi Bao, Yoseph Bar-Cohen, and Shyh-Shiuh Lih. Measurements and macro models of ionomeric polymer-metal composites (IPMC). In *Smart Structures and Materials 2002: Electroactive Polymer Actuators and Devices (EAPAD)*, volume 4695, pages 220–227. International Society for Optics and Photonics, 2002.

[122] Zheng Chen, Stephan Shatara, and Xiaobo Tan. Modeling of biomimetic robotic fish propelled by an ionic polymer–metal composite caudal fin. *IEEE/ASME transactions on mechatronics*, 15(3):448–459, 2010.

[123] Xiongfeng Yi and Zheng Chen. A robust visual tracking method for unmanned mobile systems. *Journal of Dynamic Systems, Measurement, and Control*, 141(7), 2019.

[124] Rudolph Emil Kalman. A new approach to linear filtering and prediction problems. *Journal of basic Engineering*, 82(1):35–45, 1960.

[125] Animesh Chakravarthy and Debasish Ghose. Generalization of the collision cone approach for motion safety in 3-d environments. *Autonomous Robots*, 32(3):243–266, 2012.

[126] Kaiming He, Xiangyu Zhang, Shaoqing Ren, and Jian Sun. Deep residual learning for image recognition. In *Proceedings of the IEEE conference on computer vision and pattern recognition*, pages 770–778, 2016.

[127] Furui Wang and Gangbing Song. Bolt early looseness monitoring using modified vibro-acoustic modulation by time-reversal. *Mechanical Systems and Signal Processing*, 130:349–360, September 2019.

[128] Furui Wang, Siu Chun Michael Ho, Linsheng Huo, and Gangbing Song. A novel fractal contact-electromechanical impedance model for quantitative monitoring of bolted joint looseness. *IEEE Access*, 6:40212–40220, 2018.

[129] Furui Wang, Linsheng Huo, and Gangbing Song. A piezoelectric active sensing method

for quantitative monitoring of bolt loosening using energy dissipation caused by tangential damping based on the fractal contact theory. *Smart Materials and Structures*, 27(1):015023, December 2017.

[130] Xiongfeng Yi and Zheng Chen. Visual servoing control of underwater vehicle for pipeline tracking, submitted. *IEEE Transactions on Automation Science and Engineering*.

[131] Craig W Reynolds. Steering behaviors for autonomous characters. In *Game developers conference*, volume 1999, pages 763–782. Citeseer, 1999.

[132] Kjell J Gåsvik. *Optical metrology*. John Wiley & Sons, 2003.

[133] Richard O Duda and Peter E Hart. Use of the hough transformation to detect lines and curves in pictures. *Communications of the ACM*, 15(1):11–15, 1972.

[134] Joseph LaSalle. Some extensions of liapunov's second method. *IRE Transactions on circuit theory*, 7(4):520–527, 1960.

[135] Simon Pedersen, Jesper Liniger, Fredrik F Sørensen, Kenneth Schmidt, Malte von Benzon, and Sigurd S Klemmensen. Stabilization of a rov in three-dimensional space using an underwater acoustic positioning system. *IFAC-PapersOnLine*, 52(17):117–122, 2019.

[136] Chu-Jou Wu. *6-DoF Modelling and Control of a Remotely Operated Vehicle*. PhD thesis, Flinders University, College of Science and Engineering., 2018.

[137] Lituan Wang, Lei Zhang, and Zhang Yi. Trajectory predictor by using recurrent neural networks in visual tracking. *IEEE transactions on cybernetics*, 47(10):3172–3183, 2017.

[138] Zhifen Zhang, Guangrui Wen, and Shanben Chen. Weld image deep learning-based on-line defects detection using convolutional neural networks for al alloy in robotic arc welding. *Journal of Manufacturing Processes*, 45:208–216, 2019.

[139] Jie Jiang, G Jun Zhang, Xinguo Wei, and Xiao Li. Rapid star tracking algorithm for star sensor. *IEEE Aerospace and Electronic Systems Magazine*, 24(9):23–33, 2009.

[140] Przemyslaw Woznowski, Xenofon Fafoutis, Terence Song, Sion Hannuna, Massimo Camplani, Lili Tao, Adeline Paiement, Evangelos Mellios, Mo Haghighi, and Ni Zhu. A multi-modal sensor infrastructure for healthcare in a residential environment. In *2015 IEEE International Conference on Communication Workshop (ICCW)*, pages 271–277. IEEE, 2015.

[141] Lvwen Huang, Siyuan Chen, Jianfeng Zhang, Bang Cheng, and Mingqing Liu. Real-time motion tracking for indoor moving sphere objects with a lidar sensor. *Sensors*, 17(9):1932, 2017.