Control and Visual Navigation for Unmanned Underwater Vehicles



Zhizun Xu

School of Engineering
Newcastle University

This dissertation is submitted for the degree of
Doctor of Philosophy

School of Engineering September 2020

Abstract

Control and navigation systems are key for any autonomous robot. Due to environmental disturbances, model uncertainties and nonlinear dynamic systems, reliable functional control is essential and improvements in the controller design can significantly benefit the overall performance of Unmanned Underwater Vehicles (UUVs). Analogously, due to electromagnetic attenuation in underwater environments, the navigation of UUVs is always a challenging problem.

In this thesis, control and navigation systems for UUVs are investigated. In the control field, four different control strategies have been considered: Proportional-Integral-Derivative Control (PID), Improved Sliding Mode Control (SMC), Backstepping Control (BC) and customised Fuzzy Logic Control (FLC). The performances of these four controllers were initially simulated and subsequently evaluated by practical experiments in different conditions using an underwater vehicle in a tank. The results show that the improved SMC is more robust than the others with small settling time, overshoot, and error.

In the navigation field, three underwater visual navigation systems have been developed in the thesis: ArUco Underwater Navigation systems, a novel Integrated Visual Odometry with Monocular camera (IVO-M), and a novel Integrated Visual Odometry with Stereo camera (IVO-S). Compared with conventional underwater navigation, these methods are relatively low-cost solutions and unlike other visual or inertial-visual navigation methods, they are able to work well in an underwater sparse-feature environment. The results show the following: the ArUco underwater navigation system does not suffer from cumulative error, but some segments in the estimated trajectory are not consistent; IVO-M suffers from cumulative error (error ratio is about 3 - 4%) and is limited by the assumption that the “seabed is locally flat”; IVO-S suffers from small cumulative errors (error ratio is less than 2%).

Overall, this thesis contributes to the control and navigation systems of UUVs, presenting the comparison between controllers, the improved SMC, and low-cost underwater visual navigation methods.

I would like to dedicate this thesis to my loves

Declaration

I hereby declare that except where specific reference is made to the work of others, the contents of this thesis are original and have not been submitted in whole or in part for consideration for any other degree or qualification in this, or any other university. This thesis is my own work and contains nothing which is the outcome of work done in collaboration with others, except as specified in the text and Acknowledgements. This thesis contains fewer than 65,000 words including appendices, bibliography, footnotes, tables and equations and has fewer than 150 figures.

Zhizun Xu
September 2020

Acknowledgements

I would like to acknowledge my supervisors: Dr Rose Norman, Dr Maryam Haroutunian, Dr Alan J Murphy and Mr Jeff Neasham. Dr Rose has guided my research and taken much time on revising my papers. That is extreme valuable in my PhD career with the thesis. I would also like to thank her for advices in aspects of life. Dr Maryam Haroutunian has given me many important suggestions about my study and papers. Dr Alan J Murphy always comes up with a sharp observation on my work. Mr Jeff Neasham has lent me the underwater robot "VideoRay Pro 3" and are very happy to give me suggestions on engineering. I really appreciate my supervisors. Without them, I could not complete the PhD research.

Meanwhile, I would like to acknowledge the generous support of John and Vivien Prime in funding aspects of this work. They offer me the funding to purchase the advanced sensors. That is very helpful for collecting rich data.

During the working at school laboratory, technicians have supported me on technical work. Thanks for Mr Robert Hindhaugh who made the plastic grids for me. Thanks for Mr Ian Howard-Row who helped me on electronic problems.

Special thanks go to my friends Bowen Ma, and Xutian Xue. They standing in cold water operated the ROV. Without their help, I cannot finish my experiments.

Finally, I would like to thank my wife and my parents. My wife takes care my life and makes proof reading for the thesis. My parents are encouraging and funding me throughout my PhD study.

Table of Contents

Abstract iii

Declaration vii

Acknowledgements ix

List of Figures xiv

List of Tables xix

Nomenclature xxi

List of Publications xxvii

1 Introduction 1

1.1 History of Robotics . . . 1

1.1.1 Unmanned Underwater Vehicles (UUVs) . . . 3

1.2 Remotely Operated Vehicles (ROVs) & Autonomous Underwater Vehicles (AUVs) 4

1.2.1 Applications . . . 5

1.2.2 Brief History of ROVs . . . 8

1.2.3 Brief History of AUVs . . . 10

1.3 Challenging Problems related to Control and Navigation . . . 12

1.4 Aims, Objectives and Contributions . . . 14

1.4.1 Aims and Objectives . . . 14

1.4.2 Contributions . . . 14

1.5 Thesis Outline . . . 16

2 Literature Review 19

2.1 The Architectures of AI . . . 19

2.2 Control Theories . . . 23

2.2.1 Control Algorithms for UUVs . . . 23

2.3 Navigation Systems . . . 25

2.3.1 Generic Navigation Methods . . . 25

2.3.2 Conventional Underwater Navigation Approaches . . . 29

2.3.3 Visual Odometry . . . 34

2.3.4 Visual SLAM . . . 35

2.3.5 Underwater Optical Navigation Systems . . . 37
2.3.6 Quantitative Evaluation Methods For Visual Odometries or SLAMs . . 38
2.4 Summary . . . 44

3 The Experiment ROV 45
3.1 VideoRay Pro 3 . . . 45
3.2 Modification for the ROV . . . 47
3.3 Sensors . . . 51
3.3.1 IMU sensor . . . 51
3.3.2 Gyroscope . . . 52
3.3.3 T265 Camera . . . 53
3.3.4 Monocular Camera . . . 54
3.3.5 Ping Sonar . . . 54
3.4 Waterproof Solution & Buoyancy Balancing . . . 55
3.5 Sensors Communications & Data Collection Programmes . . . 58
3.6 Summary . . . 60

I Control Theories for UUVs 61

4 Mathematical Modelling for UUVs 63
4.1 Dynamic and Kinematic Modelling . . . 63
4.1.1 Generic Modelling . . . 63
4.1.2 Modelling for VideoRay Pro 3 . . . 64
4.2 Decoupled Mode . . . 66
4.2.1 Forward Speed Control Model . . . 66
4.2.2 Steering Equations of Motion . . . 67
4.2.3 Diving Equations of Motion . . . 67
4.3 Thruster Model . . . 67
4.4 Layout of the Thrusters . . . 68
4.5 Summary . . . 69

5 Control Methodologies 71
5.1 PID Controller . . . 71
5.2 Improved SMC Method . . . 72
5.2.1 Stability Analysis . . . 75
5.3 Backstepping Control . . . 76
5.4 Customised Second-Order Fuzzy Logic Control . . . 78
5.4.1 Stability Analysis . . . 79
5.5 Summary . . . 80

6 Simulations & Practical Experiments for the Controllers 81

6.1 Simulations . . . 81
6.1.1 Quantitative Evaluation Methods . . . 82
6.1.2 Path following . . . 86
6.2 Experiments . . . 87
6.2.1 Developing Control System . . . 87
6.3 Discussion . . . 92
6.4 Summary . . . 93

II Visual Navigation Methodologies for UUVs 95

7 Camera Models & Underwater Calibration 97
7.1 Camera Modelling . . . 97
7.1.1 Perspective Camera Model . . . 97
7.2 Fisheye Model . . . 100
7.3 Omnidirectional Camera Model . . . 100
7.4 Calibration Methods . . . 101
7.5 Underwater Calibration . . . 101
7.5.1 Calibration Results for Monocular Camera . . . 102
7.5.2 Calibration Results for T265 Stereo Camera . . . 103
7.6 Summary . . . 106

8 ArUco Underwater Navigation 109
8.1 Implementation and Methodology . . . 109
8.1.1 Robustness of the Proposed Method . . . 112
8.1.2 Data Collection . . . 113
8.2 Experimental Results and Discussion . . . 113
8.2.1 Quantitative Evaluation Method . . . 117
8.3 Summary . . . 120

9 Integrated Visual Odometry with Monocular Camera 121
9.1 Assumption of the Proposed Method . . . 121
9.2 Methodology . . . 122
9.2.1 FAST Corner Detector . . . 123
9.2.2 Optical Flow Tracking . . . 124
9.2.3 Positions of Feature Points . . . 124
9.2.4 Linear Estimation . . . 126
9.2.5 Non-linear Iteration Methods . . . 129
9.3 Summary . . . 131

10 Integrated Visual Odometry with Stereo Camera 133
10.1 3D Point Reconstruction using Sonar . . . 134

10.1.1 Gaussian Filter with Sonar Information 134
10.2 Incremental Translation Recovery 135
10.3 Fusion Algorithm 136
10.4 Nonlinear Optimisation 140
10.5 Summary 142

11 Implementation & Experiments for IVO Methods 143
11.1 Implementation of the IVO methods 143
11.2 Quantitative Evaluation Methods 143
11.3 Experiment Results for IVO-M Method 147
11.3.1 Discussion 155
11.4 Experiment Results for the IVO-S Method 156
11.4.1 Discussion 165
11.5 Comparisons between the IVO Methods and other Visual Navigation Methods . 166
11.6 Summary 170

12 Conclusion & Future Work 171
12.1 Conclusion 171
12.2 Future Work 173

References 175

Appendix A Parameter and Hydrodynamic Coefficients Of VideoRay Pro 3 187
A.1 The properties of VideorayPro 3 187

List of Figures

Figure 1.1 Eric Robot (Brown, 2017) . . . 2
Figure 1.2 Tortoise Robot (LeBouthillier, 1999) . . . 3
Figure 1.3 AUVs and ROVs . . . 4
Figure 1.4 A large work class ROV system (SeeByte, 1998) . . . 5
Figure 1.5 A LBL and USBL Aided Navigation Solution (LinkQuest, 1998) . . . 5
Figure 1.6 Applications of UUVs . . . 7
Figure 1.7 CURV-I and CURV-III ROVs . . . 8
Figure 1.8 CURV-I and CURV-III ROVs . . . 10
Figure 1.9 AUVs in 70s . . . 11
Figure 1.10 AUVs Exploration . . . 12
Figure 1.11 The relationship between the AI, the Fundamental Controllers, and Navigation Systems . . . 13

Figure 2.1 Layered Architecture with Planning Layer (Wooldridge, 2001) . . . 20
Figure 2.2 T-REX Agent (Rajan et al., 2013) . . . 22
Figure 2.3 IMU Platform (Woodman, 2007) . . . 28
Figure 2.4 The Mechanical Gyroscope (Woodman, 2007) . . . 28
Figure 2.5 Ultra Short Baseline Positioning (Vickery, 1998) . . . 31
Figure 2.6 Short Baseline Positioning (Vickery, 1998) . . . 32
Figure 2.7 Long Baseline Positioning (Vickery, 1998) . . . 32
Figure 2.8 Data Collecting Car (KITTI, 2016) . . . 39
Figure 2.9 Comparison between estimated trajectory and ground truth (Zhang and Singh, 2015) . . . 39
Figure 2.10 Translational Error of V-LOAM (Zhang and Singh, 2015) . . . 40
Figure 2.11 Rotational Error of V-LOAM (Zhang and Singh, 2015) . . . 40
Figure 2.12 DSO (Direct Sparse Odometry) and ORB (Orb SLAM2) Errors Plot (Engel et al., 2018) (RT means real-time, 5x RT means 5 times real-time) 41
Figure 2.13 The Digital Poster on Floor (Carrasco et al., 2016) . . . 42
Figure 2.14 Images Captured in Synthetic Underwater Environment (Hildebrandt and Kirchner, 2010) . . . 43

Figure 3.1 VideoRay Pro 3 . . . 46
Figure 3.2 The customised laptop connected to the Panel . . . 46

Figure 3.3 Mechanism Measuring the Hydrodynamic Coefficients (Wang and Clark, 2006) . . . 47
Figure 3.4 The VideoRay Pro 3 with additional tube . . . 48
Figure 3.5 The procedure of data and command transfer. The box with the dotted line shows the additions made to the system in the present work. . . . 49
Figure 3.6 The Sketch of the Vehicle with WiFi Antenna . . . 50
Figure 3.7 Electronic Tray Top view . . . 50
Figure 3.8 Electronic Tray Back view . . . 51
Figure 3.9 Top Tray (Unit: mm) . . . 51
Figure 3.10 Bottom Tray (Unit: mm) . . . 52
Figure 3.11 Camera Location (Unit:mm) . . . 54
Figure 3.12 Sonar Location . . . 55
Figure 3.13 Potting The Penetrators . . . 56
Figure 3.14 Vacuum Test . . . 57
Figure 3.15 Adding Weights on The ROV . . . 57
Figure 3.16 The Vehicle without Weight . . . 58
Figure 3.17 Balanced Buoyancy . . . 58
Figure 3.18 Sensors Recording Test . . . 59

Figure 4.1 Videoray Pro 3 . . . 64
Figure 4.2 The Relation Between Thruster output and Control Input (Wang and Clark, 2006) . . . 68
Figure 4.3 Thrusters Layout On VideoRay Pro 3 . . . 69

Figure 5.1 Fuzzy inference systems . . . 78
Figure 5.2 Fuzzified Input $\tilde{\psi}$ . . . 79
Figure 5.3 Fuzzified Input r . . . 79
Figure 5.4 Fuzzified Output τ_R . . . 79

Figure 6.1 Simulation without Noise, Heading Angle . . . 83
Figure 6.2 Simulation Control Signal without Noise . . . 83
Figure 6.3 Simulation With Added Noise, Heading Angle . . . 84
Figure 6.4 Simulation Control Signal with Noise (full response) . . . 84
Figure 6.5 Simulation Control Signal with Noise (three seconds truncation from the full response) . . . 84
Figure 6.6 Simulation Path Following Without Noise (Dotted line denotes the reference trajectory) . . . 86
Figure 6.7 Simulation Path Following with Noise (Dotted line denotes the reference trajectory) . . . 87
Figure 6.8 The Control System . . . 88
Figure 6.9 The Control Block Diagram . . . 88
Figure 6.10 Experiment Without Disturbances, Heading Angle . . . 90

Figure 6.11 Simulation With Disturbances (Amplitude: 0.030 m, Frequency: 1.00 Hz), Heading Angle 90
Figure 6.12 Experiment With Large Disturbances (Amplitude: 0.050 m, Frequency: 1.00 Hz), Heading Angle 91
Figure 6.13 Experiments With Tube, Heading Angle 91

Figure 7.1 Pinhole Camera Model 97
Figure 7.2 Chessboard on Ground 102
Figure 7.3 Calibration Samples 102
Figure 7.4 Perspective Camera Model for Monocular Camera. Images on the left side are original, while undistorted images are on the right side. . . . 103
Figure 7.5 Omnidirectional Camera Model for T265. Images on the left side are original, while undistorted images are on the right side. 104
Figure 7.6 Fisheye Camera Model for T265. Images on the left side are original, while undistorted images are on the right side. 105
Figure 7.7 Perspective Camera Model for T265. Images on the left side are original, while undistorted images are on the right side. 106

Figure 8.1 Steps of Recognition of ArUco Markers 109
Figure 8.2 Markers in the Towing Tank 110
Figure 8.3 Reference of the Markers 111
Figure 8.4 Rows and Columns 111
Figure 8.5 3D Trajectory with ArUco Navigation System 113
Figure 8.6 Lawnmower Shape 1st 114
Figure 8.7 Lawnmower Shape 2nd 114
Figure 8.8 Loop Trajectory 1st 115
Figure 8.9 Loop Trajectory 2nd 115
Figure 8.10 Loop Trajectory 3rd 116
Figure 8.11 Random Trajectory 116
Figure 8.12 The Marker and The Scale Chart 117

Figure 9.1 Fixed Reference and Body Reference 122
Figure 9.2 IVO-M approach 123
Figure 9.3 Feature Points (Green Markers) Extracted as From Underwater Image 124

Figure 10.1 Flow Chart of IVO-S Algorithm 133

Figure 11.1 Square Strap Scale in Towing Tank 144
Figure 11.2 Square Strap 3D in Towing Tank 144
Figure 11.3 Square Strap and Operator 145
Figure 11.4 Operator without Reference Strap 145
Figure 11.5 Estimated Trajectory in 3D 146
Figure 11.6 IVO-M Square Shape (1st test) with Reference, $e_r = 5.6\%$ 147

Figure 11.7 IVO-M Square Shape (2nd test) with Reference, $e_r = 2.3\%$ 148
Figure 11.8 T265 Square Shape (1st test) 148
Figure 11.9 T265 Square Shape (2nd test) 149
Figure 11.10 IVO-M Triangle Shape (1st test) with Reference, $e_r = 0.51\%$ 149
Figure 11.11 IVO-M Triangle Shape (2nd test) with Reference, $e_r = 0.90\%$ 150
Figure 11.12 T265 Triangle Shape (1st test) 150
Figure 11.13 T265 Triangle Shape (2nd test) 151
Figure 11.14 IVO-M Figure-8 Shape (1st test) with Reference, $e_r = 3.15\%$ 151
Figure 11.15 IVO-M Figure-8 Shape (2nd test) with Reference, $e_r = 4.35$ 152
Figure 11.16 T265 Figure-8 Shape (1st test) 152
Figure 11.17 T265 Figure-8 Shape (2nd test) 153
Figure 11.18 IVO-M Triangle without Reference, $e_r = 2.7\%$ 153
Figure 11.19 T265 Triangle without Reference 154
Figure 11.20 IVO-M Circle without Reference, $e_r = 3.9\%$ 154
Figure 11.21 T265 Circle without Reference 155
Figure 11.22 IVO-S Triangular 1st, error ratio: 0.0244 157
Figure 11.23 IVO-S Triangular 2nd, error ratio: 0.0080 157
Figure 11.24 IVO-S Triangular 3rd, error ratio: 0.0183 158
Figure 11.25 IVO-S Triangular 4th, error ratio: 0.0115 158
Figure 11.26 IVO-S Square 1st, error ratio: 0.0139 159
Figure 11.27 IVO-S Square 2nd, error ratio: 0.0118 159
Figure 11.28 IVO-S Square 3rd, error ratio:0.0189 160
Figure 11.29 IVO-S Figure-8 1st, error ratio: 0.0161 160
Figure 11.30 IVO-S Figure-8 2nd, error ratio: 0.0054 161
Figure 11.31 T265 Triangular 1st 161
Figure 11.32 T265 Triangular 2nd 162
Figure 11.33 T265 Triangular 3rd 162
Figure 11.34 T265 Triangular 4th 163
Figure 11.35 T265 Square 1st 163
Figure 11.36 T265 Square 2nd 164
Figure 11.37 T265 Square 3rd 164
Figure 11.38 T265 Figure-8 1st 165
Figure 11.39 T265 Figure-8 2nd 165
Figure 11.40 ORB-SLAM2 Trying to Initialise in Sparse-Feature Environment . . 167
Figure 11.41 The Trajectory Estimated by ORB-SLAM2 (The green line is the estimated trajectory, black dots are measured feature points, red dots are active feature points.) 167
Figure 11.42 The Trajectory Estimated by the IVO-M (The violet line is the estimated trajectory.) 168

Figure 11.43 The Trajectory Estimated by the IVO-S (white line presents the reference path, every blue grid is 0.2 metre, and the violet line is the estimated trajectory) 168
Figure 11.44 OKVIS with Few Matched Feature Points and Huge Drift Error . . . 169
Figure 11.45 VINE-Mono Trying to Extract Feature Points 169
Figure 11.46 VINS-Mono with Huge Drift Error 169

List of Tables

Table2.1 Summary of Evaluation Methods 44

Table3.1 Functions of Electrical Devices 48
Table3.2 The Costs and Precision of Components 49

Table5.1 Rules List 78

Table6.1 Tuning parameters 81
Table6.2 Simulation Results Analysis With Noises 85
Table6.3 Experimental Results Analysis 92

Table8.1 ArUco Out of the Water 1st Test (Unit:Metre) 118
Table8.2 ArUco Out of the Water 2nd Test (Unit:Metre) 118
Table8.3 Underwater ArUco method Quantitative Test (Unit: Metre) 119

Table11.1 Error Ratio and Distance of each Test Using IVO-M Method 147
Table11.2 Error Ratio and Distance of each Test Using IVO-S Method 156

TableA.1 Geometry and mass property 187
TableA.2 Linear drag coefficients 188

Nomenclature

Roman Symbols

B buoyancy force (N)

$\boldsymbol{C}$ position of camera centre

$\boldsymbol{d}_{sonar}$ sonar vector

d distance from thruster to center of VideoRay Pro 3 (m)

$\boldsymbol{e}$ error vector

E cost function

e_r error ratio

$e_{average}$ the average error

F set of points

f_x camera intrinsic coefficient

f_y camera intrinsic coefficient

H mapping function

I moment of inertia (kgm^2)

$\boldsymbol{J}$ jacobian Matirx

$\boldsymbol{K}$ camera intrinsic matrix

K_d PID control gain

K_i PID control gain

K_l coefficient in sliding mode control

K_p PID control gain

K' coefficient in Nomoto's first order model

$L(r)$	distortion function
$\boldsymbol{M}$	inertial matrix for UUV modelling
$\boldsymbol{m}$	point on image plane
m	element of inertial matrix
n	propeller revolution
N_r	element at damping matrix
$N_{\dot{r}}$	element at added matrix
$\boldsymbol{p}$	feature point
p_x	camera intrinsic coefficient
p_y	camera intrinsic coefficient
$\boldsymbol{R}$	Rotation Matrix
r	yaw, attitude of vehicle (rad)
r_d	radial distortion
S	sliding space
s_a	bias noise position error (m)
$\boldsymbol{T}$	Transformation Matrix
$\boldsymbol{t}$	Translational Vector
T	force from thruster (N)
T'	coefficient in Nomoto's first order model
U	speed factor in Nomoto's first order model
$\boldsymbol{v}$	vector of UUV velocities
V	Control Lyapunov function
Var_e	the variance of error
W	gravity force (N)
w	speed of diving for UUVs (m/s)
$\boldsymbol{X}_c$	point at 3D coordinate
$\boldsymbol{X}_p$	point at pixel coordinate

x_t state at time t

X_u element at damping matrix

$X_{\dot{u}}$ element at added matrix

Y_v element at damping matrix

$Y_{\dot{v}}$ element at added matrix

Z depth of pixel (m)

z_1' first step at backstepping control

z_2' second step at backstepping control

z_d depth (m)

z_t observe value

Z_w element at damping matrix

$Z_{\dot{w}}$ element at added matrix

Greek Symbols

α coefficients

β coefficients

ε white noise

ζ_s damping ratio

$\boldsymbol{\eta}$ vector of attitude and position for UUVs

η constant coefficient

θ_g angle error (rad)

κ stabilising function

λ_1 coefficient in sliding mode control

λ_2 coefficient in sliding mode control

$\boldsymbol{\xi}$ vector in Lie algebra

ξ pole-placement parameter

ξ_1 coefficient in sliding mode control

ξ_2 coefficient in sliding mode control

ρ	density of sea water (kg/m^3)
$\boldsymbol{\sigma}$	covariance Matrix
σ	variance in Gaussian Distribution
$\boldsymbol{\tau}$	vector of control inputs for UUVs control
$\boldsymbol{\psi}$	vector of UUV attitude
Ψ	heading angle (rad)
ψ_d	desired heading angle (rad)
ω_n	pole-placement parameter
ω_s	undamped natural frequency

Acronyms / Abbreviations

AI	Artificial Intelligence
AUV	Autonomous Underwater Vehicle
BC	Backstepping Control
CTD	Conductivity, Temperature, Depth
DLT	Direct Linear Transformation
DSO	Direct Sparse Odometry
DVL	Doppler Velocity Logger
FLC	Fuzzy Logical Control
GNSS	Global Navigation Satellite System
GPS	Global Positioning System
IMTP	Institute of Marine Technology Problems
IMU	Inertial Measurement Unit
KIT	Karlsruhe Institute of Technology
KITTI	Karlsruhe Institute of Technology and Toyota Technological Institute
LBL	Long Baseline Positioning
LMI	Linear Matrix Inequalities
MBES	Multibeam Echosounder

NASA National Aeronautics and Space Administration

NB Negative Big

NM Negative Middle

NS Negative Small

OF Optical Flow

PB Positive Big

PID Proportional Integral Derivative

PIG Pine Island Glacier

PM Positive Middle

PS Positive Small

PTAM Parallel Tracking and Mapping

RAS Russian Academy of Sciences

ROV Remotely Operate Vehicle

SBL Short Baseline Positioning

SBP Spontaneous Bacterial Peritonitis

SMC Sliding Mode Control

SSS Side-Scan Sonar

STRIPS Stanford Research Institute Problem Solver

SVO Fast Semi-Direct Monocular Visual Odometry

ToF Time Of Flight

USBL Ultrashort Baseline Positioning

UUV Underwater Unmanned Vehicle

VAN Visually Augumented Navigation

V-LOAM Visual-lidar Odometry And Mapping

VPU Visual Processing Unit

V-SLAM Visual Simultaneous Localisation And Mapping

ZB Zero Big

ZM Zero Middle

ZS Zero Small

List of Publications

1. Zhizun Xu, Maryam Haroutunian, Alan J Murphy, Jeff Neasham, and Rose Norman. A Comparison of Functional Control Strategies for Underwater Vehicles: Theories, Simulations and Experiments. *Ocean Engineering* (2020).

2. Zhizun Xu, Maryam Haroutunian, Alan J Murphy, Jeff Neasham, and Rose Norman. An Integrated Visual Odometry System for Underwater Vehicles. Accepted by *IEEE Journal of Oceanic Engineering* (2020).

3. Zhizun Xu, Maryam Haroutunian, Alan J Murphy, Jeff Neasham, and Rose Norman. An Integrated Visual Odometry System with Stereo Camera for Unmanned Underwater Vehicles. Submitted to *IEEE Journal of Oceanic Engineering* (2020).

Chapter 1. Introduction

1.1. History of Robotics

Nowadays robotic technology is widely applied in many fields. Robots have been sent to Mars, operated in oceans, utilised to clean floors in homes, used to accelerate production in factories and to fight fires (Squyres et al., 2003) (Penders et al., 2011) (Yuh et al., 2012) (Yan, 2009). Time and lives are save through robotic technologies. The implementations of robotic technologies have had a dramatic impact on various aspects of modern life, including medical systems, the industrial community, transportation, exploration of oceans and deep space (Yuh et al., 2012) (Wilcox and Gennery, 1990). As the new trend of Deep Learning technology is developing, robots with learning skills and high intelligence will be created soon.

Since the emergence of civilization, humans have created handicrafts based on their images. In ancient China, a Buddhist monk named Yi Xing invented a mechanical clock (Yan and Lin, 2002). In the Ming Dynasty, the "Exploitation of the Works of Nature" was published which recorded work on agriculture and handcrafts (Golas, 2007). At the same time, automata such as Jacquet Droz were created in Europe (Voskuhl, 2007). In the late 19th century, torpedoes that could be remotely controlled emerged. They are considered as prototypes of remotely controlled vehicles (Melia, 1991). In 1877, Louis Brennan invented the Brennan torpedo, driven by propellors (Brown, 1932). An operator on the shore pulled on wires extending from inside the torpedo to spin the propellors. The speed of the torpedo was controlled by the speed of pulling the wires. In 1883, a British inventor, Ernest Wilson, invented a torpedo controlled by radio waves (Miessner, 1916). In this case, the torpedo could be controlled by a wireless device mounted on a shore or a ship.

Robot is not a new word, it originated from the Slavic language where labourers who were forced to work were called robota (Szabolcsi, 2014). In the Capek's fictional story, artificial human bodies without any thoughts were created by a new technology. In 1920, the word *robot* was first presented to describe artificial automata (Szabolcsi, 2014).

Eric, shown in Figure 1.1, is well known as the first human robot and was exhibited and made a speech in London in 1928 (Rani et al., 2006). The body of this robot was made of aluminium, and it was driven by electromagnets and a motor. Its hands and head can move by remote control. The Televox was built by Westinghouse Electric Corporation in 1926, and was made of cardboard cut-outs connected by various equipment which allowed users to switch it on or off

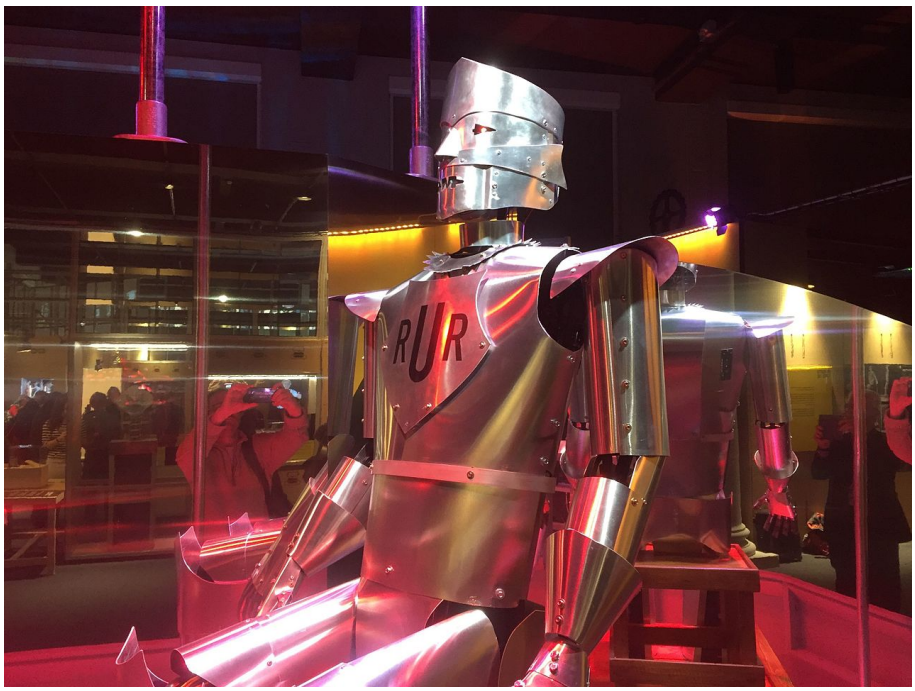


Figure 1.1 Eric Robot (Brown, 2017)

(Sharkey and Sharkey, 2009). A humanoid robot named Elektro was created in 1939 (Behnke, 2008). The users could give voice commands to control it walking around, moving its head and arms. It could also pretend to blow up balloons, and smoke. Steel gears, motors and skeleton hidden in an aluminium body supported these motions. The Japanese biologist Makoto Nishimura designed Japan's first robot named Gakutensoku in 1928, which has cognitive ability and can express emotions (Frumer, 2020).

The robots *Elmer* and *Elsie* built by William Grey Walter in 1948 were the first electronic robots (Demetriou, 2011). He aimed to show that the complicated connections among a small number of nerve cells could imitate how a brain works and accomplish complex behaviours. The two robots were three-wheeled, as shown in Figure 1.2. Because of their low velocity and the shapes of their bodies, people called them tortoises. They could exhibit phototaxis, motion in response to light, and recharge their batteries by themselves. Walter claimed novelty in that the brain processes could be stimulated by pure analogue signals.

In 1954, the first programmable robot arm named *Unimate* was created by George Devol, which started a new era of robotics production (Wallén, 2008).

After that, due to the development of electronic technologies and the digital computer, robots were developed that could be easily programmed by applying modern computer technology. That technology gave birth to industrial robots in 1978 which played important roles in factory automation (Wallén, 2008).

The research on robotics in the 1990s was accelerated because of the need to keep humans safe in hazardous environments, reduce errors caused by human operations, and reducing workload (Wallén, 2008). In addition to that, robots were utilised in other fields, including the food

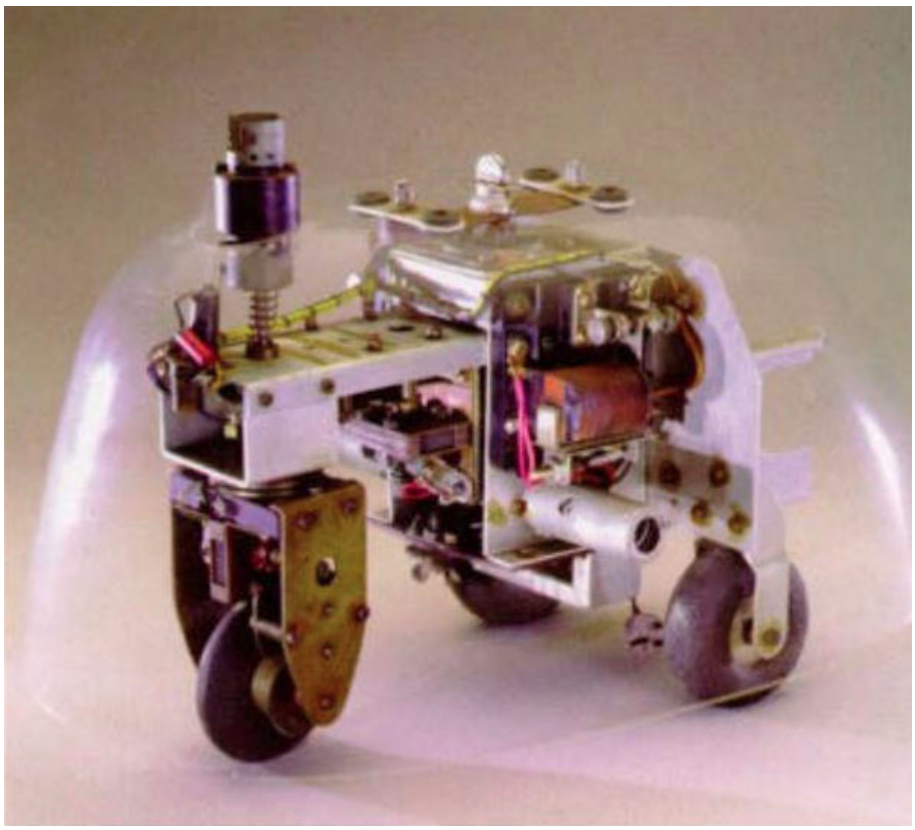


Figure 1.2 Tortoise Robot (LeBouthillier, 1999)

production, electronic operations, chemical experiments, and space exploration (Niku, 2001). Recently, robots have been implemented in household use, such as cleaning and cooking. After entering the 21st century, robotics has been integrated with artificial intelligence technology so that some smart robots could make decisions and manage their mission plans. The operator only needs to set up the target for the robot, rather than conduct pre-programmed mission plans. Recently, deep learning has drawn much attention. Such methods have been applied in mobile phones for example to detect owners' faces. In robotics, more and more researcher are trying to apply deep learning methods on robots (Sünderhauf et al., 2018).

Actually, the field of Robotics Science consists of three parts: perception, analysis and reaction (Siciliano and Khatib, 2016). The reaction of a robot system includes: movements by wheels, thrusters, and machine legs; or manipulation by machine arms, and grippers. The perception includes sensing the environment, providing the pose and position of the robot, and supplying information on the surrounding environment. The analysis is an intelligent connection between reaction and perception, and it may be a programming AI architecture, mission planning system, or decision-making algorithm.

1.1.1. Unmanned Underwater Vehicles (UUVs)

Unmanned Underwater Vehicles (UUVs) are underwater vehicles without human occupation. They can be considered as a subset of robots. Modern UUVs possess perception sensors, control



(a) AUV Launch (MBARI, 2004)



(b) ROV (SMD Company)

Figure 1.3 AUVs and ROVs

systems and communication equipment. They can sense the environment and undertake specific path planning. Hence, UUVs are essentially robots that work in underwater environments. A detailed introduction is presented in following sections.

1.2. Remotely Operated Vehicles (ROVs) & Autonomous Underwater Vehicles (AUVs)

UUVs are classified into two groups: Remotely Operated Vehicles (ROVs), which are operated by humans from a remote location, and Autonomous Underwater Vehicles (AUVs), which are independent of human operations. An ROV and an AUV are presented in Figure 1.3. ROVs are generally tethered to onshore bases or supporting ships by electric cables which may supply the power and the communication signals to the ROV. Because of the continuous and sufficient power supply, ROVs are widely applied in Offshore Engineering. Conversely, AUVs generally rely on their own batteries to remove the cable limitations. Hence, AUVs execute missions by pre-programming commands or applying Artificial Intelligence (AI) techniques and can reach some restricted areas. For this reason AUVs are commonly applied in marine science studies (Wynn et al., 2014).

A portable ROV system may simply consist of a small-size vehicle, a control box and a simple tether spool. However, work class ROV systems involve a control room, a launch and recovery system, a tether management system, and the actual vehicle. An example of a large work class ROV system is presented in Figure 1.4. For AUV systems, the situation is different. In order to track the state of an AUV, a submersible positioning and navigation system is set out prior to vehicle launch. As shown in Figure 1.5, the transponders are fixed on the seabed to make up a transponder array, by which the AUVs can be localised using acoustic methods. However the AUV typically does not need a tether management system (Wynn et al., 2014).

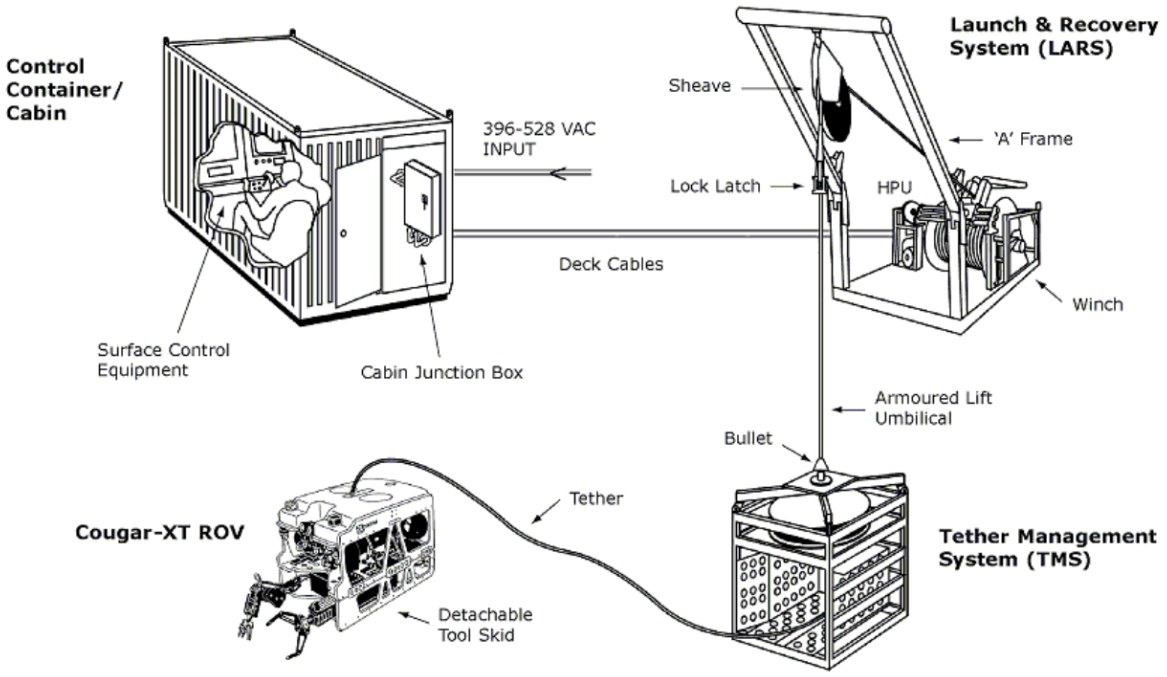


Figure 1.4 A large work class ROV system (SeeByte, 1998)

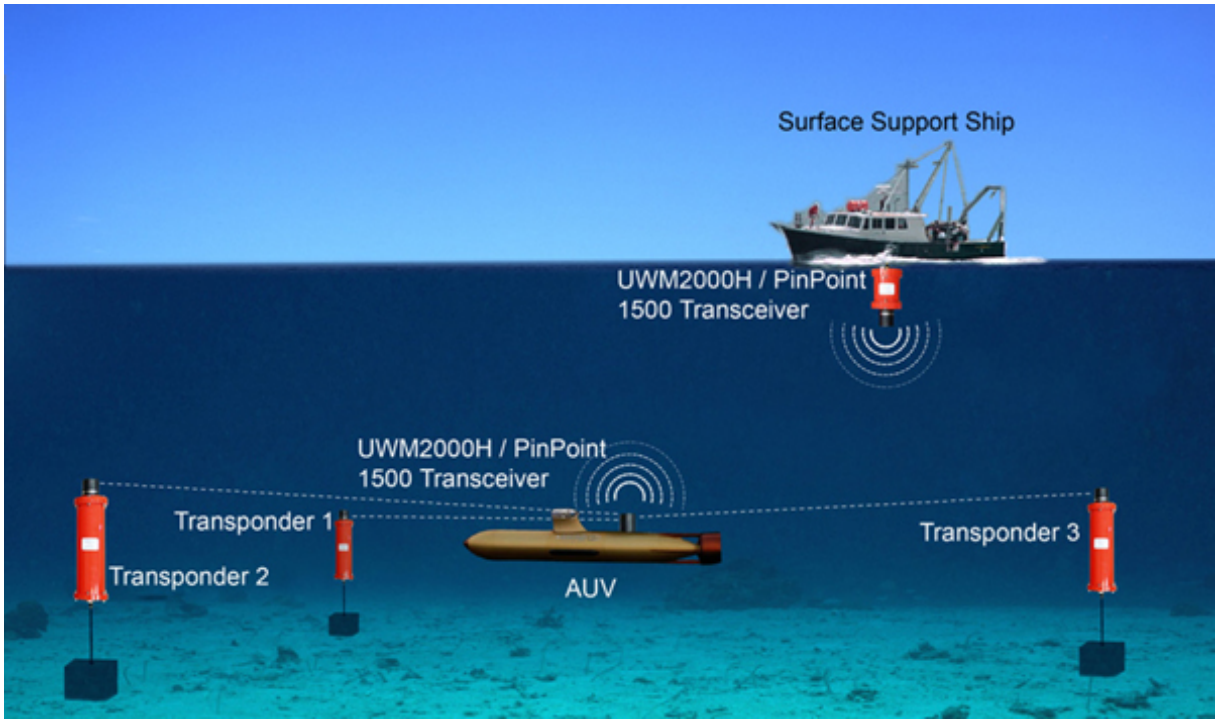


Figure 1.5 A LBL and USBL Aided Navigation Solution (LinkQuest, 1998)

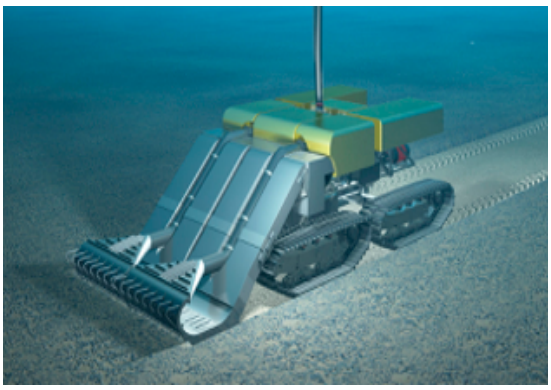
1.2.1. Applications

According to Christ and Wernli Sr (2013), the range of ROV applications is as follows:

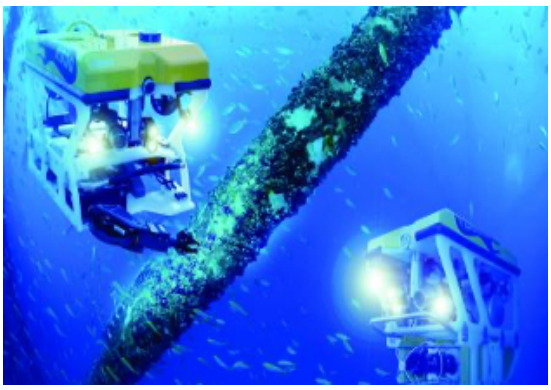
- Science research: The research organisations, such as governments, industrial companies, and universities, utilise ROVs to gather sensor data and collect samples from the

underwater environment. Frequently, manned underwater vehicles and human divers are utilised to conduct scientific investigations in shallow areas. Compared with manned submersibles, the costs of ROVs are quite low because life-support equipment is not necessary. Since ROVs are underwater robots, even if an accident happened, the no lives would be lost.

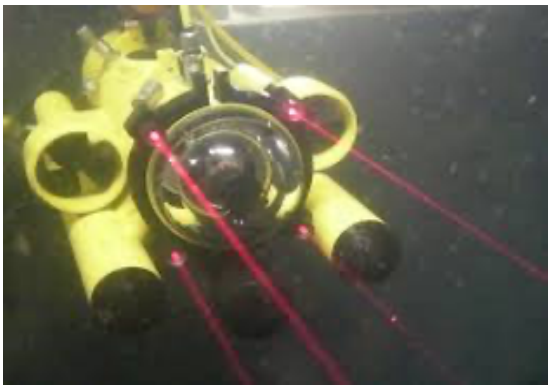
- Military: ROVs play important roles in military applications. They can execute recovery, retrieval, mine countermeasures, security and inspection tasks. More specifically, work class, huge-sized ROVs are utilised to execute recovery missions, and they are usually equipped with lifting gear for raising objects up to the surface. ROVs with a simple video camera and navigation devices can execute inspection missions.
- Fish Farms and Aquaculture: ROVs can be used to detect and monitor fish distribution. Meanwhile ROVs may be utilised to collect trepangs or other marine plants. In this way, ROVs increase the efficiency of fishing and aquacultural activities.
- Security: ROVs are able to inspect different vulnerable areas periodically.
- Public safety: The public safety is an important topic. There are many boaters who drown and ships that sink every year. Once this has happened, emergency measures including search and rescue are performed by rescue teams. ROVs can strengthen the team's capability. They can be used to gather evidence and recover victims' items from the sea.
- Oil & Gas drill support: ROVs may be applied in the process of searching for hydrocarbons, and assisting the drilling operations in deep water. When depth is more than 100 metres, ROV Support has become the standard for all drilling functions, from first spud-in to well completion. Tasks include observing the seafloor environment, sealing well cases, and directing other operations for tools and drilling equipment around the well. The regulations (API S53) published recently require a second ROV on standby, operated from a vessel that is separated from the drilling rig.
- Maintenance, repair, inspection and mining: ROVs may be utilised to maintain subsea equipment, surface structures of offshore wind turbines, fish farming, marine husbandry services, and support for oil and gas industries. The advantage of ROVs is that they can supply live data to the operator so that experts can identify if there are any problems immediately. Sometimes, ROVs are used to make modifications on subsea fixtures, as shown in Figure 1.6c and Figure 1.6b. ROVs with multiple advanced sensors are sent to scan the seabed to identity whether there is a mine. Some special ROVs can execute mining missions, as shown in Figure 1.6a.



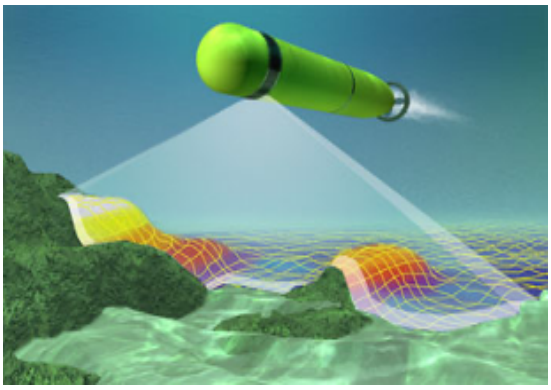
(a) Deep Mnining (IHC, 2019)



(b) Pipline Inspection (ECA, 2017)



(c) Underwater Observation (TECH, 2018)



(d) Seabed Scanning (MBARI, 2018)

Figure 1.6 Applications of UUVs

In contrast with ROVs, AUVs are mainly utilised to collect data, as shown in Figure 1.6d. According to Wynn et al. (2014), the sensors deployed on the AUVs involve instruments which are used in geophysical surveys, such as magnetometers, tools for collecting seafloor data, such as monochrome cameras, and measurements used in oceanography. The types of sensors which are expected to be installed on AUVs are determined by the missions to be executed by the AUV. Meanwhile, the travelling velocity and depth of the AUV is determined by these scientific sensors. For example, colour or monochrome cameras require the AUV to travel at slow speed and to be close to seafloor (Jerosch et al., 2006).

Sometimes, environmental disturbances may limit the operations of AUVs. For example, when AUVs execute geoscience missions, the general travelling speed is about 2.0 m/s, if environmental disturbances such as current are large in the area, the vehicles will have difficulties collecting data; and AUVs are not normally operated in shallow waters, because of the impacts of tidal currents and the risk of collisions with artificial structures. In addition to that, some areas such as military areas and fish farms, are restricted for AUVs, due to the high risk of collision and acoustic disturbances. Furthermore, since marine animals may be disturbed by acoustic or other devices, AUVs are not allowed to operate in protected areas.

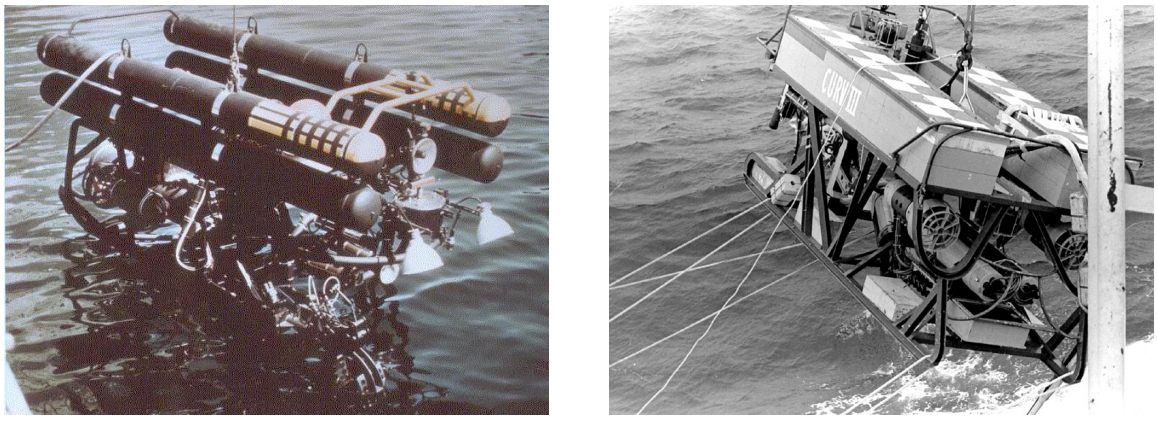
Even though there are many limitations for AUVs, collecting data using AUVs is still a common method in geoscience research. Initially, AUVs were mainly used for imaging the seafloor.

Recently, the application of AUVs has been extended to oceanographic or geochemical surveys including:

- Underwater volcanism and hydrothermal distribution (Wynn et al., 2014);
- Features of underwater fluids at low temperature states (Das et al., 2012);
- Benthic habitat mapping (Williams et al., 2010);
- Features of Seafloor morphology (Grasmueck et al., 2006).

1.2.2. Brief History of ROVs

Military implementations have spawned ROV technologies and promoted their development. In the initial stages, government support played significant roles in the construction and technological upgrades of ROVs. In the 1950s, the Royal Navy developed the Cutlet ROV to find lost torpedoes (Matsumoto and Potts, 2011). In 1961, the US Navy created the XN-3 dynamic underwater camera system, which was applied to recover torpedoes. Due to the satisfactory performances, the underwater camera system was upgraded on the Cable-controlled Underwater Research Vehicle (CURV). In 1966, the CURV I, shown in Figure 1.7a, accomplished hydrogen bomb recovery missions. The CURV III, shown in Figure 1.7b, was applied to successfully rescue a Canadian Commercial submersible in 1973. The CURV-III had also been used to explore the wreck of the *SS Edmund Fitzgerald*.



(a) CURV-I ROV (Pike, 1999) **(b)** CURV-III ROV (Pike, 1999)

Figure 1.7 CURV-I and CURV-III ROVs

Since the success of the CURV ROVs, the US Navy has continued designing new generations of ROVS. The huge-size ROV, named the Pontoon Implacement Vehicle (PIV), was developed to aid with recovery of sunk submarines. Conversely, the first portable ROV, called SNOOPY, with two controllable hydraulic thrusters and automatic depth control was also developed by the US Navy (Fugitt, 1975). Meanwhile, the first commercial ROVs, the Remotely Controllable

Vehicles (RCV) were released however, they did not draw much attention, because the market was dominated by manned underwater vehicles.

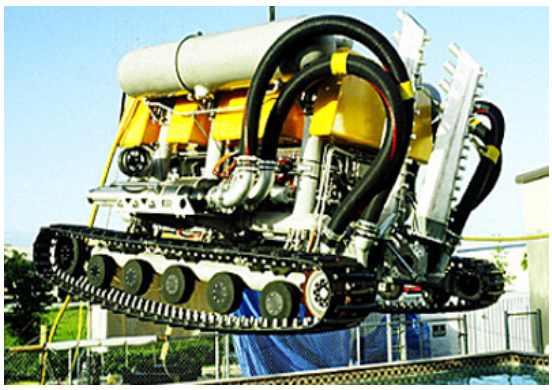
The ROV market grew rapidly from 1974 to 1982. The industrial companies started playing important roles in the development of ROVs. At the start of 1982, 85% of ROV projects were supported by governments, but by end of 1982, the proportion of ROVs owned by industrial companies had grown by 96% (Christ and Wernli Sr, 2013). With the required technology advancing, the reliability of ROVs improved significantly. Thus, in the 1980s, ROVs started being applied into offshore engineering. For example, pipeline inspection with ROVs was reported by Braathen and Sandford (1986), and MacGregor et al. (1983).

In the 1990s, the ROV industries increasingly matured and diversified. Furthermore, the requirements of underwater oil and gas discoveries exceeded the limitations of manned vehicles. Thus, ROVs were widely used by the offshore industry to install, maintain and repair the subsea facilities.

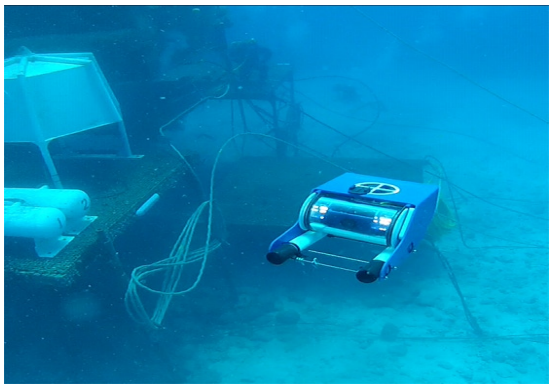
According to The World ROV Market Forecast 2011-2015 (Westwood, 2016), there were more than seven hundred and forty-seven work-class ROVs owned by about twenty-one companies in 2011. Coming into the 21st century, with the technological advancements, especially in computer science and sensor techniques, the development of ROVs entered into a new era. The new technology upgrades reduced the cost of ROVs. In this way, a small group can build its own ROV, such as OpenROV, which was built in a garage. That makes ROVs dramatically more diverse. Now ROVs may be divided into four types, work-class ROV, Light Work Class ROV, Observation Class ROV, and Micro or Mini ROV. On the other side, after advanced sensors were installed on the ROVs, such as side-scanned sonars, high resolution cameras, high precision compasses, Doppler Velocity Logs (DVLs) and elaborate Inertial Measurement Units (IMUs), an ROVs was able to localise itself and sense the environment. Applying intelligent algorithms, the workload of an ROV pilot can be reduced, and some human errors may be avoided.



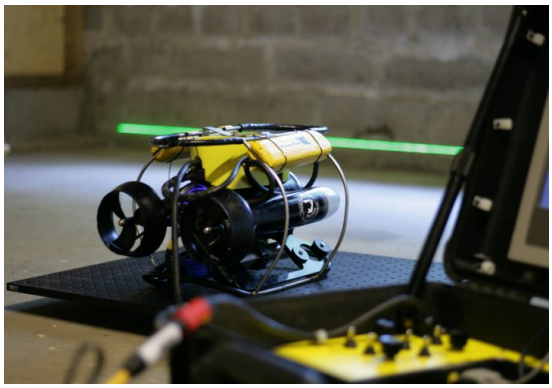
(a) Light Work Class ROV (PSS, 2020)



(b) Work Class ROV (RigZone, 2018)



(c) Observation Class ROV (FAST, 2017)



(d) Micro ROV (PDFSIM, 2020)

Figure 1.8 CURV-I and CURV-III ROVs

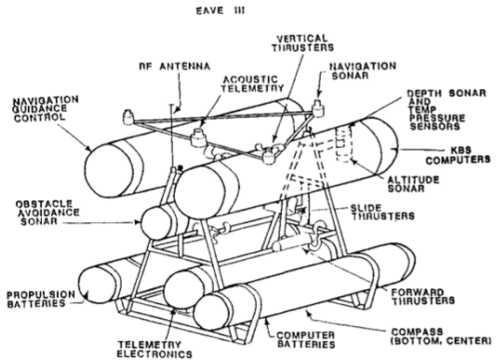
1.2.3. Brief History of AUVs

Similar to ROVs, the development of AUVs started in the 1960s to gather data or conduct special missions (Blidberg, 2001).

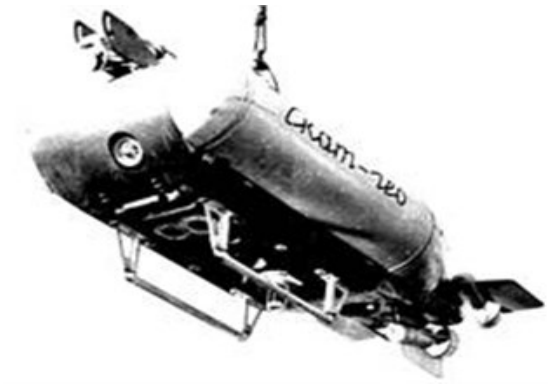
In the 1970s, two AUVs, UARS and SPURV, were developed by the University of Washington to collect oceanograhpic data in the Arctic regions. An open space-frame AUV named by EAVE constructed by the University of New Hampshire's Marine Engineering Laboratory, as shown in Figure 1.9b. In (Jalbert et al., 1988), an AUV was used as a testbed for the study of advanced concepts, such as multi-vehicle cooperation, hardware communication, and artificial intelligence algorithms. Meanwhile, Russia started the research about AUVs, the Institute of Marine Technology Problems (IMTP) in Vladivostok cooperated with the Russian Academy of Sciences (RAS) to develop the SKAT vehicles, and the first deep diving AUVs L1 and L2. However, the aim of the work conducted from 1970 to 1980 was to explore the potential of these autonomous systems. These AUVs are shown in Figure 1.9.



(a) SPURV II (Zhilenkov, 2016)



(b) EAVE III (Jalbert et al., 1988)



(c) SKAT-geo (Zhilenkov, 2016)



(d) L-2(Zhilenkov, 2016)

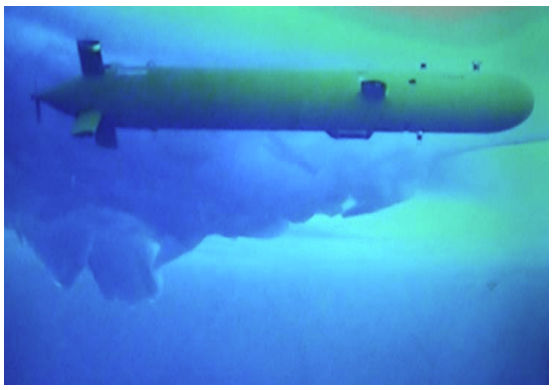
Figure 1.9 AUVs in 70s

After the 1980s, the technology advancement in the micro-electronics field contributed to the area of AUVs. The small-size sensors and computers enabled AUV to implement complicated navigation methods, guidance systems and control algorithms. Unfortunately, even with these improvements, there were still lots of difficulties to make AUVs into practical systems (Blidberg, 2001). However, the AUVs have drawn more and more attention. The size of the "International Symposium on Unmanned Untethered Submersible Technology" (UUST) grew from 24 technologists in 1987 to 320 people by 1997. In the USA, significant funding was provided for developing concept prototypes of AUVs. The DARPA Lab designed and built "UUV I" and "UUV II" to validate new technologies for the US Navy's tactical systems and the autonomous mine-hunting and mapping mission.

From 1990 to 2000, AUVs were advanced from concept prototypes into operational vehicles, which were able to execute some surveying operations. At the end of the 1990s, a large AUV with the most accurate internal navigation could be within 200-metres accuracy after surveying 50 nautical miles. In a 1999 U.S Report, efforts were made to meet military oceanographic survey requirements (Bunce et al., 1999) given by NAVOCEANO. There were four exercise participants; the Woods Hole Oceanographic Institution built the Environmental Measuring Units (REMUS) AUV; the Massachusetts Institute of Technology's Sea Grant Laboratory designed Odyssey AUV, which was 82 inches long, 22 inches in diameter; the Florida Atlantic University developed Ocean Explore AUV; the Naval Postgraduate School presented the Phoenix



(a) Autosub III AUV (McPhail et al., 2009)



(b) Arctic Explorer under Ice (Kaminski et al., 2010)

Figure 1.10 AUVs Exploration

AUV which was seven feet long, and 1.5 feet in breadth. Most of them were equipped with Side Scan Sonars (SSS), Doppler Velocity Logs (DVL), acoustic positioning systems and GPS receivers. An acoustic Doppler Velocity Log (DVL) can calculate the Doppler shift to estimate velocity relative to the seabed. There are at least three beams on it to transmit acoustic pulses, and each of them points in a different direction. However, environmental noise will impact on its performance, since it operated using acoustic signals. Furthermore, a DVL is required to be near the seabed, which is called bottom lock. The general the distance of bottom lock is about 50m.

After the 2000s, the applications of AUV technology were introduced into commercial tasks. AUVs were able to carry a variety of advanced sensors related to the marine geoscience area (Wynn et al., 2014). Thus, AUVs have been widely applied in marine geoscience. In (Chatzichristofis et al., 2012) a swarm of AUVs was utilised to explore an unknown environment. The Autosub3 AUV has been used to explore beneath the Pine Island Glacier (PIG) Ice Shelf (McPhail et al., 2009); shown in Figure 1.10b. In April 2010, the Arctic Explorer completed over 1000 km of under-ice surveying, mapping the sea floor during 10 days of continuous underwater operation (Kaminski et al., 2010); shown in Figure 1.10b.

1.3. Challenging Problems related to Control and Navigation

As mentioned above, the development of UUVs benefits from the advancements of electronic technologies. Small-size but high precision sensors, such as DVL, IMU and SSS, have become common devices in ROVs and AUVs. Micro single board computers with low power consumption and high-speed computational capability, which is comparable to desktop computers are available to allow the vehicles to implement Artificial Intelligent algorithms (AI). However, reliable fundamental controllers, which link the hardware dynamic systems and execute the commands from high-level AI modules, and navigation systems, which can provide the pose information and sense the environment for decision-making algorithms, are required to support the AI applications on the vehicles. The relationship between these systems is presented in Figure 1.11.

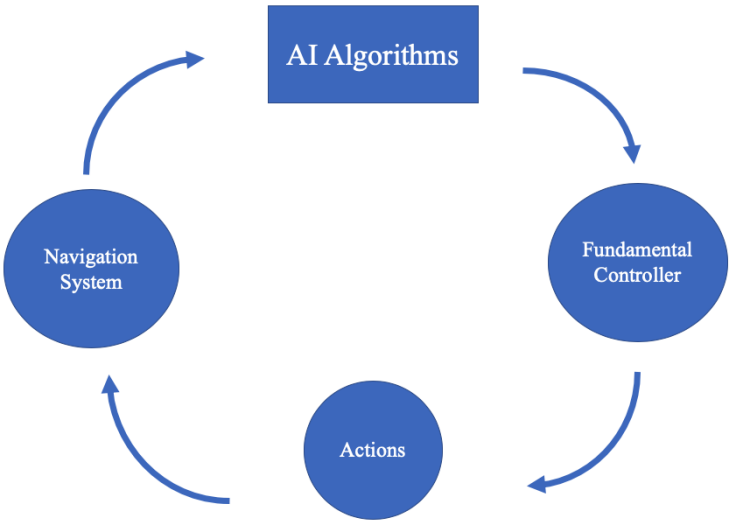


Figure 1.11 The relationship between the AI, the Fundamental Controllers, and Navigation Systems

The challenge of control lies in its' highly nonlinear characteristics. Unlike a car which has a rigid body dynamic model, underwater vehicles operate in a hydrodynamic environment that introduces nonlinear factors into the UUVs' mathematical model. Thus, if linear control was used after linearising the nonlinear parts of the model, this solution would be effective only under the specific conditions, for example at fixed cruise speed and with no large disturbances. Variations in the environmental conditions will change the hydrodynamic coefficients, and cause the failure of linear controllers. There have been numerous reports about applying nonlinear controllers or intelligent controllers to tune UUVs (Rajan et al., 2013) (Fossen and Ross, 2006) (Fossen, 2011). However, most of these just simulated the controllers, they lacked quantitative evaluation, and only focused on a single type of controller. These are discussed in detail in Chapter 2.

Navigation problems are always challegning. In the past, people invented a compass, and a sextant to navigate ships. More recently, satellites have been launched to localise and position of airplanes and ships. Unfortunately, due to the electromagnetic wave attenuation in water, it is impossible for UUVs to receive electromagnetic waves from the air. In this GPS denial space, UUVs have applied unbounded-error inertial senors, expensive arrays of acoustic beacons, and a combination of Ultra Short Base Line acoustic communication to estimate their pose (Wynn et al., 2014). Unlike the ground environment, there are sometimes few landmarks on the seabed. The visual navigation systems designed for mobile cars are not effective in the underwater environment. The drawbacks of these conventional methods will be discussed in Chapter 2.

It is normal to implement robust controllers and precise navigation systems on AUVs because they are separated from human interaction. In contrast, ROVs are connected to onshore control rooms or support ships and communication signals and power can be supplied by tethers. That means that ROVs are able to keep in touch with operators or ROV pilots all the time. It might seem that ROVs do not need automatic control and navigation methods. However, in deep water with few physical references, even experienced ROV pilots also need navigation systems to aid

their operation. According to Gagnon et al. (2012), human errors are the cause of numerous ROV accidents. In (Williams, 2004), it was reported that between 21 and 67% of ROV accidents were caused by human-factors related errors. in the US Army, Navy and Air Force. In this case, fundamental controllers can help pilots to maintain the heading angle and depth to reduce their workload. Furthermore, advanced fundamental controllers can tune ROVs to complete pre-programmed motions. Hence, AI algorithms for ROVs require fundamental controllers and navigation systems to decrease human errors.

1.4. Aims, Objectives and Contributions

1.4.1. Aims and Objectives

As described in the previous section, control and navigation are challenging problems for UUVs. The aim of the thesis is to investigate reliable controllers and low cost navigation solutions for UUVs. Reliable controllers must be robust to environmental disturbances and variation of hydrodynamic coefficients. Hence, controllers are required to be evaluated under different conditions. Low-cost navigation solutions are required to provide accurate estimations in underwater environments with the fusion techniques integrating the information from inertial and optical sensors. The objectives are listed as follows.

- Modelling the UUVs, and designing controllers. In this part, a mathematical model for UUVs is presented. The controllers are designed based on the model.
- Verification of the controllers via simulations and physical experiments. The designed controllers are evaluated by simulations and physical experiments at different conditions.
- Building a platform to collect underwater data. A platform is operated in underwater environments to collect data from optical sensors and inertial sensors.
- Development low-cost underwater visual navigation systems. Several underwater visual navigation systems are derived in this part.
- Verification of these navigation methods with the collected data. The visual navigation systems are evaluated by the collected data.

1.4.2. Contributions

In the control section of the work, four different controllers have been derived, simulated, and tested on an underwater vehicle. They are: PID controller, Fuzzy Logical Controller (FLC), Backstepping Controller (BC), and Sliding Mode Controller (SMC). The proposed SMC is different to the conventional method, as it combines linear control characteristics with the

advantages of nonlinear controllers. The rule set in the proposed FLC has been customised for the tested ROV. All of the controllers have been tested under various environmental conditions. In order to investigate the robustness to hydrodynamic characteristics, these controllers with the same coefficients were evaluated on a modified version of the vehicle. The novelties of the control part are listed as,

- A comparison among different types of controllers was conducted by simulations and practical experiments.
- An improved Sliding Mode Control combining linear and nonlinear characteristics, and a customised Fuzzy Logical Control were investigated.

In the navigation section of the work, a marker navigation system has been investigated based on ArUco Markers. The strength of the marker navigation is that there is no accumulative error. Once the algorithm detects the marker, the position error is only caused by the current marker, rather than previous information. The weakness of the method is that the estimated trajectories are not consistent at some segments, because of marker occlusion. In addition to the marker navigation, there are two novel Integrated Visual Odometries (IVO). Unlike pure visual odoemtries, the IVOs benefit from fusing data from multiple sensors (sonar, IMU, gyroscope), and estimate the pose of the camera by using the optical flow algorithm and the nonlinear regression technique. That enables the IVOs to work well even in a sparse environment with low cost sensors. Practical experiments demonstrated that the IVOs have higher accuracy than most pure visual odometries in the KITTI benchmark Rank (KITTI, 2016). The novelties of the visual navigation part are listed as,

- The ArUco Markers were implemented for underwater navigation applications;
- An IVO with Monocular camera (IVO-M) was investigated. Compared with conventional methods, the cost of the components used by the method is relative low. Compared with other inertial-visual SLAMs, it recovers the 3D points using a monocular camera and a ping sonar. Furthermore, it benefits from a combination of the FAST algorithm and the Optical Flow algorithm and a linear estimation algorithm. Because of that, it can navigate UUVs in an underwater sparse-feature environment with 3% to 4% drift error;
- An IVO with Stereo camera (IVO-S) was investigated. Similar to the IVO-M, it is a low-cost navigation solution and benefits from implementation of the FAST algorithm and Optical Flow algorithm, and the linear estimation algorithm. Unlike the IVO-M, it reconstructs 3D points by using a stereo camera, and these points are filtered by sonar depth information. After translation vectors have been derived by the linear estimation algorithm, they are corrected by a fusion algorithm prior to processing through a nonlinear optimal algorithm. Hence, the IVO-S is able to localise UUVs with a 1% to 2% drift error.

1.5. Thesis Outline

The thesis is divided into two main parts: control theories for UUVs and visual navigation system for UUVs, because it covers two independent topics. The structure of thesis is shown as follows.

- **Chapter 2**: The literature Review covers four parts, AI architectures, control theories, navigation methods, and quantitative evaluation methods.
- **Chapter 3**: Introduces the experimental ROV, Videoray Pro 3 and the modifications made to the ROV. Multiple advanced senors are described followed by the geometry transformation matrices. The waterproofing solution and buoyancy balance work are also presented. Finally, the coding work for the sensor communication and data collection is mentioned.
- **Part I**: Control Theories for UUVs.
  - **Chapter 4**: Introduces the dynamic and kinematic modelling of UUVs.
  - **Chapter 5**: Presents the derivation of the four controllers: PID, SMC, FLC, BC.
  - **Chapter 6**: The simulations and physical experiments, under different conditions, are presented to evaluate the four controllers. The performances are discussed at the end.
- **Part II**: Visual Navigation Systems for UUVs.
  - **Chapter 7**: Introduces the camera models and camera calibration. The process of underwater calibration is presented as well.
  - **Chapter 8**: The ArUco underwater navigation method is derived. The results are presented and discussed.
  - **Chapter 9**: Derivation of the Integrated Visual Odometry with Monocular Camera (IVO-M). This is a novel visual navigation system for UUVs, and can work in a sparse-feature environment.
  - **Chapter 10**: Derivation of the Integrated Visual Odometry with Stereo Camera (IVO-S). Similar to the IVO-M, it is also a novel visual navigation system for UUVs, but applies a stereo camera to build 3D positions of feature points. Hence, it has higher accuracy than the IVO-M.
  - **Chapter 11**: The experiments are conducted to verify the performances of the two IVOs. The discussion of these results is presented.
- **Chapter 12**: The main conclusion covering the control theories, and navigation methodologies is presented. The future work is also discussed.

- **Appendix**
  - Parameters and Hydrodynamic coefficients of VideoRay Pro 3.
  - Orientation Representation.

Chapter 2. Literature Review

In this chapter, recent work on Artificial Intelligence (AI) architectures, control theories and visual navigation systems are introduced in detail. In the AI section, several classes of AI architectures used for UUVs are presented. In the control sections, development of control theories and modern controllers for UUVs are summarised. In the navigation section, the generic navigation methods will be presented first; then the conventional navigation approaches will be discussed; and finally, the technologies of visual odoemtries or SLAM (Simultaneous Localisation And Mapping) will be investigated. Following that, recent parallel work on underwater visual navigation systems will be illustrated. Finally, the quantitative evaluation methods used in the Visual SLAM (V-SLAM) or odoemtry methods will be summarised.

2.1. The Architectures of AI

Recently, some AI methods have been implemented on AUVs (Carreras et al., 2000) (McGann et al., 2008) (McGann et al., 2007). Generally, there are four classes of AI architectures (Wooldridge, 2001):

The first one is logical-based architecture, in which decision making is realised through logical deductions. It is recognised as a traditional AI method, which approaches building artificially intelligent systems. In this theory, intelligent behaviour can be produced if a symbolic description of the surrounding environments, desired actions, and syntax operation are given. In fact, these symbolic representations have other names: syntactic manipulation, logical deduction, or logical formulae.

The second is reactive architecture, in which the decision-making function is developed as a form of actions corresponding directly to a situation. The reactive method is an alternative to the logical architecture. This method is called “reactive” because the systems are often conceived as simple methods reacting to an environment, without reasoning ability. The decision-making in this method is done by using a set of task accomplishing behaviours; each behaviour may work as a separated action, when perceptual input is received, the system will execute an action.

The third one is belief-desire-intention architecture, in which decision making is produced by the operation of data arrangements including the beliefs, intentions, and desires of the robot. This method is developed from practical reasoning which involves two important processes:

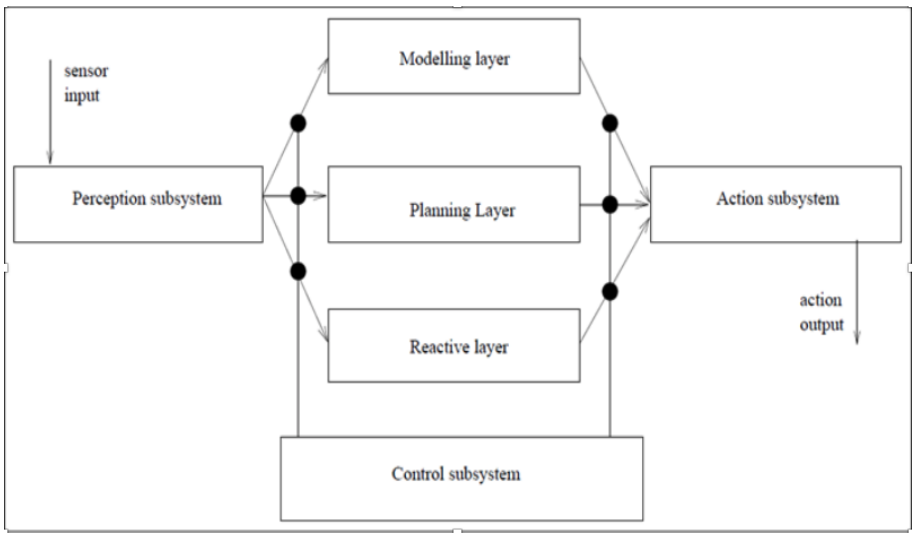


Figure 2.1 Layered Architecture with Planning Layer (Wooldridge, 2001)

deciding what goals the AUV wants to achieve, and how it is going to achieve these goals. The former process is known as deliberation, and the latter as means-ends reasoning. The environment conditions will be abstracted into a set of beliefs, which are filtered to become desires. Then the intentions selected from desires play crucial roles in the actions which are executed based on intentions.

The final one is the layered architecture in which decision making depends upon various software layers, each of the layers has explicit reasoning ability about the environment. Therefore this control structure is a multiple-layers system and the main purpose of this system is to render the UUVs with a decision-making ability required to accomplish a specific mission (Carreras et al., 2006). In this case, the vehicle becomes a physical agent because it can sense the environmental information through external sensors and make decisions. In other words, the reactions are determined by the reasoning mechanism according to the environmental information and specific missions. To be precise, the architecture makes its components present an explicit function and interface (Carreras et al., 2006). Crucially, the whole process has to be conducted in real time. An example is shown in Figure 2.1.

Researchers initially tried to apply reactive control methods to construct AUV control architectures. More specially, waypoint-based pre-programmed guidance systems were designed; potential behaviours or reactions coded, and the method of behaviour selection implemented by the computational stack (Bellingham and Leonard, 1994) (Rajan et al., 2013). The conditionals and quantitative approaches were used to swap these behaviours (Bellingham and Leonard, 1994) (Rajan et al., 2013) (Zhang et al., 2010). However, such methods have drawbacks in terms of systematicity and scalability (Rajan et al., 2013). Furthermore, the reactive methods struggle to deal with balance issues or off-nominal problems. Under complex missions, the control system is required to process information from the mission managing level, underwater localisation system, and fundamental controllers (Rajan et al., 2013).

Therefore some researchers tried to achieve AI for AUVs by building behaviour-based architectures which belong to the belief-desire-intention architecture described previously (McGann et al., 2007) (McGann et al., 2008). In 1986, a new architecture based on the behaviour-based subsumption architecture (Brooks, 1986) was constructed for an AUV. The behaviour in such an architecture is a class of control command at specific conditions. It includes communication with the GPS, diving to the desired depth, and tracking designed waypoints. A plan may be made by an operator by arranging behaviours based on setting up the beginning and end time points and inputting relative duration. Because of that, missions are defined by serial behaviours including time points and related durations. Because of the introduction of the duration, the method can have a restricted flexibility. However, this approach may lose effectiveness when it encounters many uncertain factors (McGann et al., 2008).

Due to these flaws, researchers started investigating the hierarchical deliberative architecture which is one of various layered architectures, based on planning and deductive reasoning ability. Hence, the method is able to make predictions according to environmental information. There are three layers constituting the planner. The mission planning is processed at the first layer and the fundamental controller for actuators is at the low layer. More precisely, a set of pre-programmed plans is selected by the planner to create sequential events through requirements of specific work. Subsequently, the defined plan is executed by the low layer. If conditions are contradicted, a re-planning algorithm is invoked (Carreras et al., 2006).

Typical planners were developed and expected to prove mathematical theorems. In 1971, the Stanford Research Institute Problem Solver (STRIPS) was developed (Fikes and Nilsson, 1971). States were represented by sets of symbolic syntax, and rules and a logical engine assigned true or false values for the syntax. Although the method made many contributions in the planning area, it has limitations and could hardly handle situations related to time sequences and control (Frank and Jónsson, 2003). Many AI-based robotics programs are supported by NASA (National Aeronautics and Space Administration) (Rajan et al., 2000) (Bresina et al., 2005) and they have developed many relevant projects: IPEM (Ambros-Ingerson and Steel, 1988), ROGUE (Haigh and Veloso, 1998), the LASS Architecture (Alami et al., 1998), the Remote Agent Experiment (RAX) (Muscettola et al., 1998), and the Autonomous Spacecraft Experiment (ASE) (Chien et al., 1999).

The above projects also were tested in the real world, and the experimental results were used to update algorithms in future projects. The integrating planning and scheduling (HSTS) (Garrido and Barber, 2001) was applied as a make-decision algorithm for space observations led by the US (Henry and Eaton, 1995) (Rajan et al., 2013). Subsequently, RAX (Jónsson et al., 2000) was designed to improve the intelligence level, where HSTS was installed as the on board equipment and managed the space craft successfully in a period of several weeks. The experiences from RAX contributed to the implementation of IDEA (Aschwanden et al., 2006), where the planners managed collectively to solve problems. However it does not consist of a systematic framework which means the interactions cannot be governed formally (Gregory et al., 2002).

All of the above planners are designed for spacecraft rather than AUVs. T-REX (Teleo-Reactive Executive) was the first AI control system based on planning for AUVs, developed from IDEA. In this method, the perception planning, invoking upcoming behaviour, is conducted in control loops (Rajan et al., 2013). More precisely, the method is a variant of goal-oriented methods with planning ability integrated (Muscettola et al., 1998) (McGann et al., 2008). It was designed to adopt the partitioned construct, taking valuable experiences learned from IDEA, so that the scalability is improved and robustness increased dramatically in the new agent control structure.

The T-REX, based on layer architecture, has been tested on an AUV in the real world, and it had a satisfactory performance. In detail, the structure of a T-REX Agent is shown in Figure 2.2. As mentioned before, it is implemented by multiple concurrent control loops. Every Teleo-Reactor (called reactor) has its own control loop containing specific rules. The arrows indicate the communication signals among reactors: thin arrows show observation values; thick arrows means sending desired value or commands to subsystems. A constraint-based temporal planning approach is utilised to accomplish a declarative model-based function (McGann et al., 2008). However, there are also flaws in this method, it lacks learning and fault-tolerance abilities. It cannot execute the goal selection process and integrate resource constrains for AUVs.

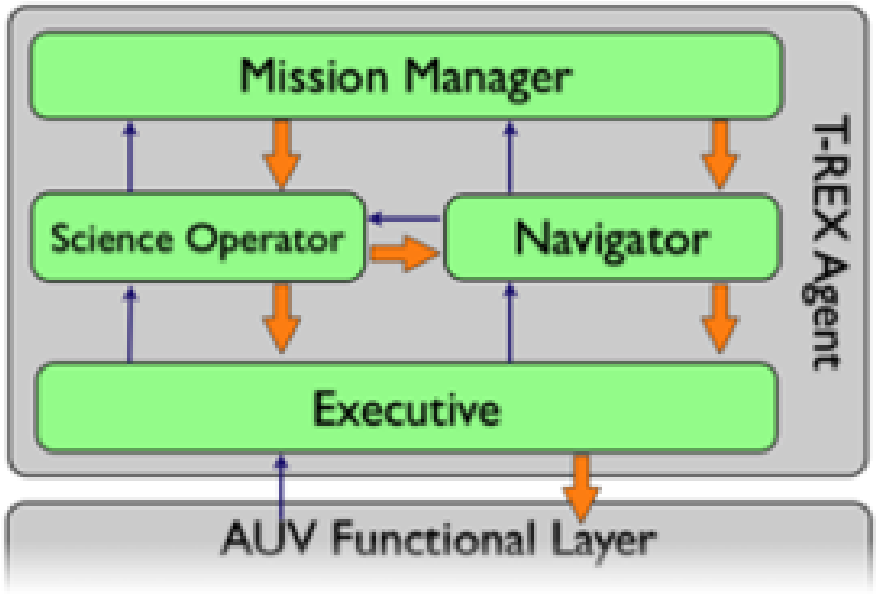


Figure 2.2 T-REX Agent (Rajan et al., 2013)

According to the reported AI architectures, the navigation methods which can sense environment, and the functional controllers which links the AI layer to the actuator, are essential components to build an AI control system. The work about navigation and control will be illustrated next.

2.2. Control Theories

Control theories started with James Watt in the eighteeth century, who developed a centrifugal governor to control the speed of a steam engine. Minorsky, Hazen, and Nyquist made great contributions to the development of control theories. Minorsky, in 1922, developed a proof of the stability conditions for functional controllers used to steer ships. Nyquist, in 1932, invented a simple method of identifying if closed-loop systems are stable. Hazen, is well-known for developing the concept of servo mechanisms in position control fields, and reported a relay servo mechanism which can be stable under a variable input (Ogata, 2002).

In the 1940s, the frequency analysis methods, such as Bode diagram methods, were developed to aid engineers designing specific linear closed loop controllers. In the early 1950s, Evans developed the famous root-locus method. The methods of frequency-response and root-locus, are essential tools for the design of linear closed loop controllers to satisfy specific requirements (Kurfess and Nagurka, 1991). However, these solutions cannot design an optimal system. After the 1950s, researchers in control theories turned to the challenge of finding an optimal system (Bennett, 1993).

Meanwhile, modern plants have become more and more complex as have the mathematical models describing them. Classical control theory is generally utilised to cope with single-input, single output systems. Hence, modern control theory started applying time-domain methods to solve complex control issues (Ogata, 2002). In this method, the state variables are used to describe the performance of the plant.

After the 1960s, adaptive and learning controllers such as Fuzzy Logical Control, were developed to tune stochastic or deterministic systems and then in the 1980s, robust controllers such as $H\infty$ and Sliding Mode Control were investigated (Edwards and Spurgeon, 1998) (Burl, 1998).

Recently, small-size computers with great computing power have been integrated into control systems. In this way, the controller can be programmed in a single board computer directly. On the other hand, modern control theory has been extended to other topics, such as biomedical, biological, and economic research in the 2000s (Banks, 2013).

2.2.1. Control Algorithms for UUVs

There are numerous reports about fundamental controllers for underwater vehicles, but the majority of them are not evaluated by real physical experiments (Fossen and Sagatun, 1991) (Kaminer et al., 1991) (Feng and Allen, 2004) (Yu et al., 2017) (Fjellstad and Fossen, 1994). Fossen and Sagatun (1991) proposed two adaptive controllers based on Barbalat's redundancy Lyapunov-like lemma in order to compensate for nonlinear factors. In this work, a hybrid controller was proposed, which used a switching mechanism and a modelling selection approach.

However, only simulation results were presented. Using an adaptive Sliding Mode Controller (SMC), Cristi et al. (1990) solved the problem of controlling an AUV in diving manoeuvres. The SMCs aim was to eliminate dynamic uncertainties. The simulation results showed that the SMC had satisfactory performance when uncertainties were high. In order to eliminate the chattering induced by SMC theory, Zhou et al. (2011) constructed a Sliding Mode Controller using the Hurwitz Matrix. The approach was validated by simulations. In the results, the proposed methods reduced the chattering of SMC successfully. SMCs have also been applied for designing fault-tolerant controllers by Corradini (Corradini et al., 2011). In this work, an approach developed from SMC theory is not only used to control the ROV, but also used to isolate faults. In the simulation results, the ROV followed reference values through application of the proposed controller, after the occurrence of a fault. However, these controllers were not compared with any other methodologies and no experimental results were presented.

Kaminer et al. (1991) applied H infinity synthesis theory to derive an AUVs control strategy . Initially, the dynamic mode was linearised around the equilibrium point. Then, the controller was derived based on H infinity theory. However, the controller was only evaluated by simulations. A dynamic mode linearised controller, which is a reduced order H infinity controller, was proposed by Feng and Allen (2004). The reduced order problem was equivalent to a rank minimisation problem restricted by a linear matrix inequality. The simulations showed that the reduced order H infinity controller had a smaller error offset than the full order H infinity controller.

As an alternative to model-based controllers, Yu et al. (2017) derived a fuzzy logical controller for AUVs to solve the problem of trajectory tracking. They compared a two-input fuzzy controller with a simplified one-input fuzzy controller. The simulation results showed robust path following performance under unknown environmental disturbances.

Instead of designing advanced controllers, Fjellstad and Fossen (1994) used a unit quaternion for attitude representation and an adaptive controller . In the simulation that was presented, tracking by orientations described by a quaternion and tracking by Euler angles are compared. Minowa and Toda (2018) investigated a high-gain observer to solve control problems of tethered AUVs. In the work, the underwater vehicles were assumed to suffer from high degree nonlinear factors of towing cables. A controller based on this observation and a controller with a Kalman Filter were compared in simulations. The results demonstrated that the proposed controller, with a high-gain observer, had a robust performance.

There are some studies where simulation and practical experiments are conducted. Zhang, Huang, Wan, Li, Cao, Su et al. (2018) designed an adaptive nonlinear second order sliding mode controller in order to reduce chattering motion caused by sliding surfaces . In lake experiments, the proposed method was compared with other SMCs. The results showed that the proposed SMC provided a faster response. In (Martin and Whitcomb, 2018), a PD controller, a linearised model-based controller and a nonlinear model-based controller with decoupled dynamic systems are investigated. Both of the model-based controllers were based on Lyapunov functions. All

three controllers were used on an ROV, and it was claimed that the tracking error of model-based controllers is significantly lower than that of PD controllers, and the nonlinear control tracking error is slightly lower than that of the linearised controller. It should be noted that the practical experiments were carried out without any environmental disturbances. Dukan et al. (2011) proposed a dynamic positioning system for an ROV . In this system, a PID controller was implemented along with an observer based on an extended Kalman filter. The observer and Doppler velocity log provided navigation information for the ROV. The experiments were carried out in the open sea so the external disturbances could not be quantified. The software was developed for operating the Minerva (the ROV name) based on LabView.

Logan (1994) presented a comparison between SMC and H-Infinity controllers for depth and heading control of an AUV. In the work, steering and diving were decoupled into independent subsystems. It was claimed that the strength of the H-Infinity control is in dealing with unmodelled dynamics whilst the merit of SMCs was to build a sliding space for limiting nonlinear factors. However, only simulation results were provided and there was no experimental verification of the results. Innocenti and Campa (1999) used two different controllers to solve a trajectory following problem and compared the results. In the work, structured uncertainties induced by estimation of hydrodynamic coefficients, linearisation of nonlinear dynamic modes and uncertain disturbances, were considered in the control derivation. One controller was designed based on an SMC whilst the other was created based on Linear Matrix Inequalities (LMI) synthesis techniques. According to the comparative simulation results, the LMI region can be adjusted to contain the closed loop poles so that the controller is more robust. In contrast, the SM controller had chattering, even though it could successfully eliminate uncertainties.

According to above reports, most of the developed controllers were only verified in simulations. Studies of comparison among controllers for UUVs belonging to different types (linear, nonlinear, and artificial intelligence algorithms) are seldom reported.

2.3. Navigation Systems

As mentioned in chapter 1, the navigation of UUVs is a challenging problem because of the rapid attenuation of electromagnetic waves in underwater environments. In this section, research about navigation systems will be described in detail.

2.3.1. Generic Navigation Methods

Navigation technology is an approach for manually or automatically identifying position and speed of a vehicle. There are also some navigation systems which are able to output the pose, angular velocity, and acceleration. The installation of a navigation system is flexible. It may be installed on the vehicle such as INS (Inertial Navigation System), or set up using fixed structures

such as acoustic transponders. The term "navigation solution" indicates the output of a navigation system or technology. In order to derive a navigation solution, an algorithm will process the data from navigation sensors, such as gyroscopes, IMUs, and acoustic transponders.

The navigation solution is intended to build the coordinates of a target (such as underwater vehicles, or mobile cars) relative to reference coordinates (such as global, earth or seabed). The vectors of the navigation solution in reference coordinates can be mapped into the third coordinate set using a transformation matrix. The attitude in the navigation solution can be presented as Euler angles, a transformation matrix, a rotation vector, and a quaternion.

In contrast to navigation systems, tracking and surveillance systems derive the pose and trajectory of the target depending on external equipment, rather than the on-board devices. However the navigation solution can be supplied by a tracking system. In the same way, the navigation system can also offer tracking funcionality by presenting the navigation solution. The positioning system estimates the position of the target (such as vehicles, aircraft), rather than velocity or pose. However, because many modern devices can estimate the position at high frequency, the velocity can be obtained from the changing position.

A state-of-art localisation algorithm is a basic requirement for any navigation algorithm. Assuming a robot with state x_t at time t, according to Thrun et al. (2005), the belief of the state is a probabilistic model of x_t given control input $u_{1:t}$ and observation values $z_{1:t}$, given by

$$bel(x_t) = p(x_t|u_{1:t}, z_{1:t}) \tag{2.1}$$

The recursive state may be presented by a linear or nonlinear dynamic equation, the ε_t term is the standard variance of white noise.

$$x_t = f(x_{t-1}, u_t, \varepsilon_t) \tag{2.2}$$

The observation model can be written as,

$$z_t = h(x_t, \delta_t). \tag{2.3}$$

where δ_t is white noise from the observation equipment. Based on Bayes' law, the prior of the recursive state is,

$$\overline{bel}(x_t) = \sum_{x_{t-1}} p(x_t|x_{t-1}, u_t) bel(x_{t-1}) \tag{2.4}$$

Hence, the posterior of state at x_t is,

$$bel(x_t) = \eta\, p(z_t|x_t)\overline{bel}(x_t) \tag{2.5}$$

where η is a constant parameter. The method is also called the Markov assumption. It was assumed that the subsequent state only depends on the previous state and control input.

The navigation methods may be grouped into three categories (Bonnor, 2014).

- Dead Reckoning: It can detect the position variation or detect the velocity and integrate it with respect to time. Hence, by summing the previous velocities, current position can be identified. Because the velocities or travelled trajectories are identified in the coordinates of the target, the attitude relative to the reference is required to obtain the trajectory in the reference frame. A measurement of heading angle is sufficient for two-dimensional motion such as mobile robots. In order to navigate three-dimensional motion, the full attitude (yaw, pitch, roll) should be measured. Actually, the dead reckoning method has been widely applied on UUVs. In (Sabet et al., 2017), a low-cost dead reckoning method is proposed with an extended Kalman filter to provide the navigation solution. In (Yan et al., 2010), Yan reported experimental evaluations of the dead reckoning algorithm for AUVs.

- Inertial Sensors: These sensors include gyroscopes and accelerometers. The travelled distance can be obtained using inertial sensors applying dead reckoning algorithms. Because the dead reckoning is a recursive algorithm, the position error will not be bounded.

- Radio and Satellite Navigation: It was in 1930 that radio was first implemented in the navigation field, and 1961 when the first navigation satellite was launched. Beacons, acoustic transponders, and Doppler Velocity Log (DVL) can be grouped as radio navigation technology. Beacons can receive signals from transmitters to give the bearing. In time of arrival methods, such as Global Navigation Satellite System (GNSS), a clocked signal is sent from the reference frame to the receiver, and the arrival time can be used to calculate the position. That method requires time synchronisation on the transmitter and the receiver.

Inertial sensors, such as IMU or gyroscopes, can estimate angular velocity and accelerations of vehicles. The mechanisms of an IMU and a gyroscope are shown in Figure 2.3 and Figure 2.4 respectively. According to Woodman (2007), inertial sensors suffer from three kinds of noise: constant bias, white noise, and flicker noise. The performance of an inertial sensor depends on its quality. That means the more expensive the sensor, the smaller noise it has. After integrating the error over time twice, the accumulative error increases as a power function with respect to time. The drift error increases dramatically and is unbounded.

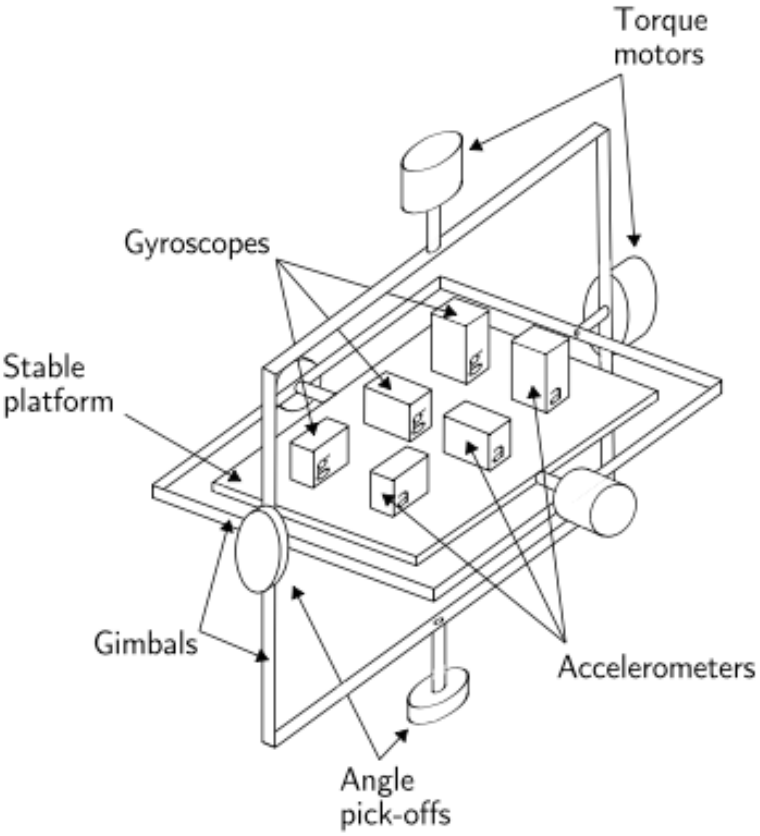


Figure 2.3 IMU Platform (Woodman, 2007)

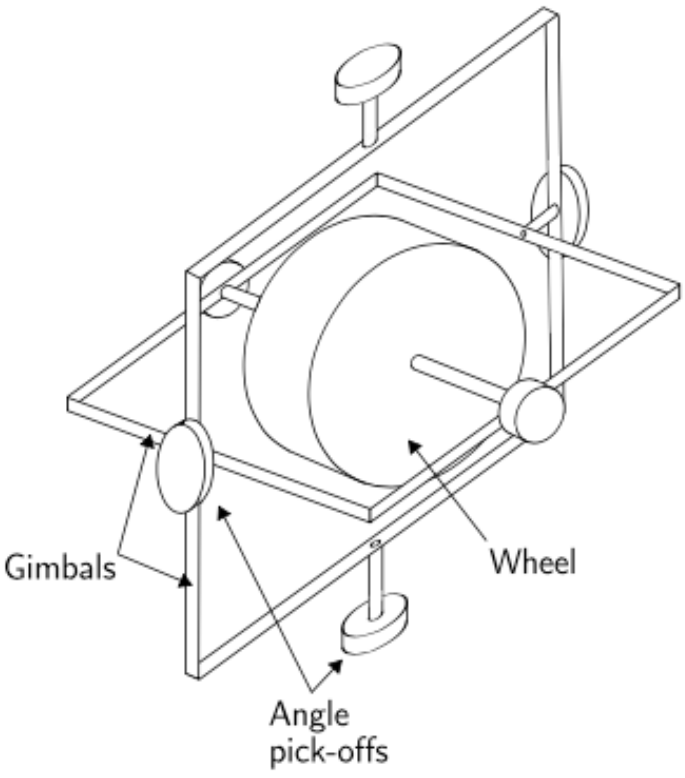


Figure 2.4 The Mechanical Gyroscope (Woodman, 2007)

The noise in accelerometers is of three types. Their descriptions and position errors after double integration are:

- Bias noise: This is a constant error or offset error, ε_i, in the output of accelerometers. The position error with time is:
$$s_a(t) = \varepsilon_i \frac{t^2}{2}; \tag{2.6}$$
- White noise: This follows the Gaussian Distribution, the standard deviation is σ_i. The δ_t is the time between successive samples. The position error with time is:
$$\sigma_a(t) = \sigma_i t^{3/2} \sqrt{\frac{\delta t}{3}} \tag{2.7}$$
- Temperature Impacts: The residual bias is caused by temperature. The errors in angles caused by residual bias increases quadratically with time.

The noise in a gyroscope also has three types, but the functions of position error with time are different.

- Bias noise: The ε_g is a constant bias. The position error is,
$$\theta_g(t) = \varepsilon_g t \tag{2.8}$$
- White noise: This is similar to the noise on the accelerometer, σ_g is the standard deviation. The position error is,
$$\sigma_\theta(t) = \sigma_g \sqrt{\delta_t t}. \tag{2.9}$$
- Temperature Impacts: the residual grows as linear function with time.

2.3.2. Conventional Underwater Navigation Approaches

The conventional navigation approaches for UUVs are (Wynn et al., 2014):

- Inertial methods: The sensors measure acceleration and angular velocity. With integrating signals processing, orientation of the device can be derived. Then, the position is estimated by integrating acceleration twice (Sabet et al., 2017);
- Acoustic Methods: The range is confirmed by acoustic echo time, and the angle can be estimated by phase-differencing methods (Wynn et al., 2014);
- Geophysical Methods: In these methods, the landmarks or feature points are utilised to localise the vehicle. This requires techniques that are able to identify and match feature points (Groves, 2015).

Inertial methods as described previously may suffer from large drift errors. The acoustic transponder navigation methods are widely applied in submarine discovery projects, especially Long Baseline (LBL) and Ultrashort Baseline (USBL) techniques (Wynn et al., 2014). However, the acoustic communications, limited by the acoustic properties and the noise in the underwater environment, are subjected to the following shortcomings:

- Low efficiency: the low frequency of acoustic signals causes low efficiency of data transfer;
- Varying speed: uncertain environment factors, including the temperature and salinity, may vary the velocity of sound;
- Low bandwidth: the volume of information transferred by acoustic signals is restricted that means the acoustic signal cannot send rich information;
- Unreliability: the noise in the underwater environment may affect acoustic communications. The underwater noise is caused by breaking waves, marine animals, seabed reflection and so on.

Sonar sensors, based on acoustic signals, play important roles in underwater navigation and detection. There are various sonars adapted to meet diverse requirements. The single-beam or multiple-beam echosounder can measure the depth and detect obstacles. However, the side scan sonar and imaging sonar provide depth images so that computer vision and pattern recognition techniques can be used to process these depth images. Therefore, navigation methods using sonar are classified into acoustic and geophysical groups (Wynn et al., 2014).

Ultra Short Baseline (USBL) devices can detect the difference of phase among each element on the same transducer, as shown in Figure 2.5 (Vickery, 1998). The comparison of phase can illustrate the attitude of the transceiver relative to the transducer. After utilising Time of Flight (ToF) technologies, the distance from the transceiver to the transducer can be identified. Generally, a pinger, a transponder, or a responder can be used in the USBL system. The pose information obtained by a USBL device is in the coordinates of the transducer, and other information is needed if the pose is to be transferred to the seabed reference.

The drawbacks of USBL methods are (Vickery, 1998):

- Needing detailed calibration to obtain good estimation;
- Requiring additional sensors to get the positions and attitudes relative to the seabed reference;
- Lacking over-determined solutions.

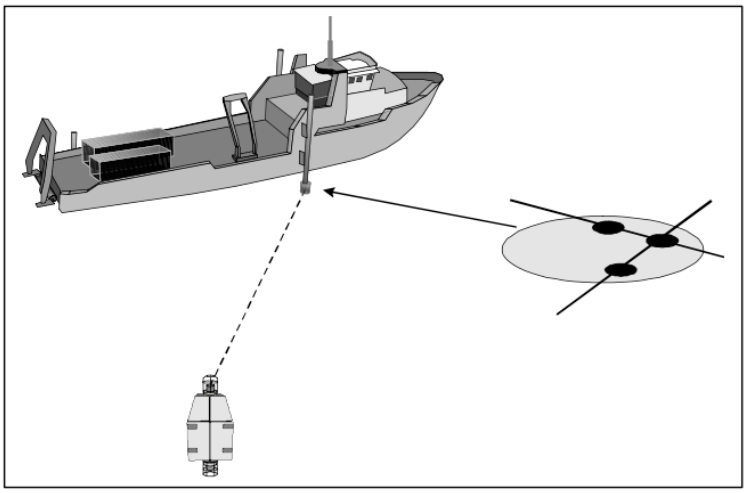


Figure 2.5 Ultra Short Baseline Positioning (Vickery, 1998)

Short Baseline (SBL) algorithms estimate attitudes to a target beacon by using multiple transceivers, that are fixed on the ship hull or floating structure, as shown in Figure 2.6. The attitude is calculated by identifying the different arrival times of an impulse acoustic signal among different transceivers. The bearing of the beacon relative to the transceivers can also be derived by interrogation techniques. Hence, the system consists of a transponder, and multiple transceivers. In this case, the attitude and bearing of the target obtained by the system are relative to the structure on which the transceivers are mounted. Other sensors are required to get the position or attitude relative to the seabed.

The shortcomings of SBL systems are:

- Long baselines are required to get high accuracy;
- A detailed dock calibration is needed;
- An offshore calibration process is required as well;
- Third-party sensors are needed to get the position in the seabed coordinates;
- More than three deployment poles for the transceiver are needed.

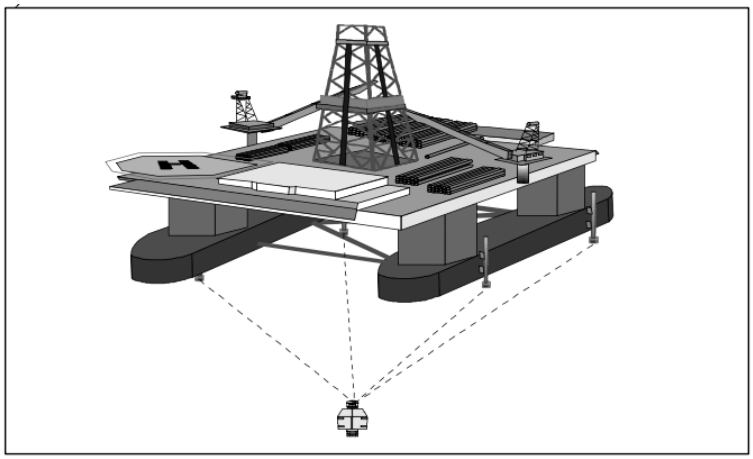


Figure 2.6 Short Baseline Positioning (Vickery, 1998)

Unlike SBL, the Long Baseline (LBL) systems are able to obtain a position of the tracking target relative to the seabed. The array of transponders must be deployed in advance, as shown in Figure 2.7. The localisation is conducted by three or more time of flight measures from the seabed transponders. Because the transponders are set up on the seabed, the position of the tracked target derived from the system is relative to the seabed.

The shortcomings of LBL methods are:

- An expert is required to operate the system due to its complexity;
- High cost because of large arrays;
- Deployment and recovery are required;
- Calibration is needed after each deployment.

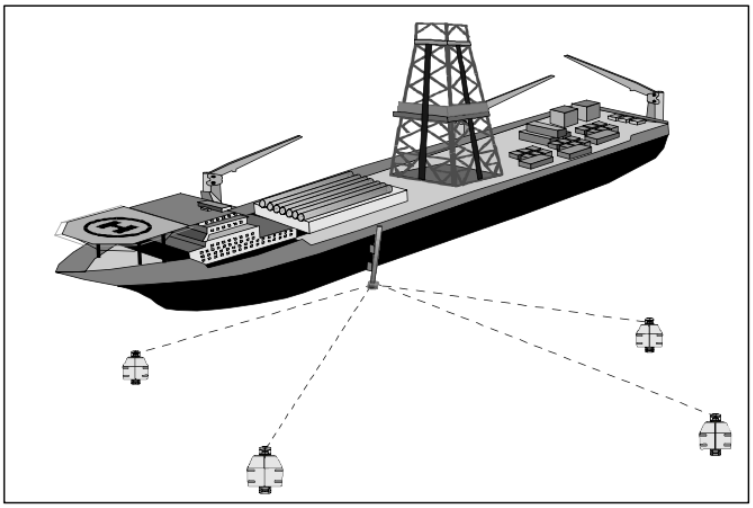


Figure 2.7 Long Baseline Positioning (Vickery, 1998)

The Doppler Velocity Log (DVL) is also an important acoustic navigation device, which is able to measure the velocity relative to the seabed. It has multiple beacons oriented in different directions. Applying the dead reckoning algorithm, the trajectory of the vehicle can be estimated, but correspondingly an unbounded error also occurs. The drift in the best DVLs can be about 0.1% of the whole trajectory (Fallon et al., 2011), and 2%-5% drift in whole trajectories can be achieved by ordinary DVLs.

Geophysical navigation methods localise vehicles relying on external environmental features for localisation. Thus, closed loop techniques can be utilised in these methods to bound position error. This technique can record feature points and landmarks. When these features are detected again, the navigation will correct the current and previous pose to bound the drift error. The classes of geophysical navigation methods are (Wynn et al., 2014):

- Magnetic: The navigation method is expected to detect the magnetic field and identify the features. A US research team has applied these methods on Navy vessels (Armstrong et al., 2010) (Armstrong et al., 2009).
- Optical: Cameras are implemented to capture images of the seabed. The visual information is used to localise the cameras.
- Sonar: The acoustic signals are utilised to gather geophysical features or landmarks. There is an algorithm to identify these features and estimate the trajectory of the camera.

Even though imaging sonars are able to provide the depth images, the low frequency and low resolution restrict their applications in underwater navigation (Bonin-Font et al., 2015). In Palomeras et al. (2018)'s report, rich navigation information is provided by optical sensors rather than acoustic systems.

The main difficulty of these traditional navigation approaches is the high cost. For acoustic methods, an array of acoustic transponders should be set up and calibrated (Kim and Eustice, 2013). The cost of the modest DVL is about 5K USD (WaterLink, 2020) and cost of the ordinary ones are about 20K USD (Wynn et al., 2013). Because of the dramatic unbounded error, inertial sensors always cooperate with other navigation methods or work as a complementary method.

Visual Odometry (VO) and Visual Simultaneous Localisation And Mapping (V-SLAM) have been successfully applied in mobile robotics and aerial robotics (Scaramuzza and Fraundorfer, 2011). One well-known visual odometry application has been implemented on NASA's projects for Mars exploration (Maimone et al., 2007). In 2005, Nister provided a visual odometry for ground Vehicle applications (Nistér et al., 2006). Scaramuzza developed a special visual odometry with omnidirectional camera to guide ground cars (Scaramuzza and Siegwart, 2008). Forster reported a fast monocular visual odometry for micro drones (Forster et al., 2014).

2.3.3. Visual Odometry

VO methods can predict the pose of a camera by implementing computer vision technologies. However, they usually lack the ability to bound drift errors.

The optical sensors for VO methods are utilised to capture environmental images and provide environment information to the visual navigation systems. Generally, there are four kinds of optical sensors. These are as follows:

- Monocular camera: This is a single camera which cannot output depth images;
- Stereo camera: This has two cameras with a fixed baseline; the depth images can be generated by computer vision techniques;
- 3D-Range camera: Both the depth and colour information can be provided by this camera. Unlike the stereo camera, the camera measures the depth using active methods such as Time-of-Flight (ToF) or structured light (Santoso et al., 2016);
- Infrared camera: This is usually utilised at night as it is independent of visible light.

The 2D motion of pixels can be estimated by the Optical Flow (OF) algorithm. It can track motions of pixels in a series of images by their constant brightness (Garratt and Chahl, 2007), based on the assumption that the brightness in a series of images remains constant.

The OF algorithm has been widely applied for drones. Assuming the altitude is known, the velocity relative to terrain is approximately derived through: $H = V/Q$, where H is the height of the drone, V is the velocity, and Q is a constant factor (Garratt and Chahl, 2007). In Garratt and Chahl (2008), an algorithm for integrating information from other sensors was utilised to estimate the velocity of a drone. Li et al. (2015) developed a high accuracy but low computational cost OF algorithm, which is popular in computer vision field. However, this algorithm has a big drawback that it assumes constant brightness. In practical conditions, the intensity of the light can be varied by environmental factors.

While the OF methods can be used to track the feature points in a series of images, it is hard to recover the camera motion in absolute scale. VOs and V-SLAM algorithms can be grouped into two different classes: one is the Feature-Base Method, and other is the Direct Method.

In feature-based methods, the feature points are extracted from images caught by the camera, and tracked in successive frames by utilising matching algorithms. Based on these corresponding feature points, the fundamental matrix is constructed through epipolar geometry. Then, camera motion in relative scale may be recovered by the fundamental matrix. Finally, an optimal iterative algorithm is adopted to minimise the re-projection error functions. The key of such a method is the matching algorithm, which is used to find corresponding feature points in

series of images. Even though the feature point matching algorithm can be considered as a robust method, there are still lots of matching errors (Forster et al., 2014).

Direct methods can estimate motions directly from intensity values in the image (Irani and Anandan, 1999), rather than extracting feature points. That requires the obvious gradient on the local intensity so that the iterative algorithm can go forward in the correct direction. The dense direction methods can exploit all pixels in the image. Through the evaluations, direct methods have better performance than feature-based methods in feature-sparse environments or in the case of high-speed motion (Lovegrove et al., 2011) (Newcombe et al., 2011). Comparing the re-projection error, the computational cost of the photometric error is higher, because it integrates the intensity of pixels. However as direct methods manipulate the intensities of a series of images directly, the image pre-process step can be skipped.

2.3.4. Visual SLAM

V-SLAM methods are extended from VO methods with loop closure techniques implemented, which are able to bound drift errors. The popular open source V-SLAM algorithms are summarised as follows.

- PTAM: This is a feature based method. In order to reduce the computational complexity, a set of keyframes is selected from numerous frames (Klein and Murray, 2007). Furthermore, the concurrency and multiple thread techniques are implemented to accelerate the process speed. Hence, it can implement real-time localisation efficiently.
- LSD-SLAM: This was developed combining the direct method and the bundle adjustment method. Since processes of extracting and matching the descriptors of features are omitted, this algorithm can run in real-time. Such an algorithm is able to construct a semi-dense map only using CPUs (Central Processing Unit). However, the descriptors of feature points are still required for loop detection, and the accuracy of the method is lower than that of ORB-SLAM (Mur-Artal et al., 2015);
- SVO (Fast Semi-Direct Monocular Visual Odometry): This is also a direct method and operates on pixel intensities. A probabilistic model is built to offer a robust way to recover the depths of feature points (Forster et al., 2014). However, in the bootstrap, the method requires the lens of the camera to be parallel to the ground.
- ORB-SLAM: This is a feature-based method developed from PTAM. It extracts the ORB descriptor (Rublee et al., 2011) for each of the feature points, and tracks these descriptors (Mur-Artal et al., 2015).
- DSO (Direct Sparse Odometry): This was developed from LSD-SLAM. A fully direct probabilistic model and nonlinear joint optimisation algorithms are implemented to improve the accuracy and to be consistent (Engel et al., 2018).

PTAM was considered to be the first V-SLAM applying bundle adjustment techniques in real-time applications. ORB-SLAM, based on PTAM, was improved by using loop-closure techniques to reduce the drift error significantly. The SVO method is only applied on downward facing cameras. Because DSO and SVO are implemented as direct methods, they are limited by drawbacks of direct methods, such as the requirement of high-rate frame and fixed exposure time. Furthermore, DSO requires photometric calibration. The effect of such a system depends heavily on the accuracy of intrinsic parameters (Rublee et al., 2011) .

While the pure visual SLAMs are accurate, low cost, and have bounded position error, the performances of these SLAM methods are dependent on the density of feature points. Therefore, these methods require the use of global shuttering cameras with long focal-length, wide angle lenses to capture more reliable feature points. Pure SLAMs usually require at least 50 feature points to recover the cameras' motion. In the SVO method, at least 100 feature points are required in the initialisation section.

In order to catch larger views, fisheye lenses and omnidirectional cameras have been applied. The fisheye lenses are used to offer a hemispherical view. Some fisheye lenses have more than 180 degree view (Schwalbe, 2005).

The omnidirectional camera is a camera having a 360-degree field of view. Similar to fisheye lenses, its projection systems cannot be presented by applying the typical perspective camera model due to high-level distortion (Scaramuzza and Ikeuchi, 2014).

There are several kinds of omnidirectional cameras (Scaramuzza and Ikeuchi, 2014) which include:

- Dioptric camera: This benefits from a composition of shaped lenses, so that some of them are able to offer a view which is larger than 180 degrees.
- Catadioptric camera: This has a special combination of typical lenses and shaped mirrors. When elliptical or parabolic mirrors are implemented, a 360-degree view can be offered.
- Polydioptic camera: This camera consists of several cameras with overlapping views.

In addition to the fisheye lenses and the omnidirectional camera, the Visual Inertial Navigation system also can improve the performances of V-SLAMs or VO algorithms with fusing the data from inertial sensors. Due to the different fusing algorithms, there are two subgroups methods: loosely coupled algorithms and tightly coupled algorithms.

In loosely coupled algorithms, poses are estimated by inertial sensors and optical sensors separately. Subsequently, these sources of information are fused to output optimal results (Huang et al., 2014). However, this may cause information loss.

Conversely, tightly coupled approaches generally use information from multiple sensors to output optimal results directly. More specifically, a nonlinear regression algorithm is applied to

minimise errors from multiple sensors. The methods are considered to have better performances than those of loosely coupled algorithms (Paull et al., 2014).

Generally, inertial sensors perform well at high speed but suffer from a big drift at slow speed (Corke, Lobo and Dias, 2007). Conversely, computer vision techniques can track feature point well over slow-speed motion but fail in high-speed motion. Leutenegger et al. (2015) and Qin and Shen (2018) developed inertial-visual odometries (OKVIS and VINE-Mono) for air drone navigation. In their work, a cost function, constructed by summing re-projection error and inertial sensor error, is minimised to solve for camera pose.

2.3.5. Underwater Optical Navigation Systems

In contrast to the common application of visual odometry in ground vehicles and drones, the development of visual odometry in underwater environments has been limited. This is because the illumination and visibility are uncertain factors in the underwater environment, and the seabed may not offer as many feature points as the ground (Paull et al., 2013). Furthermore, the cost of underwater data collection is much higher relative to ground data collection. These limitations have limited the development of applications of visual odometry on underwater vehicles.

Caccia (2007) investigated a horizontal motion estimation based on an optical laser camera. In this method, the depth is measured by laser, and the optical algorithm only estimates the linear motion. Eustice et al. (2008) used a visual odometry system to aid DVL for underwater vehicles. It was termed Visually Augumented Navigation (VAN). The DVL and VO are combined by a variant Extended Kalman Filter (EKF), where the attitude is corrected by visual odometry. In this case, visual odometry is applied to bound errors from inertial methods. Eustice et al. (2006) improved the method by replacing the EKF filter with an information filter. The improved approach was applied to underwater exploration surveying of RMS Titanic, and the results showed an improvement in accuracy. Kim and Eustice (2009) applied the VAN method to inspect ship-hulls for the U.S. Navy. The VAN method was extended by Kaess et al. (2008), and Mahon et al. (2008). In their work, optimal trajectory estimation is considered as a smoothing and mapping problem formulation and it is solved by efficient matrix factorisations. The VAN algorithm with image sonar was also utilised to estimate pose and landmarks using pose graph methods by Hover et al. (2012). Kim and Eustice (2013) made an improvement on iSLAM to estimate the trajectory of a vehicle by fusing information from DVL and a visual odometry.

Corke, Detweiler, Dunbabin, Hamilton, Rus and Vasilescu (2007) made a comparison between visual odometry, and acoustic beacons. In the paper, visual odometry with a stereo camera gave 5% along track error. Hildebrandt and Kirchner (2010) reported an IMU-aided stereo visual odometry to predict the trajectory of a vehicle. However, the algorithm was tested in a synthetic underwater environment rendered by the 3DS Max program. Two open source visual odometries, libviso2 and fovis, were applied to underwater navigation by Wirth et al. (2013).

The tank floor was covered by digital poster, in order to create a feature-rich environment. Feature point matching technology was used to obtain the ground truth. Carrasco et al. (2016) applied graph-SLAM to optimise visual odometry performance. In the experiments, a digital poster was laid on the bottom of the tank, again to offer a dense feature environment. Bellavia et al. (2017) developed a stereo visual odometry system and estimated vehicle pose using trusted data . However the authors applied the KITTI benchmark to evaluate the methods. The KITTI dataset is collected by a ground vehicle, so the results of the evaluation are not suitable for underwater navigation.

Recent parallel work has been done by Ferrera et al. (2019), Rahman et al. (2018*a*) and Rahman et al. (2018*b*). Ferrera et al. (2019) developed a real time Monocular Visual odometry system for the underwater environment. It uses the Optical Flow (OF) algorithm to track feature points and the depth information is obtained through triangulation calculations. Leutenegger et al. (2015) extended the Open Keyframe-based Visual-Inertial SLAM (OKVIS) with underwater profiling sonar. The method fuses multiple information from a stereo camera, a profiling sonar, an IMU and a pressure sensor by using a tightly-coupled nonlinear optimisation. More specifically, Rahman et al. (2018*a*) derived a cost function by summing the re-projection error, the IMU error, and the sonar error. Because of that, camera pose estimation and mapping of underwater structures are processed simultaneously by minimising the cost function. Rahman et al. (2018*b*) improved the method by applying a robust initialisation method, an image enhanced technique, and a loop-closure technique. However, these methods were not tested by quantitative evaluation methods in the underwater environment. This means that the accuracy or drift errors of these methods are not clearly understood in underwater navigation applications.

2.3.6. Quantitative Evaluation Methods For Visual Odometries or SLAMs

The quantitative evaluation for any navigation system is of critical importance. In this section, the quantitative evaluation approaches for the above visual navigation methods are summarised.

In the area of mobile robotics navigation research, Karlsurhe Institute of Technology and Toyota Technology Institute (KITTI) have provided standard datasets with corresponding truth trajectories. Furthermore, the datasets were collected by different optical sensors, inertial sensors and a Velodyne laserscanner. The ground truth is measured by a GPS system. The test platform is presented in Figure 2.8.



Figure 2.8 Data Collecting Car (KITTI, 2016)

The evaluation in this benchmark is processed by computing rotational and translational errors at different distance points. An example of evaluation is that of the V-LOAM method (Zhang and Singh, 2015). It consists of rotational error and translational error. They are presented in Figure 2.9, 2.10 and 2.11.

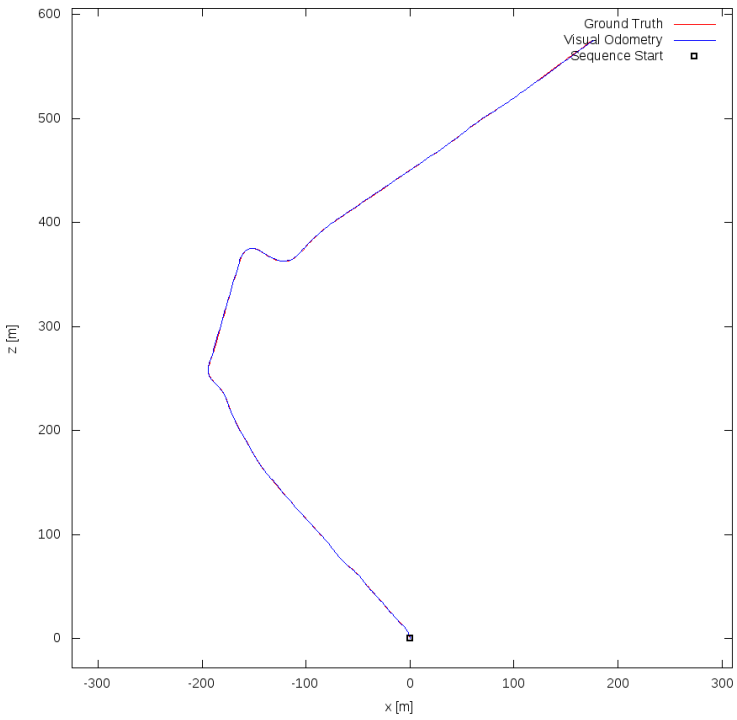


Figure 2.9 Comparison between estimated trajectory and ground truth (Zhang and Singh, 2015)

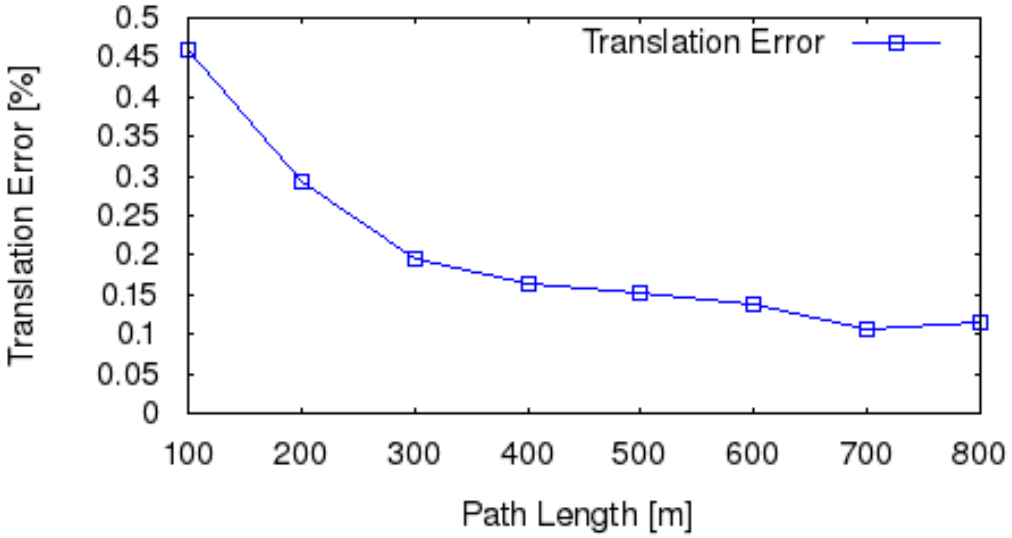


Figure 2.10 Translational Error of V-LOAM (Zhang and Singh, 2015)

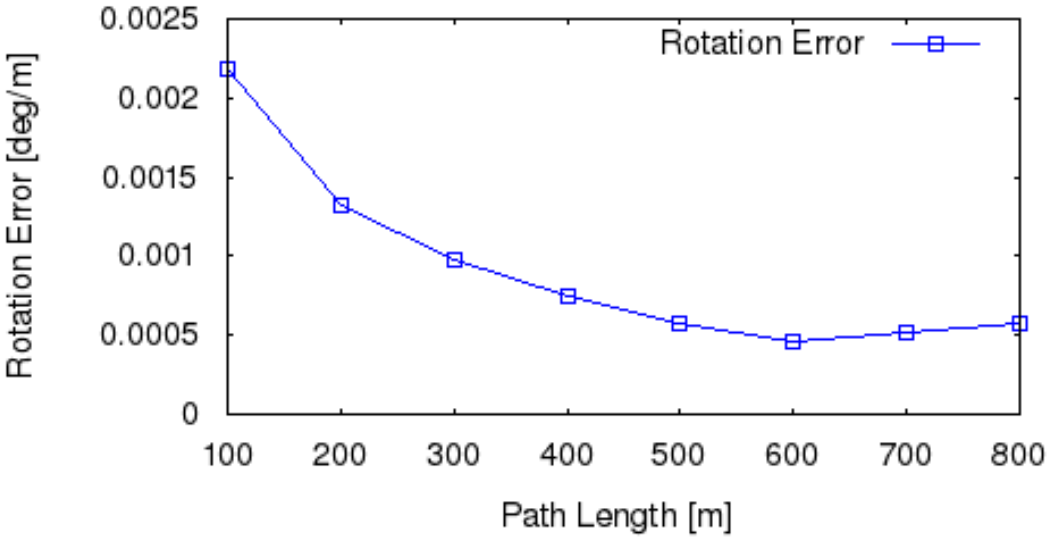


Figure 2.11 Rotational Error of V-LOAM (Zhang and Singh, 2015)

In Engel et al. (2018)'s work, the Direct Sparse Odometry is evaluated by using the following three datasets:

- The TUMmonoVO dataset. This includes 50 image sequences with corresponding calibrated camera parameters. The dataset only provides loop-closure trajectories. The accumulated drift is used to evaluate the performances of visual navigation methods. An alignment error, e_{align}, is used to present the accumulated error.
- The EuRoc MAV dataset. This includes 11 sequences containing stereo images and inertial information. A motion capture system and a laser scanner were used to collect the ground truth trajectory. The tests in this dataset were evaluated in term of the absolute trajectory error, e_{ate}.

- The ICL-NUIM dataset. This contains eight sequences. However, the environments were built by 3D render techniques. Therefore, the ground truth was defined during simulation processes.

The methodology is to analyse the error plot for the number of tracked sequences (the size of window) versus the error value (e_{ate}/e_{align}). An example is given in Figure 2.12.

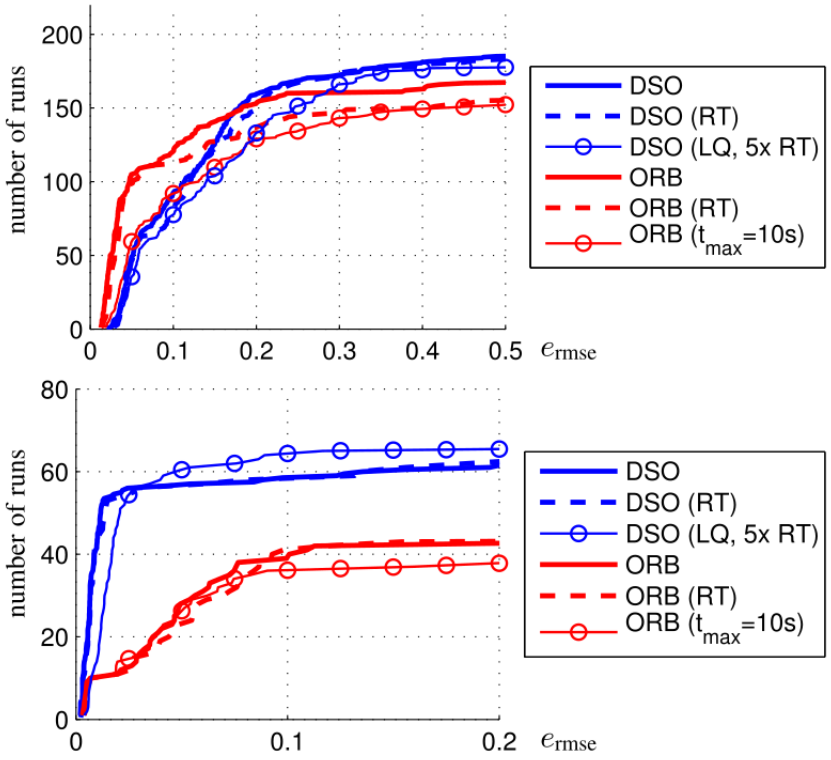


Figure 2.12 DSO (Direct Sparse Odometry) and ORB (Orb SLAM2) Errors Plot (Engel et al., 2018) (RT means real-time, 5x RT means 5 times real-time)

The evaluation of ORB-SLAM2 is followed a standard methodology, and the applied datasets were: EuRoC Dataset, KITTI Dataset, and TUM RGB-D Dataset. The TUM RGB-D Dataset can provide the ground truth trajectories which are believed to be obtained by a Qualysis Motion Capture system. The quantitative evaluation is done by computing the translational errors and the rotational errors.

In underwater navigation, cumulative errors can be estimated by detecting landmarks in underwater environments. In the reports by Ferreira et al. (2012) and Caccia (2007), landmarks were used to evaluate the methods. The position of the landmarks on the seabed is known beforehand. When the landmark appears in the frame, the position of the camera in that frame can be compared with the position of the landmark. This evaluation method using landmarks in some frames can only predict the truth position. It is hard to give the exact truth position of camera, especially, when the landmark is not positioned in the centre of frame.

Some methods were evaluated by image matching algorithms, under the condition of knowing seabed maps. In the reports of Carrasco et al. (2016) and Wirth et al. (2013), the bottom of a

pool was covered with posters representing underwater environments. Image matching technology was used to estimate approximate positions of a camera above the poster. The digital poster on the floor is shown in Figure 2.13

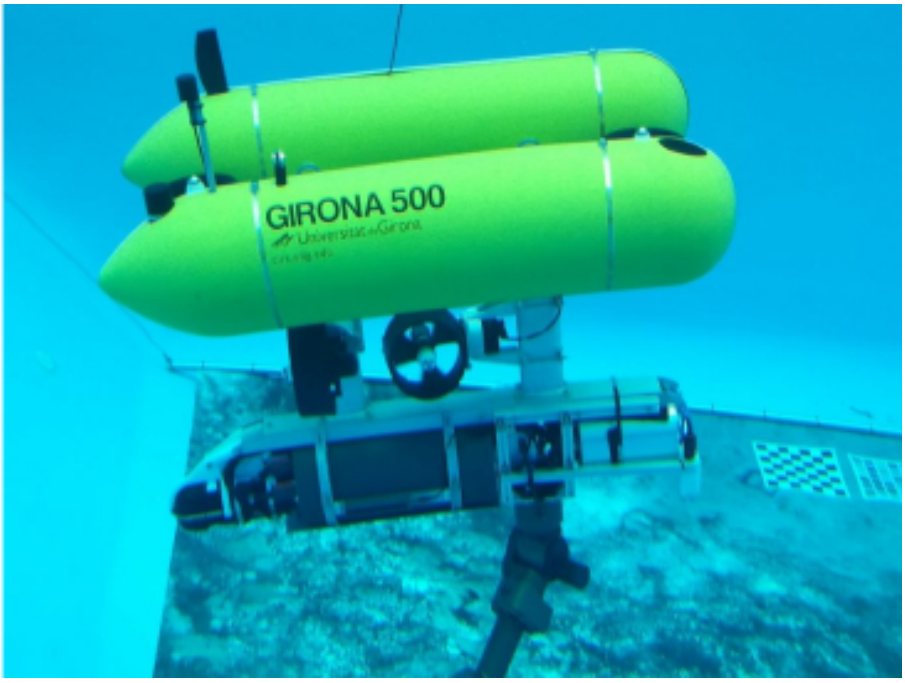


Figure 2.13 The Digital Poster on Floor (Carrasco et al., 2016)

In underwater environments, acoustic navigation systems are ideal tools for measuring reference trajectories. In (Eustice et al., 2008), dead reckoning methods and the proposed navigation method are compared. The ground truth trajectory is obtained by a long-baseline system. In the evaluation method, the estimated trajectories from two methods and the true trajectory are compared with each other. In the work by Corke, Detweiler, Dunbabin, Hamilton, Rus and Vasilescu (2007), two localisation methods are evaluated, one is acoustic localisation and the other is visual odometry. The ground truth is obtained by surface GPS. In the evaluation method, GPS trajectories and estimated trajectories were plotted.

Compared with high-cost acoustic equipment, a simulated underwater environment is a low-cost solution for evaluating underwater visual navigation methods. In the work by Hildebrandt and Kirchner (2010), the proposed method was evaluated on synthetic datasets. The synthetic underwater environment was created by using the rendering program (3ds MAX). The datasets can be collected by simulating a camera travelling in the synthetic environment. A series of images captured in the synthetic underwater environment are shown in Figure 2.14.

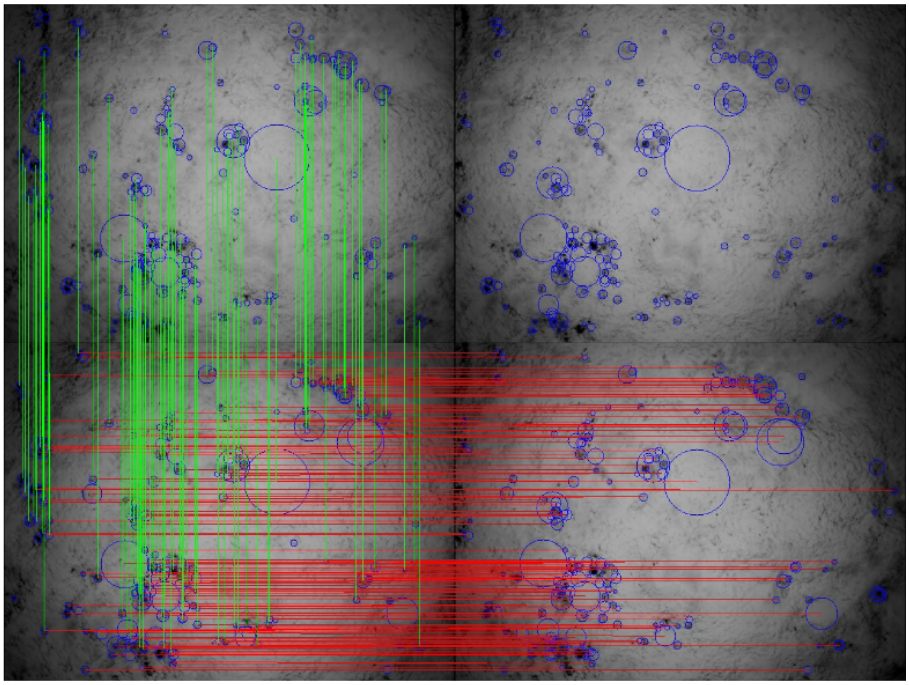


Figure 2.14 Images Captured in Synthetic Underwater Environment (Hildebrandt and Kirchner, 2010)

However, some techniques have not been verified by quantitative evaluation methods. In the paper by Bellavia et al. (2017), a stereo visual odoemtry system is developed for autonomous underwater vehicle localisation tasks. The method is evaluated by using the KITTI benchmark and Tsukuba stereo dataset. However KITTI is a mobile robot dataset, so the effectiveness of such an evaluation is doubtful. The Tsukuba stereo dataset was gathered by the MARTA AUV in the Isreali Sea, developed within the ARROWS project (Allotta et al., 2015). However, the ground truth data is not given. In this evaluation method, only estimated trajectory is provided.

The evaluation methods that have been discussed are summarised in Table 2.1.

Algorithms	Applied Dataset	Ground Truth	Evaluation Methodology
Mobile Robotics			
DSO	KITTI, TUMmonoVO, EuRoc Mav, ICL-NUIM	except TUMmonoVO	TUMmonoVO: accumulated error, others: absolute trajectory error
ORB-SLAM2	KITTI, TUM RGB-D, EuRoc Mav	All	rotational error translational error
Underwater Vehicles			
Caccia	self-collection	No	landmark comparision
Bellavia	KITTI, New Tsukuba	KITTI yes	rotational error translational error
Corke	self-collection	yes with GPS	plot GPS estimated trajectories
Pep	self-collection	seabed poster matching	rotational error translational error
Eustice	self-collection	yes long-baseline system	plotting trajectories
Hildebrandt	synthetic dataset	yes	mean error standard deviation

Table 2.1 Summary of Evaluation Methods

2.4. Summary

In this chapter, research in the fields of AI architectures, fundamental controllers and navigation approaches for UUVs has been discussed. In UUV control, most of the developed systems were verified by simulation methods, and studies comparing controllers from different types (linear, nonlinear, and AI algorithms) are rare. In underwater navigation, typical navigation approaches are high cost. Generic visual navigation methods are designed to work in dense-feature environments. In the next chapter, the experimental platform used in this thesis will be presented.

Chapter 3. The Experiment ROV

In this chapter, the unmanned underwater vehicle and sensors utilised for the experimental test and data collection are introduced. A Videoray Pro 3 ROV was selected as the test platform to verify the performance of different controllers. Sensors were added to the vehicle, including a sonar, an IMU, a gyroscope, an Intel T265 Tracking Camera, and a global shutter camera. They were used to gather underwater images and internal data such as attitudes and accelerations. In order to map the values from different sensors to the same coordinate system, the layout of the sensors and corresponding transformation matrices are also presented in this chapter.

3.1. VideoRay Pro 3

The VideoRay Pro 3 was developed by VideoRay LLC in 2004. As its name suggests, it is the third generation product in the VideoRay ROV series. There are two front lights, three thrusters, and rear and front cameras on the vehicle, as shown in Figure 3.1. The three thrusters comprise two horizontal thrusters used to control the linear velocity and the heading angle, and a vertical thruster used to adjust the depth. The vehicle comes with a portable control panel with an RS232 interface and a video output port (composite video port). Communication between a computer and the vehicle can be implemented through the RS232 port and the composite video port, as shown in Figure 3.2. This means that the customised computer can receive sensor data and real-time images from the vehicle, and send commands to the thrusters of the vehicle.



Figure 3.1 VideoRay Pro 3

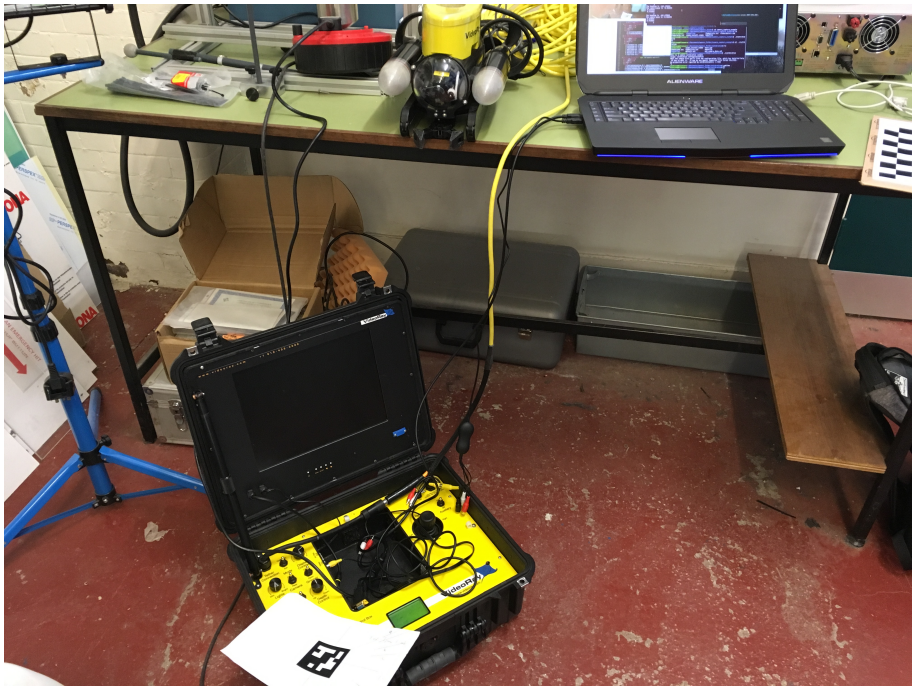


Figure 3.2 The customised laptop connected to the Panel

The rigid-body parameters of the vehicle are provided by the VideoRay company. The hydrodynamic coefficients were measured by Wang and Clark (2006), and the thruster parameters are also identified in this work. The mechanism utilised to measure the hydrodynamic coefficients is illustrated in Figure 3.3.

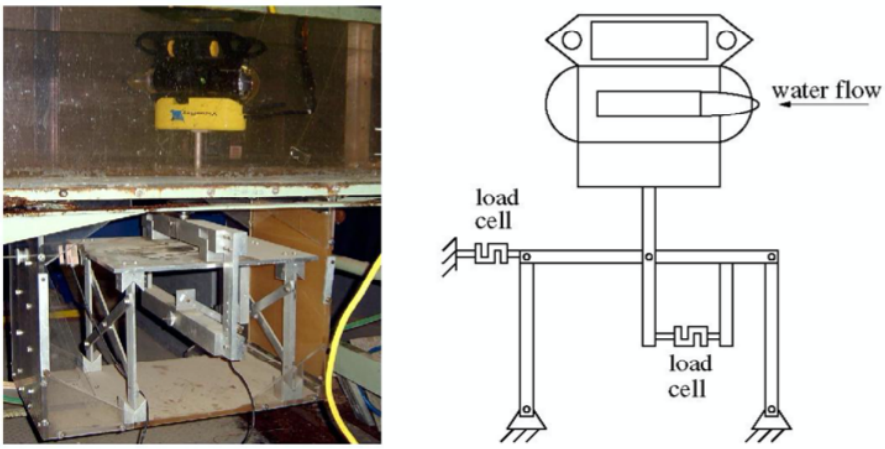


Figure 3.3 Mechanism Measuring the Hydrodynamic Coefficients (Wang and Clark, 2006)

The parameters of VideoRay Pro 3 are presented in Appendix A.

3.2. Modification for the ROV

In order to gather high-quality data, the VideoRay Pro 3 was modified. A waterproof tube with multiple precise sensors inside, has been built and installed on the bottom of the original vehicle. The main sensors involved are an IMU development kit, which is expected to provide acceleration data, an optical sensor (global shutter camera), which is able to capture grayscale images when the vehicle is in motion, a gyroscope, a sonar and an Intel RealSense T265 Tracking Camera. The gyroscope can provide the yaw angle, which is not affected by magnetic fields. The sonar is to detect the distance between the seabed (or tank bottom) and the vehicle. The T265 Tracking Camera was installed on the electronics tray in order to provide comparative results. The T265 consists of a stereo camera, an IMU senor and a processor, which can provide the trajectory of the camera directly using its own V-SLAM algorithm. Hence, the results from the proposed IVO and the T265 could be compared.

The ROV with the additional tube is shown in Figure 3.4. These electrical components are listed in Table 3.1. Because the intension was to design a low-cost underwater navigation system, the costs and precision of these electrical components are listed at Table 3.2. They were mounted on a tray, and the tray was inserted into the water-tight tube.

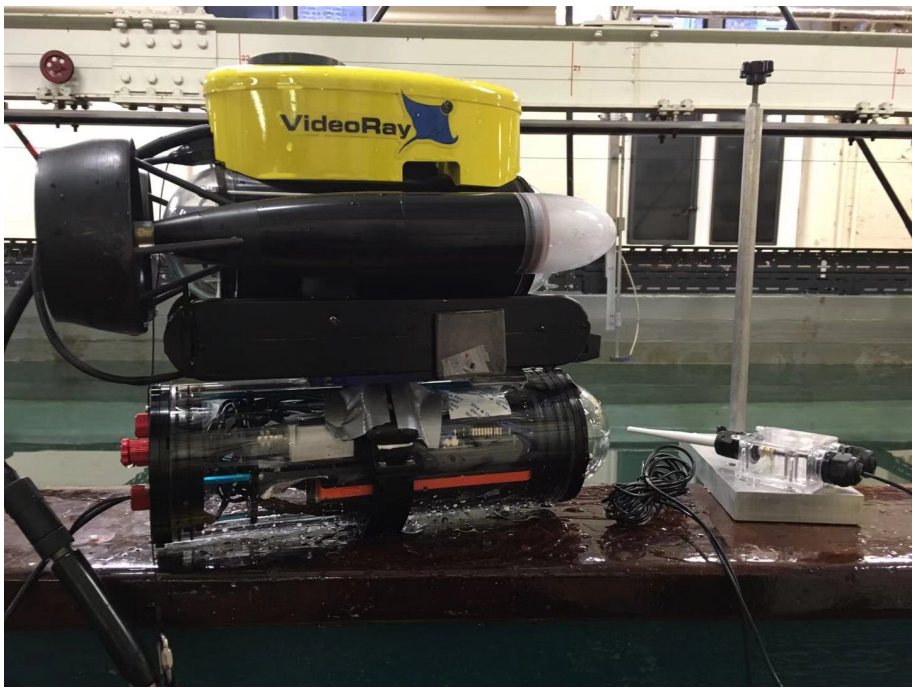


Figure 3.4 The VideoRay Pro 3 with additional tube

Electrical Component	Name	Functions
IMU Sensor	LPMS-ME1 DK	a 3-axis accelerometer, 3-axis gyroscope, and 3-axis magnetomete
Gyroscope Sensor	LPMS-NAV	a high accuracy one-axis gyroscope, and a 3-axis accelerometer
Monocular Camera	mvBlueFOX-MLC 200w	Gathering series images
Stereo Camera	Intel Tracking Camera T265	Stereo camera comparative V-SLAM algorithm
Leak Sensor	SOS Leak Sensor	Detecting leaking water
Mini Router	GL-AR300M16	Building WiFi network
Onboard Computer	LattePanda	Recording raw Data Communication with terminal

Table 3.1 Functions of Electrical Devices

Components	Costs	Precision
mvBlueFOX-MLC 200w	£200	
BlueRobotics Ping Sonar	£100	0.5% of range
LPMS-ME1	£50	$< 2^o$
LPMS-NV2	£100	$< 0.1^o$
Intel RealSense T265	£179	

Table 3.2 The Costs and Precision of Components

The raw data from the sensors are gathered by a mini computer. The onboard computer communicates with the onshore terminal through WiFi. Hence, the terminal can receive the data from the onboard computer and send commands. In this case, complex computations are processed within terminal, instead of the onboard mini computer. The procedure is described in Figure 3.5.

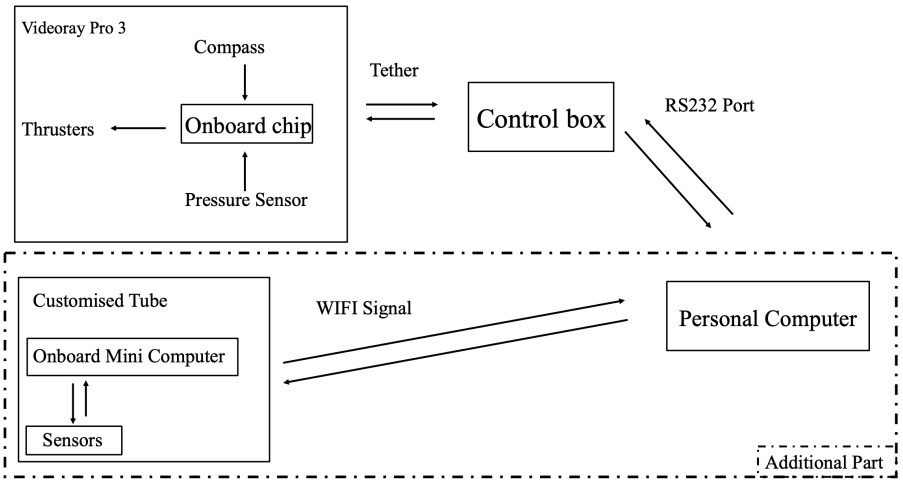


Figure 3.5 The procedure of data and command transfer. The box with the dotted line shows the additions made to the system in the present work.

However, WiFi performance is poor in underwater environments. Microwave radiation (such as WiFi) is very easily absorbed and reflected by water. Even raindrops are sufficient to cause noticeable disruption to WiFi. Hence, a WiFi Antenna should be out of the water and this was achieved by using a positively buoyant foam board platform. A sketch of this arrangement is shown in Figure 3.6.

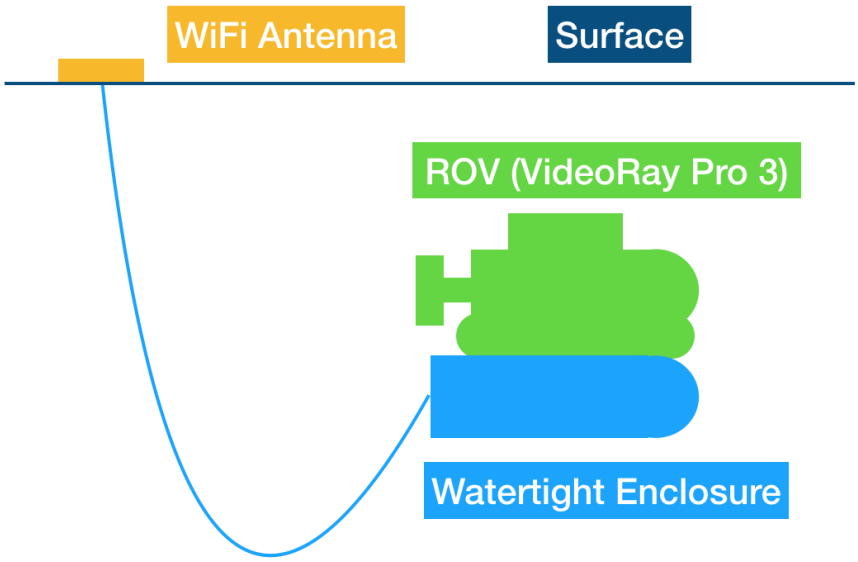


Figure 3.6 The Sketch of the Vehicle with WiFi Antenna

The layout of the sensors on the tray is illustrated in Figures 3.7, and 3.8.

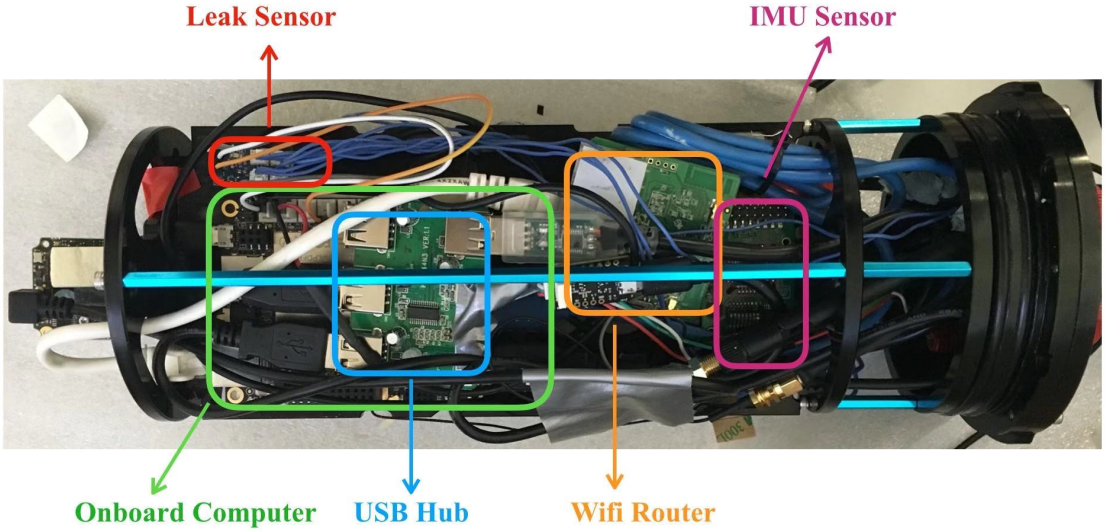


Figure 3.7 Electronic Tray Top view



Figure 3.8 Electronic Tray Back view

3.3. Sensors

The descriptions and transformation matrices of the main sensors, namely the IMU kit, the gyroscope, the monocular camera, the tracking camera, and a ping sonar, are presented in this section.

3.3.1. IMU sensor

The LPMS-ME1 DK is a low-cost IMU development kit with integrated signal processing, which can output attitudes in the form of quaternions and accelerations along xyz axes. The location of the IMU kit is shown in Figure 3.9.

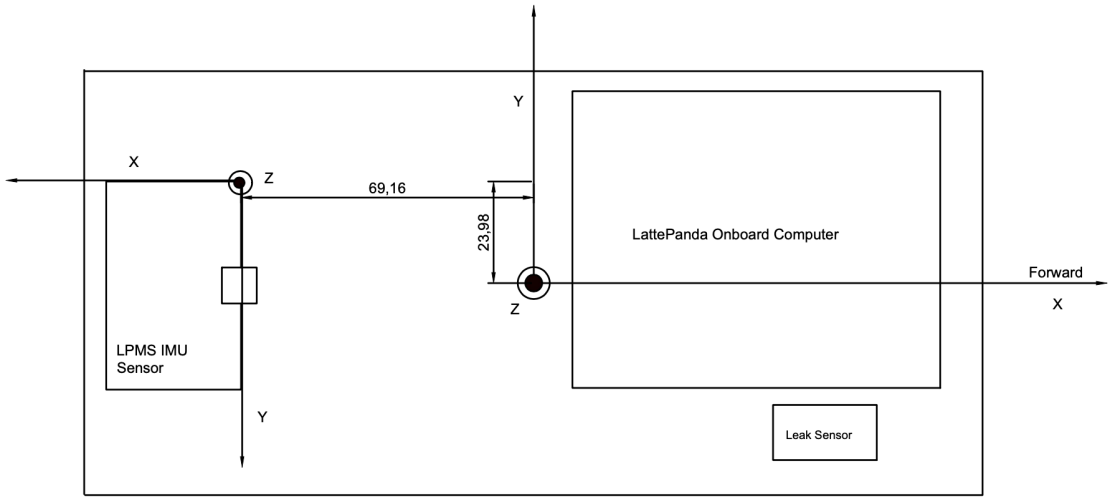


Figure 3.9 Top Tray (Unit: mm)

The transformation matrix from the reference of the IMU development kit to the body reference is presented as,

$$^{body}\boldsymbol{T}_{imu} = \begin{bmatrix} ^{body}\boldsymbol{R}_{imu} & ^{body}\boldsymbol{t}_{imu} \\ 0 & 1 \end{bmatrix}. \tag{3.1}$$

$$^{body}\boldsymbol{R}_{imu} = \boldsymbol{R}_z(\pi). \tag{3.2}$$

$$^{body}\boldsymbol{t}_{imu} = [-0.069\ 0.024\ 0.003] \tag{3.3}$$

The units are metres; $\boldsymbol{R}_z(\pi)$ indicates that the reference is rotated by 180° in the anticlockwise direction along the z-axis. Therefore, assuming $\boldsymbol{R}_{measure}$ is the rotation matrix related to the global reference frame, $\boldsymbol{R}_{body}$, the body reference relative to the global reference is,

$$^{global}\boldsymbol{R}_{body} = ^{global}\boldsymbol{R}_{measure}\ ^{body}\boldsymbol{R}_{imu}^{-1}. \tag{3.4}$$

These matrices will be used in derivations of the visual navigation methods in chapters 9 and 10.

3.3.2. Gyroscope

The LPMS-NAV2 is an inertial sensor for navigation applications, which is composed of a high accuracy one-axis gyroscope and a 3-axis accelerometer (LPMS, 2018). It can measure precise heading angles with ultra-low drift error. The location and axis of the gyroscope (LPMS-NAV2) are presented in Figure 3.10.

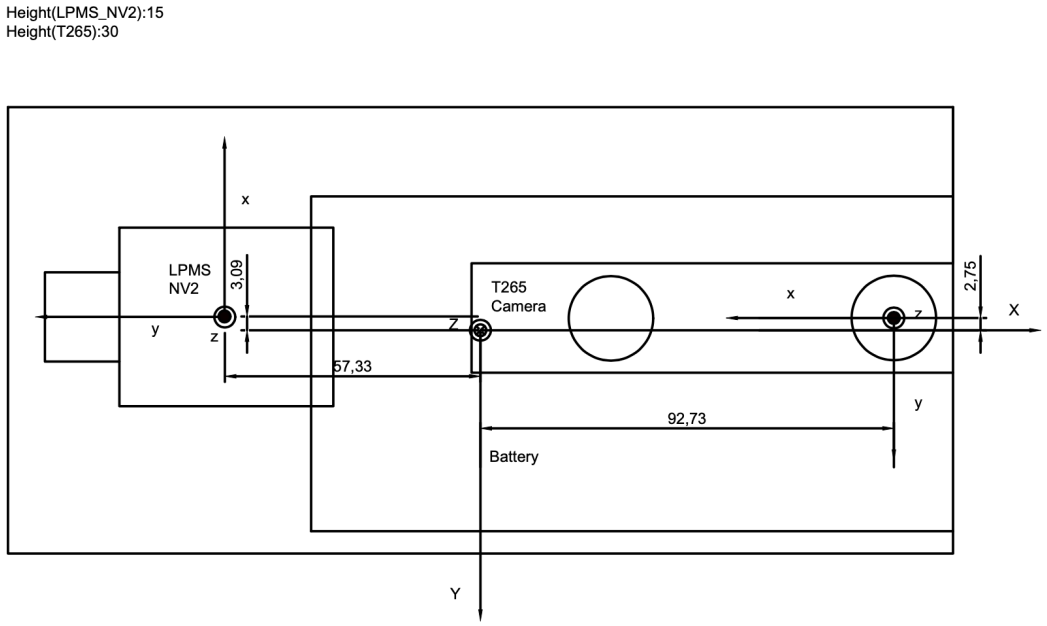


Figure 3.10 Bottom Tray (Unit: mm)

The transformation matrix from the NAV2 reference to the body reference is presented as,

$$^{body}\boldsymbol{T}_{nav} = \begin{bmatrix} ^{body}\boldsymbol{R}_{nav} & ^{body}\boldsymbol{t}_{nav} \\ 0 & 1 \end{bmatrix}. \tag{3.5}$$

$$^{body}\boldsymbol{R}_{nav} = \boldsymbol{R}_z(-\frac{\pi}{2})\boldsymbol{R}_x(\pi). \tag{3.6}$$

$$^{body}\boldsymbol{t}_{nav} = [-0.05733 \ -0.00309 \ -0.015] \tag{3.7}$$

Because the NAV2 only provides the heading angle relative to the global reference, the rotation matrix from the IMU is decomposed into Euler angles (yaw θ, roll ϕ and pitch ψ). Then, the yaw from the NAV2 is transferred to the body reference, and will replace the yaw from the IMU to generate a new rotation matrix combining IMU and Gyroscope information. Because the range of pitch of the vehicle in motion is $\pm\frac{\pi}{2}$, the Gimbal lock problem is not considered.

3.3.3. T265 Camera

The Intel RealSense T265 Tracking Camera is designed to provide tracking information using two fisheye lens sensors, an IMU and an Intel VPU (Visual Processing Unit). All of the V-SLAM algorithms run directly on the VPU, which means that the estimated trajectory can be obtained directly from the T265. It is stated that the T265 can provide less than 1% closed loop drift under intended use conditions (Realsense, 2018). The T265 has been implemented in mobile robotics projects and drone competitions to provide the navigation information (Bayer and Faigl, 2019) (Deegan et al., 2019) (Pelka et al., 2019). The installation of T265 is shown in Figure 3.10.

The transformation matrix from T265 reference to body reference is presented as,

$$^{body}\boldsymbol{T}_{t265} = \begin{bmatrix} ^{body}\boldsymbol{R}_{t265} & ^{body}\boldsymbol{t}_{t265} \\ 0 & 1 \end{bmatrix}. \tag{3.8}$$

$$^{body}\boldsymbol{R}_{t265} = \boldsymbol{R}_z(\pi)\boldsymbol{R}_x(\pi). \tag{3.9}$$

$$^{body}\boldsymbol{t}_{t265} = [0.09273 \ -0.00275 \ -0.03] \tag{3.10}$$

The estimated trajectory of the T265 can be transferred to the body reference using $\boldsymbol{T}_{t265}$, hence it can provide comparative results relative to the results from the proposed IVO method.

3.3.4. Monocular Camera

The mvBlueFOX-MLC 200w is a global shutter grayscale camera, located on the tray as shown in Figure 3.11. The camera can give 752×480 resolution images with 60 fps (frames per second). A wide-angle lens is used on the camera, and a perspective model is adopted to calibrate the camera.

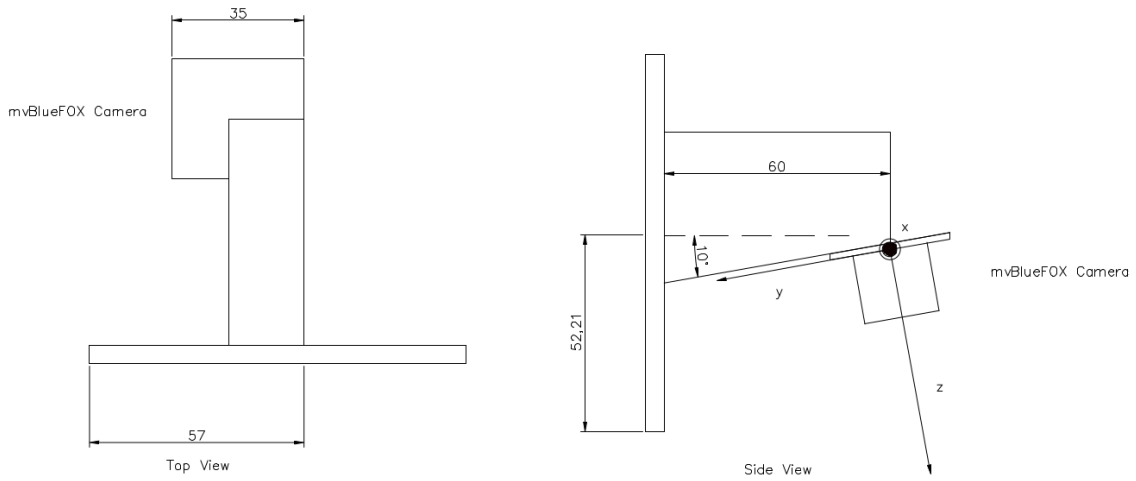


Figure 3.11 Camera Location (Unit:mm)

The transformation matrix from monocular camera reference to body reference is presented as,

$$^{body}\boldsymbol{T}_{cam} = \begin{bmatrix} ^{body}\boldsymbol{R}_{cam} & ^{body}\boldsymbol{t}_{cam} \\ 0 & 1 \end{bmatrix}, \tag{3.11}$$

$$^{body}\boldsymbol{R}_{cam} = \boldsymbol{R}_z(-\frac{\pi}{2})\boldsymbol{R}_x(\pi + \frac{\pi}{18}). \tag{3.12}$$

$$^{body}\boldsymbol{t}_{cam} = [0.166\ 0.024\ 0.003] \tag{3.13}$$

Therefore, feature points in camera coordinates can be transferred to the body coordinates by using $^{body}\boldsymbol{T}_{cam}$.

3.3.5. Ping Sonar

The Ping sonar is a multipurpose single-beam echosounder. It can be used as an altimeter for ROVs and AUVs, for bathymetry work aboard a USV, as an obstacle avoidance sonar, and in other underwater distance measurement applications. The range of the sonar is from 0.5m to 30m, and the beamwidth is 30°.

The sonar has been installed on the outside of the watertight tube. The location is shown in Figure 3.12.

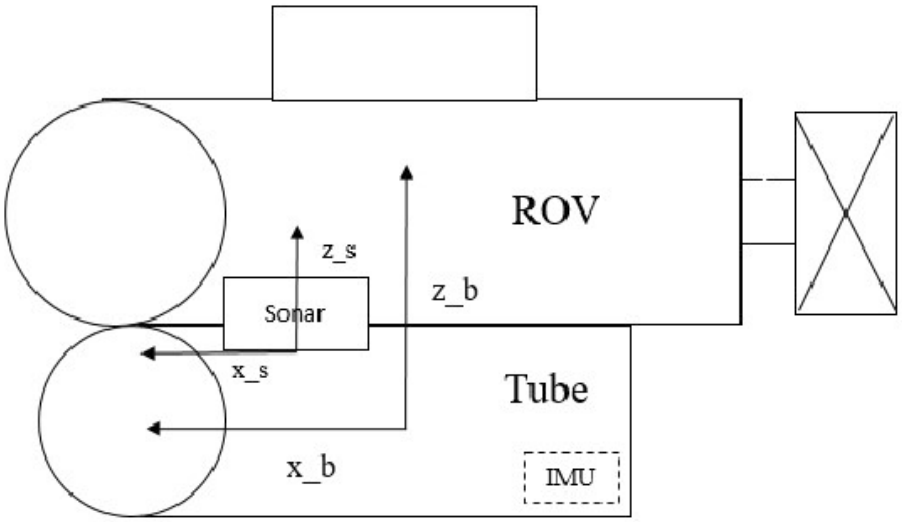


Figure 3.12 Sonar Location

Compared to the other sensors' transformation matrices, the transformation matrix for the sonar is much simpler because the two references, body and sonar, share identical rotation vectors and only have translation bias. Hence,

$$^{body}\boldsymbol{T}_{sonar} = \begin{bmatrix} ^{body}\boldsymbol{R}_{sonar} & ^{body}\boldsymbol{t}_{sonar} \\ \boldsymbol{0} & 1 \end{bmatrix}, \tag{3.14}$$

$$^{body}\boldsymbol{R}_{sonar} = \boldsymbol{I}. \tag{3.15}$$

The rotation matrix is the identity matrix and the translation vector is,

$$^{body}\boldsymbol{t}_{sonar} = [0.060,\ 0.070,\ 0.065]^T. \tag{3.16}$$

Through the $\boldsymbol{T}_{sonar}$ transformation, the depth from sonar to seabed becomes the depth from the body to the seabed.

3.4. Waterproof Solution & Buoyancy Balancing

There were two challenges for the modification. One was waterproofing, the other was the buoyancy balance. Even though the end plates of the watertight enclosure are sealed by O-ring rubber seals, the WiFi Antenna extension cable and battery charge cable needed to go pass

through cable penetrators on the back flange. Hence, potting the penetrator using epoxy glue was required.

In order to make sure the glue completely sealed the penetrator, the cables were held vertically so that the glue flowed down by gravity. This is presented in Figure 3.13.

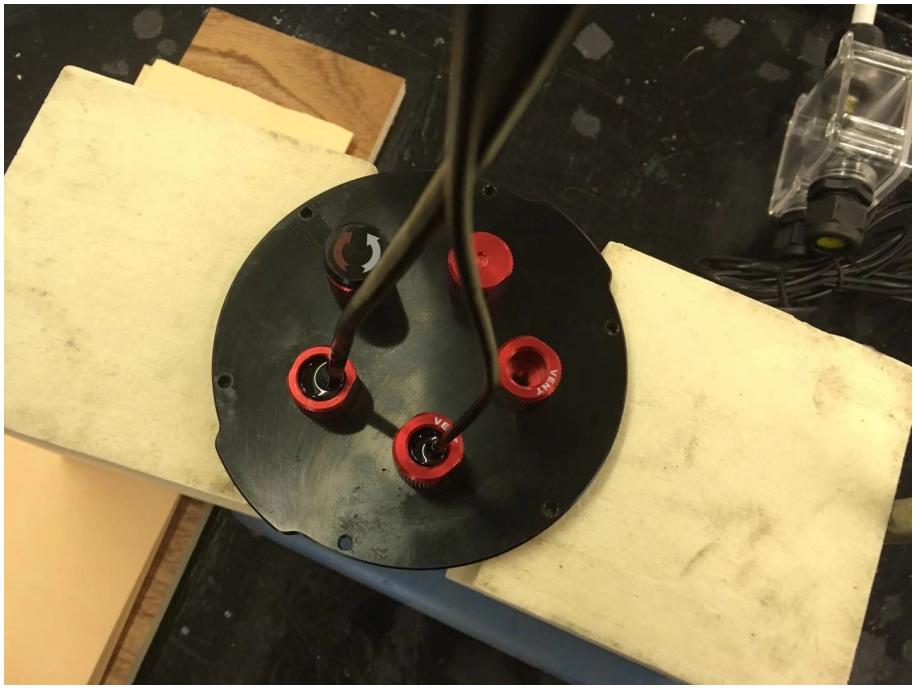


Figure 3.13 Potting The Penetrators

Finally, a vacuum test was carried out to check if there were any leaks, as show in Figure 3.14. An air pump was utilised to remove the air from the sealed tube. Hence, if the sealed part was leaking, the air would be pushed into the tube by atmospheric pressure. The gauge displays the variation of air pressure in the tube. During the operation, the vent plug cap is removed, and the vacuum plug is plugged in. Subsequently, the air in the sealed tube was pumped out to get 10-15 inHg vacuum. Then the gauge was monitored for 5-10 minutes to make sure there was no a clear variation in pressure.



Figure 3.14 Vacuum Test

The added tube changed the centre of gravity and centre of buoyancy of the entire vehicle. It also resulted in the ROV having positive buoyancy. Therefore, weights were required to be added on the body of vehicle. A small weight can lead to significant changes in buoyancy, so the operation had to be performed very carefully. The added weights on the vehicle are shown in Figure 3.15. A comparison between the vehicle without weights and the one with weights is illustrated in Figure 3.16 and Figure 3.17 respectively.

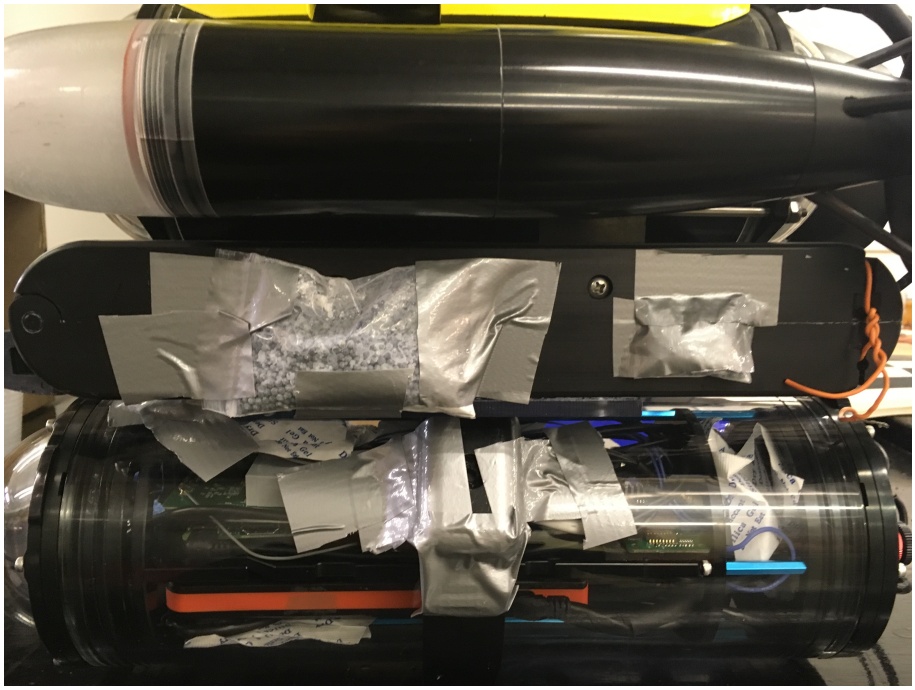


Figure 3.15 Adding Weights on The ROV

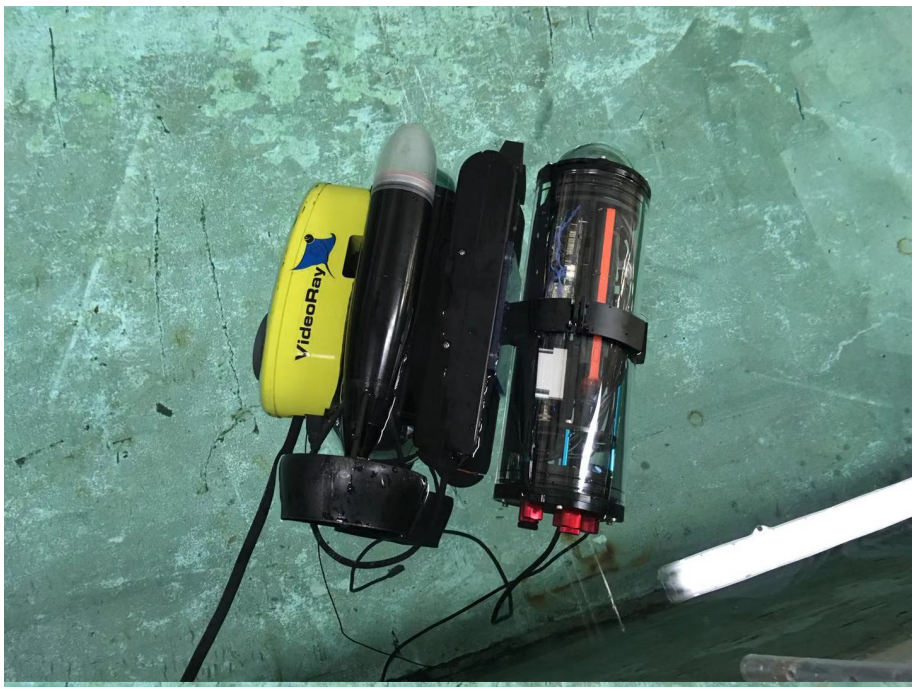


Figure 3.16 The Vehicle without Weight

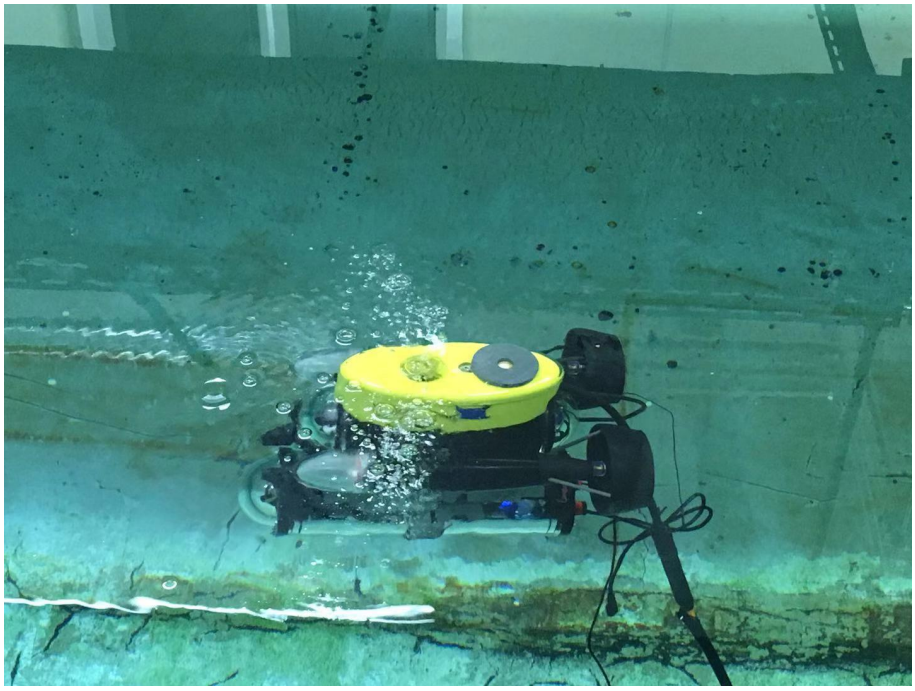


Figure 3.17 Balanced Buoyancy

3.5. Sensors Communications & Data Collection Programmes

Multiple sensors are required to communicate with the onboard computer. The gyroscope communicates with the onboard computer via the RS232 port. The associated communication programme has been coded, which can record the live data. The optical sensors including the monocular camera and stereo camera communicate with the mini computer via the ROS driver.

The sonar communicates with the computer via the USB port, and a python script was written to collect the depth information. The IMU device has an proprietary driver that can be applied directly. As the different sensors have different frequency rates for collection, every sensor has its own unique collection programme.

In the data collection process, there were many pieces of information from the different five sensors that needed to be recorded at same time. Hence, the collection programs for the different sensors run in parallel on the onboard computer to trigger the respective devices and collect data on the onboard computer temporarily. A photograph showing the testing of the recording program is presented in Figure 3.18.

A data association programme was used to synchronise the data set from the different sensors. As mentioned before, the different sensors have different collection rates. The association program was utilised to sort the data from different sensors in chronological order.

Additionally, a leak warning programme was also coded. The programme will give a warning if the leakage sensor detects any water. Even through the water-tight enclosure should guarantee sealed connections, a leak may happen accidentally.



Figure 3.18 Sensors Recording Test

3.6. Summary

The experimental platform has been introduced in this chapter. The original ROV (VideoRay Pro 3) and the modified configurations of the ROV have been used in the experimental evaluations for different controllers in Chapter 6. The modified configuration was also used in the data collection during experimental evaluation of the navigation systems in Chapter 11, because the multiple sensors can collect high quality data for the visual navigation systems. Having presented the experimental platform, the control theories will be discussed in the next chapter, which will illustrate the mathematical modelling for UUVs.

Part I

Control Theories for UUVs

Chapter 4. Mathematical Modelling for UUVs

In this Chapter, the mathematical modelling for UUVs is presented. The generic modelling is adapted for the VideoRay Pro 3, which is used in the simulation and practical tests. The dynamic and kinematic equations for UUVs are described. In order to separate the motion equations, the decoupled mode is introduced as well. Finally, the thruster parameters are presented.

4.1. Dynamic and Kinematic Modelling

4.1.1. Generic Modelling

The mathematical model of UUVs will be derived initially. A generic underwater vehicle has six Degrees Of Freedom (DOF). According to Fossen et al. (1994), the kinematic and dynamic models are

$$\dot{\boldsymbol{\eta}} = \boldsymbol{T}(\psi)\boldsymbol{v}; \tag{4.1}$$

where,

- $\boldsymbol{\eta}$ is the attitude and position vector;
- $\boldsymbol{T}$ is a transformation matrix;
- $\boldsymbol{\psi}$ is a vector presenting attitudes;
- $\boldsymbol{v}$ is a vector of velocities.

$$\boldsymbol{M}\dot{\boldsymbol{v}} = \boldsymbol{\tau} - \boldsymbol{C}(\boldsymbol{v})\boldsymbol{v} - \boldsymbol{D}(\boldsymbol{v})\boldsymbol{v} - \boldsymbol{g}(\boldsymbol{\eta}); \tag{4.2}$$

where,

- $\boldsymbol{M}$ is the inertia matrix including the physical mass and added mass;
- $\boldsymbol{D}(\boldsymbol{v})$ is a matrix including damping coefficients;

- $\boldsymbol{C}(\boldsymbol{v})$ is a matrix comprising centripetal and Coriolis terms;
- $\boldsymbol{\tau}$ is a vector containing the control inputs;
- $\boldsymbol{g}(\boldsymbol{\eta})$ is a vector indicating the buoyancy force and moments.

4.1.2. Modelling for VideoRay Pro 3

Due to the shape of the vehicle and the layout of the actuators, the rotations of roll and pitch are negligible, hence they have been removed from the dynamic model for the VideoRay Pro 3, as shown in Figure 4.1.

Hence, $\boldsymbol{\eta} = [x, y, z, \psi]$ and $\boldsymbol{v} = [u, v, w, r]$ represent the state vectors for attitude and velocity.

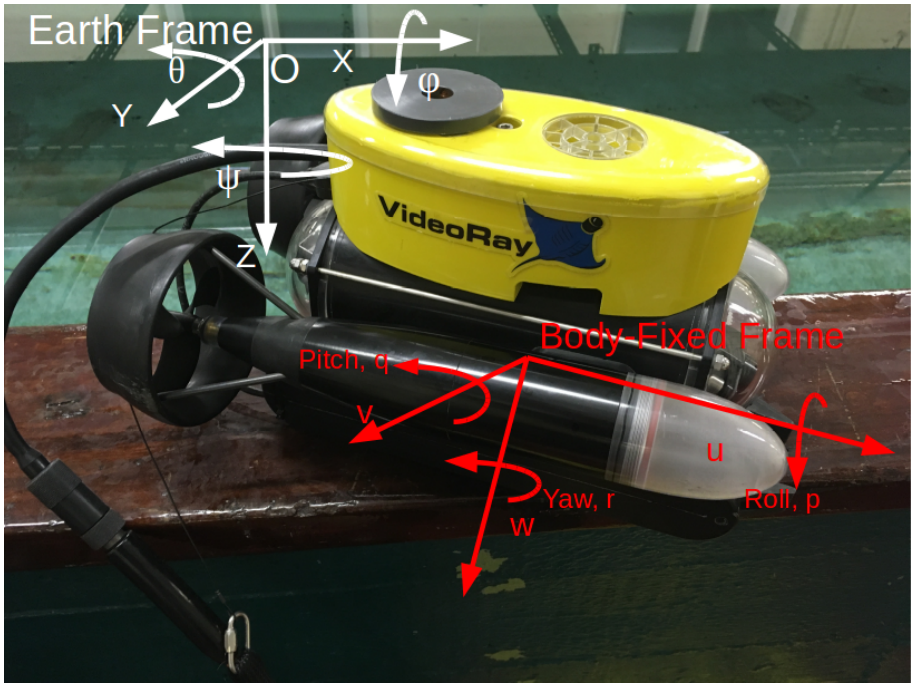


Figure 4.1 Videoray Pro 3

The rigid body mass matrix is presented as,

$$\boldsymbol{M}_{RB} = \boldsymbol{M}_{RB}^{T} = \begin{bmatrix} m & 0 & 0 & 0 \\ 0 & m & 0 & 0 \\ 0 & 0 & m & 0 \\ 0 & 0 & 0 & I_z \end{bmatrix} \tag{4.3}$$

The added mass matrix can be written as

$$\boldsymbol{M}_A = \boldsymbol{M}_A^T = \begin{bmatrix} X_{\dot{u}} & 0 & 0 & 0 \\ 0 & Y_{\dot{v}} & 0 & 0 \\ 0 & 0 & Z_{\dot{w}} & 0 \\ 0 & 0 & 0 & N_{\dot{r}} \end{bmatrix} \tag{4.4}$$

The Coriolis matrix is

$$\boldsymbol{C}(\boldsymbol{\nu}) = -\boldsymbol{C}^T(\boldsymbol{\nu}) = \begin{bmatrix} 0 & 0 & 0 & -mv \\ 0 & 0 & 0 & mu \\ 0 & 0 & 0 & 0 \\ mv & -mu & 0 & 0 \end{bmatrix} \tag{4.5}$$

Hence, $\boldsymbol{M}$ can be written as,

$$\boldsymbol{M} = \boldsymbol{M}_{RB} + \boldsymbol{M}_A. \tag{4.6}$$

The damping is shown in (4.7) including linear (D_l) and quadratic damping (D_q),

$$\boldsymbol{D}(\boldsymbol{\nu}) = \boldsymbol{D}_l + \boldsymbol{D}_q(\boldsymbol{\nu}) \tag{4.7}$$

where the damping matrix should be diagonal, as follows

$$\boldsymbol{D}_l(\boldsymbol{\nu}) = \begin{bmatrix} X_u & 0 & 0 & 0 \\ 0 & Y_v & 0 & 0 \\ 0 & 0 & Z_w & 0 \\ 0 & 0 & 0 & N_r \end{bmatrix} \tag{4.8}$$

and,

$$\boldsymbol{D}_q(\boldsymbol{\nu}) = \begin{bmatrix} X_{|u|u}|u| & 0 & 0 & 0 \\ 0 & Y_{|v|v}|v| & 0 & 0 \\ 0 & 0 & Z_{|w|w}|w| & 0 \\ 0 & 0 & 0 & N_{|r|r}|r| \end{bmatrix} \tag{4.9}$$

Restoring force is given by

$$\boldsymbol{g}(\boldsymbol{\eta}) = \begin{bmatrix} 0 \\ 0 \\ -(W-B) \\ 0 \\ 0 \\ 0 \end{bmatrix} \tag{4.10}$$

where W is the force due to gravitational pull on the system and B is the buoyancy force.

The transformation matrix is presented as:

$$\boldsymbol{T}(\boldsymbol{\eta}) = \boldsymbol{T}(\psi) = \begin{bmatrix} \cos\psi & -\sin\psi & 0 & 0 \\ \sin\psi & \cos\psi & 0 & 0 \\ 0 & 0 & 1 & 0 \\ 0 & 0 & 0 & 1 \end{bmatrix} \tag{4.11}$$

4.2. Decoupled Mode

In this section, the speed model, steering model and diving dynamic model are decoupled from the general dynamic model.

4.2.1. Forward Speed Control Model

As mentioned previously, roll and pitch are negligible for the VideoRay Pro 3 and therefore the interactions of sway, roll, pitch and yaw may be neglected. This implies that the speed equation can be written as (Fossen et al., 1994):

$$(m - X_{\dot{u}})\dot{u} = X_{|u|u}|u|u + \tau_X + X_{ext} \tag{4.12}$$

where u is the surge velocity, X_{ext} represents external disturbances due to waves and currents, and τ_X is a forward force generated by the pair of thrusters.

4.2.2. Steering Equations of Motion

Automatic steering or heading control for the VideoRay Pro 3 can be achieved by means of the pair of horizontal thrusters. The steering motion model can be written as,

$$\dot{\psi} = r; \tag{4.13}$$

$$(I_z + N_{\dot{r}})\dot{r} + (N_r + N_{|r|r}|r|)r = \tau_R. \tag{4.14}$$

where τ_R is a turning moment provided by the pair of horizontal thrusters.

4.2.3. Diving Equations of Motion

For the VideoRay Pro 3, the depth is controlled by means of a vertical thruster. Hence the diving motion model can be written simply as,

$$\dot{z_d} = w \tag{4.15}$$

$$(m + Z_{\dot{w}}\dot{w} + (Z_w + Z_{w|w|}|w|)w - (W - B) = \tau_D \tag{4.16}$$

where τ_D is a force provided by the vertical thruster.

4.3. Thruster Model

The thrusters produce forces and moments acting on the vehicle. An underwater vehicle's thrusters, both for propulsion and directional control, are highly nonlinear actuators. As there are fixed pitch propellers on the VidoeRay Pro 3, the thruster force (T) can be calculated by the forward speed (u) of the vehicle, the advance speed (u_a), and the propeller speed (n) (M, 1981):

$$T = \rho D^4(\alpha_1 + \alpha_2 \frac{u_a}{nD} n|n|), \tag{4.17}$$

where ρ is the water density, D is the diameter of the propeller and α_1 and α_2 are constants given by the propeller's properties.

A comprehensive study of thrusters and their influence on underwater vehicle manoeuvrability was presented in (Yoerger et al., 1990). By considering the energy balance of a control volume about a thruster, a simplified nonlinear equation for thrust T can be derived as:

$$\dot{n} = \beta \tau_{motor} - \alpha n|n|, \tag{4.18}$$

$$T = C_t n|n|, \tag{4.19}$$

where τ_{motor} is the input torque supplied by the thruster's motor, α, β and C_t are thruster constants. This indicates that there is a mapping from the thrust input signal to thruster force (Wang and Clark, 2007).

The mapping can be written as a membership function (f_T),

$$T = f_T(C(t)). \tag{4.20}$$

where $C(t)$ is the thruster input signal (electric current) as a function of time t and T is thruster force. The membership function f_T can be described as shown in Figure 4.2.

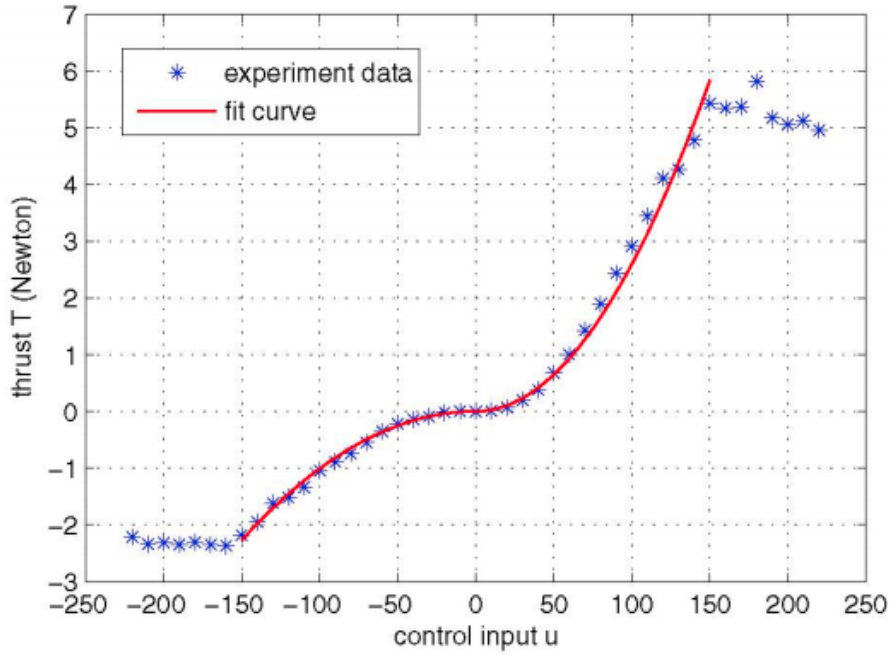


Figure 4.2 The Relation Between Thruster output and Control Input (Wang and Clark, 2006)

4.4. Layout of the Thrusters

The thrusters on the VideoRay Pro 3 are designed in such a way that the input thrust command is a value between -250 and +250, where the mapping curve used in this case was the one derived by Wang and Clark (2007). The thruster configuration of the ROV is presented in Figure 4.3 and the thruster force T_1, T_2, and T_3 are computed by a custom thrust allocation function:

$$T_1 = \frac{\tau_X}{2} + \frac{\tau_R}{2d} \tag{4.21}$$

$$T_2 = \frac{\tau_X}{2} + \frac{\tau_R}{2d} \tag{4.22}$$

$$T_3 = \tau_D \tag{4.23}$$

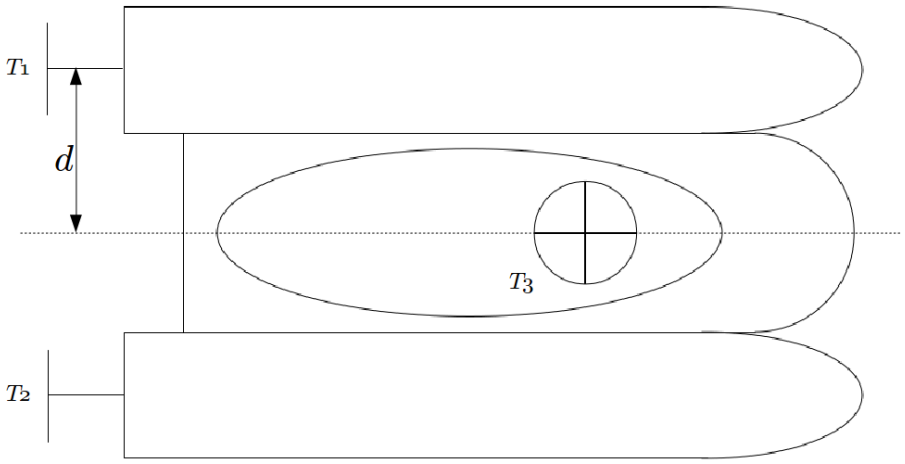


Figure 4.3 Thrusters Layout On VideoRay Pro 3

4.5. Summary

The dynamic and kinematic equations for UUVs have been derived in this chapter. These equations also have been adapted for the specific case of the VideoRay Pro 3. The model and layout of the thrusters have been illustrated in section 4.4. These equations and the mathematical modelling will be used in the derivation of controllers in the next Chapter.

Chapter 5. Control Methodologies

In this chapter, derivations of the controllers are presented. For underwater vehicles with one vertical thruster and two horizontal thrusters, the 4 DOF dynamic model may be decoupled into three non-interacting subsystems representing forward speed, steering and diving independently. These subsystems are controlled by means of the individual thrusters. In the simulations and tests, the forward speed was set to be constant; a PID controller has been implemented for diving control. There are four different controllers developed in the section: a typical linear controller: PID controller, two widely implemented nonlinear controllers: Sliding Mode Controller (SMC) and Backstepping Controller (BC), and a popular intelligent controller: Fuzzy logic Controller. Theoretically, linear controllers are supposed to be stable but have poorer performance in noisy environments; Nonlinear controllers can deal with environmental disturbances well, but they might be unstable in some situations; Intelligent controllers are developed based on experts' experience, and are independent of mathematical models of the vehicle so their performances are robust to changes in the coefficients in the model.

It should be noted that the length of the cable tethered to the ROV is less than 10 m and it is coated with foam material to attain neutral buoyancy. Therefore the influence of the cable on the dynamics of the vehicle is slight and it is neglected in this chapter.

5.1. PID Controller

The PID controller is widely used, due to its simplicity and effectiveness. For the ROV, the PID algorithm is as follows.

The Nomoto first order model is described as:

$$\dot{\psi} = r \tag{5.1}$$

$$\dot{r} = -\frac{1}{T'}(\frac{U}{L})r + \frac{K'}{T'}(\frac{U}{L})^2\tau_R \tag{5.2}$$

The PID robust control law for vehicle heading is,

$$\tau_R = \frac{T'}{K'}(\frac{L}{U})^2(\tau_{FF} - K_p e - K_d \dot{e} - K_i \int_0^t e(\tau)d\tau) \tag{5.3}$$

$$e = \psi - \psi_d \tag{5.4}$$

$$\tau_{FF} = \frac{1}{T'}(\frac{U}{L})r_d + \dot{r}_d \tag{5.5}$$

The control gains, K_p, K_i and K_d, can be found via various methods (Ogata, 2002). Here the pole-placement method (Fossen and Ross, 2006) was applied for fitting parameters in the control laws. For the case, $K_i = 0$ the error dynamics become,

$$\ddot{e} + (\frac{1}{T'}(\frac{U}{L}) + K_d)\dot{e} + K_p e = 0. \tag{5.6}$$

This suggests that,

$$K_d = 2\omega_n\zeta - (\frac{U}{L})\frac{1}{T'} \tag{5.7}$$

$$K_p = \omega_n^2 \tag{5.8}$$

where ω_n and ζ are pole-placement design parameters. The time constant for the integrator K_i may be chosen as approximately $\omega_n/10$.

5.2. Improved SMC Method

Sliding Mode Control (SMC) is a model-based controller for solving nonlinear system control issues (Utkin et al., 2009). The proposed SMC method however also benefits from the linear control characteristics, which is different with the traditional one. The sliding mode control for heading control is derived as follows.

The heading mode is,

$$\dot{\psi} = r \tag{5.9}$$

$$(I_z + N_{\dot{r}})\dot{r} + (N_r + N_{|r|r}|r|)r = \tau_R \tag{5.10}$$

The above is a nonlinear system. The sliding space (s-space) is defined as,

$$S = \lambda_1 \tilde{\psi} + \lambda_2 r \tag{5.11}$$

$$\tilde{\psi} = \psi - \psi_d \tag{5.12}$$

The derivative of the s-space is,

$$\dot{S} = \lambda_1 \dot{\psi} + \lambda_2 r = \lambda_1 r + \lambda_2 \dot{r} \tag{5.13}$$

where $\lambda_1 < 0$ and $\lambda_2 > 0$. Consider the Control Lyapunov Function (CLF),

$$V_s = \frac{1}{2} S^2 \tag{5.14}$$

Differentiating the CLF,

$$\begin{aligned} \dot{V}_s &= S\dot{S} \\ &= S(\lambda_1 r + \lambda_2 \dot{r}) \\ &= S(\frac{\lambda_2}{\lambda_1} S - \frac{\lambda_1^2}{\lambda_2} \tilde{\psi} + \lambda_2 \dot{r}) \end{aligned} \tag{5.15}$$

The $\dot{r}$ can be gained from the model equations,

$$\dot{r} = \frac{1}{(I_z + N_{\dot{r}})} (\tau_R - (N_r + N_{|r|r}|r|)r) \tag{5.16}$$

The SMC control law is,

$$\tau_R = (\hat{N}_r + \hat{N}_{|r|r}|r|)r - (\hat{I}_z + \hat{N}_{\dot{r}})(-\frac{\lambda_1^2 \tilde{\psi}}{\lambda_2^2} + \eta sign(S)) \tag{5.17}$$

The *sign* function is replaced by tanh, because the tanh function is continuous and almost equal to the *sign* function but it is differentiable. The SMC control law can be written as,

$$\tau_R = (\hat{N}_r + \hat{N}_{|r|r}|r|)r - (\hat{I}_z + \hat{N}_{\dot{r}})(-\frac{\lambda_1^2 \tilde{\psi}}{\lambda_2^2} + \eta \tanh(S)), \tag{5.18}$$

where, the hat symbol $(\hat{N}, \hat{I})$ indicates estimated parameters. From the SMC control law, the control input can be decoupled into a linear part and a nonlinear part.

$$\tau_R = \tau_{linear} + \tau_{nonlinear} \tag{5.19}$$

$$\tau_{linear} = \frac{(\hat{I}_z) + \hat{N}_{\dot{r}})\lambda_1^2}{\lambda_2^2}\tilde{\psi} \tag{5.20}$$

$$\tau_{nonlinear} = (\hat{N}_r + \hat{N}_{|r|r}|r|)r - (\hat{I}_z + \hat{N}_{\dot{r}})\eta \tanh(S) \tag{5.21}$$

For the linear part, the dynamic system can be described as:

$$\dot{\psi} = r \tag{5.22}$$

$$(I_z + N_{\dot{r}})\dot{r} + N_r r = K_l \tilde{\psi} \tag{5.23}$$

$$K_l = \frac{(\hat{I}_z) + \hat{N}_{\dot{r}})\lambda_1^2}{\lambda_2^2} > 0 \tag{5.24}$$

Here, r is usually a small value, hence $|r|r$ has been neglected. After applying the Laplace Transform, it becomes,

$$\Psi(s) = \frac{K_l}{(I_z + N_{\dot{r}})s^2 + N_r s + K_l}\psi_d; \tag{5.25}$$

$$\Psi_d(s) = \psi_d \frac{1}{s}. \tag{5.26}$$

where s indicates the frequency domain. The linear part can be considered as a second order dynamic system. Then, the damping ratio ζ_s becomes

$$\zeta_s = \frac{N_r}{2\sqrt{K_l(I_z + N_{\dot{r}})}}. \tag{5.27}$$

The undamped natural frequency ω_n is,

$$\omega_s = \sqrt{\frac{K_l}{(I_z + N_{\dot{r}})}}. \tag{5.28}$$

Hence, the λ_1 and λ_2 can be derived from the specific ζ_s and ω_s. The error distances between the actual parameters and estimated parameters are presented as follows.

$$\triangle_e = (\hat{N}_r + \hat{N}_{|r|r}r) - (N_r + N_{|r|r}r) \tag{5.29}$$

$$\ell_e = \frac{(\hat{I}_z + \hat{N}_{\dot{r}})}{(I_z + N_{\dot{r}})} \tag{5.30}$$

From (5.19) to (5.28) , the control law of the SMC is divided into two parts, one is the linear part, and the other is the nonlinear part. The nonlinear part is derived by the traditional SMC method but for the linear part, the pole-placement method is applied to confirm the properties of transient response such as delay time, rise time, settling time and overshoot. Hence the proposed SMC inherits the merits from linear controllers and nonlinear controllers. It can be robust to environmental factors and hydrodynamic coefficients, but also can have a quick response with small overshoot compared with other nonlinear controllers.

5.2.1. Stability Analysis

The derivative of the control Lyapunov Function is described as,

$$\dot{V}_s = S(\frac{\lambda_2}{\lambda_1} S - \frac{\lambda_1^2}{\lambda_2}\tilde{\psi} + \lambda_2 \dot{r}). \tag{5.31}$$

Substituting (5.13) and (5.17) into (5.31),the derivative of the CLF becomes,

$$\dot{V}_s = S(\frac{\lambda_2}{\lambda_1} S + (\ell_e - 1)\frac{\lambda_1^2}{\lambda_2}\tilde{\psi}) + \frac{\triangle_e}{(I_z + N_{\dot{r}})} - \ell_e \eta \tanh(S). \tag{5.32}$$

where $\lambda_1 < 0$, $-|\frac{\lambda_2}{\lambda_1}| = \frac{\lambda_2}{\lambda_1}$, and $\triangle_e \approx 0$, $\ell_e \approx 1$. Hence, the estimated terms are bounded, and can be rewritten as,

$$\xi_1 > |(1 - \ell_e)\frac{\lambda_1^2}{\lambda_2}\tilde{\psi}|; \tag{5.33}$$

$$\xi_2 > |\frac{\triangle_e}{(I_z + N_{\dot{r}})}|. \tag{5.34}$$

Therefore,

$$\dot{V}_s = -|\frac{\lambda_2}{\lambda_1}|S^2 + (\xi_1 + \xi_2)S - \ell_e \eta \tanh(S)S. \tag{5.35}$$

Because ℓ_e is positive, η is chosen as,

$$\eta >> (\xi_1 + \xi_2)/\ell_e \tag{5.36}$$

Hence, the derivative of CLF becomes,

$$\dot{V}_s = -|\frac{\lambda_2}{\lambda_1}|S^2 - (\ell_e \eta \tanh(S) - (\xi_1 + \xi_2))S < 0 \tag{5.37}$$

Therefore, the control law based on the SMC is stable.

5.3. Backstepping Control

Backstepping Control (BC) is also based on the control Lyapunov function (Krstic et al., 1995). The BC for heading angle is designed based on previous work from Fossen (Fossen and Ross, 2006). It has two steps because of the second order nature of the mathematical model. The first step starts with z_1'.

The control Lyapunov Function is,

$$V_1 = \frac{1}{2} z_1'^2 \tag{5.38}$$

The sliding surface is defined as,

$$z_1' = \lambda(\psi - \psi_d) \tag{5.39}$$

Hence,

$$\dot{V}_1 = z_1' \dot{z}_1' = z_1'(\lambda \dot{\psi}) = z_1' \lambda r \tag{5.40}$$

Choosing λr as the virtual controller, that is

$$\lambda r = \kappa + z_2' \tag{5.41}$$

gives

$$\dot{V}_1 = z_1'(\kappa + z_2') \tag{5.42}$$

The stabilising function κ is chosen as:

$$\kappa = -k_1 z_1' \tag{5.43}$$

where $k_1 > 0$ is the feedback gain while the new state z_2' is left undefined until the next step. This finally gives

$$\dot{V}_1 = -k_1 z_1'^2 + z_1' z_2' \tag{5.44}$$

The second step starts with z_2'

$$\dot{z'}_2 = \lambda \dot{r} - \dot{\kappa} \tag{5.45}$$

The next CLF is,

$$V_2 = V_1 + \frac{1}{2} z'^2_2. \tag{5.46}$$

Hence,

$$\dot{V}_2 = \dot{V}_1 + z_2 \dot{z}_2' = -k_1 z_1'^2 + z_1' z_2' + z_2'(\lambda \dot{r} - \dot{\kappa}). \tag{5.47}$$

Substituting the dynamic heading model into the expression for $\dot{r}$ gives

$$\dot{V}_2 = -k_1 z_1'^2 + z_2'(z_1' + \frac{\lambda}{(I_z + N_{\dot{r}})}(\tau_R - (N_r + N_{|r|r} r)) - \dot{\kappa}) \tag{5.48}$$

The control law is described as,

$$\tau_R = (N_r + N_{|r|r} r) - \frac{(I_z + N_{\dot{r}})}{\lambda}(\dot{\kappa} - z_1' + k_2 z_2'^2) \tag{5.49}$$

where $k_2 > 0$, finally giving

$$\dot{V}_2 = -k_1 z_1'^2 - k_2 z_2'^2 < 0. \tag{5.50}$$

Therefore, the proposed controller based on the BC is stable when implementing the control law,

$$\dot{\kappa} = -k_1 r \tag{5.51}$$

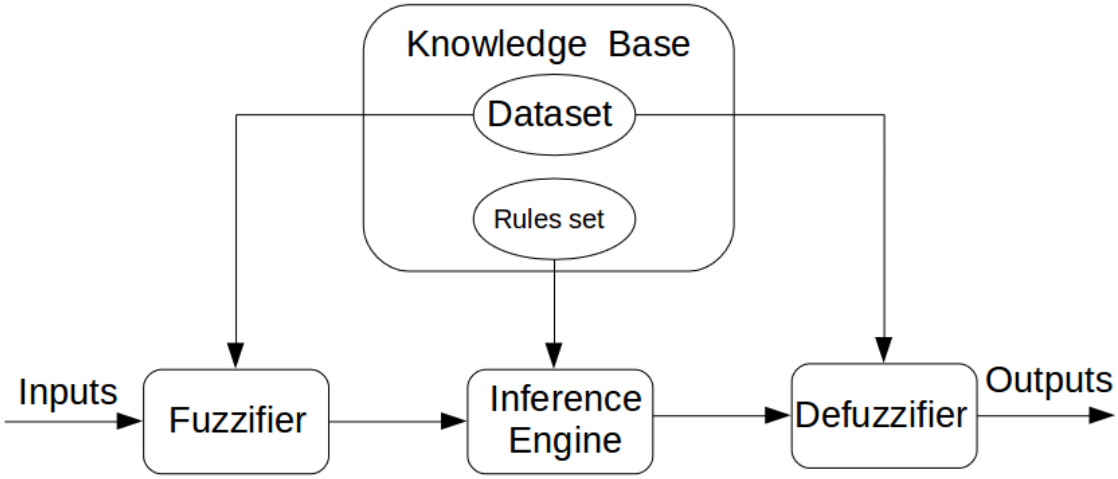


Figure 5.1 Fuzzy inference systems

5.4. Customised Second-Order Fuzzy Logic Control

A Fuzzy Logic Control (FLC) system may be designed based on human operator experience or a control engineer's knowledge (Zimmermann, 1996). The proposed FLC was designed for the underwater vehicle based on the previous work from (Zhang, 2016). The input variables $\tilde{\psi}$ and r are heading error and heading angular velocity respectively. Hence, it is a second-order FLC method, and customises the rule set for the VideoRay Pro 3. The process of fuzzy inference is shown in Figure 5.1. The inputs are fuzzified first according to the membership functions so that the input variables are described by fuzzy sets.

In the inference process, rules and logical operators are used to map the fuzzified inputs on to the fuzzified outputs. The rules list is presented in Table 5.1.

r \ $\tilde{\psi}$	NB	NM	NS	ZE	PS	PM	PB
NB	PB	PB	PB	PB	PM	PS	ZE
NM	PB	PB	PB	PM	PS	ZE	NS
NS	PB	PB	PM	PS	ZE	NS	NM
ZE	PB	PM	PS	ZE	NS	NM	NB
PS	PM	PS	ZE	NS	NM	NB	NB
PM	PS	ZE	NS	NM	NB	NB	NB
PB	ZE	NS	NM	NB	NB	NB	NB

Table 5.1 Rules List

In Table 5.1, "NB" is Negative Big, "NS" indicates Negative Small, and "NM" indicates Negative Middle. Similarly, "P" is positive. And "ZE" means zero. Finally, the defuzzification generates the output variable σ_R. This process also relies on related membership functions. The input fuzzification membership functions are presented in Figure 5.2 and Figure 5.3 and the output membership function is presented in Figure 5.4.

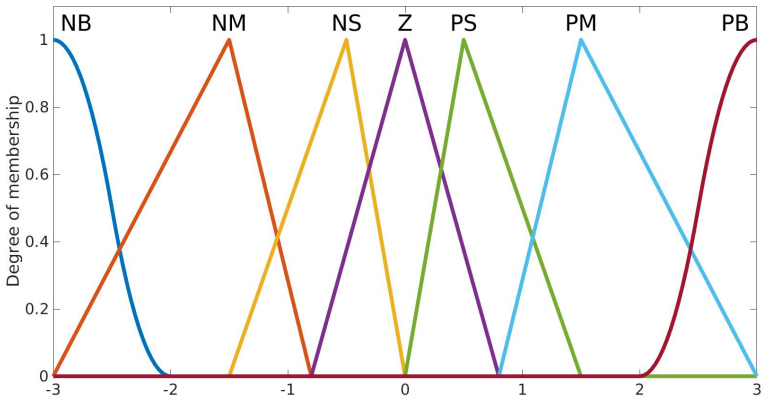


Figure 5.2 Fuzzified Input $\tilde{\psi}$

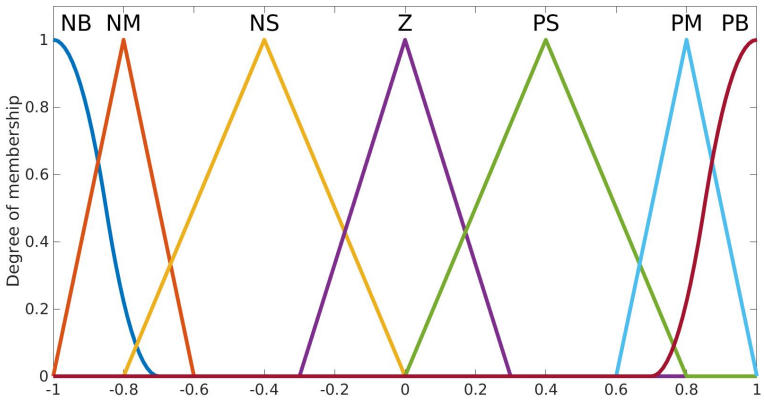


Figure 5.3 Fuzzified Input r

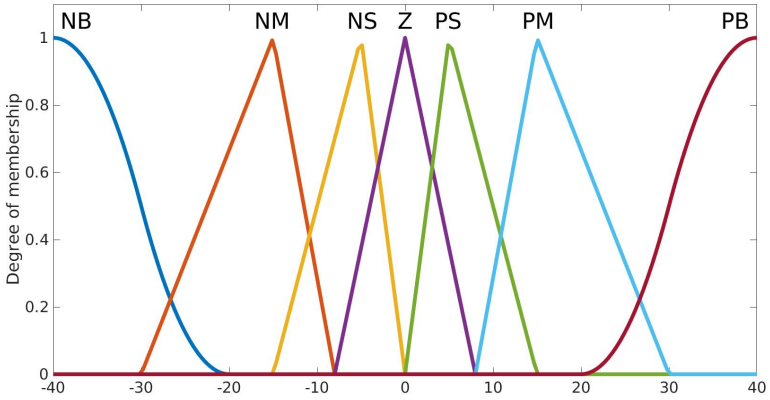


Figure 5.4 Fuzzified Output τ_R

5.4.1. Stability Analysis

For FLC, the CLF method can be employed to check its stability. The approach is analogous to the stability analysis of the SMC. The space (S_f) can be regarded as,

$$S_f = \tilde{\psi} + r. \tag{5.52}$$

The derivative of the CLF is described as,

$$\dot{V} = \dot{S_f} S_f = S_f(-S_f + \tilde{\psi} + \dot{(r)}) \tag{5.53}$$

Then,

$$\dot{V} = S_f(-S_f + \tilde{\psi} + \frac{1}{(I_z + N_{\dot{r}})}(\tau_R - (N_r + N_{|r|r}|r|)r)) \tag{5.54}$$

because the membership functions are piecewise functions. Because there are $7 \times 7 = 49$ rules, there are 49 situations, but for each situation, $\tilde{\psi}$ and r are bounded. Hence, it can be verified that for each situation the generated output (τ_R) can make sure that $\dot{V} < 0$.

5.5. Summary

The derivations of controllers (PID, SMC, FLC, and BC) have been presented in this chapter. The PID controller was derived based on the pole-placement method. The nonlinear controllers (SMC and BC) was derived from the Lyapunov law. The SMC algorithm has been improved by integrating linear control characteristics. The FLC has been customised for the Videoray Pro 3. The proofs of stability have also been shown. These controllers will be tested through simulations and experiments in Chapter 6.

Chapter 6. Simulations & Practical Experiments for the Controllers

In this chapter, simulations and physical experiments are presented to verify the controllers under different conditions using the original ROV. Additionally, the modified ROV was utilised to test the robustness of controllers on the different hydrodynamic coefficients. The results are quantified, and discussed at the end of this chapter.

6.1. Simulations

The controllers were simulated on the VideoRay Pro 3 in C++. In the simulations, the desired heading angle was set to 172^o, and the initial heading angle was 0.0^o. Firstly, the controllers were simulated in noise-free conditions and the results are shown in Figure 6.1. The associated control signals are presented in Fig .6.2. The simulation was then run with added noise on the yaw angular velocity, as shown in Figure 6.3, again with the associated control signals plotted in Figure 6.4. The noise ε is considered as white noise, it follows a Gaussian Distribution as,

$$\varepsilon \sim N(0, \sigma_s^2). \tag{6.1}$$

The units are rad/s. In the simulation, the $\sigma_s = 0.1$. The tuning parameters for the different controllers are listed in Table 6.1.

PID Controller			
Parameter	K_p	K_d	
Value	0.64	1.50	
SMC			
Parameter	λ_1	λ_2	η
Value	22.30	3.78	102.36
BC			
Parameter	K_1	K_2	λ
Value	1.30	401.12	1.22

Table 6.1 Tuning parameters

6.1.1. Quantitative Evaluation Methods

The performance may be analysed in terms of several control properties:

- Settling Time: The settling time is the time required for the response curve to reach and stay within a range, about the desired value, of a size specified in terms of an absolute percentage of the desired value (in this case 5%).
- Overshoot: The overshoot is the maximum peak value of the response curve. If the peak value is much bigger than the desired value, then it is common to use the overshoot for evaluation of a control method. It is defined by

$$Overshoot = c(t_p) - Q. \tag{6.2}$$

  where c is the response curve, and t_p is the peak time. The peak time is the time required for the response to reach the first peak of the overshoot. Q is the desired value.

- Average Error: The average error is used to represent the steady error. It is defined by,

$$e_{average} = \frac{\sum_{t=t_s}^{T_i} \tilde{\psi(t)}}{T_i - t_s}. \tag{6.3}$$

  where t_s is the Settling Time, and T_i is the total time. Because the numerator in the polynomial is a linear term, the units of $e_{average}$ are o (degrees).

- Variance of Error: The Variance of Error is used to describe the oscillation of the response curve after t_s. It is defined by,

$$Var_e = \frac{\sum_{t_s}^{T_i} (\tilde{\psi(t)} - e_{average})^2}{T_i - t_s}. \tag{6.4}$$

  Because the numerator in the polynomial is quadratic, the units of Var_e are o^2.

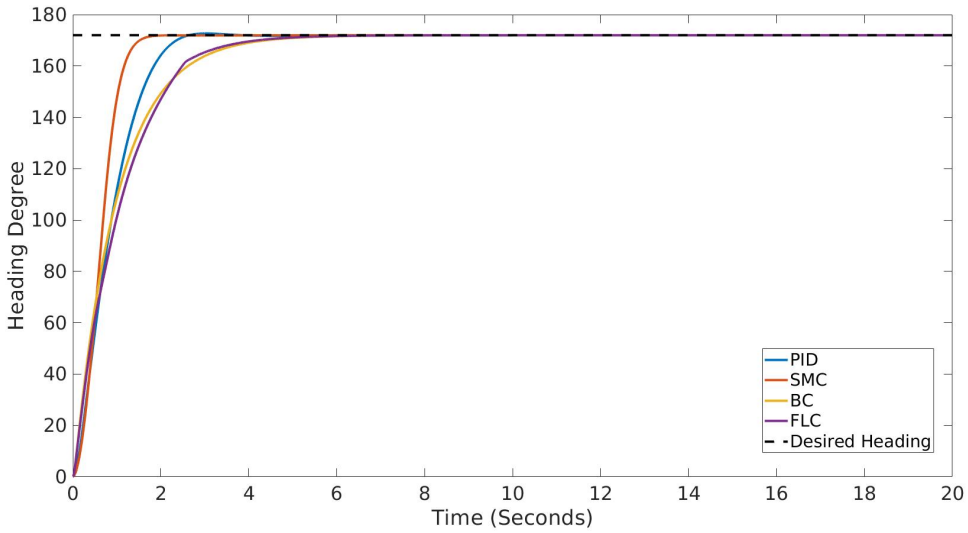


Figure 6.1 Simulation without Noise, Heading Angle

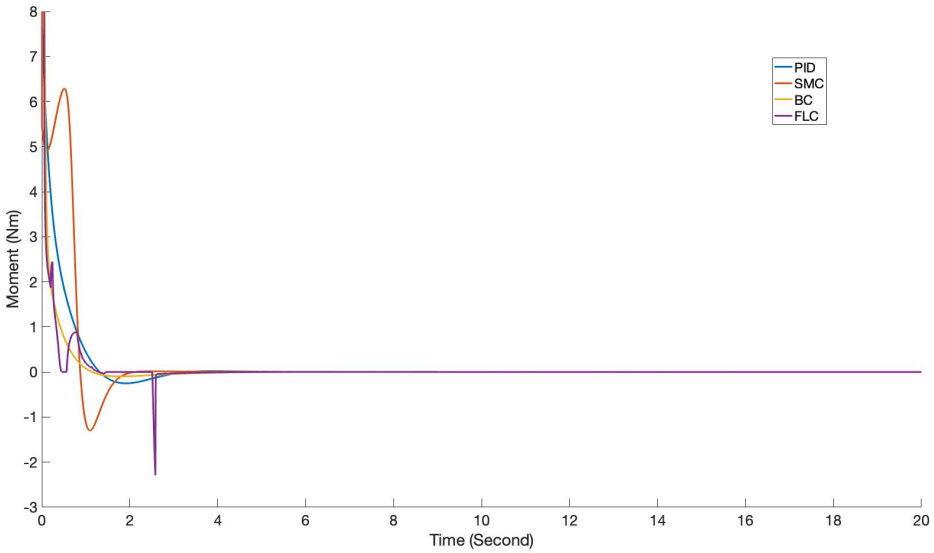


Figure 6.2 Simulation Control Signal without Noise

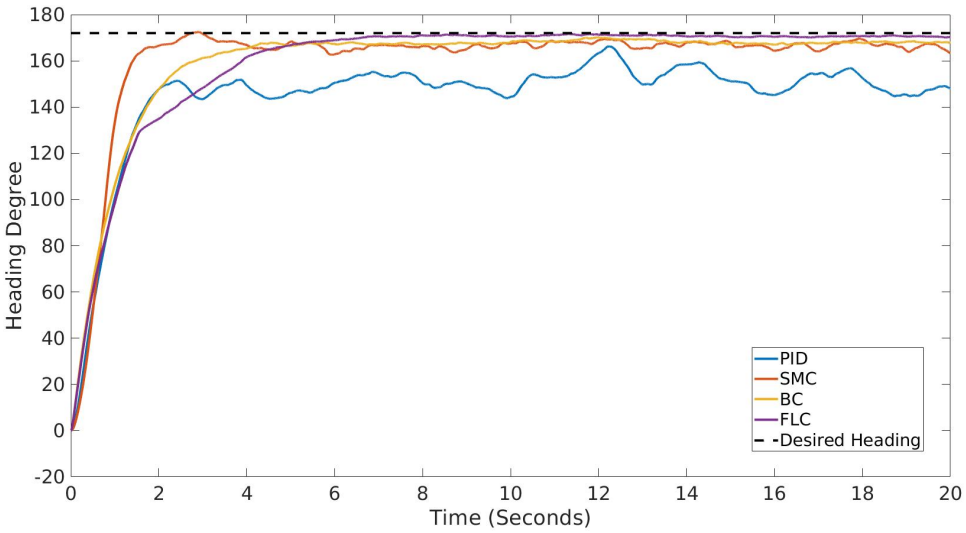


Figure 6.3 Simulation With Added Noise, Heading Angle

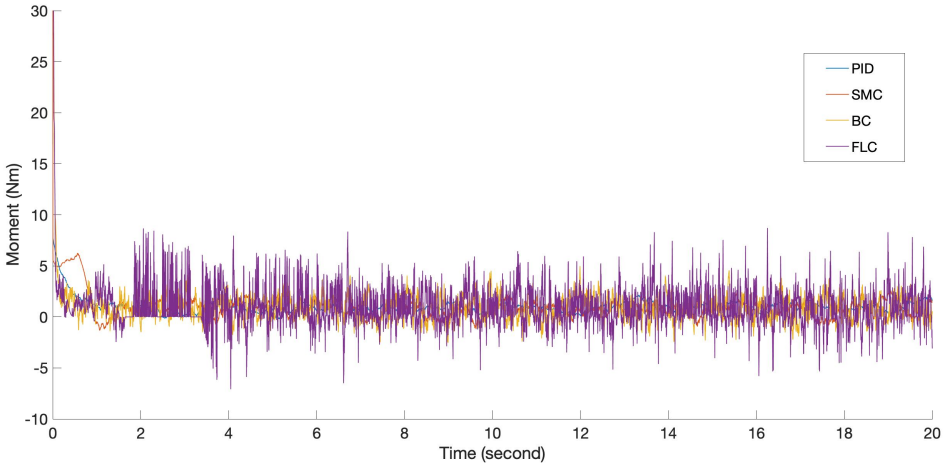


Figure 6.4 Simulation Control Signal with Noise (full response)

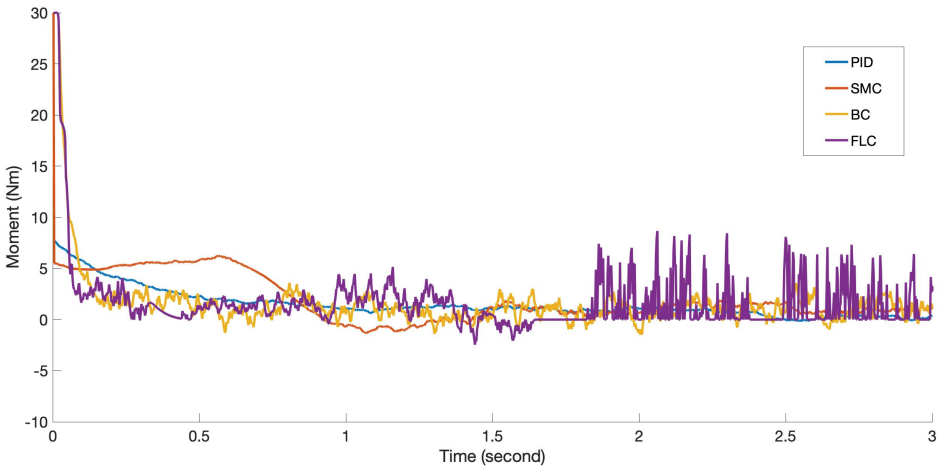


Figure 6.5 Simulation Control Signal with Noise (three seconds truncation from the full response)

The results of the analysis of the controller outputs are shown in Table 6.2. For the case without any noise, the four controllers had very similar performances. Due to the pole-placement method used for tuning, the PID controller had a small Average Error relative to the BC and FLC. This indicates that the steady error of a PID in idealised conditions can be small. The SMC had the shortest Settling Time, smallest Average Error and smallest Variance of Error among the four controllers. The reason for this is that the SMC has two parts: a linear part, and a nonlinear part. The merits of the linear controller improve the performance of the SMC. Meanwhile, the performances of the BC and FLC were very similar.

Situations	Properties	PID	SMC	BC	FLC
Noises-Free	Settling Time (s)	1.97	1.24	2.95	2.74
	Overshoot (^{o})	0.70	0	0	0
	Average Error (^{o})	0.13	0.1	0.5	0.50
	Variance of Error($^{o^2}$)	0.60	0.38	1.87	1.88
Added Noise	Settling Time (s)	12.00	1.57	3.42	4.24
	Overshoot (^{o})	0	0.49	0	0
	Average Error (^{o})	19,79	5.14	4.18	1.71
	Variance of Error ($^{o^2}$)	28.78	2.77	1.00	1.89

Table 6.2 Simulation Results Analysis With Noises

For the case with added noise, the PID controller ceased to perform satisfactorily. The Settling Time, Average Error, and Variance of Error all increased dramatically. This suggested that the PID cannot control the underwater vehicle effectively. The model-based nonlinear controllers (BC and SMC) and the intelligent controller (FLC) were still able to control the vehicle towards the desired heading angle. Compared to the BC, the FLC had a small Average Error, but a big Variance of Error. This means that the heading of the vehicle controlled by the FLC had large oscillations, as shown in Figure 6.3, but the centre of the oscillation was close to the desired heading angle. The performance of the SMC also declined sharply. The reason for this was that the linear control part of the proposed SMC had a negative impact on the overall performance. In Figure 6.4, the FLC is a second-order system and aims to tune the angular velocity to be zero at the desired heading angle. In the simulations with noise added, the random noise is imposed on the angular velocity. Hence, the FLC outputs the large and high frequency moments to reduce the angular velocity. In order to show them more clearly, a fragment of three seconds duration is truncated from the full response (Figure 6.4), as shown in Figure 6.5.

The simulations were executed under three assumptions. 1) the dynamic model parameters were known exactly and never changed. 2) the actuators can produce the forces and moments required by the control laws immediately. 3) the disturbances are white noise following a Gaussian Distribution. While under ideal conditions these assumptions can be met, they cannot be satisfied in real conditions including white noise. Therefore the section 6.2 presents practical experiments for comparison.

6.1.2. Path following

Applying Way Point Guidance by Line of Sight (LOS) (Caharija et al., 2016), the controllers can keep the vehicle tracking desired trajectories. The path is given by a set of way points $(0,0),(1,0),(1,0.5),(0,0.5),(0,1),(1,1),(1,1.5),(0,1.5)$, where the unit is metres. The next way point will be selected once the vehicle reaches a circle of acceptance with a radius around the way point. The radius for the PID controller was set to 0.1m, due to its large steady-state error. The other controllers have a 0.05m acceptance radius. The results are presented in Figure 6.6 and Figure 6.7 for the ideal and with white noise cases respectively.

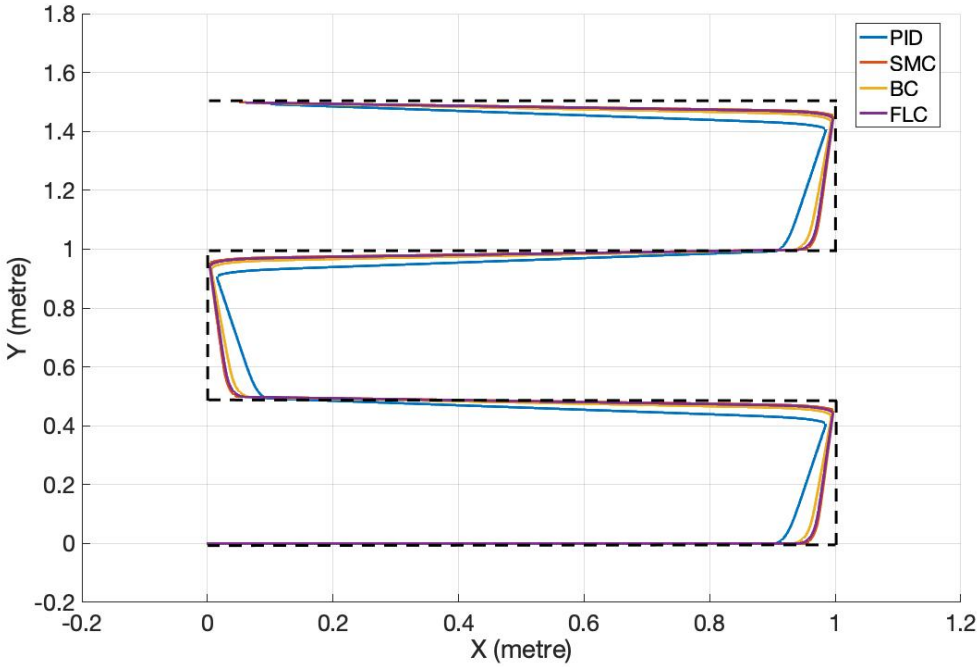


Figure 6.6 Simulation Path Following Without Noise (Dotted line denotes the reference trajectory)

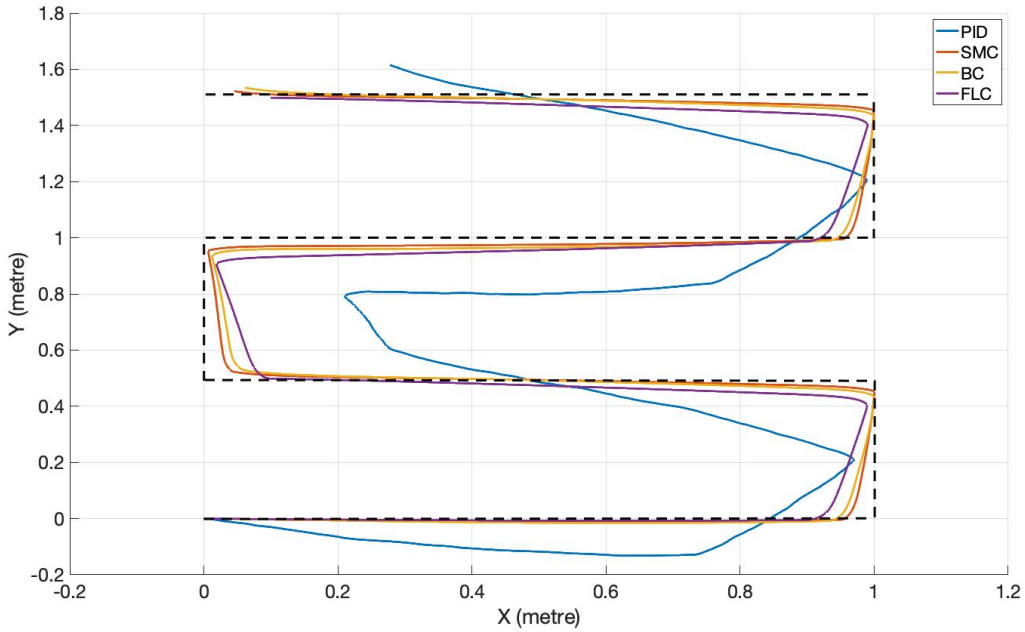


Figure 6.7 Simulation Path Following with Noise (Dotted line denotes the reference trajectory)

The path-following results are consistent with the results from the heading angle control. In the noise-free condition, all controllers can guide the vehicle in following the desired path, but the PID had a bigger offset compared with others. However, in the condition with white noise added, the PID nearly failed to follow the path and the proposed SMC turned the vehicle with the smallest cross track error among the four controllers.

6.2. Experiments

6.2.1. Developing Control System

The Control System for the VideoRay Pro 3 was developed based on the ROS platform, which includes communication, manual control, automatic control, guidance, and navigation modules. The serial port libraries were implemented to read data from the vehicle and send commands to control the thrusters through the RS232 interface. The control commands were generated by the control algorithms after receiving the sensor information. The system is shown at Figure 6.8, and the control block diagram is shown at Figure 6.9.

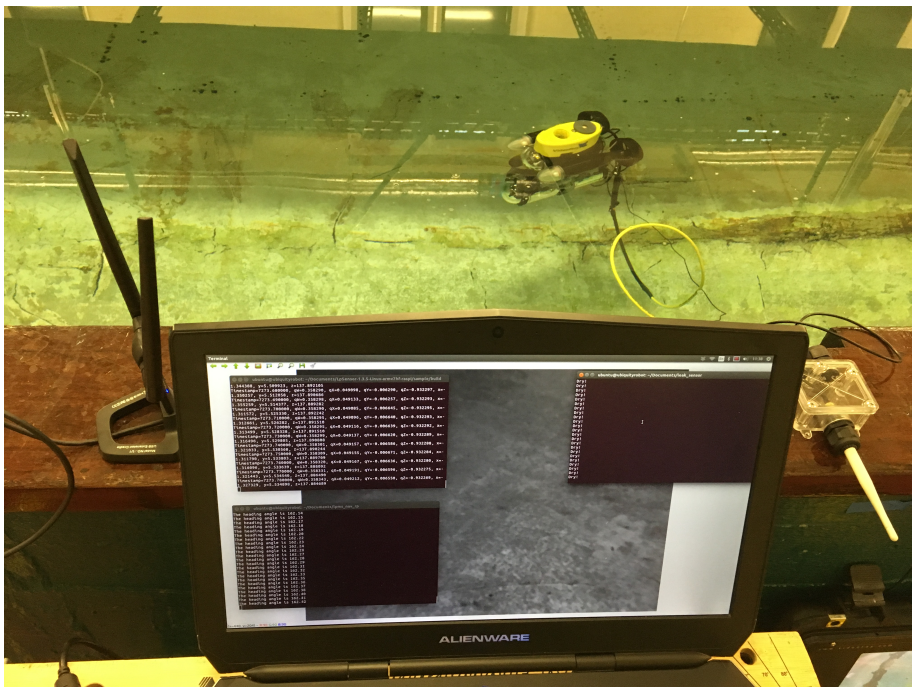


Figure 6.8 The Control System

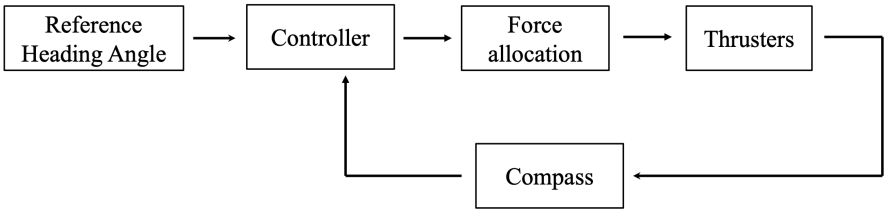


Figure 6.9 The Control Block Diagram

In order to investigate the robustness of controllers to changes in the hydrodynamic and dynamic models, the modified vehicle with the tube installed on the bottom was tested in addition to the original VideoRay Pro 3, as shown in Figure 3.4. In this way, the dynamic and hydrodynamic models of the vehicle have been changed, and the controllers have been tested with the same coefficients on the original and modified ROV.

The experimental tests were carried out in the towing tank at Newcastle University (Bowes, 2010), where quantified disturbances could be generated. The vehicle was submerged keeping the depth at about 0.3m. The controllers were initially evaluated without any disturbances and results are shown in Figure 6.10, and then waves were introduced, as shown in Figure 6.11. The height of the waves was $0.030m$ and the frequency was $1.0Hz$. Finally, larger waves were generated in the tank to verify the robustness of the four controllers. The height of the larger waves was $0.050m$ and the frequency was maintained at $1.0Hz$, as shown in Figure 6.12. In the experiments, the desired heading angle was 210^o. The ROV could make a turn from 360^o to 210^o, but in order to evaluate the heading controllers' performance clearly, the ROV started at 0^o.

Generally, waves only affect the motion of surface ships with submerged vehicles influenced by current. However, in shallow water, underwater vehicles would be affected by waves. Furthermore, the waves in the towing tank can effectively cause a current as well. These conditions could therefore be very close to the realistic conditions that an ROV might encounter. The analysis of the results is presented in Table 6.3.

For the tests without any disturbances, the model-based nonlinear controllers had fast responses, as shown in Table 6.3. Although the PID had a longer Settling Time, the Variance of Error of the PID was 12.94, the smallest value among the four controllers. In contrast, the FLC had a large Variance of Error and Overshoot, but the Settling Time of the FLC was shorter than that of the other controllers. This indicates that the linear controllers were much more stable but the nonlinear ones are more robust. For the proposed SMC, the linear part of the design can improve its performance in terms of the Variance of Error, and the nonlinear part results in a small Average Error and short Settling Time.

For the experiments with disturbances, as shown in Figure 6.11, and Figure 6.12, the results were analogous to the simulation results with the performances of the PID declining significantly, and the Average Error and Variance of Error continuously increasing. Conversely, the Variance of Error for the model-based nonlinear controllers increased slowly. The Settling Time remained almost at the same value. The Average Error was actually smaller than that in the disturbance-free situations. Because the waves increased the oscillation frequency of the ROV heading angle, the Average Error converged to zero and it can be seen that the model-based nonlinear controllers (BC and SMC) were more robust than the other controllers. For the the FLC, Settling Time remained similar, the Overshoot and Average Error increased significantly, but the Variance of Error increased significantly. This indicates that the FLC had large oscillations and, in fact, large overshoot and significant oscillations are common issues for intelligent controllers (Narendra and Mukhopadhyay, 1992).

The experiments with the tube, as shown in Figure 6.13, were used to show the robustness of these controllers. Because the water-tight tube is installed on the bottom of the vehicle, it impacts the hydrodynamic parameters dramatically. Meanwhile the parameters of the controllers remained the same as in the previous tests. Therefore, this test can indicate the robustness of these controllers to hydrodynamic variation. According to the results, the FLC and PID controllers failed in this test whereas the nonlinear state controllers were able to guide the vehicle to the desired heading angle successfully. In detail, the BC was able to make the vehicle turn to the desired angle more rapidly than the SMC, but the SMC is more stable than BC, due to the large oscillation.

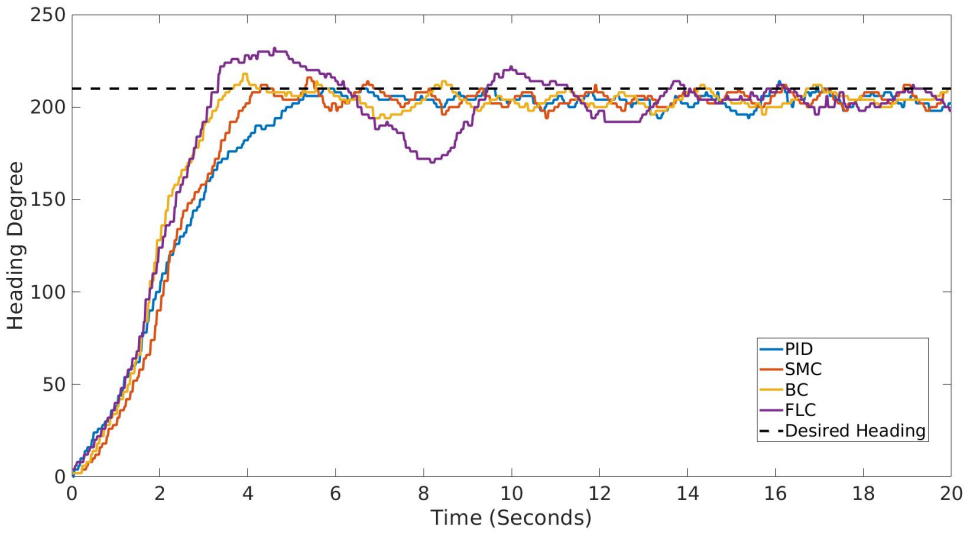


Figure 6.10 Experiment Without Disturbances, Heading Angle

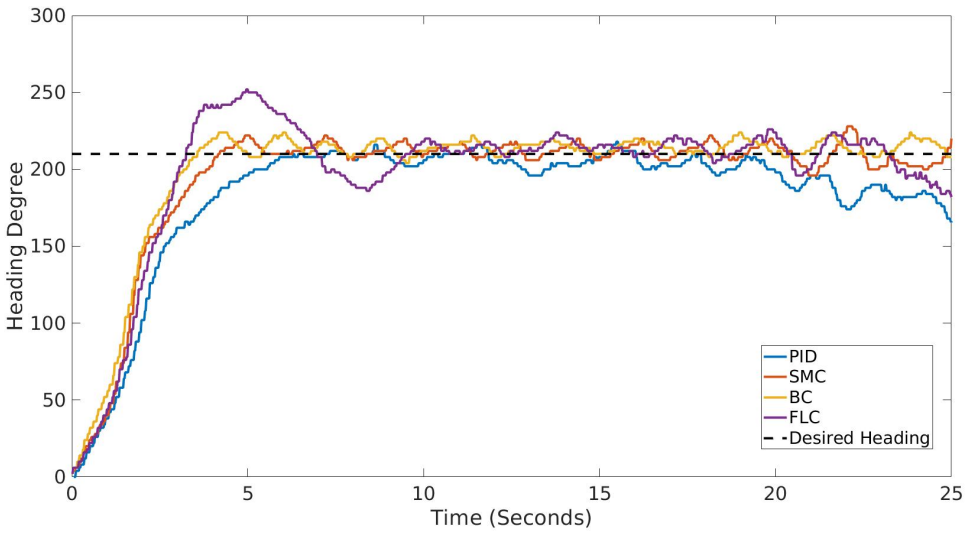


Figure 6.11 Simulation With Disturbances (Amplitude: 0.030 m, Frequency: 1.00 Hz), Heading Angle

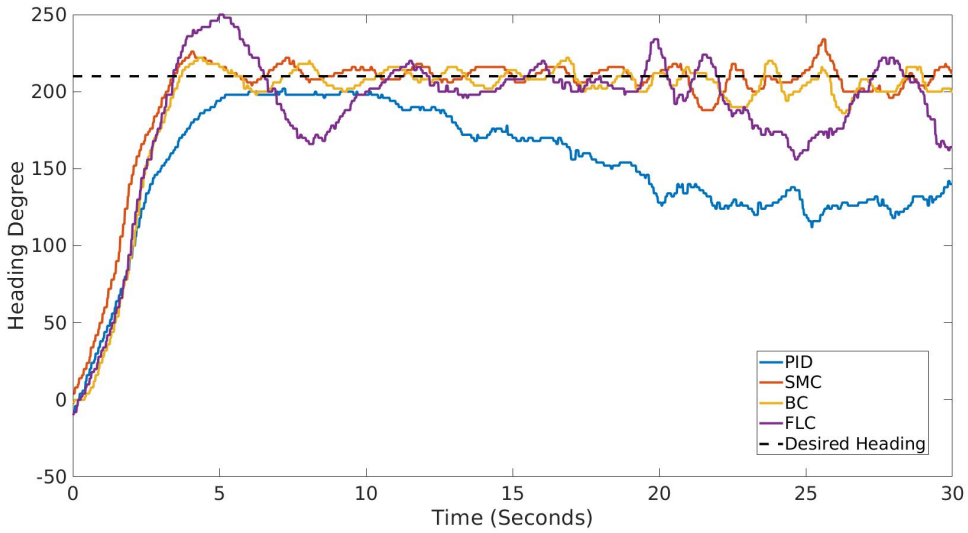


Figure 6.12 Experiment With Large Disturbances (Amplitude: 0.050 m, Frequency: 1.00 Hz), Heading Angle

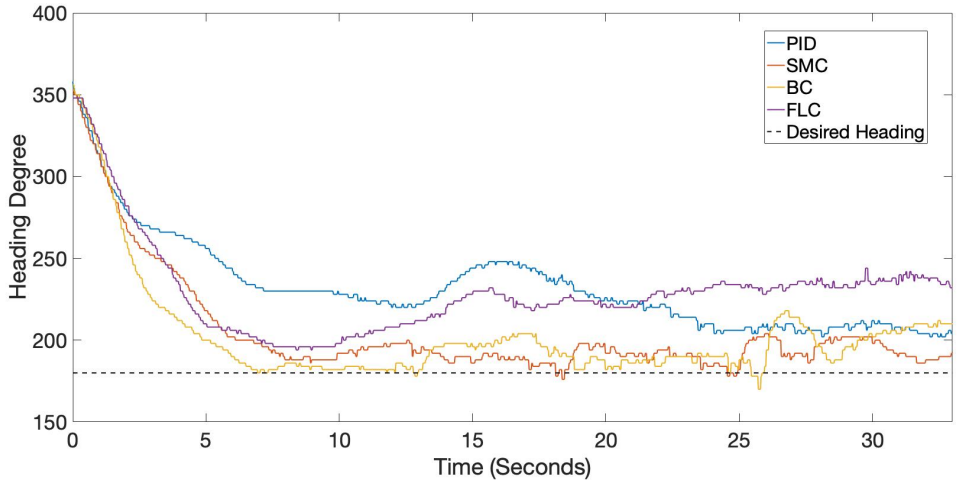


Figure 6.13 Experiments With Tube, Heading Angle

Situations	Properties	PID	SMC	BC	FLC
No Disturbances	Settling Time (s)	4.88	3.90	3.30	3.08
	Overshoot (^{o})	4	6	8	22
	Average Error (^{o})	6.03	5.20	5.54	5.03
	Variance of Error ($^{o^2}$)	12.94	14.60	19.65	179.21
Disturbances (Amplitude: 0.030 m Frequency: 1.00 Hz)	Settling Time (s)	5.18	3.82	3.18	3.06
	Overshoot (^{o})	8	18	14	42
	Average Error (^{o})	10.95	1.8287	4.00	3.72
	Variance of Error ($^{o^2}$)	156.81	36.18	22.28	208.89
Large Disturbances (Amplitude: 0.050 m Frequency: 1.00 Hz)	Settling Time (s)	6.00	3.18	3.44	3.40
	Overshoot (^{o})	0	24	12	40
	Average Error (^{o})	52.16	-0.40	2.70	8.89
	Variance of Error ($^{o^2}$)	833.67	51.83	54.27	428.07
Adding Tube	Settling Time (s)	Inf	7.96	6.08	Inf
	Overshoot (^{o})	0	0	0	0
	Average Error (^{o})	None	11.58	14.97	None
	Variance of Error ($^{o^2}$)	None	25.65	113.70	None

Table 6.3 Experimental Results Analysis

6.3. Discussion

Based on the results of the simulations and experimental evaluations, the features of different controllers are summarised as follows.

1. The linear controller (PID) has a long Settling Time and large Average Error, but a small Variance of Error. After disturbances were added, the errors increased rapidly. Therefore, the controller is more stable but less robust relative to the others. However, after the ROV model was changed, it failed to guide the vehicle to the reference heading angle. This shows that the PID is sensitive to hydrodynamic model variance. After the model was changed, the parameters in the PID controller needed to be upgraded.

2. The proposed Sliding Mode Controller consists of a linear controller and a Sliding Mode Controller. Due to the pole-placement methods used for the linear controller, it was more stable compared with the BC. Meanwhile, the Sliding Mode Control part enabled it to be more robust than the other controllers. It is also robust to model variance, because it was still able to control the vehicle to turn to the desired heading angle after the tube was added.

3. The Backstepping Controller is a model-based nonlinear controller, derived by using the Lyapunov Function. When the external disturbances were increased, the performance of

the BC was unchanged. Therefore, it is relatively robust. In addition, it has a small Overshoot compared with the SMC. Similar to the SMC, the BC continued to work well after the ROV model was changed. This demonstrates that the nonlinear controllers have greater robustness to model variance.

4. The Fuzzy Logic Controller is an intelligent controller. It is constructed through experts' knowledge, rather than mathematical modelling. Hence, it is relatively easy to understand and to get satisfactory performance from the logical operators. However, it has large oscillations. This drawback is common in intelligent controllers (Narendra and Mukhopadhyay, 1992). After the ROV modification, the FLC failed to control the vehicle, which indicates that the member function and rules need to be upgraded after the modification of the ROV.

6.4. Summary

The simulation and experiments have been presented to verify the controllers. In the section 6.3, the merits and shortcomings of each controller have been described. According to the results, the improved SMC, combining the advantages of linear control and nonlinear control, demonstrated a better performance compared to the other controllers. The control section of this thesis has now been completed. In order to implement the automation on UUVs, an underwater navigation system is essential. In the next part, visual navigation systems for UUVs will be discussed.

Part II

Visual Navigation Methodologies for UUVs

Chapter 7. Camera Models & Underwater Calibration

In this chapter, the mathematical models for a generic camera and a camera with fisheye lens are described. In addition, the underwater calibration for the cameras is presented in this chapter.

7.1. Camera Modelling

7.1.1. Perspective Camera Model

A Charge-Coupled Device (CCD) optical sensor with general lenses can be described by a perspective camera model (also known as a pinhole camera model). The pixel coordinates are shown in Figure 7.1.

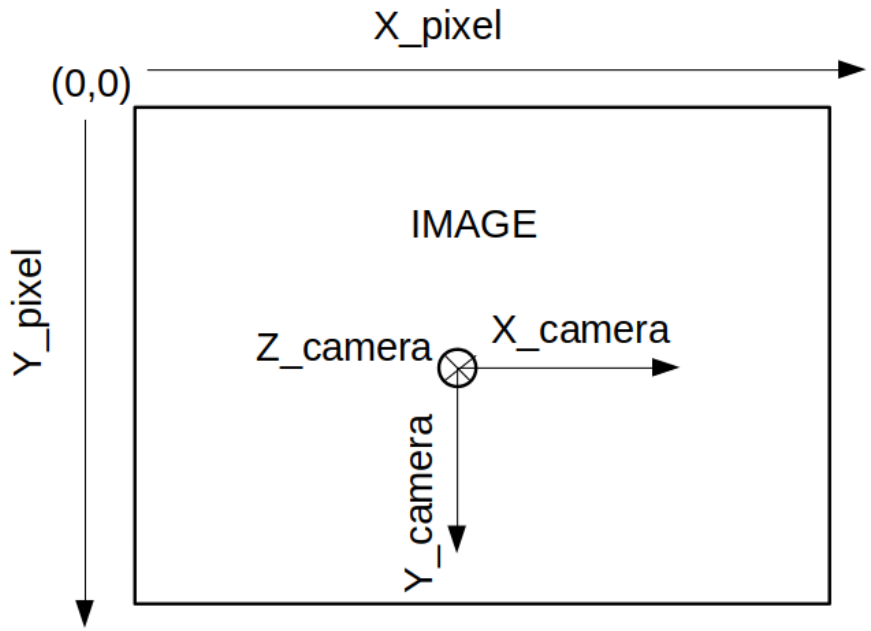


Figure 7.1 Pinhole Camera Model

The intrinsic matrix for the camera $\boldsymbol{K}$ is,

$$\boldsymbol{K} = \begin{bmatrix} f_x & 0 & p_x \\ 0 & f_y & p_y \\ 0 & 0 & 1 \end{bmatrix} \tag{7.1}$$

Furthermore, $\boldsymbol{K}$ is a map from the camera coordinates $\mathbb{R}^3$ to pixel coordinates $\mathbb{R}^2$, which is,

$$\boldsymbol{X}_p = [\boldsymbol{K}|\boldsymbol{0}]\boldsymbol{X}_c. \tag{7.2}$$

Here, $\boldsymbol{X}_p = [x_p,\ y_p,\ 1]$, and $\boldsymbol{X}_c = [x_c, y_c, z_c, 1]^T$. Equation (7.2) can be rewritten as:

$$x_p = \frac{f_x x_c}{z_c} + p_x; \tag{7.3}$$

$$y_p = \frac{f_y y_c}{z_c} + p_y. \tag{7.4}$$

From pixel coordinate to camera coordinate, there is a map as follows,

$$\frac{x_c}{z_c} = \frac{(x_p - p_x)}{f_x}; \tag{7.5}$$

$$\frac{y_c}{z_c} = \frac{(y_p - p_y)}{f_y}. \tag{7.6}$$

$\boldsymbol{X}_p$ to $\boldsymbol{X}_c$ can be written as,

$$\boldsymbol{X}_c(\lambda) = \lambda \boldsymbol{K}^{-1}\boldsymbol{X}_p + \boldsymbol{C}. \tag{7.7}$$

Here, $\boldsymbol{X}_c = [\lambda x_c/z_c,\ \lambda y_c/z_c,\ \lambda]^T$, and $\boldsymbol{X}_p = [x_p,\ y_p,\ 1]$. $\boldsymbol{C}$ is the position of the camera centre. Hence, in order to get the real position of points, the one parameter λ should be equal to depth z_c, which is the distance from the feature points to the camera. In this case, $\boldsymbol{C}$ can be described as,

$$\boldsymbol{C} = [\ 0,\ 0,\ 0,\ 1]^T. \tag{7.8}$$

The camera calibration is a very important consideration. It is used to estimate the parameters of a lens and image sensors. These parameters play crucial roles in visual localisation and depth estimation.

In addition to the intrinsic parameters, the identification of coefficients for the distortion correction function is conducted as part of the calibration process. Generally, the coefficients in the distortion correction map $L(r)$ are positive such that $L(0) = 1$. To be linearised, the nonlinear $L(r_d)$ is approximated by using Taylor's theorem: $L(r) = 1 + k_1 r + k_2 r_d^2 + k_3 r_d^3 + ...$ (Hartley and Zisserman, 2003). The distortion is processed in the image plane, and each pixel is corrected by,

$$\hat{x} = x_c + L(r_d)(x - x_c) \tag{7.9}$$

$$\hat{y} = y_c + L(r_d)(y - y_c) \tag{7.10}$$

Here, $(\hat{x}, \hat{y})$ is the corrected pixel coordinate. The coordinate (x, y) is the measured pixel coordinate, the coordinate (x_c, y_c) is the centre of the image plane, and $r_d^2 = (x - x_c)^2 + (y - y_c)^2$.

In the OpenCV model, there are radial and tangential factors for the distortion model. The radial factor can be written as,

$$x_{corrected} = x(1 + k_1 r_d^2 + k_2 r_d^4 + k_3 r_d^6) \tag{7.11}$$

$$y_{corrected} = y(1 + k_1 r_d^2 + k_2 r_d^4 + k_3 r_d^6) \tag{7.12}$$

The other problem is that camera lenses are usually not exactly parallel to the image plane. Hence, a tangential distortion correction is required. The formula of the distortion map is:

$$x_{corrected} = x + [2p_1 xy + p_2(r_d^2 + 2x^2)]; \tag{7.13}$$

$$y_{corrected} = y + [p_1(r_d^2 + 2y^2) + 2p_2 xy]. \tag{7.14}$$

In OpenCV, five coefficients are usually required to complete the distortion process, as

$$Distortion_{coefficients} = (k_1, k_2, p_1, p_2, k_3). \tag{7.15}$$

While the perspective camera model with the general distortion models is compatible with wide-angle or narrow-angle lenses (Kannala and Brandt, 2006), it is not applicable to the fisheye model and the omnidirectional camera model. In sections 7.2 and 7.3, these two models are discussed.

7.2. Fisheye Model

The main difference between the perspective or pinhole camera and a fisheye camera model is the distortion part. The perspective coordinates of P is $[a,b]^T$, where

$$a = X/Z, b = Y/Z; \tag{7.16}$$

$$r^2 = a^2 + b^2; \tag{7.17}$$

$$\theta = atan(r). \tag{7.18}$$

Fisheye distortion is described by,

$$\theta_d = \theta(1 + k_1\theta^2 + k_2\theta^4 + k_3\theta^6 + k_4\theta^8). \tag{7.19}$$

The distortion point coordinates are $[x', y']^T$ where

$$x' = (\theta_d/r)a; \tag{7.20}$$

$$y' = (\theta_d/r)b. \tag{7.21}$$

The distortion point $[u,v]^T$ is converted into the image plane coordinate, described as,

$$u = f_x(x' + \alpha y') + c_x; \tag{7.22}$$

$$v = f_y y' + c_y. \tag{7.23}$$

7.3. Omnidirectional Camera Model

The unified projection model of an omnidirectional camera is based on Mei and Rives's work (Mei and Rives, 2007). The point $\boldsymbol{m} = (x,y,1), \boldsymbol{m} \in R^3$ is on the image plane, and the point $\boldsymbol{p} = (X,Y,Z), \boldsymbol{p} \in R^3$ is in the unit sphere, and there is a nonlinear transform h to map $\boldsymbol{m} \rightarrow \boldsymbol{p}$:

$$h(\boldsymbol{m}) = \begin{bmatrix} \frac{\xi+\sqrt{1+(1-\xi^2)(x^2+y^2)}}{x^2+y^2+1}x \\ \frac{\xi+\sqrt{1+(1-\xi^2)(x^2+y^2)}}{x^2+y^2+1}y \\ \frac{\xi+\sqrt{1+(1-\xi^2)(x^2+y^2)}}{x^2+y^2+1} - \xi \end{bmatrix}. \tag{7.24}$$

Because $h(\boldsymbol{m})$ is in homogeneous coordinates, equation (7.24) can be simplied as,

$$h(\boldsymbol{m}) \sim \begin{bmatrix} x \\ y \\ g(x,y) \end{bmatrix}, \tag{7.25}$$

where,

$$g(x,y) = 1 - \xi \frac{x^2+y^2+1}{\xi + \sqrt{1+(1-\xi^2)(x^2+y^2)}}. \tag{7.26}$$

7.4. Calibration Methods

Generally, the calibration problem is solved by minimising the cost function (Mei and Rives, 2007):

$$E = \frac{1}{2}\sum_{i=1}^{m}[H(\boldsymbol{\Theta}, g_i) - p_i]^2, \tag{7.27}$$

Here, a grid based on a chessboard consists of m points of g_i coordinates with corresponding values p_i. The term H is a function mapping grid points into the image plane and $\boldsymbol{\Theta}$ involves intrinsic, distortion, and extrinsic parameters.

The Levenberg Marquadt approach can be applied to minimise the cost function (7.27), solving for $\boldsymbol{\Theta}$. In order to minimise the cost function effectively, the initial estimates of camera parameters are processed by Direct Linear Translation (DLT) (Hartley and Zisserman, 2003).

In practice, the camera calibration is processed by OpenCV tools, which are developed based on (Wang and Lee, 2003). Camera calibrations in underwater environments are introduced in the following sections.

7.5. Underwater Calibration

In this work, a classic black-white chessboard was placed on the bottom of the towing tank, as shown in Figure 7.2.

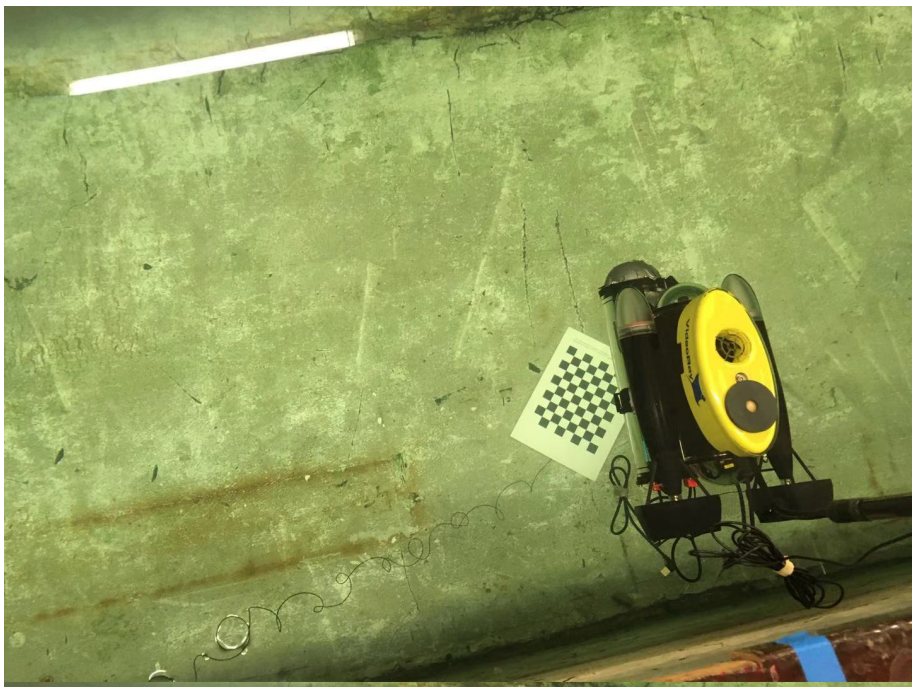


Figure 7.2 Chessboard on Ground

The vehicle was operated remotely to obtain images of the chessboard from different views. A sample of the pictures captured by the camera is shown in Figure 7.3.

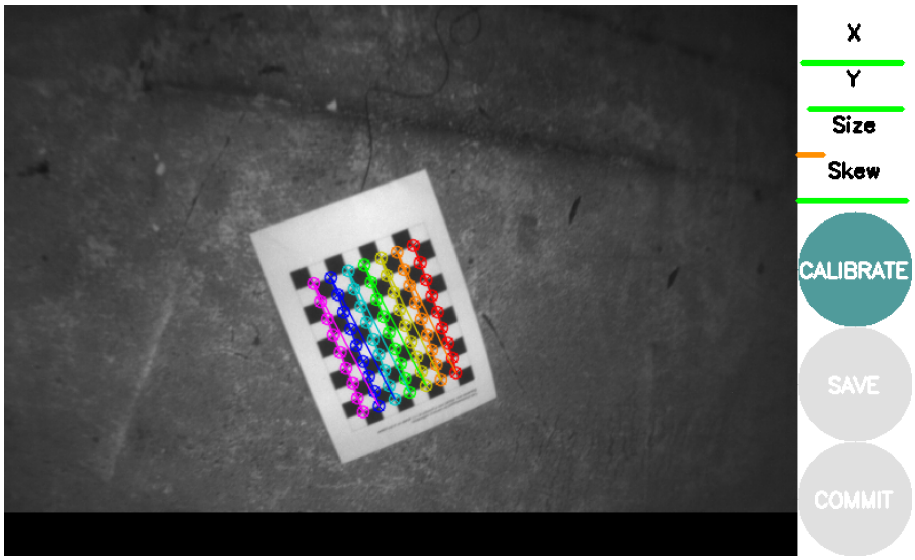


Figure 7.3 Calibration Samples

7.5.1. Calibration Results for Monocular Camera

A wide-angle lens has been mounted on the Monocular Camera mvBlueFOX-MLC 200w. Hence, the perspective model is sufficient to describe its performance. Applying the OpenCV calibration tool, the results are shown in Figure 7.4.

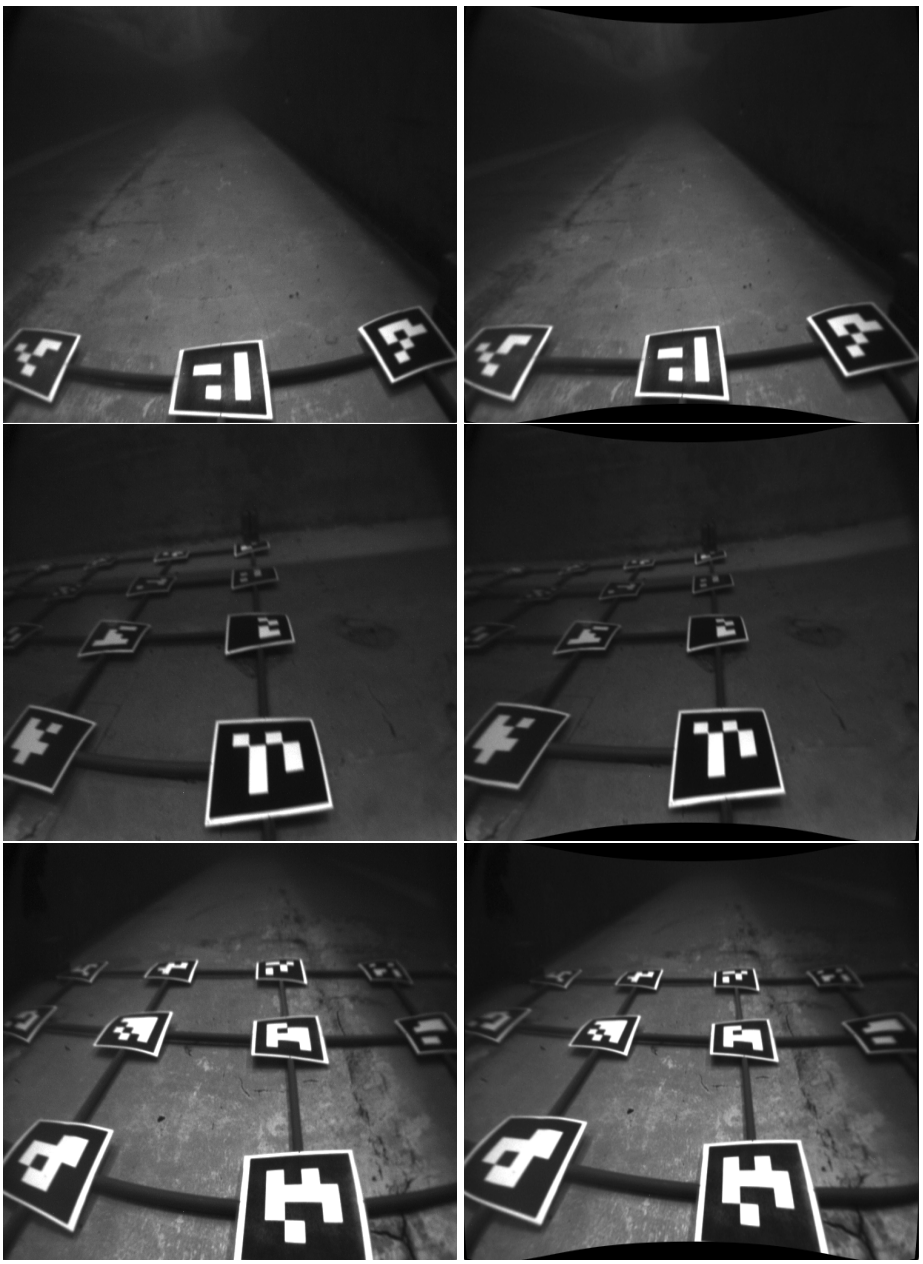


Figure 7.4 Perspective Camera Model for Monocular Camera. Images on the left side are original, while undistorted images are on the right side.

7.5.2. Calibration Results for T265 Stereo Camera

Unlike the monocular camera, the T265 tracking camera has a fisheye lens, which can enlarge the view of the camera, but distorts the image significantly. Hence, after obtaining the camera parameters, different camera models (the fisheye model and the omnidirectional camera model) were tested to undistort the raw images. Figure 7.5 shows a comparison between raw images and undistorted images from the omnidirectional camera model. Figure 7.6 shows the comparative results using the fisheye model. Figure 7.7 shows the comparative results using the perspective camera model.

Figures 7.5 to 7.7 all show the pixels close to the image centre as undistorted compared to the pixels at the edges of the images. In Figure 7.7, the results from the perpective camera model seem to be over undistorted. The results from the fisheye model and the omnidirectional model are similar but the pixels at the edges of the images were undistorted more successfully in the results from the fisheye camera model.

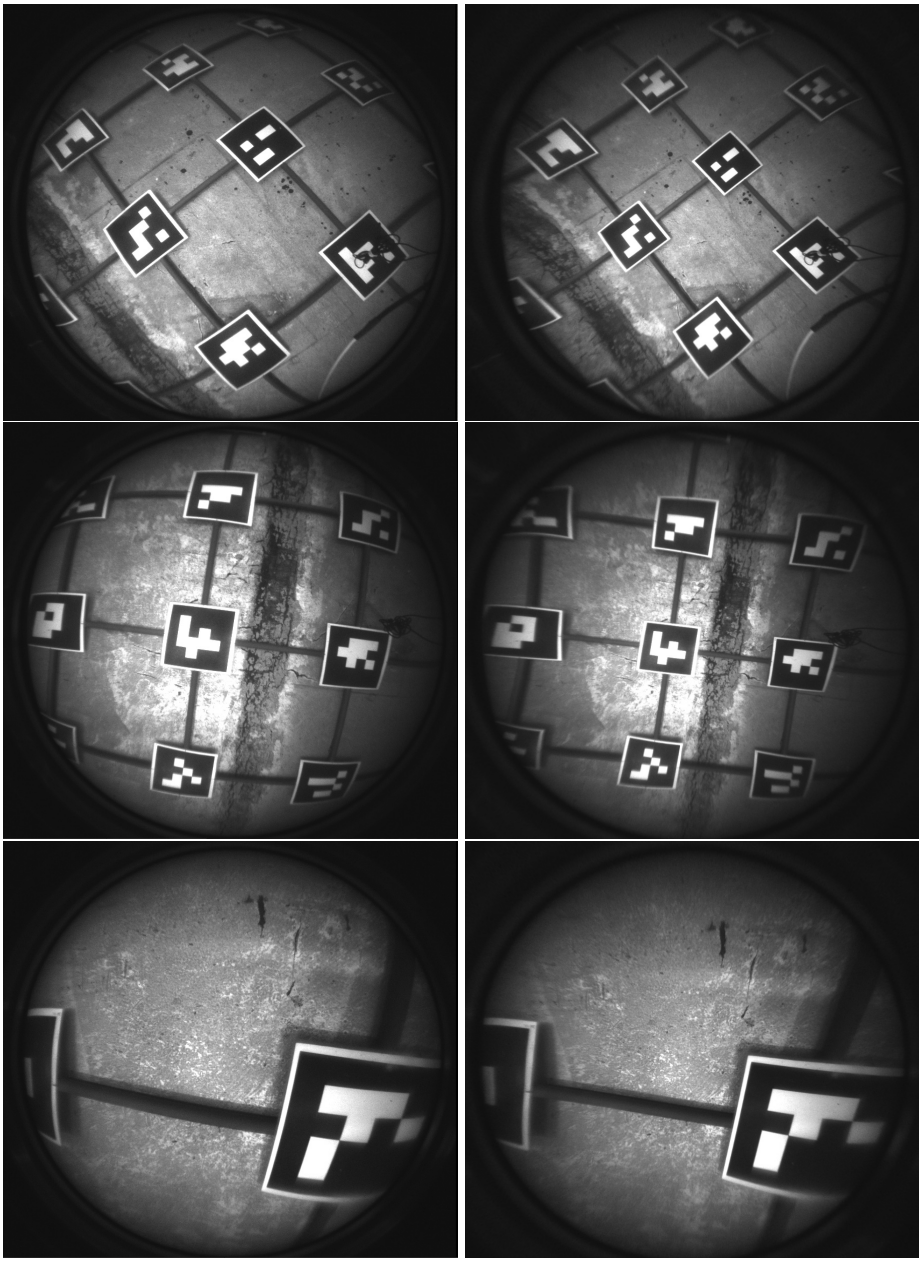


Figure 7.5 Omnidirectional Camera Model for T265. Images on the left side are original, while undistorted images are on the right side.

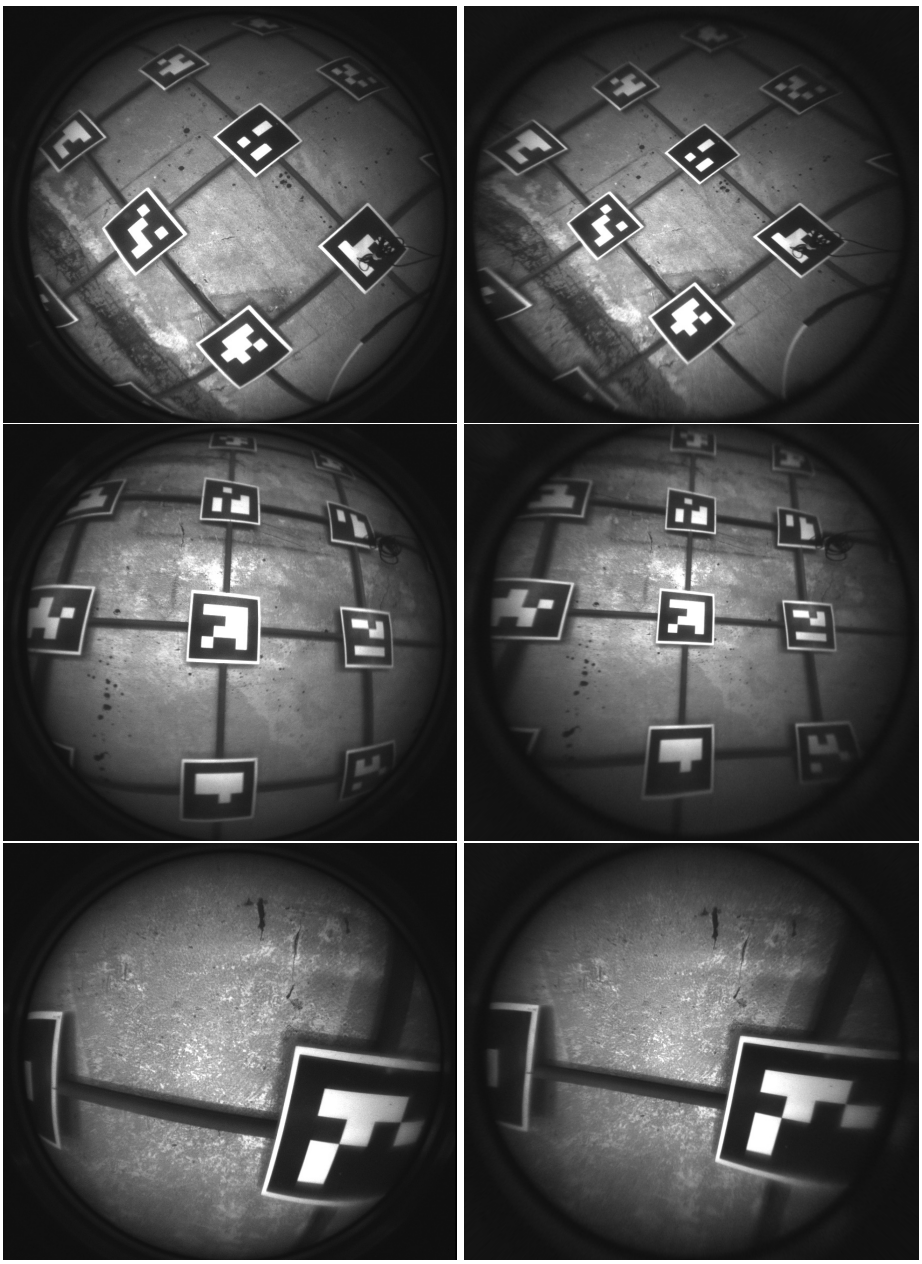


Figure 7.6 Fisheye Camera Model for T265. Images on the left side are original, while undistorted images are on the right side.

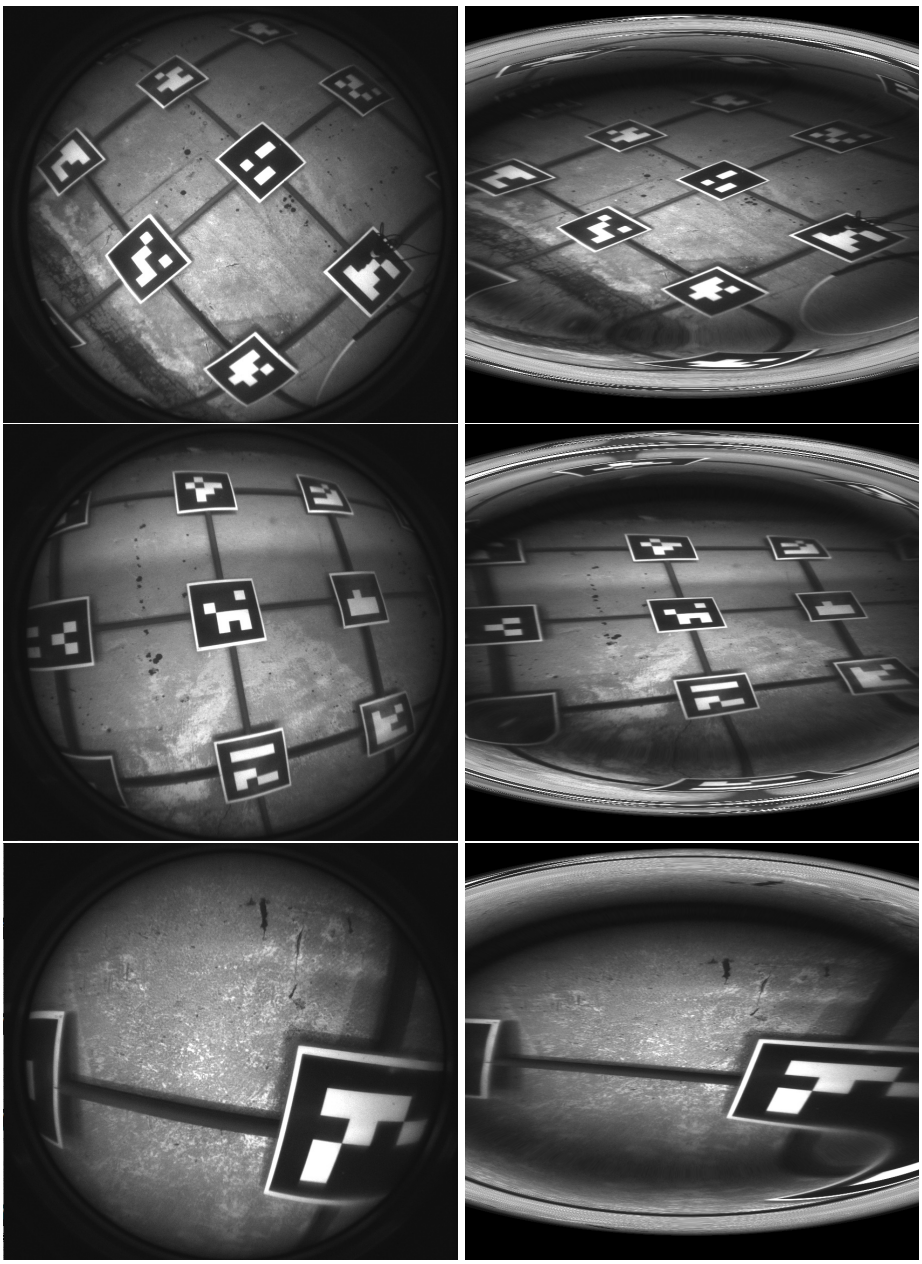


Figure 7.7 Perspective Camera Model for T265. Images on the left side are original, while undistorted images are on the right side.

7.6. Summary

The camera modelling and calibration have been described in this chapter. During the calibration, the cameras mounted on the VideoRay Pro 3 captured patterns in the underwater environment. The tools from OpenCV libraries were utilised to output camera matrices and distortion coefficients. The monocular camera with wide-angle lens was calibrated by using a generic perspective model. The T265 tracking camera has a fisheye lens, which makes the situation complex. After making a comparison of different calibration models, the results show that the fisheye model is suitable for the T265 camera. Hence, in Chapter 9, the perspective

model of the camera will be used to calibrate the monocular camera. In Chapter 10, the fisheye model will be utilised to calibrate the T265 tracking camera.

Chapter 8. ArUco Underwater Navigation

This chapter describes a low cost underwater navigation system developed based on the ArUco Marker Detection algorithm. The ArUco Marker system belongs to the fiducial marker approach, developed by Garrido-Jurado. It can offer a solution for augmented reality and robot localisation, based on square markers (Garrido-Jurado et al., 2014). In the paper, the authors firstly presented a general method for producing a marker dictionary (a set of ArUco Markers). The generating method is used to find the maximum inter-marker distance and the number of bit transitions. Then, there is a method for automatically detecting markers in images and correcting possible errors. An approach of combining multiple markers and an occlusion mask was also provided in the paper. The flow block of how to recognise the markers is described in Figure 8.1.

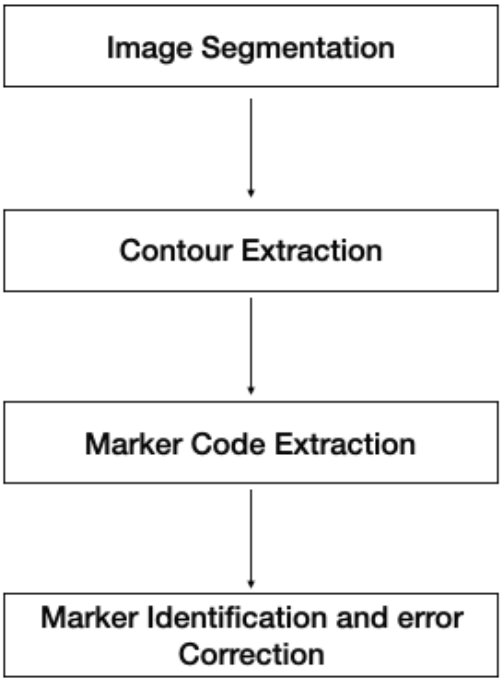


Figure 8.1 Steps of Recognition of ArUco Markers

8.1. Implementation and Methodology

The ArUco libraries from OpenCV are implemented in this navigation method. By calling the *estimatedPoseSingleMarkers* function in the libraries, the rotation vector and translational vector

of the marker relative to the camera can be obtained. With the Rodrigues' formula (Mebius, 2007), the rotation vector is converted to the associated rotation matrix. Hence, the transformation matrix ${}^{camera}\boldsymbol{T}_{marker}$ is derived.

${}^{marker}\boldsymbol{T}_{camera}$ is needed to obtain the position of the camera in a marker's coordinate system. It can be derived as,

$$ {}^{marker}\boldsymbol{T}_{camera} = {}^{camera}\boldsymbol{T}_{marker}^{-1}. \tag{8.1} $$

Assuming,

$$ {}^{marker}\boldsymbol{T}_{camera} = \begin{bmatrix} \boldsymbol{R}_{mc} & \boldsymbol{t}_{mc} \\ \boldsymbol{0} & 1 \end{bmatrix} \tag{8.2} $$

Then,

$$ {}^{marker}\boldsymbol{T}_{camera} = \begin{bmatrix} \boldsymbol{R}_{mc}^{T} & -\boldsymbol{R}^{T}\boldsymbol{t}_{mc} \\ \boldsymbol{0} & 1 \end{bmatrix}. \tag{8.3} $$

where, $\boldsymbol{u}_m = -\boldsymbol{R}_{mc}^{T}\boldsymbol{t}_{mc}$ is the translational vector of the camera relative to the marker. The term $\boldsymbol{R}_{mc}^{T}$ is the rotation matrix of the camera relative to the marker.

In the experiment, the matrix of markers was set out on the bottom of the towing tank. The physical layout of the markers is shown in Figure 8.2, and the reference of the markers is plotted in Figure 8.3.

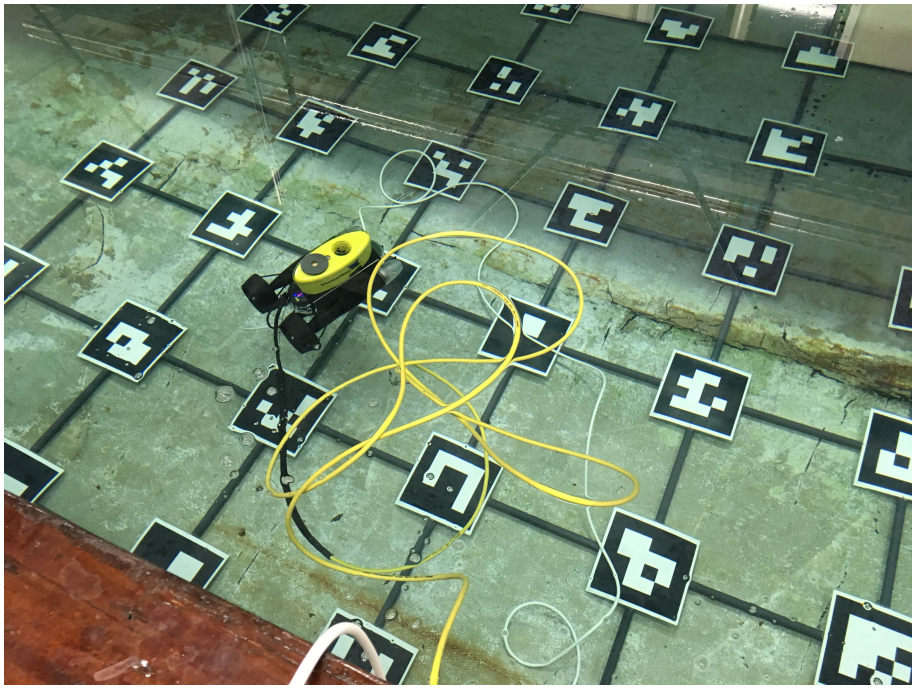


Figure 8.2 Markers in the Towing Tank

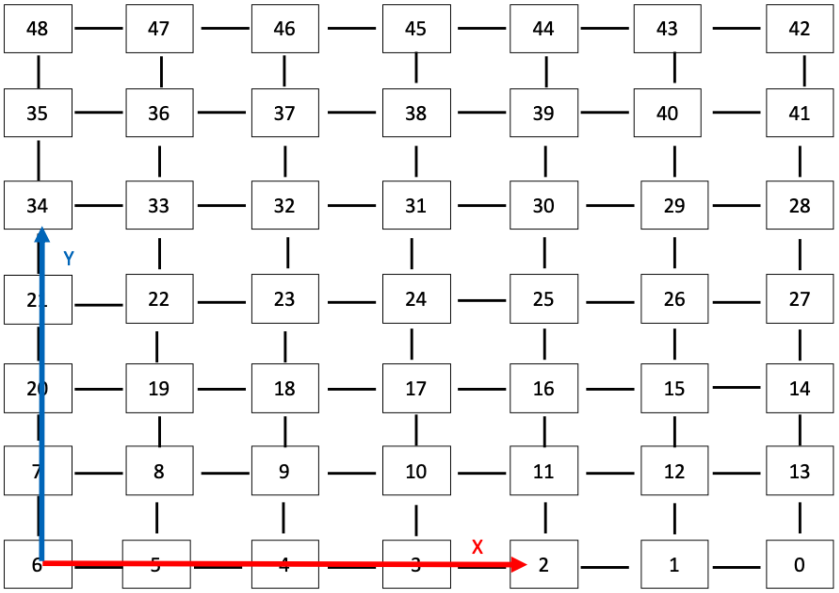


Figure 8.3 Reference of the Markers

According to the figures, the origin of the global coordinate system is located at the Id.6 marker.

After obtaining the relative pose to a specific marker, the transfer function (Tf) is derived as,

$$Id = Tf(r,c) = 6r + (-1)^r(6-c) \tag{8.4}$$

Where, Id is the unique id of the marker. Symbols r and c are the rth row and cth column respectively, and the arrangement of rows and columns is shown in Figure 8.4.

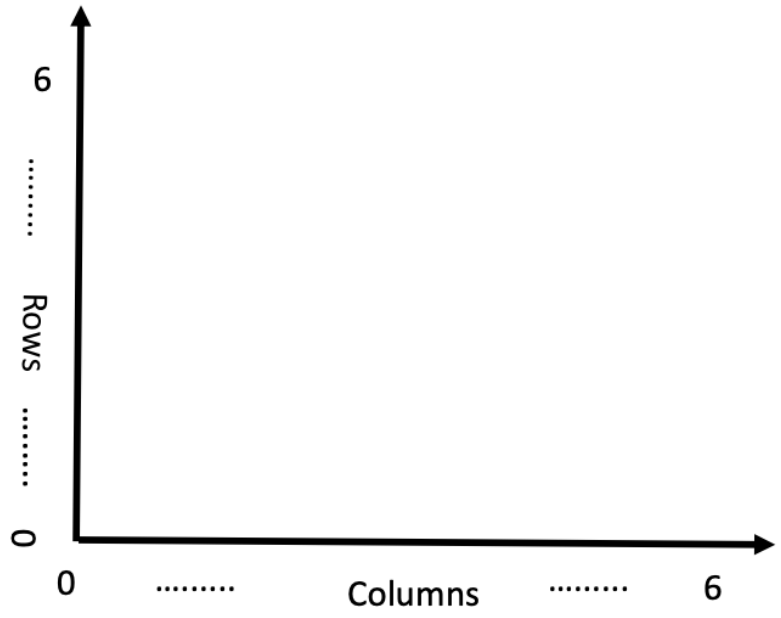


Figure 8.4 Rows and Columns

The inverse function of Tf can be used to get r and c, $Id \rightarrow (r,c)$ with Tf^{-1}. The inverse function Tf^{-1} can be derived by the following two equations,

$$r = [\frac{Id}{6}]; \tag{8.5}$$

$$c = (-1)^{r+1}(Id - 6r) + 6. \tag{8.6}$$

Where, [] indicates the integer division operator. Using these equations, the (c,r) is calculated. Because the distance between markers on the grid is 0.5 metres, the position of the marker in the global coordinate system is known by multiplying the constant scalar 0.5 by c and r as follows,

$$x_m = 0.5c; \tag{8.7}$$

$$y_m = 0.5r. \tag{8.8}$$

The camera pose relative to the marker ($\boldsymbol{u}_m$) with the position of marker in the global reference $(x_m, y_m, 0)$ can be transferred to $\boldsymbol{u}_g$ in the global coordinate system. Because all markers are parallel, the rotation matrix remains the same. Hence, the derivation of the camera pose in the global reference frame is completed.

8.1.1. Robustness of the Proposed Method

The ArUco algorithm can detect multiple markers and estimate the relative poses simultaneously. The estimation from the nearest marker is selected because the position error will increase as the distance increases.

There are other robust methods, for example: the well-known RANSAC (Random Sample Consensus), and the weighted average method. However, they can both cause errors in estimated poses. In general, the nearest marker to the camera can provide accurate pose estimation in its coordinate system. The RANSAC method selects one marker randomly to estimate pose. However the marker may be at a significant distance from the camera with large error, and the estimated pose from the marker may be close to the poses estimated from other markers which are also at a distance. In this case, the RANSAC method will fail to find the optimal pose. The weighted average method can be used to reduce the error caused by a white noise. However, the error in the ArUco method is caused by a steadily growing noise as the distance to the markers increases, rather than by white noise. Because of that, the weighted average method is unsuccessful in reducing error in this navigation method.

8.1.2. Data Collection

The modified Videoray Pro 3 ROV was used to collect the images in the towing tank. The Intel T265 Tracking camera presented in Chapter 3 is a stereo camera, and only the left side images are used in this algorithm.

During the collection process, the vehicle was operated remotely by a controller to follow specific paths.

8.2. Experimental Results and Discussion

The navigation system provides the 3D trajectory in absolute scale, as shown in Figure 8.5.

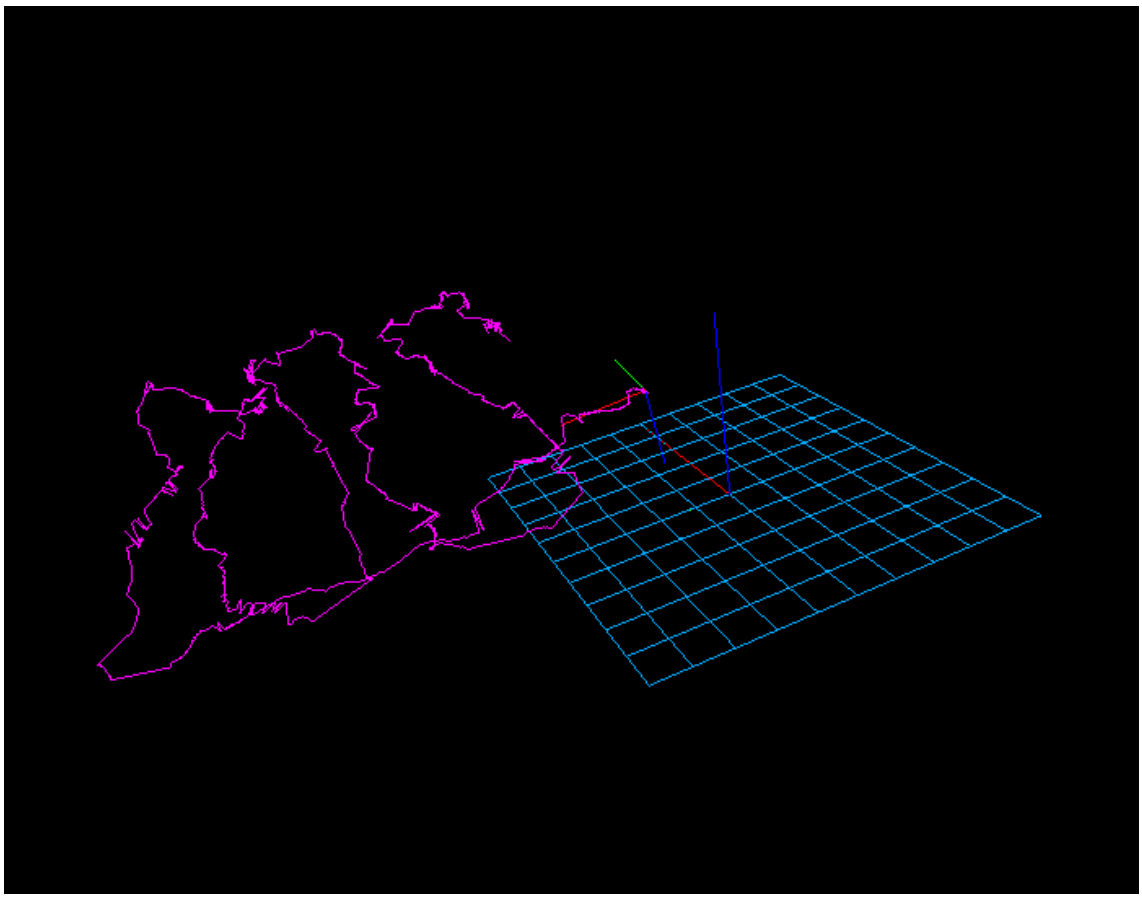


Figure 8.5 3D Trajectory with ArUco Navigation System

Because the vehicle was controlled to mainly move in the horizontal plane, the 3D trajectories were converted into the 2D plots, as shown in Figure 8.6 to Figure 8.11.

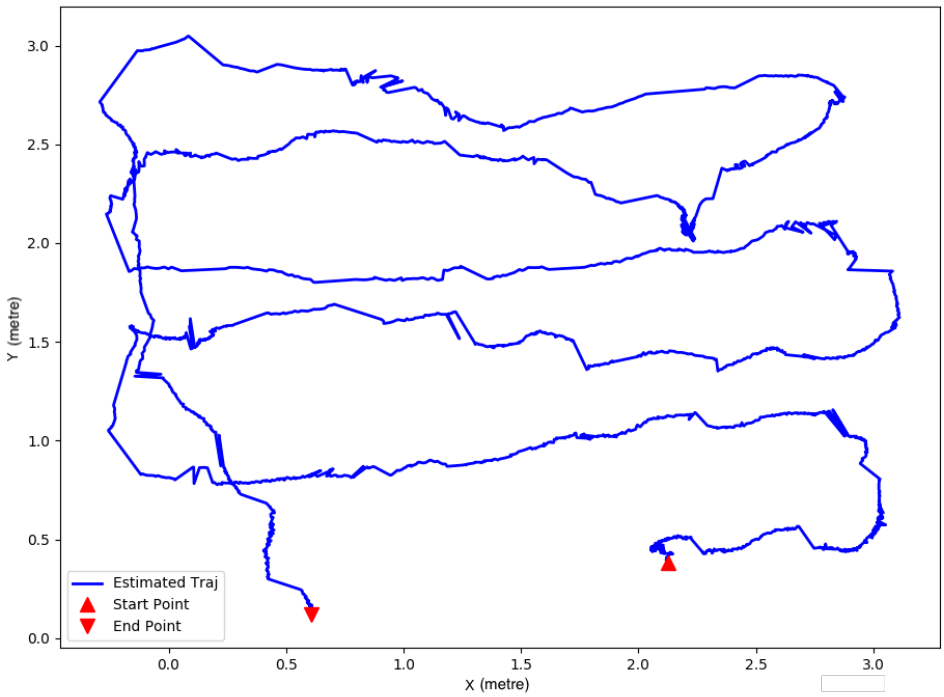


Figure 8.6 Lawnmower Shape 1st

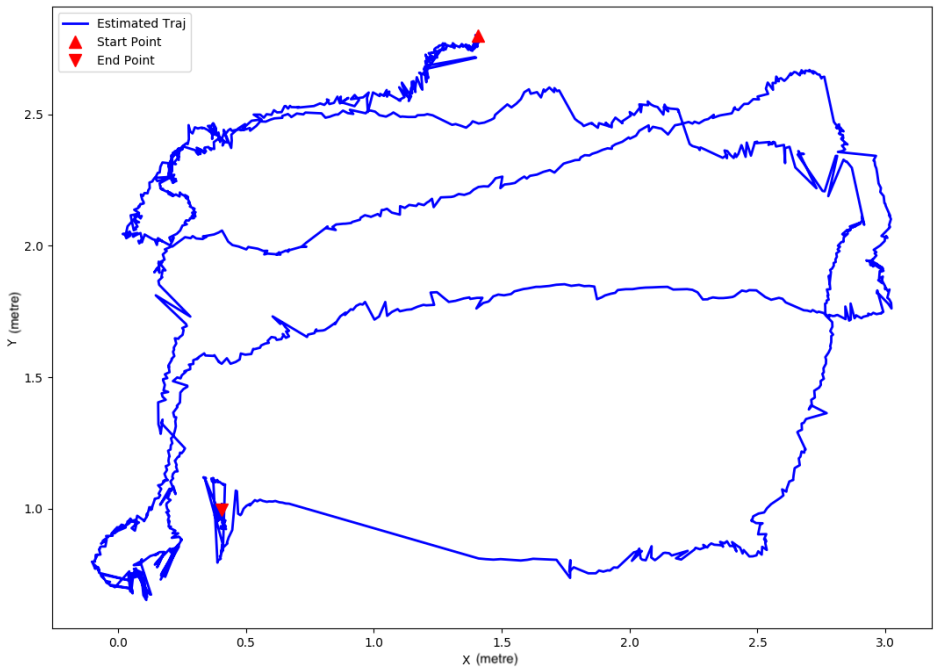


Figure 8.7 Lawnmower Shape 2nd

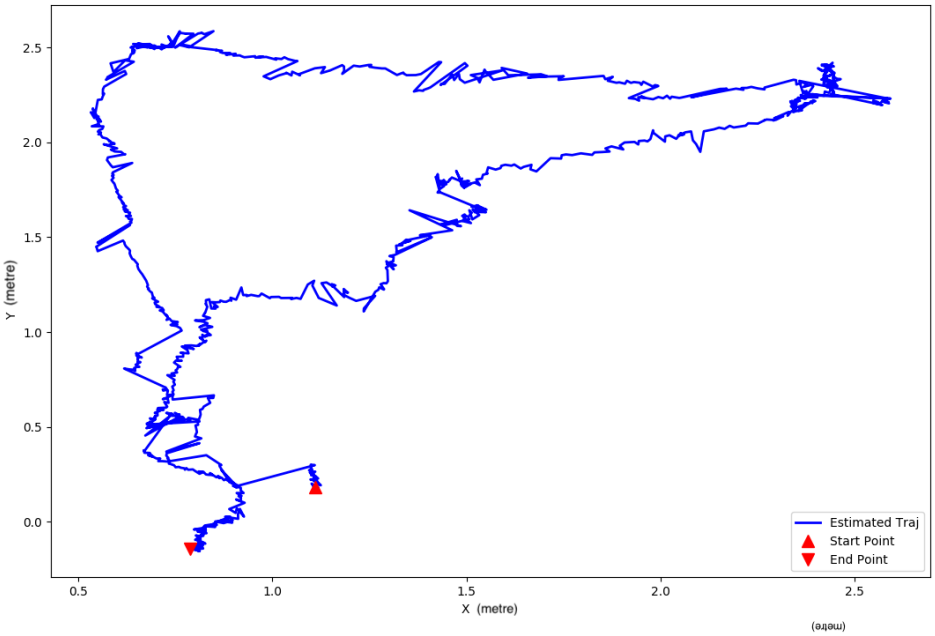


Figure 8.8 Loop Trajectory 1st

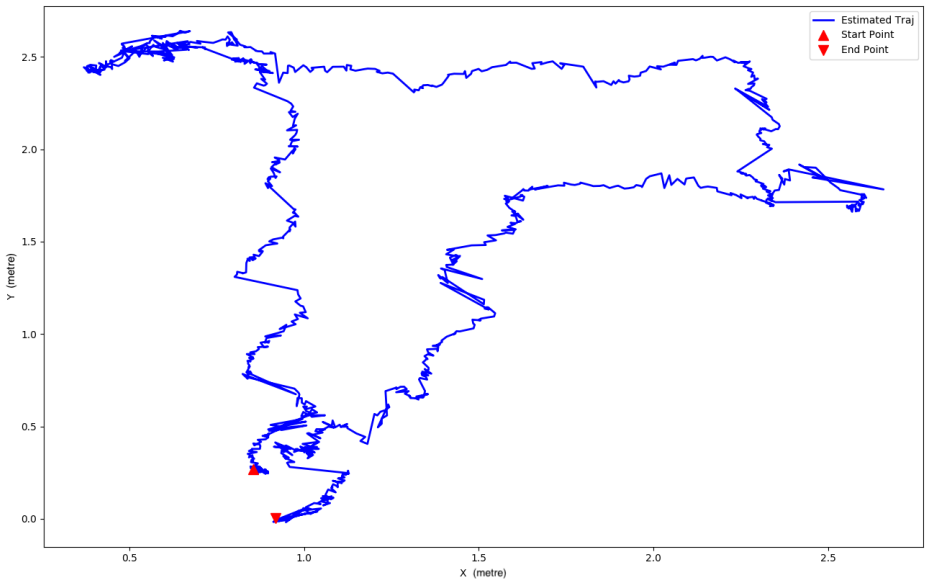


Figure 8.9 Loop Trajectory 2nd

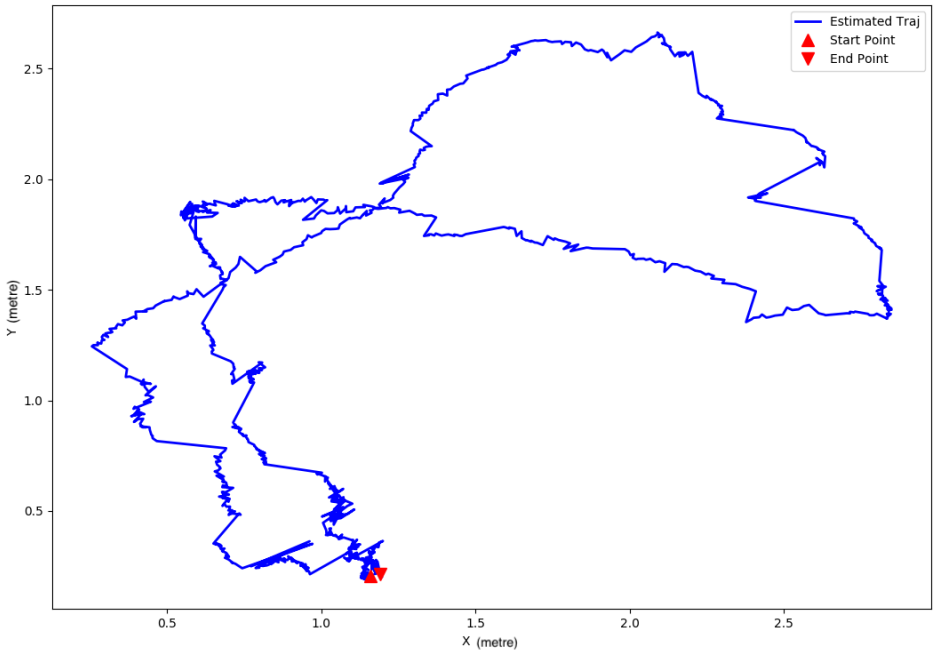


Figure 8.10 Loop Trajectory 3rd

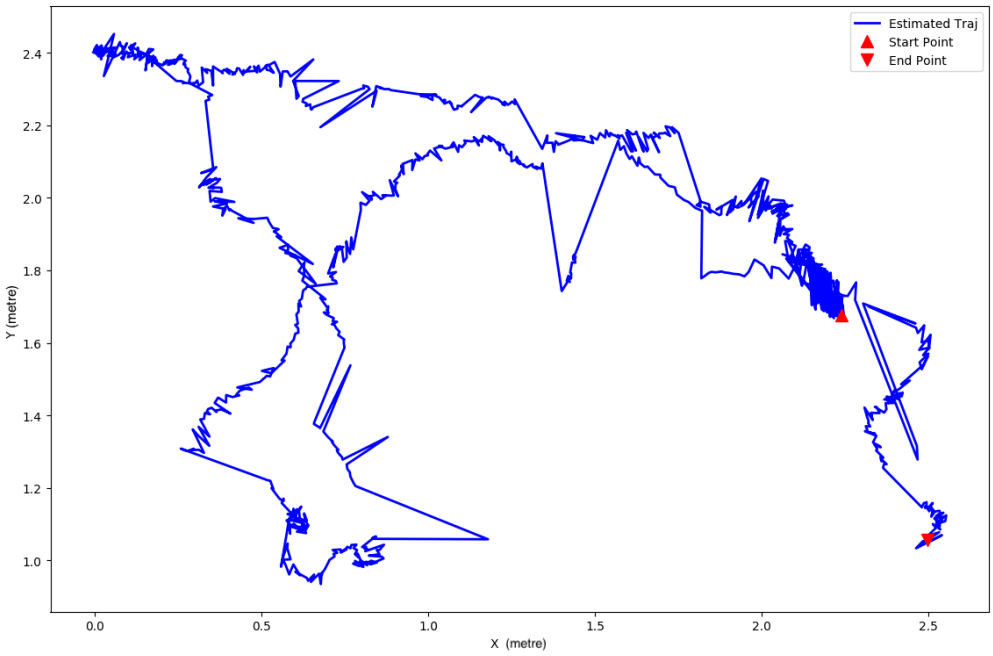


Figure 8.11 Random Trajectory

Due to a lack acoustic localisation devices, there are no reference trajectories to evaluate the performances of the proposed navigation method. In Figures 8.6 and 8.7, the vehicle was

operated to travel along an approximated lawnmower path. The method presents reasonable estimated trajectories. In Figure 8.8 to Figure 8.10, the vehicle was controlled to complete a closed loop. In Figure 8.8 and 8.9, the estimated trajectories are almost closed. The offsets may be caused by operation error controlling the ROV or by hydrodynamic disturbances (waves, current). Finally, the vehicle was controlled to complete two random journeys. The results are shown at Figure 8.11. In the following section, a quantitative evaluation method is introduced.

8.2.1. Quantitative Evaluation Method

Accuracy tests for the ArUco algorithm were conducted out of the water first. The results are listed in Table 8.1 and Table 8.2. X_r, Y_r, Z_r are the real coordinates relative to the markers, while X_m, Y_m, and Z_m are the measured coordinates. According to these results, the ArUco algorithm can measure the camera position with high accuracy, within a half-metre distance range. In these tests, the marker was fixed, and the camera was moved on the scale chart. They are shown at Figure 8.12.

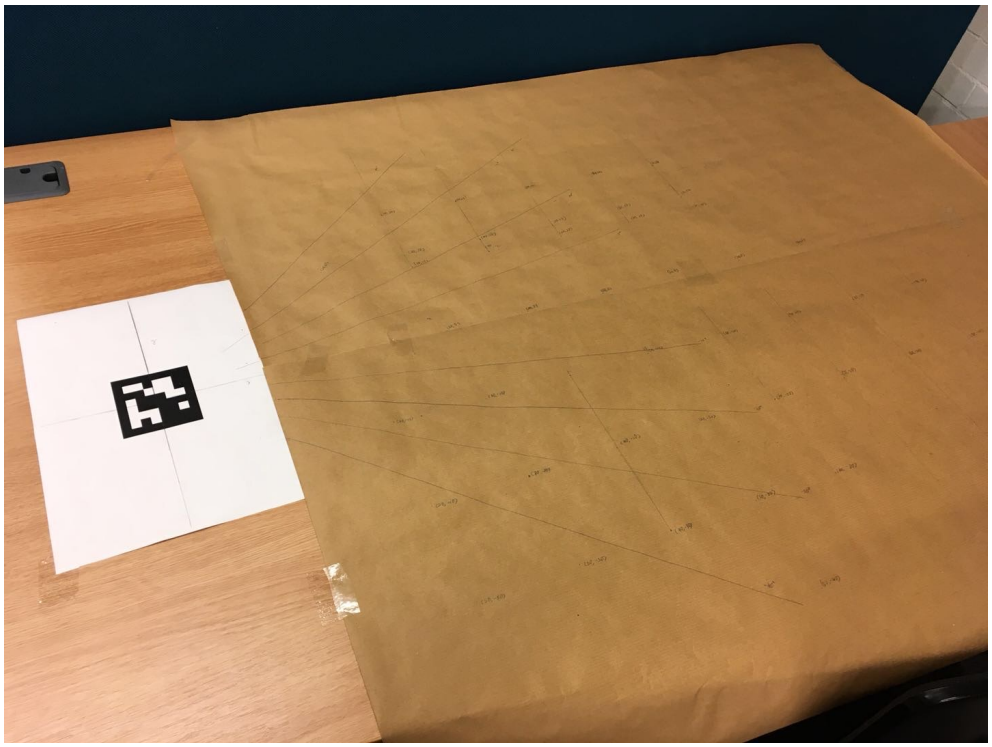


Figure 8.12 The Marker and The Scale Chart

Number	X_r	Y_r	Z_r	X_m	Y_m	Z_m
1	-0.009	-0.276	0.166	0	-0.278	0.155
2	0.052	-0.174	0.026	0.055	-0.175	0.035
3	0.078	0.124	0.026	0.08	0.125	0.03
4	-0.23	0.084	0.109	-0.235	0.086	0.098
5	0.118	0.239	0.126	0.125	0.246	0.125
6	0.031	0.618	0.157	0.044	0.583	0.153
7	0.697	-0.001	0.154	0.693	0	0.153
8	0.006	-0.331	0.13	0	-0.325	0.125
9	-0.08	-0.906	0.334	-0.12	-0.892	0.344
10	0.5	-0.002	0.31	0.502	-0.005	0.307
Mean	0.0091	0.0072	0.0078			
Standard Deviation	0.0114	0.0106	0.0066			

Table 8.1 ArUco Out of the Water 1st Test (Unit:Metre)

Number	X_r	Y_r	Z_r	X_m	Y_m	Z_m
1	-0.706	0.038	0.352	-0.694	0.02	0.33
2	-0.562	0.204	0.307	-0.568	0.125	0.327
3	0.461	-0.147	0.293	0.491	-0.139	0.327
4	0.762	0.069	0.345	0.695	0.06	0.327
5	0.643	0.016	0.385	0.62	0.016	0.349
6	0.814	0.078	0.356	0.725	0.081	0.349
7	-0.105	0.792	0.356	-0.097	0.725	0.349
8	-0.125	0.61	0.343	-0.105	0.592	0.349
9	0.817	0.171	0.356	0.765	0.149	0.349
10	0.483	0.176	0.321	0.510	0.149	0.349
Mean	0.0334	0.0253	0.0185			
Standard Deviation	0.0274	0.0268	0.0116			

Table 8.2 ArUco Out of the Water 2nd Test (Unit:Metre)

In the underwater test, the positions of ArUco Makers are known beforehand. The maker in the image can indicate the rough position of the camera. Hence, an example of the comparison between position estimated by the proposed method and position estimated by the maker appearing in the image is shown in Table 8.3. In the tests, the camera to capture the target marker at different attitudes and distances. According to these results, the estimated position is very close to the associated marker position. In the out-of-water experiments, the camera used to capture the target markers was positioned at different attitudes and distances. The results show that the error is related to the distance between the camera and the marker. More specifically,

when the camera moves closer to the marker, the error decreases. Conversely, the error increases, when the distance between the camera and the marker becomes larger. Within a camera to marker range of half a metre, the error can be limited in the range of $\pm 0.02m$.

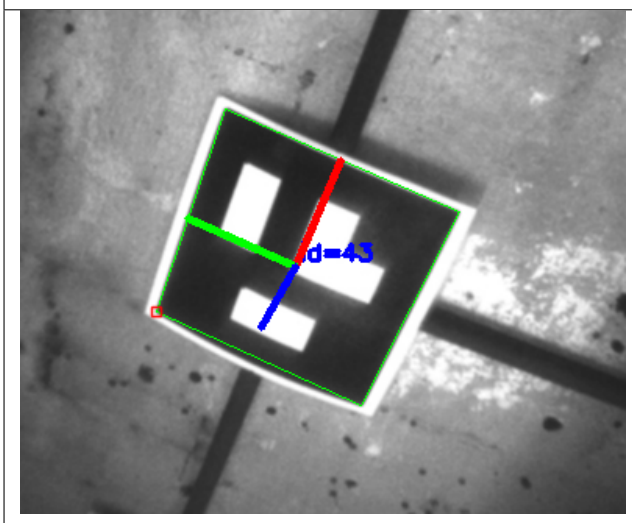
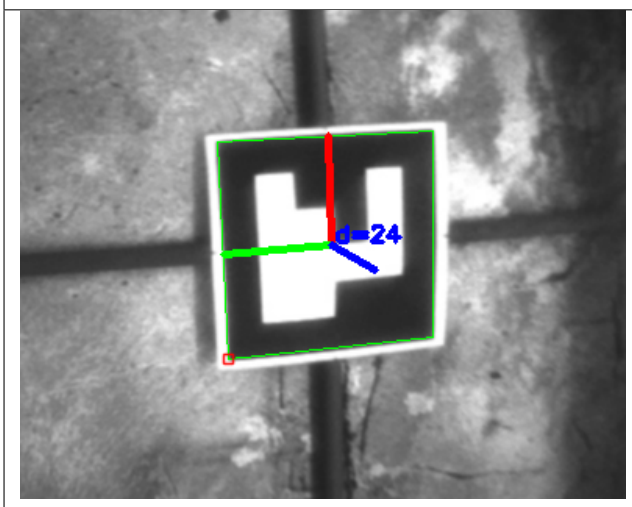
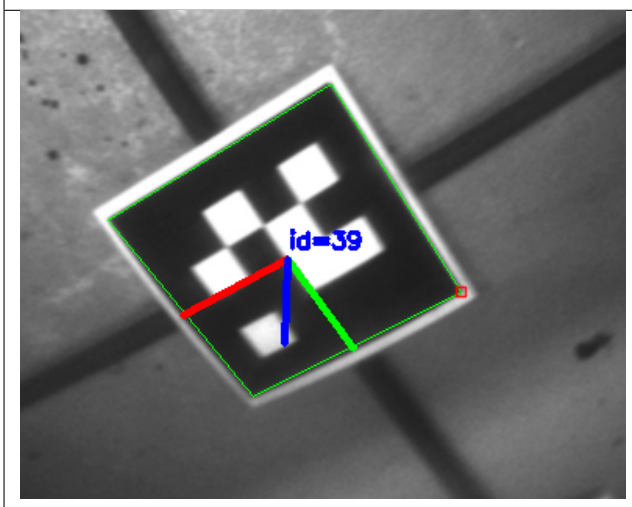
Image	Marker Position	Estimated Position
	(2.5, 2.5)	(2.73, 2.97)
	(1.5, 1.5)	(1.61, 1.67)
	(2.0, 2.5)	(1.86, 2.30)

Table 8.3 Underwater ArUco method Quantitative Test (Unit: Metre)

Unlike odometry methods, which suffer from unbounded cumulative error, the navigation method based on the ArUco markers is not dependent on the previous states. Hence, the error of the algorithm is bounded. However, in Figure 8.11, there are discontinuities in the estimated trajectory. These are caused by mistakes in the marker detection algorithm. The detection algorithm may recognise the marker incorrectly when the marker is far from the camera, or is only partially visible. Because of this, the trajectories generated by the marker navigation system are not consistent.

8.3. Summary

This Chapter introduced the ArUco Markers and an underwater navigation system based on the markers. Through the experimental results, the navigation approach has been demonstrated to provide precise estimation of the vehicle's motion. The advantage of the marker navigation over odometries is that the error is bounded, and cumulative error does not exist. However, there are some limitations. The chief drawback is that the markers need to be set out in advance, and marker arrangement should be known. Secondly, the marker detection algorithm may make mistakes in the following situations: when the camera is moving rapidly; if the camera is far away from the target markers; or is the marker is partly occluded. In these cases, the marker navigation will be unstable. In the following chapters, two novel underwater navigation methods which are independent of artificial markers will be introduced.

Chapter 9. Integrated Visual Odometry with Monocular Camera

In this chapter, a novel Integrated Visual Odometry with Monocular camera (IVO-M) is derived. The integrated visual odometry algorithm benefits from the inclusion of inertial sensors and sonar. The derivation of the method is based on one main assumption: the seabed is locally flat. The process of the IVO-M is: Firstly, the 3D positions of feature points in camera coordinates are identified efficiently with the rotation matrix obtained from a low cost inertial unit and depth from an underwater sonar; Secondly, the translation vector is bounded from the OF algorithm; Finally, a nonlinear optimisation solver is used to correct the attitude from the inertial sensors and output the optimal incremental motion between two frames. Therefore, the computational complexity of the methodology is relatively light. In feature-poor environments, this method is still able to accomplish estimation of the pose of the vehicle, and predict trajectory in absolute scale.

9.1. Assumption of the Proposed Method

The novel IVO-M method is based on one main assumption.

Assumption 1. *The seabed is locally flat.*

In order to compute the depth of the image pixel using depth information from the sonar, a fixed reference has been created. The rotation in the reference is the same as the rotation relative to the global reference, but the translational vector is zero relative to the body reference, as shown in Figure 9.1.

The transformation matrix from the body reference to the fixed reference is given by Equation (9.1).

$$ {}^{fix}\boldsymbol{T}_{body} = \begin{bmatrix} {}^{global}\boldsymbol{R}_{body} & \mathbf{0} \\ 0 & 1 \end{bmatrix}. \tag{9.1} $$

The transformation matrix will be used in the Methodology section to calculate the linear translation. The other coordinate transformations are discussed in the following sections.

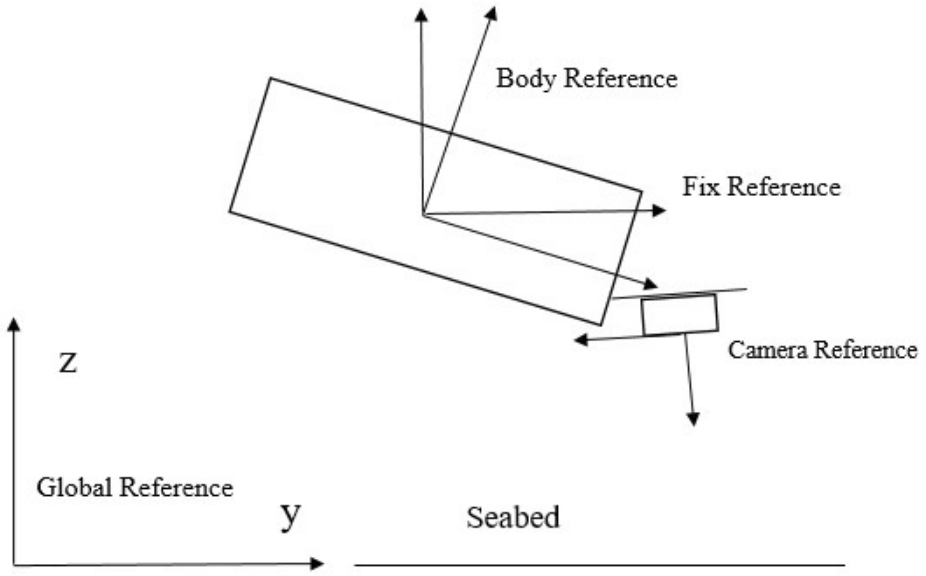


Figure 9.1 Fixed Reference and Body Reference

9.2. Methodology

In the IVO-M method, the optical sensor is used to identify feature points on the seabed. In actual situations, underwater vehicles could maintain constant distance from the seabed, and peaks and troughs caused by small objects on the seabed can be ignored.

In this section, the methodology of the proposed IVO-M is presented in detail. The IVO-M is based on a dual-frame Visual Odometry algorithm. The incremental motion between two frames is predicted and the incremental motions are accumulated to make up a whole trajectory. In the first step, the FAST Corner detector (see subsection 9.2.1) is applied to extract the feature points from the previous frame. Secondly, the optical flow algorithm is employed to find the corresponding feature points in the next frame. Thirdly, the 3D positions of the feature points in the two different frames are estimated using depth measured by the sonar. Linear motion is then predicted using attitude from the inertial sensors i.e. the IMU and compass. Finally, linear motion and attitude are selected as initial values to minimise the projected error by using the Levenberg Marquardt algorithm. The process of the IVO-M method is presented in Figure 9.2.

In ideal situations, the principal axis of the camera is strictly vertical to the seabed. Hence the relationship between altitude and velocity is approximately given by $h = v/q$, where h is the height, v denotes the horizontal forward velocity, and q indicates the magnitude of the optical-flow vector in response to translational movement (Garratt and Chahl, 2007). In practice, the principal axis of the camera will not be perfectly vertical to the seabed, due to camera installation, and roll and pitch motions of the vehicle.

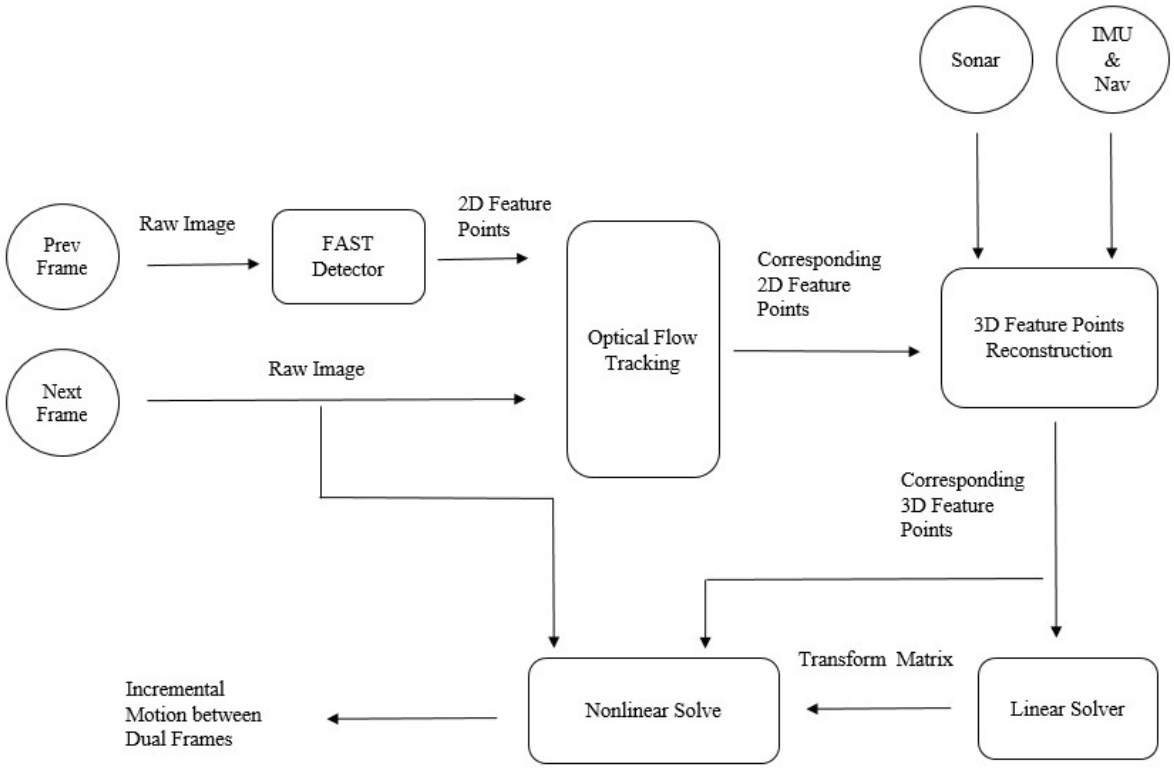


Figure 9.2 IVO-M approach

Through tracked feature points, the orientation of the current frame relative to the previous frame can be solved. In the conventional algorithm, at least eight points are needed to obtain the fundamental matrix. However, in sparse texture environments, the result becomes unreliable. In the proposed method, the orientation of the vehicle can be determined from the IMU sensors, and height of the vehicle above the seabed/tank bottom can also be obtained by the sonar. With these two constraints, the translational vector can be calculated from a single feature point.

9.2.1. FAST Corner Detector

The FAST Corner Detector is applied for extracting feature points from an image (Rosten and Drummond, 2006). There are different kinds of feature detectors such as SIFT (Lowe, 2004), Harris (Harris and Pike, 1988) and ORB (Rublee et al., 2011) which can provide high quality feature information, but they are too computationally intensive for use in real-time applications (Rosten and Drummond, 2006). According to (Scaramuzza and Fraundorfer, 2011), the FAST algorithm is much more computationally efficient and has acceptable robustness. Another key point is that it can detect many more feature points in sparse environments compared to other algorithms.

Because the FAST algorithm is not robust to high levels of noise, the images captured from the camera are processed by a Gaussian blur filter in order to remove noise from the image. The feature points extracted from an example image are shown in Figure 9.3.

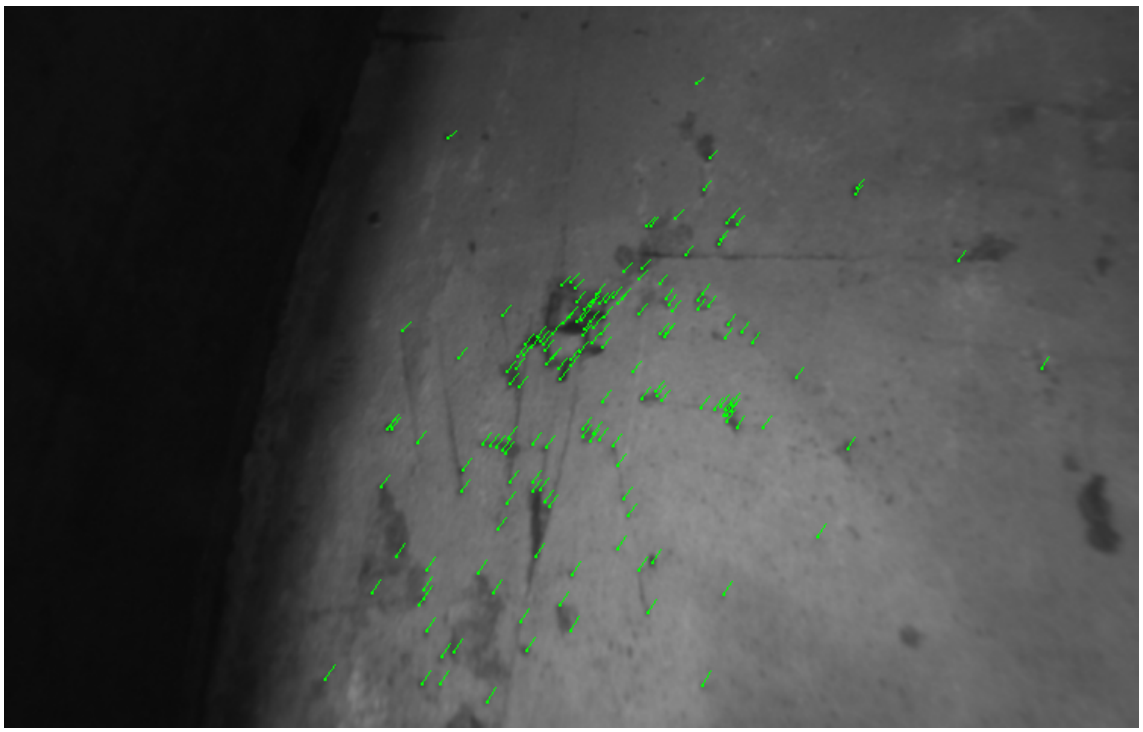


Figure 9.3 Feature Points (Green Markers) Extracted as From Underwater Image

9.2.2. Optical Flow Tracking

The Optical flow algorithm, used to track the apparent motions of brightness across a series of images, relies on the approximation of motion fields computed from a set of image sequences. The algorithm is based on the brightness constancy assumption that the intensity of the pixel remains the same despite small changes of position over time (Ke et al., 2018), which is described by Equation (9.2).

$$\nabla I(\boldsymbol{x},t)\boldsymbol{u} + I_t(\boldsymbol{x},t) = 0. \tag{9.2}$$

Here, ∇I and I_t denote spatial and temporal partial derivatives of the image I, and $\boldsymbol{u}$ denotes the 2D velocity (Sun et al., 2010).

After seabed images are captured by an optical sensor, feature points are extracted from these images and then the Optical flow algorithm is applied for tracking these points on the next frame. Hence, the correspondence of feature points is determined.

9.2.3. Positions of Feature Points

The 3D positions of feature points are reconstructed using the depth from the sonar. Assuming that the seabed is flat planar, the one parameter λ equation is,

$$\boldsymbol{X}_{fix}(\lambda) = {}^{fix}\boldsymbol{T}_{body}\,{}^{body}\boldsymbol{T}_{camera}\boldsymbol{X}_{c}(\lambda). \tag{9.3}$$

Here, λ is the depth of the corresponding pixel, $\boldsymbol{X}_{c}(\lambda)$ is a feature point position in camera coordinates and $\boldsymbol{X}_{fix}(\lambda)$ is the feature point position in fixed coordinates. The definition of ${}^{fix}\boldsymbol{T}_{camera}$ is,

$${}^{fix}\boldsymbol{T}_{camera} = {}^{fix}\boldsymbol{T}_{body}\,{}^{body}\boldsymbol{T}_{camera}. \tag{9.4}$$

The depth can be described in sonar coordinates by:

$$\boldsymbol{d}_{sonar} = [\,0,\ 0,\ z_{sonar},\ 1]^{T}. \tag{9.5}$$

In the fixed coordinates, $\boldsymbol{d}_{sonar}$ becomes

$$\boldsymbol{d}_{fix} = {}^{fix}\boldsymbol{T}_{body}\,{}^{body}\boldsymbol{T}_{sonar}\boldsymbol{d}_{sonar}; \tag{9.6}$$

Therefore, because the seabed is assumed to be flat planar and the distance to the seabed and feature points are both in fixed coordinates, they share the same depth from the x-plane to the seabed. This should be satisfied by the equation:

$$\boldsymbol{X}_{fix}(\lambda)^{3} = \boldsymbol{d}^{3}_{fix}. \tag{9.7}$$

Here, $\boldsymbol{X}_{fix}(\lambda)^{3}$ and $\boldsymbol{d}^{3}_{fix}$ mean the third element in $\boldsymbol{X}_{fix}$ and $\boldsymbol{d}_{fix}$ respectively. Using this condition, λ is solved according to Equation (9.6) and the perspective camera model:

$$\boldsymbol{d}^{3}_{fix} = \lambda\ [r_{31},\ r_{32},\ r_{33}]\boldsymbol{K}^{-1}\boldsymbol{X}_{p} + t_{34}. \tag{9.8}$$

Here, r_{31}, r_{32} and r_{33} are the 1st, 2nd and 3rd elements in the third row of ${}^{fix}\boldsymbol{T}_{camera}$, t_{34} is the 4th element in the third row of the matrix and $\boldsymbol{K}$ is the camera matrix.

Finally, according to Equation (9.8) the scalar λ should be,

$$den = r_{31}\frac{x_p - p_x}{f_x} + r_{32}\frac{y_p - p_y}{f_y} + r_{33} \tag{9.9}$$

$$\lambda = \frac{\boldsymbol{d}^{3}_{fix} - t_{34}}{den}. \tag{9.10}$$

According to Equation (9.3), the position of a feature point in camera coordinates can be derived with known λ. Hence, the positions for 3D feature points in camera coordinates are gained through one image from the camera and depth information from the sonar.

9.2.4. Linear Estimation

As mentioned before, the proposed IVO-M is based on a dual-frame algorithm. The incremental linear motion is predicted between one frame and the next frame. Feature points between the two frames are extracted by the FAST algorithm and reconstructed in three dimensions.

Assuming $\boldsymbol{P}$ is the 3D position of a feature point in one frame, and $\boldsymbol{P}'$ is the same feature point in the next frame, the rotation matrix $\boldsymbol{R}_\Delta$ and translation $\boldsymbol{t}$ satisfy Equation (9.11):

$$\boldsymbol{P}' = \boldsymbol{R}_\Delta \boldsymbol{P} + \boldsymbol{t}. \tag{9.11}$$

The rotation matrix $\boldsymbol{R}_\Delta = {}^{next}\boldsymbol{R}_{prev}$ indicates the rotation from one frame to the next frame caught by the camera. The ${}^{global}\boldsymbol{R}_{imu}$ is obtained from the IMU sensor directly. Because the points' positions are described in camera coordinates, the pose change should be mapped from the IMU to the camera.

$${}^{imu}\boldsymbol{R}_{camera} = {}^{body}\boldsymbol{R}_{imu}^{-1}\,{}^{body}\boldsymbol{R}_{camera}; \tag{9.12}$$

$${}^{global}\boldsymbol{R}_{camera} = {}^{global}\boldsymbol{R}_{imu}\,{}^{imu}\boldsymbol{R}_{camera}; \tag{9.13}$$

$${}^{next}\boldsymbol{R}_{prev} = {}^{global}\boldsymbol{R}_{next}^{-1}\,{}^{global}\boldsymbol{R}_{prev}. \tag{9.14}$$

The translation vector can be described by,

$$\boldsymbol{t}_\Delta = \boldsymbol{P}' - \boldsymbol{R}_\Delta \boldsymbol{P}. \tag{9.15}$$

$$\boldsymbol{T}_\Delta = {}^{next}\boldsymbol{T}_{prev} = \begin{bmatrix} \boldsymbol{R}_\Delta & \boldsymbol{t}_\Delta \\ \boldsymbol{0} & 1 \end{bmatrix} \tag{9.16}$$

Assuming that there is a series of images from time t_1 to t_n, ${}^{t_n}\boldsymbol{T}_{t_1}$ can be derived as,

$${}^{t_n}\boldsymbol{T}_{t_1} = {}^{t_n}\boldsymbol{T}_{t_{n-1}}\,{}^{t_{n-1}}\boldsymbol{T}_{n-2}\ldots{}^{t_2}\boldsymbol{T}_{t_1}. \tag{9.17}$$

Generally, the pose of the camera at t_1 is set as an origin of the trajectory. In this case, the pose of camera at t_n in the global coordinates is,

$$^{global}\boldsymbol{T}_{t_n} = {}^{t_n}\boldsymbol{R}_{t_1}^{-1}. \tag{9.18}$$

Robust Estimation

Assuming there are N feature points, the error function can be written as,

$$E_p = \sum_{n=1}^{N} ||\boldsymbol{t} - (\boldsymbol{P}'_n - \boldsymbol{R}\boldsymbol{P}_n)||_2^2. \tag{9.19}$$

The minimum E_p with respect to the translation vector, $\boldsymbol{t}$, will be:

$$\boldsymbol{t}_{min} = \frac{\sum_{n=1}^{N}(\boldsymbol{P}'_n - \boldsymbol{R}\boldsymbol{P}_n)}{N}. \tag{9.20}$$

Error Analysis

Assuming there is white noise on the pixel coordinates and depth measured by the sonar:

$$\hat{x}_p = x_p + \varepsilon_{xp}; \tag{9.21}$$

$$\hat{y}_p = y_p + \varepsilon_{yp}; \tag{9.22}$$

$$\hat{f}_{fix}^3 = f_{fix}^3 + \varepsilon_f. \tag{9.23}$$

where ε_{xp}, ε_{yp}, and ε_f are white noise. The variances are σ_{xp}, σ_{yp} and σ_f, and the means are all zero.

The λ matrix was obtained through Equations (9.6), and (9.8). The Jacobian matrix for λ with respect to pixel coordinates and depth, is derived as follows,

$$\boldsymbol{J}_\lambda = [\ \frac{\partial\lambda}{\partial x_p}\ \frac{\partial\lambda}{\partial y_p}\ \frac{\partial\lambda}{\partial f_{fix}^3}]. \tag{9.24}$$

$$\boldsymbol{C}_{\lambda} = \begin{bmatrix} \sigma_{xp} & 0 & 0 \\ 0 & \sigma_{yp} & 0 \\ 0 & 0 & \sigma_{f_{fix}^3} \end{bmatrix} \tag{9.25}$$

Therefore, the σ_{λ} is,

$$\sigma_{\lambda} = \boldsymbol{J}_{\lambda} \boldsymbol{C}_{\lambda} \boldsymbol{J}_{\lambda}^{T}. \tag{9.26}$$

and λ is,

$$\lambda \sim N(\overline{\lambda}, \sigma_{\lambda}). \tag{9.27}$$

The derivation of error for $\boldsymbol{t}$ is similar to that for λ. The matched error is also called tracked error, because the feature tracking algorithm cannot match all feature points correctly. Hence, this matched error is described by the white noise on the corresponding feature points in the next frame pixel coordinates: x'_p and y'_p. The Jacobian matrix of $\boldsymbol{t}$ with x'_p, y'_p , λ and λ', is derived as

$$\boldsymbol{J}_t = [\ \frac{\partial(\boldsymbol{P}' - \boldsymbol{RP})}{\partial x'_p}\ \frac{\partial(\boldsymbol{P}' - \boldsymbol{RP})}{\partial y'_p}\ \frac{\partial(\boldsymbol{P}' - \boldsymbol{RP})}{\partial \lambda}\ \frac{\partial(\boldsymbol{P}' - \boldsymbol{RP})}{\partial \lambda'}] \tag{9.28}$$

According to Equation (9.8) and the camera model, the $\boldsymbol{J}_t$ can be described in detail as:

$$\boldsymbol{J}_t = \begin{bmatrix} \frac{\lambda}{f_x} & 0 & \frac{r_{11}(x_p - p_x)}{f_x} + \frac{r_{12}(y_p - p_y)}{f_y} + r_{13} & \frac{x'_p - p_x}{f_x} \\ 0 & \frac{\lambda}{f_y} & \frac{r_{21}(x_p - p_x)}{f_x} + \frac{r_{22}(y_p - p_y)}{f_y} + r_{23} & \frac{y'_p - p_y}{f_y} \\ 0 & 0 & \frac{r_{31}(x_p - p_x)}{f_x} + \frac{r_{32}(y_p - p_y)}{f_y} + r_{33} & 1 \end{bmatrix} \tag{9.29}$$

$$\boldsymbol{C}_t = \begin{bmatrix} \sigma_{xp'} & 0 & 0 & 0 \\ 0 & \sigma_{yp'} & 0 & 0 \\ 0 & 0 & \sigma_{\lambda} & 0 \\ 0 & 0 & 0 & \sigma_{\lambda'} \end{bmatrix} \tag{9.30}$$

$$\boldsymbol{\Sigma}_t = \boldsymbol{J}_t \boldsymbol{C}_t \boldsymbol{J}_t^{T} \tag{9.31}$$

For $\boldsymbol{t}$, the error can be described by,

$$\boldsymbol{t} \sim N(\bar{\boldsymbol{t}}, \boldsymbol{\Sigma}). \tag{9.32}$$

Therefore, drift error in the linear approach is introduced by noise on the sonar, pixel coordinate error and match error. According to Equations (9.29) and (9.30), smaller pixel motion indicates smaller covariance in its distribution. In order to reduce these errors, the Gaussian Blur filter has been applied to remove the pixel noise and a high-frame-rate monocular camera is used to reduce the pixel motion between dual frames. Noise on the IMU is not considered here because the attitude of the vehicle is corrected by the non-linear optimal method, which is discussed in the next section.

9.2.5. Non-linear Iteration Methods

Due to its benefits in iteration methods (Blanco, 2010), the pose of the vehicle $\boldsymbol{T} \in \boldsymbol{SE(3)}$ is described by the associated Lie algebra, $\boldsymbol{\xi} \in \mathfrak{se}(3)$, and $\boldsymbol{\xi} \in \mathbb{R}^6$. The first 3 elements in $\boldsymbol{\xi}$ represent the rotation and the latter 3 elements represent the translation. According to Lie algebra, there is an exponential map from $\boldsymbol{\xi}$ to $\boldsymbol{T}$ (Blanco, 2010).

$$\boldsymbol{T} = exp(\hat{\boldsymbol{\xi}}). \tag{9.33}$$

The term $\hat{\boldsymbol{\xi}}$ is a skew symmetric matrix of $\boldsymbol{\xi}$. Using the Direct method, it is described by probability Maximum Likelihood as,

$$p(\boldsymbol{P}_1, \boldsymbol{P}_2, \ldots \boldsymbol{P}_n, I', I|\boldsymbol{\xi}). \tag{9.34}$$

Here, $\boldsymbol{P}_1, \boldsymbol{P}_2, \ldots \boldsymbol{P}_n$ are 3D feature points in the previous frame, and I' and I are the intensity maps of the next frame and previous frame, they are denoted by $\boldsymbol{Z}$:

$$p(\boldsymbol{Z}|\boldsymbol{\xi}) \propto exp(-\frac{1}{2}\sum_{n=1}^{N}\{(I(u(\boldsymbol{P}_n)) - I'(u(exp(\hat{\xi})\boldsymbol{P}_n)))^T \boldsymbol{\Sigma}_n^{-1}(I(u(\boldsymbol{P}_n)) - I'(u(exp(\hat{\xi})\boldsymbol{P}_n)))\}). \tag{9.35}$$

Where u is a map from 3D feature points to 2D pixel points in the image plane, $\boldsymbol{P} \in \mathbb{R}^3, \boldsymbol{v} \in \mathbb{R}^2, u(\boldsymbol{P}) \to \boldsymbol{v}$. According to the perspective camera model, u is described as,

$$(X, Y, Z)^T \to (fX/Z + p_x, fY/Z + p_y)^T. \tag{9.36}$$

The prior of $\boldsymbol{\xi}$ is predicted from the IMU sensor. The function is,

$$p(\boldsymbol{\xi}_{imu}) \sim N(\overline{\boldsymbol{\xi}}_{imu}, \boldsymbol{\Sigma}_{imu}). \tag{9.37}$$

The posterior of $\boldsymbol{\xi}$ becomes,

$$p(\boldsymbol{\xi}|\boldsymbol{Z}) \propto p(\boldsymbol{Z}|\boldsymbol{\xi})p(\boldsymbol{\xi}). \tag{9.38}$$

The optimal pose is obtained by minimising the negative log-posterior,

$$\boldsymbol{\xi}_{min} = \arg\min_{\boldsymbol{\xi}}\{-log(p(\boldsymbol{\xi}|\boldsymbol{Z}))\}. \tag{9.39}$$

It is equivalent to minimising the cost function,

$$f = \{\ \frac{1}{2}\sum_{n=1}^{N} \boldsymbol{e}_{pn}^T \boldsymbol{\Sigma}_n^{-1} \boldsymbol{e}_{pn} + \frac{1}{2}\boldsymbol{e}_{imu}^T \boldsymbol{\Sigma}_{imu}^{-1} \boldsymbol{e}_{imu}\ \} \tag{9.40}$$

$$\boldsymbol{e}_{pn} = \boldsymbol{P}'_n - exp(\hat{\xi})\boldsymbol{P}_n \tag{9.41}$$

$$\boldsymbol{e}_{imu} = \boldsymbol{\xi} - \boldsymbol{\xi}_{imu}. \tag{9.42}$$

f can be written as,

$$f = \frac{1}{2}\boldsymbol{E}_f^T \boldsymbol{\Sigma}_f^{-1} \boldsymbol{E}_f. \tag{9.43}$$

$$\boldsymbol{E}_f = diag(\boldsymbol{e}_{p1},\ \boldsymbol{e}_{p2},\ \ldots,\ \boldsymbol{e}_{pn},\ \boldsymbol{e}_{imu}) \tag{9.44}$$

$$\boldsymbol{\Sigma}_f = diag(\boldsymbol{\Sigma}_1,\ \boldsymbol{\Sigma}_2,\ \ldots,\ \boldsymbol{\Sigma}_n,\ \boldsymbol{\Sigma}_{imu}). \tag{9.45}$$

The Jacobian matrix is,

$$\boldsymbol{J} = \frac{\partial \boldsymbol{E}_f}{\partial \boldsymbol{\xi}} = diag(\frac{\partial \boldsymbol{e}_{p1}}{\partial \boldsymbol{\xi}},\ \ldots,\ \frac{\partial \boldsymbol{e}_{pn}}{\partial \boldsymbol{\xi}},\ \frac{\partial \boldsymbol{e}_{imu}}{\partial \boldsymbol{\xi}}) \tag{9.46}$$

$$\begin{aligned}
\frac{\partial \boldsymbol{e}_{pn}}{\partial \boldsymbol{\xi}} &= \frac{\partial (I(u(\boldsymbol{P}_n)) - I'(u(exp(\hat{\boldsymbol{\xi}})\boldsymbol{P}_n)))}{\partial \boldsymbol{\xi}} \\
&= \frac{\partial - I'(u(exp(\hat{\boldsymbol{\xi}})\boldsymbol{P}_n))}{\partial \boldsymbol{\xi}} \\
&= -\frac{\partial I'}{\partial u}\frac{\partial u}{\partial exp(\hat{\xi})\boldsymbol{P}_n}\frac{\partial exp(\hat{\xi})\boldsymbol{P}_n}{\partial \boldsymbol{\xi}}
\end{aligned} \tag{9.47}$$

$$\frac{\partial exp(\hat{\xi})\boldsymbol{P}_n}{\partial \boldsymbol{\xi}} = \begin{bmatrix} \boldsymbol{I}_{3\times 3} & -[(exp(\hat{\boldsymbol{\xi}})\boldsymbol{P}_n)]_\times \\ \boldsymbol{0}_{1\times 3} & \boldsymbol{0}_{1\times 3} \end{bmatrix}. \tag{9.48}$$

The $\frac{\partial I'}{\partial u}$ and $\frac{\partial u(\boldsymbol{P})}{\partial \boldsymbol{P}}$ may be derived from Equations (9.2) and (9.36). The derivative of the regularization term ($\boldsymbol{e}_{imu}$) is,

$$\begin{aligned} \frac{\partial \boldsymbol{e}_{imu}}{\partial \boldsymbol{\xi}} &= \frac{\partial(\boldsymbol{\xi} - \boldsymbol{\xi}_{imu})}{\partial \boldsymbol{\xi}} \\ &= \boldsymbol{I}_{6\times 6}. \end{aligned} \tag{9.49}$$

Hence, with the Jacobian matrix, different iterative approaches can be used to obtain the optimal results. For instance, applying the Gauss-Newton iteration algorithm, the update for $\boldsymbol{\xi}$ is,

$$\boldsymbol{\Delta}_\xi = -(\boldsymbol{J}^T \boldsymbol{\Sigma}_f \boldsymbol{J})^{-1} \boldsymbol{J}^T \boldsymbol{\Sigma}_f^{-1} \boldsymbol{E}_f \tag{9.50}$$

$$\boldsymbol{\xi}_{new} = \boldsymbol{\xi}_{old} + \boldsymbol{\Delta}_\xi \tag{9.51}$$

The Levenberg-Marquardt iteration approach can also be applied to minimise the cost function with respect to $\boldsymbol{\xi}$.

Error Analysis

Assuming that, $\overline{\boldsymbol{\xi}}$ is obtained from minimisation of the cost function, the covariance matrix of $\boldsymbol{\xi}$ is (Hartley and Zisserman, 2003),

$$\boldsymbol{\Sigma}_\xi = (\boldsymbol{J}_f^T \boldsymbol{\Sigma}_f^{-1} \boldsymbol{J}_f)^{-1}|_{\boldsymbol{\xi}=\overline{\boldsymbol{\xi}}} \tag{9.52}$$

where $\boldsymbol{\xi}$ is expected to obey,

$$\boldsymbol{\xi} \sim N(\overline{\boldsymbol{\xi}}, \boldsymbol{\Sigma}_\xi). \tag{9.53}$$

9.3. Summary

In this Chapter, the IVO-M method has been introduced. The experimental evaluation will be presented in Chapter 11. The IVO-M is limited by the assumption that the seafloor should be locally flat. The condition can be met easily, when it is applied in artificial underwater environments. Once it encounters a complex underwater terrain, the method may fail in

navigation. In the next chapter, a novel Integrated Visual Odometry with Stereo Camera (IVO-S) will be introduced, which will remove the requirement of the "seafloor locally flat" assumption.

Chapter 10. Integrated Visual Odometry with Stereo Camera

In this chapter, a novel Integrated Visual Odometry with Stereo Camera (IVO-S) method is derived. It combines data from multiple sensors including a stereo camera, inertial sensors and a ping sonar. Unlike the IVO-M, it utilises a stereo camera to recover the 3D position of feature points. The depth from the sonar is used to remove unreliable feature points using a Gaussian distribution model, and then to correct the pose of the vehicle. Hence, the proposed method is robust to estimate the absolute scale trajectory of the vehicle in a sparse feature environment.

In this method, two pairs of images from two stereo frames are undistorted and blurred firstly. The blur operation is utilised to smooth these images, so that the noise in the images is removed. Hence, reliable 3D feature points can be detected and recovered. Then, linear and nonlinear motion recovering methods are applied to estimate the pose of the camera. The flow chart of the IVO-S method is shown in Figure 10.1.

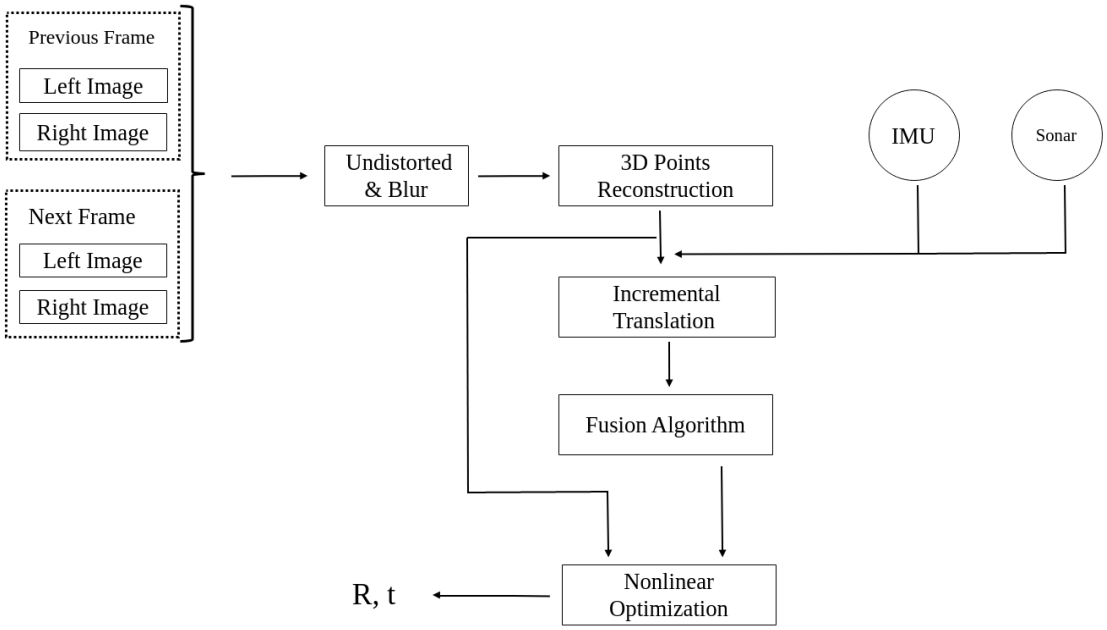


Figure 10.1 Flow Chart of IVO-S Algorithm

Through the calibration described in Chapter 7, the intrinsic matrix $\boldsymbol{K}$, extrinsic parameters, and undistorting polynomial can be obtained.

10.1. 3D Point Reconstruction using Sonar

In this section, a reconstruction algorithm for 3D points is discussed. Although, there are many 3D reconstruction methods that use monocular video sequences, or a combination of monocular camera with either a laser range scanner or a structured light (Parchami and Mariottini, 2014), here stereo reconstruction is applied. In this case, the depth of each feature points can be recovered in absolute scale without moving the camera horizontally.

The feature points are extracted by using the FAST Algorithm. The Optical Flow Tracking algorithm is then used to match the feature points among dual stereo frames. There are four associated feature points generated per set, F_l in the previous left-side frame, F_r in the previous right-side frame, F_l' in the next left-side frame, and F_r' in the next right-side frame. A pair of stereo feature point sets can be used to reconstruct the 3D positions of feature points in that frame. Assuming $\boldsymbol{p}_{nl} \in F_l$, $\boldsymbol{p}_{nr} \in F_r$, there is a map H, $\boldsymbol{p}_{nl} \times \boldsymbol{p}_{np} \rightarrow \boldsymbol{p}_n$, and $\boldsymbol{p}_n \in \mathbb{R}^3$ is the 3D position of the nth feature points.

In order to obtain map $H(\boldsymbol{x}_l, \boldsymbol{x}_r)$, applying the camera model, there is,

$$\boldsymbol{X}/Z = \boldsymbol{K}^{-1}\boldsymbol{x}_l. \tag{10.1}$$

where $\boldsymbol{X}$ is the 3D feature point of $\boldsymbol{x}_l$ in left camera coordinate, $\boldsymbol{K}$ is the camera matrix. And, Z is,

$$Z = \frac{fT}{x_r - x_l} \tag{10.2}$$

Here, the Z term is the depth of the pixel, $(x_r - x_l)$ is the displacement between two correspondence pixels. The terms x_r and x_l are the first elements in vectors $\boldsymbol{x}_r$ and $\boldsymbol{x}_l$, and f is the focal length, one of intrinsic parameters. After the H is derived, there is $H(\boldsymbol{p}_{nl}, \boldsymbol{p}_{nr}) = \boldsymbol{p}_n$, where $\boldsymbol{p}_n$ is the 3D feature point in left camera coordinates in the previous frame. Analogously, using $H(\boldsymbol{p}'_{nl}, \boldsymbol{p}'_{nr}) = \boldsymbol{p}'_n$, the associated 3D feature point in the next frame, $\boldsymbol{p}'_n$ can be obtained as well.

10.1.1. Gaussian Filter with Sonar Information

Both the FAST feature point detection algorithm and the Optical Flow matching method have noise associated with them, which indicates that there are errors on the 2D feature points and matched feature points. Generally, the errors are considered as white noise obeying a Gaussian distribution. Finally, the errors are integrated into 3D feature points. Therefore, the unreliable feature points are required to be removed. A Gaussian filter is used to remove unreliable feature points. The sonar provides the depth (d_{sonar}) from the floor to the vehicle, and the 3D feature

points lie at floor level. Hence, the depth of feature point z_n is expected to lie around d_{sonar}. After translating d_{sonar} into camera coordinates, the filter G_s is presented as,

$$G_s(z_n) = \frac{1}{\sqrt{2\pi}\sigma} exp(-\frac{(z_n - d_{cam})^2}{2\sigma^2}). \tag{10.3}$$

$$\boldsymbol{d}_{cam} = {}^{body}\boldsymbol{T}_{cam}^{-1}\ {}^{body}\boldsymbol{T}_{sonar}\boldsymbol{d}_{sonar} \tag{10.4}$$

Where, the $\boldsymbol{d}_{sonar}$ is the vector measured by the sonar. The term d_{cam} is the third element of $\boldsymbol{d}_{cam}$ and z_n is the third element of the nth 3D feature point. If the output of G_s is higher than the threshold, this 3D feature point is considered as being reliable.

10.2. Incremental Translation Recovery

The incremental translation ${}^2\boldsymbol{t}_1$ between two frames can be computed with the attitude of the vehicle. After applying the Gaussian Filter, the reliable feature point $\boldsymbol{p}_n$ on the previous frame and a corresponding point $\boldsymbol{p}'_n$ on the next frame are known, and the rotation matrix is ${}^2\boldsymbol{R}_1$ from $\boldsymbol{t}_{prev}$ to $\boldsymbol{t}_{next}$. The ${}^2\boldsymbol{t}_1$ is derived as,

$${}^2\boldsymbol{t}_1 = \boldsymbol{p}'_n - {}^2\boldsymbol{R}_1\boldsymbol{p}_n \tag{10.5}$$

The ${}^2\boldsymbol{R}_1$ term is derived by geometry transformation. Let $\boldsymbol{R}_1$ and $\boldsymbol{R}_2$ be the rotation matrix of the previous frame and the next frame with respect to the original reference. The ${}^{global}\boldsymbol{R}_{imu}$ is the rotation matrix obtained from the IMU and gyroscope sensors at the previous frame time point.

$$\boldsymbol{R}_1 = {}^{global}\boldsymbol{R}_{imu}\ {}^{imu}\boldsymbol{R}_{body}\ {}^{body}\boldsymbol{R}_{t265} \tag{10.6}$$

Replacing ${}^{global}\boldsymbol{R}_{imu}$ with that at next frame time point, then $\boldsymbol{R}_2$ is obtained. Hence,

$${}^2\boldsymbol{R}_1 = \boldsymbol{R}_2^{-1}\boldsymbol{R}_1 \tag{10.7}$$

N is the number of corresponding feature points. Analogous to the RANSAC method, in this robust estimation, a pair of feature points and associated ${}^2\boldsymbol{t}_1$, ${}^2\boldsymbol{R}_1$ are used to compute distance as,

$$d_p(\boldsymbol{p}_n, \boldsymbol{p}'_n,\ {}^2\boldsymbol{R}_1,\ {}^2\boldsymbol{t}_1) = \sum_{n=1}^{N} norm\{\boldsymbol{p}'_n - {}^2\boldsymbol{T}_1\boldsymbol{p}_n\}. \tag{10.8}$$

Here, ${}^2\boldsymbol{T}_1 = \{{}^2\boldsymbol{R}_1, {}^2\boldsymbol{t}_1\}$ is the transformation matrix. Unlike the random algorithm, the size of feature points is limited. The ${}^2\boldsymbol{t}_1$ term with minimum distance is expected to be identified. The algorithm for 3D reconstruction and translation recovery is presented as Algorithm 1. After the Gaussian filter, the size of feature points is limited to be small. Hence, the robust estimation does not increase the computational time.

Algorithm 1: 3D Reconstruction and Incremental Translation Recover

Result: ${}^2\bar{\boldsymbol{t}}_1$
2D Feature Point Set on Left Image of previous frame $F_l \leftarrow$ by Fast-Detection Algorithm;
Tacking the points in the set F_l on right image of previous frame on previous frame, left image on next frame, right image on next frame, generating the points sets F_r, F'_l, and F'_r respectively;
$d_0 \leftarrow \infty$;
$n \leftarrow 0$;
Set $A \leftarrow$ Null;
Set $\Gamma \leftarrow$ Null;
while $n < Sizeof(F_l)$ **do**
  $\boldsymbol{p}_{nl} \in F_l$, $\boldsymbol{p}_{nr} \in F_r$ $\boldsymbol{p}'_{nl} \in F'_l$, $\boldsymbol{p}'_{nr} \in F'_r$;
  $\boldsymbol{p}_n \leftarrow H(\boldsymbol{p}_{nl}, \boldsymbol{p}_{nr})$, $\boldsymbol{p}'_n \leftarrow H(\boldsymbol{p}'_{nr}, \boldsymbol{p}'_{nl})$;
  if $Gs(\boldsymbol{p}_n) > Threshold$ **then**
    ${}^2\boldsymbol{t}_1 \leftarrow \boldsymbol{p}'_n - {}^2\boldsymbol{R}_1\boldsymbol{p}_n$;
    Add $\boldsymbol{p}_n$ and $\boldsymbol{p}'_n$ into A;
    Add ${}^2\boldsymbol{t}_1$ into Γ;
  else
    Continue;
  end
  n++;
end
$n \leftarrow 0$;
while $n < Sizeof(\Gamma)$ **do**
  $\boldsymbol{p}_n \in A$, $\boldsymbol{p}'_n \in A$;
  $\boldsymbol{t}_n \in \Gamma$;
  $d_n \leftarrow d_p(\boldsymbol{t}_n, {}^2\boldsymbol{R}_1, \boldsymbol{p}_n, \boldsymbol{p}'_n)$;
  if $d_n < d_0$ **then**
    ${}^2\bar{\boldsymbol{t}}_1 \leftarrow \boldsymbol{t}_n$;
    $d_0 \leftarrow d_n$;
  else
    Continue;
  end
  n++;
end

10.3. Fusion Algorithm

Assuming there are $\boldsymbol{p}$, $\boldsymbol{p}'$, $\boldsymbol{R}_1$, $\boldsymbol{R}_2$, $\boldsymbol{t}_1$, $\boldsymbol{d}_2$ as observed information where $\boldsymbol{p}$ and $\boldsymbol{p}'$ are feature points in 3D coordinates in the previous frame and next frame respectively. The term $\boldsymbol{t}_1$ is the

translation vector of the previous frame relative to the original reference and d_2 is the depth from the vehicle to seabed measured by the sonar, and already transferred to the camera reference. Then ${}^2\boldsymbol{t}_1$ is the translation vector from the previous frame to the next frame .

$$p({}^2\boldsymbol{t}_1|\boldsymbol{p},\boldsymbol{p}',\boldsymbol{R}_1,\boldsymbol{R}_2,d_2,\boldsymbol{t}_1) = \eta\ p(d_2|{}^2\boldsymbol{t}_1,\boldsymbol{R}_2,\boldsymbol{R}_2,\boldsymbol{t}_1)p({}^2\boldsymbol{t}_1|\boldsymbol{p},\boldsymbol{p}',\boldsymbol{R}_2,\boldsymbol{R}_1) \tag{10.9}$$

Assuming that these probabilities obey the Gaussian distribution,

$$p(d_2|{}^2\boldsymbol{t}_1,\boldsymbol{R_2},\boldsymbol{R_1},\boldsymbol{t}_1) \sim N(d_2|\mu_d,\sigma_d^2), \tag{10.10}$$

$$p({}^2\boldsymbol{t}_1|\boldsymbol{p},\boldsymbol{p}',\boldsymbol{R}_1) \sim N({}^2\boldsymbol{t}_1|\boldsymbol{\mu}_t,\Sigma_t). \tag{10.11}$$

According to view geometry, terms $\boldsymbol{\mu}_t$ and μ_d can be derived as,

$$\boldsymbol{\mu}_t = \boldsymbol{p}' - {}^2\boldsymbol{R}_1\boldsymbol{p} \tag{10.12}$$

$$\mu_d = (\boldsymbol{T}_1\ {}^2\boldsymbol{T}_1^{-1})_{[3,4]}. \tag{10.13}$$

Let the $\hat{d_2}$ and ${}^{\hat{2}}\boldsymbol{t}_1$ be,

$${}^2\hat{\boldsymbol{t}}_1 = {}^2\boldsymbol{t}_1 - \boldsymbol{\mu}_t \tag{10.14}$$

$$\hat{d_2} = T(d_2) - \mu_d. \tag{10.15}$$

Here, $T(d_2)$ is a transform function which is used to transfer the d_2 from sonar coordinates to the original reference frame. In the IVO method, $T(d_2) = d_2 - d_{t0}$, d_{t0} is the depth information from the first frame. The $\boldsymbol{T}_1$, ${}^2\boldsymbol{T}_1$ and ${}^2\boldsymbol{R}_1$ are derived as,

$$\boldsymbol{T}_1 = \begin{bmatrix} \boldsymbol{R}_1 & \boldsymbol{t}_1 \\ 0 & 1 \end{bmatrix} \tag{10.16}$$

$${}^2\boldsymbol{T}_1 = \begin{bmatrix} {}^2\boldsymbol{R}_1 & {}^2\boldsymbol{t}_1 \\ 0 & 1 \end{bmatrix} \tag{10.17}$$

$${}^2\boldsymbol{R}_1 = \boldsymbol{R}_2^{-1}\boldsymbol{R}_1 \tag{10.18}$$

Therefore, the related probabilities are presented as,

$$p(d_2|^2\boldsymbol{t}_1, \boldsymbol{R_2}, \boldsymbol{R_1}, \boldsymbol{t}_1) = \frac{1}{\sqrt{2\pi}\sigma} exp(-\frac{\hat{d_2^2}}{2\sigma^2}) \tag{10.19}$$

$$p(^2\boldsymbol{t}_1|\boldsymbol{p}, \boldsymbol{p}', \boldsymbol{R}_1) = \frac{1}{\sqrt{2\pi}|\Sigma|^{1/2}} exp(-\frac{1}{2}\ ^2\hat{\boldsymbol{t}}_1^T\ \Sigma^{-1}\ ^2\hat{\boldsymbol{t}}_1) \tag{10.20}$$

Hence, in order to find the optimal value for $^2\boldsymbol{t}_1$, the posterior should be maximised. It is equivalent to finding a minimum point of the negative log of the posterior,

$$^2\boldsymbol{t}_1 = \underset{^2\boldsymbol{t}_1}{\arg\min}\{-log(p(^2\boldsymbol{t}_1|\boldsymbol{p}, \boldsymbol{p}', \boldsymbol{R}_1, \boldsymbol{R}_2, d_2, \boldsymbol{t}_1))\} \tag{10.21}$$

Making the derivative of Equation (10.21) equal to zero, so that the minimum point is found.

$$\frac{\partial\ -log(p(^2\boldsymbol{t}_1|\boldsymbol{p}, \boldsymbol{p}', \boldsymbol{R}_1, \boldsymbol{R}_2, d_2, \boldsymbol{t}_1))}{\partial\ ^2\boldsymbol{t}_t} = 0 \tag{10.22}$$

$$\begin{gathered}\frac{\partial - log(p(^2\boldsymbol{t}_1|\boldsymbol{p}, \boldsymbol{p}', \boldsymbol{R}_1, \boldsymbol{R}_2, d_2, \boldsymbol{t}_1))}{\partial^2\boldsymbol{t}_t} = \\ \frac{\partial - log(p(d_2|^2\boldsymbol{t}_1, \boldsymbol{R}_2, \boldsymbol{R}_1, \boldsymbol{t}_1))}{\partial^2\boldsymbol{t}_1} + \\ \frac{\partial - log(p(^2\boldsymbol{t}_1|\boldsymbol{p}, \boldsymbol{p}', \boldsymbol{R}_2, \boldsymbol{R}_1))}{\partial^2\boldsymbol{t}_1} = 0\end{gathered} \tag{10.23}$$

Hence, Equation (10.23) is decomposed into two factors,

$$\frac{\partial\ -log(p(d_2|^2\boldsymbol{t}_1, \boldsymbol{R}_2, \boldsymbol{R}_2, \boldsymbol{t}_1))}{\partial\ ^2\boldsymbol{t}_1} = \frac{\hat{d_2}}{\sigma^2}\frac{\partial\ \hat{d_2}}{\partial\ ^2\boldsymbol{t}_1} \tag{10.24}$$

Before continuing to solve Equation (10.24), $^2\boldsymbol{T}_1^{-1}$ is,

$$^2\boldsymbol{T}_1 = \begin{bmatrix} ^2\boldsymbol{R}_1^T & -^2\boldsymbol{R}_1^T\boldsymbol{t}_1 \\ 0 & 1 \end{bmatrix}. \tag{10.25}$$

In this case, $\boldsymbol{T}_1\ ^2\boldsymbol{T}_1^{-1}$ is calculated as,

$$\boldsymbol{T}_1\ ^2\boldsymbol{T}_1^{-1} = \begin{bmatrix} ^2\boldsymbol{R}_1\boldsymbol{R}_1^T & \boldsymbol{t}_1 - {}^2\boldsymbol{R}_1\boldsymbol{R}_1^T\ ^2\boldsymbol{t}_1 \\ 0 & 1 \end{bmatrix} \tag{10.26}$$

Now the second term of Equation (10.24) is,

$$\begin{aligned}\frac{\partial\,\hat{d}_2}{\partial\,{}^2\boldsymbol{t}_1} &= \frac{\partial\,T(d_2)-({}^1\boldsymbol{T}_1\,{}^2\boldsymbol{T}_1^{-1})_{[3,4]}}{\partial\,{}^2\boldsymbol{t}_1} \\ &= \frac{\partial T(d_2)-(\boldsymbol{t}_1-{}^1\boldsymbol{R}_1\,{}^2\boldsymbol{R}_1^T\,{}^2\boldsymbol{t}_1)_{[3]}}{\partial^2\boldsymbol{t}_1}\end{aligned} \tag{10.27}$$

The term $\boldsymbol{r}_3$ is the column vector of the third row of $\boldsymbol{R}^*$. The second term of Equation (10.24) is,

$$\begin{aligned}\frac{\partial\,\hat{d}_2}{\partial\,{}^2\boldsymbol{t}_1} &= \frac{\partial T(d_2)-(\boldsymbol{t}_1-\boldsymbol{R}^*\,{}^2\boldsymbol{t}_1)_{[3]}}{\partial^2\boldsymbol{t}_1} \\ &= \frac{\partial\ -\boldsymbol{t}_1^{[3]}+\boldsymbol{r}_3^T\,{}^2\boldsymbol{t}_1)}{\partial\,{}^2\boldsymbol{t}_1} \\ &= \frac{\partial\,\boldsymbol{r}_3^T\,{}^2\boldsymbol{t}_1}{\partial\,{}^2\boldsymbol{t}_1} \\ &= \boldsymbol{r}_3\end{aligned} \tag{10.28}$$

Therefore, the result of Equation (10.24) is,

$$\begin{aligned}\frac{\hat{d}_2}{\sigma^2}\frac{\partial\,\hat{d}_2}{\partial\,{}^2\boldsymbol{t}_1} &= \frac{\hat{d}_2}{\sigma^2}\boldsymbol{r}_3 \\ &= \frac{T(d_2)-(\boldsymbol{t}_1^{[3]}-\boldsymbol{r}_3^T\,{}^2\boldsymbol{t}_1)}{\sigma^2}\boldsymbol{r}_3 \\ &= \frac{T(d_2)\boldsymbol{r}_3-(\boldsymbol{t}_1^{[3]}-\boldsymbol{r}_3^T\,{}^2\boldsymbol{t}_1)\boldsymbol{r}_3}{\sigma^2}\end{aligned} \tag{10.29}$$

For Equation (10.23), there is

$$\begin{aligned}\frac{\partial-log(p({}^2\boldsymbol{t}_1|\boldsymbol{p},\boldsymbol{p}',\boldsymbol{R}_2,\boldsymbol{R}_1))}{\partial\,{}^2\boldsymbol{t}_1} &= \Sigma^{-1}\,{}^2\hat{\boldsymbol{t}}_1\frac{\partial\,{}^2\hat{\boldsymbol{t}}_1}{\partial\,{}^2\boldsymbol{t}_1} \\ &= \Sigma^{-1}\,{}^2\hat{\boldsymbol{t}}_1 \\ &= \Sigma^{-1}\,({}^2\boldsymbol{t}_1-(\boldsymbol{p}'-\,{}^2\boldsymbol{R}_1\boldsymbol{p}))\end{aligned} \tag{10.30}$$

Combining the two derivative factors, the result of Equation (10.23) is,

$$\Sigma^{-1}\,({}^2\boldsymbol{t}_1-(\boldsymbol{p}'-\,{}^2\boldsymbol{R}_1\boldsymbol{p}))+\frac{T(d_2)\boldsymbol{r}_3-(\boldsymbol{t}_1^{[3]}-\boldsymbol{r}_3^T\,{}^2\boldsymbol{t}_1)\boldsymbol{r}_3}{\sigma^2}=0 \tag{10.31}$$

The minimum point for ${}^2\boldsymbol{t}_1$ in Equation (10.21) is,

$${}^2\boldsymbol{t}_1=(\Sigma\boldsymbol{r}_3\boldsymbol{r}_3^T+\sigma^2\boldsymbol{I})^{-1}\{\sigma^2(\boldsymbol{p}'-\,{}^2\boldsymbol{R}_1\boldsymbol{p})-\Sigma(T(d_2)-\boldsymbol{t}_1^{[3]})\boldsymbol{r}_3\} \tag{10.32}$$

Based on the robust algorithm in the previous section, Equation (10.32) can be rewritten as,

$$^{2}\boldsymbol{t}_1 = (\Sigma \boldsymbol{r}_3 \boldsymbol{r}_3^T + \sigma^2 \boldsymbol{I})^{-1}\{\sigma^2\ ^{2}\bar{\boldsymbol{t}}_1 - \Sigma(T(d_2) - \boldsymbol{t}_1^{[3]})\boldsymbol{r}_3\} \tag{10.33}$$

10.4. Nonlinear Optimisation

In the nonlinear iterative method, the associated Lie algebra $\boldsymbol{\xi}$, which comprises two separate 3-element vectors, is to describe the pose of the vehicle relative to the origin rather than $\boldsymbol{T}$ (Blanco, 2010). This is because it is much more numerically stable in the iterative process. The first 3-element vector in $\boldsymbol{\xi}$ is the rotation vector and the second 3-element vector is the translation vector. According to Lie algebra theory, the exponential map from $\boldsymbol{\xi}$ to $\boldsymbol{T}$ is (Blanco, 2010),

$$\boldsymbol{T} = exp(\hat{\boldsymbol{\xi}}). \tag{10.34}$$

The $\hat{\boldsymbol{\xi}}$ denotes the skew symmetric matrix of the vector $\boldsymbol{\xi}$. It is written in the probability Maximum Likelihood form as,

$$p(\boldsymbol{p}_1, \boldsymbol{p}_2, \ldots \boldsymbol{p}_n, \boldsymbol{p}'_1, \boldsymbol{p}'_2, \ldots, \boldsymbol{p}'_n | \boldsymbol{\xi}). \tag{10.35}$$

Where, $\boldsymbol{p}_1, \boldsymbol{p}_2, \ldots \boldsymbol{p}_n, \boldsymbol{p}'_1, \boldsymbol{p}'_2, \ldots, \boldsymbol{p}'_n$ can be regarded as the observed feature points in body coordinates, which can be written in short form as $\boldsymbol{Z}$. Hence,

$$p(\boldsymbol{Z}|\boldsymbol{\xi}) \propto exp(-\frac{1}{2}\sum_{n=1}^{N}\{(\boldsymbol{p}'_n - exp(\hat{\xi})\boldsymbol{p}_n)^T \boldsymbol{\Sigma}_n^{-1}(\boldsymbol{p}'_n - exp(\hat{\xi})\boldsymbol{p}_n)\}). \tag{10.36}$$

The prior of $\boldsymbol{\xi}$ is obtained from the inertial sensors. The equation is,

$$p(\boldsymbol{\xi}_{imu}) \sim N(\bar{\boldsymbol{\xi}}_{imu}, \boldsymbol{\Sigma}_{imu}). \tag{10.37}$$

The posterior of $\boldsymbol{\xi}$ is,

$$p(\boldsymbol{\xi}|\boldsymbol{Z}) \propto p(\boldsymbol{Z}|\boldsymbol{\xi})p(\boldsymbol{\xi}). \tag{10.38}$$

The optimisation solution is equivalent to minimising the negative log-posterior with the Lie algebra vector,

$$\boldsymbol{\xi}_{min} = \arg\min_{\boldsymbol{\xi}}\{-log(p(\boldsymbol{\xi}|\boldsymbol{Z}))\}. \tag{10.39}$$

This can be written as a cost function as,

$$f = \{ \frac{1}{2} \sum_{n=1}^{N} \boldsymbol{e}_{pn}^T \boldsymbol{\Sigma}_n^{-1} \boldsymbol{e}_{pn} + \frac{1}{2} \boldsymbol{e}_{imu}^T \boldsymbol{\Sigma}_{imu}^{-1} \boldsymbol{e}_{imu} \} \tag{10.40}$$

$$\boldsymbol{e}_{pn} = \boldsymbol{p}'_n - exp(\hat{\boldsymbol{\xi}}) \boldsymbol{p}_n \tag{10.41}$$

$$\boldsymbol{e}_{imu} = \boldsymbol{\xi} - \boldsymbol{\xi}_{imu}. \tag{10.42}$$

f is,

$$f = \frac{1}{2} \boldsymbol{E}_f^T \boldsymbol{\Sigma}_f^{-1} \boldsymbol{E}_f. \tag{10.43}$$

$$\boldsymbol{E}_f = diag(\boldsymbol{e}_{p1}, \boldsymbol{e}_{p2}, \ \ldots, \boldsymbol{e}_{pn}, \boldsymbol{e}_{imu}) \tag{10.44}$$

$$\boldsymbol{\Sigma}_f = diag(\boldsymbol{\Sigma}_1, \boldsymbol{\Sigma}_2, \ \ldots, \boldsymbol{\Sigma}_n, \boldsymbol{\Sigma}_{imu}). \tag{10.45}$$

The Jacobian matrix may be written as,

$$\boldsymbol{J} = \frac{\partial \boldsymbol{E}_f}{\partial \boldsymbol{\xi}} = diag(\frac{\partial \boldsymbol{e}_{p1}}{\partial \boldsymbol{\xi}}, \ \ldots, \ \frac{\partial \boldsymbol{e}_{pn}}{\partial \boldsymbol{\xi}}, \ \frac{\partial \boldsymbol{e}_{imu}}{\partial \boldsymbol{\xi}}) \tag{10.46}$$

$$\begin{aligned} \frac{\partial \boldsymbol{e}_{pn}}{\partial \boldsymbol{\xi}} &= \frac{\partial (\boldsymbol{p}'_n - \exp(\hat{\boldsymbol{\xi}}) \boldsymbol{p}_n)}{\partial \boldsymbol{\xi}} \\ &= \frac{\partial - exp(\hat{\boldsymbol{\xi}}) \boldsymbol{p}_n}{\partial \boldsymbol{\xi}} \\ &= - \begin{bmatrix} \boldsymbol{I}_{3\times 3} & -[(exp(\hat{\boldsymbol{\xi}}) \boldsymbol{p}_n)]_\times \\ \boldsymbol{0}_{1\times 3} & \boldsymbol{0}_{1\times 3} \end{bmatrix}. \end{aligned} \tag{10.47}$$

The partial derivative of the regularisation term ($\boldsymbol{e}_{imu}$) is,

$$\begin{aligned} \frac{\partial \boldsymbol{e}_{imu}}{\partial \boldsymbol{\xi}} &= \frac{\partial (\boldsymbol{\xi} - \boldsymbol{\xi}_{imu})}{\partial \boldsymbol{\xi}} \\ &= \boldsymbol{I}_{6\times 6}. \end{aligned} \tag{10.48}$$

Hence, with the Jacobian matrix, iterative approaches can be utilised to provide the solutions. For instance, if the Gauss-Newton iteration algorithm is applied, the update for $\boldsymbol{\xi}$ in every iterative step is,

$$\boldsymbol{\Delta}_{\xi} = -(\boldsymbol{J}^T \boldsymbol{\Sigma}_f \boldsymbol{J})^{-1} \boldsymbol{J}^T \boldsymbol{\Sigma}_f^{-1} \boldsymbol{E}_f \tag{10.49}$$

$$\boldsymbol{\xi}_{new} = \boldsymbol{\xi}_{old} + \boldsymbol{\Delta}_{\xi} \tag{10.50}$$

In the present implementation, the Levenberg-Marquardt iteration approach was applied, which will discussed in the next Chapter.

10.5. Summary

This chapter has illustrated the IVO-S method. It uses a stereo camera to estimate the 3D position of feature points. Combining the data from multiple sensors enables the method to estimate pose accurately in a sparse environment with low cost but high precision. The method will be evaluated and tested in Chapter 11.

Chapter 11. Implementation & Experiments for IVO Methods

In this chapter, the evaluation of the IVO-S and IVO-M are presented through physical experiments. The implementations of the IVO algorithms are introduced briefly, and the quantitative experimental methods are then presented. The results for IVO-S and IVO-M will be illustrated, followed by associated discussions.

11.1. Implementation of the IVO methods

The IVO algorithms were implemented in C++ language. The OpenCV libraries were utilised to process the images, including undistortion of images, the FAST corner detection and image smoothing. In the nonlinear algorithm, the Levenberg Marquardt (LM) approach was used to solve the nonlinear least square problem. In the code, the Ceres Solver was applied to obtain the solutions. This is an open source C++ library for solving optimisation problems (Agarwal and Mierle, 2018). Pangolin is a light-weight portable library for managing OpenGL. It was applied to draw the 3D trajectory and provide a graphical user interface.

11.2. Quantitative Evaluation Methods

The underwater collection was done using the modified ROV. As mentioned before, there are multiple precise sensors in the watertight tube. The collected data was stored in the onboard computer temporarily. Once the test finished, the data was moved to a hard disk via WIFI.

The modification changed the centres of gravity and bouyancy of the vehicle, which had a negative impact on the manoeuvrability. Hence, during evaluation of the IVO algorithms, an operator held the vehicle with two half metre sticks, and moved along the reference line of the test. Simultaneously, the sensors in the watertight tube kept recording the data.

The experimental environment was the towing tank at Newcastle University (Bowes, 2010). A square defined by straps was set out, providing a reference line. The scale and 3D pictures of the square are illustrated in Figure 11.1, and Figure 11.2. The vehicle was controlled by the operator manually within the square. Hence, the vehicle was forced to travel along the path drawn by the straps, as shown in Figure 11.3. Experiments without a reference path were conducted as well,

as shown in Figure 11.4. In these cases, the vehicle was operated to travel along closed shapes, such as squares, triangles and circles.

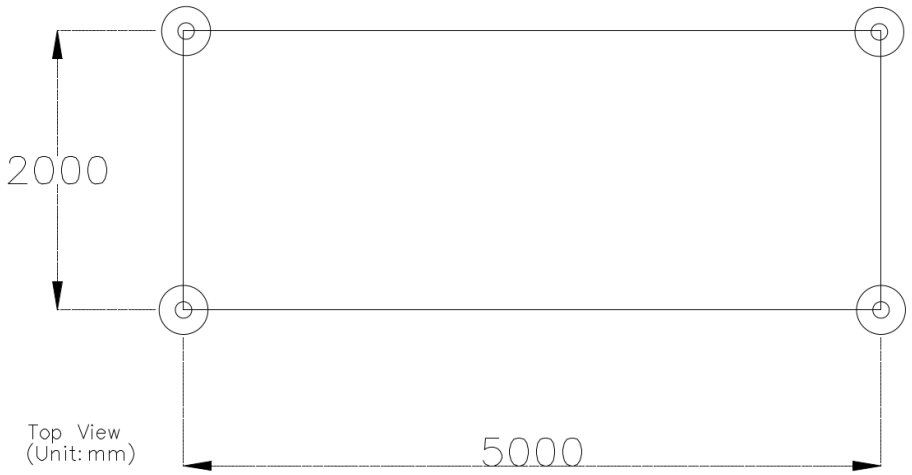


Figure 11.1 Square Strap Scale in Towing Tank

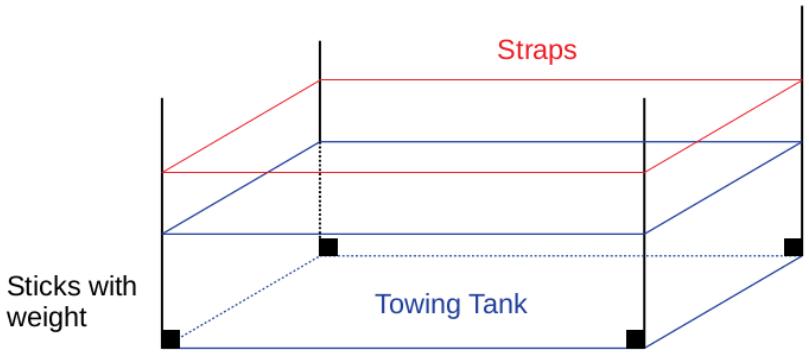


Figure 11.2 Square Strap 3D in Towing Tank



Figure 11.3 Square Strap and Operator



Figure 11.4 Operator without Reference Strap

These tests were conducted by operating the vehicle to travel along different reference paths. The IVO methods (IVO-M and IVO-S) and T265 V-SLAM were used to estimate the trajectory of the vehicle. Actually, the algorithms offer the trajectory of the camera but the camera is installed on the vehicle. Hence, the position of the camera can be regarded as the position of the vehicle. The attitude of the vehicle can be obtained through the transformation matrices. The shapes of the reference paths were: a square, a triangle and a figure-8 shape. In each case, the vehicle returned to the starting point at the end of the test, hence each estimated trajectory is

expected to be closed. The experimental errors are caused by the manual operation and the navigation algorithms. The errors generated by the algorithms have been discussed in Chapter 9 and Chapter 10. The operator may not return to the starting point exactly, nor hold the vehicle so that it travels along the reference path exactly. These factors will contribute the experimental errors. Due to experimental errors, the final pose of the ROV in the estimated trajectory has an offset relative to the starting point. The error ratio can be calculated by dividing the position error by the length of whole trajectory, which is,

$$e_r = \frac{offset}{trajectory\ length}. \tag{11.1}$$

Such an evaluation method is very popular in verification of navigation system performance (Zhang and Singh, 2015). The accumulative error may be reduced when the vehicle travels in a loop shape, because the error accumulated in the return journey may offset the error accumulated in the forward journey. In this case, at the end point of the forward journey, the error may be higher. Hence in the experiments, reference paths are introduced to evaluate the performances of the algorithms. The algorithms estimate 3D trajectories as shown in Figure 11.5, and they have been plotted in 2D figures as a top view. Each shape of reference path was tested multiple times to prove the repeatability of the method. Because the trajectories estimated by the T265 have a big drift error relative to the size of the reference, the reference lines are not shown in the results from the T265.

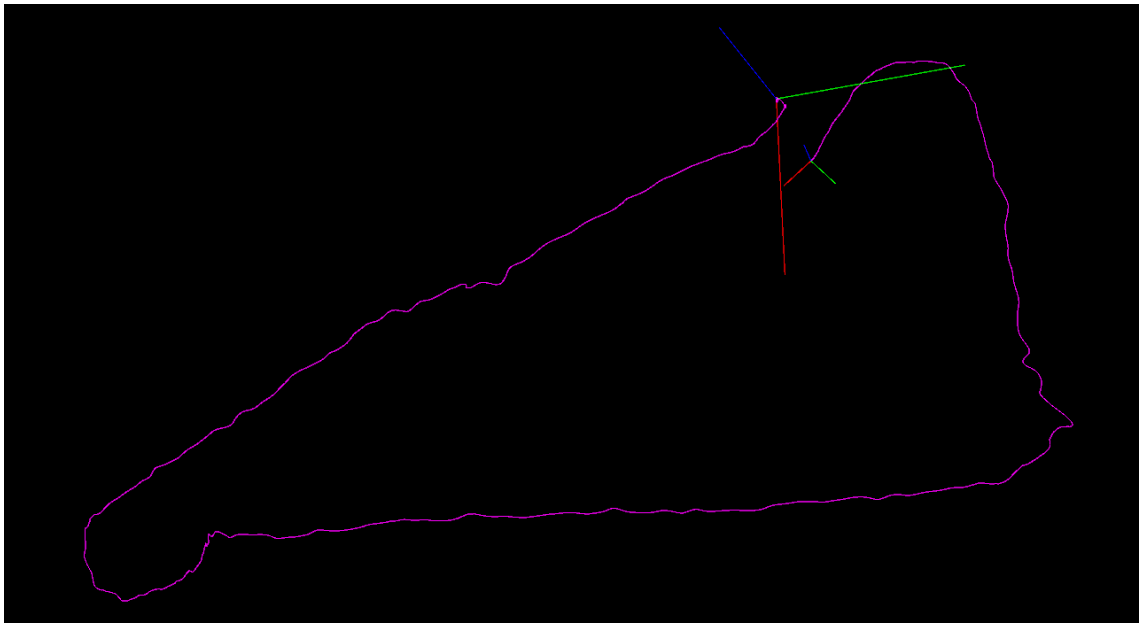


Figure 11.5 Estimated Trajectory in 3D

The experiments for the IVO-M and IVO-S are described separately. Both of them have comparative results from V-SLAM running on the Intel Tracking Camera T265.

11.3. Experiment Results for IVO-M Method

The estimated trajectories from the IVO-M method and estimated results from V-SLAM on the T265 are plotted in 2D figures, shown from Figure 11.6 to Figure 11.21. The errors and length of each test based on the IVO-M method are presented in Table 11.1. The reference path is drawn in a black dotted line in these results. For the results given in Table 11.1, the experiments were conducted a total of eight times: the square, triangle, and figure-8 shapes with reference path were tested twice and a circle shape and a triangle shape without reference path were tested once each.

Shape	Distance(metre)	Error Ratio
With Reference Path		
Square 1st	17.5219	0.0560
Square 2nd	16.8559	0.0230
Triangular 1st	15.7125	0.0051
Triangular 2nd	16.1831	0.0090
Figure-8 1st	22.6775	0.0315
Figure-8 2nd	22.7320	0.0435
Without Reference Path		
Triangular	14.9377	0.0270
Circle	17.8133	0.0391

Table 11.1 Error Ratio and Distance of each Test Using IVO-M Method

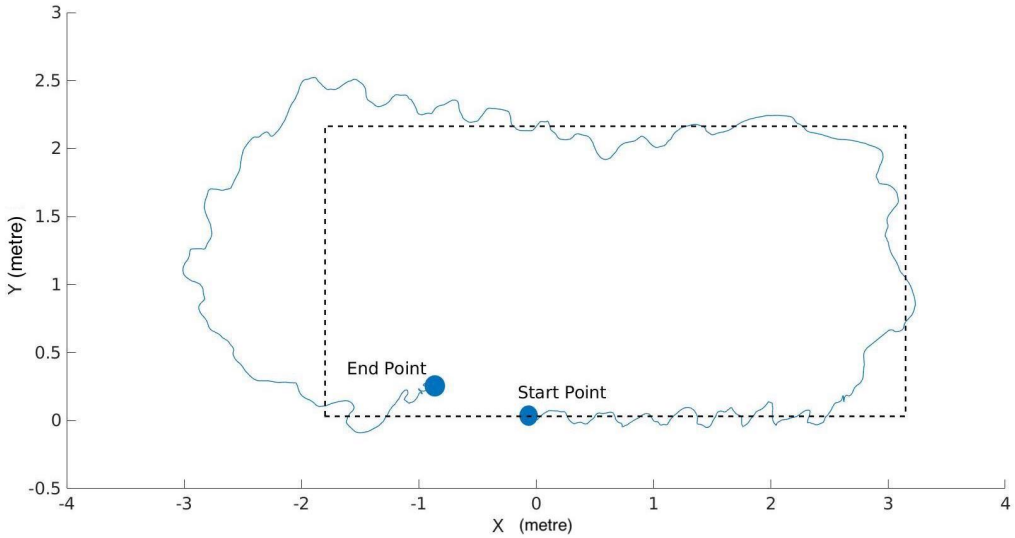


Figure 11.6 IVO-M Square Shape (1st test) with Reference, $e_r = 5.6\%$

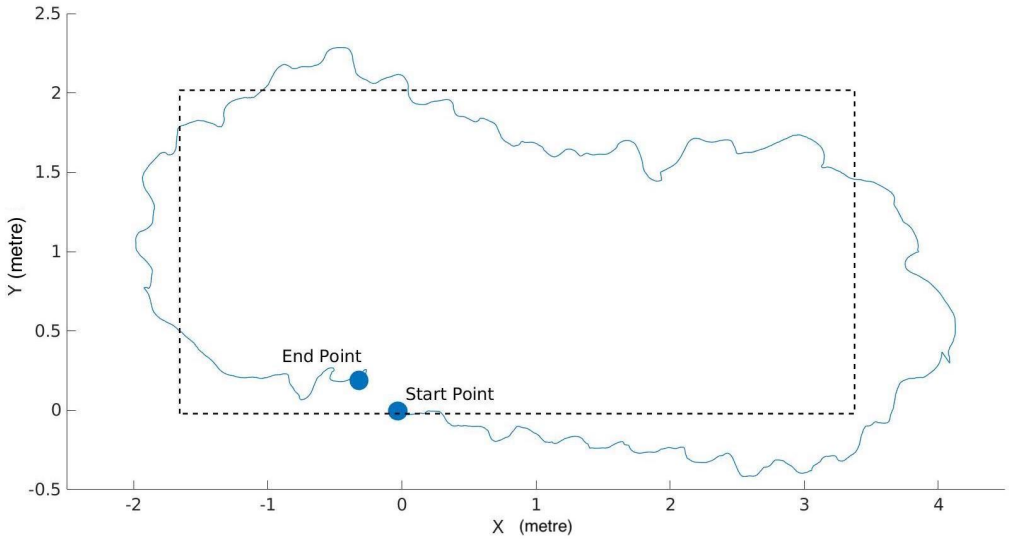


Figure 11.7 IVO-M Square Shape (2nd test) with Reference, $e_r = 2.3\%$

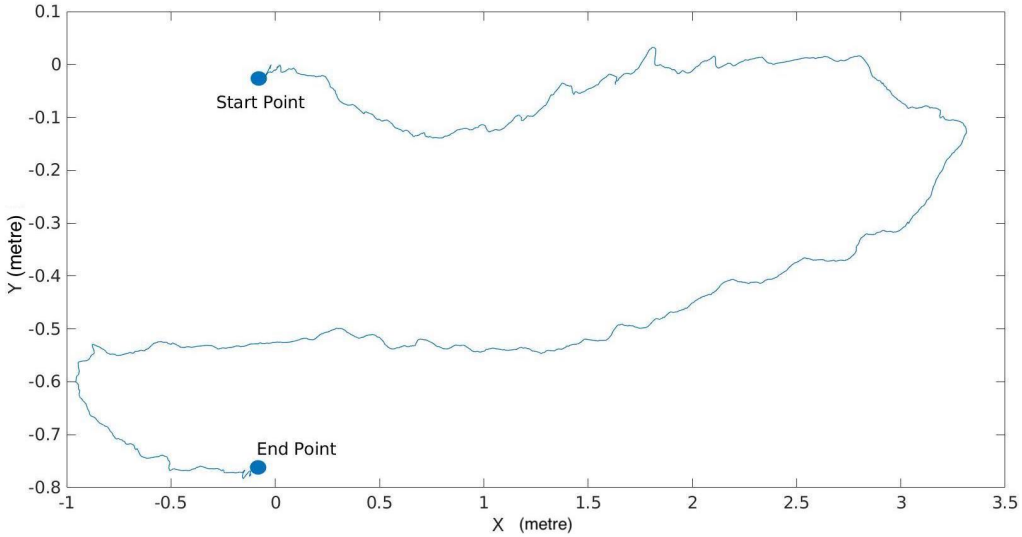


Figure 11.8 T265 Square Shape (1st test)

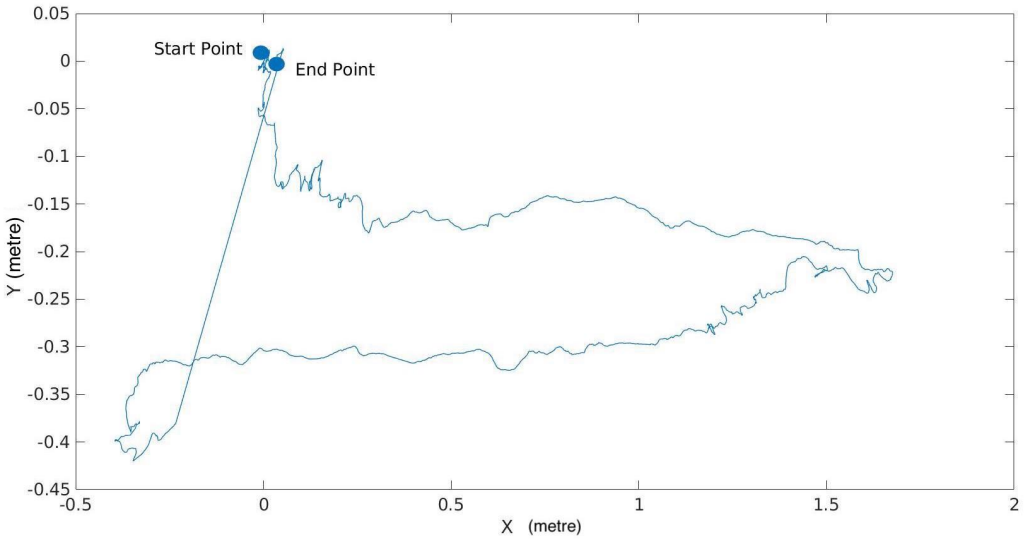


Figure 11.9 T265 Square Shape (2nd test)

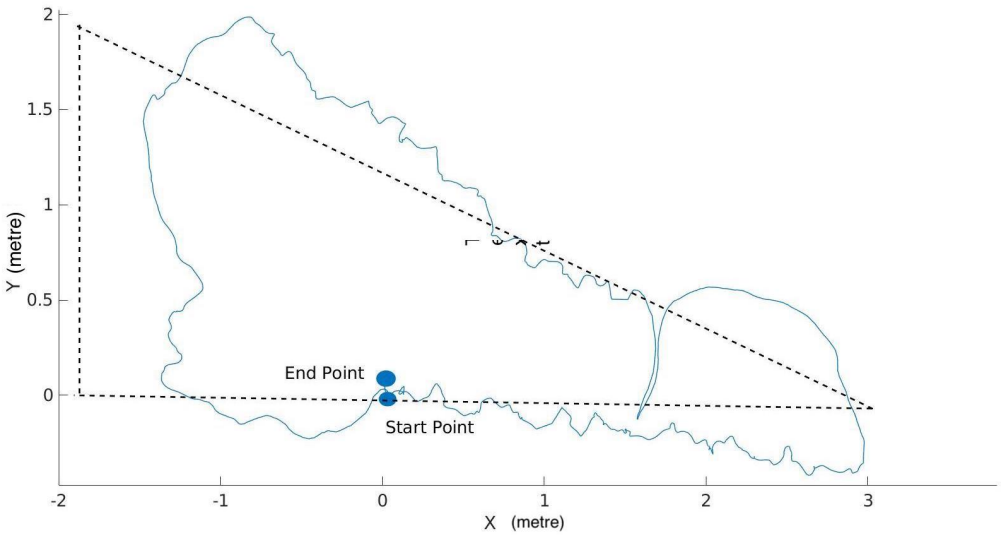


Figure 11.10 IVO-M Triangle Shape (1st test) with Reference, $e_r = 0.51\%$

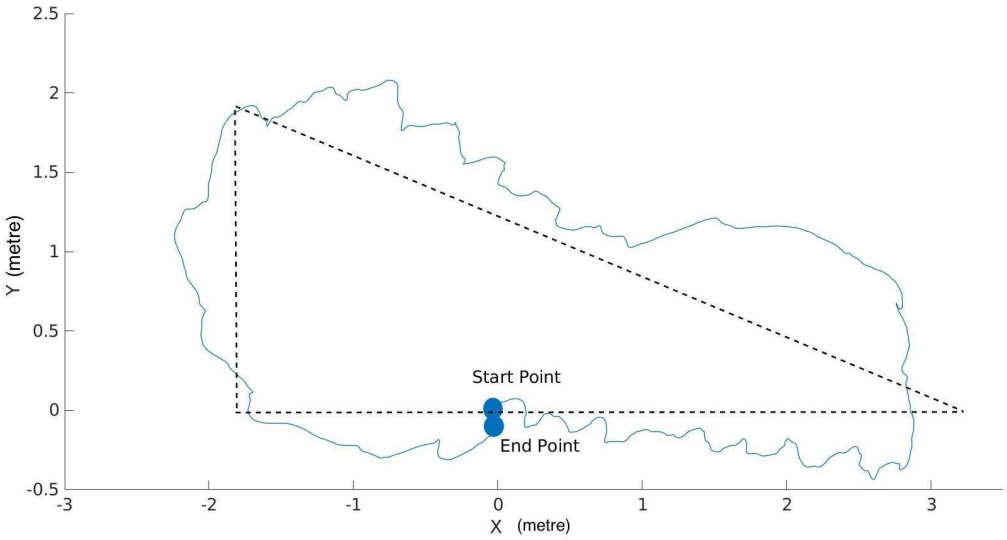


Figure 11.11 IVO-M Triangle Shape (2nd test) with Reference, $e_r = 0.90\%$

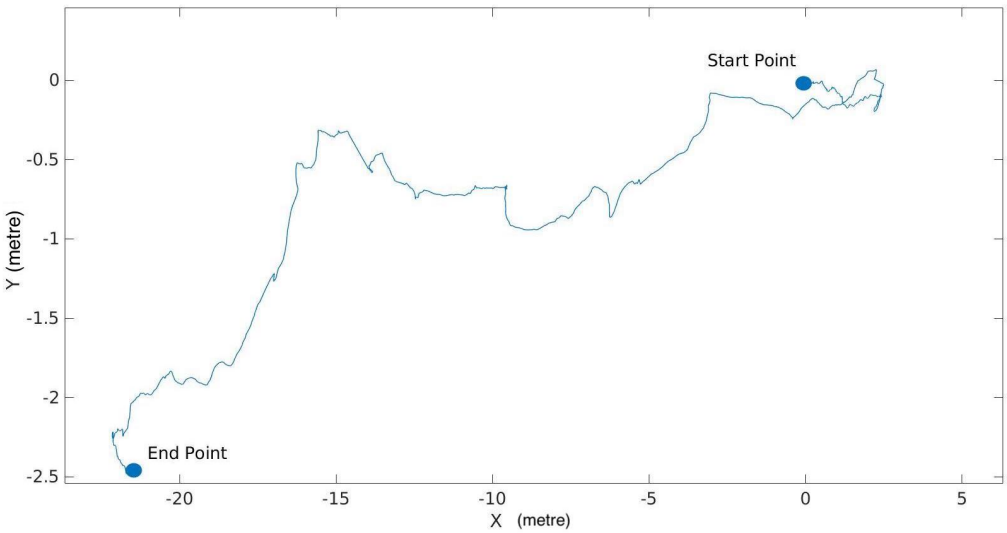


Figure 11.12 T265 Triangle Shape (1st test)

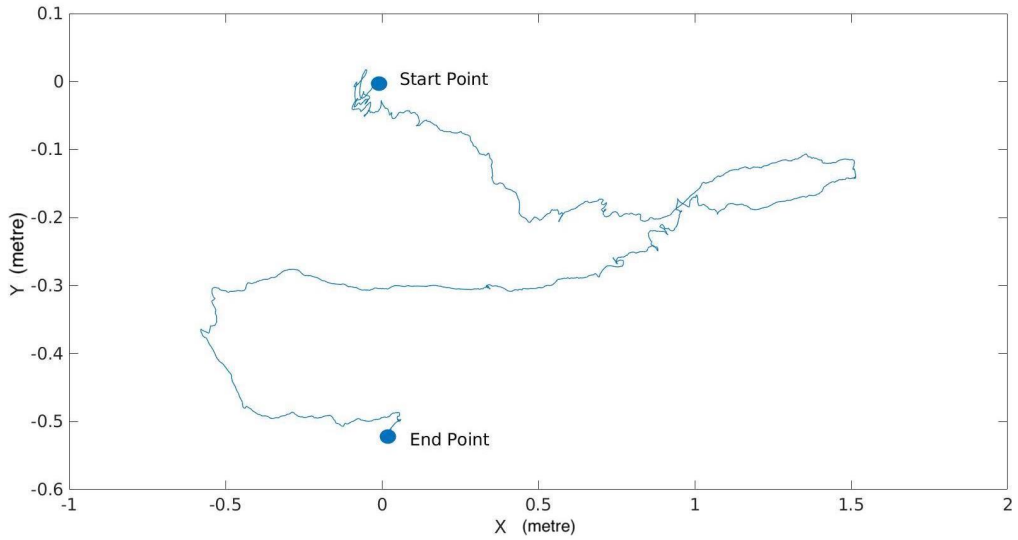


Figure 11.13 T265 Triangle Shape (2nd test)

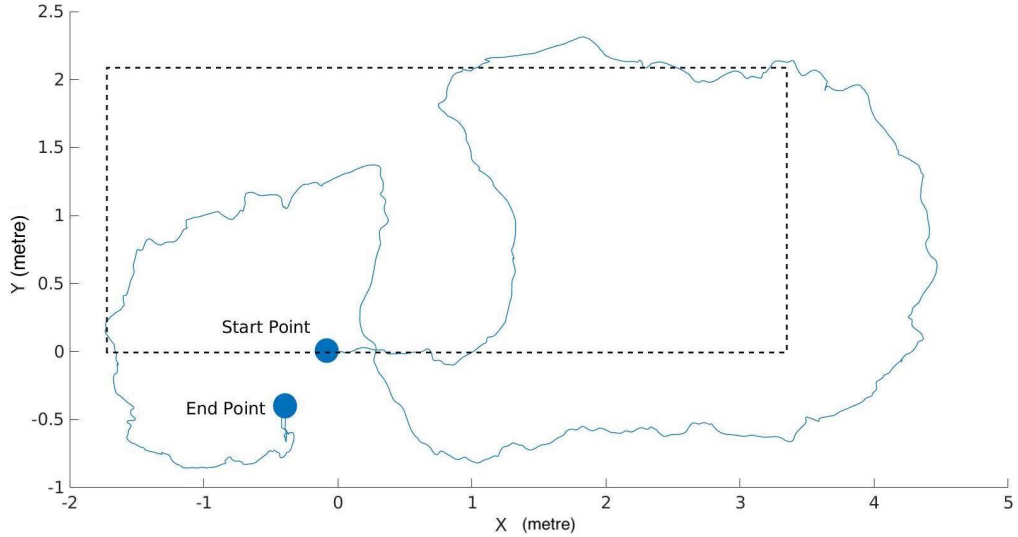


Figure 11.14 IVO-M Figure-8 Shape (1st test) with Reference, $e_r = 3.15\%$

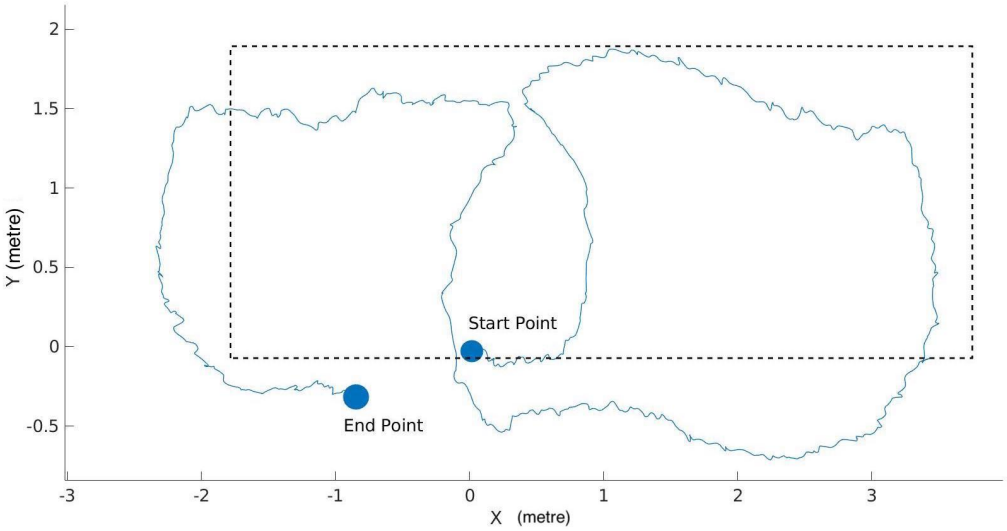


Figure 11.15 IVO-M Figure-8 Shape (2nd test) with Reference, $e_r = 4.35$

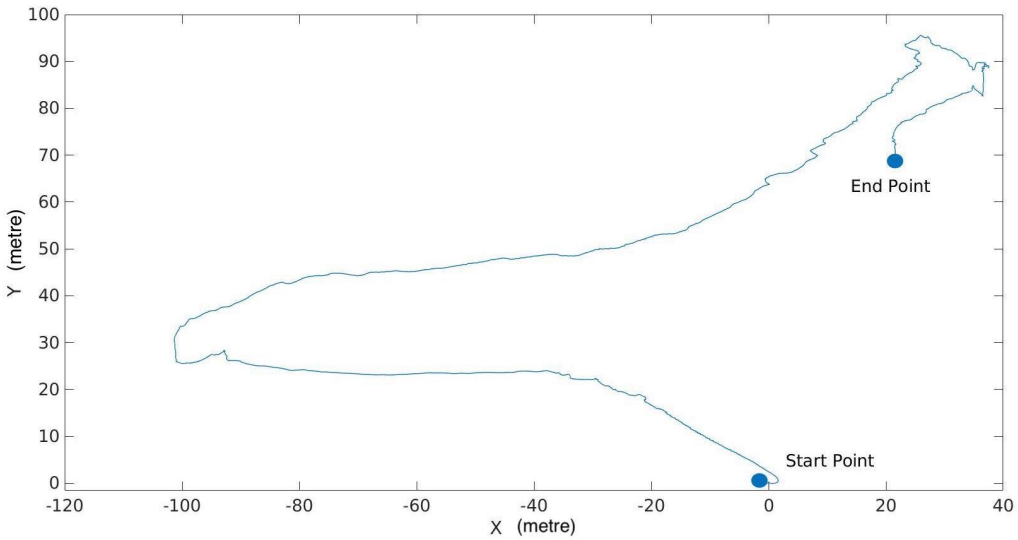


Figure 11.16 T265 Figure-8 Shape (1st test)

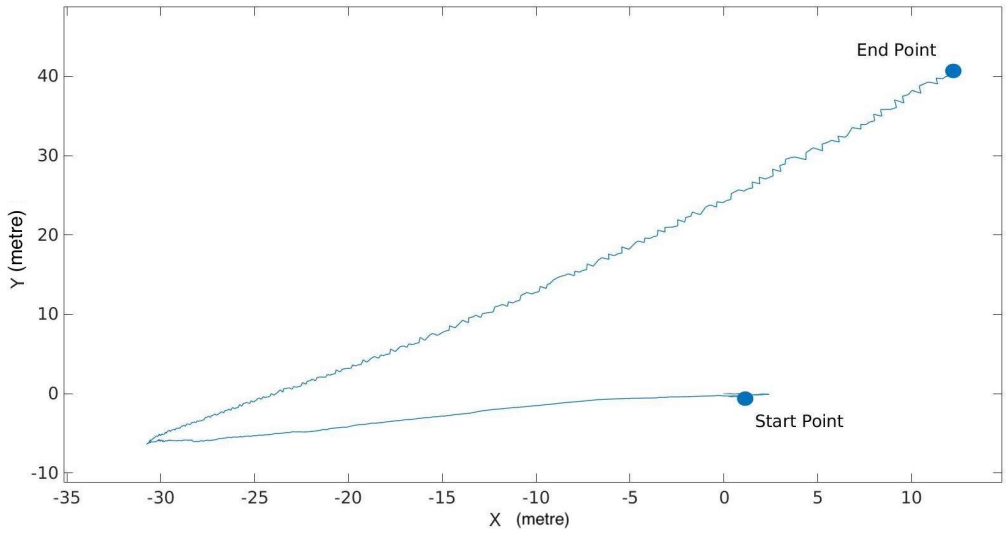


Figure 11.17 T265 Figure-8 Shape (2nd test)

The results without reference path are presented in Figure 11.18 to Figure 11.21. The vehicle tries to follow the triangle and the circle shapes and return to the start point at the end. The error ratio is shown in the figure caption for these results.

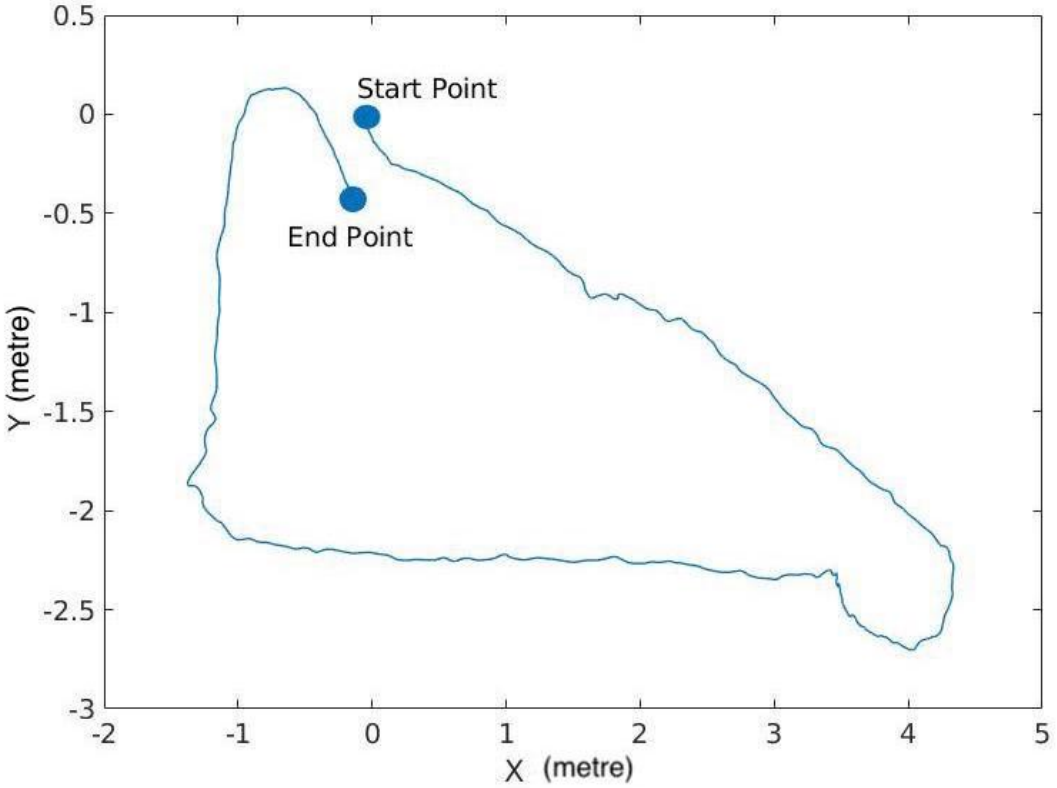


Figure 11.18 IVO-M Triangle without Reference, $e_r = 2.7\%$

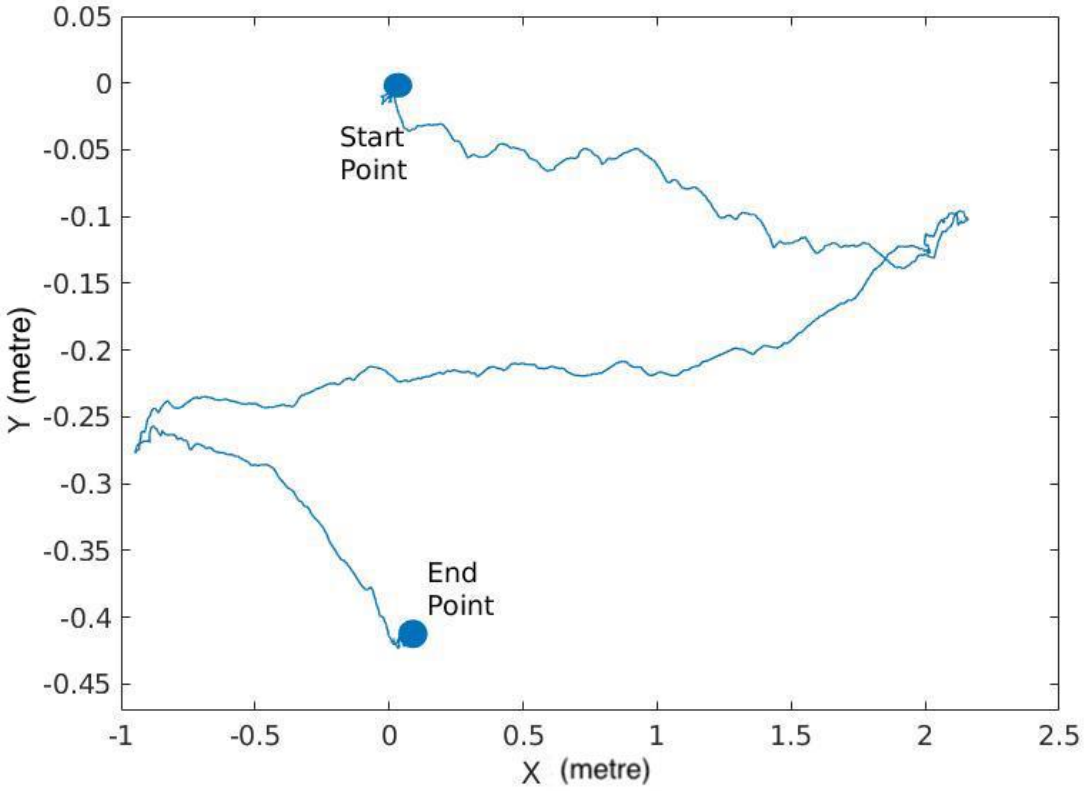


Figure 11.19 T265 Triangle without Reference

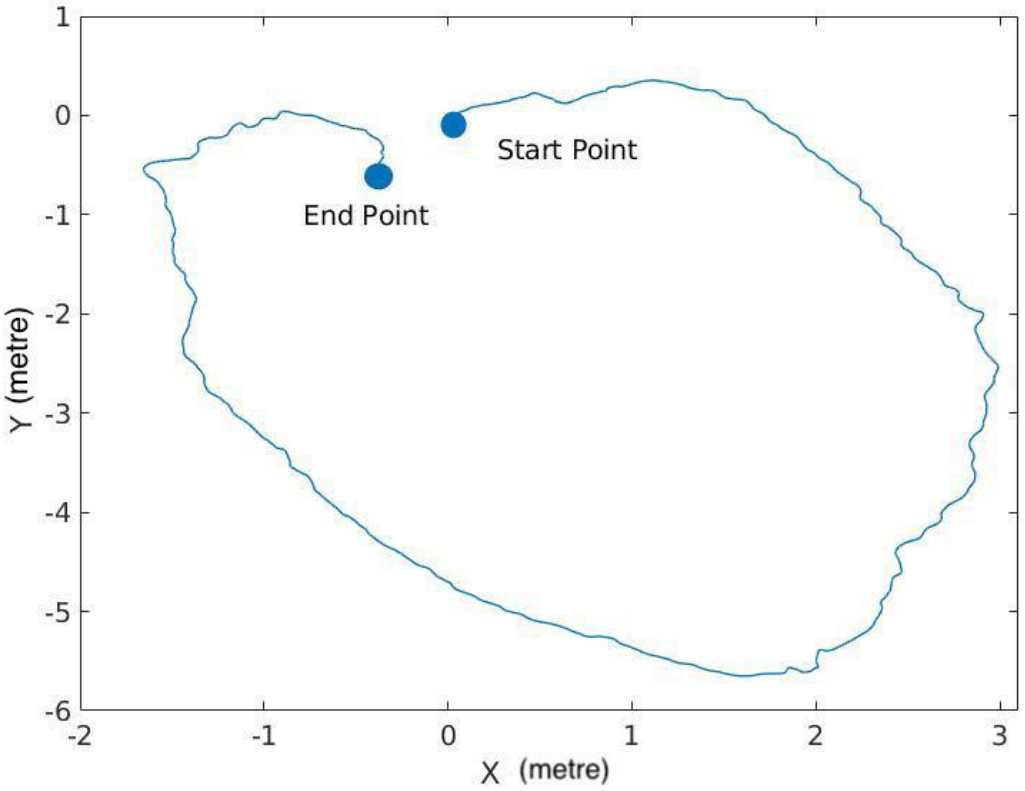


Figure 11.20 IVO-M Circle without Reference, $e_r = 3.9\%$

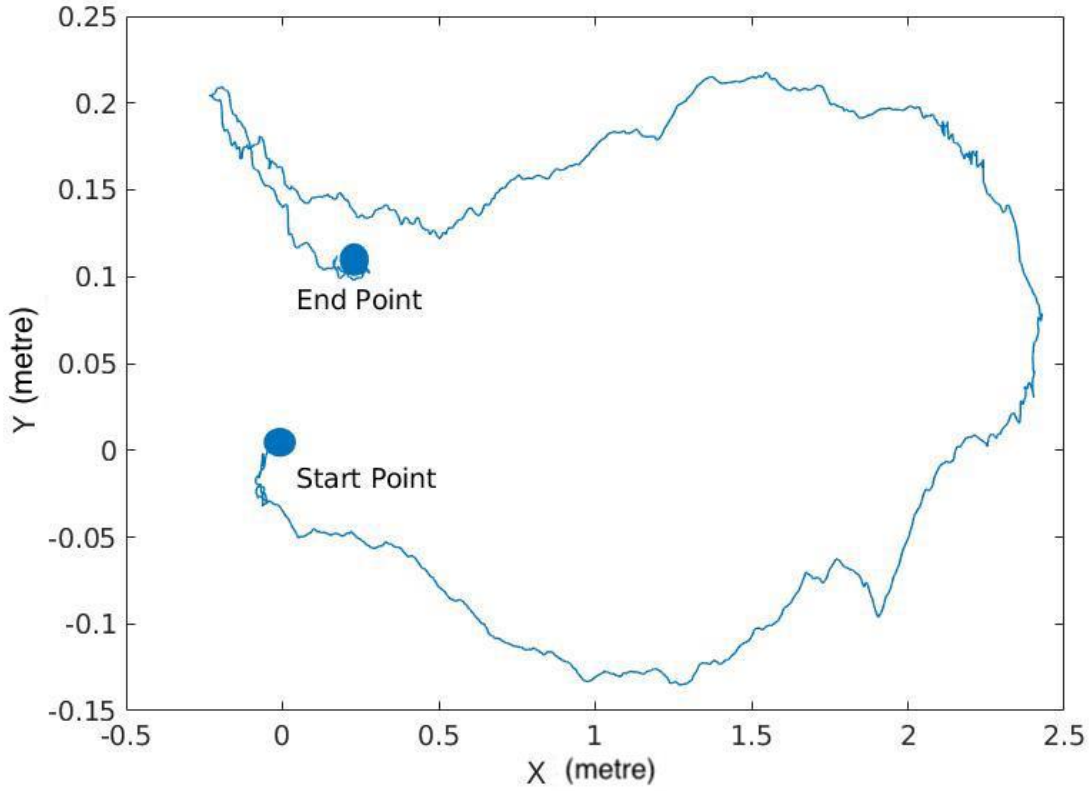


Figure 11.21 T265 Circle without Reference

11.3.1. Discussion

From the results, it can be seen that the performance of the proposed IVO method is significantly better than that of the T265. Actually, the T265 failed totally in terms of navigation. The reasons for this are: Firstly, the navigation tests were in a sparse environment and the V-SLAM algorithm used on the T265 requires a dense texture environment, which can provide enough feature points; Secondly, the experiments were conducted in a laboratory with a complex electromagnetic environment, and the inertial sensors in the modified ROV were more robust than the ones in the T265. Hence, in most of the T265 results, the estimated trajectory lost the correct orientation and the linear estimation from the T265 was poor.

Comparing the results with the reference in Figure 11.6 and Figure 11.7, the vehicle was expected to travel along a square reference path, and the estimated trajectories also show a square shape. Because the vehicle was held by an operator, it could not follow the reference path exactly; it was particularly difficult at the corners for the operator to turn the vehicle exactly along the marked path. In Figure 11.6, at the start of the test, the estimated trajectory followed the reference line exactly but by the time it had passed the 3rd corner, the estimated trajectory had a relatively large offset. This indicates that there is an accumulated error resulting in the offset and the corresponding error ratio is 5.6% in the first test, 2.3% in the second test.

In Figure 11.10 and Figure 11.11, the vehicle followed a triangular path. In these tests, the estimated position almost returned to the origin point and the estimated trajectory followed the reference line closely. At each corner, the vehicle had to make the turn in advance, nevertheless the error ratios are small at 0.51% and 0.90%.

In Figure 11.14 and Figure 11.15, the vehicle followed a path in a figure-8 shape within the rectangle set out by the straps. The estimated trajectories show an approximately figure-8 shape within the rectangle, although there is some offset. The error ratio is 3.15% in the first test, and 4.15% in the second test.

The tests without the reference shape used paths in the form of triangle and circle shapes. In Figure 11.18, the estimated trajectory at each corner has a semicircle shape because the operator held the vehicle at a distance to avoid the monocular camera, hence when making a turn, the camera in the vehicle travelled along a semicircle. In Figure 11.18 and Figure 11.20, the estimated trajectories show a triangle and a circle respectively. The trajectories are almost closed, however, due to the accumulated error, there are acceptable offsets of 2.7% and 3.9%.

Compared with the T265, for which the documentation declares the drift error is less than 1% (Realsense, 2018), the novel IVO method has a much better performance in the texture-sparse and complex electromagnetic environment.

11.4. Experiment Results for the IVO-S Method

The estimated trajectories from the IVO-S method are again plotted in 2D figures, shown from Figure 11.22 to Figure 11.30. The errors and length of each test based on the IVO-S method are presented in Table 11.2. The reference path is drawn in black dotted lines in these results. For the results given in Table 11.2 the experiments were conducted a total of nine times: the triangle shape was tested four times, the square shape was tested three times, and the figure-8 shape was tested twice.

Shape	Distance(metre)	Error Ratio
Triangular 1st	12.9853	0.0244
Triangular 2nd	14.3150	0.0080
Triangular 3rd	11.4760	0.0183
Triangular 4th	12.1408	0.0115
Square 1st	15.6336	0.0139
Square 2nd	16.5278	0.0118
Square 3rd	15.6826	0.0189
Figure-8 1st	18.3097	0.0161
Figure-8 2nd	20.3615	0.0054

Table 11.2 Error Ratio and Distance of each Test Using IVO-S Method

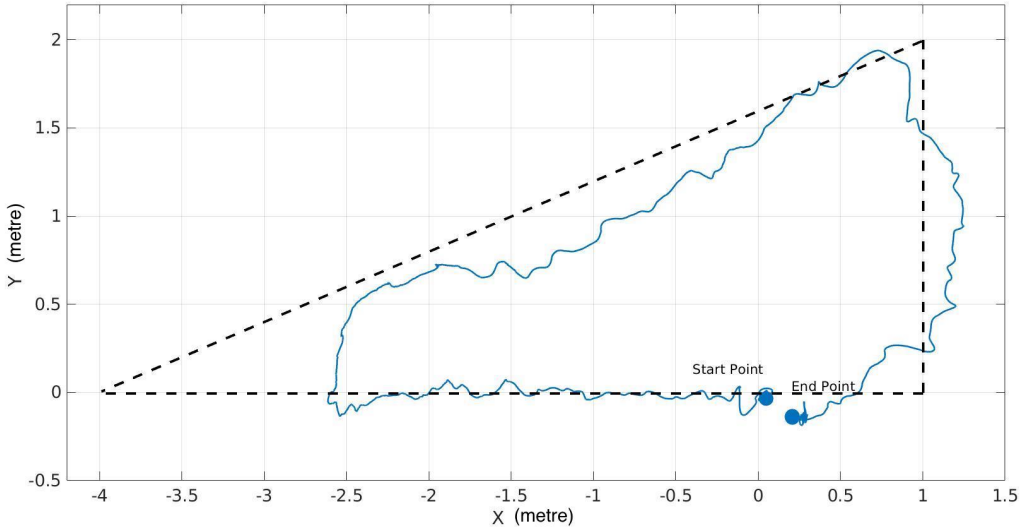


Figure 11.22 IVO-S Triangular 1st, error ratio: 0.0244

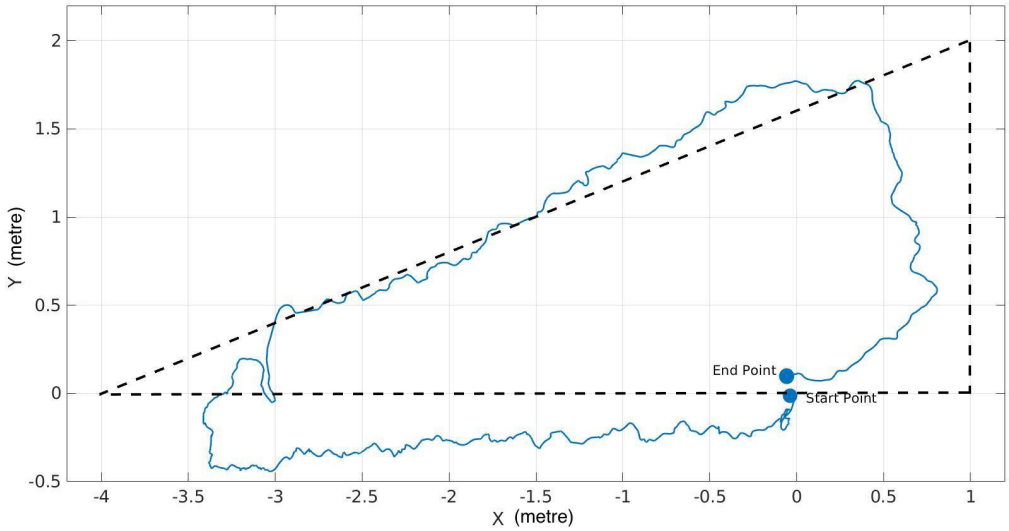


Figure 11.23 IVO-S Triangular 2nd, error ratio: 0.0080

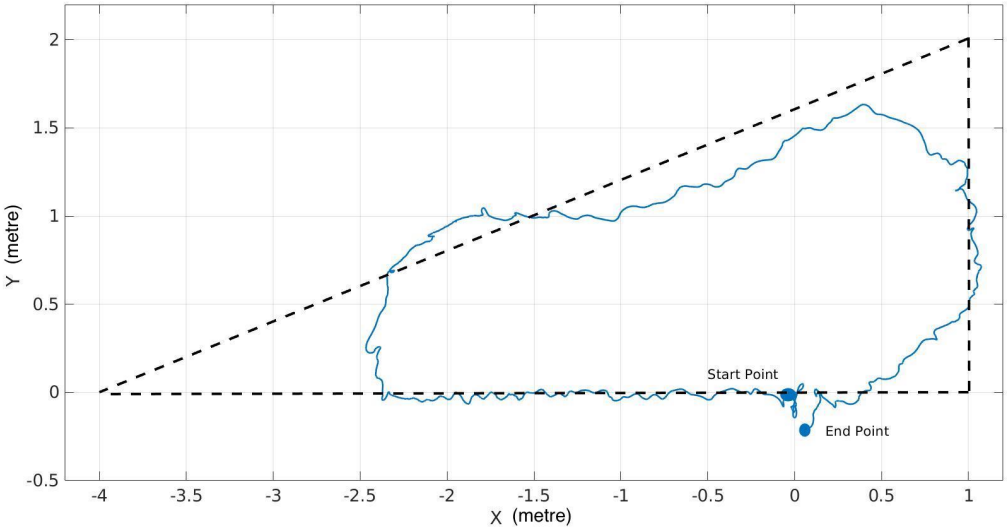


Figure 11.24 IVO-S Triangular 3rd, error ratio: 0.0183

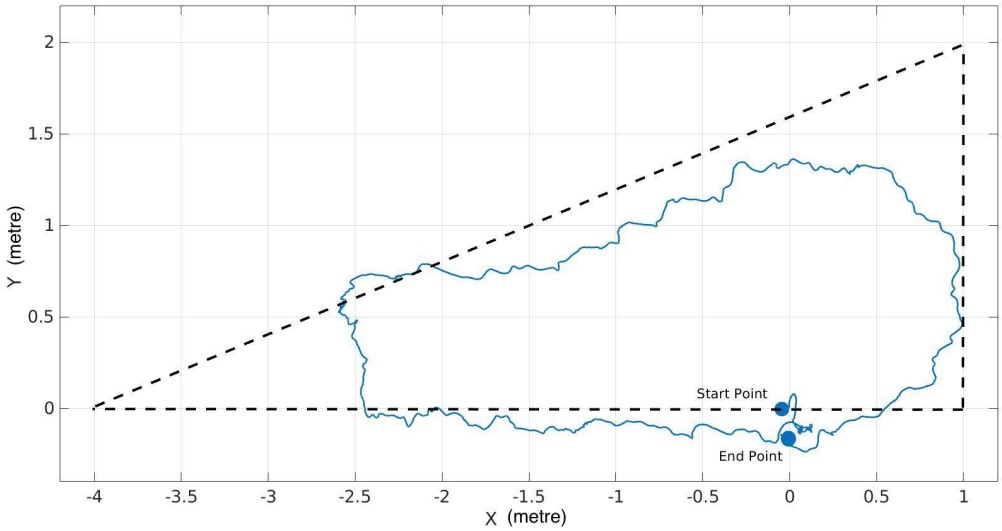


Figure 11.25 IVO-S Triangular 4th, error ratio: 0.0115

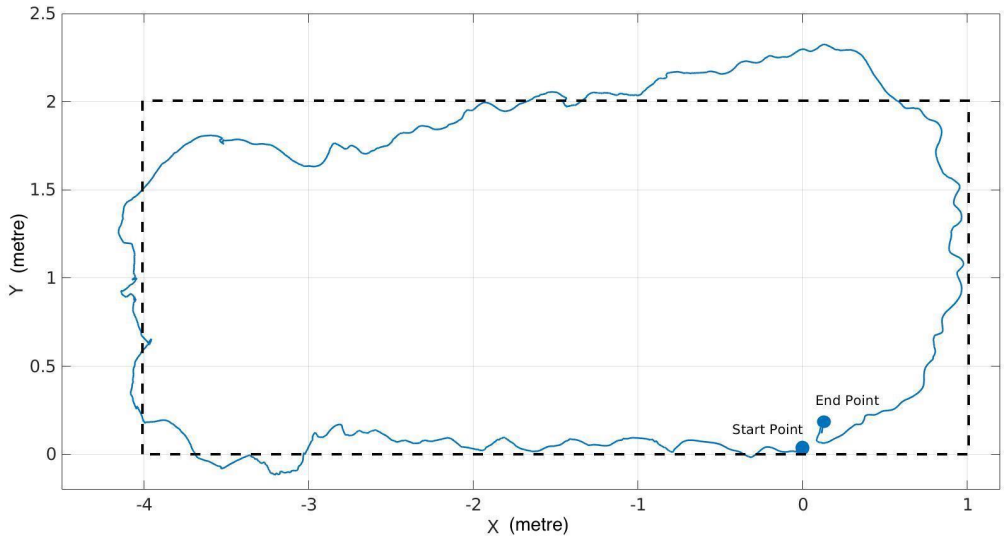


Figure 11.26 IVO-S Square 1st, error ratio: 0.0139

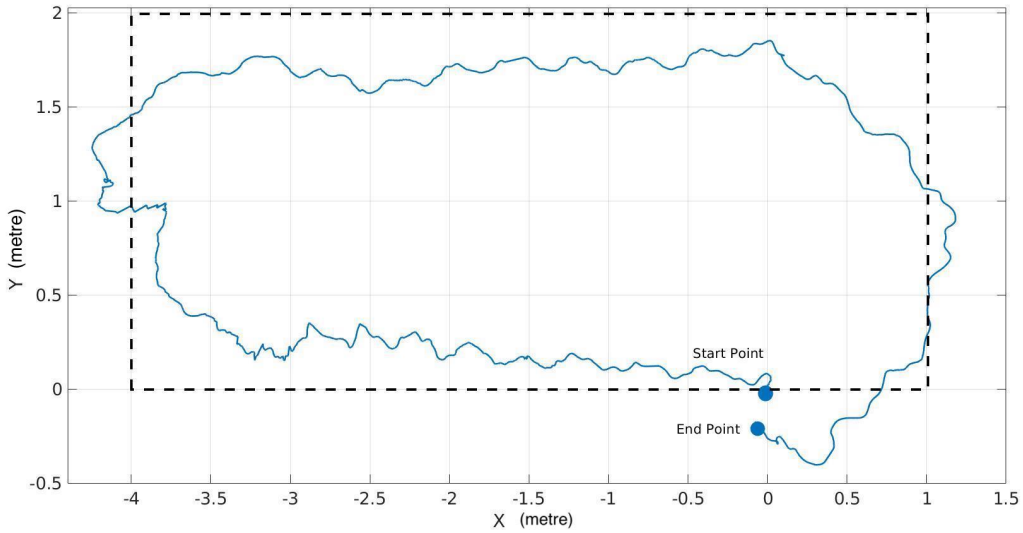


Figure 11.27 IVO-S Square 2nd, error ratio: 0.0118

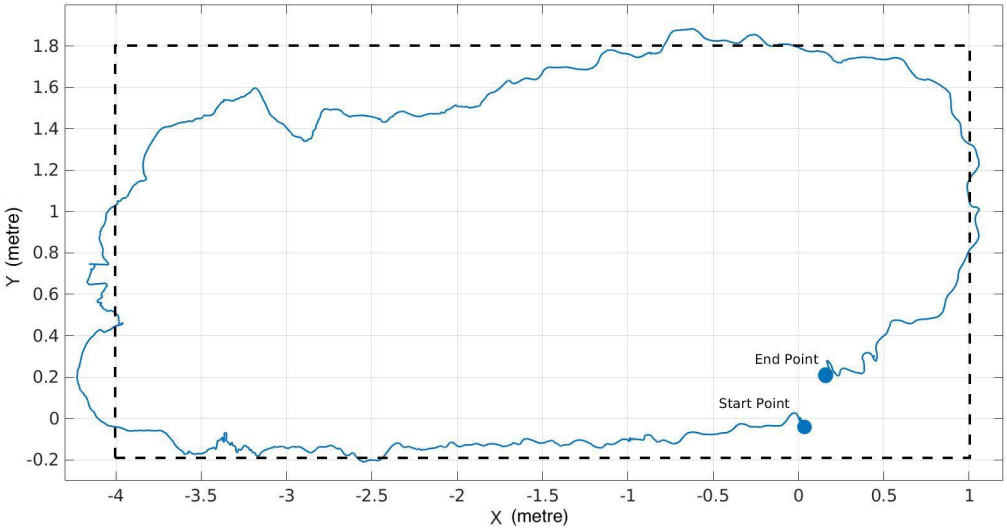


Figure 11.28 IVO-S Square 3rd, error ratio:0.0189

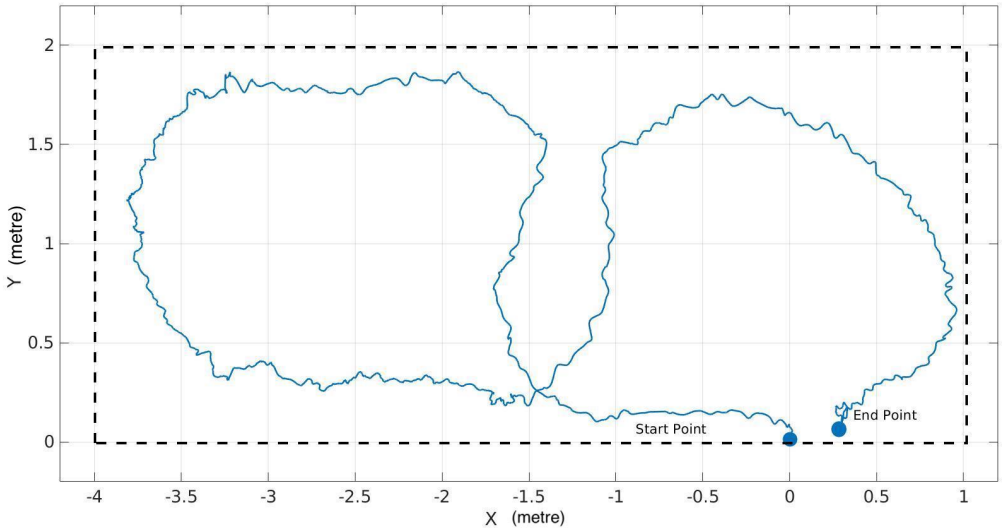


Figure 11.29 IVO-S Figure-8 1st, error ratio: 0.0161

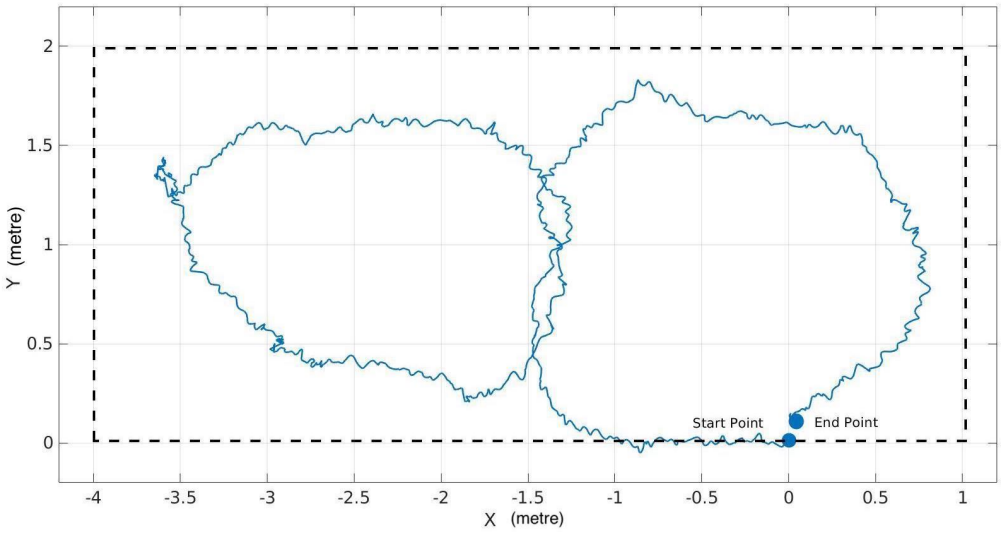


Figure 11.30 IVO-S Figure-8 2nd, error ratio: 0.0054

The T265 camera pose was derived from the V-SLAM algorithm running in the T265 camera VPU. During each test, the estimated trajectory from the T265 camera was recorded. The results of the T265 camera are presented from Figure 11.31 to Figure 11.39. Because the estimated pose has a huge drift relative to the real pose, the reference path cannot be meaningfully drawn in these figures.

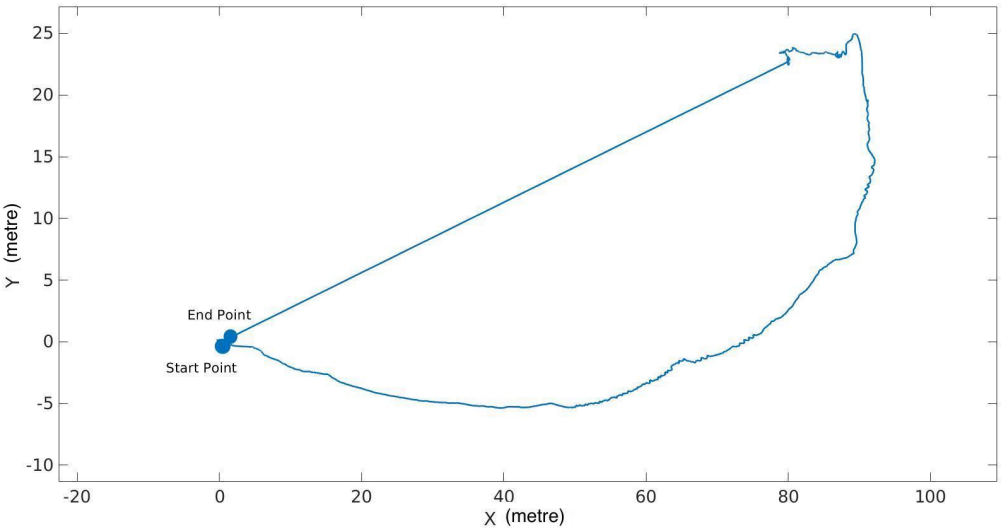


Figure 11.31 T265 Triangular 1st

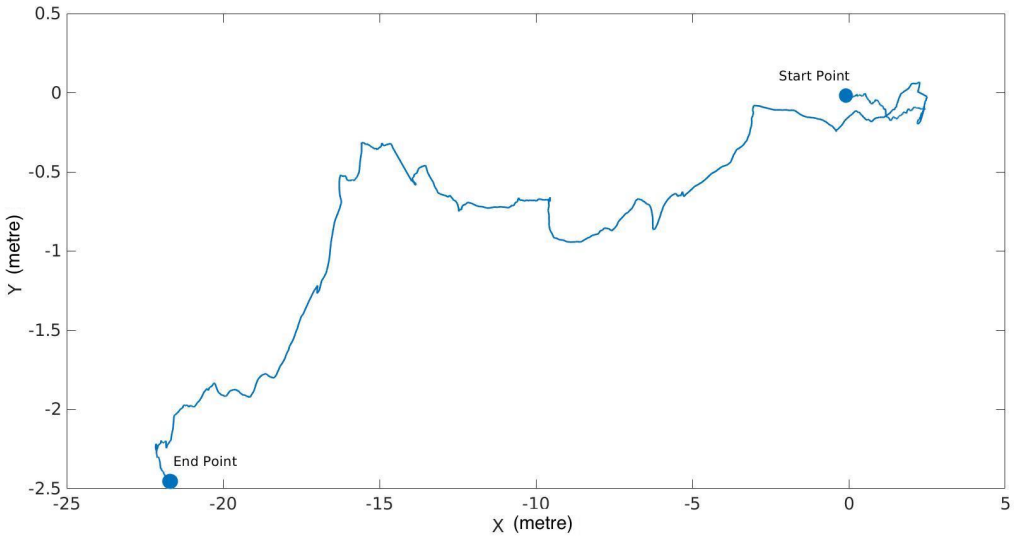


Figure 11.32 T265 Triangular 2nd

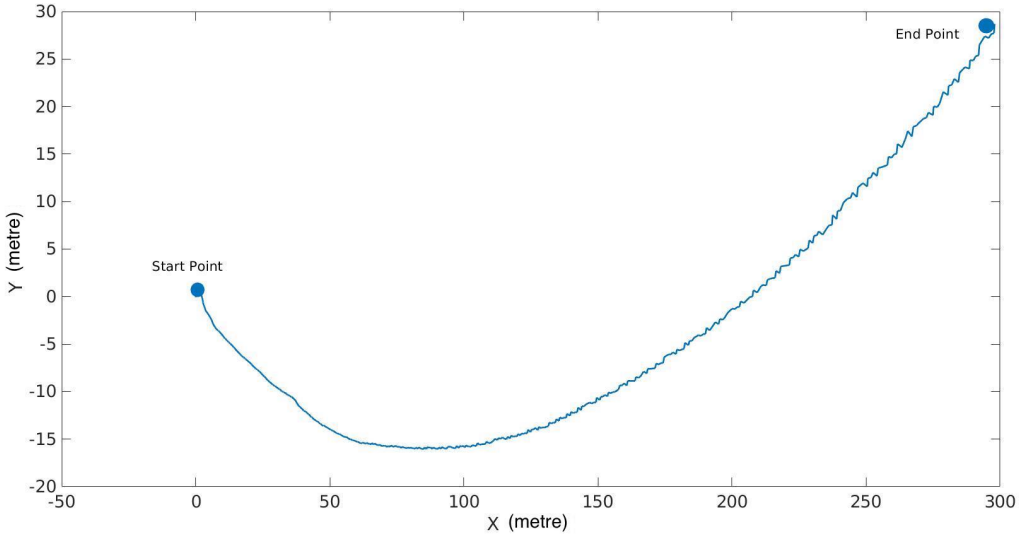


Figure 11.33 T265 Triangular 3rd

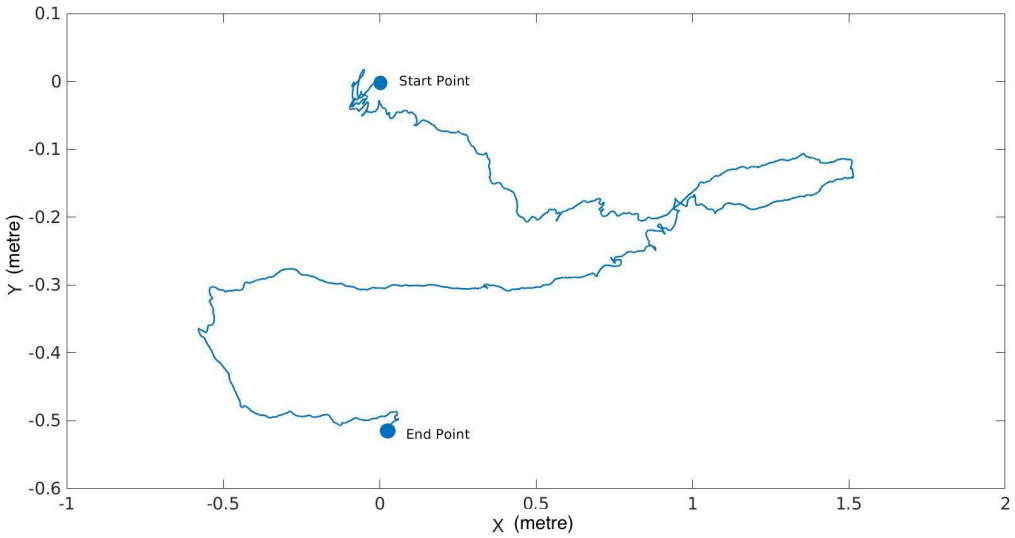


Figure 11.34 T265 Triangular 4th

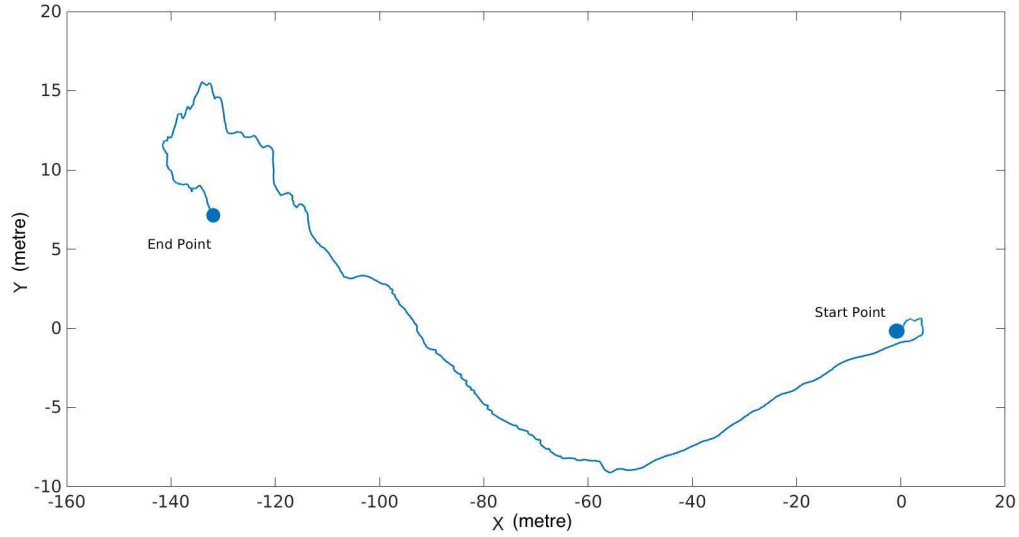


Figure 11.35 T265 Square 1st

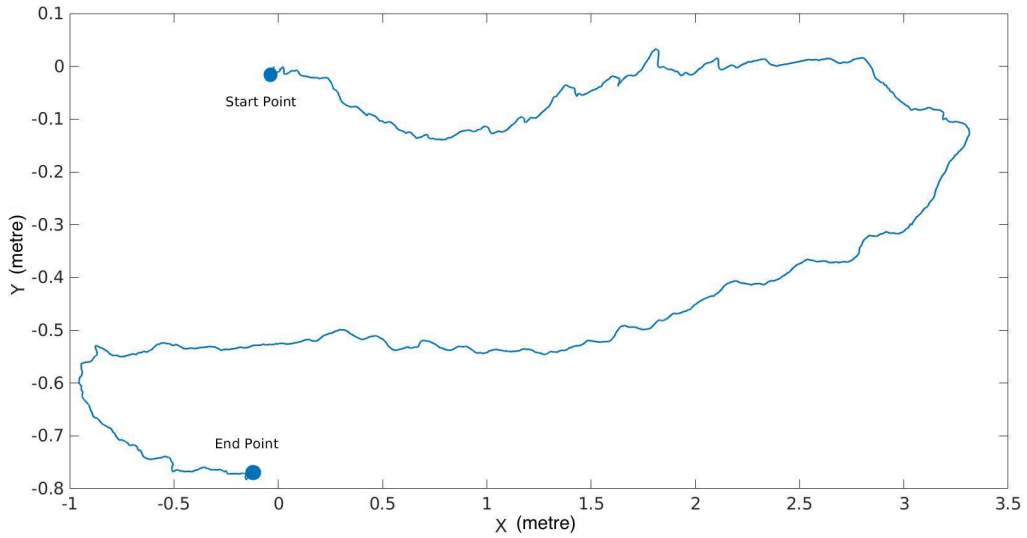


Figure 11.36 T265 Square 2nd

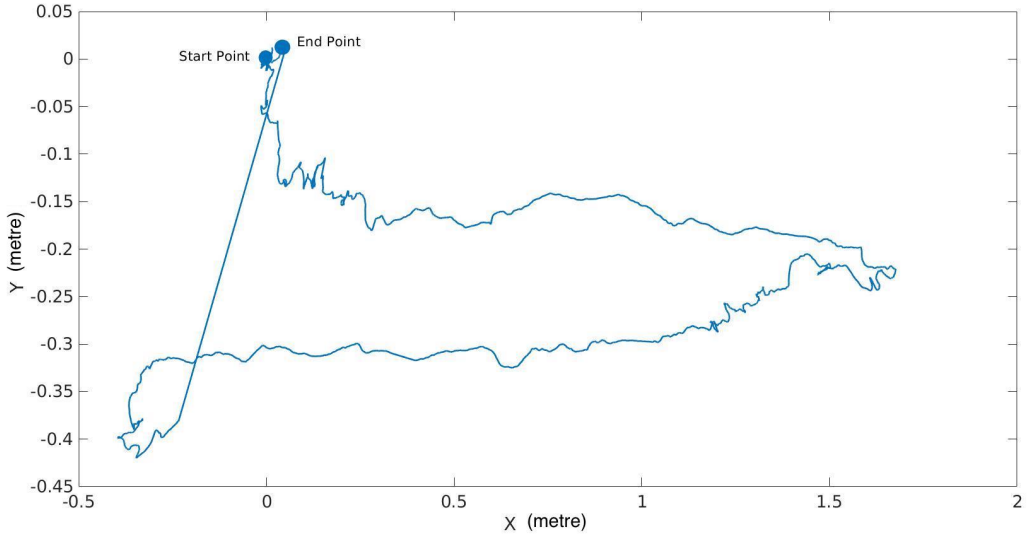


Figure 11.37 T265 Square 3rd

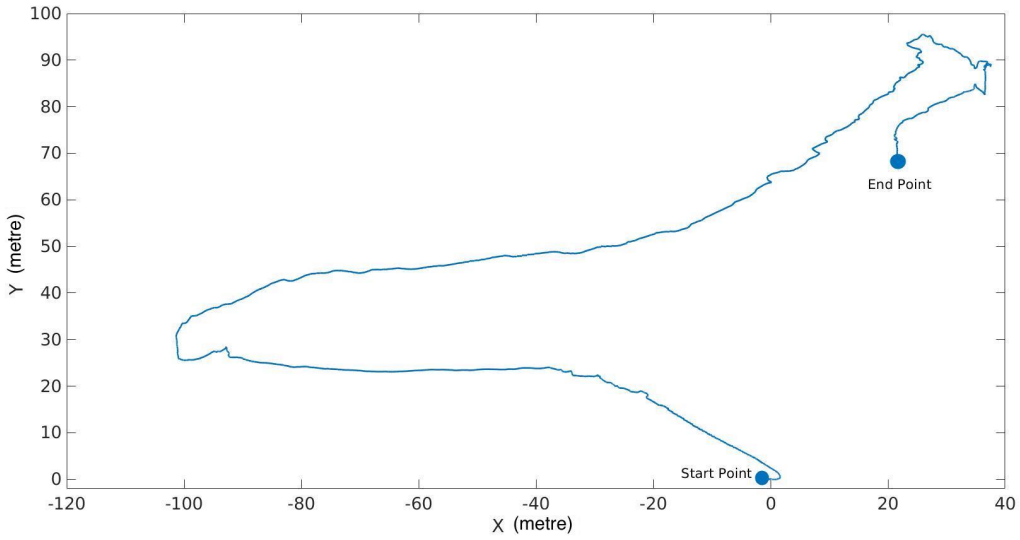


Figure 11.38 T265 Figure-8 1st

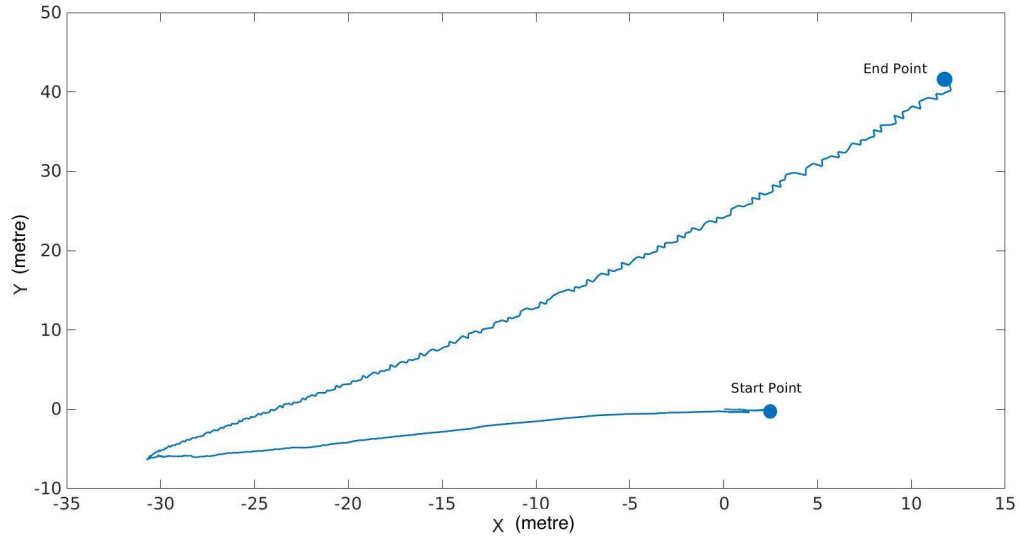


Figure 11.39 T265 Figure-8 2nd

11.4.1. Discussion

According to the experimental results, the IVO-S method estimated the trajectories accurately. However, the trajectories were not estimated successfully by the T265 camera. Again, the reason is that the towing tank is a sparse context and the electromagnetic environment in the laboratory is complex. These two factors hampered the performance of the low-cost IMU development kit. Furthermore, the T265 camera is designed to work in a dense-feature environment. In most of the T265 results, it failed not only in orientation estimation, but also in linear motion calculation. During these practical experiments, the operator held the vehicle with two half metre sticks. Hence, when they took a step, the camera would have a small circular motion. It resulted in estimated trajectories that are not straight lines.

From Figure 11.22 to Figure 11.25, the vehicle was forced to follow a triangular path by the operator. Because of the narrow space in each corner, the operator could not follow the reference path exactly. In this way, the vehicle had to make a turn in advance rather than moving along sharp angles. In the triangular-shape tests, the estimated position almost returned to the original point, and the error ratios are about 1%.

From Figure 11.26 to Figure 11.28, the IVO-S was evaluated in square reference paths. It is obvious that the estimated trajectories approximated a square shape. As mentioned before, because the vehicle was held by an operator, it could not follow the reference path exactly, and it was particularly hard at the corners for the operator to turn the vehicle exactly along the vertical angles. Along some segments of the estimated trajectories, the vehicle is estimated to go across the reference path. Since the operator moved close to the reference square holding the vehicle with the half metre sticks, the vehicle may have crossed the reference line sometimes. In Figure 11.26 and Figure 11.27, the IVO-S has about 1% error ratio. In Figure 11.28, the error ratio is less than 2%.

The vehicle was operated to follow a figure-8 shape within the reference rectangle in Figure 11.29 and Figure 11.30. The IVO-S method drew approximately figure-8 shape trajectories. Using the evaluation methods, the error ratio is 1.61% in Figure 11.29, and 0.54% in Figure 11.30. These results also illustrate that the method has the relatively high accuracy estimation, even among visual SLAMs, according to the KITTI benchmark (Menze and Geiger, 2015).

The performances of the T265 camera are presented from Figure 11.31 to Figure 11.39. In the sparse-feature and complex electromagnetic environments, the T265 camera failed in navigation, although it is expected to have less than 1% drift error (Realsense, 2018).

11.5. Comparisons between the IVO Methods and other Visual Navigation Methods

The proposed IVO methods were compared with four different Visual SLAM algorithms: ORB-SLAM2, SVO, VINS-Mono, and OKVIS. These algorithms were evaluated on the same dataset (a triangular shape) with the underwater calibrated camera parameters and coordinate transformation matrices. The ORB-SLAM2 method has been implemented for navigating AUVs in (Zhang, Ila and Kneip, 2018), and for mapping an underwater cave in (Weidner et al., 2017). The VINS-Mono and OKVIS algorithms, integrating inertial sensors, have been used to track the pose of a camera in an underwater environment (Rahman et al., 2018*a*,*b*). These methods can work well in feature-dense environments, but they work less successfully in sparse-feature and complex magnetic environments. The SVO is a direct visual odometry method using invariant intensity values in images to recover the camera motion, but it requires at least 100 points to initialise. Therefore, when the SVO was tested using the recorded dataset, it became stuck in the initialisation stage and failed. The other methods are feature-based methods, so their performances rely on the quality of feature points. In these methods, descriptors are derived to

match corresponding points. The sparse-feature environment, with few high-quality feature points, resulted in the failure of these methods. More specifically the ORB-SLAM2 monocular algorithm was hardly able to complete the initialisation process, as shown in Figure 11.40, and lost feature points easily in the sparse-feature underwater environment. Hence, the trajectory estimated by the ORB-SLAM2 was not complete, as shown in Figure 11.41. Meanwhile, the trajectory estimated by the IVO-M is shown in Figure 11.42, and by the IVO-S is shown in Figure 11.43.

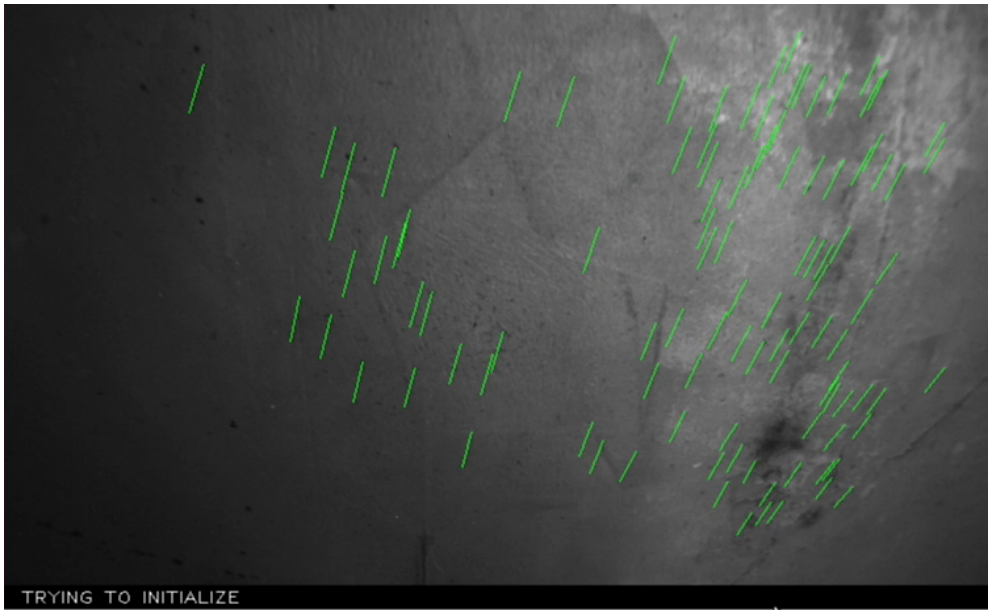


Figure 11.40 ORB-SLAM2 Trying to Initialise in Sparse-Feature Environment

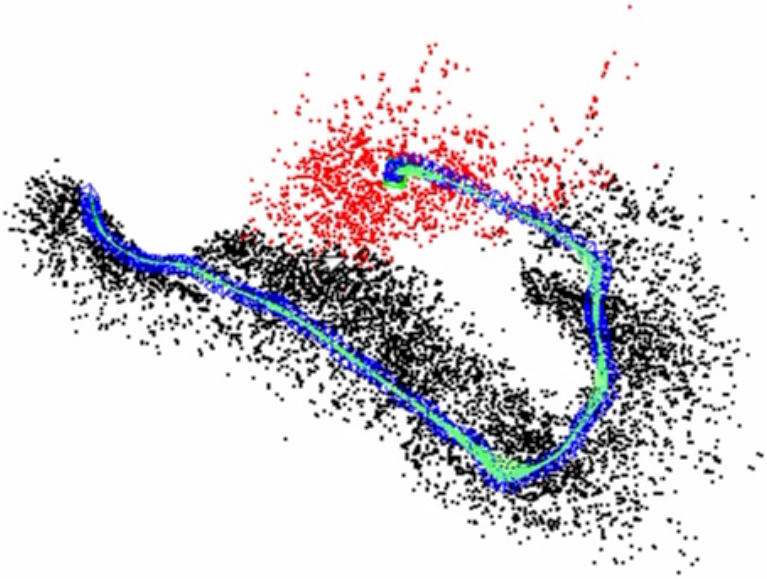


Figure 11.41 The Trajectory Estimated by ORB-SLAM2 (The green line is the estimated trajectory, black dots are measured feature points, red dots are active feature points.)

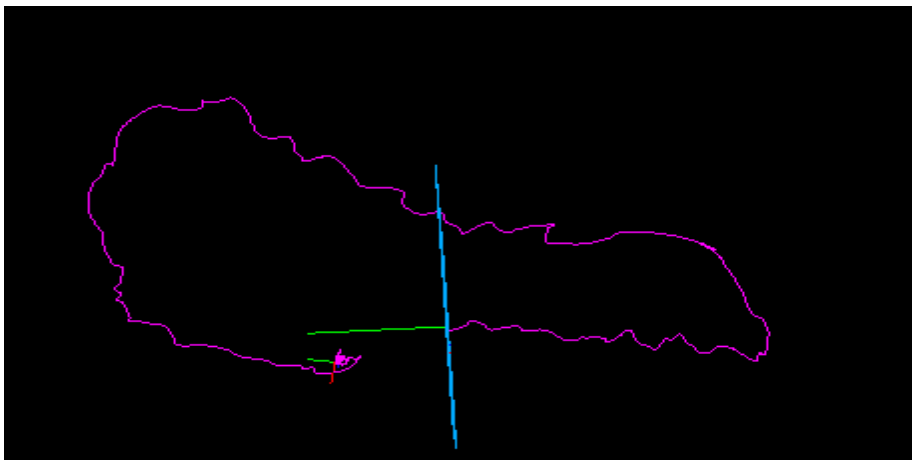


Figure 11.42 The Trajectory Estimated by the IVO-M (The violet line is the estimated trajectory.)

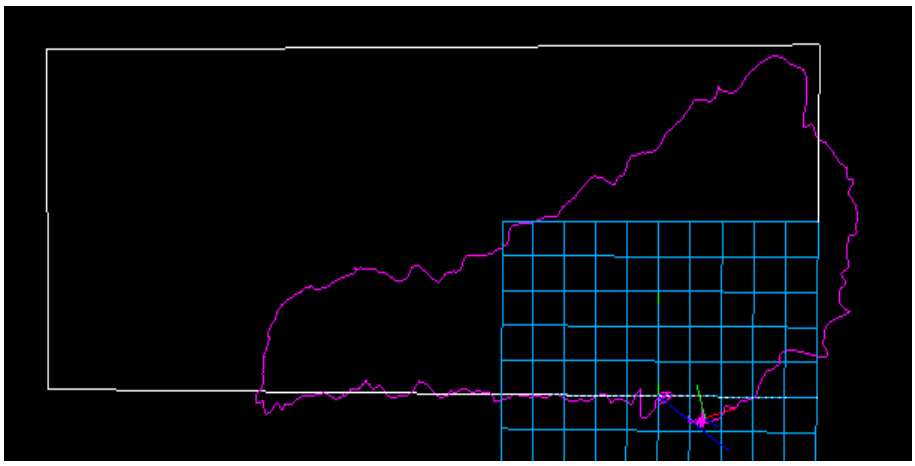


Figure 11.43 The Trajectory Estimated by the IVO-S (white line presents the reference path, every blue grid is 0.2 metre, and the violet line is the estimated trajectory)

Running the dataset on the VINS-Mono and OKVIS algorithms was unsuccessful. The complex magnetic field in the towing tank caused huge errors on the IMU data, because low-cost IMU kits usually use magnetic vectors to correct orientation. The main issue was that the limited feature points could not constrain the drift error from the IMU acceleration data, so that the final drift error was very large. Figure 11.44 and Figure 11.45 illustrate that they could hardly detect any reliable feature points in the sparse-feature underwater environment. Because the T265 V-SLAM is an IMU-Aided VO, it along with the VINS-Mono, OKVIS, and V-SLAM have a common problem: unconstrained drift error, as shown in Figure 11.44 and Figure 11.46.

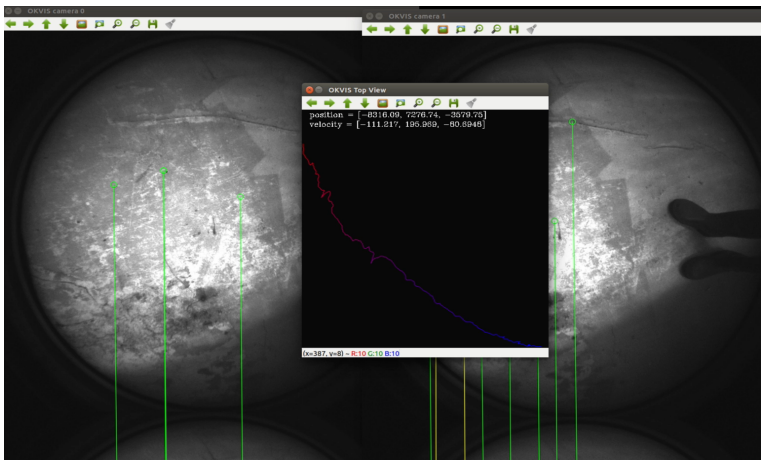


Figure 11.44 OKVIS with Few Matched Feature Points and Huge Drift Error

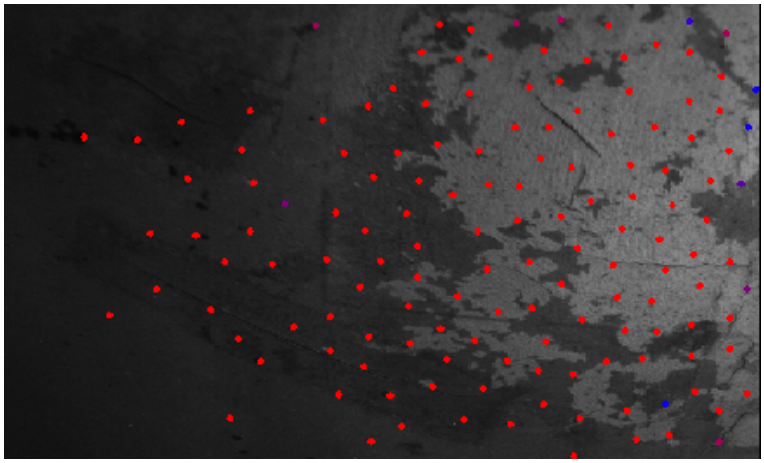


Figure 11.45 VINE-Mono Trying to Extract Feature Points

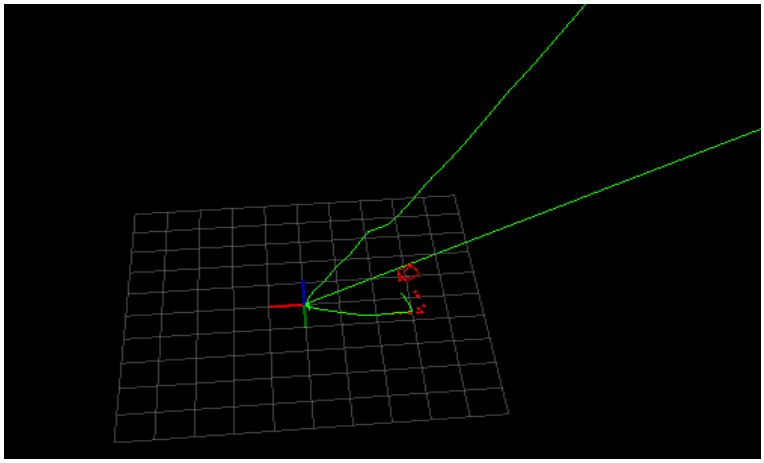


Figure 11.46 VINS-Mono with Huge Drift Error

In contrast to the above methods, the IVO methods utilise the FAST algorithm to detect feature points and the Optical Flow algorithm to track these identified feature points, rather than using complicated descriptors to match feature points. Furthermore, integrating the sonar and inertial sensor information, IVO methods can skip the initialisation stage, which is commonly applied in

monocular visual navigation methods. Hence, these characteristics allow the IVO method to perform much more successfully than other techniques in underwater sparse-feature environments, as shown in Figure 11.42 and Figure 11.43.

11.6. Summary

This chapter presented the results of the quantitative evaluations. According to these results, the V-SLAM running on the T265 camera and other visual navigation methods failed on the customised datasets, because of the sparse-feature and complex electromagnetic environments. The error rate of IVO-M is about 3% to 4%, and the method is limited by the "seafloor locally flat" assumption. The IVO-S has an error rate of less than 2%, and is free of the assumption. With the navigation work completed, the conclusions and future work will be presented in the next chapter.

Chapter 12. Conclusion & Future Work

12.1. Conclusion

In the thesis, four controllers have been developed and evaluated to find reliable controllers for UUVs. They are: the improved SMC, the customised FLC, the proposed BC, and the PID controller. Subsequently, three low-cost underwater visual navigation systems have been presented to find low-cost but high-accuracy underwater visual navigation methods. They are: the ArUco underwater navigation, the Integrated Visual Odometry with Monocular Camera (IVO-M), and the Integrated Visual Odometry with Stereo Camera (IVO-S). Overall, the objectives presented in Chapter 1 are met,

- Modelling the UUVs, and designing controllers. In Chapter 4, the generic UUV model was described. Besides, the model for the specific ROV (VideoRay Pro 3) was also constructed with the dynamic equations decoupled into three subsystems: forward speed, diving, and steering. In Chapter 5, the improved SMC, the customised FLC, the proposed BC, and the PID controller have been derived.

- Verification of the controllers via simulations and physical experiments. In Chapter 6, the four different control strategies were then simulated and physically tested for heading control under different conditions. In the simulations, a white noise was used to simulate current disturbance. These four controllers were used to guide the vehicle to desired heading angles and to follow a designed path with the Way-Point-Guidance algorithm under noise-free and white noise conditions. The physical experiments were conducted in the towing tank. Similar to the simulations, the four controllers were implemented to control the vehicle to desired angles under different conditions. The noise in the experiments was generated by the wave makers in the towing tank. Meanwhile, these controllers were tested on both the original and the modified vehicle to verify their robustness to different hydrodynamic models. The results show that the improved SMC performed better than others with small settling time, overshoot, average error and variance of error;

- Building a platform to collect underwater data. A platform is operated in underwater environments to collect data from optical sensors and inertial sensors. In Chapter 3, a water-tight tube within multiple sensors was installed on the bottom of the Vidoeray Pro 3.

It worked as a platform to collect data in underwater environments, as mentioned in Chapter 11;

- Development of a low-cost underwater visual navigation systems. In Chapter 7, the camera model and underwater calibration were presented first. Subsequently, in Chapter 8, the ArUco underwater navigation system was discussed. It was developed based on the ArUco marker detection algorithm (Garrido-Jurado et al., 2014). The ArUco markers were positioned on the bottom of the towing tank ahead of the testing. Through use of the marker detection algorithm, pose of a camera in captured marker coordinates can be known. Because the positions of the markers are known, the camera poses in global coordinates can be obtained. In Chapters 9 and 10, two Integrated Visual Odometry (IVO) methods have then been investigated. Unlike the ArUco underwater navigation method, the IVO methods implemented the FAST algorithm and the Optical Flow algorithm to detect and track feature points in underwater environments. In the IVO with Monocular camera (IVO-M), the 3D reconstruction of feature points is conducted by fusing information from a depth sonar. Translation vectors are derived by linear estimation with integrating data from multiple inertial sensors. The rotation vectors from the inertial sensors were corrected by a nonlinear optimisation algorithm. In the IVO with Stereo Camera, 3D points are reconstructed by a stereo camera. The depth information from the sonar is used to select reliable feature points by using a Gaussian filter. The results from linear estimations were corrected by a fusion algorithm before processing a nonlinear regression algorithm. Therefore, the IVO methods do not require any artificial markers in underwater environments;
- Verification of these navigation methods with the collected data. In Chapter 11, after evaluation on the recorded datasets, the IVO methods performed successfully in sparse-feature underwater environments, while other generic visual navigation methods failed in the same conditions.

There are two main contributions in the control part.

- A comparison of four different controllers based on three different control algorithms is made. These controllers are evaluated not only through simulations, but also in physical experiments with quantified disturbances and different hydrodynamic models. The performances from the controllers are compared and discussed based on their own theories.
- An improved SMC and a customised second-order FLC have been developed. The proposed SMC benefits from the advantages of both linear and nonlinear control, and has better performance compared to the traditional Backstepping nonlinear controller. The FLC is built into second-order structure, with customised rules set for the VideoRay Pro 3.

The novelties in the visual navigation part are:

- The ArUco markers have been implemented in underwater navigation. The position error is bounded in this method. However, because of poor illumination, and large distances, the ArUco Marker detection algorithm may fail sometimes. That may cause discontinuity in some segments of the estimated trajectory;
- The novel IVO-M system has been developed in this thesis. This is a low cost underwater navigation system that can work in sparse feature environments with a single camera. The accuracy of the system (error ratio about 3% to 4%) is acceptable. However, it is limited by the "seafloor locally flat" assumption, and has unbounded position error;
- The novel IVO-S system has also been developed in the thesis. The system is a low cost navigation solution with high accuracy (error ratio is less than 2%). It can work in a sparse-context environment and is free of the IVO-M's assumption. However, the position error is unbounded in the system.

With regard to the control development, the SMC method is expected to work in underwater environments even in the presence of large disturbances. In terms of the navigation part, the ArUco underwater navigation system is expected to work in environments where the markers can be placed beforehand, such as for inspection of subsea installations and path following missions. The IVO methods could be implemented for missions where the vehicle travels near the seabed and covering limited space.

12.2. Future Work

Although the controllers and visual navigation methods have been introduced in the thesis, there are still some aspects which can be improved.

- The position errors are unbound in the IVO methods. Therefore, a visual SLAM needs to be developed based on IVO methods, in order to bound the position error.
- The controller and navigation system are not combined in the thesis. In the future, the navigation system and the controller can be made to interact. In this case, the underwater environment will be sensed by the navigation system, and the controller will react according to the information from the navigation system.

Hence, in the future, an AI system for UUVs need to be developed. The visual navigation module, decision-making module, and a controller will communicate with each other to take appropriate action.

References

Agarwal, S. and Mierle, K. (2018), 'Ceres solver', Available at http://ceres-solver.org. Date accessed: 3 January 2020.

Alami, R., Chatila, R., Fleury, S., Ghallab, M. and Ingrand, F. (1998), 'An architecture for autonomy', *The International Journal of Robotics Research* **17**(4), 315–337.

Allotta, B., Costanzi, R., Ridolfi, A., Colombo, C., Bellavia, F., Fanfani, M., Pazzaglia, F., Salvetti, O., Moroni, D., Pascali, M. A. et al. (2015), 'The arrows project: adapting and developing robotics technologies for underwater archaeology', *IFAC-PapersOnLine* **48**(2), 194–199.

Ambros-Ingerson, J. A. and Steel, S. (1988), Integrating planning, execution and monitoring., *in* '1988 Association for the Advancement of Artificial Intelligence', AAAI Press, Saint Paul, USA.

Armstrong, B., Pentzer, J., Odell, D., Bean, T., Canning, J., Pugsley, D., Frenzel, J., Anderson, M. and Edwards, D. (2009), 'field measurement of surface ship magnetic signature using multiple auvs', *in* '2009 OCEANS', IEEE, Bermen, Germany.

Armstrong, B., Wolbrecht, E. and Edwards, D. (2010), Auv navigation in the presence of a magnetic disturbance with an extended kalman filter, *in* '2010 OCEANS', IEEE, Sydney, Australia.

Aschwanden, P., Baskaran, V., Bernardini, S., Fry, C., Moreno, M., Muscettola, N., Plaunt, C., Rijsman, D. and Tompkins, P. (2006), Model-unified planning and execution for distributed autonomous system control., *in* '2006 AAAI Fall Symposium: Spacecraft Autonomy', AAAI Press, Hyatt Regency Crystal City, USA.

Banks, H. T. (2013), *Modeling and control in the biomedical sciences*, Vol. 6, Springer Science & Business Media, Heidelberg, Germany.

Bayer, J. and Faigl, J. (2019), On autonomous spatial exploration with small hexapod walking robot using tracking camera intel realsense t265, *in* '2019 European Conference on Mobile Robots', IEEE, Prague, Czech.

Behnke, S. (2008), 'Humanoid robots-from fiction to reality?', *KI-Zeitschrift* **22**(4), 5–9.

Bellavia, F., Fanfani, M. and Colombo, C. (2017), 'Selective visual odometry for accurate auv localization', *Autonomous Robots* **41**(1), 133–143.

Bellingham, J. G. and Leonard, J. (1994), Task configuration with layered control, *in* '1994 IARP Workshop on Mobile Robots for Subsea Environments', Monterey, USA.

Bennett, S. (1993), *A history of control engineering, 1930-1955*, IET, London, UK.

Blanco, J.-L. (2010), A tutorial on se (3) transformation parameterizations and on-manifold optimization, Technical report, University of Malaga, Malaga, Spain.

Blidberg, D. R. (2001), The development of autonomous underwater vehicles (auv); a brief summary, *in* '2001 International Conference on Robotics and Automation', IEEE, Seoul, South Korea.

Bonin-Font, F., Oliver, G., Wirth, S., Massot, M., Negre, P. L. and Beltran, J.-P. (2015), 'Visual sensing for autonomous underwater exploration and intervention tasks', *Ocean Engineering* **93**, 25–44.

Bonnor, N. (2014), 'Principles of gnss, inertial, and multisensor integrated navigation systems–second editionpaul d. groves artech house, 2013, 776 pp isbn-13: 978-1-60807-005-3', *The Journal of Navigation* **67**(1), 191–192.

Bowes, P. (2010), 'Hydrodynamics Laboratory', Available at: https://www.ncl.ac.uk/engineering/about/facilities/marineoffshoresubseatechnology/hydrodynamics/#towingtank. Date accessed: 10 September 2019.

Braathen, N. and Sandford, A. (1986), Pipeline inspection by rov, *in* '1986 Submersible Technology', Springer, Heidelberg, Germany.

Bresina, J. L., Jónsson, A. K., Morris, P. H. and Rajan, K. (2005), Activity planning for the mars exploration rovers., *in* '2005 International Conference on Automated Planning and Scheduling', Monterey, USA.

Brooks, R. (1986), 'A robust layered control system for a mobile robot', *IEEE journal on robotics and automation* **2**(1), 14–23.

Brown, M. (2017), 'Eric the robot', Available at: https://commons.wikimedia.org/w/index.php?curid=56876106. Date accessed: 11 February 2020.

Brown, W. B. (1932), 'The brennan torpedo', *Royal United Services Institution. Journal* **77**(507), 592–597.

Bunce, J. A., Waddell, J., Boatman, J. and Carroll, J. (1999), 'Us naval oceanography and the role of auvs', *The International Hydrographic Review* **76**(1).

Burl, J. B. (1998), *Linear optimal control: H (2) and H (Infinity) methods*, Addison-Wesley Longman Publishing Co., Inc., Boston, USA.

Caccia, M. (2007), 'Vision-based rov horizontal motion control: Near-seafloor experimental results', *Control Engineering Practice* **15**(6), 703–714.

Caharija, W., Pettersen, K. Y., Bibuli, M., Calado, P., Zereik, E., Braga, J., Gravdahl, J. T., Sørensen, A. J., Milovanović, M. and Bruzzone, G. (2016), 'Integral line-of-sight guidance and control of underactuated marine vehicles: Theory, simulations, and experiments', *IEEE Transactions on Control Systems Technology* **24**(5), 1623–1642.

Carrasco, P. L. N., Bonin-Font, F. and Codina, G. O. (2016), Stereo graph-slam for autonomous underwater vehicles, *in* '2016 13th International Conference Intelligent Autonomous Systems', Springer, Padova, Italy.

Carreras, M., Batlle, J., Ridao, P. and Roberts, G. (2000), 'An overview on behaviour-based methods for auv control', *IFAC Proceedings Volumes* **33**(21), 141–146.

Carreras, M., Ridao, P., Garcia, R. and Batlle, J. (2006), 'Behaviour control of uuvs', *IEE Control Engineering Series* **69**, 67.

Chatzichristofis, S., Kapoutsis, A., Kosmatopoulos, E., Doitsidis, L., Rovas, D. and de Sousa, J. B. (2012), 'The noptilus project: Autonomous multi-auv navigation for exploration of unknown environments', *IFAC Proceedings Volumes* **45**(5), 373–380.

Chien, S., Knight, R., Stechert, A., Sherwood, R. and Rabideau, G. (1999), Integrated planning and execution for autonomous spacecraft, *in* '1999 IEEE Aerospace Conference', IEEE, Snowmass, USA.

Christ, R. D. and Wernli Sr, R. L. (2013), *The ROV manual: a user guide for remotely operated vehicles*, Butterworth-Heinemann, Oxford, UK.

Corke, P., Detweiler, C., Dunbabin, M., Hamilton, M., Rus, D. and Vasilescu, I. (2007), Experiments with underwater robot localization and tracking, *in* '2007 IEEE International Conference on Robotics and Automation', IEEE, Roma, Italy.

Corke, P., Lobo, J. and Dias, J. (2007), 'An introduction to inertial and visual sensing', *The International Journal of Robotics Research* **26**(6), 519–535.

Corradini, M. L., Monteriu, A. and Orlando, G. (2011), 'An actuator failure tolerant control scheme for an underwater remotely operated vehicle', *IEEE Transactions on Control Systems Technology* **19**(5), 1036–1046.

Cristi, R., Papoulias, F. A. and Healey, A. J. (1990), 'Adaptive sliding mode control of autonomous underwater vehicles in the dive plane', *IEEE journal of Oceanic Engineering* **15**(3), 152–160.

Das, J., Py, F., Maughan, T., O'reilly, T., Messié, M., Ryan, J., Sukhatme, G. S. and Rajan, K. (2012), 'Coordinated sampling of dynamic oceanographic features with underwater vehicles and drifters', *The International Journal of Robotics Research* **31**(5), 626–646.

Deegan, M., Dziedzic, A., Jiang, C., Moon, R., Pisarski, D. and Wunderly, R. (2019), 'Autonomous quadrotors for the 2019 international aerial robotics competition', *International Aerial Robotics Competition* .

Demetriou, G. A. (2011), *Mobile Robots-Current Trends*, IntechOpen, London, UK, chapter Mobile robotics in education and research, pp. 27–48.

Dukan, F., Ludvigsen, M. and Sørensen, A. J. (2011), Dynamic positioning system for a small size rov with experimental results, *in* '2011 OCEANS', IEEE, Santander, Spain.

ECA (2017), 'Subsea inspection & work', Available at: https://www.ecagroup.com/en/maritime/subsea-inspection-work. Date accessed: 21 January 2019.

Edwards, C. and Spurgeon, S. (1998), *Sliding mode control: theory and applications*, CRC Press, Boca Raton, USA.

Engel, J., Koltun, V. and Cremers, D. (2018), 'Direct sparse odometry', *IEEE transactions on pattern analysis and machine intelligence* **40**(3), 611–625.

Eustice, R. M., Pizarro, O. and Singh, H. (2008), 'Visually augmented navigation for autonomous underwater vehicles', *IEEE Journal of Oceanic Engineering* **33**(2), 103–122.

Eustice, R. M., Singh, H., Leonard, J. J. and Walter, M. R. (2006), 'Visually mapping the rms titanic: Conservative covariance estimates for slam information filters', *The international journal of robotics research* **25**(12), 1223–1242.

Fallon, M. F., Kaess, M., Johannsson, H. and Leonard, J. J. (2011), Efficient auv navigation fusing acoustic ranging and side-scan sonar, *in* '2011 IEEE International Conference on Robotics and Automation', IEEE, Shanghai, China.

FAST (2017), 'Open rov change generation', Available at: https://www.youtube.com/watch?v=G493LxywWuE. Date accessed: 22 May 2020.

Feng, Z. and Allen, R. (2004), 'Reduced order h infinity control of an autonomous underwater vehicle', *Control Engineering Practice* **12**(12), 1511–1520.

Ferreira, F., Veruggio, G., Caccia, M. and Bruzzone, G. (2012), 'Real-time optical slam-based mosaicking for unmanned underwater vehicles', *Intelligent Service Robotics* **5**(1), 55–71.

Ferrera, M., Moras, J., Trouvé-Peloux, P. and Creuze, V. (2019), 'Real-time monocular visual odometry for turbid and dynamic underwater environments', *Sensors* **19**(3), 687.

Fikes, R. E. and Nilsson, N. J. (1971), 'Strips: A new approach to the application of theorem proving to problem solving', *Artificial intelligence* **2**(3-4), 189–208.

Fjellstad, O.-E. and Fossen, T. I. (1994), 'Position and attitude tracking of auv's: a quaternion feedback approach', *IEEE Journal of Oceanic Engineering* **19**(4), 512–518.

Forster, C., Pizzoli, M. and Scaramuzza, D. (2014), Svo: Fast semi-direct monocular visual odometry, *in* '2014 IEEE international conference on robotics and automation', IEEE, Hong Kong, China.

Fossen, T. I. (2011), *Handbook of marine craft hydrodynamics and motion control*, WILEY, Hoboken, USA.

Fossen, T. I. and Sagatun, S. I. (1991), Adaptive control of nonlinear underwater robotic systems, *in* '1991 the International Conference on Robotics and Automation', IEEE, Sacramento, California, pp. 1687–1694.

Fossen, T. I. et al. (1994), *Guidance and control of ocean vehicles*, WILEY, New York, USA.

Fossen, T. and Ross, A. (2006), 'Nonlinear modelling, identification and control of uuvs', *IEE Control Engineering Series* **69**, 13.

Frank, J. and Jónsson, A. (2003), 'Constraint-based attribute and interval planning', *Constraints* **8**(4), 339–364.

Frumer, Y. (2020), 'The short, strange life of the first friendly robot: Japan's gakutensoku was a giant pneumatic automaton that toured through asia—until it mysteriously disappeared', *IEEE Spectrum* **57**(6), 42–48.

Fugitt, R. (1975), Design and operation of two remotely manned undersea vehicles, *in* '1975 OCEANS', IEEE, San Diego, USA.

Gagnon, J.-F., Durantin, G., Vachon, F., Causse, M., Tremblay, S. and Dehais, F. (2012), 'Anticipating human error before it happens: towards a psychophysiological model for online prediction of mental workload', *Open Archive Toulouse Archive Ouverte* . Available at: https://pdfs.semanticscholar.org/6f71/a5199f7bde3513554b710fc82c687f7cb301.pdf.

Garratt, M. A. and Chahl, J. S. (2008), 'Vision-based terrain following for an unmanned rotorcraft', *Journal of Field Robotics* **25**(4-5), 284–301.

Garratt, M. and Chahl, J. (2007), An optic flow damped hover controller for an autonomous helicopter, *in* '2007 the 22nd International UAV Systems Conference', IEEE, Bristol, UK.

Garrido, A. and Barber, F. (2001), 'Integrating planning and scheduling', *Applied Artificial Intelligence* **15**(5), 471–491.

Garrido-Jurado, S., Muñoz-Salinas, R., Madrid-Cuevas, F. J. and Marín-Jiménez, M. J. (2014), 'Automatic generation and detection of highly reliable fiducial markers under occlusion', *Pattern Recognition* **47**(6), 2280–2292.

Golas, P. J. (2007), *Graphics and Text in the Production of Technical Knowledge in China*, Brill, Beijing, China, chapter 3.'Like obtaining a great treasure': The illustrations in Song Yingxing's the exploitation of the works of nature, pp. 569–614.

Grasmueck, M., Eberli, G. P., Viggiano, D. A., Correa, T., Rathwell, G. and Luo, J. (2006), 'Autonomous underwater vehicle (auv) mapping reveals coral mound distribution, morphology, and oceanography in deep water of the straits of florida', *Geophysical Research Letters* **33**(23).

Gregory, N. M., Dorais, G. A., Fry, C., Levinson, R. and Plaunt, C. (2002), Idea: Planning at the core of autonomous reactive agents, *in* '2002 International NASA Workshop on Planning and Scheduling for Space', Citeseer, Moffett Field, USA.

Groves, P. D. (2015), 'Principles of gnss, inertial, and multisensor integrated navigation systems', *IEEE Aerospace and Electronic Systems Magazine* **30**(2), 26–27.

Haigh, K. Z. and Veloso, M. M. (1998), *Autonomous agents*, Springer, Heidelberg, Germany, chapter Interleaving planning and robot execution for asynchronous user requests, pp. 79–95.

Harris, C. G. and Pike, J. (1988), '3d positional integration from image sequences', *Image and Vision Computing* **6**(2), 87–90.

Hartley, R. and Zisserman, A. (2003), *Multiple view geometry in computer vision*, Cambridge University Press, Cambridge, UK.

Henry, G. W. and Eaton, J. A. (1995), Robotic telescopes: current capabilities, present developments, and future prospects for automated astronomy: proceedings of a symposium held as part of the 106th annual meeting of the astronomical society of the pacific, flagstaff, arizona, 28-30 june 1994, *in* '1995 Robotic Telescopes. Current Capabilities, Present Developments, and Future Prospects for Automated Astronomy', Astronomical Society of the Pacific Conference Series, Laramie, USA.

Hildebrandt, M. and Kirchner, F. (2010), Imu-aided stereo visual odometry for ground-tracking auv applications, *in* '2010 OCEANS', IEEE, Sydney, Australia.

Hover, F. S., Eustice, R. M., Kim, A., Englot, B., Johannsson, H., Kaess, M. and Leonard, J. J. (2012), 'Advanced perception, navigation and planning for autonomous in-water ship hull inspection', *The International Journal of Robotics Research* **31**(12), 1445–1464.

Huang, G., Kaess, M. and Leonard, J. J. (2014), Towards consistent visual-inertial navigation, *in* '2014 IEEE International Conference on Robotics and Automation', IEEE, Hong Kong, China.

IHC (2019), 'Deep-sea mining', Available at: https://www.royalihc.com/en/products/mining/dredge-mining/deep-sea-mining. Date accessed: 10 February 2020.

Innocenti, M. and Campa, G. (1999), Robust control of underwater vehicles: sliding mode vs. lmi synthesis, *in* '1999 Proceedings of the American Control Conference', Vol. 5, IEEE, San Diego, USA, pp. 3422–3426.

Irani, M. and Anandan, P. (1999), About direct methods, *in* '1999 International Workshop on Vision Algorithms', Springer, Corfu, Greece.

Jalbert, J., Shevenell, M., Chappel, S., Welsh, R. and Blidberg, R. (1988), Eave iii-untethered auv submersible, *in* '1988 OCEANS', IEEE, Baltimore, USA.

Jerosch, K., Schlüter, M. and Pesch, R. (2006), 'Spatial analysis of marine categorical information using indicator kriging applied to georeferenced video mosaics of the deep-sea håkon mosby mud volcano', *Ecological Informatics* **1**(4), 391–406.

Jónsson, A. K., Morris, P. H., Muscettola, N., Rajan, K. and Smith, B. D. (2000), Planning in interplanetary space: Theory and practice., *in* '2000 International Conference on Automated Planning and Scheduling', AAAI Press, Nancy, France.

Kaess, M., Ranganathan, A. and Dellaert, F. (2008), 'isam: Incremental smoothing and mapping', *IEEE Transactions on Robotics* **24**(6), 1365–1378.

Kaminer, I., Pascoal, A. M., Silvestre, C. J. and Khargonekar, P. P. (1991), Control of an underwater vehicle using h/sub infinity/synthesis, *in* '1991 Proceedings of the 30th IEEE Conference on Decision and Control', IEEE, Brighton, England.

Kaminski, C., Crees, T., Ferguson, J., Forrest, A., Williams, J., Hopkin, D. and Heard, G. (2010), 12 days under ice–an historic auv deployment in the canadian high arctic, *in* '2010 IEEE/OES Autonomous Underwater Vehicles', IEEE, Monterey, USA.

Kannala, J. and Brandt, S. S. (2006), 'A generic camera model and calibration method for conventional, wide-angle, and fish-eye lenses', *IEEE transactions on pattern analysis and machine intelligence* **28**(8), 1335–1340.

Ke, Q., Liu, J., Bennamoun, M., An, S., Sohel, F. and Boussaid, F. (2018), *Computer Vision for Assistive Healthcare*, Elsevier, London, UK, chapter Computer vision for human–machine interaction, pp. 127–145.

Kim, A. and Eustice, R. (2009), Pose-graph visual slam with geometric model selection for autonomous underwater ship hull inspection, *in* '2009 IEEE/RSJ International Conference on Intelligent Robots and Systems', IEEE, Louis, USA.

Kim, A. and Eustice, R. M. (2013), 'Real-time visual slam for autonomous underwater hull inspection using visual saliency', *IEEE Transactions on Robotics* **29**(3), 719–733.

KITTI (2016), 'Welcome to the kitti vision benchmark suite!', Available at: http://www.cvlibs.net/datasets/kitti/. Date accessed: 20 March 2018.

Klein, G. and Murray, D. (2007), Parallel tracking and mapping for small ar workspaces, *in* '2007 6th IEEE and ACM International Symposium on Mixed and Augmented Reality', IEEE, Washington, USA.

Krstic, M., Kanellakopoulos, I., Kokotovic, P. V. et al. (1995), *Nonlinear and adaptive control design*, WILEY, New York, USA.

Kurfess, T. R. and Nagurka, M. L. (1991), 'Understanding the root locus using gain plots', *IEEE Control Systems Magazine* **11**(5), 37–40.

LeBouthillier, E. (1999), 'W. grey walter and his turtle robots', Available at: http://web.csulb.edu/~wmartinz/rssc/content/w-grey-walter-and-his-turtle-robots.html. Date accessed: 11 February 2020.

Leutenegger, S., Lynen, S., Bosse, M., Siegwart, R. and Furgale, P. (2015), 'Keyframe-based visual–inertial odometry using nonlinear optimization', *The International Journal of Robotics Research* **34**(3), 314–334.

Li, P., Garratt, M. and Lambert, A. (2015), 'Monocular snapshot-based sensing and control of hover, takeoff, and landing for a low-cost quadrotor', *Journal of Field Robotics* **32**(7), 984–1003.

LinkQuest (1998), 'Pinpoint applications', Available at: https://www.link-quest.com/html/lbl_applications.htm. Date accessed: 22 May 2020.

Logan, C. L. (1994), A comparison between h-infinity/mu-synthesis control and sliding-mode control for robust control of a small autonomous underwater vehicle, *in* '1994 Symposium on Autonomous Underwater Vehicle Technology', IEEE, Cambridge, USA.

Lovegrove, S., Davison, A. J. and Ibanez-Guzmán, J. (2011), Accurate visual odometry from a rear parking camera, *in* '2011 IEEE Intelligent Vehicles Symposium', IEEE, Baden, Germany.

Lowe, D. G. (2004), 'Distinctive image features from scale-invariant keypoints', *International journal of computer vision* **60**(2), 91–110.

LPMS (2018), 'Lpms na2 introduction', Available at: https://lp-research.com/lpms-nav2/. Date accessed: 17 June 2019.

M, B. (1981), Ship Propulsion Losses Related to Automated Steering and Prime Mover Control, PhD thesis, Technical University of Denmark.

MacGregor, D. et al. (1983), Fibre optics improving deepwater rov pipeline inspection, *in* '1983 Offshore Europe', Society of Petroleum Engineers, Aberdeen, UK.

Mahon, I., Williams, S. B., Pizarro, O. and Johnson-Roberson, M. (2008), 'Efficient view-based slam using visual loop closures', *IEEE Transactions on Robotics* **24**(5), 1002–1014.

Maimone, M., Cheng, Y. and Matthies, L. (2007), 'Two years of visual odometry on the mars exploration rovers', *Journal of Field Robotics* **24**(3), 169–186.

Martin, S. C. and Whitcomb, L. L. (2018), 'Nonlinear model-based tracking control of underwater vehicles with three degree-of-freedom fully coupled dynamical plant models: theory and experimental evaluation', *IEEE Transactions on Control Systems Technology* **26**(2), 404–414.

Matsumoto, G. and Potts, T. (2011), 'Two ways of researching the underwater world', *Brooks/Cole Enrichment Module* **26**.

MBARI (2004), 'Autonomous underwater vehicles', Available at: https://www.mbari.org/at-sea/vehicles/autonomous-underwater-vehicles/. Date accessed: 22 May 2020.

MBARI (2018), 'Seafloor mapping auv', Available at: https://www.mbari.org/at-sea/vehicles/autonomous-underwater-vehicles/seafloor-mapping-auv/. Date accessed: 11 February 2020.

McGann, C., Py, F., Rajan, K., Thomas, H., Henthorn, R. and McEwen, R. (2007), T-rex: A model-based architecture for auv control, *in* '2007 Workshop on Planning and Plan Execution for Real-World Systems', Providence, USA.

McGann, C., Py, F., Rajan, K., Thomas, H., Henthorn, R. and McEwen, R. (2008), A deliberative architecture for auv control, *in* '2008 IEEE International Conference on Robotics and Automation', IEEE, Nice, France.

McPhail, S. D., Furlong, M. E., Pebody, M., Perrett, J., Stevenson, P., Webb, A. and White, D. (2009), Exploring beneath the pig ice shelf with the autosub3 auv, *in* '2009 OCEANS', IEEE, Biloxi, USA.

Mebius, J. E. (2007), 'Derivation of the euler-rodrigues formula for three-dimensional rotations from the general formula for four-dimensional rotations', *arXiv preprint* . Available at: https://arxiv.org/abs/math/0701759.

Mei, C. and Rives, P. (2007), Single view point omnidirectional camera calibration from planar grids, *in* '2007 IEEE International Conference on Robotics and Automation', IEEE, Roma, Italy.

Melia, T. M. (1991), *Damn the torpedoes: a short history of US naval mine countermeasures, 1777-1991*, Naval Historical Center, Washington, USA.

Menze, M. and Geiger, A. (2015), Object scene flow for autonomous vehicles, *in* '2015 Conference on Computer Vision and Pattern Recognition', IEEE, Boston, USA.

Miessner, B. F. (1916), *Radiodynamics: the wireless control of torpedoes and other mechanisms*, D. Van Nostrand Company, New York City, USA.

Minowa, A. and Toda, M. (2018), 'A high-gain observer-based approach to robust motion control of towed underwater vehicles', *IEEE Journal of Oceanic Engineering* **99**, 1–14.

Mur-Artal, R., Montiel, J. M. M. and Tardos, J. D. (2015), 'Orb-slam: a versatile and accurate monocular slam system', *IEEE transactions on robotics* **31**(5), 1147–1163.

Muscettola, N., Nayak, P. P., Pell, B. and Williams, B. C. (1998), 'Remote agent: To boldly go where no ai system has gone before', *Artificial intelligence* **103**(1-2), 5–47.

Narendra, K. S. and Mukhopadhyay, S. (1992), 'Intelligent control using neural networks', *IEEE Control Systems* **12**(2), 11–18.

Newcombe, R. A., Lovegrove, S. J. and Davison, A. J. (2011), Dtam: Dense tracking and mapping in real-time, *in* '2011 International Conference on Computer Vision', IEEE, Barcelona, Spain.

Niku, S. B. (2001), *Introduction to robotics: analysis, systems, applications*, Vol. 7, Prentice Hall, Upper Saddle River, USA.

Nistér, D., Naroditsky, O. and Bergen, J. (2006), 'Visual odometry for ground vehicle applications', *Journal of Field Robotics* **23**(1), 3–20.

Ogata, K. (2002), *Modern control engineering*, Prentice Hall, India.

Palomeras, N., Vallicrosa, G., Mallios, A., Bosch, J., Vidal, E., Hurtos, N., Carreras, M. and Ridao, P. (2018), 'Auv homing and docking for remote operations', *Ocean Engineering* **154**, 106–120.

Parchami, M. and Mariottini, G.-L. (2014), A comparative study on 3-d stereo reconstruction from endoscopic images, *in* '2014 International Conference on PErvasive Technologies Related to Assistive Environments', ACM, New York City, USA.

Paull, L., Saeedi, S., Seto, M. and Li, H. (2013), 'Auv navigation and localization: A review', *IEEE Journal of Oceanic Engineering* **39**(1), 131–149.

Paull, L., Saeedi, S., Seto, M. and Li, H. (2014), 'Auv navigation and localization: A review', *IEEE Journal of Oceanic Engineering* **39**(1), 131–149.

PDFSIM (2020), 'Savante's subsea laser technology', Available at: https://www.savante.co.uk/micro-rov-laser-inspection. Date accessed: 22 May 2020.

Pelka, M., Majek, K. and Bedkowski, J. (2019), Testing the affordable system for digitizing usar scenes, *in* '2019 IEEE International Symposium on Safety, Security, and Rescue Robotics', IEEE, Wurzburg, Germany.

Penders, J., Alboul, L., Witkowski, U., Naghsh, A., Saez-Pons, J., Herbrechtsmeier, S. and El-Habbal, M. (2011), 'A robot swarm assisting a human fire-fighter', *Advanced Robotics* **25**(1-2), 93–117.

Pike, J. (1999), 'Cable-controlled undersea recovery vehicle (curv)', Available at: https://fas.org/irp/program/collect/curv.htm. Date accessed: 3 July 2020.

PSS (2020), 'Rov operations', Available at: http://www.pss-offshore.com/rov-operations/. Date accessed: 22 May 2020.

Qin, T. and Shen, S. (2018), Online temporal calibration for monocular visual-inertial systems, *in* '2018 IEEE/RSJ International Conference on Intelligent Robots and Systems', IEEE, Madrid, Spain.

Rahman, S., Li, A. Q. and Rekleitis, I. (2018*a*), Sonar visual inertial slam of underwater structures, *in* '2018 IEEE International Conference on Robotics and Automation', IEEE, Brisbane, Australia.

Rahman, S., Li, A. Q. and Rekleitis, I. (2018*b*), 'An underwater slam system using sonar, visual, inertial, and depth sensor', *arXiv preprint* . Available at: https://arxiv.org/abs/1810.03200.

Rajan, K., Bernard, D., Dorais, G., Gamble, E., Kanefsky, B., Kurien, J., Millar, W., Muscettola, N., Nayak, P., Rouquette, N. et al. (2000), Remote agent: An autonomous control system for the new millennium, *in* '2000 European Conference on Artificial Intelligence', IOS Press, Berlin, Germany.

Rajan, K., Py, F. and Barreiro, J. (2013), *Marine Robot Autonomy*, Springer, New York City, USA, chapter Towards deliberative control in marine robotics, pp. 91–175.

Rani, P., Liu, C., Sarkar, N. and Vanman, E. (2006), 'An empirical study of machine learning techniques for affect recognition in human–robot interaction', *Pattern Analysis and Applications* **9**(1), 58–69.

Realsense, I. (2018), 'Intel realsense t265 tracking camera', Available at: https://www.intelrealsense.com/tracking-camera-t265/. Date accessed: 14 May 2019.

RigZone (2018), 'How do rovs work?', Available at: https://www.rigzone.com/training/insight.asp?insight_id=343&c_id=. Date accessed: 22 May 2020.

Rosten, E. and Drummond, T. (2006), Machine learning for high-speed corner detection, *in* '2006 European Conference on Computer Vision', Springer, Graz, Austria.

Rublee, E., Rabaud, V., Konolige, K. and Bradski, G. (2011), Orb: An efficient alternative to sift or surf, *in* '2011 International conference on computer vision', IEEE, Barcelona, Spain.

Sabet, M. T., Daniali, H. M., Fathi, A. and Alizadeh, E. (2017), 'A low-cost dead reckoning navigation system for an auv using a robust ahrs: Design and experimental analysis', *IEEE Journal of Oceanic Engineering* **43**(4), 927–939.

Santoso, F., Garratt, M. A., Pickering, M. R. and Asikuzzaman, M. (2016), '3d mapping for visualization of rigid structures: A review and comparative study', *IEEE Sensors Journal* **16**(6), 1484–1507.

Scaramuzza, D. and Fraundorfer, F. (2011), 'Visual odometry [tutorial]', *IEEE robotics & automation magazine* **18**(4), 80–92.

Scaramuzza, D. and Ikeuchi, K. (2014), *Omnidirectional camera*, Springer, Heidelberg, Germany.

Scaramuzza, D. and Siegwart, R. (2008), 'Appearance-guided monocular omnidirectional visual odometry for outdoor ground vehicles', *IEEE transactions on robotics* **24**(5), 1015–1026.

Schwalbe, E. (2005), Geometric modelling and calibration of fisheye lens camera systems, *in* '2005 Panoramic Photogrammetry Workshop, Int. Archives of Photogrammetry and Remote Sensing', Berlin, Germany.

SeeByte (1998), 'A rov system', Available at: http://www.seebyte.com/. Date accessed: 1 May 2020.

Sharkey, N. and Sharkey, A. (2009), 'Electro-mechanical robots before the computer', *Proceedings of the Institution of Mechanical Engineers, Part C: Journal of Mechanical Engineering Science* **223**(1), 235–241.

Siciliano, B. and Khatib, O. (2016), *Springer handbook of robotics*, Springer, Heidelberg, Germany.

Squyres, S. W., Arvidson, R. E., Baumgartner, E. T., Bell, J. F., Christensen, P. R., Gorevan, S., Herkenhoff, K. E., Klingelhöfer, G., Madsen, M. B., Morris, R. V. et al. (2003), 'Athena mars rover science investigation', *Journal of Geophysical Research: Planets* **108**(E12).

Sun, D., Roth, S. and Black, M. J. (2010), Secrets of optical flow estimation and their principles, *in* '2010 IEEE Computer Society Conference on Computer Vision and Pattern Recognition', IEEE, San Francisco, USA.

Sünderhauf, N., Brock, O., Scheirer, W., Hadsell, R., Fox, D., Leitner, J., Upcroft, B., Abbeel, P., Burgard, W., Milford, M. et al. (2018), 'The limits and potentials of deep learning for robotics', *The International Journal of Robotics Research* **37**(4-5), 405–420.

Szabolcsi, R. (2014), 'The birth of the term robot', *Advances in Military Technology, ISSN1802-2308* **9**(1), 117–128.

TECH, S. (2018), 'Mini-rov observer', Available at: https://www.subsea-tech.com/mini-rov-observer/. Date accessed: 3 February 2020.

Thrun, S., Burgard, W. and Fox, D. (2005), *Probabilistic robotics*, MIT, Cambridge, USA.

Utkin, V., Guldner, J. and Shi, J. (2009), *Sliding mode control in electro-mechanical systems*, CRC Press, Boca Raton, USA.

Vickery, K. (1998), Acoustic positioning systems. a practical overview of current systems, *in* '1998 Workshop on Autonomous Underwater Vehicles', IEEE, Cambridge, USA.

Voskuhl, A. (2007), 'Producing objects, producing texts: accounts of android automata in late eighteenth-century europe', *Studies in History and Philosophy of Science Part A* **38**(2), 422–444.

Wallén, J. (2008), *The history of the industrial robot*, Linköping University Electronic Press, Linkoping, Sweden.

Wang, J.-S. and Lee, C. G. (2003), 'Self-adaptive recurrent neuro-fuzzy control of an autonomous underwater vehicle', *IEEE transactions on robotics and automation* **19**(2), 283–295.

Wang, W. and Clark, C. M. (2006), Modeling and simulation of the videoray pro iii underwater vehicle, *in* '2006 OCEANS', IEEE, Singapore.

Wang, W. and Clark, C. M. (2007), Modeling and simulation of the videoray pro 3 underwater vehicle, *in* '2006 OCEANS', IEEE, Singapore.

WaterLink (2020), 'DVL A50', https://waterlinked.com/product/dvl-a50/. Date accessed: 4 Dec 2020.

Weidner, N., Rahman, S., Li, A. Q. and Rekleitis, I. (2017), Underwater cave mapping using stereo vision, *in* '2017 IEEE International Conference on Robotics and Automation', IEEE, Singapore.

Westwood, D. (2016), 'The world rov market forecast 2011–2015', Available at: https://www.yumpu.com/en/document/view/33110405/ the-world-rov-market-forecast-2011-2015-douglas-westwood. Date accessed: 7 September 2020.

Wilcox, B. H. and Gennery, D. B. (1990), *1990 Autonomous robot vehicles*, Springer, New York City, USA, chapter A Mars Rover for the 1990's, pp. 444–449.

Williams, K. W. (2004), A summary of unmanned aircraft accident/incident data: Human factors implications, Technical report, The FAA Civil Aerospace Medical Institute, Oklahoma City, USA.

Williams, S. B., Pizarro, O., Jakuba, M. and Barrett, N. (2010), Auv benthic habitat mapping in south eastern tasmania, *in* 'Field and Service Robotics', Springer, pp. 275–284.

Wirth, S., Carrasco, P. L. N. and Codina, G. O. (2013), Visual odometry for autonomous underwater vehicles, *in* '2013 OCEANS', IEEE, Bergen, Norway.

Woodman, O. J. (2007), An introduction to inertial navigation, Technical report, Computer Laboratory, University of Cambridge, UK.

Wooldridge, M. (2001), Intelligent agents: The key concepts, *in* '2001 ECCAI Advanced Course on Artificial Intelligence', Springer, Prague, Czech.

Wynn, R. B., Huvenne, V. A., Le Bas, T. P., Murton, B. J., Connelly, D. P., Bett, B. J., Ruhl, H. A., Morris, K. J., Peakall, J., Parsons, D. R. et al. (2014), 'Autonomous underwater vehicles (auvs): Their past, present and future contributions to the advancement of marine geoscience', *Marine Geology* **352**, 451–468.

Wynn, R., Bett, B., Evans, A., Griffiths, G., Huvenne, V., Jones, A., Palmer, M., Dove, D., Howe, J., Boyd, T. et al. (2013), *Investigating the feasibility of utilizing AUV and Glider technology for mapping and monitoring of the UK MPA network. Final report for Defra project MB0118*, National Oceanography Centre, Liverpool, UK.

Yan, H.-S. and Lin, T.-Y. (2002), A systematic approach to the reconstruction of ancient chinese escapement regulators, *in* '2002 International Design Engineering Technical Conferences and Computers and Information in Engineering Conference', ASME, Montreal, Canada.

Yan, J. (2009), 'Robotic floor cleaner'. US Patent 7,568,259.

Yan, Z., Peng, S., Zhou, J., Xu, J. and Jia, H. (2010), Research on an improved dead reckoning for auv navigation, *in* '2010 Chinese Control and Decision Conference', IEEE, pp. 1793–1797.

Yoerger, D. R., Cooke, J. G. and Slotine, J.-J. (1990), 'The influence of thruster dynamics on underwater vehicle behavior and their incorporation into control system design', *IEEE Journal of Oceanic Engineering* **15**(3), 167–178.

Yu, C., Xiang, X., Lapierre, L. and Zhang, Q. (2017), 'Nonlinear guidance and fuzzy control for three-dimensional path following of an underactuated autonomous underwater vehicle', *Ocean Engineering* **146**, 457–467.

Yuh, J., Ura, T. and Bekey, G. A. (2012), *Underwater robots*, Springer Science & Business Media, New York City, USA.

Zhang, G.-c., Huang, H., Wan, L., Li, Y.-m., Cao, J., Su, Y.-m. et al. (2018), 'A novel adaptive second order sliding mode path following control for a portable auv', *Ocean Engineering* **151**, 82–92.

Zhang, J. (2016), Further improvements of fuzzy track keeping controller for ships, Master's thesis, Dalian Maritime University.

Zhang, J., Ila, V. and Kneip, L. (2018), Robust visual odometry in underwater environment, *in* '2018 OCEANS-MTS/IEEE Kobe Techno-Oceans', IEEE, Kobe, Japan.

Zhang, J. and Singh, S. (2015), Visual-lidar odometry and mapping: Low-drift, robust, and fast, *in* '2015 IEEE International Conference on Robotics and Automation', IEEE, Seattle, USA.

Zhang, Y., McEwen, R. S., Ryan, J. P. and Bellingham, J. G. (2010), 'Design and tests of an adaptive triggering method for capturing peak samples in a thin phytoplankton layer by an autonomous underwater vehicle', *IEEE Journal of Oceanic Engineering* **35**(4), 785–796.

Zhilenkov, A. (2016), 'The study of the process of the development of marine robotics', *Vibroengineering Procedia* **8**, 17–21.

Zhou, H.-Y., Liu, K.-Z. and Feng, X.-S. (2011), 'State feedback sliding mode control without chattering by constructing hurwitz matrix for auv movement', *International Journal of Automation and Computing* **8**(2), 262–268.

Zimmermann, H.-J. (1996), *Fuzzy Set Theory—and Its Applications*, Springer, New York, USA, chapter Fuzzy control, pp. 203–240.

Appendix A. Parameter and Hydrodynamic Coefficients Of VideoRay Pro 3

A.1. The properties of VideorayPro 3

Parameter	Value	Units	Description
L	0.36	m	vehicle length
W	0.35	m	vehicle width
H	0.23	m	vehicle height
I_{xx}	0.02275	kgm^2	moment of inertia
I_{yy}	0.02391	kgm^2	moment of inertia
I_{zz}	0.02532	kgm^2	moment of inertia

Table A.1 Geometry and mass property

	Analytical	Experimental
$X_{\dot{u}}$	1.9404	NA
$Y_{\dot{v}}$	6.0572	NA
$Z_{\dot{w}}$	3.9482	NA
$K_{\dot{p}}$	0.0326	NA
$M_{\dot{q}}$	0.0175	NA
$N_{\dot{r}}$	0.0321	0.0118
$X_{u\|u\|}$	8.2845	6.0418
$Y_{v\|v\|}$	23.689	30.731
$Z_{w\|w\|}$	20.523	26.357
$K_{p\|p\|}$	0.0048	NA
$M_{q\|q\|}$	0.0069	NA
$N_{r\|r\|}$	0.0089	0.4504
X_u	8.2845	6.0418
Y_v	23.689	30.731
Z_w	20.523	26.357
K_p	0.0048	NA
M_q	0.0069	NA
N_r	0.0089	0.4504

Table A.2 Linear drag coefficients