A Hybrid Control Architecture Development for the
Guidance, Navigation and Control of the Tethra Prototype
Submersible Vehicle

Thesis submitted to the University of Limerick for the Degree of
Doctor of Philosophy

by
Levente Molnar, B.Eng.

Supervised by: Dr. Daniel Toal and Dr. Edin Omerdic

Mobile & Marine Robotics Research Group
Department of Electronics and Computer Engineering
University of Limerick, Limerick

July 2006
Trademarks

Windows 2000™ and Windows XP™ are trademarks of Microsoft Corporation.

MicroC/OS-II is a trademark of Micrium.

MATLAB™, Simulink™, Fuzzy Logic Toolbox™ and Virtual Reality Toolbox™ are trademarks of The MathWorks Inc.

LabVIEW™ is a trademark of National Instruments.
Decloration / Statements

DECLARATION

I hereby certify that this material is entirely my own work and has not been taken from the work of others save and to the extent that such work has been cited and acknowledged within the text of my work.

Signed ................................... (candidate)

Date ........................................
To my parents
Acknowledgements

I wish to express my deep and sincere gratitude to my supervisors, Dr. Daniel Toal and Dr. Edin Omerdic, for giving me the opportunity to work on this project, for their guidance, support and encouragement throughout the research and writing phases of this thesis.

I have made many friends along the way. They have helped me, one way or another, in my struggle to complete the research: Especially, I would like to thank to my colleagues Sean, James, Pepijn, Jamie and Trevor for their advices, camaraderie and support. Working with you as part of the team, was an honour and always a challenge!

I am also grateful for the administrative and technical support at the department that made possible to carry out the work I am presenting.

I would also like to thank my family, my parents and sister for the support they provided me through my entire life and in particular, I must acknowledge my wife, Camelia, without whose care, moral support and understanding I would not have finished this. I would like to thank for all her patience with me.
Abstract

The Mobile & Marine Robotics Research Centre (MMRRC) at the University of Limerick is developing a highly manoeuvrable AUV platform to address the challenges of high-resolution seabed survey in both shallow and deep water. The work of the author in the development of a novel guidance, navigation and control (GNC) system for the Tethra AUV is described in the thesis. A full and comprehensive GNC system, with open architecture, has been designed, implemented and tested with many novel features.

The main components of the control system include: Control Allocation, Virtual Joystick, Low-level controllers, Fuzzy controller for Obstacle Avoidance and Behaviour Coordinator/Arbitration.

As result of the research and development of the author the Tethra AUV has been endowed with a comprehensively integrated Guidance Navigation and Control system, enabling the execution of full survey scale operations in challenging near seabed scenarios.
Nomenclature

Symbols

ROV model:

\[
\mathbf{\tau} = \begin{bmatrix}
X \\
Y \\
Z \\
K \\
M \\
N
\end{bmatrix}
\]

- Generalised vector of total forces and moments exerted by thrusters

\{B\} - Body-fixed frame

\{E\} - Earth-fixed frame

\textit{O} - Origin of body-fixed frame

\textit{CG} - Centre of gravity

\[
\mathbf{E} \mathbf{\eta} = \begin{bmatrix}
E \mathbf{\eta}_1 \\
E \mathbf{\eta}_2
\end{bmatrix}
\]

- Position and orientation vector

\[
E \mathbf{\eta}_1 = \begin{bmatrix}
X \\
y \\
z
\end{bmatrix}
\]

- Position vector

\[
E \mathbf{\eta}_2 = \begin{bmatrix}
\phi \\
\theta \\
\psi
\end{bmatrix}
\]

- Orientation vector

\[
\mathbf{B} \mathbf{v} = \begin{bmatrix}
B \mathbf{v}_1 \\
B \mathbf{v}_2
\end{bmatrix}
\]

- Linear and angular velocity vector

\[
B \mathbf{v}_1 = \begin{bmatrix}
u \\
v \\
w
\end{bmatrix}
\]

- Linear velocity vector

\[
B \mathbf{v}_2 = \begin{bmatrix}
p \\
q \\
r
\end{bmatrix}
\]

- Angular velocity vector
\( \mathbf{J}_1^{(E \mathbf{n}_1)} \) – Transformation matrix \( ^B \mathbf{v}_1 \rightarrow ^E \mathbf{n}_1 \)

\( \mathbf{J}_2^{(E \mathbf{n}_2)} \) – Transformation matrix \( ^B \mathbf{v}_2 \rightarrow ^E \mathbf{n}_2 \)

\( \mathbf{M}_{RB} \) – Rigid-body inertia matrix

\( \mathbf{C}_{RB}^{(B \mathbf{v})} \) – Rigid-body Coriolis and centripetal matrix

\( \mathbf{M}_A \) – Added-mass matrix

\( \mathbf{M} \) – Inertia matrix (including added mass)

\( \mathbf{C}_A^{(B \mathbf{v})} \) – Added-mass Coriolis and centripetal matrix

\( \mathbf{C}^{(B \mathbf{v})} \) – Coriolis and centripetal matrix (including added mass)

\( \mathbf{D}^{(B \mathbf{v})} \) – Total hydrodynamic damping matrix

\( ^B \mathbf{g}^{(E \mathbf{n})} \) – Vector of restoring (gravitational and buoyant) forces and moments

\( ^B \mathbf{\tau}_E \) – Vector of environmental forces and moments

\( ^B \mathbf{\tau} \) – Vector of propulsion forces and moments (exerted by the thrusters)

\( \mathbf{g} \) – Acceleration due to gravity

\( p \) – Number of thrusters in the general case

\( 'Th \) – General thruster

\( 'T \) – Propeller thrust (force)

\( 'Q_e \) – Propeller torque (moment), generated by rotation

\( 'Q_r \) – Propeller torque (moment), generated by \( 'T \)

\( 'Q \) – Total torque (moment), exerted by thruster

\( 'HT \) – \( i^{th} \) horizontal thruster

\( 'VT \) – \( i^{th} \) vertical thruster

\( 'r \) – Position vector of the thruster \( 'Th \) relative to \( O \)

\( 'e \) – Orientation vector of the thruster \( 'Th \)

\( 'c \) – Spin direction coefficient

Control allocation for underwater vehicles:

\( p \) – Number of thrusters

\( \mathbf{T} \) – Thruster configuration matrix

\( \mathbf{K} \) – Force coefficient matrix

\( \mathbf{u} \) – Control vector
\( \mathbf{f} \) – Vector of control forces

\( \mathbf{B} \) – Thruster control matrix

\( \mathbf{\tau} \) – Total vector of propulsion forces and moments

\( \mathbf{\tau}_x, \mathbf{\tau}_y, \mathbf{\tau}_z \) – Surge, sway and yaw force

\( \mathbf{\tau}_K, \mathbf{\tau}_M, \mathbf{\tau}_N \) – Roll, pitch and yaw moment

\( \mathbf{T}_m \) – Maximum thruster force

\( \mathbf{u}_m \) – Maximum control variable

\( \mathbf{\tau}_{xm} \) – Maximum surge force

\( \mathbf{\tau}_{zm} \) – Maximum heave force

\( \mathbf{\tau}_{km} \) – Maximum roll moment

\( \mathbf{\tau}_{nm} \) – Maximum yaw moment

\( \mathbf{u}_{HT} \) – Normalised true control input (horizontal plane)

\( \mathbf{u}_{VT} \) – Normalised true control input (vertical plane)

\( \mathbf{\tau}_{HT} \) – Normalised virtual control input (horizontal plane)

\( \mathbf{\tau}_{VT} \) – Normalised virtual control input (vertical plane)

\( \mathbf{B}_{HT} \) – Control effectiveness matrix (horizontal plane)

\( \mathbf{B}_{VT} \) – Control effectiveness matrix (vertical plane)

\( \mathbf{N}_X, \mathbf{N}_Y, \mathbf{N}_K, \mathbf{N}_N \) – Normal vector to the planes \( \mathbf{\pi}_X, \mathbf{\pi}_Y, \mathbf{\pi}_K \) and \( \mathbf{\pi}_N \), respectively

\( \mathbf{\Omega}_{HT} \) – Normalised constrained control subset (horizontal plane)

\( \mathbf{\Omega}_{VT} \) – Normalised constrained control subset (vertical plane)

\( \mathbf{\Phi}_{HT} \) – Attainable command set (horizontal plane)

\( \mathbf{\Phi}_{VT} \) – Attainable command set (vertical plane)

\( \mathbf{\Phi}_{HT} \) – Virtual control space (horizontal plane)

\( \mathbf{\Phi}_{VT} \) – Virtual control space (vertical plane)
<table>
<thead>
<tr>
<th>Acronyms &amp; abbreviations</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>AI</td>
<td>Artificial Intelligence</td>
</tr>
<tr>
<td>AMS</td>
<td>Attainable Moment Set</td>
</tr>
<tr>
<td>AUV</td>
<td>Autonomous Underwater Vehicle</td>
</tr>
<tr>
<td>CG</td>
<td>Centre of Gravity</td>
</tr>
<tr>
<td>DVL</td>
<td>Doppler Velocity Log</td>
</tr>
<tr>
<td>DOF</td>
<td>Degree of Freedom</td>
</tr>
<tr>
<td>FCOA</td>
<td>Fuzzy Controller for Obstacle Avoidance</td>
</tr>
<tr>
<td>GNC</td>
<td>Guidance Navigation and Control</td>
</tr>
<tr>
<td>GI</td>
<td>Generalised Inverse</td>
</tr>
<tr>
<td>HCU</td>
<td>Hand Control Unit</td>
</tr>
<tr>
<td>HEA PRTLI</td>
<td>Higher Education Authority Programme for Research in Third Level Institutions</td>
</tr>
<tr>
<td>HT</td>
<td>Horizontal Thruster</td>
</tr>
<tr>
<td>IMU</td>
<td>Inertial Measurement Unit</td>
</tr>
<tr>
<td>INS</td>
<td>Inertial Navigation System</td>
</tr>
<tr>
<td>PF</td>
<td>Path Following</td>
</tr>
<tr>
<td>PNG</td>
<td>Proportional Navigation Guidance</td>
</tr>
<tr>
<td>RC</td>
<td>Radio Control</td>
</tr>
<tr>
<td>ROV</td>
<td>Remotely Operated Vehicle</td>
</tr>
<tr>
<td>SI</td>
<td>International System of Units</td>
</tr>
<tr>
<td>TCM</td>
<td>Thruster Control Matrix</td>
</tr>
<tr>
<td>TT</td>
<td>Trajectory Tracking</td>
</tr>
<tr>
<td>UUV</td>
<td>Unmanned Underwater Vehicle</td>
</tr>
<tr>
<td>VRML</td>
<td>Virtual Reality Modeling Language</td>
</tr>
<tr>
<td>VT</td>
<td>Vertical Thruster</td>
</tr>
<tr>
<td>WP</td>
<td>Way-Point</td>
</tr>
</tbody>
</table>
# Table of Contents

DECLARATION / STATEMENTS ............................................................................................................. II

ACKNOWLEDGEMENTS ...................................................................................................................... IV

ABSTRACT ........................................................................................................................................ V

NOMENCLATURE .......................................................................................................................... VI

   Symbols......................................................................................................................................... vi

   Acronyms & abbreviations .......................................................................................................... ix

TABLE OF CONTENTS .................................................................................................................. X

TABLE OF FIGURES ..................................................................................................................... XVII

LIST OF TABLES ............................................................................................................................. XXII

CHAPTER 1: INTRODUCTION ......................................................................................................... 1

   1.1 BACKGROUND .................................................................................................................. 1

   1.2 MOTIVATION ................................................................................................................... 2

   1.3 AIMS AND OBJECTIVES ................................................................................................. 5

   1.4 OVERVIEW OF CHAPTER CONTENTS ............................................................................ 5

   1.5 LIST OF MAIN CONTRIBUTIONS .................................................................................... 7

CHAPTER 2: LITERATURE REVIEW ............................................................................................... 8

   2.1 INTRODUCTION ............................................................................................................... 8

   2.2 PROPULSION SYSTEM MODELING & CONTROL ALLOCATION .................................. 8

      2.2.1 One-state model ........................................................................................................ 8

      2.2.2 Two-state model ....................................................................................................... 9

      2.2.3 Three-state model .................................................................................................... 10

      2.2.4 Propeller thrust and torque modeling ...................................................................... 11

      2.2.5 Quasi-steady thrust and torque .............................................................................. 11

         Thruster modeling (Caccia, Indiveri et al., 2000) .......................................................... 12
2.2.6 Propeller thrust and torque modeling ................................................................. 13

2.2.7 Non-symmetrical propeller T-curves ................................................................. 15

2.2.8 Vectorisation ........................................................................................................ 15

Bilinear thrusters model in vector form ..................................................................... 16

Affine thrusters model in vector form ......................................................................... 17

2.2.9 Dynamic model of permanent magnet DC-motor ........................................... 17

2.2.10 Control allocation methods ............................................................................... 21

Direct control allocation method ............................................................................... 22

Generalised inverse .................................................................................................... 22

Daisy chain method .................................................................................................. 23

Optimisation based methods ....................................................................................... 23

Control allocation method of over-actuated unmanned underwater vehicles ........... 24

2.3 CONTROL ARCHITECTURES .............................................................................. 25

2.3.1 Control architectures for autonomous mobile robots ...................................... 25

Reactive control ........................................................................................................ 28

Behaviour-based control .......................................................................................... 29

Subsumption architecture (Brooks, 1986) ................................................................. 32

Hybrid deliberative/reactive architectures .................................................................. 36

2.3.2 Control architectures for AUVs ......................................................................... 39

Surveys of control architectures for AUVs ................................................................. 39

State-configured layered control (Bellingham, Beaton et al., 1989) ......................... 40

An experimental comparison of hierarchical and subsumption architectures (Byrnes, MacPherson et al., 1992) ................................................................. 42

A tri-level hybrid software architecture (Healey, Pascoal et al., 1995) ....................... 42

Classification of AUV control architectures (Valavanis, Gracanin et al., 1997) .......... 43

Classification of AUV control architectures (Ridao, Batlle et al., 1999) ................. 44

A comparative evaluation of four behaviour-based control architectures (Carreras, Batlle et al., 2000) ................................................................. 45

Specifications for robot control architectures (Coste-Maniere and Simmons, 2000) . 45

The O^{2}CA^{2} hybrid control architecture (Ridao, Battle et al., 2002) ...................... 46
Behaviour-based hybrid coordination method & Reinforcement learning (Carreras, 2003)...........47
Enhanced subsumption architecture: Command fusion & Prioritisation of the safety-critical
(emergency fail-safe) behaviours (Toal, 2004) ........................................................................48
2.3.3 Concluding summary remarks on survey of control architectures for AUVs...........49
2.4 WAY-POINT GUIDANCE SYSTEMS ..................................................................................51
2.4.1 Conceptual elements ......................................................................................................51
Trajectory tracking and path following control .......................................................................52
Way-point representation .........................................................................................................53
Line-of-sight guidance ............................................................................................................53
2.4.2 Review of way-point guidance methods for aircraft, mobile robots & marine surface
vessels 55
Way-point guidance laws for aircraft ....................................................................................55
Way-point guidance laws & path planning for land-based mobile robots ...............................56
Way-point guidance laws for marine surface vessels ..............................................................56
2.4.3 Way-point guidance laws for UUVs ......................................................................59
Smooth way-point transition (Bakaric, Vukic et al., 2004) ......................................................63
2.5 FUZZY LOGIC CONTROL FOR AUTONOMOUS NAVIGATION ..................................66
2.6 CONCLUDING REMARKS .................................................................................................66

CHAPTER 3: TETHRA AUV DESIGN CONCEPTS ..................................................................69
3.1 INTRODUCTION ..............................................................................................................69
3.2 MECHANICAL SYSTEM DESIGN ....................................................................................72
3.3 VEHICLE POWER SUBSYSTEMS ................................................................................73
3.4 ACTUATORS ....................................................................................................................74
3.5 ACTIVE BALLAST SYSTEM ............................................................................................75
3.6 SENSORS .........................................................................................................................77
3.6.1 Initial design ...............................................................................................................77
Depth & Ballast transducer ......................................................................................................77
Vector 2XG compass .............................................................................................................78
CHAPTER 4: GUIDANCE, NAVIGATION AND CONTROL FOR TETHRA AUV

4.1 INTRODUCTION........................................................................................................... 92
4.2 CONCEPTUAL ELEMENTS.............................................................................................. 92
4.3 GUIDANCE ...................................................................................................................... 93
4.4 NAVIGATION ................................................................................................................. 93
4.5 CONTROL ....................................................................................................................... 94
4.5.1 Introduction ............................................................................................................ 94
4.5.2 Control allocation .................................................................................................. 96
   Assumptions.................................................................................................................. 96
   Thruster configuration................................................................................................... 96
Problem formulation ....................................................................................................................... 99
Normalisation................................................................................................................................ 103
Problem formulation in normalised form...................................................................................... 106
Constrained control subset $\Omega$ ............................................................................................... 108
Attainable command set $\Phi$ ...................................................................................................... 109
Feasible solution of the control allocation problem ........................................................................ 111
Visualisation .................................................................................................................................. 112
Approximation error ...................................................................................................................... 117

4.5.3 Low-level controllers .......................................................................................................... 119
Auto-Heave controller ................................................................................................................... 120
Auto-Heading controller ............................................................................................................... 122
Surge-Brake controller .................................................................................................................. 123

4.5.4 Virtual joystick ................................................................................................................... 123

4.5.5 Fuzzy Controller for Obstacle Avoidance .......................................................................... 123

4.5.6 Behaviour Co-ordinator/Arbitration .................................................................................. 128

4.6 CONCLUDING REMARKS .................................................................................................... 130

CHAPTER 5: WAY-POINT GUIDANCE SYSTEM............................................................................. 133

5.1 INTRODUCTION ..................................................................................................................... 133
5.2 WAY-POINTS ....................................................................................................................... 133
5.3 MISSION PLANNER .............................................................................................................. 134
5.4 TRACKING WAY-POINTS PLANNER .................................................................................. 135

5.5 “ROTATE TOWARDS TARGET” STRATEGIES ................................................................. 137

5.5.1 Strategy 1: “Rotate towards Target with Surge-Brake” .................................................... 137

5.5.2 Strategy 2: “Rotate towards Target without Surge-Brake” ............................................. 137

5.6 “MOVE TO TARGET” STRATEGIES .................................................................................. 138

5.6.1 Strategy 1: “Separate Auto-Heading & Cross-Track Compensation” ............................ 138

5.6.2 Strategy 2: “Auto-Heading Integrated Cross-Track Compensation” .............................. 140

5.6.3 Strategy 3: “Auto-Heading with Look-Ahead” for smooth way-point transition... 140
5.7 “MISSED WAY-POINT HANDLING” STRATEGIES ................................................................. 141
5.7.1 Strategy 1: “Range Exceeded – Missed Way-point” .................................................. 142
5.7.2 Strategy 2: “Target Reacquisition after Obstacle – Acceptance Cone” .................... 142
5.8 “TRANSITION TO NEW TRANSEC” STRATEGIES ....................................................... 143
5.8.1 Strategy 1: “Transition to new Transect with Look-Ahead” ..................................... 144
5.8.2 Strategy 2: “Transition to New Transect using the Aiding Way-Points” (Bezier curve profiling) ........................................................................................................................................ 145
5.9 VERTICAL BEHAVIOURS ................................................................................................. 147
5.9.1 Strategy 1: “Planar way-point guidance” .................................................................... 147
5.9.2 Three-dimensional way-point guidance ..................................................................... 148
      Strategy 1: “Staircase profile” ....................................................................................... 148
      Strategy 2: “3D way-point guidance by the shortest path” ........................................... 149
5.10 CONCLUDING REMARKS ............................................................................................. 152

CHAPTER 6: TESTING & EVALUATION ................................................................................. 154

6.1 INTRODUCTION .............................................................................................................. 154
6.2 REAL-WORLD TESTS (INITIAL DESIGN) .................................................................... 154
6.3 REAL-WORLD TESTS (NEW DESIGN) ........................................................................ 158
6.4 SIMULATION RESULTS .................................................................................................. 159
      6.4.1 Group A: Planar way-point guidance test cases using the “Separate Auto-Heading & Cross-Track Compensation” strategy ................................................................. 161
      6.4.2 Group B: Planar way-point guidance test cases using the “Auto-Heading Integrated Cross-Track Compensation” strategy ................................................................. 163
      6.4.3 Group C: Planar way-point guidance test cases using the “Look-Ahead” approach for smooth way-point transition .................................................................................. 165
      6.4.4 3D way-point guidance test cases ........................................................................ 166
      6.4.5 Way-point guidance test cases using the aiding way-points .................................. 167
      6.4.6 Current diagrams .................................................................................................. 168
6.5 COMPARATIVE ANALYSIS ........................................................................................................ 190

Testing and evaluation of different fuzzy controllers (FCOA safety margin) ......................... 192

6.6 CONCLUDING REMARKS .................................................................................................... 194

CHAPTER 7: CONCLUSIONS & FUTURE WORK ...................................................................... 196

7.1 INTRODUCTION ..................................................................................................................... 196

7.2 REVIEW OF THE THESIS .................................................................................................... 196

7.3 REALISATION OF AIMS & OBJECTIVES ........................................................................... 198

7.3.1 Realisation of aims ........................................................................................................... 198

7.3.2 Accomplishments of objectives ....................................................................................... 199

7.4 MAIN CONTRIBUTIONS ...................................................................................................... 200

7.5 CONCLUSION & FUTURE WORK ....................................................................................... 203

References .................................................................................................................................. 205

Appendix A: Real-time Tethra simulator .................................................................................. 214

Appendix B: Technical specifications of the Tethra AUV ............................................................. 225

Appendix C: Technical specifications of PHINS INS, RDI DVL and Microbath depth/altitude

sensor .......................................................................................................................................... 226

Appendix D: Kalman filter ........................................................................................................... 232

Appendix E: Publications ............................................................................................................ 234
Table of Figures

**Figure 2.1** Propeller thrust (force) $T$ as a function of propeller revolution $n$ and ambient water velocity $u_a$ (bilinear thruster model)...............................................................................................14

**Figure 2.2** Thrust and torque as vector variables............................................................................................................................16

**Figure 2.3** Circuit diagram for an armature controlled DC-motor. The rotor windings have an effective inductance $L$ and resistance $R$. The applied voltage $V$ is the control input. .18

**Figure 2.4** Propeller friction and load as function of shaft angular velocity. .........................................................20

**Figure 2.5** Block diagram describing DC-motor dynamics.........................................................................................21

**Figure 2.6** Robot control architecture............................................................................................................................26

**Figure 2.7** Deliberative versus reactive reasoning................................................................................................................26

**Figure 2.8** Hierarchical control flow. ...............................................................................................................................27

**Figure 2.9** Behaviour-based architecture.............................................................................................................................32

**Figure 2.10** The subsumption approach...............................................................................................................................35

**Figure 2.11** Typical deliberative/hierarchical planning strategies.........................................................................................38

**Figure 2.12** Hybrid control architecture..............................................................................................................................38

**Figure 2.13** Hierarchical Hybrid Coordination Node (Carreras, 2003)........................................................................................47

**Figure 2.14** Definition of modified LOS vector..................................................................................................................54

**Figure 2.15** Illustration of the smooth way-point transition..............................................................................................63

**Figure 2.16** Definition of the auxiliary point $A_k$ and normalised distance difference $\varepsilon_k$ .........64

**Figure 2.17** Magnitude factors for the desired heading correction: the distance factor (left); the turn factor (right).................................................................................................................................65

**Figure 3.1** Virtual reality model of Tethra (left); Real Tethra on the edge of the test pool (right). ..........................................................................................................................................................71

**Figure 3.2** Mechanical design of Tethra.................................................................................................................................72

**Figure 3.3** Vehicle power subsystems.................................................................................................................................74

**Figure 3.4** Active ballast system................................................................................................................................................76

**Figure 3.5** (a) Depth & Ballast Transducers; (b) Vector 2XG Compass; (c) MT9 Inertial Measurement Unit; (d) PA200-20 and PA500-6 Tritech precision altimeters.............77

**Figure 3.6** Toolskid navigational sensors: (a) IXSEA Phins, (b) RDI DVL, (c) CDL Microbath.80

**Figure 3.7** Functional block diagram of Phins...........................................................................................................................81

**Figure 3.8** Block diagram indicating distributed electronic design of Tethra .............................................................83

**Figure 3.9** MT9 IMU & DKF COM integration........................................................................................................................87

**Figure 3.10** Overall system architecture.................................................................................................................................88
Figure 4.1 Guidance, Navigation and Control for thrusters-propelled AUVs.................................. 93
Figure 4.2 General control system architecture with control allocation............................................ 95
Figure 4.3 Thruster configuration of Tethra.................................................................................. 97
Figure 4.4 Relationship between propeller spin direction and direction of the thrust and the torque vector ($c_{HT} = +1$ left, $c_{HT} = -1$ right). .................................................................................. 99
Figure 4.5 Relationship between thruster configuration and controllable DOFs. ....................... 103
Figure 4.6 Four cases for finding the maximum modules of force and moment vectors: (a) maximum surge force, (b) maximum yaw moment, (c) maximum heave force, (d) maximum roll moment.................................................................................................................. 104
Figure 4.7 Different stages of the control allocation................................................................. 112
Figure 4.8 Mapping between virtual and true control spaces (horizontal plane)............... 114
Figure 4.9 Mapping between virtual and true control spaces (vertical plane)............................. 116
Figure 4.10 Approximation error.............................................................................................. 117
Figure 4.11 Comparison of T- and S-approximations................................................................ 118
Figure 4.12 Internal structure of the Low-Level Controllers component................................. 119
Figure 4.13 Control scheme for depth set-point/trajectory tracking........................................ 120
Figure 4.14 Determination of the input to the feedforward compensation block....................... 121
Figure 4.15 Control scheme for heading set-point tracking.................................................... 122
Figure 4.16 Internal structure of the Surge-Brake controller.................................................... 123
Figure 4.17 Configuration of altimeters for obstacle detection.................................................. 124
Figure 4.18 Membership functions for input variables $sA$, $sB$, $sC$, $sD$, $sE$ & $sF$...................... 125
Figure 4.19 Internal structure of the FCOA ............................................................................. 126
Figure 4.20 Behaviour co-ordinator/Arbitration......................................................................... 128
Figure 4.21 Winner Control Cluster.......................................................................................... 129
Figure 4.22 Example of competitive arbitration......................................................................... 129
Figure 5.1 Mission planner as a state machine......................................................................... 135
Figure 5.2 Tracking way-points planner as a state machine.................................................... 136
Figure 5.3 Passing through a way-point during simulation....................................................... 136
Figure 5.4 Way-point guidance parameters............................................................................... 138
Figure 5.5 Different shapes of the tangent hyperbolic function.................................................. 139
Figure 5.6 Trajectories obtained with “Auto-Heading with Look-Ahead” Move to Target strategy and “Rotate towards Target without Surge-Brake” Rotate towards Target strategy for two different sizes of the virtual pool (scaling factor 5 (left) and 15 (right)). .......... 141
Figure 5.7 Missed waypoint handling criteria............................................................................ 142
Figure 5.8 Missed waypoint handling based on the concept of acceptance cone....................... 143
Figure 5.9 Typical trajectory for transect (segment) tracking.................................................... 143
Figure 5.10 Definition of the auxiliary point $A_k$ and normalised distance difference $\varepsilon_k$ ..........144

Figure 5.11 Magnitude factors for the desired heading correction: (left) the distance factor; (right) the turn factor .........................................................................................................................145

Figure 5.12 Transition between segments using “Transition to new Transect” scheme for two different layouts of way-points. .........................................................................................................................145

Figure 5.13 Definition of the aiding way-points. ..........................................................................................................................145

Figure 5.14 Transition to new transect using Bezier curve profiling. ..........................................................................................................................147

Figure 5.15 Two cases for staircase profile: $z_{k+1} > z_k$ (left), $z_{k+1} < z_k$ (right). In the first stage an auxiliary point $WP_i$ is reached; then, standard behaviours for motion in the horizontal plane are used to reach the target point $WP_{k+1}$. ..........................................................................................................................149

Figure 5.16 3D way-point guidance by the shortest path. ..........................................................................................................................149

Figure 5.17 Performance of the Auto-Heave controller operating in depth-tracking mode when the switch $S$ is open (left) and closed (right). Feedforward compensation is not active in the former case and tracking error is noticeable during tracking. Activation of feedforward compensation leads to cancellation of the tracking error. .................151

Figure 5.18 Various 3D way-point guidance scenarios: (a) constant altitude-keeping; (b) staircase profiling; (c) depth-tracking without feedforward compensation (switch $S$ open); (d) depth-tracking with feedforward compensation (switch $S$ closed). ..........................152

Figure 6.1 Functionally equivalent Proportional-Integral Controller with Anti-windup techniques implemented on Tethra (Initial design) for heading-keeping. .................................................................155

Figure 6.2 Testing of the simple Proportional-Integral controller with anti-windup techniques for heading control of the stationary Tethra. ..........................................................................................................................156

Figure 6.3 Testing of robustness and stability: After external disturbance has been applied, the heading controller returned Tethra back to the initial position. .................................................................156

Figure 6.4 Testing the active ballast system. ..........................................................................................................................158

Figure 6.5 Testing in the Arena sport centre (Tethra, New design). ..........................................................................................................................159

Figure 6.6 Open water testing of the vehicle in remotely operated mode, on the Lough Derg lake. ..........................................................................................................................159

Figure 6.7 Case A.1: Way-point Guidance using “Separate Auto-Heading & Cross-Track Compensation”, Approximation method: Scaling. ..........................................................................................................................169

Figure 6.8 Case A.2: Way-point Guidance using “Separate Auto-Heading & Cross-Track Compensation”, Approximation method: Truncation. ..........................................................................................................................170

Figure 6.9 Case A.3: Way-point Guidance using “Separate Auto-Heading & Cross-Track Compensation”, “Rotate towards Target without Surge Brake” ..........................................................................................................................171
Figure 6.10 Case A.4: Way-point Guidance using “Separate Auto-Heading & Cross-Track Compensation”, with external disturbances (ocean currents) .................................................. 172

Figure 6.11 Case A.5: Way-point Guidance using “Separate Auto-Heading & Cross-Track Compensation”, with obstacle avoidance and missed-waypoint handling .................. 173

Figure 6.12 Case B.1: Way-point Guidance using “Auto-Heading Integrated Cross-Track Compensation” ........................................................................................................... 174

Figure 6.13 Case B.2: Way-point Guidance using “Auto-Heading Integrated Cross-Track Compensation”, “Rotate towards Target without Surge Brake” ...................... 175

Figure 6.14 Case B.3: Way-point Guidance using “Auto-Heading Integrated Cross-Track Compensation”, with external disturbances (ocean currents) ......................... 176

Figure 6.15 Case B.4: Way-point Guidance using “Auto-Heading Integrated Cross-Track Compensation”, with external disturbances (ocean currents) and missed way-point handling .................................................................................................................... 177

Figure 6.16 Case C.1: Way-point Guidance using “Auto-Heading with Look-Ahead” ........ 178

Figure 6.17 Case C.2: Way-point Guidance using “Look-Ahead Transition to New Transect” . 179

Figure 6.18 Case D.1: Way-point Guidance using constant altitude keeping without feedforward compensation ............................................................................................................ 180

Figure 6.19 Case D.2: 3D Way-point Guidance using stair-case profile ................................. 181

Figure 6.20 Case D.3: 3D Way-point Guidance without feedforward compensation ............ 182

Figure 6.21 Case D.4: 3D Way-point Guidance with feedforward compensation ................ 183

Figure 6.22 Case D.5: 3D Way-point Guidance using constant depth keeping along transects and feed-forward compensation at “Look-Ahead Transition to New Transects” ........... 184

Figure 6.23 Case E.1: Way-point Guidance using “Transition to New Transect using the Aiding Way-points” ........................................................................................................... 185

Figure 6.24 Case E.2: 3D Way-point Guidance using constant depth keeping along transects and feed-forward compensation at “Transition to New Transect using the Aiding Way-points” ........................................................................................................... 186

Figure 6.25 Thrusters’ current draw for case A.1 ................................................................. 187

Figure 6.26 Thrusters’ current draw for case A.2 .................................................................. 187

Figure 6.27 Thrusters’ current draw for case A.3 ................................................................. 187

Figure 6.28 Thrusters’ current draw for case A.4 ................................................................. 187

Figure 6.29 Thrusters’ current draw for case A.5 ................................................................. 187

Figure 6.30 Thrusters’ current draw for case B.1 ................................................................. 187

Figure 6.31 Thrusters’ current draw for case B.2 ................................................................. 188

Figure 6.32 Thrusters’ current draw for case B.3 ................................................................. 188

Figure 6.33 Thrusters’ current draw for case B.4 ................................................................. 188

Figure 6.34 Thrusters’ current draw for case C.1 ................................................................. 188
Figure 6.35 Thrusters’ current draw for case C.2

Figure 6.36 Thrusters’ current draw for case D.1

Figure 6.37 Thrusters’ current draw for case D.2

Figure 6.38 Thrusters’ current draw for case D.3

Figure 6.39 Thrusters’ current draw for case D.4

Figure 6.40 Thrusters’ current draw for case D.5

Figure 6.41 Thrusters’ current draw for case E.1

Figure 6.42 Thrusters’ current draw for case E.2

Figure 6.43 Power consumption of thrusters for planar way-point guidance using “Separate Auto-Heading & Cross-Track compensation”, with “Rotate towards Target, with Surge-Brake” (case A.1) (left) and “Rotate towards Target, without Surge-Brake” (case A.3) (right)

Figure 6.44 Power consumption of thrusters for planar way-point guidance using “Auto-Heading Integrated Cross-Track compensation”, with “Rotate towards Target, with Surge-Brake” (case B.1) (left) and “Rotate towards Target, without Surge-Brake” (case B.2) (right)

Figure 6.45 Case MF-1: (a) Vehicle trajectory– top view (b) Thrusters’ power consumption (c) Membership functions for the input variables (overlapping region [1,2])

Figure 6.46 Case MF-2: (a) Vehicle trajectory– top view (b) Thrusters’ power consumption (c) Membership functions for the input variables (overlapping region [2,3])

Figure 6.47 Case MF-3: (a) Vehicle trajectory– top view (b) Thrusters’ power consumption (c) Membership functions for the input variables (overlapping region [3,4])
## List of Tables

<table>
<thead>
<tr>
<th>Table</th>
<th>Description</th>
<th>Page</th>
</tr>
</thead>
<tbody>
<tr>
<td>Table 4.1</td>
<td>Position vectors for thruster configuration of Tethra.</td>
<td>98</td>
</tr>
<tr>
<td>Table 4.2</td>
<td>Orientation vectors for thruster configuration of Tethra.</td>
<td>98</td>
</tr>
<tr>
<td>Table 4.3</td>
<td>Vertices of Ω and Φ.</td>
<td>110</td>
</tr>
<tr>
<td>Table 4.4</td>
<td>Nomenclature for Ω and Φ.</td>
<td>110</td>
</tr>
<tr>
<td>Table 4.5</td>
<td>Control allocation problem for motion in the horizontal plane.</td>
<td>113</td>
</tr>
<tr>
<td>Table 4.6</td>
<td>Coordinates of vertices of Ω&lt;sub&gt;HT&lt;/sub&gt; and Φ&lt;sub&gt;HT&lt;/sub&gt;.</td>
<td>113</td>
</tr>
<tr>
<td>Table 4.7</td>
<td>Control allocation problem for motion in the vertical plane.</td>
<td>115</td>
</tr>
<tr>
<td>Table 4.8</td>
<td>Coordinates of vertices of Ω&lt;sub&gt;HT&lt;/sub&gt; and Φ&lt;sub&gt;HT&lt;/sub&gt;.</td>
<td>115</td>
</tr>
<tr>
<td>Table 4.9</td>
<td>Fuzzy rule table.</td>
<td>127</td>
</tr>
<tr>
<td>Table 6.1</td>
<td>Group A: planar way-point guidance test cases using Separate Auto-Heading &amp; Cross-Track compensation.</td>
<td>161</td>
</tr>
<tr>
<td>Table 6.2</td>
<td>Group B: planar way-point guidance test cases using Auto-Heading Integrated Cross-Track compensation.</td>
<td>163</td>
</tr>
<tr>
<td>Table 6.3</td>
<td>Group C: Planar way-point guidance test cases using the Look-Ahead approach for smooth way-point transition.</td>
<td>165</td>
</tr>
<tr>
<td>Table 6.4</td>
<td>Group D: 3D way-point guidance test cases.</td>
<td>166</td>
</tr>
<tr>
<td>Table 6.5</td>
<td>Group E: Way-point guidance test cases using the aiding way-points.</td>
<td>167</td>
</tr>
<tr>
<td>Table 6.6</td>
<td>Cross-reference table linking test cases and current diagrams.</td>
<td>168</td>
</tr>
<tr>
<td>Table 6.7</td>
<td>Comparison of power and energy consumption for cases A.1 and A.3.</td>
<td>191</td>
</tr>
<tr>
<td>Table 6.8</td>
<td>Comparison of power and energy consumption for cases B.1 and B.2.</td>
<td>191</td>
</tr>
<tr>
<td>Table 6.9</td>
<td>Comparison of power and energy consumption for cases MF-1, MF-2 and MF-3.</td>
<td>194</td>
</tr>
</tbody>
</table>
Chapter 1: Introduction

1.1 Background

Many engaged in sub sea operations and exploration refers to the sub marine environment as ‘Inner Space’. The challenges in exploring and illuminating this ‘Inner Space’ are in many ways the equal of the challenges of exploration of outer space. Over Millennia, much has been discovered of outer space through feet firm on the ground observation, Sky gazing, and charting the heavens are human endeavours that mankind has engaged in back through long history. The advent of the modern telescope and latterly exploration with radio telescopes and spacecraft has resulted in an explosion in our knowledge of the heavens and the origins of the universe.

While it is possible to see heavenly bodies with the naked eye at vast remove it is by contrast barely possible to see more than a few meters through the water of the ocean. New life forms and treasures of the deep are coming to light in recent times at an ever increasing rate as underwater technology enables us to access the ‘Inner Space’ with underwater vehicles and sensor systems.

This thesis details the work of the author in the area of unmanned underwater vehicles and in the development of the autonomous underwater vehicle – Tethra. The work described covers a range of areas including research, conceptual design, detail design, embedded electronics, control systems development and testing. The result of this research and development has endowed Tethra with a comprehensively integrated Guidance Navigation and Control system, enabling the execution of full survey scale operations in challenging near seabed scenarios. In reality the developed solutions have now though outstripped the usefulness of Tethra, which was originally built as a laboratory R&D test bed vehicle for AUV experimentation on a low budget. Who knows,
with some unforeseen benefactor, or research-funding source, Tethra, which is the pride of the author, may yet be enhanced to Tethra II.

This thesis is also a partial contribution to the research project “MSR 3.2 - Deep Ocean Habitat Mapping Using an ROV”. This HEA PRTLI III funded project is collaboration between Mobile & Marine Robotics Research Centre (MMRRC) at University of Limerick and Earth & Ocean Science, NUIGalway. The main objective of MSR 3.2 is high-resolution video and sonar imaging up to 1300 m depth using an ROV with state-of-the-art navigation sensors. Within the MMRRC there is a group of researchers working on related marine robotic project areas. As well as the work of the author detailed in this thesis there is significant collaborative research underway involving the researchers in overlapping areas of marine robotics, control, sensor technologies and marine applications. Recent and ongoing research projects in the centre which overlap in parts with the work of this thesis include: AUV system identification and control using neural networks, design of sonar sensor solutions for proximal object detection and classification, deep ocean habitat mapping using an ROV with payload sensor (scanning sonars, camera systems and navigation sensors), development of a real-time high-resolution imaging sonar system simulator and development of a virtual underwater laboratory. This thesis is concerned about the development and design of novel GNC system for ROVs/AUVs. Together, the projects underway address many of the challenges to the development of AUVs for near seabed operations.

1.2 Motivation

The exploration and exploitation of the deep oceans is a challenging, expensive and time-consuming activity. High-resolution (acoustic & video) seabed survey requires deployment of sensors on platforms close to the seabed. The unstructured, harsh and hazardous nature of the ocean environment and close proximity to seabed and other
hazards produces serious problems for unmanned underwater vehicles (UUV) during the mission. Due to the complexity of the underwater environment, the vehicle control system must avoid obstacles and compensate for various external disturbances acting on the vehicle (sea currents, drag effects of umbilical, etc.) in the presence of uncertainties in determination of position and orientation (navigation errors). The physical shape and actuator configuration of the vehicle (number, position and orientation of thrusters & control surfaces) impose constraints that limit the freedom of control actions. All these factors add significant challenges to the task of maintaining the vehicle on the desired path and performing the survey mission. If the control system is not designed in an optimal way, tracking errors lead to unnecessary extension of survey mission time, increasing the survey mission costs. In addition, a non-optimal control allocation of actuators leads to inefficient usage of available energy resources. In the case of autonomous underwater vehicle (AUV), this can lead to undesired cancellation of the survey mission, due to lack of energy (discharged battery).

There are significant challenges to be addressed in the acquisition of high-resolution seabed surveys in both shallow and deep water. High-resolution imagery (video & sonar) requires that equipment is flown close to the seabed. Highly manoeuvrable autonomous underwater vehicles (AUVs) can accommodate such equipment and address the challenges.

Challenges for deep water high-resolution survey include:

- A remotely operated vehicle (ROV) needs the constant and costly attendance of a surface vessel (€10,000 – €20,000 and higher typical day rates for charter of National Institute survey vessels, rates in excess of €80,000 day rates for ships are not uncommon in the commercial Offshore Oil and Gas and other sectors).
• Long tethers and surface vessel operating sea conditions impose severe operating restrictions giving narrow weather windows.

• Mono-hull AUVs have recently become commercially available for survey but are generally only capable of operation remote from the seabed and hazards.

• High-resolution survey of rough terrain (reefs, sea mounds, etc) requires high manoeuvrability, not achievable with mono-hull AUVs with one or two thrusters and control surfaces.

• Thruster faults, which are more common in hazardous scenarios, generally cause the survey mission to be aborted and can cause the loss of the vehicle.

Challenges for inshore and harbour high-resolution survey include:

• In-shore and harbour waters are often more hazardous than the deep. The seabed can be strewn with debris (wrecks, moorings, submerged buoys, fishing nets, etc.). Natural hazards (rocks, reefs, kelp beds, shoals, currents, rips, wave zone, naturally occurring flotsam and jetsam) all represent potential threats to the operation of marine vehicles.

• Operation of any tethered or towed system in such hazardous situations is prone to failure through collision or snagging and vehicle/system loss.

• In such waters marine vessels usually stay within buoyed channels. For survey purposes vessels must per course go beyond the marked safe channel limits.

• Swath widths in shallow water are narrow requiring multiple surface vessel survey passes.

• Surface vessel passes may be restricted by shallows and other hazards / obstacles, making comprehensive surveys difficult.

The Mobile & Marine Robotics Research Centre at UL is developing a highly manoeuvrable AUV platform to address the challenges of high-resolution seabed survey
in both shallow and deep water. The overall system integrates state-of-the-art survey equipment, a multi-thruster open-frame AUV Tethra and an advanced guidance & control system. Tethra design concepts are described in chapter 3, while technical specifications can be found in Appendix B.

1.3 **Aims and objectives**

The aim of the research is design and development of guidance, navigation and control (GNC) system for open-frame thrusters-propelled ROVs/AUVs.

The specific objectives of the thesis are:

- Explore the existing methods for guidance, navigation and control for underwater vehicles.
- Identify methods that will be used as basis for novel GNC system to meet requirements of the research project.
- Formulate and solve control allocation problem for Tethra AUV.
- Design and develop a GNC system architecture.
- Develop the real-time simulator to test algorithms.
- Verify the performance of the GNC system in real-world applications.

1.4 **Overview of chapter contents**

Chapter 2, "Literature Review", provides an overview of different approaches proposed in the literature to solve the problem of guidance, navigation and control of thrusters-propelled open-frame underwater vehicles, including an overview of propulsion systems models, control allocation techniques; mobile robot control architectures and best practice postulates for implementation of a hybrid control architecture on AUVs. Methods used for way-point guidance systems and fuzzy logic control for autonomous navigation is described at the end of the chapter.
Chapter 3, "Tethra AUV Design Concepts", provides background into the development of the Tethra AUV. It details the design of the Tethra vehicle including mechanical configuration, thrusters, power electronics, actuators, embedded controllers, low budget sensor suite integration & testing, high precision instrument integration and control software.

Chapter 4, "Guidance, Navigation and Control for Tethra AUV", describes the overall system, which include a set of hardware and software solutions for guidance, navigation and control of the Tethra AUV. The main focus of this chapter is description of control system architecture, while guidance and navigation systems are described in chapters 5 and 3, respectively.

Chapter 5, "Way-Point Guidance System", provides in-depth insight into the main features of the proposed guidance system, including full description of behaviours used to perform different stages of a mission.

Chapter 6, "Testing & Evaluation", evaluates the performance of the proposed overall system both in real-world experiments and in computer simulations. In essence, this chapter effectively combines the material presented in previous chapters into a collection of representative cases in order to demonstrate the performance of the proposed/developed system in typical underwater scenarios.

Chapter 7, "Conclusions and Further Work", reviews the thesis, lists and describes the contributions and makes suggestions for further work.

In the main body "overview-type" material is presented, while more technical details and descriptions can be found in appendices, which are included at the end of the thesis. Appendices included in the thesis are listed in the following.

Appendix A describes the Tethra simulator.

Appendix B contains technical details and specifications for Tethra AUV.
Chapter 1: Introduction

Appendix C provides technical specifications of Phins INS, RDI DVL and Microbath depth/altitude sensors.

Appendix D contains a brief description of the general properties of a Kalman filter.

Appendix E contains a list of published and submitted papers produced during the course of the work described in the thesis. Copies of published papers are also included.

1.5 List of main contributions

The main contributions of the work presented in this thesis are summarised as follows:

- Decomposition of control problem for UUVs into regulation and actuator selection tasks.
- Full implementation of the hybrid control architecture with safety critical behaviours with highest priority.
- Formulation of the control allocation problem in normalised form and its solution, including clear geometrical interpretation.
- Introducing the control buffer concept with clusters for virtual joystick and low-level controller clusters.
- Development of Fuzzy Logic controller for 3D obstacle avoidance.
- Realisation of mission & tracking way-point planners as state machines.
- Fusion of task executors using arbitration.
- Design of a simulation model with virtual reality display.

These contributions are discussed throughout the thesis and summarised in section 7.4.
Chapter 2: Literature Review

2.1 Introduction

This chapter provides an overview of different approaches proposed in the literature to solve the problem of guidance, navigation and control of thrusters-propelled open-frame underwater vehicles. Outlines of the main methods and recent advances are given in this chapter, while terminology and a full mathematical description of methods relevant to the work in this thesis can be found in chapters 4 and 5. The chapter is organised as follows: Section 2.2 provides recent advances in the modeling of propulsion systems and control allocation. Section 2.3 briefly describes control architectures for mobile robots and AUVs. Methods for way-point guidance systems are described in section 2.4. A brief description of fuzzy logic control techniques employed for autonomous navigation is given in section 2.5. The concluding remarks are presented in section 2.6.

2.2 Propulsion system modeling & control allocation

This section describes the model of a propulsion system, used to obtain the expression of the axial thrust and torque developed by a thruster, as function of the propellers angular velocity. Different models of propulsion system were proposed in literature (Yoerger, Cooke et al., 1990), (Healey, Rock et al., 1995), (Blanke, Lindegaard et al., 2000). A short review of these models is given in this section, while more information can be found in (Fossen, 2002).

2.2.1 One-state model

In (Yoerger, Cooke et al., 1990) it was found that the ability of thrusters to produce force is greatly influenced by axial and cross-flow effects. The one-state model for propeller shaft speed $n$ with propeller thrust $T$ as output can be written as:
Chapter 2: Literature Review

\[ J_n \dot{n} + K_n n = \tau \]  \hspace{1cm} (2.1)

\[ T = T(n,u_p) \]  \hspace{1cm} (2.2)

where \( u_p \) is the axial flow velocity in the propeller disc and \( \tau \) is the control input (shaft torque). The main drawback of the one-state model is its inability to describe overshoots in thrust, but instead refers to a situation where the speed of response depends on the commanded thrust level (Yoerger, Cooke et al., 1990). (Healey, Rock et al., 1995) have determined from experimental data that thrust response to stepwise inputs exhibits an overshoot. (Healey, Rock et al., 1995) have changed the one-state model, to describe the overshoots, which are observed under conditions of rapid command changes.

### 2.2.2 Two-state model

The major contribution of the model developed in (Healey, Rock et al., 1995) is that it includes two state variables (motor shaft speed \( n \) and water velocity \( u_p \)), as opposed to the one-state model proposed by (Yoerger, Cooke et al., 1990):

\[ J_n \dot{n} + K_n n = \tau - Q_e \]  \hspace{1cm} (2.3)

\[ m \dot{u}_p + d_f (u_p - u) (u_p - u) = T \]  \hspace{1cm} (2.4)

\[ T = T(n,u_p) \]  \hspace{1cm} (2.5)

\[ Q_e = Q_e(n,u_p) \]  \hspace{1cm} (2.6)

where \( u \) is the forward speed of the vehicle and \( Q_e \) is the propeller torque. The model was obtained by modeling a control volume of water around the propeller as a mass-damper system (Blanke, Lindegaard et al., 2000).
2.2.3 Three-state model

For a fixed pitch propeller, the nonlinear state equations have three states: propeller shaft speed $n$, vessel speed relative to the water $u$ and inflow velocity at the propeller disc (ambient water speed) $u_a$ (Blanke, Lindegaard et al., 2000):

\[
J_m \ddot{n} + K_m n = \tau - Q_e \tag{2.7}
\]

\[
m_f \dot{u}_p + d_f u_p + d_f |u_p| (u_p - u_a) = T \tag{2.8}
\]

\[
(m - X_u) \dot{u} - X_u u - X_u |u| = (1 - t)T \tag{2.9}
\]

\[
T = T(n, u_p) \tag{2.10}
\]

\[
Q_e = Q_e(n, u_p) \tag{2.11}
\]

where damping in surge is modelled as the sum of linear laminar skin friction, $-X_u u$ and non-linear quadratic drag, $-X_u |u|$. Similarly, linear damping, $d_f u_p$, is included in the axial flow model, since quadratic damping, $d_f |u_p| u_p$, alone would give an unrealistic response at low speeds. Linear skin friction gives exponential convergence to zero at low speeds (Fossen, 2002).

The ambient water velocity $u_a$ is computed by using the steady-state condition:

\[
u_a = (1 - w)u \tag{2.12}
\]

where $w, 0 < w < 1$ is the wake fraction number.
Chapter 2: Literature Review

2.2.4 Propeller thrust and torque modeling

In addition to propeller shaft speed $n$, vessel speed relative to the water $u$ and ambient water speed $u_a$, other dynamics effects will influence the propellers thrust and torque (Fossen, 2002). The most significant effects are (Fossen, 2002):

1. Air suction;
2. Cavitation;
3. In-and-out-of-water-effects (Wagner’s effect);
4. Wave influenced boundary layer effect;
5. Kuessner effect (gust).

For deeply submerged vehicles the first four effects can be neglected, and the Kuessner effect causes usually small rapid oscillating thrust components. Under these assumptions, the thrust and torque model can be realized using a quasy-steady representation (Fossen, 2002).

2.2.5 Quasi-steady thrust and torque

Quasi-steady modeling of thrust and torque is usually done in terms of lift and drag curves, which are transformed to thrust and torque by using the angle of incidence (Fossen, 2002). The non-dimensional thrust and torque coefficients are found by neglecting the unsteady flow effects, measuring and using steady-state values for $T$, $Q$ and $n$. The mathematical expressions for $Q$ and $T$ (bilinear thruster model) is given by:

$$T = T_{\|n\|} |\nu| - T_{\|u_a\|} |\nu| u_a$$
$$Q = Q_{\|n\|} |\nu| - Q_{\|u_a\|} |\nu| u_a$$

(2.13)

where $T_{\|n\|}$, $T_{\|u_a\|}$, $Q_{\|n\|}$ and $Q_{\|u_a\|}$ are positive propeller coefficients given by the propeller characteristics.
Chapter 2: Literature Review

Thruster modeling (Caccia, Indiveri et al., 2000)

According to Caccia et al. (2000), in many application the thruster dynamics can be neglected with respect to the vehicle dynamics, as the servo velocity loop of the controlled thruster system has a negligible time constant with respect to the overall vehicle’s time constant (Caccia, Indiveri et al., 2000), (Bruzzone, Bono et al., 2001).

The propeller thrust is modeled as:
\[
\tau = c_v n|n| - c_v n v_a
\]  

(2.14)

where \(v_a\) is the velocity of the fluid through the propeller’s blade and \(-c_v n v_a\) is considered as a saturation term.

Neglecting the motor dynamics, the expression of the thruster force is derived from the conventional form of the bilinear thrusters model (2.14), and is modeled as:
\[
\tau = c_v V V
\]  

(2.15)

where \(c_v\) is an unknown constant and \(V\) is the control voltage applied to the thrusters servo-amplifiers.

Since the thruster’s time constant has been neglected with respect to the overall vehicle’s one, the propulsor-applied voltage \(V\) is assumed to be proportional (2.15) to the propeller revolution rate \(n\). The voltage-to-thrust coefficients \(c_v\) are identified with thrust tunnel experiments (Alessandri, Caccia et al., 1998), (Caccia, Indiveri et al., 2000).

As stated by Caccia, for open-frame vehicles the propeller-hull interactions cannot be neglected (Caccia, Indiveri et al., 2000). Thruster efficiency is reduced when the propeller boosts the water against the vehicle’s hull. A thrusters installation coefficient \(\eta\) is
introduced, based on the assumption that the propeller works in free water when the revolution rate \( n \) is positive (Caccia, Indiveri et al., 2000):

\[
\eta = \begin{cases} 
1, & \text{if } n \geq 0 \\
\eta^*, & \text{otherwise}
\end{cases}
\]

Parameter \( \eta^* \) from (2.16) is determined by in-water tests of the vehicle. Therefore, the equation of the thrust assumes the form \( \tau = \eta c_r n |\eta|, 0 < \eta \leq 1 \) (Bruzone, Bono et al., 2001).

### 2.2.6 Propeller thrust and torque modeling

In the general case a UUV has \( p \) thrusters \( T_{th_1}, T_{th_2}, \ldots, T_{th_p} \). Each thruster \( T_{th_i}, i = 1, p \) exerts thrust (force) \( T \) and torque (moment) \( Q \). The position vector \( r = [r_x, r_y, r_z]^T \) determines the position of the point of attack of the force \( T \), relative to \( \{B\} \), the body fixed reference frame. The force \( T \) also generates the moment \( Q = r \times T \), such that the total moment vector exerted by the thruster is given by \( Q = Q_c + Q \) (Omerdic, 2004). The orientation of the thruster \( T_{th} \) relative to the \( \{B\} \) is defined by the unit vector \( e = [e_x, e_y, e_z]^T \). The vector \( e \) shows the positive direction of the force \( T \). The contributions of each thruster are summed together to form the vector of propulsion forces and moments \( \tau \):

\[
\tau = \begin{bmatrix} T \\ Q \end{bmatrix} = \begin{bmatrix} \sum_{i=1}^{p} T_i \\ \sum_{i=1}^{p} Q_i \end{bmatrix}
\]

Figure 2.1 displays typical \( T \)-curves as a function of propeller angular velocity \( n \) and the ambient water velocity \( u_a \). Typical parabolic shape is deformed, depending on the size and
sign of $u_a$. For example, for the same angular velocity $n$, the propeller will generate a larger force in the case $u_a = 0$ than for $u_a > 0$. This can be confirmed in the Tethra simulator (Appendix A) by changing the thruster model (bilinear/affine) and monitoring force responses.

\[ T(n) = T_{\alpha}n|\|n\|n| \]
\[ Q_\alpha(n) = Q_{\alpha}|n|\|n\| \]  \hspace{1cm} (2.18)

**Figure 2.1** Propeller thrust (force) $T$ as a function of propeller revolution $n$ and ambient water velocity $u_a$ (bilinear thruster model).

In practical implementations, for vehicles moving at slow speeds, the bilinear thruster model (2.13) can be approximated using the *affine thruster model*, neglecting the ambient water velocity, thus $u_a = 0$ (Ridao, Tiano et al., 2004):


2.2.7 Non-symmetrical propeller T-curves

It is important to note that the efficiency of the propeller may not be the same for both possible spin directions, i.e. for the same propeller revolution \( |n_1| = |n_2| \) the corresponding thrusts (forces) may not be equal \( |T_1| > |T_2| \). This means that, in real applications, the \( T \)-curve is not symmetrical, i.e. \( T(n) \) is not a odd function of \( n \). An efficient way to compensate for non-symmetrical propeller \( T \)-curve is proposed in (Omerdic, 2004). Essentially, in order to compensate for non-symmetry, an auxiliary control variable \( u' \) is introduced, making the propeller \( T \)-curve temporarily symmetrical. In this way, relationship between thrust (force) \( T \) and \( u' \) is linear and ranges of variables are symmetrical. As a result, it is easy to normalise control constraints and the control allocation problem can be formulated in the normalised form, which is more understandable and easier to solve.

The same approach is used to compensate for non-symmetrical \( T \)-curves of Tethra’s propellers. Results of bollard pull tests performed with the Tethra AUV in September 2004 showed that the maximum thrust (force) exerted by a thruster in one direction is almost two times higher than in the opposite direction. This non-symmetry imposes significant challenges to control the system to perform tracking tasks accurately. However, evaluation results in Chapter 6 demonstrate that the proposed control system is able to fully compensate for non-symmetrical propeller \( T \)-curves.

2.2.8 Vectorisation

In order to simulate forces and moments exerted by thrusters, it is necessary to use vector variables (Omerdic, 2004). Each force and moment needs to be decomposed in the body-
fixed frame \( \{B\} \). In the following, it is assumed that the position and the orientation of the thruster is determined by vectors \( \mathbf{r} \) and \( \mathbf{e} \), respectively (see Figure 2.2).

\[ \mathbf{r} \text{ and } \mathbf{e} \]

**Figure 2.2** Thrust and torque as vector variables.

**Bilinear thrusters model in vector form**

The forward speed \( u \) in (2.12) should be interpreted as a projection of the linear velocity vector \( \mathbf{v}_i = [u \ v \ w]^T \) on the orientation vector \( \mathbf{e} \) (Omerdic, 2004). Consequently, the bilinear thruster model can be represented in vector form as

\[
\begin{align*}
\mathbf{u}_a &= (1-w)^9 \mathbf{v}_i^T \cdot \mathbf{e} \\
T(n, u_a) &= T_{n|p|n}|n|-T_{n|p|e}|n|u_a, \quad T_{n|e|n} = \begin{cases} T_{n|p|n}^+ & n > 0 \\ T_{n|p|n}^- & n < 0 \end{cases} \\
\mathbf{T} &= T\mathbf{e} \\
Q_e(n, u_a) &= Q_{n|p|e}|n|-Q_{p|e}|n|u_a \\
Q_e &= cQ_e \mathbf{e}, \quad Q_r = \mathbf{r} \times \mathbf{T} \\
Q &= Q_e + Q_r 
\end{align*}
\]

(2.19)

where spin direction coefficient \( c \) is equal to +1 (-1) if the force vector \( \mathbf{T} \) and the torque vector \( Q_e \) have the same (opposite) direction.
Affine thrusters model in vector form

The vector form of the affine thruster model is obtained from the bilinear thruster model (2.19), assuming $u_a = 0$:

$$T(n) = T_{a|n|n}, \quad T_{a|n|} = \begin{cases} T^+_{a|n|}, & n > 0 \\ T^-_{a|n|}, & n < 0 \end{cases}$$

$$T = Te$$

$$Q_e(n) = Q_{e|n|n}$$

$$Q_e = cQ_e, \quad Q_r = r \times T$$

$$Q = Q_e + Q_r$$

(2.20)

2.2.9 Dynamic model of permanent magnet DC-motor

The dynamics of permanent magnet DC-motors is analysed in the following, since they are commonly used as actuators in underwater vehicles. Other types of motors, in particular AC-motors and so-called brushless DC-motors are also used as actuators for UUVs, but they will not be discussed here.

A DC-motor works on the principle that a current-carrying conductor in a magnetic field experiences a force (Spong, Hutchinson et al., 2006). The motor itself consists of a fixed stator and a movable rotor that rotates inside the stator. A commutator is required to periodically switch the direction of the current through the armature to keep it rotating in the same direction.

The stator of permanent magnet DC-motor consists of a permanent magnet. The torque on the rotor is then controlled by controlling the armature current.
Consider the schematic diagram of Figure 2.3, where

\[ L = \text{armature inductance [H],} \]
\[ R = \text{armature resistance [Ω],} \]
\[ V_a = \text{armature voltage [V],} \]
\[ i_a = \text{armature current [A],} \]
\[ E_b = \text{back emf [V],} \]
\[ K_m = \text{torque constant [Nm/A],} \]
\[ K_b = \text{back emf constant [Vs/rad],} \]
\[ J_m = \text{motor inertia [kgm}^2]\text{],} \]
\[ \omega = \text{shaft angular velocity [rad/s],} \]
\[ \tau_m = \text{generated torque [Nm],} \]
\[ \tau_i = \text{load torque [Nm],} \]
\[ \phi = \text{magnetic flux due to stator [Wb],} \]

The differential equation for the armature current is then
Chapter 2: Literature Review

\[
L \frac{di_a}{dt} + Ri_a = V_a - E_b
\]  
(2.21)

Since the flux \( \phi \) is constant in case of permanent magnet DC-motor, the torque developed by the motor is

\[
\tau_m = K_m \phi_i = K_m i_a
\]  
(2.22)

The back emf is

\[
E_b = K_b \phi \omega = K_b \omega
\]  
(2.23)

It can be shown that the numerical values of \( K_m \) and \( K_b \) are the same provided SI system of units is used.

The mechanical part of the motor equations is derived using Newton’s law, which states that the product of inertial load and the derivative of angular rate is equal to the sum of all torques about the motor shaft. This results in the equation:

\[
J_m \frac{d\omega}{dt} = \sum \tau_j = K_m i_a - \tau_f - \tau_i
\]  
(2.24)

where \( \tau_f \) is a total friction (including the Coulomb friction and the viscous friction) and \( \tau_i \) is a propeller load. The Robotic Systems Lab in the Australian National University has been developing an AUV, Kambara, since 1998 (Silpa-Anan, 2001). A thruster model, including models for motor friction and propeller load, is published in (Silpa-Anan, 2001). Since both Tethra and Kambara are actuated by the same model of thrusters, i.e. MinnKota electric outboard trolling motors, the same models of friction and propeller load as in (Silpa-Anan, 2001) are utilised in the Tethra simulator.

The motor friction is modelled as:
\( \tau_f = k_{f0} \text{sgn}(\omega) + k_{f1} \omega \)  

(2.25)

where \( k_{f0} = 4.80 \times 10^{-3} \) and \( k_{f1} = 1.48 \times 10^{-3} \).

The propeller load is modelled as

\[ \tau_l = a_0 \omega^3 + a_1 \omega^2 + a_2 \omega \]  

(2.26)

where \( a_0 = 9.31 \times 10^{-7} \), \( a_1 = -2.75 \times 10^{-6} \) and \( a_2 = 2.14 \times 10^{-3} \).

Figure 2.4 Propeller friction and load as function of shaft angular velocity.

A block diagram of DC-motor dynamics, based on equations (2.21) - (2.26), is shown in Figure 2.5. This block diagram is vectorised and used in the Tethra simulator as part of the propulsion system (see Appendix A for more information about the Tethra simulator).
Estimated parameters for the DC motor model are given in the following:

\[
J_m = 8.12 \times 10^{-3} [kgm^2] \\
K_M = K_b = 0.1235 \\
L = 0.0523 [H] \\
R = 0.208 [\Omega] 
\] (2.27)

### 2.2.10 Control allocation methods

The control allocation problem can be defined as the determination of the actuator control values that generate a given set of desired or commanded forces and moments. Different methods for problem solution are briefly described here, while more details with description of terminology and precise mathematical formulation can be found in section 4.5.2.
Chapter 2: Literature Review

**Direct control allocation method**

In (Durham, 1993) a method is described for the solution of the constrained control allocation problem. The method is based on the fact that, for overactuated systems, the linear mapping $\mathbf{m} = \mathbf{B} \mathbf{u}$, $\mathbf{m} \in \Phi$, $\mathbf{u} \in \Omega$ is many-to-one on the interior of $\Phi$ and one-to-one on the boundary of $\Phi$, under assumption that any $k$ columns of the *control effectiveness matrix* $\mathbf{B}$ are linearly independent (linear independency condition), where $k$ is the number of rows of $\mathbf{B}$. The first part of the method involves determining the boundary of the *Attainable Moment Set* (AMS), denoted by $\Phi$. The second part concerns the determination of the desired control (true control vector) in three steps. In the first step, the desired moment vector $\mathbf{m}$ is extended, the intersection point $\mathbf{m}^*$ with AMS is determined and the scaling factor $f$ such that $\mathbf{m}^* = f\mathbf{m}$ is calculated. In the second step, the unique control vector $\mathbf{u}^*$ such that $\mathbf{m}^* = \mathbf{B}\mathbf{u}^*$ is found. Finally, the control vector is scaled with the inverse scaling vector $1/f$ to find $\mathbf{u} \ (\mathbf{u} = (1/f)\mathbf{u}^*)$, which produce $\mathbf{m}$. The original algorithm (Durham, 1993) was slow and difficult to implement. An elegant approach, proposed in (Durham, 1994) reduced considerably the number of computations. (Petersen and Bodson, 1999) proposed a fast implementation using spherical coordinates and look-up tables.

**Generalised inverse**

For systems where the number of controls is equal with the number of systems being controlled, a straightforward method is to invert the control effectiveness matrix. For over-actuated systems this approach can be extended using the Generalised Inverse matrix (Omerdic, 2004). The most common choice for a Generalised Inverse solution is the Moore-Penrose pseudoinverse:
Generalised inverse solutions have the advantages of being relatively simple to compute and allowing some control in distribution of control energy among the available effectors (Omerdic, 2004).

\[
B^+ = B^T(BB^T)^{-1}
\]  \hspace{1cm} (2.28)

**Daisy chain method**

The method of daisy chaining allows a specified set of controls to be used only when the rest of the controls fail to achieve the desired moment (Bordignon, 1996).

The controls are divided into two groups: the first group, \(u_1\), consists of the controls which may be used at all times (conventional aerodynamic/control surfaces), and the second group, \(u_2\), contains the control which will be used only when the first group of control fails to generate the desired moment (controls which move slower or are subjected to higher stresses) (Bordignon, 1996). The control effectiveness matrix is partitioned to correspond to these control groupings:

\[
Bu = \begin{bmatrix} B_1 & B_2 \end{bmatrix} \begin{bmatrix} u_1 \\ u_2 \end{bmatrix} = B_1u_1 + B_2u_2
\]  \hspace{1cm} (2.29)

Daisy chaining control allocation enables limiting the usage of certain control effectors.

**Optimisation based methods**

The control allocation problem can be pragmatically seen as a determination of the feasible control vector \(u\) for a given vector of objectives (moments) \(y\), such that \(Bu = y\). If the solution is not unique, the best one must be found. If the solution does not exist, vector \(u\) has to be determined such that \(Bu\) approximates \(y\) as well as possible. The following mathematical formulations of the control allocation have been proposed in (Bodson, 2002):
Chapter 2: Literature Review

Error minimization problem:

Given a matrix $B$ find a vector $u$ such that $J = \|Bu - y\|$ is minimised, subject to $u_{min} \leq u \leq u_{max}$.

Control minimisation problem:

Given a matrix $B$, a vector $u_p$ and a vector $u_i$ such that $u_{min} \leq u_i \leq u_{max}$, find a vector $u$ which minimises $J = \|u - u_p\|$ subject to $Bu = Bu_i$ and $u_{min} \leq u \leq u_{max}$.

Mixed optimisation problem:

Given a matrix $B$, a vector $y$ and a vector $u_p$, find a vector $u$ which minimises $J = \|Bu - y\| + \varepsilon \|u - u_p\|$ subject to $u_{min} \leq u \leq u_{max}$.

Control allocation method of over-actuated unmanned underwater vehicles

A hybrid approach for control allocation of overactuated unmanned underwater vehicles is proposed in (Omerdic, 2004). Standard pseudoinverse method for solution of the control allocation problem is able to find a feasible solution only on a subset of the attainable command set. This subset is called the feasible region for pseudoinverse. Some other methods, like direct control allocation or fixed-point iteration method, are able to find the feasible solution on the entire attainable command set. A novel, hybrid approach for control allocation, proposed in (Omerdic, 2004), (Omerdic and Toal, 2005), is based on integration of the pseudoinverse and the fixed-point iteration method. It is implemented as a two-step process. The pseudoinverse solution is found in the first step. Then the feasibility of the solution is examined analysing its individual components. If violation of actuator constraint(s) is detected, the fixed-point iteration method is activated in the second step. In this way, the hybrid approach is able to allocate the exact solution, optimal in the $l_2$ sense,
inside the entire attainable command set. This solution minimises a control energy cost function, the most suitable criteria for underwater applications.

### 2.3 Control architectures

A review of the theory for mobile robot control architectures is given below in section 2.3.1. It discusses the different types of architectures available, and emphasises the advantages and limitations of each of them. Control architectures applicable for AUVs are then described in section 2.3.2.

#### 2.3.1 Control architectures for autonomous mobile robots

The control architecture of a mobile robot defines how the job of generating actions from perceptions is organized. In order for a mobile robot to function usefully and reliably in unstructured and dynamic environments, it must be able to perceive its surroundings and generate a set of appropriate actions (Arkin, 1998), (Brooks, 1999). This requires the use of an underlying architectural framework, which is responsible for the sensing and reasoning processes of the robot. The architecture must be able to gather perceptual and state information and generate a set of actions for the robot to follow (see Figure 2.6). The architecture is basically a collection of software building blocks used to construct the robot's control system (Arkin, 1998). During the past decades, two different types of control architectures have dominated the robotics scene. The two architectures (illustrated in Figure 2.7) differ in the type of reasoning they use (Arkin, 1998). One uses deliberative reasoning to determine what actions should be taken. The other uses reactive reasoning to tightly couple sensor input with actuator output in a reactive way. Figure 2.7 shows a comparison of the two methods.
The deliberative / hierarchical control, also known as planner-based control, is the “classical” AI approach to robot control. Implemented examples include Shakey, a robot designed and built at the Stanford Research Institute in the late 1960s (Nilsson, 1969). In the hierarchical
architecture information from the world in the form of sensor data has to filter through a
number of intermediate stages of interpretation before finally becoming available for a
response. These stages or subsystems between sensors and actuators are said to be
horizontally organised. Each subsystem is then a complex program in itself and all
subsystems must work together perfectly for the robot controller to function at all (Arkin,
1998).

These subsystems would typically include:

1. A perception subsystem to manage sensing devices connected to the robot.
2. A world modeling subsystem to convert the sensory input into a description of where
   the robot is in relation to its internal model of the environment.
3. A planning subsystem to plan how the robot will achieve its goals given the current
   world state.
4. A task execution subsystem to produce detailed motion commands based on the plan
   generated.
5. A motor control subsystem where commands are put into effect.

The control flow for this scheme is illustrated in Figure 2.8.

![Figure 2.8 Hierarchical control flow.](image)

There are many serious disadvantages manifested in this architecture. To construct a
symbolic representation for the world assumptions must be made which may severely restrict
the usefulness of the representation. There is no direct communications between high-level control and low-level peripherals (sensors, actuators). Early robots that used this type of control approach demonstrated high levels of sophistication but were very limited in their usefulness as they were incapable of operating outside of controlled environments (Arkin, 1998), (Brooks, 1999). Environmental and sensory noise could overwhelm their internal world model, making these robot controllers unreliable. Another disadvantage with this approach is that response time (time between sensor input and system action) is long. Early robots using purely deliberative control were frequently unable to perform computations fast enough to demonstrate true dynamic reactive behaviour when dealing with unforeseen situations (Arkin, 1998). As the complexity of the environment increases, the time needed to perceive, model and plan about the world increases exponentially (Flanagan, Toal et al., 2003). An additional disadvantage of this architecture is its lack of flexibility, sensor fusion/integration is difficult. For example to add a new sensor component to the system, significant modifications must be made to the whole system.

**Reactive control**

Reactive control evolved as an alternative to traditional “deliberative control” methods in an effort to overcome the problems associated with the latter. The control approach is modelled on the reflexive actions exhibited by living creatures such as insects (Arkin, 1998).

The study of animal behaviour can provide models that a roboticist can operationalise within a robotic system. Roboticists have struggled to provide their machines with animals’ simplest capabilities: the ability to perceive and act within the environment in a meaningful and purposeful manner. The animal behaviour provides: a definition for intelligence; an existence proof for the creation of intelligent mobile systems; and models that roboticists can mimic or
from which they can draw inspiration (Arkin, 1998). Instead of constructing a world model, planning a course of action within that model and mapping that plan into specific actions, reactive architectures are designed to respond directly to sensor stimuli from the environment. A reactive robotic system tightly couples perception to action without the use of intervening abstract representations or time history (Arkin, 1998). A collection of predefined rules governs the relationship between actuator response and sensor stimuli.

By sensing the environment at a rapid rate, uncertainty in perception is avoided. Any false readings, which are obtained, will only have a very limited impact (Arkin, 1998). By continuously acting on the perceived world, any uncertainty on what actions are to be carried out is also avoided. Reactive architectures have a very fast response time. The guiding principle behind reactive control is the belief that "the world is its own best model" (Brooks, 1986). When introduced, reactive control systems demonstrated navigation capabilities that were quicker and more capable than those of planning systems. For a simple environment the robot designer is able to predict most of the events that might occur and construct suitable responses to guide the robot to achieve its goal.

There are disadvantages to using a purely reactive control approach. A purely reactive robot lacks the run-time flexibility and representational power (the ability to learn and adapt its responses to stimuli) that are needed in a truly dynamic real world environment. It is difficult for a purely reactive system to execute a specific mission plan or to become more proficient in its actions (Arkin, 1998), (Brooks, 1999).

**Behaviour-based control**

The main aim of behaviour-based robotics is to eliminate the reliance on a centralized, complete representation of the world state. Internal state is needed only to keep track of those
aspects of the world state that are inaccessible to the sensors and are required for action selection in each behaviour (Russell and Norvig, 1995).

Behaviour-based robotic systems serve best when the real world cannot be accurately characterized or modelled (Arkin, 1998). Behaviour-based robotic architectures were developed in response to the fact that uncertainty, unpredictability and noise from the world cannot be removed, and they choose instead to deal with these issues.

The theory of behaviour-based control is grounded in the idea that complex behaviour in a robot emerges through the competition and/or co-operation of simple behaviours in the context of an environment. Within the behaviour-based control paradigm the use of symbolic representational knowledge (internal world model) is generally viewed as an impediment to efficient and effective robot control (Arkin, 1998).

Behaviour-based control representations are parallel, distributed, and reactive in order to accommodate real-time demands. A behaviour is a routine, which performs a certain set of actions in response to a given stimulus from sensors (Arkin, 1998). Each behaviour realises an individual connection between some kind of sensor data and actuation. The whole system is built step by step from a very low level. Successive levels can be added incrementally to enhance the functionality of the robot.

Behaviour-based architectures have the following general characteristics (Arkin, 1998):

- Behaviours serve as the basic building blocks for robotic actions. Each behaviour consists of a sensorimotor pair. Information from the sensor determines the motor reflex response.
• Use of explicit abstract representational knowledge is avoided in the generation of a response. Purely reactive systems react directly to the world as it is sensed. This avoids the need for intervening abstract representation knowledge.

• These architectures are inherently modular from a software design perspective. This allows the competency of the robot to be increased by adding new behaviours, without redesigning or discarding the old ones.

• Interaction of the robot is through the robot environment.

• Behaviours are relatively simple and tend to be more reactive than deliberative

In behaviour-based robotics the control problem is decomposed into a collection of task-achieving modules or behaviours. A behaviour can be described as a process or control law that achieves and/or maintains a goal.

For example, an ‘obstacle avoidance’ behaviour maintains the goal of preventing collisions. Each behaviour can take inputs from the robot's sensors (e.g., camera, sonar, tactile) or from other behaviours in the system, and send outputs to the robot's actuators (e.g., thrusters, control surfaces, valves, etc.) or to other behaviours.

Behaviours can be implemented incrementally, starting from a very low level, and gradually growing in complexity to enhance the functionality of the robot. The structure of a behaviour-based system can be represented as a global architecture with behaviours organised in parallel and outputs channelled into a coordinator mechanism that produces an appropriate response. This structure is illustrated in the diagram in Figure 2.9.
Figure 2.9 Behaviour-based architecture.

Behaviour-based control differs from purely reactive control by the way complementary and contradictory actions are “arbitrated” to arrive at the final robot action. The fusion schemes used by the arbitrator(s) are usually designed in such a way as to place higher weights on more critical behaviours such as obstacle avoidance and failsafe functionality (Toal, 2004).

The main robot control architectures that employ a behaviour-based rationale are: Brooks’ ‘subsumption architecture’ (Brooks, 1986), Arkin’s ‘schema-based’ architecture (Arkin, 1987), (Arkin, 1998), the Process Description Language (Steels, 1992), and Action Selection Dynamics (Maes, 1989).

A subsumption-like, layered control architecture has been chosen for use in mobile and autonomous underwater robot development at the University of Limerick and as such will be discussed in more details than the others.

**Subsumption architecture (Brooks, 1986)**

The subsumption (or layered) architecture was first proposed as a method for the control of robots by Rodney Brooks at Massachusetts Institute of Technology (Brooks, 1986). Subsumption is a method of reducing a robot’s control architecture into a set of task achieving behaviours or competences represented as separate layers (Carreras, Batlle et al., 2000). In the subsumption control approach, the control system is decomposed into a number
of horizontally arranged layers. Data and control are distributed through all layers, and each layer processes its own information (sensory and commands), i.e. there is no global data structure. Each layer in the subsumption architecture incorporates elements of all the vertical tasks found in the classical AI system such as perception, planning, task execution, and motor control. A given layer in the architecture is capable of implementing a particular behaviour or “competence” such as the ability to move away from an obstacle, to follow a beacon, or to explore the robots environment. All of these layers work in parallel, without a high-level supervisor (Brooks, 1986).

This decomposition leads to a different architecture, with different implementation strategies possible at the hardware level and with a large number of advantages concerning robustness, buildability and testability (Arkin, 1998).

The key aspects for the design of subsumption-style robots are situatedness and embodiment. *Situatedness* refers to the robot ability to sense its current surroundings and avoid the use of abstract representations. *Embodiment* represents that the robots are physical creatures and thus experience the world directly rather than through simulation (Arkin, 1998).

**Levels of competence**

A level of competence is an informal specification of a desired class of behaviours for a robot over all environments it will encounter (Brooks, 1986). In subsumption, the architecture is built using a number of levels of competence. Brooks used the following levels of competence:

0. Avoid contact with objects (whether the objects move or are stationary)

1. Wander aimlessly around without hitting things.
2. ‘Explore’ the world by seeing places in the distance, which look reachable and heading for them.

3. Build a map of the environment and plan routes from one place to another.

4. Notice changes in the static environment.

5. Reason about the world in terms of identifiable objects and perform tasks related to certain objects.

6. Formulate and execute plans, which involve changing the state of the world in some desirable way.

7. Reason about the behaviour of objects in the world and modify plans accordingly.

An important point is that higher levels of competence include as a subset each earlier level of competence. Since a level of competence defines a class of valid behaviours it can be seen that higher levels of competence provide additional constraints on that class (Brooks, 1986). The robustness of the robot’s control system grows as the level of competence increases. Taking the ‘explore’ competence as an example. Since this includes as a subset the lower level ‘wander’ and ‘avoid’ competences, the robot can operate under the influence of the ‘explore’ competence safe in the knowledge that it will not collide with an obstacle.

Layers of control

The key idea of levels of competencies is that we can build layers of a control system corresponding to each level of competence and simply add a new layer to an existing set to move to the next level of competence (Brooks, 1986). Layers can be built incrementally in subsumption so new competences are built upon existing ones. When designing the control system in subsumption, the level 0 competence is added first. This competence is the most fundamental ability the robot must have and is usually a capability necessary for survival.
such as obstacle avoidance ability. A control layer is added which achieves this. Once a reliable and functional level 0 competence has been tested and debugged, we have a working control system and further control layers can be added to the control system. A typical level 0 competence might endow a robot with the ability to avoid obstacles. Another control layer can be implemented that allows the robot wander freely, however this layer does not have the ability to avoid obstacles. Instead, this is taken care of by layer 0. Together, layer 0 and layer 1 constitute level 1 competence (Brooks, 1986).

Higher-level layers can examine lower level data, and can send data to lower levels to suppress normal data flow. Low-level layers always run, however, without being aware of the layers above them (Leyden, 2000).

In behaviour based architectures, a number of behaviours run concurrently. Many of these will try to drive the same actuator at the same time. To overcome this situation, an arbitration function has to be implemented. The arbitration function has to select a single behavioural response from a multitude of possible ones (Leyden, 2000). There are a number of ways in which this can be done. In subsumption, a fixed priority arbitration scheme is used. A higher-level layer is able to subsume a lower level one (Figure 2.10). This is where the name subsumption comes from.

![Figure 2.10 The subsumption approach.](image)
Chapter 2: Literature Review

The task of controlling the robot is broken down into a number of different behaviours with no single behaviour responsible for the whole problem of achieving the overall goal. This greatly simplifies the design of the architecture, since behaviours can be built up incrementally. This architecture exhibits true dynamic reactive characteristics since behaviours are triggered directly by information perceived by the robot’s sensors into performing set actions. With meagre computing power robot behaviours can be implemented that endow the robot with sophisticated, almost insect-like qualities (Arkin, 1998).

Hybrid deliberative/reactive architectures

Hybrid deliberative/reactive robot architectures have emerged combining aspects of traditional AI symbolic methods and their use of abstract representational knowledge, adding considerable flexibility over purely reactive systems. Dynamically reconfiguring the control system based on deliberation (reasoning over world models) is an important addition to the overall competence of general-purpose robots (Arkin, 1995).

Many authors feel that hybrid systems capable of incorporating both deliberative reasoning and reactive execution are needed to deliver full potential of robotic systems (Arkin, 1995). The hybrid style combines both reactive and deliberative control in a heterogeneous architecture. It facilitates the design of efficient low-level control with a connection to high-level reasoning. The connection between the two levels can be tricky, however, and must be carefully designed and implemented to provide the right blend of reactivity and deliberation (Coste-Maniere and Simmons, 2000).

Rather than simply following contours, and avoiding obstacles randomly, hybrid systems permit the control system to perform behaviours in sequence. A typical behaviour sequence to navigate a submersible robot in a test tank might be: "submerge robot to fixed depth, move
forward following a fixed heading, stop robot on approach to pool wall, and return robot to surface." This kind of sequencing is difficult in a purely reactive system. In hybrid systems, behaviours operating in the reactive control layer can be activated and deactivated so as to accomplish more abstract goals.

Lyons in (Lyons and Hendriks, 1992) describes three different ways in which planning and reaction can be tied (Arkin, 1998):

- **Hierarchical integration of planning and reaction:** Deliberative planning and reactive execution are involved with different activities, time scales, and spatial scope. Hence a multilevel hierarchical system can be structured that integrates both activities (Figure 2.11.a). Planning or reacting depends on the situation at hand. In many ways, this is closely aligned with the traditional deliberative approach with one fundamental distinction: the higher, deliberative levels are epistemologically distinct from the lower, reactive ones, that is, the nature and type of knowledge and reasoning is distinct.

- **Planning to guide reaction:** Another alternative model involves permitting planning to configure and set parameters for the reactive control system. Execution occurs solely under the reactive system’s auspices, with planning accruing both prior to and concurrent with execution, in some cases projecting the outcome of continuously formulated plans and reconfiguring the reactive system as needed (Figure 2.11.b).

- **Coupled planning-reacting:** Planning and reacting are concurrent activities, each guiding the other (Figure 2.11.c).
In hybrid architectures the deliberative and reactive control paradigms are employed as different layers, these layers having different levels of abstraction.

Gat advocates that hybrid control architectures are invariably structured in three layers (Gat, 1998), consisting of a deliberative layer, control / sequencing layer and a reactive layer (Figure 2.12). The deliberative layer is responsible for performing time-consuming computational tasks such as planning, dealing with goal oriented aspects. The sequencer is responsible for controlling sequences of primitive actions. It initiates and terminates primitive activities by activating and deactivating sets of modules in the reactive layer. The reactive layer deals with real-time issues of the interaction with the environment (Gat, 1992).

![Figure 2.12 Hybrid control architecture.](image-url)
2.3.2 Control architectures for AUVs

Advances in sensing, control, communications, and computing technologies have enabled the development of autonomous vehicles that perform critical missions in harsh and unforgiving environments (Valavanis, Gracanin et al., 1997). As the complexity of missions increases, the demands on sensing, computing, communication, and control increase. The control architecture for an autonomous underwater vehicle should be able to perform seamless integration of a wide range of sensors, accurately gauge and monitor the status of the vehicle, perform the stated mission, and preserve itself and surroundings at all times. Furthermore, the nonlinear dynamic behaviour of the vehicle, arising from external disturbances due to ocean currents, and uncertain dynamics of the underwater environment add the complexity associated with the AUV control (Valavanis, Gracanin et al., 1997).

As a piece of software, an AUV’s control system is special in that, it must cope with data in many different grades of granularity (sensor readings versus user-supplied mission data) and yield purposeful action on different time scales (collision reflexes versus optimal long-term mission organization). A ‘control architecture’ is required to give an AUV control program structure, organization and to integrate these incoherent pieces into coherent overall action (Valavanis, Gracanin et al., 1997).

Surveys of control architectures for AUVs

The various descriptions and reviews available on autonomous mobile robot control architectures ((Arkin, 1998), (Leyden, 2000), (Coste-Maniere and Simmons, 2000), (Habib, 2002), (Carreras, 2003), (Toal, 2004)) and their implementations on autonomous underwater vehicles ((Healey, Pascoal et al., 1995), (Valavanis, Gracanin et al., 1997), (Ridao, Batlle et al., 1999), (Carreras, Batlle et al., 2000), (Ridao, Yuh et al., 2000)) demonstrate that the field
is an area of active research with important contributions and progresses, but where further challenges still need to be addressed. The following sub sections describe pertinent points and example control architecture approaches used on different AUVs.

**State-configured layered control (Bellingham, Beaton et al., 1989)**

Bellingham, Beaton *et al.* (1989) advocate that the most important requirement to deploy operational AUVs is to have a robust software development to maintain vehicle integrity while accomplishing mission goals (Bellingham, Beaton *et al.*, 1989). The Sea Squirt testbed vehicle provided a good platform to study the use of layered control architecture for autonomously planning underwater missions. Bellingham, Consi *et al.* (1990) claimed that the execution of real-world missions requires that competing behaviours be regulated by a top level finite state machine, which divides a mission into phases and permits activation only of those behaviours which are relevant to a given phase (Byrnes, MacPherson *et al.*, 1992).

The application software of the vehicle is divided into three major categories: sensor processing, software for performing closed loop control and software for planning the vehicle’s mission. Sensor processing functions are moved outside of the layered control architecture, as they present computationally intensive tasks. A separate controller is employed from the layered control architecture to handle the non-linear dynamics, closed at a frequency high enough to guarantee the stability of the vehicle (Bellingham, Consi *et al.*, 1990). The interface between the closed-loop controller and layered control architecture is designed such that the planner will pass commands in the form of setpoints. In addition to the division of the application specific software, the AUV software development has also required the following issues to be addressed (Bellingham, Consi *et al.*, 1990):
• The layered control architecture needs to be reconfigurable to allow the AUV to perform a variety of missions;

• Masking is used to avoid behavioural conflict, and to construct plans that simultaneously satisfy the objectives of much behaviour. Behaviours provide masks instead of simple commands, where the concept of the mask is associated with a set of acceptable commands and a set of unacceptable commands (Ridao, Batlle et al., 1999).

• Robust algorithms are needed to avoid trapping. Trapping usually occurs when control of the vehicle oscillates between two or more behaviours. Two trapping modes are observed in (Bellingham, Consi et al., 1990): behaviour conflict trapping and non-ideal sensor performance. Behaviour conflict trapping is defined as any circumstance where one behaviour is frustrated by another behaviour or combination of other behaviours, e.g. a vehicle attempting to proceed to a waypoint may be frustrated by the obstacle avoidance behaviour that cannot successfully bypass an array of obstacles. (Bellingham, Consi et al., 1990) Recovery from behaviour conflict traps is accomplished by reverting to recovery behaviours such as wall-following, or applying a random perturbation to the system. Non-ideal sensor performance (sensor system with large discontinuity, hysteretic behaviour or limited operational range) can potentially cause the vehicle to oscillate between behaviours in a way that impedes or prevents the vehicle from achieving its mission (Bellingham, Consi et al., 1990). Behaviours dependant on acoustic systems (inaccurate data generated by multipath) are considered especially susceptible to the latter type of trapping (Bellingham, Consi et al., 1990). Another example of non-ideal sensor performance
caused by the limited operational range experienced with the Tethra vehicle is described in section 6.2.

**An experimental comparison of hierarchical and subsumption architectures (Byrnes, MacPherson et al., 1992)**

An experimental comparison of hierarchical and subsumption software architectures used for the high-level mission control of AUV systems is carried out in simulation and described in (Byrnes, MacPherson et al., 1992). The experiment involved a traversal of a number of way-points, using the simple line-of-sight guidance. Simulation was not carried out in real-time due to the limitations of a graphic workstation, aiming to achieve a maximum frame rate in the graphic system. After a series of experiments, Byrnes et al. concluded that both the hierarchical and subsumption approaches as the strategic level of AUV control results in essentially identical behaviour of the system, i.e. the two approaches led to behaviourally equivalent controllers (Byrnes, MacPherson et al., 1992). Hence, the choice of the method for mission control can be determined in a pragmatic way, or as dictated by the experience of the programmer.

**A tri-level hybrid software architecture (Healey, Pascoal et al., 1995)**

In (Healey, Pascoal et al., 1995), the authors are in favour of some form of tri-level software architecture for complex missions. In a review on control architectures, the same three level structure is identified, with examples provided, as in classic approaches (organization, coordination and execution levels); in systems of reactive planning (hierarchic architecture consisting of a temporal planner, coordination level and functional level) and in hybrid approaches (strategic, tactical and execution level).

A hybrid system consisting of three-level software architecture (Strategic, Tactical and Execution levels) is described in (Healey, Marco et al., 1996). The three levels separate the
control requirements into easily modularized functions, enclosing logically intense discrete state transitioning using asynchronously generated signals for control of the mission and real time synchronised controllers that stabilize the vehicle motion to callable commands (Healey, Marco et al., 1996).

**Classification of AUV control architectures (Valavanis, Gracanin et al., 1997)**

Four AUV control architectures are identified in (Valavanis, Gracanin et al., 1997): the hierarchical architecture, the heterarchical architecture, the subsumption architecture, and the hybrid architecture. A survey of eleven AUV control architecture is provided with examples for the hierarchical and hybrid control architectures; examples for the subsumption architecture include the enhanced version, of state-configured layered control architecture (Bellingham, Consi et al., 1990); and as illustration for the heterarchical architecture a variant example of mixed hierarchical and heterarchical architecture is given.

(Valavanis, Gracanin et al., 1997) proposes two-level hybrid control architecture for implementation on AUVs. A state diagram residing at the supervisory control level determines the sequences of AUV tasks/operations throughout the various phases of the mission, providing a state-configured functionality for the overall system. The functional control bottom level is composed of functionally independent modules arranged in a heterarchical, parallel structure. The functional level modules directly control the vehicle’s actuators, sensors, and hardware component residing directly on the vehicle (Valavanis, Gracanin et al., 1997). A conceptual level design is also presented for embedded hardware control architecture.
Classification of AUV control architectures (Ridao, Batlle et al., 1999)

A comprehensive survey of twenty two control architectures can be found in (Ridao, Batlle et al., 1999). The three main methodologies applied for mobile robot control architectures consist of the deliberative, behavioural and hybrid architectures. In addition, a tool-kit control architecture category is also included, specifying the case when no apparent control architecture is present. The hybrid control architecture is currently the most popular, as it merges the advantages of the deliberative and behavioural architectures while minimising their limitations. Deliberative elements offer the hybrid system predictable functions and mission planning capabilities, while the behavioural elements exhibit timely reaction to sensor input using behaviour-based approaches in a changing and dynamic environment.

The review on the different control architectures concludes with a set of properties that an effective AUV control architecture should possess (Ridao, Batlle et al., 1999). A hybrid control architecture structured in three layers is proposed for implementation on the Garbi underwater robot. The higher-level deliberative layer is responsible for mission planning and re-planning on demand by lower-level layers or by an operator; the intermediate control execution level’s task is to activate the low-level behaviours and to pass parameters to them; while the functional reactive layer contains the physical sensor and actuator interfaces (Ridao, Batlle et al., 1999).

The same classification of the AUV control architectures into three categories of deliberative, reactive and hybrid architectures is described in (Ridao, Yuh et al., 2000). The authors reinforce the advantage of using the hybrid architecture for AUV control, stating that “many control architectures recently proposed converge to a similar structure that addresses the use of reusable and modularised software packages such as task modules and behaviours linked together for both predictability and reactivity”. A description of the ITOCA, (Intelligent Task
Oriented Control Architecture), hybrid control architecture developed for the SAUVIM semi-autonomous underwater vehicle is also provided in (Ridao, Yuh et al., 2000). The hybrid control architecture consists of three levels, i.e. the planning, control and execution layers.

**A comparative evaluation of four behaviour-based control architectures (Carreras, Batlle et al., 2000)**

A comparative evaluation of four behaviour based control architectures, i.e. Schema-based, Subsumption, Process Description Language and Action Selection Dynamics is described in (Carreras, Batlle et al., 2000). Behaviours implemented in these architectures include the ‘go to’, ‘obstacle avoidance’ and ‘avoid trapping’ behaviours. The emergent behaviour is determined by the specific coordination method of the behaviour based control architecture under experimentation. Competitive methods such as the subsumption and action selection dynamics have only one behaviour active at a time. In terms of robustness, tuning time and modularity, competitive methods exhibit better performances. The disadvantage of the competitive methods is that command fusion is not allowed, and in the case of a situation of goal-seeking and obstacle avoidance the generated trajectory is non-optimal (in terms of smoothness and trajectory length) (Carreras, Batlle et al., 2000).

**Specifications for robot control architectures (Coste-Maniere and Simmons, 2000)**

(Coste-Maniere and Simmons, 2000) describe the issues to be addressed by a robotic architecture as “… the need to achieve high-level complex goals, the need to interact with a complex, often dynamic environment, while ensuring the system’s own dynamics, the need to handle noise and uncertainty, and the need to be reactive to unexpected changes”. Although not referring to AUV control architecture, the authors classify most of the architectural styles into three categories: hierarchical, behavioural and hybrid, where the latter is indicated as the most recent style. The authors discuss how the development of
complex robotic systems can be facilitated using control architectures, focusing on the specification, execution and validation phases.

**The O³CA² hybrid control architecture (Ridao, Battle et al., 2002)**

The Object Oriented Control Architecture for Autonomy (O³CA²), hybrid control architecture designed for the Garbi underwater robot, is presented in (Ridao, Carreras et al., 2001), (Ridao, Battle et al., 2002). The hybrid control architecture consists of three layers: the deliberative/planning, control execution and functional reactive layer. The deliberative layer described in (Ridao, Battle et al., 2002) is responsible for mission planning, when a new task is specified by the user or if a task requires re-planning in case it has failed. The control execution layer, relying on a Finite State Machine Design, makes the actual state evolve from the begin state to the end state through a sequence of states (Ridao, Battle et al., 2002).

The reactive layer, thoroughly described throughout the paper, is in charge of the real-time control of the vehicle, and provides three reactive mechanisms: behaviours, monitors and timers. The reactive layer’s task is also to interface sensor sub-systems through sensor fusion, obtaining position estimates through a dead-reckoning navigation system; and to provide the interface to the low-level controllers by means of set-points. Four behaviours using fuzzy logic systems, the ‘Go to’, ‘Spin’, ‘Avoid’ and the ‘Keep depth & Avoid bottom’ are described. The outputs of the behaviours are the vehicle’s desired angular, surge and heave speed values.

Monitors are used for detecting situations in order to inform the control execution to adapt the enabled behaviours to the situation. Examples of monitors include the ‘Achieved position’, and enable and disable monitors of the individual behaviours. Timers are used for
generating time-based events, and they are mainly used to determine task deadlines (Ridao, Battle et al., 2002).

**Behaviour-based hybrid coordination method & Reinforcement learning (Carreras, 2003)**

The control architecture proposed in (Carreras, 2003) is a hybrid control architecture, from the perspective of the behaviour-based layer consisting of a set of behaviours and the coordinator. A hybrid coordination methodology is proposed (Carreras, Batlle et al., 2001), between competition and cooperation, trying to benefit from the advantages of both. Competitive coordination in navigation tasks confers robustness to the systems with the drawback of non-optimal trajectory performance. Cooperative methods offer the advantage of optimal, i.e. smooth, short and safe, trajectory performance. A behaviour response consists of a vector of normalized robot control action, i.e. robot velocity for a particular DOF, \( v_i = (v_{i,x}, v_{i,z}, v_{i,\text{yaw}}) \), where \( |v_i| = [0 \ 1] \), and as well has an associated activation level \( a_i = [0 \ 1] \). The behaviour-based layer acts over the low-level controller, generating the actions to be followed, therefore advocating for a similar control system as described in (Bellingham, Consi et al., 1990).

The concept of Hierarchical Hybrid Coordination Nodes (HHCN) is introduced, to accomplish a hierarchical layering and complete coordination process for the behaviours (Figure 2.13).

![Figure 2.13 Hierarchical Hybrid Coordination Node (Carreras, 2003).](image)
The coordination method imposed by the HHCN changes, depending on the activation level of the behaviours and the hierarchy between them. When a dominant behaviour \( b_d \) is completely activated, \( a_d = 1 \), the response of the node will be equal to the dominant behaviour, therefore the HHCN behaves competitively. If the dominant behaviour is partially activated, \( 0 < a_d < 1 \), the dominant \( b_d \) and non-dominant behaviour \( b_{nd} \) responses will be combined in a cooperative manner (Carreras, 2003). The activation level of the HHCN determined response, is the sum of the activation levels of the input responses, where \( a_{nd} \) is multiplied by a reduction factor. The reduction factor depends on the activation of the dominant behaviour and on the value of the integer parameter \( k \) (Carreras, 2003).

A Neural Q-learning approach is used to learn the mapping between the state and action spaces; where the state space is the sensor information perceived by the robot, needed by a behaviour to accomplish its goals, and the action space is the velocity set-points the robot should follow (Carreras, Ridao et al., 2002). The feasibility of the SONQL in a real-time task has been demonstrated, the algorithm being able to learn state/action mapping of one DOF for a reduced number of iterations (Carreras, 2003). The feasibility of the SONQL algorithm for AUV high-level control is demonstrated, by performing tests on the ODIN using two behaviours (tracking and recovery) and also carrying out tests on the URIS utilising four behaviours (tracking, repulsion, recovery and teleoperation) (Carreras, Yuh et al., 2005).

Enhanced subsumption architecture: Command fusion & Prioritisation of the safety-critical (emergency fail-safe) behaviours (Toal, 2004)

In (Toal, 2004) it is stated that, in terms of coordination methods used for behaviour output, pragmatism can point to a mix of competitive and co-operative approaches between the behaviours of one robot. In some instances cooperation of behaviours is a preferred solution,
e.g. using fuzzy logic to combine behaviours such as “Obstacle avoidance” and “Goal seeking” in the context of navigation. Using command fusion, the fuzzy logic based navigation system implemented on a land-based mobile robot, has offered considerable advantages over the standard subsumption architecture (Leyden, Toal et al., 1999), (Flanagan, Toal et al., 2003).

A significant finding made in (Toal, 2004) from the perspective of control architectures employed on AUVs, points to the use of a hybrid control architecture (combining aspects of the deliberative and behaviour-based approaches), which provides set-points for the underlying embedded closed-loop controllers (Bellingham, Consi et al., 1990).

Toal proposes a different ordering priority of safety critical behaviours, as opposed to the standard subsumption idea. To overcome the shortcomings of the subsumption architecture, i.e. obstacle avoidance behaviour residing at bottom level, an alternative approach is proposed to reverse the priority ordering between layers, such that the obstacle avoidance, or more importantly, safety critical behaviour such as collision avoidance are always active, residing at the top level of the control architecture, ready to wrest control of the robot from the other behaviours (Toal, 2004).

Toal advocates that with highest priority, safety critical behaviours in place, the control approach taken to organise other behaviours, mission level planning and control, is a matter for pragmatic engineering design and does not require dogmatic stricture.

2.3.3 Concluding summary remarks on survey of control architectures for AUVs

The majority of the control architecture surveys crystallize the relevance of a hybrid layered control structure.
In (Bellingham, Consi et al., 1990) a separate vehicle controller is employed that does not reside within the layered control architecture, as the latter has a different spatial and temporal scope. Sensor processing functions are removed from the individual behaviours and placed outside of the layered control architecture, as the sensor processing requirements (e.g. dead-reckoning, inertial) are much greater and necessitate more CPU resources than does the mission planner (Bellingham, Consi et al., 1990).

The interface between the high-level (mission planner, control execution and reactive layer) and the low-level control is usually designed by means of behaviours passing set-points to the embedded closed-loop controllers (Bellingham, Consi et al., 1990), (Carreras, 2003), (Toal, 2004).

The hybrid control architecture is especially attractive as it merges the behaviour-based architecture’s timely reactive responses to sensor inputs, while it also accommodates mission level planning capabilities and symbolic state description/representation (Ridao, Batlle et al., 1999), (Toal, 2004).

In the three-level hybrid structure, the control execution layer is responsible for sequencing the execution of the behaviours. Although sometimes its functionality is split between the deliberative and reactive layers, an attractive and convenient design could be achieved using finite state machines/state diagrams (Valavanis, Gracanin et al., 1997), (Ridao, Battle et al., 2002).

Safety critical, emergency fail-safe behaviours should reside at the top level of the behaviour-based structure, ready to take control over the vehicles under safety critical conditions as determined by sensors (e.g. electronics enclosure leak). Under normal conditions, while the
safety critical behaviours reside at the top of the hierarchy, they are inactive due to no sensed emergency (Toal, 2004).

Yet, despite all these advances, Whitcomb states that there is no industry standard, only in-house solutions exists and the field of control architectures remains an area of active research (Whitcomb, 2000).

2.4 Way-point guidance systems

A navigation system provides information about a vehicle, determining its position, course and distance travelled. The guidance module processes the navigation data of the vehicle and computes the reference (desired) position, velocity and acceleration to be used by the control system. The control system is responsible for determining the necessary control forces and moments to be provided by the vessel in order to satisfy the proposed control objective (Fossen, 2002). A report on various guidance laws and their relevance to autonomous underwater vehicle is discussed in (Naeem, Sutton *et al.*, 2003). Some of the guidance law categories that are presented as applicable for AUV guidance include: vision based guidance (Wettergreen, Gaskett *et al.*, 1999), (Dunbabin, Corke *et al.*, 2004), (Antich, Ortiz *et al.*, 2004); missile control variant proportional navigation guidance (Naeem, Sutton *et al.*, 2004); electromagnetic guidance (Feezor, Yates Sorrell *et al.*, 2001); and the way-point guidance by the line-of-sight (LOS).

2.4.1 Conceptual elements

Way-point guidance systems are commonly employed for aircraft, land-based mobile robots, ships and underwater vehicles. Way-point guidance systems consist of a way-point generator with human interface (Fossen, 2002). For AUVs, way-points are generated prior to mission.
Chapter 2: Literature Review

However, aiding (virtual) way-points can be generated to provide trajectory replanning and/or reconfiguration in real-time.

**Trajectory tracking and path following control**

Trajectory tracking requires a vehicle to track a time-parameterized reference curve. Path following control aims at forcing a vehicle to converge to and follow a desired spatial path, without any temporal considerations (Lapierre, Soetanto et al., 2003). The path following objective can occur when it is required that an AUV examine an area by performing a “lawn mowing” manoeuvre along desired tracks with great accuracy, at speeds determined by an end-user (Lapierre, Soetanto et al., 2003).

Breivik and Fossen (2005) describe the trajectory tracking problem as involving the simultaneous construction of a geometric path and the dynamic behaviour of a path particle (Breivik and Fossen, 2005b). Space and time assignments are mixed into one single assignment, demanding that any actual target system is located at a specific point in space at a specific, pre-assigned instant in time.

In manoeuvring (path following), the spatial localization is of primary importance; and it must ensure that the actual system converges to and follows a desired geometric path, without any temporal requirements. The following task objectives are defined for the manoeuvring problem (Breivik and Fossen, 2005b), (Fossen, 2002):

**Geometric task:** Make the position of the actual system converge to and follow desired geometric path.

**Dynamic task:** Make the speed of the actual system converge to and track a desired speed assignment.
Way-point representation

The route of a surface vessel or an AUV is usually defined in terms of way-points.

A way-point database usually stores information about the following (Fossen, 2002):

- Cartesian coordinates of $WP_k$ way-point $(x_k, y_k, z_k)$, or $(x_k, y_k)$ for surface vessels and AUV planar (constant depth) way-point guidance cases;
- Line-of-sight heading towards the next way-point;
- Radius of acceptance sphere (circle), $R_k$;
- Speed information, etc.

Line-of-sight guidance

Line-of-sight guidance is an attractive method for heading control, where the line-of-sight vector is calculated from the vessel to the next way-point, or to a point on the path (Fossen, 2002). In case of a surface vessel or planar way-point guidance for a submersible vehicle, the line-of-sight vector can be taken as the vector from the body fixed origin $(x, y)$ to the next way-point $(x_k, y_k)$ (Fossen, 2002), where the vessel position measurement is provided by the navigation system, and the way-point Cartesian coordinates are available from the way-point database. The line-of-sight heading is calculated from

$$\psi_{LOS} = \text{atan2}(y_k - y(t), x_k - x(t))$$

(2.30)

where the \text{atan2} functions provide a four-quadrant heading solution such that $\psi_{LOS} \in [-\pi, \pi]$.

An alternative approach to avoid large non-compensated cross-track errors is to modify the LOS vector such that it is taken from the vessel coordinate centre point to the intersecting
Chapter 2: Literature Review

point \( p_{\text{LOS}}(x_{\text{LOS}}, y_{\text{LOS}}) \) on the path (Figure 2.14), at a distance \( n \) ship lengths \( L_{pp} \) ahead of the vessel (Fossen, Breivik \textit{et al.}, 2003).

![Figure 2.14 Definition of modified LOS vector.](image)

In this case the desired heading angle is determined as:

\[
\psi_{\text{LOS}} = \tan^{-1}(y_{\text{LOS}} - y(t), x_{\text{LOS}} - x(t))
\]

(2.31)

where the LOS coordinates \((x_{\text{LOS}}, y_{\text{LOS}})\) are given by:

\[
\left(\frac{y_{\text{LOS}} - y(t)}{x_{\text{LOS}} - x(t)}\right)^2 = \frac{nL_{pp}^2}{k_{1}^2 - k_{11}^2}
\]

(2.32)

Equation (2.32) refers to the norm of the LOS vector, and also states that the slope of the path between the way-points \((x_{k-1}, y_{k-1})\) and \((x_{k}, y_{k})\) is constant (Fossen, Breivik \textit{et al.}, 2003).

A way-point is considered to be reached, if the vehicle lies within the acceptance circle with radius \( R_{k} \). This condition can be written as:

\[
(x_k - x(t))^2 + (y_k - y(t))^2 \leq R_k^2
\]

(2.33)
Way-point switching should occur at this stage, incrementing the way-point index and extracting guidance data related to the next way-point from the database. In order to avoid the abrupt change of the desired heading value at the switching process, it is a common technique to pass the heading value through a reference model, i.e. first order low-pass filter, in order to smooth out the discontinuous switching (Fossen, Breivik et al., 2003). Due to the robustness of the low level controllers and protection built inside the thrusters control electronics, such as slew-rate limiters etc., discontinuous switching does not cause negative implications on the stability of the vehicle.

2.4.2 Review of way-point guidance methods for aircraft, mobile robots & marine surface vessels

Way-point guidance laws for aircraft

In (Rysdyk, 2003) trajectories are proposed and a guidance law is designed, resulting in constant line-of-sight orientation relative to an aircraft. The proposed guidance law is implemented based on the ‘good helmsman’ behaviour. Using the Serret-Frenet frame for a 2D path, the vector of vehicle velocity is determined relative to the desired path in the horizontal plane, where the relative position is measured from the vehicle to the closest point on the desired path (Rysdyk, 2003).

A horizontal way-point guidance algorithm employing a linear quadratic regulator for the line-following guidance is described in (Whang and Hwang, 2002). Way-point guidance is achieved by applying sequentially line-following guidance to each line segment. Optimal way-point changing points minimizing the accelerations required for way-point changing are derived in (Whang and Hwang, 2002).
Way-point guidance laws & path planning for land-based mobile robots

A way-point navigation method for land-based application is described in (Tomitaka, Kobayashi et al., 2003). The way-point navigation method is performed on a prototype intelligent wheel chair, to autonomously navigate an outdoor crowded or congested street without collision. The path for the way-points is changing with the state of the moving obstacles (Tomitaka, Kobayashi et al., 2003). Moving obstacles are detected and a dynamic path planning algorithm is applied for collision avoidance. The dynamic path planning is classified as either long term, i.e. path planning to reach the goal point, or short term path planning, to reach each way-point (Tomitaka, Kobayashi et al., 2003). The latter is interpolated using C-spline interpolation between way-points. To avoid obstacles, a path tracking algorithm with virtual way-points is developed (Tomitaka, Kobayashi et al., 2003).

An implementation of a spline curve algorithm for mobile robot path-planning is described in (Eren, Fung et al., 1999). The spline method is also considered for the calculation of a suitable path between two points in space by taking into consideration the obstructions on the path as well as the tasks to be achieved.

Hwang, Arkin et al. (2003) describe a control algorithm for operation of a mobile robot in a supervisory manner. Path points are generated using operator input for trajectory generation (Hwang, Arkin et al., 2003). An on-line piecewise cubic Bezier curve trajectory generation algorithm is developed to create a smooth trajectory between the way-points created with human input.

Way-point guidance laws for marine surface vessels

A fuzzy guidance controller used for an autonomous boat for way-point tracking is described in (Vaneck, 1997). The fuzzy controller for way-point tracking is recognized as relatively
easy to develop (not requiring a complex mathematical model of the vehicle’s dynamics),
simple to tune and robust to external disturbances (Vaneck, 1997).

The proposed guidance problem is to navigate to and cross on a specified heading, through a
series of way-points. A simple dead-reckoning algorithm is used to estimate position of a
vehicle between GPS position updates. Position and heading are the crisp inputs into the
fuzzy controller, while the output of the fuzzy controller is the boat’s commanded rudder
angle. Way-points are defined by their geographic coordinates, crossing heading and arrival
radius. The “model free” nature of the fuzzy controller is found attractive as it allows rapid
design and implementation of the control laws without initially having to develop nonlinear
dynamic models or complex control system architectures (Vaneck, 1997).

Pettersen and Lefeber (2001) describe the way-point tracking control of a ship, moving at
constant surge velocity. The control aim described in the paper is to make the ship follow a
straight line between two way-points using yaw torque control. One approach mentioned in
(Pettersen and Lefeber, 2001) to achieve this objective, is to use a conventional autopilot,
controlling the heading using yaw torque control, with a combined line-of-sight algorithm. A
goal is to minimize the cross-track error (the shortest distance between the ship and the
straight line), thus to control the sway position to zero. The paper determines a method to
control both the heading and the sway position using the yaw control. A feedback control is
derived that controls the cross-track error, with both the sway position and the heading to
zero, such that the ship heading is tangential to the straight line trajectory (Pettersen and
Lefeber, 2001). In order to steer the ship towards the straight line, according to the good
helmsman behaviour, the ship course angle is used (given that the ship has a positive forward
velocity), rather than to use the ship sway velocity (Pettersen and Lefeber, 2001).
Chapter 2: Literature Review

A nonlinear fuzzy autopilot for ship track-keeping is presented in (Omerdic, Roberts et al., 2003). The autopilot has four inputs (actual and desired heading, rate of change of heading and offset from the desired path) and one output (command rudder angle). The track-keeping problem is decomposed into two subtasks: follow the desired heading, and bring the ship onto the desired path and keep tracking. The autopilot consists of two autopilots, each of these having a simple structure with a design based on the reasoning and behaviour of an experienced helmsman.

A ship manoeuvring design for an underactuated marine craft, including a LOS guidance system and a nonlinear feedback tracking controller is described in (Fossen, Breivik et al., 2003). The output space is reduced from 3 DOF \((x_d, y_d, \psi_d)\), to 2 DOF, i.e. \(\psi_d\) and \(u_d\), using a LOS projection algorithm. The tracking task \(\psi(t) \rightarrow \psi_d(t)\) is achieved using only one control (using the rudder), while tracking of the speed assignment \(u_d\) is performed by the remaining control (the main propeller) (Fossen, Breivik et al., 2003). Control laws in surge and yaw are derived using backstepping, resulting in a dynamic feedback controller where the dynamics of the uncontrolled sway mode enters the yaw control law (Fossen, Breivik et al., 2003).

A guidance based path following approach is proposed as an alternative to the traditional trajectory tracking problem. A nonlinear model-based controller is designed for a fully actuated vessel, to comply with the guidance commands. The guidance based path following is illustrated for the task of manoeuvring marine surface vessels along a given path (Breivik and Fossen, 2004).
2.4.3 Way-point guidance laws for UUVs

Waypoint guidance by line-of-sight is described in (Healey and Lienard, 1993). Multivariable sliding mode autopilots are employed for the combined control of AUV steering, depth and speed during complex flight manoeuvres. Healey states that well behaved autopilot systems enable the use of various guidance schemes to achieve way-point and path tracking. In the presence of ocean currents for slow-speed guidance it was noted that some of the way-points may not be achievable and also that the vehicle’s cross-track error has increased (Healey and Lienard, 1993). To achieve the ability to compensate for the effects of ocean currents an approach is proposed to include a feedforward term in the steering controller based on the current magnitude and direction estimate.

A control strategy for an underactuated underwater vehicle for tracking of a space trajectory is presented in (Alonge, D'Ippolito et al., 2001). The class of UUV considered in the paper employs main thrusters and two stern planes. For the particular UUV, a classical approach would employ three autopilots, i.e. combined pitch-depth, steering and forward velocity autopilot. This approach can lead to poor performances when executing complex manoeuvres, which emphasizes the nonlinearity and the multivariable characteristics of the system (Alonge, D'Ippolito et al., 2001). Instead of the conventional approach, the paper treats in detail the design of a nonlinear multivariable dynamic model. The kinematic control law is derived in order to force the system to track the reference trajectory. Reference values are given to those velocities, which can be controlled by means of available control variables. A dynamic control law is derived which forces the system to track the above velocity reference values (Alonge, D'Ippolito et al., 2001).
The requirements of a simple path following controllers are summarised in (Encarnacao and Pascoal, 2000) and (Lapierre, Soetanto et al., 2003), including calculation of the distance between the vehicle’s centre of mass and the closest point on the path, the angle between the vehicle’s total velocity vector and the tangent to the path at the shortest point, with the overall aim of reducing both to zero. These requirements motivated a development of a kinematic model of the vehicle in terms of the path’s frame, that is, a frame that moves along the path. Controller design is realised in two steps: a kinematic controller is developed in the first step, using the yaw rate as a virtual control input; the dynamic controller is derived from the kinematic controller using backstepping techniques in the second step (Encarnacao and Pascoal, 2000), (Lapierre, Soetanto et al., 2003).

Encarnacao and Pascoal (2000) addresses the issues of control of marine vehicles in three-dimensional space. A hypothetical vehicle is considered that has its main body axis aligned with the total velocity vector and a nonlinear kinematic controller is derived for that vehicle to steer it to a 3D reference path. With the total velocity vector aligned to the tangent to the path, possible errors in roll do not affect path following and should not be compensated. The goal of the kinematic controller is to drive the distances to the path, i.e. cross-track error and depth error, as well as the angular displacements, to zero. The kinematic controller was designed assuming that the pitch and yaw angular velocities were the actual control signals and the ocean current disturbance is known. The observer is designed to estimate current velocity components. Finally, the kinematic controller is extended to the dynamic case to deal with model parameter uncertainty (Aguiar and Pascoal, 2002).

The problem of dynamic positioning and way-point tracking of underactuated AUVs in presence of constant unknown ocean currents and parametric model uncertainty is addressed
in (Aguiar and Pascoal, 2002). A nonlinear adaptive control law is derived for the surge force and yaw torque, such that the vehicle’s centre of mass converges in the presence of a constant unknown ocean current to the last way-point, after visiting in ordered sequence all the previous way-points.

Lapierre, Soetanto et al. (2003) proposed a new methodology for the design of path following systems for AUVs. The key idea of the control law development described in the paper is to control the rate of progression of a virtual target to be tracked along the path (Lapierre, Soetanto et al., 2003). This approach overcomes a singularity problem, usually occurring when the position of the virtual target is defined by the projection of the vehicle on the path. A singularity can occur in this case, when the vehicle is located at the centre of curvature of a circular path.

Implementation of a way-point tracking control in horizontal plain using a line-of-sight guidance scheme for a biomimetic autonomous underwater vehicle is described in (Guo, Tsai et al., 2003). To attempt the performance of fish-like precise hovering and agile turning, the vehicle testbed is realised to fully resemble the shape and the motion modes of a fish. To achieve way-point tracking capability the forward-swimming mode and the turning mode of the biomimetic AUV is controlled for motion control (Guo, Tsai et al., 2003).

The trajectory tracking problem, demands that a vehicle should be located at a specific point in space at a pre-assigned instant in time, representing a feedforward, open-loop type of solution at the vehicle-path interaction level. The path following emphasizes spatial localization as a primary task objective, considering the dynamic aspect sacrifiable, thus representing a more flexible and robust alternative than the trajectory tracking problem (Breivik and Fossen, 2005a). The guidance based path following scheme proposed in
(Breivik and Fossen, 2005a) as a theoretical framework, is successfully employed for the purpose of manoeuvring AUVs along 3D designated paths. Two different concepts are illustrated for the problem of an actual system to converge to and to follow along a desired geometric path. In such a scenario, the actual particle is defined as a position variable belonging to the actual system, whose goal is to converge to the geometric path, while the path particle is a position variable belonging to the desired geometric path, and is restricted to move along it at all times. The “Trajectory Tracking” problem requires the simultaneous construction of the geometric path and the dynamic behaviour of the path particle. The “Path Following” strategy involves the separate construction of the geometric path, with spatial localization considered as a primary task objective, while the dynamic aspect of the path can be neglected.

A nonlinear, model-based velocity and attitude controller is designed by using the backstepping technique (Breivik and Fossen, 2005a). Instead of controlling position and attitude it is chosen to control linear velocity and attitude, resembling the action of a helmsman who uses velocity to manoeuvre, as such a technique is equally effective for fully actuated or underactuated vehicles (Breivik and Fossen, 2005a). Simulation results presented in the paper demonstrate the capability of the approach when applied to a monohull-shaped underactuated AUV model.

Fuel-optimal guided navigation is presented in (Kim and Ura, 2003). A switched finite-time point-to-point control strategy for underactuated AUVs is described in (Sankaranarayanan, Mahindrakar et al., 2003).
Chapter 2: Literature Review

**Smooth way-point transition (Bakaric, Vukic et al., 2004)**

Improvements to the basic algorithm of the way-point guidance by the line-of-sight are presented in (Bakaric, Vukic et al., 2004). Way-point guidance by the line-of-sight is acclaimed to be simple, computationally inexpensive and flexible to sudden changes in the desired path specification. Nevertheless, it can achieve non-optimal vehicle trajectory, ignore possible side disturbances such as sea currents, and can lead to deadlock situations (Bakaric, Vukic et al., 2004). The authors described methods to compensate for several weak points of the algorithm. A missed way-point detection method is described along with an approach for sea current disturbance rejection. The proposed method is described in detail in the following, since it is implemented and tested in the Tethra simulation environment (see Chapter 6:). A variant of the method is also implemented for path following in “lawn mower” fashion, see section 5.6.3.

This way-point guidance scheme takes into account the position of the next way-point \( WP_{k+1} \) before the current way-point \( WP_k \) is reached, and corrections are applied on the reference heading generated by the line-of-sight heading (see Figure 2.15). The reference heading \( \psi_r \) is calculated as the sum of the line-of-sight heading \( \psi_{LOS} \) and the correction term \( \psi_c \):

\[
\psi_r = \psi_{LOS} + \psi_c
\]  

(2.34)

**Figure 2.15** Illustration of the smooth way-point transition.
An auxiliary point \( A_k(x_{Ak}, y_{Ak}) \) is defined as a point that lies on segment \( WP_k WP_{k+1} \) such that \( d(WP_k, A_k) = R_k \). The coordinates of the point \( A_k \) are given by:

\[
y_k \neq y_{k+1} \Rightarrow \begin{cases} 
  y_{Ak} = y_k + R_k \frac{\text{sgn}(y_{k+1} - y_k)}{\sqrt{1 + \frac{(x_{k+1} - x_k)(y_{k+1} - y_k)}{y_{k+1} - y_k}}} \\
  x_{Ak} = x_k + \frac{x_{k+1} - x_k}{y_{k+1} - y_k}(y_{Ak} - y_k)
\end{cases}
\]

(2.35)

\[
x_k \neq x_{k+1} \Rightarrow \begin{cases} 
  x_{Ak} = x_k + R_k \frac{\text{sgn}(x_{k+1} - x_k)}{\sqrt{1 + \frac{(x_{k+1} - x_k)(y_{k+1} - y_k)}{x_{k+1} - x_k}}} \\
  y_{Ak} = y_k + \frac{y_{k+1} - y_k}{x_{k+1} - x_k}(x_{Ak} - x_k)
\end{cases}
\]

(2.36)

If the current position of the vehicle is denoted by \( S(x, y) \), distances \( d_{Ak} = d(S, A_k) \) and \( d_{WPK} = d(S, WP_i) \) can be found from (Bakarić, Vukic et al., 2004):

\[
d_{WPK}^2 = (x_{WPK} - x)^2 + (y_{WPK} - y)^2
\]

(2.37)

**Figure 2.16** Definition of the auxiliary point \( A_k \) and normalised distance difference \( \varepsilon_k \).

In order to simplify analysis of all different possibilities, it is convenient to introduce normalised distance difference \( \varepsilon_k \) (see Figure 2.16):
Scalar $\varepsilon_k$ serves as a good indicator of the turn needed at the current way-point $WP_k$ in order to obtain smooth transition: if the next way-point $WP_{k+1}$ lies in the direction of the current waypoint, $\varepsilon_k$ is close to 1, while $\varepsilon_k$ near -1 means that a U-turn needs to be performed.

The correction term $\psi_c$ can be constructed from two magnitude factors $f_d$ and $f_A$ (see Figure 2.17), according to

$$\psi_c = \text{sgn}(\varepsilon_k)f_d(d_{WP_k})f_A(\varepsilon_k) \quad (2.39)$$

![Figure 2.17](image)

Figure 2.17 Magnitude factors for the desired heading correction: the distance factor (left); the turn factor (right).

The shapes of $f_d$ and $f_A$ are found by simulation. The sign of the correction term is determined from

$$\text{sgn}(\psi_c) = \text{sgn}[(x_{Ak} - x)(y_k - y) - (x_k - x)(y_{Ak} - y)] \quad (2.40)$$

Smooth way-point transition at turns is achieved by means of reference heading corrections. The authors conclude that acceptable quality of guidance through way-points is achieved by intuitive additions to the simple line-of-sight guidance algorithm.
2.5 Fuzzy logic control for autonomous navigation

In general the objective to address the motion planning and control problem in the field of robotics, is to find collision-free trajectories for a robot, in static or dynamic environments containing some obstacles, between a start and goal configuration (Zavlangas, Tzafestas et al., 2000).

Fuzzy logic has been widely employed for autonomous navigation (Tsourveloudis, Doitsidis et al., 2005) of aerial, land based robots (Toal, 2004), (Leyden, 2000) autonomous boats (Vaneck, 1997) or unmanned underwater vehicles (Kato, 1995), (Kanakakis, Valavanis et al., 2004).

The broad applicability of fuzzy logic in autonomous navigation is established on the suitable knowledge representation of inherently vague notions achieved through fuzzy IF-THEN rules (Tsourveloudis, Doitsidis et al., 2005). Furthermore, in the majority of fuzzy logic applications in navigation, a mathematical model of the dynamics of the vehicle is not required in the design process (Vaneck, 1997), only the problem-specific heuristic control knowledge is needed for the inference engine design (Tsourveloudis, Doitsidis et al., 2005).

2.6 Concluding remarks

A comprehensive literature survey has been undertaken in this chapter. Research fields covered in this survey include the following topics: propulsion systems & control allocation, control architectures for mobile robots & AUVs, way-point guidance systems and fuzzy logic control for autonomous navigation.

A short review of propeller models has been presented, including one-state, two-state and three-state models. A quasi-steady model of thrust and torque has been described and used to
develop bilinear & affine thruster models in vector form. A dynamic model of DC-motor has also been described, including nonlinear elements, such as friction and propeller load in the water. Material presented in this section was used to build simulation model of the propulsion system as part of Tethra simulator. The simulation loop of propulsion system is executed much faster than the other loops and it includes many non-linear effects present in real applications, such as slew rate limiter, viscous and Coulomb friction, non-symmetrical propeller T-curve, etc.

Significant efforts have been undertaken in the research community over the last two decades to solve the control allocation problem for modern aircraft. Different methods were proposed such as direct control allocation, optimisation based methods using $l_1$ norm and $l_2$ norm, the fixed-point method and daisy chain control allocation. Bar some differences, control allocation problems for modern aircraft and UUV have many similarities. For this reason, the main idea behind above mentioned methods is described in this chapter. Since the control allocation approach is in the core of the proposed control system, it is expanded and adapted to underwater applications in chapter 4, including the full description and precise mathematical formulation of the underlying concept.

Following this a review of control architectures for mobile robots and AUVs was presented. Based on this review and best practice for control architecture for AUVs, the approach taken for Tethra’s control architecture is to implement hybrid control architecture. As espoused by (Toal, 2004), the safety critical control behaviours are given the highest priority such that they can always abort a mission if an emergency situation arises. A detailed description of the safety critical behaviours implemented on Tethra are described in (Love, 2004) and
(Toal, 2004). They include a leak detect behaviour, a never exceed pressure depth behaviour and battery critically low behaviour.

With safety critical behaviours in place with highest priority, the development of other control functionality can progress without detailed concern for that safety criticality. Chapters 4 and 5 describe how the hybrid control architecture is implemented on Tethra, both on the physical and virtual vehicle. The implemented three-level architecture with real-time control taking set points from behaviours employs several novel features, including:

- State machine implementation of mission and tracking way-point planners,
- Flexible and transparent fusion of control clusters into winner cluster, who has an exclusive right to control the actuators,
- Complete and integrated architecture, which deals with all levels of control from mission control (for example survey planning and control) through behaviour implementation down to a novel implementation of low-level vehicle control employing control allocation separated from activation.

These aspects of Tethra’s control are described in detail in chapters 4 and 5.
Chapter 3: Tethra AUV Design Concepts

The name Tethra - An obscure and ancient Irish verb, noun and adjective. It produced a verb meaning 'to flood', 'to burst forth', giving just some examples. It has occurred as a proper name in the mythological cycle in reference to a 'Fomorian' king, aka. Muredhach. As an adjective, a rough sea would be intended. It also had the attributes of a War Goddess! Quite a name for our test vehicle until tamed through control and AI.

3.1 Introduction

The author has played a significant role in the development of Tethra, primarily focusing on embedded controller development and integration of sensors. Tethra is however the product of the combined effort of researchers in the Mobile & Marine Robotics Research Centre at the University of Limerick.

Tethra is a prototype AUV, designed at the University of Limerick to carry out research in underwater robotics and control. An open-frame design has been employed facilitating convenient placement and easy reconfiguration of actuators, control sensors and payload sensors. The vehicle was developed as a test bed for AUV/ROV controller experimentation for operation scenarios requiring low speed and multi thruster capabilities. Current research activities in the Mobile & Marine Robotics Research Centre at UL are focused on design and development of a highly maneuverable and fully autonomous underwater vehicle to address the challenges for high-resolution seabed survey in deep water (strong sea currents etc.) and inshore water (close proximity of hazards, such as wrecks, submerged buoys, etc.).

The Tethra prototype vehicle can be operated in dual (ROV/AUV) modes for system testing and proving. It can also be operated by Radio Control without a tether during launch and
retrieval while on the surface. This radio control enables the vehicle to be driven away from or back to quay/pool side under pilot control and also enables trimming of ballast prior to switch over to auto mode for a dive mission.

The Tethra robot platform provides a means to evaluate different combinations of sensors, actuator, and controller configurations and work is ongoing on the integration of new sensors, real-time target embedded computing platforms and the development of control algorithms to extend the robot’s current capabilities. For the Tethra AUV, the open frame multiple thrusters design has been adopted to meet manoeuvrability requirements for target applications such as transect tracking during high resolution marine surveys, underwater filming following benthic contours or bottom following for video survey close to the seabed.

Tethra early design and development was on a low budget. As such vehicle design choices, in terms of thrusters and original sensors employed on the craft, were made based on tight cost constraints. At a mid point through Tethra development, a suite of high precision navigation instruments was acquired in the research group through another project. These instruments include an IxSea PHINS Inertial Navigation System; an RDI Doppler Velocity Log (DVL); a combined MicroBath altimeter & depth sensor (high-precision DigiQuartz pressure sensor), a Trimble differential GPS system and six Tritech digital precision altimeters for obstacle avoidance. A high-resolution multi-beam sonar SeaBat 7125 (payload sensor) was also acquired. Since the time of acquiring these precision instruments, much of Tethra control development has been based on the use of these instruments in place of the low budget sensors used during the early design and development.

This chapter details the design of the Tethra vehicle including mechanical configuration, thrusters, power electronics, embedded controllers, low budget sensor suite integration &
testing, high precision instrument integration, control/experiment development using Control Centre. Note worthy is the fact that early development cycles proved long and demanding. The cycle of design integration, coding, mobilizing, pool testing, data logging and analysis feeding back to design proved to be a very long process which significantly impeded the rate of development. A virtual instance (simulation model) of the Tethra vehicle and the test pool, was developed in the research group to replicate the physical vehicle and test pool to address this problem (Figure 3.1). This approach allows much of the development to be tested and proved in simulation before porting to the physical vehicle. A significant aspect of the work of this thesis therefore involves testing in simulation and the details of these experiments are presented in chapter 6.

![Figure 3.1 Virtual reality model of Tethra (left); Real Tethra on the edge of the test pool (right).](image)

The virtual development was undertaken to accurately reflect the physical instances of vehicle and test environment and to this end the Control Centre concept was developed to allow rapid prototyping of developed control code by progressively porting from the simulated environment through hardware-in-the-loop (HIL) testing to implementation on Tethra using the Control Centre concept which is described in chapter 3 and Appendix A.
Chapter 3: Tethra AUV Design Concepts

The following sections describe the hardware, embedded controller electronics, software, sensors, the simulator virtual instance of Tethra and the ‘Control Centre’.

3.2 Mechanical system design

The mechanical configuration of the Tethra craft is based on an open frame design. Two central stainless steel cylindrical hulls house the vehicle’s main electrical and electronic systems. These central hulls are interconnected by two hollow stainless steel box sections and are suspended within a stainless steel tubular frame (Nolan, 2003). The overall shape of the craft is an inverted tetrahedron; a shape that is inherently stable in the water provided the vehicle’s centre of gravity is kept low.

The upper central hull is used to house the robot’s onboard computer, principal electrical systems and control circuitry while the bottom hull is used to house the heaviest components of the craft; the onboard battery power source (Figure 3.2).

![Figure 3.2 Mechanical design of Tethra.](image)

The weight of whole craft is about 180 kg without the navigational sensor suite payload. Extra buoyancy is incorporated into the craft in the form of four buoyancy tubes, two foam-filled and two float-filled. These are positioned as high as possible on the crafts tubular frame in order to enhance craft stability. The fixed buoyancy afforded by the tubes can be manually
adjusted to give the whole vehicle slightly positive buoyancy in the water. Two additional buoyancy tubes are available to accommodate for the increased weight when navigation sensors are attached. In these conditions the vehicle weight is typically between 260-300 kg, depending on the employed payload sensors.

### 3.3 Vehicle power subsystems

Tethra’s power source is comprised of five 12-volt batteries (Figure 3.3) housed in the AUV’s lower hull enclosure. The largest of the five batteries, a 12-volt 97Ah lead gel battery delivers power to the crafts four thrusters, lower voltage supplies are also derived from this battery to power system sensors and electronics. These lower voltages are achieved by means of a 12 to 5volt dc/dc converter and a 12 to +/-12-volt bipolar dc/dc converter. The AUV’s onboard computer is powered via a separate 5-volt dc/dc converter and two small 12 volt 6 Ah battery connected in parallel. Two other small 1.8 Ah 12-volt batteries connected in series are used to provide a power supply compatible with actuators requiring 24-volts.
3.4 Actuators

The MinnKota thrusters used are 12-volt trolling motors that have been adapted for use on the Tethra vehicle. Similar commercially available electric outboard trolling motors have been used for the Kambara vehicle (Wettergreen, Gaskett et al., 1999), and for the Oberon submersible (Williams, Newman et al., 2000).

Each thruster is capable of delivering 180N thrust. There are four controllable degrees of freedom due to the particular thrusters, thus being able to control the yaw moment, surge force, heave force and the roll moment.

To identify the thruster response static bollard pull tests have been carried out. In these tests the power amplifiers and the thermal trip switches used in Tethra limited the maximum thruster current that could be applied. The motor control amplifiers used due to budgetary constraints were Sonik3 Marine Max PWM amplifiers designed for use in RC systems. These amplifiers can not provide the rated thrust of the MinnKota thrusters due to current
Chapter 3: Tethra AUV Design Concepts

limitations. The pull tests were performed with PWM duty cycle up to 40% of maximum. Across this range the thrust / duty cycle showed a linear response with 20N at 10% up to 80N at 40% in forward direction. The same tests were carried out in the reverse direction, the thrust / duty cycle indicating a response of 10N at 10% up to 40N at 40%, revealing a non-symmetrical force range of the thrusters. The static bollard pull tests have helped to identify $T_{\pm}$ thrust coefficients and to determine the relationship between propeller thrust and propeller revolution. Identification of the thrust coefficient was decisive for proper implementation of the “Correction” stage, required to compensate for the non-symmetrical thruster force ranges.

3.5 Active ballast system

An active ballast control system (Figure 3.4) is incorporated into the vehicle’s design to facilitate adjustments to buoyancy during the course of a mission (Love, Toal et al., 2003). This system allows neutral buoyancy control at any design depth and also allows the craft to carry out controlled dives and surfacing from depth under fault conditions. The system consists of four 2-litre ballast tanks, a compressed air supply, and a series of motorised and solenoid actuated valves that manage the distribution of air and water in the system.

To increase ballast, water is allowed to flood the tanks and air is vented. To decrease ballast the supply of pressurized air is used to force water from the tanks (Toal, 2004). The flow of air is controlled using two 24-volt solenoid operated uni-directional valves. The flow of water is controlled using a single bi-directional 24-volt electrically actuated ball valve. When the required amount of ballast is achieved the tanks are sealed off. A differential pressure transducer is used to measure the pressure difference between the ballast tanks and the outside environment.
If pressure differential is above a certain limit, due to the craft changing depth from that at which ballast was set and sealed off, the ballast control function is modified. With a differential pressure above 1 bar as monitored by the differential pressure transducer, the internal ballast tank pressure is equalised to the outside ambient pressure by adding or venting air before the bi-directional water valve is operated. This is done to make the craft safer and ensure the effectiveness of neutral buoyancy control at greater depths (Love, Toal et al., 2003).
3.6 Sensors

3.6.1 Initial design

In the early stage of Tethra development, due to budget limitations, sensor suite mounted on the body of vehicle included Vector 2XG Compass, Depth & Ballast Transducer, Tritech Precision Altimeters and Xsens MT9 Inertial Measurement Unit (Figure 3.5).

![Figure 3.5](image)

**Figure 3.5** (a) Depth & Ballast Transducers; (b) Vector 2XG Compass; (c) MT9 Inertial Measurement Unit; (d) PA200-20 and PA500-6 Tritech precision altimeters.

**Depth & Ballast transducer**

The depth transducer (Figure 3.5.a.) used is a 10 bar g transducer from the UCC range of sensors manufactured by Parker Filtration, U.K (Nolan, 2003). It has a 1/4” BSPT threaded male fitting port fitting. The transducer can be powered from an 11-30Vdc unregulated supply. We are using a regulated 12volt supply to power the transducer. The transducer uses strain gauge technology in its design and outputs pressure readings as an amplified 0-5v DC output. This analogue output is compatible with Tethra’s data acquisition hardware. The depth transducer provides feedback for the Depth Rate and Auto Depth controller. The ballast pressure transducer used is also a UCC transducer with a pressure range of 0 to 20-bar. This pressure transducer has a similar size and configuration. It also offers a 5-volt dc amplified output. The maximum compressed air/water pressure designed for within the
pressurised active ballast system is 10 bar pressure. Therefore a 20 bar pressure transducer was chosen for the ballast system to allow pressures that may marginally exceed 10-bar to be measured accurately.

**Vector 2XG compass**
The Vector 2XG is a tilt compensated electronic magnetometer/compass (Figure 3.5.b.). The Vector 2XG compass utilises mechanical gimbals to minimise the effect of tilt (pitch and/or roll) on its two magneto-inductive compass sensors maintaining heading accuracy over the range of tilt. The compass unit’s sensors will still remain in the horizontal plane when the module is tilted up to 15° from that plane. The Vector2XG module has been fitted in its own watertight glass fibre enclosure. This enclosure is mounted to the top of the vehicle frame. The Vector 2XG has a synchronous serial port for communication with a host system. In this case the host system is an embedded computer onboard Tethra. The serial digital output port has been configured to deliver the heading information in a binary format.

**Tritech precision altimeters**
Tethra’s primary obstacle avoidance requirement is met with an array of simple directional sensors, used to detect any obstacle within a critical distance of the robot thus allowing the robot to react appropriately.

Six Tritech altimeters are used for this purpose (Figure 3.5.c.): a 200kHz operating frequency altimeter provides altitude information within the operating range of 1 to 100 metres, while five additional 500kHz operating frequency altimeters provides distance readings in the horizontal plane, within the operating range of 0.1 to 10 metres. The altimeters’ versatile feature of simultaneous, analog and serial output capabilities allows easy interfacing. In the
present set-up range measurements are obtained interrogating the altimeters by means of serial data communication. The downward looking sonar provides feedback for the Altitude Rate and Auto Altitude controller, while the horizontally placed sonars are primarily used for obstacle avoidance and navigation in controlled environment (diving test pool).

In an early stage of development, all the above sensor units, i.e. the electronic magnetometer, ballast/depth pressure transducers and sonar altimeters were interfaced to a single PIC microcontroller based Input/Output coprocessor board.

**Xsens MT9 Inertial Measurement Unit**

The Xsens IMU consists of three miniature solid-state accelerometers (to measure accelerations along x, y and z axis in the body-fixed frame), the rate-gyro sensors (vibrating beam concept, measures angular velocity based on the Coriolis effect) and three magnetometers. In addition it provides also temperature information.

The IMU can deliver raw data - without calibration, direct from the sensor. In addition, other data is available, including attitude (orientation between the body-fixed coordinate system and the earth-fixed reference coordinate system), and 3D calibrated accelerations, rate of turn and earth magnetic field readings in the body-fixed coordinate system.

The IMU’s proprietary sensor fusion algorithm calculates the absolute orientation, integrating in time the angular velocity from the rate-gyro sensors, and using as external reference the vertical acceleration - provided by the accelerometers (gravitational acceleration), and North - provided by the magnetometers. The sensor provides the absolute orientation in the format of Euler angles (roll, pitch, yaw) or quaternions \((q_0, q_1, q_2, q_3)\). The
Chapter 3: Tethra AUV Design Concepts

IMU system provided sensor feedback for the low-level control system development. Heading readings were used initially as feedback for the Auto Heading controller.

3.6.2 New design

As mentioned in Chapter 1:, as part of the research project “Deep Ocean Habitat Mapping Using an ROV” funded by HEA PRTLI III, the Mobile & Marine Robotics Research Centre has acquired the state-of-the-art toolskid navigation sensors, including an Ixsea Phins Inertial Navigation System (INS); an RDI Doppler Velocity Log (DVL) and a CDL MicroBath Depth & Altitude Sensor.

![Image of toolskid navigational sensors: (a) IXSEA Phins, (b) RDI DVL, (c) CDL Microbath.](image.png)

**Figure 3.6** Toolskid navigational sensors: (a) IXSEA Phins, (b) RDI DVL, (c) CDL Microbath.

**IXSEA Phins**

The Phins navigation system is made up of three components (IXSEA, 2004):

1. The IMU (comprising three 0.01 degree per hour fibre optic gyroscopes and three 500 micro-gravity accelerometers) providing raw accelerations and rotations in the body-fixed frame;
2. The pure inertial navigation system, resolving the inertial measurements in the
navigation frame and updating the attitude, heading, velocity and position of the
Phins;
3. The Kalman filter (see Appendix D), integrating the measurements from the pure
inertial system with measurements from the external sensors to provide corrections to
inertial system errors.

![Figure 3.7 Functional block diagram of Phins.](image)

A functional block diagram of Phins is shown in Figure 3.7. The Phins Kalman filter
(Appendix D) is designed to optimally integrate data from aiding sensors, such as GPS,
acoustic positioning systems, Doppler Velocity Log (DVL) and depth sensor. Depending on
their availability, the aiding sensors can be used simultaneously or separately. The Phins is
able to switch automatically between them (Napolitano, Cottreau et al., 2002). To achieve
the best performance, very accurate error models of each instrument are integrated in the
filter. Moreover, Phins is able to detect failures in external sensors and to reject erroneous measurement (Napolitano, Cottreau et al., 2002).

**Doppler Velocity Log**

The DVL is an RDI Workhorse Navigator DVL, operating at 600kHz. The Doppler transducer unit has four downward-looking beam transducers oriented at about 30º from the instrument vertical axis and each rotated through 90º from the next about the vertical axis (see Figure 3.6.b). Doppler frequency shifts for the four beams, which result from relative motion between the DVL and the seabed, are resolved into $x$, $y$ and $z$ velocities of the sensor/vehicle.

**CDL Microbath**

The Microbath unit consists of a Mesotech altimeter and a Digiquartz depth sensor.

**3.7 Electronic and electrical hardware design**

The robot electronic control hardware is designed in the form of a distributed control architecture (Figure 3.8), in which a number of separate modules are used to perform complex tasks. The advantage of distributing the workload to different modules is that it frees the main processor or controller from performing repetitive albeit complex time-consuming tasks.
Chapter 3: Tethra AUV Design Concepts

3.7.1 Onboard embedded control system

Onboard PC104 computers

Autonomous control delegates control of robot actuators to the robot’s onboard computers.

The first onboard computer is the CM/P5e embedded 166 MHz Pentium CPU module with 32 Mbytes of RAM and 8 Mbytes of “Disk-On-Chip” flash memory. The module conforms to the PC104 standard, allowing a range of useful I/O expansion boards to be mapped into its memory space. Three Input/Output co-processor boards are currently in use, based on the 16F877A PIC microcontroller and interfacing sensors and actuators to the onboard computer.

The first I/O coprocessor board is responsible for sensor data acquisition; sensors such as an electronic magnetometer and the six Tritech precision altimeters are interfaced to this board.

Figure 3.8 Block diagram indicating distributed electronic design of Tethra.
The second coprocessor board generates the required PWM control signals for open loop motor control. The third board provides interface to the pressure/depth and ballast transducers; it is responsible for variable buoyancy control by actuating the ballast tanks’ valves and it also serves as a platform for the emergency buoyancy control failsafe behaviour which drives the craft up to the surface in case of an emergency by blowing the ballast tanks with air. This coprocessor board also provides low-level control functions such as timing and sequencing of valves.

The second onboard computer is an embedded 300 MHz Geode CPU module with 128 Mbytes of RAM and 2 Gigabytes of IDE flash-drive. This second embedded CPU board was required for Windows based software development, to fully support Xsens Inertial Measurement Unit and PHINS INS software’s platform requirements and output capabilities. Having a second CPU module allows greater flexibility in the future for further sensor/actuator or control interface development and testing.

**Embedded software development**

The control software for the main embedded CPU board is developed in Borland C, and runs under the real-time multitasking operating system MicroC/OS-II. The MicroC/OS-II is a highly portable, ROMable, scalable, pre-emptive real-time, multitasking kernel (RTOS) for microprocessors and microcontrollers with performance comparable and in some cases exceeding that of many commercially available kernels. Written in ANSI C for maximum portability, MicroC/OS-II has been ported to more than 40 different processor architectures ranging from 8- to 64-bit CPUs. MicroC/OS is being used by thousands of developers worldwide in applications ranging from cameras to avionics. Certifiable for use in safety-critical systems, this RTOS has been proven to be robust and reliable (Labrosse, 1999). The
real-time kernel always executes the highest priority task ready to run. The operating system is deterministic, allowing the user to know how much time MicroC/OS-II will take to execute a function or service and when a higher priority task will get control of the CPU. The MicroC/OS-II also provides an interrupt management facility and a host of other services useful in program development such as mailboxes, semaphores, fixed-sized memory partitions and time related functions. The RTOS allows creation of up to 56 application tasks, each of these being assigned a unique priority, its own set of CPU registers, and its own stack area (Labrosse, 1999).

The Tethra craft has been endowed with software level functionality for the integrated hardware and sensor systems by developing a framework as detailed in the following. To date three interrupt service routines have been developed to service pre-programmed events as indicated by their assigned hardware interrupts:

- A new set of acquired sensor data consisting of compass heading, depth-, ballast pressure information and sonar range readings is signalized by the hardware interrupt IRQ5, generated by a PIC I/O coprocessor board.

- The second ISR for hardware interrupt IRQ9 services interrupts generated by the COM3 serial port, signalizes the incoming IMU data string from the Geode CPU board. The IMU data string provides information about Tethra’s orientation vector (roll, pitch and yaw), yaw angular rate and the number of rotations around the Z axis. The latter is intended to be followed to keep track of the tethers’ torsion. The yaw and yaw rate data is derived by a Discrete Kalman Filter algorithm (Appendix D) running on the Geode processor.
The third ISR services the interrupts generated by the hardware interrupt IRQ3, generated by COM2 serial port, signalizing incoming control data string from the Labview GUI interface (Control Centre) to the Pentium CPU board. This GUI control string contains conditioned sensor information; low-level controller setpoints (i.e. desired depth, desired altitude, desired heading); controller parameters such as proportional, derivative gain; and expressions of forces and moments as generated by the hand control unit – joystick for operation in ROV mode.

The Pentium CPU board’s data string sent to the GUI consists of the sampling time to be able to follow the CPU’S program execution; fluxgate compass heading, depth/ballast pressure information, six sonar range readings and IMU data for provisional sensor data conditioning. The data string also contains the surge, heave force; yaw and roll moment control allocation expressions, calculated by the low-level controllers running on Tethra’s onboard computer. The force and moment expressions allow the assessment of the parameters of the controllers and their on-line tuning from the GUI interface.

Soon after initial testing, the bidirectional serial communication, consisting of the data string sent from the real-time kernel and control string received from the Control Centre has been replaced / upgraded using an Ethernet link. Bidirectional communication between the main CPU and the Control Centre uses UDP protocol, and is realised in the real-time kernel, using the WATTCP network programming library (Engelke, 2003).

The behaviour building blocks of the reactive control architecture were developed as tasks within the real-time multitasking operating system, on top of the aforementioned functional framework. The behaviours that were developed on the vehicle include the obstacle
avoidance behaviours, transect/trajectory following, height-off-bottom control, depth control, diving and surfacing behaviours along with emergency ballast control behaviours.

Visual C++ was used for software development on the Geode processor based embedded CPU board. At the initial stage of development, the MT9 IMU capabilities were integrated in software application by using the MT9 Software Development Kit, which incorporates the IMU as a COM object (Figure 3.9). A custom Discrete Kalman Filter (Appendix D) was designed in Matlab within the research group to improve the estimation of yaw (heading) and yaw rate. Matlab COM builder was used to create the DKF COM object (.dll file), which was easily integrated into control software developed in Visual C++. Later, this module was removed after acquisition of the more powerful Kalman filter incorporated into the PHINS unit.

The state-of-the-art INS (IXSEA PHINS) and external aiding sensors (RDI Workhorse Navigator DVL and Microbath depth sensor) provide a significant improvement over the earlier integrated navigation sensors. The Geode CPU board and the attached serial port
expansion board fully support both the hardware and software interfacing requirements of these sensors. An NMEA parser described in (Variakojis, 2002) is implemented in Visual C++ and used for parsing of the PHINS Standard and PHINS Control output protocols. The computationally high overhead parsing task of incoming data through the serial port is realised by the Geode CPU board. After processing, data are retransmitted in a customisable format to the real-time target and/or to the Control Centre, through the Ethernet link using UDP protocol.

3.7.2 Operation modes

Besides radio control available on the water surface, there are fundamentally two modes of operation available for the Tethra underwater vehicle: AUV mode and ROV mode. This operation duality can be used to prove ROV capability and also provides for surface monitoring of AUV mode operation close to the seabed and other hazards. The dual approach is prudent as a significant focus of the research is towards developing autonomous capabilities for operation scenarios not easily addressed by torpedo-shaped AUV survey systems e.g. operation in confined spaces and close to the seabed. The overall system architecture is depicted in Figure 3.10.

![Figure 3.10 Overall system architecture.](image)
3.7.3 Control Centre

The Control Centre is a LabVIEW application, which coordinates the work of the other two modules. Depending on the operation mode, it sends control strings to and receives data strings from the onboard system/simulator. In a special mode, it can forward navigation data from the INS to simulation, which can be used to visualise motion of the virtual Tethra in the virtual pool in real-time. This feature provides a framework to evaluate the performance of the INS by direct visual comparison of motion of the real and virtual Tethra. In addition, the Control Centre provides a user-friendly GUI to monitor navigation data in real-time, to analyse and tune parameters of the low-level controllers, to acquire data for off-line system identification and to test the performance of the overall control system. The Control Centre has been built using a modular approach, where each unit of the control block diagram is performed as a sub VI (virtual instrument). In this way, the Control Centre is used for fast development, testing and integration of different control strategies using rapid control prototyping (RCP) and hardware-in-the-loop.

3.7.4 Real-time simulator

The AUV simulator is developed in the Matlab/Simulink environment. It consists of the nonlinear model of the vehicle in 6 DOF, propulsion system and virtual reality display with virtual model of Tethra in the virtual pool. The virtual and real world objects (Tethra & test pool) have identical shape and size. Navigation data from real experiments can be used to visualise motion in the virtual pool in real time. More information about the AUV simulator can be found in Appendix A.
3.7.5 Flexible connection points

The concept of a flexible connection point has been developed to increase efficiency and flexibility of the overall system. This concept allows the Control Centre and onboard Systems to be connected in different points. Using RCP, each module (sub VI) in Control Centre is rigorously tested in simulation and real-world tank tests. After all simulation tests are passed, the functionally equivalent module is realised as a separate thread in the multi-tasking environment in OS1 and a new connection point is generated. This duality of code implementation enables fast prototype development in LabVIEW and efficient porting to the real-time target. Thus, one part of the control code is executed in the Control Centre and the other part in OS1. When the final version passes all acceptance tests, the entire control code is implemented in OS1 as separate threads and the Control Centre plays only a monitoring role.

3.7.6 Communication

Ethernet

Communication between onboard systems and the surface unit is established using Ethernet. Although the TCP protocol ensures reliable transmission across a network, the UDP protocol was chosen to transfer data, due to its simplicity and speed. As a result of the distributed control design, for surface control, the reliability of transmission is not critical and rare lost segments are not very problematic. From the control point of view, a more important issue is to keep time synchronisation with sensor data and to reduce delays in the control loop.

Radio controller

The actuators of the Tethra vehicle may also be controlled using remote radio controller, in addition to autonomous mode control and the hand control unit. Under remote radio control a
Chapter 3: Tethra AUV Design Concepts

9-channel radio link allows control of all craft actuators and facilitates the transfer of robot control to and from the onboard autonomous controller. This ‘transfer’ is made possible through the use of a bank of analogue switches that route either R/C control signals or autonomous control signals to the actuator circuitry. The mode of control is determined by the state of a single ‘mode select’ R/C channel input applied to the analogue switches (Nolan, 2003). The radio link is intended for use solely when the craft is on the surface for launch and retrieval of the vehicle and to facilitate a manual trim of craft buoyancy to a slightly positively buoyant state prior to switching to autonomous control.

3.8 Conclusion

This chapter has described the Tethra vehicle design and its sub systems, both before and after the acquisition of the precision navigation suite of sensors. The author has worked within the Tethra development team and has had prime responsibility for the development of the embedded controller electronics and code.

The design-implement-test-evaluate cycle including mobilising for pool and lake tests proved very long. This long cycle constrained the rate of development and for this reason a Tethra simulator including a virtual reality instance of Tethra and the test pool has been developed. Also the Control Centre approach, which allowed progressive development in simulation, with hardware in the loop testing and right through to testing developed code on the vehicle in pool and lake tests significantly shortened the development life cycle and accelerated the development of the vehicle control systems. The following chapters describe these developments and testing in detail.
Chapter 4: Guidance, Navigation and Control for Tethra AUV

4.1 Introduction

A novel guidance, navigation and control (GNC) system for the Tethra AUV is described in this chapter. Material presented in this chapter is closely related with the basic concepts of propulsion system and control allocation introduced in section 2.2. The hybrid control architecture is modified to accommodate thruster configuration of Tethra.

This chapter is organised as follows: section 4.2 provides conceptual elements and reveals the main idea of the proposed GNC system. Guidance and navigation are defined in sections 4.3 and 4.4, respectively. Section 4.5 describes the control system. Topics include introducing basic concepts, nomenclature and terminology of control allocation, description of normalisation procedure, problem formulation, solution and its geometrical interpretation.

The architecture of the control system is also described, including a full description of the main components. Section 4.6 summarises concluding remarks.

4.2 Conceptual elements

The overall system for thrusters-propelled AUVs is constructed from three independent blocks, denoted as the guidance, navigation and control (GNC) systems (Fossen, 2002). These systems interact with each other through data and signal transmission, as depicted in Figure 4.1. Components of the control system (Control Allocation, Low-level Controllers, Virtual Joystick and Behaviour Coordinator/Arbitration) are described in the rest of this
Chapter 4: Guidance, Navigation and Control for Tethra AUV

chapter, while the component of the guidance system (Trajectory Generator) is described in chapter 5.

4.3 Guidance

Guidance is the action of the system that continuously computes the reference (desired) position, velocity and acceleration of a vessel to be used by the control system (Fossen, 2002). The basic components of a guidance system are motion sensors, external weather data and a computer, which collects and processes the information and provides the results to the control system. Advanced optimisation techniques are utilised to compute the optimal trajectory or path using different optimisation criteria, such as fuel optimisation, minimum time navigation, collision avoidance etc. A full description of the guidance system used in the Tethra simulator is given in chapter 5.

4.4 Navigation

Navigation is the science of directing a craft by determining its position, velocity, acceleration, course and distance travelled. This is usually done by using a satellite navigation system combined with motion sensors like accelerometers and gyros. The most
advanced navigation system for marine applications is the inertial navigation system (INS). As mentioned in section 3.6.2 (page 80), the state-of-the-art toolskid navigation sensors, including Phins INS, RDI DVL & Microbath Depth & Altitude Sensor, is used for navigation purpose and testing of control algorithms proposed in this chapter. Technical specifications of these sensors can be found in Appendix C.

4.5 Control

4.5.1 Introduction

*Control* is the action of determining the necessary control forces and moments to be provided by the vessel in order to satisfy a certain control objective, which is usually derived in conjunction with the guidance system (Fossen, 2002). Constructing the control algorithms include the design of feedback and feedforward control laws. The outputs from the navigation system (position, velocity and acceleration) are used for the feedback control, while feedforward control is implemented using signals available in the guidance system and other external sensors.

The overall control system (Figure 4.1) utilizes a hybrid control approach (Arkin, 1998), which combines the advantages of a top-down traditional (hierarchical) AI approach and a bottom-up behaviour-based approach. The control architecture is structured in three layers: the deliberative layer, the control execution layer and the functional reactive layer (Carreras, 2003). The deliberative layer transforms the mission into a set of tasks. The reactive layer takes care of the real-time issues related to the interactions with the environment. It calls a set of low-level controllers with auto-tuning capabilities, followed by a control allocator. The control execution layer interacts between the upper and lower layers, supervising the accomplishment of the tasks. Thus, the hybrid architecture takes advantage of the
Chapter 4: Guidance, Navigation and Control for Tethra AUV

hierarchical planning aspects of traditional AI and the reactive & real-time aspects of the behaviour-based approach.

The proposed/implemented control system for Tethra AUV utilises control allocation. A general control system architecture with control allocation is discussed in the following.

The majority of marine vessels represent control systems, for which it is possible to split the control design into the following steps (Omerdic, 2004):

- **REGULATION TASK**: Design a control law, which specifies the total control effort to be produced (net force, moment, etc.).

- **ACTUATOR SELECTION TASK**: Design a control allocator, which maps the total control effort (demand) onto individual actuator settings (thrust forces, control surface deflections, etc.).

![General control system architecture with control allocation.](image)

**Figure 4.2** General control system architecture with control allocation.

Figure 4.3 illustrates the configuration of the overall control system (Omerdic and Roberts, 2004). The control system consists of a control law (specifying the virtual control input, \( \mathbf{v} \in \mathbb{R}^k \)) and a control allocator (allocating the true control input, \( \mathbf{u} \in \mathbb{R}^m \), where \( m \geq k \)), which distributes control demand among the individual actuators). In the system, the actuators generate a total control effect, \( \mathbf{v}_{\text{sys}} \in \mathbb{R}^k \), which is applied as the input to the system dynamics block and which determines the system behaviour. The main objective of the control allocation is to ensure that the condition \( \mathbf{v}_{\text{sys}} = \mathbf{v} \) is satisfied for all attainable \( \mathbf{v} \). The standard constrained linear control allocation problem can be formulated as follows:
For a given $\mathbf{v}$, find $\mathbf{u}$ such that

$$\mathbf{Bu} = \mathbf{v} \quad \text{(4.1)}$$

$$\mathbf{u}_\mathbf{L} \leq \mathbf{u} \leq \mathbf{u}_\mathbf{U} \quad \text{(4.2)}$$

where the control effectiveness matrix $\mathbf{B}$ is a $k \times m$ matrix with rank $k$. Constraint (4.2) includes actuator position and rate constraints, where the inequalities apply component-wise.

### 4.5.2 Control allocation

**Assumptions**

The full simulation model of the propulsion system includes the dynamics of the thrusters control loop and the affine thruster model. However, the control allocation problem for Tethra is formulated and solved under the following assumptions (Omerdic, 2004):

- the dynamics of the thruster control loop are neglected,
- the relationship between propeller thrust/torque and the control variable is given by a modified version of the affine thruster model.

The influence of neglected factors on the performance of the Tethra control system can be investigated using the AUV simulator incorporating the bilinear thruster model and the dynamics of the thruster control loop.

**Thruster configuration**

Tethra has two horizontal thrusters ($'HT', i = 1, 2$) and two vertical thrusters ($'VT', i = 1, 2$). Figure 4.3). The origin of the body-fixed reference frame $\{B\}$ is chosen to coincide with the CG (centre of gravity).
Chapter 4: Guidance, Navigation and Control for Tethra AUV

Figure 4.3 Thruster configuration of Tethra.

The axes are chosen to coincide with the principal axes of inertia and they are defined as:

- longitudinal axis $x_B$ (directed to front),
- transversal axis $y_B$ (directed to starboard),
- normal axis $z_B$ (directed down)

Thruster $\text{HT}$ ($\text{VT}$) exerts thrust (force) $\text{T}_{\text{HT}}$ ($\text{T}_{\text{VT}}$) and torque (moment) $\text{Q}_{\text{HT}}$ ($\text{Q}_{\text{VT}}$).

The position vector $\text{r}_{\text{HT}} = \begin{bmatrix} r_x \\ r_y \\ r_z \end{bmatrix}_{\text{HT}}$ ($\text{r}_{\text{VT}} = \begin{bmatrix} r_x \\ r_y \\ r_z \end{bmatrix}_{\text{VT}}$) determines the position.
of the point of attack of the force \( ^iT_{HT} \) \((^iT_{VT})\), relative to \( \{B\} \) (Table 4.1). The force \( ^iT_{HT} \) \((^iT_{VT})\) also generates the moment \( ^iQ_{HT} = ^i\mathbf{r}_{HT} \times ^i\mathbf{T}_{HT} \) \((^iQ_{VT} = ^i\mathbf{r}_{VT} \times ^i\mathbf{T}_{VT})\), so the total moment vector exerted by the thruster \( ^iHT \) \((^iVT)\) is given by \( ^iQ_{HT} = ^iQ_{eHT} + ^iQ_{HT} \) \((^iQ_{VT} = ^iQ_{eVT} + ^iQ_{VT})\). The orientation of the thruster \( ^iHT \) \((^iVT)\) relative to \( \{B\} \) is defined by the unit vector \( ^ieHT = \left[ ^ie_x \ ^ie_y \ ^ie_z \right] \), \((^ieVT = \left[ ^ie_x \ ^ie_y \ ^ie_z \right])\) (Table 4.2). The vector \( ^ieHT \) \((^ieVT)\) shows the positive direction of the force \( ^iT_{HT} \) \((^iT_{VT})\). This means that, if the propeller angular velocity is positive, it will exert the force \( ^iT_{HT} \) \((^iT_{VT})\) in the direction of \( ^ieHT \) \((^ieVT)\). Otherwise, the force \( ^iT_{HT} \) \((^iT_{VT})\) is opposite to \( ^ieHT \) \((^ieVT)\).

**Table 4.1** Position vectors for thruster configuration of Tethra.

<table>
<thead>
<tr>
<th>Horizontal thrusters</th>
<th>Vertical thrusters</th>
</tr>
</thead>
<tbody>
<tr>
<td>( ^1r_{HT} )</td>
<td>( ^2r_{HT} )</td>
</tr>
</tbody>
</table>
| \(
\begin{bmatrix}
  r_{hx} \\
  -r_{hy} \\
  0 
\end{bmatrix} \) | \(
\begin{bmatrix}
  r_{hx} \\
  r_{hy} \\
  0 
\end{bmatrix} \) | \( 0 \) | \( 0 \) |

**Table 4.2** Orientation vectors for thruster configuration of Tethra.

<table>
<thead>
<tr>
<th>Horizontal thrusters</th>
<th>Vertical thrusters</th>
</tr>
</thead>
<tbody>
<tr>
<td>( ^1e_{HT} )</td>
<td>( ^2e_{HT} )</td>
</tr>
</tbody>
</table>
| \(
\begin{bmatrix}
  1 \\
  0 \\
  0 
\end{bmatrix} \) | \( 1 \) | \( 0 \) | \( 0 \) |

98
The relationship between propeller spin direction and direction of the thrust and the torque vector is described by introducing a *spin direction coefficient* \( c_{HT} \) \( (c_{VT}) \). The value \( c_{HT} (c_{VT}) = +1(-1) \) means that the force vector \( \textbf{T}_{HT} \) \( (\textbf{T}_{VT}) \) and the torque vector \( \textbf{Q}_{HT} \) \( (\textbf{Q}_{VT}) \) have the same (opposite) direction (Figure 4.4).

**Figure 4.4** Relationship between propeller spin direction and direction of the thrust and the torque vector \( (c_{HT} = +1 \text{ left, } c_{HT} = -1 \text{ right}).

**Problem formulation**

In order to formulate the control allocation problem without separation into horizontal and vertical plane, the following nomenclature is introduced: subscripts \( HT \) and \( VT \) are removed, and indexes 1 & 2 (3 & 4) are assigned to thrusters \( HT \) & \( HT \) \( (VT \) & \( VT \)), respectively.

Vector of forces and moments, exerted by thruster \( Th \), can be written as:

\[
\vec{\tau} = \begin{bmatrix} \vec{T} \\ \vec{Q} \end{bmatrix} = \begin{bmatrix} \vec{T} \textbf{e} \\ \vec{T} (\vec{r} \times \textbf{e}) \end{bmatrix} = \begin{bmatrix} \vec{e}_x \\ \vec{e}_y \\ \vec{e}_z \\ (\vec{r} \times \textbf{e})_x \\ (\vec{r} \times \textbf{e})_y \\ (\vec{r} \times \textbf{e})_z \end{bmatrix} \vec{T} \tag{4.3}
\]

Superposition of the individual contributions \( \vec{\tau}_i \), \( i = 1,4 \) leads to total vector of propulsion forces and moments \( \vec{\tau} \) (Omerdic, 2004):
where $\mathbf{T} \in \mathbb{R}^{6 \times 4}$ is the \textit{thruster configuration matrix} and $\mathbf{f} \in \mathbb{R}^4$ is vector of control forces.

Substituting $\tau = K'u'$ in (4.4) yields

$$\tau = \begin{bmatrix} e_1^T \\ e_2^T \\ e_3^T \\ e_4^T \end{bmatrix} = \begin{bmatrix} 1 \quad 0 \\ 0 \quad 2 \quad 0 \\ 0 \quad 0 \quad 3 \quad 0 \\ 0 \quad 0 \quad 0 \quad 4 \end{bmatrix} \begin{bmatrix} u_1 \\ u_2 \\ u_3 \\ u_4 \end{bmatrix} = \mathbf{T}\mathbf{K}\mathbf{u} \quad (4.5)$$

where $\mathbf{K} \in \mathbb{R}^{4 \times 4}$ is the \textit{force coefficient matrix} and $\mathbf{u} \in \mathbb{R}^4$ is the \textit{control vector}. Introducing

$$\mathbf{B} = \mathbf{T}\mathbf{K} \quad (4.6)$$

where $\mathbf{B} \in \mathbb{R}^{6 \times 4}$ is the \textit{thruster control matrix}, (4.5) can be rewritten as

$$\tau = \mathbf{B}\mathbf{u} \quad (4.7)$$

A zero-row in $\mathbf{B}$ means that the corresponding DOF is not directly controllable with the thruster configuration of Tethra.

Assuming that thrusters are identical, the coefficients $'K$ are the same for all thrusters,

$$'K = 'K = 'K = 'K = K \quad (4.8)$$

and equation (4.6) can be simplified as

$$\mathbf{B} = \mathbf{T}\mathbf{K} = \mathbf{T}\left(\mathbf{K}\mathbf{I}_p\right) = \mathbf{K}\left(\mathbf{T}\mathbf{I}_p\right) = \mathbf{K}\mathbf{T} \quad (4.9)$$

Recall from (4.2) that each component $'u$ of the control vector $\mathbf{u}$ is limited by the constraint
Chapter 4: Guidance, Navigation and Control for Tethra AUV

\[ -u_m^i \leq u^i \leq u_m^i, \quad i = 1, 4 \quad (4.10) \]

Constraint (4.10) represents thruster velocity saturation, i.e. the physical construction of the thruster 'Th' imposes velocity limitations and the thruster cannot rotate faster than the maximum velocity. For the control vector \( \mathbf{u} \) the set of constraints (4.10) can be written in compact vector form as

\[ -\mathbf{u}_m \leq \mathbf{u} \leq \mathbf{u}_m \quad (4.11) \]

where

\[ \mathbf{u}_m = [u_m^1 \ u_m^2 \ u_m^3 \ u_m^4]^T \quad (4.12) \]

The constrained control subset \( \Omega \) is defined as a set of all control vectors \( \mathbf{u} \) which satisfy (4.11). The general control allocation problem for the open-frame underwater vehicles can be formulated as:

For given \( \mathbf{r} \), find \( \mathbf{u} \in \Omega \) such that \( \mathbf{Bu} = \mathbf{r} \).

The cross products in (4.5) are determined as

\[ ^1r \times e = \begin{bmatrix} 0 \\ 0 \\ r_{ny} \end{bmatrix} \quad (4.13) \]

\[ ^2r \times e = \begin{bmatrix} 0 \\ 0 \\ -r_{ny} \end{bmatrix} \quad (4.14) \]

\[ ^3r \times e = \begin{bmatrix} -r_{vy} \\ 0 \\ 0 \end{bmatrix} \quad (4.15) \]
Chapter 4: Guidance, Navigation and Control for Tethra AUV

$$^4 \mathbf{r} \times ^4 \mathbf{e} = \begin{bmatrix} r_{xy} \\ 0 \\ 0 \end{bmatrix} \quad (4.16)$$

Finally, the thruster control matrix $\mathbf{B}$ is

$$\mathbf{B} = \mathbf{K} \mathbf{T} = \mathbf{K} \begin{bmatrix} 1 & 1 & 0 & 0 \\ 0 & 0 & 0 & 0 \\ 0 & 0 & 1 & 1 \\ 0 & 0 & -r_{xy} & r_{xy} \\ 0 & 0 & 0 & 0 \\ r_{xy} & -r_{xy} & 0 & 0 \end{bmatrix} \quad (4.17)$$

It can be seen that the uncontrollable DOF for the Tethra craft are sway and pitch, since the second and the fifth row of $\mathbf{B}$ are zero-rows. Removing uncontrollable DOFs yields:

$$\mathbf{\tau} = \begin{bmatrix} \tau_x \\ \tau_z \\ \tau_K \\ \tau_N \end{bmatrix} = \begin{bmatrix} K & K & 0 & 0 \\ 0 & 0 & K & K \\ 0 & 0 & -K r_{xy} & K r_{xy} \\ K r_{xy} & -K r_{xy} & 0 & 0 \end{bmatrix} \begin{bmatrix} ^1 \mathbf{u} \\ ^2 \mathbf{u} \\ ^3 \mathbf{u} \\ ^4 \mathbf{u} \end{bmatrix} \quad (4.18)$$

where $\mathbf{B} \in \mathbb{R}^{4 \times 4}$ is a reduced thruster control matrix. Each component $^i \mathbf{u}$ of the control vector $\mathbf{u}$ is limited by the constraint (thruster velocity saturation) (4.10).
Figure 4.5 Relationship between thruster configuration and controllable DOFs.

**Normalisation**

Using the procedure described in (Omerdic, 2004), relevant vectors and matrices will be normalised, in order to make the problem more understandable and easier to visualize and solve. Normalisation means that vector components are divided by their maximum values, such that each component is a dimensionless number that lies between −1 and +1. Normalised vectors and matrices are underlined, in order to distinguish them from the standard nomenclature.

The first step is to find maximum values (modules) of the surge & heave forces and the roll & yaw moments. Four characteristic cases are indicated in Figure 4.6. It can be seen that

\[
\tau_{Xm} = 2T_m = 2Ku_m \Rightarrow K = \frac{\tau_{Xm}}{2u_m} \tag{4.19}
\]

\[
\tau_{Zm} = 2T_m = 2Ku_m \Rightarrow K = \frac{\tau_{Zm}}{2u_m} \tag{4.20}
\]
\[ \tau_{km} = 2T_m r_{sy} = 2K u_m r_{sy} \Rightarrow K_{r_{sy}} = \frac{\tau_{km}}{2u_m} \]  \hspace{1cm} (4.21)

\[ \tau_{nm} = 2T_m r_{hy} = 2K u_m r_{hy} \Rightarrow K_{r_{hy}} = \frac{\tau_{nm}}{2u_m} \]  \hspace{1cm} (4.22)

**Figure 4.6** Four cases for finding the maximum modules of force and moment vectors: (a) maximum surge force, (b) maximum yaw moment, (c) maximum heave force, (d) maximum roll moment.

The second step is to substitute expressions (4.19)-(4.22) in (4.18) as follows:
Chapter 4: Guidance, Navigation and Control for Tethra AUV

\[
\begin{bmatrix}
\tau_X \\
\tau_Z \\
\tau_K \\
\tau_N \\
\tau_m
\end{bmatrix} =
\begin{bmatrix}
K & K & 0 & 0 \\
0 & 0 & K & K \\
0 & 0 & -Kr_y & Kr_y \\
Kr_y & -Kr_y & 0 & 0 \\
\end{bmatrix}
\begin{bmatrix}
u \\
\tau_1 \\
\tau_2 \\
\tau_3 \\
\tau_4
\end{bmatrix}
\]

The final step is to rewrite (4.23) in the normalised form as follows:

\[
\begin{bmatrix}
\tau_{Xn} \\
\tau_{Zn} \\
\tau_{Kn} \\
\tau_{Nm}
\end{bmatrix} =
\begin{bmatrix}
\frac{\tau_{Xn}}{2u_m} & \frac{\tau_{Xn}}{2u_m} & 0 & 0 \\
0 & 0 & \frac{\tau_{Zn}}{2u_m} & \frac{\tau_{Zn}}{2u_m} \\
0 & 0 & -\frac{\tau_{Kn}}{2u_m} & \frac{\tau_{Kn}}{2u_m} \\
\frac{\tau_{Nm}}{2u_m} & -\frac{\tau_{Nm}}{2u_m} & 0 & 0
\end{bmatrix}
\begin{bmatrix}
u \\
\tau_1 \\
\tau_2 \\
\tau_3 \\
\tau_4
\end{bmatrix}
\]

Normalisation has a number of advantages compared to standard formulation (Omerdic, 2004):

1. Components of the vectors \( \tau \) and \( \mathbf{u} \) are dimensionless numbers, restricted to the standard interval \([-1, +1]\). This enables better understanding and easier visualisation of the problem.

2. All physical parameters are removed from the matrix \( \mathbf{B} \) during the normalisation process. The compact form of \( \mathbf{B} \) simplifies calculations and leads to clear geometric interpretation of the control allocation problem.
Problem formulation in normalised form

The control allocation problem for the Tethra craft can be formulated using normalised variables as follows:

For given $\bm{\tau}$, find $\bm{u} \in \Omega$ such that $\bm{Bu} = \bm{\tau}$.

In the following, the problem is analysed in more detail from the general control allocation perspective.

- The true control input is $\bm{u} = \begin{bmatrix} 1 \\ u_m \\ 2 \\ u_m \\ 3 \\ u_m \\ 4 \\ u_m \end{bmatrix} \in \mathbb{R}^4 \ (m = 4)$,

- The virtual control input is $\bm{\tau} = \begin{bmatrix} \tau_X \\ \tau_Nm \\ \tau_Z \\ \tau_N \end{bmatrix} \in \mathbb{R}^4 \ (k = 4)$

- The control effectiveness matrix is given by $\bm{B} = \begin{bmatrix} 1 & 1 & 0 & 0 \\ 2 & 2 & 1 & 1 \\ 0 & 0 & 2 & 2 \\ 0 & 0 & 1 & 1 \\ 1 & -1 & 2 & 2 \\ 2 & -1 & 2 & 0 \end{bmatrix}$
• Actuator position constraints are
\[ \begin{bmatrix} -1 & -1 & -1 & -1 \\ -1 & 1 & 1 & 1 \\ -1 & -1 & -1 & -1 \\ -1 & 1 & 1 & 1 \end{bmatrix} \begin{bmatrix} u_1 \\ u_2 \\ u_3 \\ u_4 \end{bmatrix} \leq \begin{bmatrix} 1 \\ 1 \\ 1 \\ 1 \end{bmatrix} \]

Equation \( \mathbf{Bu} = \tau \) represents the system of equations
\[
\begin{align*}
\frac{1}{2} u_1 + \frac{1}{2} u_2 &= \tau_X \\
\frac{1}{2} u_1 + \frac{1}{2} u_4 &= \tau_Z \\
\frac{1}{2} u_3 + \frac{1}{2} u_4 &= \tau_K \\
\frac{1}{2} u_2 - \frac{1}{2} u_4 &= \tau_N
\end{align*}
\] (4.25)

Each equation in (4.25) represents a hyperplane in \( \mathbb{R}^4 \). Consequently, (4.25) can be rewritten as
\[
\begin{align*}
\pi_X : & \quad \mathbf{N}_X^T \cdot \mathbf{u} = \tau_X \\
\pi_Z : & \quad \mathbf{N}_Z^T \cdot \mathbf{u} = \tau_Z \\
\pi_K : & \quad \mathbf{N}_K^T \cdot \mathbf{u} = \tau_K \\
\pi_N : & \quad \mathbf{N}_N^T \cdot \mathbf{u} = \tau_N
\end{align*}
\] (4.26)

where normal vectors \( \mathbf{N}_X, \mathbf{N}_Z, \mathbf{N}_K \) and \( \mathbf{N}_N \) are defined as
\[
\begin{align*}
\mathbf{N}_X &= \begin{bmatrix} \frac{1}{2} & \frac{1}{2} & 0 & 0 \end{bmatrix}^T \\
\mathbf{N}_Z &= \begin{bmatrix} 0 & 0 & \frac{1}{2} & \frac{1}{2} \end{bmatrix}^T \\
\mathbf{N}_K &= \begin{bmatrix} 0 & 0 & -\frac{1}{2} & \frac{1}{2} \end{bmatrix}^T \\
\mathbf{N}_N &= \begin{bmatrix} \frac{1}{2} & -\frac{1}{2} & 0 & 0 \end{bmatrix}^T
\end{align*}
\] (4.27)
are orthogonal on the hyperplanes $\pi_X$, $\pi_Z$, $\pi_K$ and $\pi_N$, respectively. The intersection of these hyperplanes is a point (solution) $\mathbf{u}$ that satisfies $\mathbf{Bu} = \mathbf{r}$. However, this solution must be feasible, i.e. it must satisfy the actuator position constraints. The actuator position constraints determine the constrained control subset $\Omega$, that is, the unit four-dimensional hypercube in $\mathbb{R}^4$:

$$\Omega = \{ \mathbf{u} \in \mathbb{R}^4 \mid \| \mathbf{u} \| \leq 1 \} \subset \mathbb{R}^4$$  \hspace{1cm} (4.28)

The procedure to obtain the feasible solution of the control allocation problem is described in section “Feasible solution of the control allocation problem” (page 111).

**Constrained control subset $\Omega$**

The following nomenclature was adopted for referring to $\Omega$ in (Omerdic, 2004). The boundary of the constrained control subset $\Omega$ is denoted by $\partial(\Omega)$. A control vector belongs to $\partial(\Omega)$ if and only if at least one of its components is at a limit. Vertices are the points on $\partial(\Omega)$ where each control receives a limit (min or max). In the general case, the number of vertices is equal to $2^m$. Vertices are numerated using the following rule: if the vertex is represented in a binary form, then “0” in the $k^\text{th}$ position indicates that the corresponding control component $u^k$ is at a minimum $u^k_\text{min}$, while “1” indicates that it is at a maximum $u^k_\text{max}$. Edges are lines that connect vertices and that lie on $\partial(\Omega)$. The edges are generated by varying only one of the $m$ control components, while the remaining $m-1$ are at their limits, associated with the two connected vertices. In the general case, the number of edges is equal to $2^{m-1}\binom{m}{1}$. Two vertices are connected by an edge, if and only if their binary representations differ in only one bit. A facet is defined as the set in the control space
obtained by taking all but two control components at their limits and varying the two free control components within their limits. Facets are plane surfaces on $\partial(\Omega)$ that contain two adjacent edges, i.e. two edges that have a common vertex. In the general case, the number of facets is equal to $2^{n-2} \binom{m}{2}$.

**Attainable command set $\Phi$**

The control effectiveness matrix $\mathbf{B}$ performs a linear transformation from the true control space $\mathbb{R}^m$ to the virtual control space $\mathbb{R}^k$. The image of $\Omega \subset \mathbb{R}^m$ is called the attainable command set, denoted by $\Phi$ (Omerdic, 2004). The attainable command set $\Phi$ is a subset of the virtual control space $\Phi = \{ \tau : \|\tau\| \leq 1 \}$ and represents a convex polyhedron, whose boundary $\partial(\Phi)$ is the image of the facets of $\Omega$.

Coordinates of vertices of $\Omega$ and $\Phi$ are given in Table 4.3, while the nomenclature is given in Table 4.4.

Both $\Omega$, the constrained control subset, and $\Phi$, the attainable command set, are four-dimensional convex bodies and cannot be easily visualised.
Chapter 4: Guidance, Navigation and Control for Tethra AUV

Table 4.3 Vertices of $\Omega$ and $\Phi$.

<table>
<thead>
<tr>
<th>$\Omega$</th>
<th>$\Phi$</th>
</tr>
</thead>
<tbody>
<tr>
<td>$u_{HT}$</td>
<td>$\tau_X$</td>
</tr>
<tr>
<td>$u_{HT}^2$</td>
<td>$\tau_N$</td>
</tr>
<tr>
<td>$u_{VT}$</td>
<td>$\tau_K$</td>
</tr>
<tr>
<td>$u_{VT}^2$</td>
<td>$\tau_J$</td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>Vertices</th>
<th>Edges</th>
<th>Facets</th>
</tr>
</thead>
<tbody>
<tr>
<td>0, 1, 2, 3, 4, 5, 6, 7, 8, 9, A, B, C, D, E, F</td>
<td>01, 02, 04, 08, 13, 15, 19, 23, 26, 2A, 37, 3B, 45, 46, 4C, 57, 5D, 67, 6E, 7F, 89, 8A, 8C, 9B, 9D, AB, AE, BF, CD, CE, DF, EF</td>
<td>0132, 0154, 0198, 0264, 02A8, 04C8, 1375, 13B9, 15D9, 2376, 23BA, 26EA, 37FB, 4576, 45DC, 46EC, 57FD, 67FE, 89BA, 89DC, 8AEC, 9BFD, ABFE, CDFE</td>
</tr>
</tbody>
</table>
Feasible solution of the control allocation problem

Since $B$ is a non-singular square matrix ($\det(B) = -0.25 \neq 0$), it is straightforward to find the solution of the problem:

$$Bu = \tau \Rightarrow u = B^{-1}\tau$$

$$B^{-1} = \begin{bmatrix} 1 & 0 & 0 & 1 \\ 1 & 0 & 0 & -1 \\ 0 & 1 & -1 & 0 \\ 0 & 1 & 1 & 0 \end{bmatrix}$$

Since $\|B^{-1}\| = 2$, it is easy to verify that $B^{-1}(\Phi_\tau) \supset \Omega$, i.e. $\exists \tau \in \Phi_\tau : u = B^{-1}\tau \in \Omega$. In other words, the solution (control vector $u$) can be feasible ($u \in \Omega$) or unfeasible ($u \notin \Omega$), depending on the position of $\tau$ in the virtual control space $\Phi_\tau$. In order to find a feasible approximation of an unfeasible solution $u$, two common approximations are used in the literature: $T$-approximation (truncation) and $S$-approximation (scaling). In the case of $T$-approximation, the approximation $u^*_T \in \partial(\Omega)$ ($\partial(\Omega)$ denotes the boundary of $\Omega$) is obtained from $u$ by truncating (clipping) all components, which exceed their constraints. In contrast, the $S$-approximation $u^*_S$ is obtained by scaling the unfeasible solution $u$ to the boundary $u^*_S \in \partial(\Omega)$ by the factor $f$:

$$f = \min \left( \frac{1}{\max \{u_i \mid u_i \neq 0\}} \right)$$

The output of the control allocator (normalised feasible control vector $u^*$), cannot be directly applied to drive the thrusters, i.e. it must be further processed. In the first stage (called “Transformation”, see Figure 4.7), it must be transformed into the vector of desired thruster
velocities \( \mathbf{n} \), using the transformation \( \mathbf{n}_i = \text{sgn} \, n_i \sqrt{|n_i|} \) for each component. In the second stage (called “Adaptation”), vector \( \mathbf{n} \) is transformed into the vector \( \mathbf{n}_0 \), which has the form adapted to the Thruster Control Unit of Tethra (each component of \( \mathbf{n}_0 \) is an integer number between -100 and +100).

**Figure 4.7** Different stages of the control allocation.

**Visualisation**

In order to visualise the solution and provide geometrical interpretation, the control allocation problem for Tethra is decomposed into two subproblems (Table 4.5), namely for motion in horizontal plane (thrusters \(^1\text{HT}\) and \(^2\text{HT}\)) and vertical plane (thrusters \(^1\text{VT}\) and \(^2\text{VT}\)).
Table 4.5 Control allocation problem for motion in the horizontal plane.

Find \( \mathbf{u}_{HT} \in \Omega_{HT} \) such that

\[
\begin{bmatrix}
\tau_x \\
\tau_N
\end{bmatrix} =
\begin{bmatrix}
1 & 1 \\
2 & 2
\end{bmatrix}
\begin{bmatrix}
\frac{1}{2} & \frac{1}{2} \\
1 & 1
\end{bmatrix}
\begin{bmatrix}
1 \\
2
\end{bmatrix}
\mathbf{B}_{HT}
\begin{bmatrix}
\mathbf{u}_{HT} \\
\mathbf{u}_{HT}
\end{bmatrix}
\]

True control input:

\[
\mathbf{u}_{HT} =
\begin{bmatrix}
\frac{1}{2} \\
\mathbf{u}_{HT}
\end{bmatrix}
\in \mathbb{R}^2 \quad (m = 2)
\]

Virtual control input:

\[
\tau_{HT} =
\begin{bmatrix}
\tau_x \\
\tau_N
\end{bmatrix}
\in \mathbb{R}^2 \quad (k = 2)
\]

Control effectiveness matrix:

\[
\mathbf{B}_{HT} =
\begin{bmatrix}
1 & 1 \\
2 & 2
\end{bmatrix}
\in \mathbb{R}^{2 \times 2}
\]

Inverse of control effectiveness matrix:

\[
\mathbf{B}_{HT}^{-1} =
\begin{bmatrix}
1 & 1 \\
1 & -1
\end{bmatrix}
\in \mathbb{R}^{2 \times 2}
\]

Table 4.6 Coordinates of vertices of \( \Omega_{HT} \) and \( \Phi_{HT} \).

<table>
<thead>
<tr>
<th>Constrained Control Subset ( \Omega_{HT} )</th>
<th>Attainable Command Set ( \Phi_{HT} )</th>
</tr>
</thead>
<tbody>
<tr>
<td>( \mathbf{u}<em>{HT} ), ( \mathbf{u}</em>{HT} )</td>
<td>( \tau_x ), ( \tau_N )</td>
</tr>
</tbody>
</table>
| \( \begin{bmatrix}
0 & -1 \\
1 & -1 \\
2 & 1 \\
3 & 1
\end{bmatrix} \) | \( \begin{bmatrix}
0 & -1 \\
1 & 0 \\
2 & 0 \\
3 & 1
\end{bmatrix} \) |

If a virtual control input \( \tau_{HT} \) lies inside or on the boundary of the \( \Phi_{HT} \), then the inverse solution \( \mathbf{u}_{HT} = \mathbf{B}_{HT}^{-1} \mathbf{\tau}_{HT} \) is feasible and lies inside \( \Omega_{HT} \), \( \mathbf{u}_{HT} \in \Omega_{HT} \) or on the boundary \( \mathbf{u}_{HT} = \partial(\Omega_{HT}) \). Mapping between virtual and true control spaces for motion in the horizontal plane is visualised in Figure 4.8.
Figure 4.8 Mapping between virtual and true control spaces (horizontal plane).
Table 4.7 Control allocation problem for motion in the vertical plane.

Find \( \mathbf{u}_{VT} \in \Omega_{VT} \) such that
\[
\begin{bmatrix}
\Sigma_z \\
\Sigma_K \\
\Sigma_T
\end{bmatrix}
= \begin{bmatrix}
1 & 1 \\
\frac{1}{2} & \frac{1}{2} \\
-\frac{1}{2} & -\frac{1}{2}
\end{bmatrix}
\begin{bmatrix}
u_{VT} \\
u_m \\
u_{VT}
\end{bmatrix}
\]

True control input:
\[
\mathbf{u}_{VT} = \begin{bmatrix}
u_{VT} \\
u_m \\
u_{VT}
\end{bmatrix} \in \mathbb{R}^2 \quad (m = 2)
\]

Virtual control input:
\[
\Xi_{VT} = \begin{bmatrix}
\Xi_z \\
\Xi_K \\
\Xi_T
\end{bmatrix} \in \mathbb{R}^2 \quad (k = 2)
\]

Control effectiveness matrix:
\[
\mathbf{B}_{VT} = \begin{bmatrix}
1 & 1 \\
\frac{1}{2} & \frac{1}{2} \\
-\frac{1}{2} & -\frac{1}{2}
\end{bmatrix} \in \mathbb{R}^{2 \times 2}
\]

Inverse of control effectiveness matrix:
\[
\mathbf{B}_{VT}^{-1} = \begin{bmatrix}
1 & -1 \\
1 & 1
\end{bmatrix} \in \mathbb{R}^{2 \times 2}
\]

Table 4.8 Coordinates of vertices of \( \Omega_{VT} \) and \( \Phi_{VT} \).

<table>
<thead>
<tr>
<th>Constrained Control Subset ( \Omega_{VT} )</th>
<th>Attainable Command Set ( \Phi_{VT} )</th>
</tr>
</thead>
</table>
| \( \begin{bmatrix}
1 \\
2 \\
3
\end{bmatrix} \) | \( \begin{bmatrix}
0 \\
-1 \\
1
\end{bmatrix} \) |

If a virtual control input \( \Xi_{VT} \) lies inside or on the boundary of the \( \Phi_{VT} \), then the inverse solution \( \mathbf{u}_{VT} = \mathbf{B}_{VT}^{-1} \Xi_{VT} \) is feasible and lies inside \( \Omega_{VT} \), \( \mathbf{u}_{VT} \in \Omega_{VT} \) or on the boundary \( \mathbf{u}_{VT} = \partial(\Omega_{VT}) \). Mapping between virtual and true control spaces for motion in the vertical plane is visualised in Figure 4.9.
Figure 4.9 Mapping between virtual and true control spaces (vertical plane).
Chapter 4: Guidance, Navigation and Control for Tethra AUV

**Approximation error**

The approximation error is defined as \( e = \tau_d - \tau \), where \( \tau_d \) is desired normalised vector of forces and moments, and \( \tau = Bu \). In order to be able to compare different approximations, two scalar errors are introduced (Omerdic, 2004):

- **Direction error:** \( \theta = \cos^{-1} \frac{\tau_d \cdot \tau}{\|\tau_d\| \|\tau\|} \),

- **Magnitude error:** \( \|e\|_2 = \|\tau_d - \tau\|_2 \).

The direction error represents the angle between \( \tau_d \) and \( \tau \), while the magnitude error represents the module of the approximation error vector \( e \) (Figure 4.10.a). In the case when \( \theta = 0 \), the approximation \( \tau \) preserves the direction of the original vector \( \tau_d \) (Figure 4.10.b).

Example 4.1 (Comparison of \( T \) - and \( S \) -approximations)

Assume that desired virtual control input is \( \tau_{dHT} = \begin{bmatrix} 0.6 & 0.8 \end{bmatrix}^T \). The inverse solution is unfeasible, that is \( u_{HT} = B_{HT}^{-1} \tau_{HT} = \begin{bmatrix} 1.4 & -0.2 \end{bmatrix}^T \). Since the first component violates constraints, the \( T \)-approximation is \( u_{HT}^T = \begin{bmatrix} 1.0 & -0.2 \end{bmatrix}^T \), the obtained virtual control input is \( \tau_{HT}^T = B_{HT} u_{HT}^T = \begin{bmatrix} 0.4 & 0.6 \end{bmatrix}^T \) and the approximation errors are \( e_{HT}^T = \tau_d - \tau_{HT} = \begin{bmatrix} 0.2 & 0.2 \end{bmatrix}^T \), \( \|e_{HT}^T\| = 0.2828 \) and \( \theta^T = 3.1798^\circ \). For the \( S \)-approximation the scaling factor is
\[ f_{HT} = \min \left( \frac{1}{\max \| \begin{bmatrix} 0.4 \\ -0.2 \end{bmatrix} \|} \right) = 0.7143, \text{ which yields } \mathbf{u}_{HT}^S = f_{HT} \mathbf{u}_{HT} = \begin{bmatrix} 1.0 \\ -0.1429 \end{bmatrix}^{T}. \]

\[ \mathbf{e}_{HT} = \mathbf{t}_{HT} - \mathbf{t}_{HT}^S = \begin{bmatrix} 0.1714 \\ 0.2286 \end{bmatrix}^{T}, \| \mathbf{e}_{HT}^S \| = 0.2857 \text{ and } \theta^s = 0.0000^\circ \text{ (Figure 4.11).} \]
4.5.3 Low-level controllers

The *set-point tracking problem* is the problem of tracking a constant or step reference command. The *trajectory tracking problem* is the problem of tracking time varying reference input commands (Spong, Hutchinson et al., 2006). A set of low-level controllers (Figure 4.12) includes an Auto-Heave Controller (to maintain/track desired depth/altitude), an Auto-Heading (to maintain desired heading) and a Surge-Brake Controller (to stop surge motion in the absence of a demand to move forward/backward, like a car “brake”). Each controller can be separately activated, depending on the state of the “Active” inputs. Auto-tuning features of the low-level controllers are realised using the Response Optimisation Blockset in Simulink. In the ROV operation mode, the ROV pilot activates low-level controllers using joystick buttons, depending on the state of the mission. In the AUV operation mode, the active states of low-level controllers are controlled by the Behaviour Co-ordinator/Arbitration component through the Winner Control Cluster (see Figure 4.20).

Figure 4.12 Internal structure of the Low-Level Controllers component.
Auto-Heave controller

Problems of depth and altitude control are dual to each other. In the following the depth controller is described in more detail, while an equivalent altitude controller can be obtained by slight modification of input signals. In order to track time varying trajectories in the vertical plane, a feedforward control scheme is introduced to improve the performance of the Auto-Depth controller.

![Control scheme for depth set-point/trajectory tracking.](image)

Suppose that $z_d(t)$ is an arbitrary depth reference trajectory and consider the block diagram shown in Figure 4.13, where $z$ is actual depth, $w$ is actual vertical speed in the body-fixed frame ($w$ is very close to $\dot{z}$, since the roll and pitch angles are very small), $w_f = \dot{z}_d$ is the desired vertical speed, $K_p$, $K_d$ and $K_f$ are proportional, derivative and feedforward gains, respectively. The output of the controller is the $\tau_z$ component of the output vector $\tau_{LLC}$. A feedforward control scheme consists of adding a feedforward path to the output of PD controller using switch S. When the switch S is open, the feedforward path is not active and the depth controller is a standard PD controller for the set-point tracking problem. When the switch S is closed, the feedforward path becomes active and the depth controller becomes an enhanced PD controller for the trajectory tracking problem.
The input to the feedforward block is determined from

\[ w_{ff} = \frac{d_z}{d_{sy}} u \]

where \( d_{sy} \) is the distance between the vehicle and target way-point, projected to the \( x\)-\( y \) plane, \( d_z \) is the difference in depth and \( u \) is the linear velocity of the vehicle along the longitudinal axis in the body fixed frame (Figure 4.14).

Feedforward control is usually employed to speed up the system response (Marshall, 1978). The feedforward control has the following main disadvantages/limitations (Becerra, 2003):

- The availability of a reference signal correlated to the disturbance;
- Any changes in the parameters of the process cannot be compensated by a feedforward controller;
- Feedforward control requires a good process model;
- The feedforward control due to its open-loop nature should never be used except in conjunction with an associated feedback loop (Marshall, 1978). In pure feedforward control there is no monitoring on the controlled variable, i.e. if the controlled variable strays from its set-point, there is no corrective action to eliminate the error (Ecosse, 1990).
For the time-varying depth tracking of the Tethra vehicle a combined feedback-feedforward control system is employed, where the feedback control handles process uncertainties and disturbances, while the feedforward gives the desired response to reference signals (Astrom, 2002).

In order to compensate for steady-state errors for the Auto-Heave Controller in the case when the vehicle is not neutrally buoyant, the Heave Auto-Correction algorithm was developed, which is able to reduce steady-state error to zero. Essentially, the steady-state error is determined during a tuning phase and added as a constant offset to the input $z_d$.

**Auto-Heading controller**

The Auto-Heading controller is realised as a standard PD controller for the set-point tracking problem, where $\psi_d$ is the heading set-point, $\psi$ is actual heading, $r$ is the actual rotation rate about the vertical axis in the body-fixed frame ($r$ is very close to $\psi$, since the roll and pitch angles are very small), and $K_p$ & $K_d$ are proportional and derivative gains, respectively. The internal structure of the Auto-Heading controller is indicated in Figure 4.15. The output of the controller is the $\tau_N$ component of the output vector $\tau_{LLC}$.

![Figure 4.15 Control scheme for heading set-point tracking.](image-url)
Chapter 4: Guidance, Navigation and Control for Tethra AUV

Surge-Brake controller

The Surge-Brake controller is designed such that in the active state it generates normalised surge force to oppose actual forward motion of the vehicle, i.e. its role is the same as the role of a mechanical damper, since it exerts the force to stop the surge motion in the absence of a demand to move forward/backward. The internal structure of the Surge-Brake controller is depicted in Figure 4.16, where the input to the controller is the actual forward speed $u$ and $-\alpha$ ($\alpha > 0$) is the proportional gain. The output of the controller is the $\tau_x$ component of the output vector $\mathbf{\tau}_{LLL}$.

![Figure 4.16 Internal structure of the Surge-Brake controller.](image)

4.5.4 Virtual joystick

The virtual joystick in Figure 4.1 is used to generate the four-component vector $\mathbf{\tau}_{HCU}$. In the ROV operation mode, the vector $\mathbf{\tau}_{HCU}$ is created from the ROV pilot commands using the Hand Control Unit (HCU), i.e. the input device including joystick, knobs and buttons. In AUV operation mode, the vector $\mathbf{\tau}_{HCU}$ is controlled by the Behaviour Coordinator/Arbitration component through the Winner Control Cluster in order to mimic ROV pilot commands (Figure 4.20).

4.5.5 Fuzzy Controller for Obstacle Avoidance

To address the challenges of deploying UUVs in close proximity to the seabed and other hazards, a three-dimensional obstacle avoidance algorithm has been developed. A pragmatic approach for obstacle avoidance in the horizontal plane is that the vehicle to travel along the
side of the obstacle, i.e. to follow the obstacle boundary defining a close proximity path around the obstacle, until the vehicle reaches a position where it can break away and return to course (Caccia, Bruzzone et al., 2000), (Healey, 2004). A similar functionality is proposed for Tethra’s collision avoidance using fuzzy logic control. The obstacle avoidance algorithm employs a fuzzy logic controller and has been tested in the Tethra simulator. Tethra has six Tritech digital precision altimeters for obstacle detection, which can be divided into two groups (Figure 4.17):

**Horizontal altimeters:** right sA, front-left sB, front-right sC, front sE and left sF are mounted on the front side of the vehicle in the horizontal plane with 30 degree spacing, aiming to provide an effective sensing coverage and detection of close proximity obstacles in the front field view.

**Vertical altimeter:** down sD is mounted on the bottom, facing down, and it provides altitude information.
The output of each altimeter is distance (in meters) to the obstacles in the direction of the sensor’s central axis. These distances are used as inputs to the fuzzy controller for obstacle avoidance (FCOA). The outputs of the FCOA are the normalised components $\xi_X$, $\xi_Z$, $\xi_N$ of the virtual joystick, see section 4.5.4. Membership functions (N = ‘Near’, F = ‘Far’) for inputs $sA$, $sB$, $sC$, $sD$, $sE$ & $sF$ are shown in Figure 4.18.

![Figure 4.18 Membership functions for input variables $sA$, $sB$, $sC$, $sD$, $sE$ & $sF$.](image)

The inference system implemented inside the FCOA (see Figure 4.19) is a Sugeno-type with constant output membership functions. The rule table of the FCOA is given in Table 4.9 in compact form. The fuzzy rules are defined in such a way that in the presence of an obstacle the vehicle is rotated and driven along the side the obstacle. If all altimeters indicate close proximity to obstacles, the vehicle is commanded to go up.

Let a fuzzy controller has $r$ inputs and let input $i, i = 1, \ldots, r$ has $n_i$ linguistic terms (membership functions). In that case there are at most $N = n_1 n_2 \ldots n_r$ different rules available to form the rule base. In the case of the FCOA, there are 6 inputs and each input has 2 membership functions. Therefore $N = 2^6 = 64$ is the maximum number of rules for the inference system of the FCOA, using the linguistic terms Near and Far to represent membership functions for the input variables $sA$, $sB$, $sC$, $sD$, $sE$ and $sF$. Another choice of membership functions could consist of using the linguistic terms Close, Near and Far.
However, maximum number of rules in this case would be $N = 3^6 = 729$, making its implementation impractical and time-consuming.

**Figure 4.19** Internal structure of the FCOA
Table 4.9 Fuzzy rule table.

<table>
<thead>
<tr>
<th>#</th>
<th>sA</th>
<th>sB</th>
<th>sC</th>
<th>sD</th>
<th>sE</th>
<th>sF</th>
<th>$\xi_x$</th>
<th>$\xi_y$</th>
<th>$\xi_v$</th>
</tr>
</thead>
<tbody>
<tr>
<td>0</td>
<td>Far</td>
<td>Far</td>
<td>Far</td>
<td>Far</td>
<td>Far</td>
<td>Far</td>
<td>0.00</td>
<td>0.00</td>
<td>0.00</td>
</tr>
<tr>
<td>1</td>
<td>Far</td>
<td>Far</td>
<td>Far</td>
<td>Far</td>
<td>Near</td>
<td>Far</td>
<td>0.50</td>
<td>0.00</td>
<td>0.25</td>
</tr>
<tr>
<td>2</td>
<td>Far</td>
<td>Far</td>
<td>Far</td>
<td>Near</td>
<td>Far</td>
<td>Far</td>
<td>0.50</td>
<td>0.00</td>
<td>-0.35</td>
</tr>
<tr>
<td>3</td>
<td>Far</td>
<td>Far</td>
<td>Near</td>
<td>Far</td>
<td>Far</td>
<td>Far</td>
<td>0.50</td>
<td>0.00</td>
<td>0.00</td>
</tr>
<tr>
<td>4</td>
<td>Far</td>
<td>Far</td>
<td>Near</td>
<td>Far</td>
<td>Far</td>
<td>Far</td>
<td>0.50</td>
<td>0.00</td>
<td>0.00</td>
</tr>
<tr>
<td>5</td>
<td>Far</td>
<td>Far</td>
<td>Near</td>
<td>Far</td>
<td>Far</td>
<td>Far</td>
<td>0.50</td>
<td>0.00</td>
<td>0.25</td>
</tr>
<tr>
<td>6</td>
<td>Far</td>
<td>Far</td>
<td>Near</td>
<td>Far</td>
<td>Far</td>
<td>Far</td>
<td>0.50</td>
<td>0.00</td>
<td>-0.35</td>
</tr>
<tr>
<td>7</td>
<td>Far</td>
<td>Far</td>
<td>Near</td>
<td>Far</td>
<td>Far</td>
<td>Far</td>
<td>0.50</td>
<td>0.00</td>
<td>0.00</td>
</tr>
<tr>
<td>8</td>
<td>Far</td>
<td>Far</td>
<td>Near</td>
<td>Far</td>
<td>Far</td>
<td>Far</td>
<td>0.25</td>
<td>0.00</td>
<td>0.50</td>
</tr>
<tr>
<td>9</td>
<td>Near</td>
<td>Far</td>
<td>Near</td>
<td>Far</td>
<td>Far</td>
<td>Far</td>
<td>0.25</td>
<td>0.00</td>
<td>0.75</td>
</tr>
<tr>
<td>10</td>
<td>Far</td>
<td>Near</td>
<td>Far</td>
<td>Near</td>
<td>Far</td>
<td>Far</td>
<td>0.25</td>
<td>0.00</td>
<td>0.50</td>
</tr>
<tr>
<td>11</td>
<td>Far</td>
<td>Far</td>
<td>Near</td>
<td>Far</td>
<td>Near</td>
<td>Far</td>
<td>0.25</td>
<td>0.00</td>
<td>0.50</td>
</tr>
<tr>
<td>12</td>
<td>Far</td>
<td>Near</td>
<td>Far</td>
<td>Near</td>
<td>Far</td>
<td>Far</td>
<td>0.25</td>
<td>0.00</td>
<td>0.50</td>
</tr>
<tr>
<td>13</td>
<td>Far</td>
<td>Near</td>
<td>Far</td>
<td>Near</td>
<td>Far</td>
<td>Far</td>
<td>0.25</td>
<td>0.00</td>
<td>0.75</td>
</tr>
<tr>
<td>14</td>
<td>Far</td>
<td>Near</td>
<td>Far</td>
<td>Near</td>
<td>Far</td>
<td>Far</td>
<td>0.25</td>
<td>0.00</td>
<td>0.50</td>
</tr>
<tr>
<td>15</td>
<td>Far</td>
<td>Near</td>
<td>Far</td>
<td>Near</td>
<td>Far</td>
<td>Far</td>
<td>0.25</td>
<td>0.00</td>
<td>0.50</td>
</tr>
<tr>
<td>16</td>
<td>Far</td>
<td>Near</td>
<td>Far</td>
<td>Near</td>
<td>Far</td>
<td>Far</td>
<td>0.25</td>
<td>0.00</td>
<td>0.75</td>
</tr>
<tr>
<td>17</td>
<td>Far</td>
<td>Near</td>
<td>Far</td>
<td>Near</td>
<td>Far</td>
<td>Far</td>
<td>0.25</td>
<td>0.00</td>
<td>0.75</td>
</tr>
<tr>
<td>18</td>
<td>Far</td>
<td>Near</td>
<td>Far</td>
<td>Near</td>
<td>Far</td>
<td>Far</td>
<td>0.25</td>
<td>0.00</td>
<td>0.75</td>
</tr>
<tr>
<td>19</td>
<td>Far</td>
<td>Near</td>
<td>Far</td>
<td>Near</td>
<td>Far</td>
<td>Far</td>
<td>0.25</td>
<td>0.00</td>
<td>0.50</td>
</tr>
<tr>
<td>20</td>
<td>Far</td>
<td>Near</td>
<td>Far</td>
<td>Near</td>
<td>Far</td>
<td>Far</td>
<td>0.25</td>
<td>0.00</td>
<td>0.75</td>
</tr>
<tr>
<td>21</td>
<td>Far</td>
<td>Near</td>
<td>Far</td>
<td>Near</td>
<td>Far</td>
<td>Far</td>
<td>0.25</td>
<td>0.00</td>
<td>0.75</td>
</tr>
<tr>
<td>22</td>
<td>Far</td>
<td>Near</td>
<td>Far</td>
<td>Near</td>
<td>Far</td>
<td>Far</td>
<td>0.25</td>
<td>0.00</td>
<td>0.75</td>
</tr>
<tr>
<td>23</td>
<td>Far</td>
<td>Near</td>
<td>Far</td>
<td>Near</td>
<td>Far</td>
<td>Far</td>
<td>0.25</td>
<td>0.00</td>
<td>0.75</td>
</tr>
<tr>
<td>24</td>
<td>Far</td>
<td>Near</td>
<td>Far</td>
<td>Near</td>
<td>Far</td>
<td>Far</td>
<td>0.25</td>
<td>0.00</td>
<td>0.75</td>
</tr>
<tr>
<td>25</td>
<td>Far</td>
<td>Near</td>
<td>Far</td>
<td>Near</td>
<td>Far</td>
<td>Far</td>
<td>0.25</td>
<td>0.00</td>
<td>0.75</td>
</tr>
<tr>
<td>26</td>
<td>Far</td>
<td>Near</td>
<td>Far</td>
<td>Near</td>
<td>Far</td>
<td>Far</td>
<td>0.00</td>
<td>0.00</td>
<td>1.00</td>
</tr>
<tr>
<td>27</td>
<td>Far</td>
<td>Near</td>
<td>Far</td>
<td>Near</td>
<td>Far</td>
<td>Far</td>
<td>0.00</td>
<td>0.00</td>
<td>0.75</td>
</tr>
<tr>
<td>28</td>
<td>Far</td>
<td>Near</td>
<td>Far</td>
<td>Near</td>
<td>Far</td>
<td>Far</td>
<td>0.00</td>
<td>0.00</td>
<td>1.00</td>
</tr>
<tr>
<td>29</td>
<td>Far</td>
<td>Near</td>
<td>Far</td>
<td>Near</td>
<td>Far</td>
<td>Far</td>
<td>0.00</td>
<td>0.00</td>
<td>1.00</td>
</tr>
<tr>
<td>30</td>
<td>Far</td>
<td>Near</td>
<td>Far</td>
<td>Near</td>
<td>Far</td>
<td>Far</td>
<td>0.00</td>
<td>0.00</td>
<td>0.75</td>
</tr>
<tr>
<td>31</td>
<td>Far</td>
<td>Near</td>
<td>Far</td>
<td>Near</td>
<td>Far</td>
<td>Far</td>
<td>0.00</td>
<td>0.00</td>
<td>0.75</td>
</tr>
<tr>
<td>32</td>
<td>Near</td>
<td>Far</td>
<td>Far</td>
<td>Far</td>
<td>Far</td>
<td>Far</td>
<td>0.25</td>
<td>0.00</td>
<td>-0.50</td>
</tr>
<tr>
<td>33</td>
<td>Near</td>
<td>Far</td>
<td>Far</td>
<td>Far</td>
<td>Far</td>
<td>Far</td>
<td>0.25</td>
<td>0.00</td>
<td>-0.50</td>
</tr>
<tr>
<td>34</td>
<td>Near</td>
<td>Far</td>
<td>Far</td>
<td>Far</td>
<td>Far</td>
<td>Far</td>
<td>0.25</td>
<td>0.00</td>
<td>-0.50</td>
</tr>
<tr>
<td>35</td>
<td>Near</td>
<td>Far</td>
<td>Far</td>
<td>Far</td>
<td>Far</td>
<td>Far</td>
<td>0.25</td>
<td>0.00</td>
<td>-0.50</td>
</tr>
<tr>
<td>36</td>
<td>Near</td>
<td>Far</td>
<td>Far</td>
<td>Far</td>
<td>Far</td>
<td>Far</td>
<td>0.25</td>
<td>0.00</td>
<td>-0.50</td>
</tr>
<tr>
<td>37</td>
<td>Near</td>
<td>Far</td>
<td>Far</td>
<td>Far</td>
<td>Far</td>
<td>Far</td>
<td>0.25</td>
<td>0.00</td>
<td>-0.50</td>
</tr>
<tr>
<td>38</td>
<td>Near</td>
<td>Far</td>
<td>Far</td>
<td>Far</td>
<td>Far</td>
<td>Far</td>
<td>0.25</td>
<td>0.00</td>
<td>-0.50</td>
</tr>
<tr>
<td>39</td>
<td>Near</td>
<td>Far</td>
<td>Far</td>
<td>Far</td>
<td>Far</td>
<td>Far</td>
<td>0.25</td>
<td>0.00</td>
<td>-0.50</td>
</tr>
<tr>
<td>40</td>
<td>Near</td>
<td>Far</td>
<td>Far</td>
<td>Far</td>
<td>Far</td>
<td>Far</td>
<td>0.25</td>
<td>0.00</td>
<td>0.00</td>
</tr>
<tr>
<td>41</td>
<td>Near</td>
<td>Far</td>
<td>Far</td>
<td>Far</td>
<td>Far</td>
<td>Far</td>
<td>0.25</td>
<td>0.00</td>
<td>0.00</td>
</tr>
<tr>
<td>42</td>
<td>Near</td>
<td>Far</td>
<td>Far</td>
<td>Far</td>
<td>Far</td>
<td>Far</td>
<td>0.25</td>
<td>0.00</td>
<td>0.00</td>
</tr>
<tr>
<td>43</td>
<td>Near</td>
<td>Far</td>
<td>Far</td>
<td>Far</td>
<td>Far</td>
<td>Far</td>
<td>0.25</td>
<td>0.00</td>
<td>0.00</td>
</tr>
<tr>
<td>44</td>
<td>Near</td>
<td>Far</td>
<td>Far</td>
<td>Far</td>
<td>Far</td>
<td>Far</td>
<td>0.25</td>
<td>0.00</td>
<td>0.00</td>
</tr>
<tr>
<td>45</td>
<td>Near</td>
<td>Far</td>
<td>Far</td>
<td>Far</td>
<td>Far</td>
<td>Far</td>
<td>0.25</td>
<td>0.00</td>
<td>0.00</td>
</tr>
<tr>
<td>46</td>
<td>Near</td>
<td>Far</td>
<td>Near</td>
<td>Far</td>
<td>Near</td>
<td>Far</td>
<td>0.25</td>
<td>0.00</td>
<td>0.00</td>
</tr>
<tr>
<td>47</td>
<td>Near</td>
<td>Far</td>
<td>Near</td>
<td>Far</td>
<td>Near</td>
<td>Far</td>
<td>0.25</td>
<td>0.00</td>
<td>0.00</td>
</tr>
<tr>
<td>48</td>
<td>Near</td>
<td>Far</td>
<td>Near</td>
<td>Far</td>
<td>Near</td>
<td>Far</td>
<td>0.25</td>
<td>0.00</td>
<td>0.00</td>
</tr>
<tr>
<td>49</td>
<td>Near</td>
<td>Far</td>
<td>Near</td>
<td>Far</td>
<td>Near</td>
<td>Far</td>
<td>0.25</td>
<td>0.00</td>
<td>0.00</td>
</tr>
<tr>
<td>50</td>
<td>Near</td>
<td>Far</td>
<td>Near</td>
<td>Far</td>
<td>Near</td>
<td>Far</td>
<td>0.25</td>
<td>0.00</td>
<td>0.00</td>
</tr>
<tr>
<td>51</td>
<td>Near</td>
<td>Far</td>
<td>Near</td>
<td>Far</td>
<td>Near</td>
<td>Far</td>
<td>0.25</td>
<td>0.00</td>
<td>0.00</td>
</tr>
<tr>
<td>52</td>
<td>Near</td>
<td>Far</td>
<td>Near</td>
<td>Far</td>
<td>Near</td>
<td>Far</td>
<td>0.25</td>
<td>0.00</td>
<td>0.00</td>
</tr>
<tr>
<td>53</td>
<td>Near</td>
<td>Far</td>
<td>Near</td>
<td>Far</td>
<td>Near</td>
<td>Far</td>
<td>0.25</td>
<td>0.00</td>
<td>0.00</td>
</tr>
<tr>
<td>54</td>
<td>Near</td>
<td>Far</td>
<td>Near</td>
<td>Far</td>
<td>Near</td>
<td>Far</td>
<td>0.25</td>
<td>0.00</td>
<td>0.00</td>
</tr>
<tr>
<td>55</td>
<td>Near</td>
<td>Far</td>
<td>Near</td>
<td>Far</td>
<td>Near</td>
<td>Far</td>
<td>0.25</td>
<td>0.00</td>
<td>0.00</td>
</tr>
<tr>
<td>56</td>
<td>Near</td>
<td>Far</td>
<td>Near</td>
<td>Far</td>
<td>Near</td>
<td>Far</td>
<td>0.25</td>
<td>0.00</td>
<td>0.00</td>
</tr>
<tr>
<td>57</td>
<td>Near</td>
<td>Far</td>
<td>Near</td>
<td>Far</td>
<td>Near</td>
<td>Far</td>
<td>0.25</td>
<td>0.00</td>
<td>0.00</td>
</tr>
<tr>
<td>58</td>
<td>Near</td>
<td>Far</td>
<td>Near</td>
<td>Far</td>
<td>Near</td>
<td>Far</td>
<td>0.25</td>
<td>0.00</td>
<td>0.00</td>
</tr>
<tr>
<td>59</td>
<td>Near</td>
<td>Far</td>
<td>Near</td>
<td>Far</td>
<td>Near</td>
<td>Far</td>
<td>0.25</td>
<td>0.00</td>
<td>0.00</td>
</tr>
<tr>
<td>60</td>
<td>Near</td>
<td>Far</td>
<td>Near</td>
<td>Far</td>
<td>Near</td>
<td>Far</td>
<td>0.25</td>
<td>0.00</td>
<td>0.00</td>
</tr>
<tr>
<td>61</td>
<td>Near</td>
<td>Far</td>
<td>Near</td>
<td>Far</td>
<td>Near</td>
<td>Far</td>
<td>0.25</td>
<td>0.00</td>
<td>0.00</td>
</tr>
<tr>
<td>62</td>
<td>Near</td>
<td>Far</td>
<td>Near</td>
<td>Far</td>
<td>Near</td>
<td>Far</td>
<td>0.25</td>
<td>0.00</td>
<td>0.00</td>
</tr>
<tr>
<td>63</td>
<td>Near</td>
<td>Far</td>
<td>Near</td>
<td>Far</td>
<td>Near</td>
<td>Far</td>
<td>0.25</td>
<td>0.00</td>
<td>0.00</td>
</tr>
</tbody>
</table>
4.5.6 **Behaviour Co-ordinator/Arbitration**

Currently, two high-level task executors have been implemented: AUV Tracking (for way-point guidance) and AUV Obstacle Avoidance (Figure 4.20). Each of these task executors is competing to take control of actuators. The control buffer concept has been developed to provide transparency and easy fusion of different task executor control demands. Each task executor has its own control cluster and mask inside the control buffer (Figure 4.21). The control cluster consists of Hand Control Unit (HCU) components (to simulate a virtual joystick), a set of settings for low-level controllers (set points and on/off switches to activate/deactivate individual controllers) and a priority value, reserved for future use in competitive arbitration within the reactive layer.

![Figure 4.20 Behaviour co-ordinator/Arbitration.](image)
Figure 4.21 Winner Control Cluster.

Each control cluster is masked with a corresponding mask, depending on the state of the mission and navigation data (interaction with the real world). A mask is built from weights and logic gates, which are bundled into the same structure as the control cluster. Masking is performed by multiplying (ANDing) corresponding fields in the control cluster and mask (see Figure 4.22).

Figure 4.22 Example of competitive arbitration

The priority level of the task executor is determined by the mask content (Figure 4.22). In this way it is possible to control the degree of co-operation and competition between different
task executors. After masking, control clusters are bundled into the Winner Control Cluster, which has exclusive actuator control.

### 4.6 Concluding remarks

A novel guidance, control and navigation system for Tethra AUV is presented in this chapter. The main focus has been given to description of the control system in this chapter, while navigation and guidance systems are described in chapters 3 and 5, respectively.

The guidance system continuously computes and updates the reference (desired) position, velocity and acceleration, which are forwarded to the control system. The navigation system is used to determine the actual position, velocity, acceleration and attitude (orientation) of the vehicle. These signals are used as feedback signals to the control system, which is used to determine the necessary control forces and moments to be applied to drive the vehicle in order to satisfy the control objective and perform mission tasks. It utilises the hybrid control architecture, introduced in section 2.3, which is modified to accommodate the thruster configuration of Tethra. The main components of the control system include: Control Allocation, Virtual Joystick, Low-level controllers, Fuzzy controller for Obstacle Avoidance and Behaviour Coordinator/Arbitration.

The control problem is decomposed into two subtasks: a Regulation Task (which determines the necessary forces and moments to satisfy the control objective) and an Actuator Selection Task (which maps the total control effort (forces and moments) onto individual actuators, i.e. input voltages to be applied to drive thrusters). The later task is performed by the Control Allocation component. The control allocation problem for Tethra AUV has been fully described and solved, including a geometrical interpretation and visualisation of the solution.
A set of low-level controllers includes an Auto-Heave Controller (to maintain/track desired depth/altitude), an Auto-Heading (to maintain desired heading) and a Surge-Brake Controller (to stop surge motion in the absence of a demand to move forward/backward, like a mechanical damper). The Auto-Heave controller is enhanced with the feedforward control path in order to cancel tracking error for time-varying depth/altitude input trajectories.

The virtual joystick has been introduced to mimic ROV pilot commands. In the ROV operation mode, the virtual joystick is driven by an ROV pilot. In the AUV operation mode, it is controlled by the Behaviour Coordinator/Arbitration component.

A three-dimensional algorithm for obstacle avoidance has been presented in this chapter. It utilises a Sugeno-type Fuzzy logic controller with 6 inputs, 3 outputs and 64 rules. Inputs are provided from altimeters mounted on the vehicle.

The Behaviour coordinator/Arbitration component is used to realise the Regulation subtask mentioned above. Currently, two task executors have been implemented: AUV Tracking (for way-point guidance) and AUV Obstacle Avoidance. Each of these task executors compete to take control of actuators. The outputs of task executors are integrated using masking and prioritisation, and bundled into the Winner Control Cluster, which has an exclusive actuator control. In this way, the Behaviour coordinator/Arbitration component converts the problem of the realisation of mission tasks to the form required by the Control Allocation component.

Evaluation of the proposed guidance, navigation and control system is performed in chapter 6, where a number of representative test cases demonstrate key features & performance of the proposed system.

The main contribution of this chapter are summarised in the following:
Chapter 4: Guidance, Navigation and Control for Tethra AUV

- Decomposition of control problem for UUVs into regulation and actuator selection tasks.
- Full implementation of the hybrid control architecture with safety critical behaviours with highest priority.
- Formulation of the control allocation problem in normalised form and its solution, including clear geometrical interpretation.
- Introducing the control buffer concept with clusters for virtual joystick and low-level controller clusters.
- Development of Fuzzy Logic controller for 3D obstacle avoidance.
Chapter 5: Way-Point Guidance System

5.1 Introduction

As mentioned in section 4.1, the main role of the guidance system is to continuously provide the reference position, velocity and acceleration of a vessel to the control system. The main features of the proposed guidance system are described in this chapter.

This chapter is organised as follows: the concept of way-points is introduced in section 5.2. The mission planner and tracking way-points planner are described in sections 5.3. and 5.4, respectively. “Rotate toward Target”, “Move to Target”, “Missed Way-Point Handling” and “Transition to new Transect” strategies are described in sections 5.5, 5.6, 5.7 and 5.8, respectively. Vertical behaviours (strategies for motion in the vertical plane) are described in section 5.9. Finally, concluding remarks are summarised in section 5.10.

5.2 Way-Points

In order to carry out a specific mission with a submersible vehicle, such as multi-beam or side-scan sonar survey at a constant depth or underwater filming close to the seabed at a constant altitude, transect following capabilities are required. This functionality is achieved with a way-point guidance algorithm, described in the following. The route of an underwater vehicle is defined in terms of way-points. The way-point guidance algorithm has been implemented as a higher level planner/sequencer; acting as the deliberative layer in the overall control architecture, transforming the mission into a set of tasks. Before the mission is started, a way-point database is created using a way-point generator with human input. Each way-point is defined using Cartesian coordinates \((x_k, y_k, z_k)\). In addition, other way-point
properties like desired speed \((U_k)\) and radius of acceptance \((R_k)\) can be defined. This means that the vehicle should pass through the way-point \((x_k, y_k, z_k)\) at forward speed \((U_k)\) and the way-point is considered to be reached when the centre of gravity of the vehicle enters the sphere with centre at \((x_k, y_k, z_k)\) and radius \((R_k)\).

### 5.3 Mission Planner

The main task of the mission planner is the decomposition of the mission into a set of tasks. A typical AUV survey mission includes the following stages (Gaiffe, 2002):

1. **PHINS Rough Alignment**: 5 minutes approx.; inertial sensor data are used to estimate heading, roll and pitch angles; available aiding sensor: GPS.
2. **PHINS Fine Alignment**: 20 minutes approx.; Kalman filter is activated to compute and estimate position and speed with optimal accuracy; available aiding sensor: GPS.
3. **Diving**: until desired depth/altitude is reached; available aiding sensors: USBL, DVL, depth, altimeter.
4. **Calibration Turn**: PHINS calibration with DVL; available aiding sensors: USBL, DVL, depth, altimeter.
5. **Tracking Way-Points**: trajectory tracking – actual seabed survey; available aiding sensors: USBL, DVL, depth, altimeter.
6. **Back to Surface**: AUV is going back to surface after mission is completed or aborted; available aiding sensors: USBL, DVL, depth, altimeter.

The mission planner has been realised as a state machine in the Control Centre (Figure 5.1). The control execution layer supervises the activity of the lower-level reactive layer, making assessment over the current situation, and based on external conditions or in-state calculation,
decides which state needs to be executed next. Each state in the machine represents a stage of a typical AUV mission. States are executed in a sequential order. The output of each state is a high-level task, which is transmitted to the Behaviour Co-ordinator/Arbitration component (Figure 4.20).

![Mission planner as a state machine.](image)

**Figure 5.1** Mission planner as a state machine.

The emergency task (activated by the self-diagnostic and vehicle safety check module) has the highest priority and its activation changes the order of execution of the state machine, aborting the mission and sending the vehicle back to the surface (Toal, Flanagan *et al.*, 2002).

### 5.4 Tracking Way-Points Planner

Actual way-point guidance is performed inside the Tracking Way-Points state. The main task of the tracking way-points planner is the decomposition of the way-point guidance algorithm
into a set of tasks. The tracking way-points planner is realised as a state machine (Figure 5.2) whose parent is the Tracking Way-Points super-state in Figure 5.1. Depending on the predetermined tracking mode (constant depth or constant altitude), the corresponding operation mode of the Auto-Heave low-level controller (Depth or Altitude) is permanently activated in the Tracking control cluster during the way-point guidance.

**Figure 5.2** Tracking way-points planner as a state machine.

**Figure 5.3** Passing through a way-point during simulation.
Two basic low-level behaviours (“Rotate towards Target” and “Move to Target”) are executed in the loop, until the tracking is finished or the mission is aborted. For example, the behaviour “Rotate towards Target” is executed when the previous target (way-point $WP_k$) has been reached (see Figure 5.3) and the AUV needs to rotate in the direction of the new target (next way-point $WP_{k+1}$).

5.5 “Rotate towards Target” strategies

Two different strategies for the “Rotate towards Target” behaviour are described in the following.

5.5.1 Strategy 1: “Rotate towards Target with Surge-Brake”

Activate both the Surge-Brake Controller (brake effect) and the Auto-Heading Controller with set-point $\psi_d = \alpha_k$. This strategy is energy-consuming, since a certain amount of energy is wasted in order to change the rotation direction of propellers. The rate of change is limited by a slew rate limiter inside the thruster control loop, to smooth out the abrupt change and limit the current inside the thrusters motor. The strategy is appropriate if the survey requires as precise as possible following of transects between each way-point.

5.5.2 Strategy 2: “Rotate towards Target without Surge-Brake”

Activate only Auto-Heading Controller with set-point $\psi_d = \alpha_k$. This strategy is potentially more efficient than Strategy 1 from the energy point of view, since there is no need to change the rotation direction of the propellers. However, the price that is paid is the slightly increased tracking error during rotation. The strategy may be preferred in lawn mower survey
patterns where transect following at the end/start of segments is not so important, once transect following along survey lines is good.

5.6 “Move to Target” strategies

The behaviour “Move to Target” is executed next, after “Rotate towards Target” is finished:

The Tracking problem during the “Move to Target” behaviour is decomposed into two subtasks:

1. **Subtask 1:** Following the desired heading with the desired forward speed;

2. **Subtask 2:** Bringing the vehicle onto the desired path, (Figure 5.4).

![Figure 5.4 Way-point guidance parameters.](image)

Three different strategies for the “Move to Target” behaviour are described in the following.

5.6.1 Strategy 1: “Separate Auto-Heading & Cross-Track Compensation”

The HCU components (virtual joystick) of the Tracking Control Cluster have the following settings:

\[ \bar{\mathbf{r}}_x = f(U_k, U_{k+1}, p) \quad \bar{\mathbf{r}}_z = \bar{\mathbf{r}}_N = g(d) \]  

(5.1)

where \( U_k \) and \( U_{k+1} \) are normalised desired speeds at way-points \( WP_k \) and \( WP_{k+1} \) respectively, \( p \) is the distance between the vehicle and \( WP_{k+1} \) and \( d \) is the cross-track error. The cross-track error (normal distance from the path) is defined as
Chapter 5: Way-Point Guidance System

\[
d = \frac{\begin{bmatrix} x_{k+1} - x_k & y_{k+1} - y_k \\ x - x_k & y - y_k \end{bmatrix}}{\begin{bmatrix} x_{k+1} - x_k \\ y_{k+1} - y_k \end{bmatrix}}
\]  
(5.2)

The sign of \(d\) is related to the position of the vehicle relative to the path: \(d\) is positive (negative) if the vehicle is located on the left (right) side of the path.

The component \(\tau_x\) is used to perform the Subtask 1. In the ideal case Subtask 2 could be achieved by appropriate lateral action (controlling the sway DOF by sway force \(\tau_y\)). Since the sway DOF is not directly controllable by the current thrusters configuration of Tethra, Subtask 2 is performed by generating a heading corrective component, \(\tau_N\), whose role is to change the heading of the vehicle in order to reduce offset from the desired path (cross-track error). Since the surge force is directly related to the forward speed, one choice for function \(f\) in (5.1) is a linear interpolation, as follows:

\[
f = \frac{p}{\|WP_{k+1} - WP_k\|} (U_k - U_{k+1}) + U_{k+1}
\]  
(5.3)

Due to the symmetry of the problem and sign convention, the function \(g\) in (5.1) must be an odd function with respect to \(d\), with values in the first and the third quadrant. One choice for \(g\) is to use the tangent hyperbolic function (see Figure 5.5).

![Figure 5.5 Different shapes of the tangent hyperbolic function.](image)

139
The Surge-Brake Controller is disabled, while the Auto-Heading Controller is active with the set-point equal to the line-of-sight (LOS) heading:

\[ \tan(\psi_{\text{LOS}}) = \frac{y_{k+1} - y}{x_{k+1} - x} \]  (5.4)

### 5.6.2 Strategy 2: “Auto-Heading Integrated Cross-Track Compensation”

An alternative strategy is to compensate for the cross-track error using only the Auto-Heading controller. The HCU components (virtual joystick) of the Tracking Control Cluster have the following settings:

\[ \tau_x = f(U_k, U_{k+1}, p) \quad \tau_z = \tau_K = \tau_N = 0 \]  (5.5)

The Auto-Heading Controller is active with the set-point equal to

\[ \psi_r = \psi_{\text{LOS}}^0 + \psi_d \]  (5.6)

The initial (constant) line-of-sight heading \( \psi_{\text{LOS}}^0 \) in (5.6) is calculated only once, at the beginning of the stage. In order to reduce the cross-track error a variable heading component \( \psi_d \) is added to the initial line-of-sight heading \( \psi_{\text{LOS}}^0 \). The variable heading component is calculated from

\[ \psi_d = A_2 \tanh(W_2d) \]  (5.7)

where \( A_2 \) and \( W_2 \) are tuneable parameters.

### 5.6.3 Strategy 3: “Auto-Heading with Look-Ahead” for smooth way-point transition

This strategy is based on an approach for smooth way-point transition proposed in (Bakaric, et al., 2004) and described in section 2.4.3. The HCU components (virtual joystick) of the
Chapter 5: Way-Point Guidance System

Tracking Control Cluster are given by (5.5) while the reference heading is 0, and by (5.5) during smooth way-point transition.

The Auto-Heading Controller is active with the reference heading found from (2.31). Simulation results obtained with the following settings:

- Move to Target strategy = “Auto-Heading with Look-Ahead”,
- Rotate towards Target strategy = “Rotate towards Target without Surge-Brake”,

are shown in Figure 5.6 for two different sizes of the virtual pool.

![Figure 5.6](image)

**Figure 5.6** Trajectories obtained with “Auto-Heading with Look-Ahead” Move to Target strategy and “Rotate towards Target without Surge-Brake” Rotate towards Target strategy for two different sizes of the virtual pool (scaling factor 5 (left) and 15 (right)).

In Figure 5.6 (left), lawn mower transects are purposefully short to show the effect of the look-ahead strategy. As transect length increases Figure 5.6 (right), the section of transect with increased cross-track errors induced by the look-ahead strategy, becomes relatively shorter.

### 5.7 “Missed Way-Point Handling” strategies

Due to obstacle avoidance manoeuvres and/or the influence of external disturbances, the vehicle may miss target way-points. Two strategies have been developed for missed way-point handling, as described in the following.
5.7.1 Strategy 1: “Range Exceeded – Missed Way-point”

If the vehicle is located inside the circle with centre at $WP_k$ and radius $\rho_k$ (Figure 5.7), the heading corrective action is applied and the vehicle tries to reach the target while minimizing the cross-track error. Otherwise, if the vehicle is outside the circle and still has not reached the target, the heading corrective action is abandoned and the vehicle attempts to reach the target following the shortest path.

![Figure 5.7 Missed waypoint handling criteria.](image)

5.7.2 Strategy 2: “Target Reacquisition after Obstacle – Acceptance Cone”

Another strategy for missed way-point handling is to define the acceptance cone, as indicated in Figure 5.8. If the vehicle is located inside the acceptance cone after it completes an obstacle avoidance manoeuvre, the heading corrective action is applied and the vehicle attempts to reach the target while minimizing the cross-track error. Otherwise, the vehicle will abandon cross-track error minimization and approach the target following the shortest path.
5.8 “Transition to new Transect” strategies

It is a common requirement that the desired trajectory is composed of parallel segments (transects) and the vehicle is required to track these segments with constant speed and high accuracy. However, during transition between segments, size of the cross-track error is not important and the only requirement is that a vehicle smoothly moves to the beginning of the new segment. For this reason, the passage from one transect to the next can be realised with a circular arc trajectory. Such a strategy would be typical for surveys with a surface vessel and a similar approach can be realised for a UUV. A typical trajectory in case of segment tracking is depicted in Figure 5.9.

Figure 5.8 Missed waypoint handling based on the concept of acceptance cone.

Figure 5.9 Typical trajectory for transect (segment) tracking.
Chapter 5: Way-Point Guidance System

The “Transition to new Transect” strategies take into account the position of the next way-point \( WP_k \) before the current (target) way-point \( WP_{k-1} \) is reached. A flag is associated with each \( WP_{k-1} \) way-point from the user built database prior to the mission, indicating whether the segment \( WP_{k-2}WP_k \) represents a transect (the flag is lowered), or the vehicle can transit from achieved way-point \( WP_{k-1} \) to \( WP_k \) following a circular arc trajectory (the flag is raised).

In the current implementation of the transition to new transect routine, this flag is raised (or lowered) if the following condition is satisfied for the segment lengths:

\[
WP_{k-2}WP_k < WP_{k-2}WP_{k-1} \quad \text{(raised) \quad (5.8)}
\]

\[
WP_{k-2}WP_k > WP_{k-2}WP_{k-1} \quad \text{(lowered). \quad (5.9)}
\]

5.8.1 Strategy 1: “Transition to new Transect with Look-Ahead”

This scheme for “Transition to new Transect with Look-Ahead” behaviour is based on the strategy described earlier in section 5.6.3, the “Auto-Heading with Look-Ahead” for smooth way-point transition strategy developed initially for the “Move to Target” behaviour.

![Figure 5.10 Definition of the auxiliary point \( A_k \) and normalised distance difference \( \varepsilon_k \).](image-url)
5.8.2 Strategy 2: “Transition to New Transect using the Aiding Way-Points” (Bezier curve profiling)

The second “Transition to new Transect” strategy uses the cubic spline interpolation to define additional way-points and to obtain their coordinates along the curve.

A cubic Bezier curve is defined by four points (Figure 5.13):

- Start Point \((x_0, y_0)\)
- Control Point 1 \((x_1, y_1)\)
- Control Point 2 \((x_2, y_2)\)
A set of two equations define the coordinates of the points on the curve:

\[
x(t) = a_xt^3 + b_xt^2 + c_xt + x_0 \\
y(t) = a_yt^3 + b_yt^2 + c_yt + y_0
\]  
(5.10)

If the Start Point, End Point and the two Control Points are known, the following set of equations is used to obtain the coefficient values:

\[
\begin{align*}
  c_x &= 3(x_2 - x_0) \\
  b_x &= 3(x_2 - x_1) - c_x \\
  a_x &= x_3 - x_0 - c_x - b_x \\
  c_y &= 3(y_1 - y_0) \\
  b_y &= 3(y_2 - y_1) - c_y \\
  a_y &= y_3 - y_0 - c_y - b_y
\end{align*}
\]  
(5.11)  

(5.12)

**Figure 5.13** Definition of the aiding way-points.

Both equations (5.11) & (5.12) are evaluated for
where $n$ represents the number of Bezier points along the curve.

An example trajectory using the “Transition to New Transect using the Aiding Way-Points” (Bezier curve profiling) strategy is shown in Figure 5.14.

\[ t_i \in [0,1], \quad i = 0, \ldots, n - 1 \]  
(5.13)

\[ \text{Virtual Pool - Top view} \]

**Figure 5.14** Transition to new transect using Bezier curve profiling.

### 5.9 Vertical behaviours

Three strategies have been developed for motion in the vertical plane. These strategies are described in the following.

#### 5.9.1 Strategy 1: “Planar way-point guidance”

This strategy covers the case when all way-points are at the same constant depth. In this case, the Auto-Heave controller described in section 4.5.3 is used in “depth” operation mode with input (set-point) equal to the desired constant depth of way-points, and the control problem becomes a set-point tracking (depth-keeping) problem. Different strategies of the “Move to
Target” and “Rotate towards Target” behaviours, carried out with constant depth-keeping requirements and presented in chapter 6, illustrate this case.

5.9.2 Three-dimensional way-point guidance

The planar line-of-sight way-point guidance, described previously, assumes constant depth input to the Auto-Depth controller. In this section the algorithm is extended to cover the three-dimensional case to cope with the effects of depth/altitude changes. Way points are distributed in the space with different depth coordinates and the desired path is composed of straight segments connecting these points.

A typical UUV survey mission for benthic habitat mapping may require side-scan/multi-beam sonar surveying with occasional video triangle transects for ground truthing. A way-point guidance scheme with full three-dimensional manoeuvring capabilities can enable the following UUV scenarios:

1. Side-scan/multi-beam sonar surveying at a constant depth along a transect defined by two way-points with a combined video survey triangle to be carried out at a constant altitude, orthogonal to the transect when the target waypoint has been reached.

2. Side-scan/multi-beam sonar surveying such that some transects are planned at different depths for a larger (or smaller) swath coverage.

Strategy 1: “Staircase profile”

The main idea of this strategy is described in the following: if depth coordinates \( z_k \) and \( z_{k+1} \) of two consecutive way-points \( WP_k \) and \( WP_{k+1} \) are different, then the first objective of this strategy is to reach desired depth \( z_{k+1} \) by vertical motion of the vehicle. After this stage is
completed, the Move to Target behaviour is activated and target is reached using standard behaviours for motion in the horizontal plane.

Figure 5.15 Two cases for staircase profile: \( z_{k+1} > z_k \) (left), \( z_{k+1} < z_k \) (right). In the first stage an auxiliary point \( WP_k' \) is reached; then, standard behaviours for motion in the horizontal plane are used to reach the target point \( WP_{k+1} \).

Strategy 2: “3D way-point guidance by the shortest path”

The vehicle will follow the shortest path in an attempt to fulfil the minimum tracking error criteria. To achieve this, the vertical distance \( d \) between the vehicle and the path segment in the vertical plane is continuously calculated as it moves to the target way-point. This distance is used as a set-point (reference) for the Auto-Heave controller.

Figure 5.16 3D way-point guidance by the shortest path.

The normal \( N \) to the vertical plane is given by
The equation of the plane is given by

\[ \mathbf{N} \cdot (\mathbf{S} - \mathbf{T}) = 0 \]  

(5.15)

where \( \mathbf{S}(x, y, z) \) is the actual position of the vehicle, \( \mathbf{T}(x', y', z') \) is the intersection point of the desired path with the vertical plane, \( \mathbf{WP}_k(x_k, y_k, z_k) \) is the last reached way-point and \( \mathbf{WP}_{k+1}(x_{k+1}, y_{k+1}, z_{k+1}) \) is the next (target) way-point.

The equation of the line passing through \( \mathbf{WP}_k \) and \( \mathbf{WP}_{k+1} \) is given by

\[ \mathbf{T} = \mathbf{WP}_k + u(\mathbf{WP}_{k+1} - \mathbf{WP}_k) \]  

(5.16)

where \( u \) is a scalar parameter. Combining (5.16) and (5.15) yields

\[ u = \frac{\mathbf{N} \cdot (\mathbf{S} - \mathbf{WP}_k)}{\mathbf{N} \cdot (\mathbf{WP}_{k+1} - \mathbf{WP}_k)} \]  

(5.17)

Replacing \( u \) from (5.17) into (5.16) yields

\[ \mathbf{T} = \mathbf{WP}_k + \frac{\mathbf{N} \cdot (\mathbf{S} - \mathbf{WP}_k)}{\mathbf{N} \cdot (\mathbf{WP}_{k+1} - \mathbf{WP}_k)}(\mathbf{WP}_{k+1} - \mathbf{WP}_k) \]  

(5.18)

The mode of operation (standard or enhanced, see Figure 4.13, page 120) of the Auto-Heave controller is controlled by the state of the switch \( S \). The input to the feedforward control path is given by (4.32). Two cases corresponding to two different states of the switch \( S \) are indicated in Figure 5.17. When the switch \( S \) is open, the feedforward path is not active and the depth controller is a standard PD controller for the set-point tracking problem. However, since the set-point in this case is time varying signal, tracking error is noticeable during the manoeuvre. When the switch \( S \) is closed, the feedforward path becomes active and the depth
controller becomes an enhanced PD controller for the trajectory tracking problem, resulting in cancellation of the tracking error.

Comparison of the responses with and without the feedforward path included (Figure 5.17) gives a clear indication of the speeding up of the response due to the anticipatory action provided by the feedforward control (Marshall, 1978).

![Figure 5.17 Performance of the Auto-Heave controller operating in depth-tracking mode when the switch S is open (left) and closed (right). Feedforward compensation is not active in the former case and tracking error is noticeable during tracking. Activation of feedforward compensation leads to cancellation of the tracking error.](image)

The constant altitude keeping is obtained by choosing the altitude mode of the Auto-Heave controller and setting the desired altitude as set-point (reference) for the controller.

Trajectories obtained using various 3D way-point guidance scenarios are displayed in Figure 5.18.
Figure 5.18 Various 3D way-point guidance scenarios: (a) constant altitude-keeping; (b) staircase profiling; (c) depth-tracking without feedforward compensation (switch $S$ open); (d) depth-tracking with feedforward compensation (switch $S$ closed).

5.10 Concluding Remarks

The aim of this chapter was to describe the main features of the way-points guidance system, which represents one of the main components of the proposed GNC system for the Tethra AUV, presented in chapter 4.

The way-points guidance system has been implemented through the mission planner and tracking way-points planner, which are realised by means of state machines. The mission planner includes typical AUV survey mission stages. The tracking way-points planner is realised as a state machine whose parent is the tracking way-points super-state in the mission
planner state machine. Two basic low-level behaviours (“Rotate towards Target” and “Move to Target”) are executed in the loop, until the tracking is finished or the mission is aborted. Depending on criteria to be optimised, a number of strategies are proposed for these basic behaviours. This basic set of behaviours is extended to handle missed way-points & smooth transition between transects. The feedforward control scheme is proposed to provide very accurate trajectory tracking in 3D space.

In planning a mission, different combinations of the strategies may be chosen, depending on the priority of the mission objectives. For example, the “Separate Auto-Heading & Cross-Track Compensation” move to target strategy combined with the “Rotate towards Target without Surge Brake” and “Look-Ahead Transition to New Transect” strategy could be chosen for constant depth lawn mower survey requiring good trajectory following. The “Separate Auto-Heading & Cross-Track Compensation” move to target strategy combined with the “Rotate towards Target with Surge Brake” strategy could be useful for constant altitude video surveying. Chapter 6 gives experimental / simulation results, for different combinations of strategies.

To summarise, the main contribution of this chapter are:

- Realisation of mission & tracking way-point planners as state machines.
- Fusion of task executors using arbitration.
Chapter 6: Testing & Evaluation

6.1 Introduction

A novel GNC system for the Tethra AUV has been presented in chapters 4 and 5. The performance of the proposed system is evaluated in this chapter through both real-world applications and computer simulations. Real-world tests have been performed in a diving test pool, swimming pool and Lough Derg Lake. A real-time simulator has been developed in Matlab/LabVIEW environment (Appendix A). A number of representative test cases are used to evaluate the performance of the proposed GNC system in different scenarios.

This chapter is organised as follows: Sections 6.2 and 6.3 present results of real-world tests with the initial and new Tethra design, respectively. Simulation results, presented in section 6.4, evaluate the performance of the GNC system through the series of test cases. Comparative analysis from the energy efficiency point of view is given in section 6.5. finally, concluding remarks are given in section 6.6.

6.2 Real-world tests (Initial design)

A number of real-world tests have been performed in the diving pool in order to test the initial design of Tethra. Proportional plus Integral controllers have been developed for heading, depth and altitude control. With these simple controllers, “Maintain Heading”, “Maintain Depth” and “Maintain Altitude” behaviours have been developed and tested with the stationary vehicle.

Rigorous testing of these behaviours in transit over distance to assess the performance in transect following e.g. with the craft maintaining altitude and heading while moving over the ground is not possible with the size of the test pool, other than short over and back motions.
Using a Proportional-Integral controller with an Anti-windup scheme, a heading-keeping controller has been developed and tested for the stationary Tethra. As illustrated in Figure 6.1, the anti-windup scheme provides a negative feedback to the input of the integrator from the difference between the actual actuator output and the upper saturation limit in order to prevent the motor commands from saturating. An additional anti-windup scheme is used to reduce the integral part, for a heading error less than a certain minimum limit, in order to keep only proportional control for small heading errors. The number of overshoots, experienced in a previous experimentation without this measure included, was significantly reduced.

![Figure 6.1 Functionally equivalent Proportional-Integral Controller with Anti-windup techniques implemented on Tethra (Initial design) for heading-keeping.](image)
Chapter 6: Testing & Evaluation

Figure 6.2 Testing of the simple Proportional-Integral controller with anti-windup techniques for heading control of the stationary Tethra.

In order to test robustness and stability of the heading control scheme, a series of disturbances were applied to Tethra using a rod to rotate it from the stationary position. The heading controller performed well in all cases, bringing the vehicle quickly back to the original position, as indicated in Figure 6.3.

Figure 6.3 Testing of robustness and stability: After external disturbance has been applied, the heading controller returned Tethra back to the initial position.

In the case of the downward pointing altimeter sonar, with a minimum range of 1 m, an interesting test result came into light. During testing of the “Maintain Altitude” behaviour the
expected results were realised for altitudes above 1 m. When the vehicle was disturbed by an external force (e.g. pushed downwards with a stick) the behaviour restored the vehicle to the desired altitude. However, for situations where the altitude set point was close to the minimum 1m range reading for the sonar off the bottom, an undesirable result was obtained in some tests. If the vehicle was disturbed downwards below the 1m minimum range, the sensor gave spurious noisy output. The effect of these noisy readings was that the behaviour drove the vehicle downwards, since the average altimeter sensor output was greater than the set point reading. Hence, special attention must be devoted to cover cases when the output values of altimeters are not reliable, i.e. when the actual altitudes are close to the minimum range.

A simple test has been performed with both an obstacle avoidance and cruise behaviour active. Tethra was programmed to maintain speed with heading perpendicular to the pool sides and simple reverse thrust behaviour was programmed when the forward looking sonars came within 1 m of the pool sides (min range for the higher frequency horizontal pointing sonars is 0.1 m). The thrusters were driven with 20% PWM duty cycle. The vehicle went to and from between the sides until the lateral accumulated drift brought the vehicle close to the pool end. On detection of the pool edge within 1m and reversing the thrusters the vehicle progressed a further 0.5 m approximately before forward motion was arrested.

The first version of the active ballast system was also successfully tested in the diving pool. Tethra was submerged to the depth of 3 m and recovered to the surface using the buoyancy control system, with vertical thrusters disabled. (see Figure 6.4).
6.3 Real-world tests (New design)

Preliminary tests of the new design of Tethra, including the state-of-the-art toolskid navigation sensors (Phins INS, RDI DVL and Microbath depth/altitude sensor), were performed in 50 m long pool in the Arena Sport Centre at the University of Limerick and also at nearby Lough Derg. Tethra was successfully deployed and basic behaviours were tested during these tests. Unfortunately, with limited time slots due to high costs for 50m pool and lake boat hire, mobilisation difficulties (which on later investigation proved to be a simple system interconnect fault), launch and recovery effort and limited overall resources, navigation and control data were not recorded in these tests. Data relevant to system identification were not collected during these tests. However, full scale real-world tests, including system identification, tuning of low-level controllers and evaluation of proposed GNC system will be performed in future work, as mentioned in section 7.5. Tethra operation in the pool and on the lake has been captured on video both under radio control and ROV mode control. Some pictures taken during these tests are shown in Figure 6.5 and Figure 6.6.
6.4 Simulation results

Performance of the overall control scheme has been investigated in the Tethra real-time simulation environment, both in the absence and presence of external disturbances (currents). A selection of simulation results is presented for a simple way-point guidance example with eight way-points. In these experiments, normalised speed is set to 0.2 and radius of acceptance is set to 1.5 m for each way-point. In order to provide more space for testing of different behaviours, a pool scaling factor 5 is used in simulations for the majority of the test cases, i.e. the size of the virtual pool is five time bigger than the size of the real pool. However, in real-world applications the segment lengths would be much longer.
Chapter 6: Testing & Evaluation

In order to improve the clarity and readability, the same page layout is used to present the results for all test cases. Short description of the layout diagrams is given in the following:

- Diagrams (a) and (b) display the pool & trajectories from different view points. In particular, a top view is given in diagram (a), while a 3D side view is shown in diagram (b).

- Diagrams (c) – (h) display time history of signals (inputs, outputs and states) corresponding to the part of the trajectory between points A and B. In order to get clear and readable time diagrams, only this part of the trajectory is selected to be displayed. In particular, diagram (c) displays normalised, desired and actual forces and moments for motion in the horizontal plane (upper part) and the vertical plane (lower part). Diagram (d) displays normalised desired and actual propeller angular velocities for motion in the horizontal plane (upper part) and the vertical plane (lower part). Components of the linear velocity vector (and its norm) are shown in the upper (lower) part of diagram (e), respectively.

- Diagram (f) displays roll and pitch over time.

- The upper part of diagram (g) displays components of the error vector, while its norm is shown in the middle part of the same diagram. The lower part displays stage indicators, showing which stage is active at which time.

- Finally, signals shown in diagram (h) are grouped as follows (from top to bottom): The top diagram displays desired and actual heading signals for the Auto-Heading controller. The second from top diagram displays desired and actual depth signals for the Auto-Heave (Auto-Depth) controller. Input and output of the Surge-Brake controller are shown
in the third diagram. Finally, status of low-level controllers showing their activity over time is illustrated in the bottom diagram.

For the purpose of a brief assessment over the different way-point strategies, the test cases are arranged in several groups according to their features and design objective.

6.4.1 Group A: Planar way-point guidance test cases using the “Separate Auto-Heading & Cross-Track Compensation” strategy

Table 6.1 Group A: planar way-point guidance test cases using Separate Auto-Heading & Cross-Track compensation

<table>
<thead>
<tr>
<th>Rotate towards Target</th>
<th>Case A.1 (Figure 6.7) (page 169)</th>
<th>Case A.2 (Figure 6.8) (page 170)</th>
<th>Case A.3 (Figure 6.9) (page 171)</th>
<th>Case A.4 (Figure 6.10) (page 172)</th>
<th>Case A.5 (Figure 6.11) (page 173)</th>
</tr>
</thead>
<tbody>
<tr>
<td>External Disturbances (Ocean Currents)</td>
<td>Not present</td>
<td>Not present</td>
<td>Not present</td>
<td>Present</td>
<td>Not present</td>
</tr>
<tr>
<td>Approximation</td>
<td>Scaling</td>
<td>Truncation</td>
<td>Scaling</td>
<td>Scaling</td>
<td>Scaling</td>
</tr>
<tr>
<td>Missed Way-Point Handling</td>
<td>Range Exceeded (not active)</td>
<td>Range Exceeded (not active)</td>
<td>Range Exceeded (not active)</td>
<td>Range Exceeded (not active)</td>
<td>Target ReAcq. after Obstacle (active)</td>
</tr>
<tr>
<td>Vertical behaviour</td>
<td>Keep Constant Depth</td>
<td>Keep Constant Depth</td>
<td>Keep Constant Depth</td>
<td>Keep Constant Depth</td>
<td>Keep Constant Depth</td>
</tr>
<tr>
<td>Fuzzy Controller for Obstacle Avoidance</td>
<td>Not active</td>
<td>Not active</td>
<td>Not active</td>
<td>Not active</td>
<td>Active</td>
</tr>
</tbody>
</table>

Case A.1: Figure 6.7 (page 169) shows excellent tracking performance of the control system in the absence of external disturbances. Tracking error is very small during Move to Target stages. Small depth errors are noticeable during Rotate to Target stages due to the scaling approximation for unfeasible control vectors. Also, since the Surge-Brake controller is active during Rotate towards Target stages, a change in propeller rotation direction is required resulting in a certain waste of energy during these stages.
Case A.2: Figure 6.8 (page 170), diagram (h), second from top, shows an improved tracking performance compared to the previous case at the Rotate towards Target stage. Using T-approximation, only the out-of-range component of the unfeasible control vector is clipped (saturated), i.e. the component which exceeds the feasible region, as opposed to the previous S-approximation (scaling) method where all components of the vector are scaled, thus causing a change in depth. With the truncation approximation method, excellent performance is achieved in depth-keeping as well.

Case A.3: In this test case (Figure 6.9) (page 171) the Surge-Brake controller is not used during the rotation stages, which leads to better efficiency from the energy point of view. An increased 3D tracking error is noticeable in diagram (g), but the heading corrective action was able partially to compensate for it.

Case A.4: Figure 6.10 (page 172) displays corresponding simulation result for the case when the external disturbance (ocean current) is included and the “Separate Auto-Heading & Cross-Track Compensation” is used for the “Move to Target” behaviour. Due to limitations of the existing thruster configuration of Tethra (the sway DOF is not directly controllable), relatively significant tracking errors appear on path segments perpendicular to the current direction. Unavoidable sideways drifts are partially compensated by the heading correction.

Case A.5: Way-point guidance with an obstacle avoidance manoeuvre is shown in Figure 6.11 (page 173). The simulation experiment uses two obstacles with different sizes & positions, and two different settings of acceptance cone angles in order to demonstrate different scenarios of re-approaching the path after obstacle avoidance. For both obstacles the vehicle successfully detects the presence of the obstacle and performs avoidance, moving along the side of the obstacle at a safe distance. After the first obstacle avoidance manoeuvre
is completed, the vehicle is located outside the acceptance cone. Target way-point vector and line-of-sight heading information entries are updated in the way-point database and tracking is resumed in accordance to missed way-point Strategy 2 – “Target Reacquisition after Obstacle”. The vehicle rotates towards the target from the position where the acceptance cone has been exceeded and approaches the target following the shortest path. After finishing the avoidance of the second obstacle, the vehicle is still located inside the acceptance cone. The heading corrective action is applied and the vehicle reaches the next way-point while minimizing the cross-track error.

6.4.2 Group B: Planar way-point guidance test cases using the “Auto-Heading Integrated Cross-Track Compensation” strategy

Table 6.2 Group B: planar way-point guidance test cases using Auto-Heading Integrated Cross-Track compensation

<table>
<thead>
<tr>
<th></th>
<th>Case B.1 (Figure 6.12) (page 174)</th>
<th>Case B.2 (Figure 6.13) (page 175)</th>
<th>Case B.3 (Figure 6.14) (page 176)</th>
<th>Case B.4 (Figure 6.15) (page 177)</th>
</tr>
</thead>
<tbody>
<tr>
<td>Rotate towards Target</td>
<td>Rotate towards Target with Surge Brake</td>
<td>Rotate towards Target without Surge Brake</td>
<td>Rotate towards Target with Surge Brake</td>
<td>Rotate towards Target with Surge Brake</td>
</tr>
<tr>
<td>External Disturbances (Ocean Currents)</td>
<td>Not present</td>
<td>Not present</td>
<td>Present</td>
<td>Present</td>
</tr>
<tr>
<td>Approximation</td>
<td>Scaling</td>
<td>Scaling</td>
<td>Scaling</td>
<td>Scaling</td>
</tr>
<tr>
<td>Missed Way-Point Handling</td>
<td>Range Exceeded (not active)</td>
<td>Range Exceeded (not active)</td>
<td>Range Exceeded (not active)</td>
<td>Range Exceeded (active)</td>
</tr>
<tr>
<td>Vertical behaviour</td>
<td>Keep Constant Depth</td>
<td>Keep Constant Depth</td>
<td>Keep Constant Depth</td>
<td>Keep Constant Depth</td>
</tr>
</tbody>
</table>

1 The following settings are used for simulation of ocean currents: (0.10 m/s < Vc < 0.15 m/s, αc = 0°, βc = 270°).
Case B.1: Similar to case A.1, Figure 6.12 (page 174) shows excellent tracking performance of the control system in the absence of external disturbances, when the “Auto-Heading Integrated Cross-Track Compensation” is used for the Move to Target behaviour. Tracking error is very small along the path segments, and small depth errors appearing at rotation stages are caused by scaling of all components of unfeasible control vector during rotation stages.

Case B.2: Figure 6.13 (page 175), illustrates the case when the Surge-Brake controller is not used during the rotation stages. The tracking performance is similar to case A.2.

Case B.3: Figure 6.14 (page 176) displays simulation results for case when the external disturbance (ocean current) is included and the “Auto-Heading Integrated Cross-Track Compensation” is used for the Move to Target behaviour. Due to limitations of the existing thruster configuration of Tethra (the sway DOF is not directly controllable), relatively significant tracking errors appear on path segments perpendicular to the current direction. Unavoidable sideways drifts are partially compensated by the heading corrective action, using the Auto-Heading controller, instead of yaw component of the virtual joystick, as in case A.4. The intensity of the heading correction is controlled by varying the scaling factor $A_2$ in (5.7). For the parameter value of $A_2 = 1.15$, compensation for the cross-track error is good and the vehicle reaches all target way-points without activating the missed way-point handling strategy.

Case B.4: Case (Figure 6.15) (page 177) is similar to the previous test case, and simulation is performed in the presence of external disturbances. For the specific parameter value of $A_2 = 1.00$ used in (5.7), compensation for the cross-track error is inefficient, therefore when moving perpendicular to the current direction the vehicle exceeds the range before entering
Chapter 6: Testing & Evaluation

the target way-point acceptance sphere. According to the “Range Exceeded – Missed Way-point” Strategy 1, the vehicle rotates towards the target from that position and approaches the target following the shortest path.

6.4.3 Group C: Planar way-point guidance test cases using the “Look-Ahead” approach for smooth way-point transition

Table 6.3 Group C: Planar way-point guidance test cases using the Look-Ahead approach for smooth way-point transition.

<table>
<thead>
<tr>
<th>Rotate towards Target</th>
<th>Case C.1 (Figure 6.16) (page 178)</th>
<th>Case C.2 (Figure 6.17) (page 179)</th>
</tr>
</thead>
<tbody>
<tr>
<td>Rotate towards Target without Surge Brake</td>
<td>Rotate towards Target without Surge Brake</td>
<td>Rotate towards Target without Surge Brake</td>
</tr>
<tr>
<td>Move to Target</td>
<td>Auto-Heading with Look-Ahead</td>
<td>Separate Auto-Heading &amp; Cross-Track Compensation</td>
</tr>
<tr>
<td>Transition to New Transect</td>
<td>None</td>
<td>Look-Ahead Transition to New Transect</td>
</tr>
<tr>
<td>Approximation</td>
<td>Scaling</td>
<td>Scaling</td>
</tr>
<tr>
<td>Missed Way-Point Handling</td>
<td>Range Exceeded (not active)</td>
<td>Range Exceeded (not active)</td>
</tr>
<tr>
<td>Vertical behaviour</td>
<td>Keep Constant Depth</td>
<td>Keep Constant Depth</td>
</tr>
</tbody>
</table>

Case C.1: Figure 6.16 (page 178) shows simulation results for smooth way-point transition. The size of the virtual pool was chosen to be 15 times bigger than an original size of the real pool in order to illustrate the functionality of the strategy on real-scale survey transect lengths. It can be seen that this way-point guidance scheme takes into account the position of the next way-point before the current way-point is reached. Trajectory diagrams (a) and (b) display smooth way-point tracking performance.

Case C.2: In this case, illustrated in Figure 6.17 (page 179), high tracking accuracy with constant forward speed is obtained along path segments. Transition between segments is performed along circular arc curves in “lawn mower” fashion.
6.4.4 3D way-point guidance test cases

Table 6.4 Group D: 3D way-point guidance test cases.

<table>
<thead>
<tr>
<th></th>
<th>Case D.1 (Figure 6.18) (page 180)</th>
<th>Case D.2 (Figure 6.19) (page 181)</th>
<th>Case D.3 (Figure 6.20) (page 182)</th>
<th>Case D.4 (Figure 6.21) (page 183)</th>
<th>Case D.5 (Figure 6.22) (page 184)</th>
</tr>
</thead>
<tbody>
<tr>
<td>Rotate towards Target</td>
<td>Rotate towards Target with Surge Brake</td>
<td>Rotate towards Target with Surge Brake</td>
<td>Rotate towards Target with Surge Brake</td>
<td>Rotate towards Target with Surge Brake</td>
<td>Rotate towards Target without Surge Brake</td>
</tr>
<tr>
<td>Approximation</td>
<td>Scaling</td>
<td>Scaling</td>
<td>Scaling</td>
<td>Scaling</td>
<td>Scaling</td>
</tr>
<tr>
<td>Missed Way-Point Handling</td>
<td>Range Exceeded (not active)</td>
<td>Range Exceeded (not active)</td>
<td>Range Exceeded (not active)</td>
<td>Range Exceeded (not active)</td>
<td>Range Exceeded (not active)</td>
</tr>
<tr>
<td>Vertical behaviour</td>
<td>Keep Constant Altitude</td>
<td>staircase profile</td>
<td>Depth tracking</td>
<td>Depth tracking</td>
<td>Keep Constant Depth &amp; Depth tracking</td>
</tr>
<tr>
<td>Feedforward compensation</td>
<td>Not active</td>
<td>Not active</td>
<td>Not active</td>
<td>Active</td>
<td>Active</td>
</tr>
</tbody>
</table>

Case D.1: Figure 6.18 (page 180) demonstrates the performance of the proposed GNC system for constant altitude keeping requirements. The Auto-Heave controller operates in the altitude mode and feedforward compensation is not active (switch S in Figure 4.12 is open) causing noticeable altitude tracking error above the slope. The trajectory of the vehicle is inherently 3D, as defined by the shape of the bottom of the pool.

Case D.2: 3D way-point guidance with stair-case profiling is shown in Figure 6.19 (page 181). Instead of following the shortest path between consecutive way points, this approach performs two-step process: desired depth is reached in the first step, and target way-point is reached in the second step using planar motion algorithms.

Case D.3: Figure 6.20 (page 182) displays tracking performance for 3D way-point guidance in the case when feedforward compensation is not active. The Auto-Heave controller operates in the depth mode. It can be seen from (h2) that the vehicle is not able to accurately
follow the path: during ascending (descending) the vehicle is located below (above) the desired path.

**Case D.4:** This case (Figure 6.21) (page 183) is similar to previous case D.3, but this time with active feedforward compensation. Feedforward action causes the depth tracking error to disappear and the vehicle is able to track the desired 3D path with high accuracy.

**Case D.5:** Figure 6.22 (page 184) illustrates a possible scenario of acoustic sea-floor surveying of transects at different depth values for different widths of swath coverage. The “Look-Ahead Transition to New Transect” strategy with active feedforward compensation is used to drive the vehicle along a 3D circular arc trajectory manoeuvre, while transiting to the new transect. Diagram (h2) shows good tracking performance.

### 6.4.5 Way-point guidance test cases using the aiding way-points

**Table 6.5** Group E: Way-point guidance test cases using the aiding way-points.

<table>
<thead>
<tr>
<th></th>
<th>Case E.1 (Figure 6.23) (page 185)</th>
<th>Case E.2 (Figure 6.24) (page 186)</th>
</tr>
</thead>
<tbody>
<tr>
<td>Rotate towards Target</td>
<td>Rotate towards Target without Surge Brake</td>
<td>Rotate towards Target without Surge Brake</td>
</tr>
<tr>
<td>Move to Target</td>
<td>Separate Auto-Heading &amp; Cross-Track Compensation</td>
<td>Separate Auto-Heading &amp; Cross-Track Compensation</td>
</tr>
<tr>
<td>Transition to New Transect</td>
<td>Transition to New Transect with the Aiding Way-Points</td>
<td>Transition to New Transect with the Aiding Way-Points</td>
</tr>
<tr>
<td>Approximation</td>
<td>Scaling</td>
<td>Scaling</td>
</tr>
<tr>
<td>Missed Way-Point Handling</td>
<td>Range Exceeded (not active)</td>
<td>Range Exceeded (not active)</td>
</tr>
<tr>
<td>Vertical behaviour</td>
<td>Keep Constant Depth</td>
<td>Depth Tracking</td>
</tr>
<tr>
<td>Feedforward compensation</td>
<td>Not active</td>
<td>Active</td>
</tr>
</tbody>
</table>

**Case E.1:** Figure 6.23 (page 185) illustrates a planar way-point guidance test case using the “Transition to New Transect using the Aiding Way-points” strategy. Good tracking performance along the path segments is shown in Figure 6.23, diagram (g1).
Case E.2: This case (Figure 6.22) (page 186) is similar to the one described at case D.5, with constant depth-keeping along the path segments and 3D depth tracking with feedforward compensation at transition to new transect. For transition to new transect, the same method of Bezier curve aiding way-points are used as described in the previous case E.1. The depth value entries for the aiding way-points database are determined using linear interpolation. Figure 6.24, diagrams (g1) and (h2) indicate good tracking capability along the segments, and good depth-tracking performance along the 3D transitions.

6.4.6 Current diagrams

Time diagrams showing current draw of thrusters during simulation are indicated in Figure 6.25 - Figure 6.42. Table 6.6 can be used for cross-referencing test cases with corresponding current diagrams.

Table 6.6 Cross-reference table linking test cases and current diagrams.

<table>
<thead>
<tr>
<th>Test case</th>
<th>Current diagram</th>
<th>Test case</th>
<th>Current diagram</th>
<th>Test case</th>
<th>Current diagram</th>
</tr>
</thead>
<tbody>
<tr>
<td>A.1</td>
<td>Figure 6.25</td>
<td>B.2</td>
<td>Figure 6.31</td>
<td>D.2</td>
<td>Figure 6.37</td>
</tr>
<tr>
<td></td>
<td>(page 187)</td>
<td></td>
<td>(page 188)</td>
<td></td>
<td>(page 189)</td>
</tr>
<tr>
<td>A.2</td>
<td>Figure 6.26</td>
<td>B.3</td>
<td>Figure 6.32</td>
<td>D.3</td>
<td>Figure 6.38</td>
</tr>
<tr>
<td></td>
<td>(page 187)</td>
<td></td>
<td>(page 188)</td>
<td></td>
<td>(page 189)</td>
</tr>
<tr>
<td>A.3</td>
<td>Figure 6.27</td>
<td>B.4</td>
<td>Figure 6.33</td>
<td>D.4</td>
<td>Figure 6.39</td>
</tr>
<tr>
<td></td>
<td>(page 187)</td>
<td></td>
<td>(page 188)</td>
<td></td>
<td>(page 189)</td>
</tr>
<tr>
<td>A.4</td>
<td>Figure 6.28</td>
<td>C.1</td>
<td>Figure 6.34</td>
<td>D.5</td>
<td>Figure 6.40</td>
</tr>
<tr>
<td></td>
<td>(page 187)</td>
<td></td>
<td>(page 188)</td>
<td></td>
<td>(page 189)</td>
</tr>
<tr>
<td>A.5</td>
<td>Figure 6.29</td>
<td>C.2</td>
<td>Figure 6.35</td>
<td>E.1</td>
<td>Figure 6.41</td>
</tr>
<tr>
<td></td>
<td>(page 187)</td>
<td></td>
<td>(page 188)</td>
<td></td>
<td>(page 189)</td>
</tr>
<tr>
<td>B.1</td>
<td>Figure 6.30</td>
<td>D.1</td>
<td>Figure 6.36</td>
<td>E.2</td>
<td>Figure 6.42</td>
</tr>
<tr>
<td></td>
<td>(page 187)</td>
<td></td>
<td>(page 188)</td>
<td></td>
<td>(page 189)</td>
</tr>
</tbody>
</table>
Figure 6.7 Case A.1: Way-point Guidance using “Separate Auto-Heading & Cross-Track Compensation”, Approximation method: Scaling.

169
Figure 6.8 Case A.2: Way-point Guidance using “Separate Auto-Heading & Cross-Track Compensation”, Approximation method: Truncation.
Figure 6.9 Case A.3: Way-point Guidance using “Separate Auto-Heading & Cross-Track Compensation”, “Rotate towards Target without Surge Brake”
Figure 6.10 Case A.4: Way-point Guidance using “Separate Auto-Heading & Cross-Track Compensation”, with external disturbances (ocean currents)
Figure 6.11 Case A.5: Way-point Guidance using “Separate Auto-Heading & Cross-Track Compensation”, with obstacle avoidance and missed-waypoint handling
Figure 6.12 Case B.1: Way-point Guidance using “Auto-Heading Integrated Cross-Track Compensation”
Figure 6.13 Case B.2: Way-point Guidance using “Auto-Heading Integrated Cross-Track Compensation”, “Rotate towards Target without Surge Brake”
Chapter 6: Testing & Evaluation

Figure 6.14 Case B.3: Way-point Guidance using “Auto-Heading Integrated Cross-Track Compensation”, with external disturbances (ocean currents)
Figure 6.15 Case B.4: Way-point Guidance using “Auto-Heading Integrated Cross-Track Compensation”, with external disturbances (ocean currents) and missed way-point handling
Figure 6.16 Case C.1: Way-point Guidance using "Auto-Heading with Look-Ahead"
Figure 6.17 Case C.2: Way-point Guidance using “Look-Ahead Transition to New Transect”
**Figure 6.18** Case D.1: Way-point Guidance using constant altitude keeping without feedforward compensation
Figure 6.19 Case D.2: 3D Way-point Guidance using stair-case profile
Figure 6.20 Case D.3: 3D Way-point Guidance without feedforward compensation
Figure 6.21 Case D.4: 3D Way-point Guidance with feedforward compensation
Figure 6.22 Case D.5: 3D Way-point Guidance using constant depth keeping along transects and feed-forward compensation at “Look-Ahead Transition to New Transects”
Chapter 6: Testing & Evaluation

Figure 6.23 Case E.1: Way-point Guidance using “Transition to New Transect using the Aiding Way-points”
Figure 6.24 Case E.2: 3D Way-point Guidance using constant depth keeping along transects and feed-forward compensation at “Transition to New Transect using the Aiding Way-points”
Chapter 6: Testing & Evaluation

Figure 6.25  Thrusters’ current draw for case A.1

Figure 6.26  Thrusters’ current draw for case A.2

Figure 6.27  Thrusters’ current draw for case A.3

Figure 6.28  Thrusters’ current draw for case A.4

Figure 6.29  Thrusters’ current draw for case A.5

Figure 6.30  Thrusters’ current draw for case B.1
Figure 6.31 Thrusters’ current draw for case B.2

Figure 6.32 Thrusters’ current draw for case B.3

Figure 6.33 Thrusters’ current draw for case B.4

Figure 6.34 Thrusters’ current draw for case C.1

Figure 6.35 Thrusters’ current draw for case C.2

Figure 6.36 Thrusters’ current draw for case D.1
Figure 6.37 Thrusters’ current draw for case D.2

Figure 6.38 Thrusters’ current draw for case D.3

Figure 6.39 Thrusters’ current draw for case D.4

Figure 6.40 Thrusters’ current draw for case D.5

Figure 6.41 Thrusters’ current draw for case E.1

Figure 6.42 Thrusters’ current draw for case E.2
6.5 Comparative analysis

The main purpose of this section is to compare energy consumption of thrusters for four selected representative cases. The electrical power of thrusters DC-motor is equal to

\[ P(t) = V_e(t)I(t) \]  
(6.1)

(see Figure 2.3). The electrical energy consumed over time \( T \) is defined as

\[ W(T) = \int_0^T P(t)dt \]  
(6.2)

Electrical energy is calculated for horizontal and vertical thrusters over time \( T \), which is equal to the length of the mission, i.e. time interval from the initial position until the last way-point is reached. The following nomenclature is adopted to compare selected test cases:

\[ W_{HT}(T) = W_{1HT}(T) + W_{2HT}(T) \]  
(total energy consumed by horizontal thrusters)  
(6.3)

\[ W_{VT}(T) = W_{1VT}(T) + W_{2VT}(T) \]  
(total energy consumed by vertical thrusters)  
(6.4)

\[ \overline{P}_{HT} = \frac{W_{HT}(T)}{T} \]  
(average power of horizontal thrusters)  
(6.5)

\[ \overline{P}_{VT} = \frac{W_{VT}(T)}{T} \]  
(average power of vertical thrusters)  
(6.6)

![Figure 6.43](image)

**Figure 6.43** Power consumption of thrusters for planar way-point guidance using “Separate Auto-Heading & Cross-Track compensation”, with “Rotate towards Target, with Surge-Brake” (case A.1) (left) and “Rotate towards Target, without Surge-Brake” (case A.3) (right).
Chapter 6: Testing & Evaluation

Table 6.7 Comparison of power and energy consumption for cases A.1 and A.3.

<table>
<thead>
<tr>
<th></th>
<th>$T$ [s]</th>
<th>$W_{HT}$ [Wh]</th>
<th>$W_{VT}$ [Wh]</th>
<th>$\bar{P}_{HT}$ [W]</th>
<th>$\bar{P}_{VT}$ [W]</th>
</tr>
</thead>
<tbody>
<tr>
<td>Case A.1</td>
<td>804.88</td>
<td>8.5006</td>
<td>4.9588</td>
<td>38.02</td>
<td>22.179</td>
</tr>
<tr>
<td>Case A.3</td>
<td>744.17</td>
<td>8.5948</td>
<td>4.5846</td>
<td>41.578</td>
<td>22.178</td>
</tr>
</tbody>
</table>

It can be seen that the length of the mission is significantly shorter for case A.3 than A.1. The price paid is higher average power and increased tracking error.

![Graph](image)

Figure 6.44 Power consumption of thrusters for planar way-point guidance using “Auto-Heading Integrated Cross-Track compensation”, with “Rotate towards Target, with Surge-Brake” (case B.1) (left) and “Rotate towards Target, without Surge-Brake” (case B.2) (right).

Table 6.8 Comparison of power and energy consumption for cases B.1 and B.2.

<table>
<thead>
<tr>
<th></th>
<th>$T$ [s]</th>
<th>$W_{HT}$ [Wh]</th>
<th>$W_{VT}$ [Wh]</th>
<th>$\bar{P}_{HT}$ [W]</th>
<th>$\bar{P}_{VT}$ [W]</th>
</tr>
</thead>
<tbody>
<tr>
<td>Case B.1</td>
<td>810.84</td>
<td>8.3401</td>
<td>4.9942</td>
<td>37.029</td>
<td>22.174</td>
</tr>
<tr>
<td>Case B.2</td>
<td>744.37</td>
<td>8.7965</td>
<td>4.5851</td>
<td>42.543</td>
<td>22.175</td>
</tr>
</tbody>
</table>

The conclusion is similar as in previous case, that is, use of the Surge-Brake extends the length of the mission with improved tracking accuracy.
Testing and evaluation of different fuzzy controllers (FCOA safety margin)

A safety margin (safe distance from an obstacle during avoidance stage) depends on the position of the overlapping region of membership functions: higher safety margin is obtained by shifting the overlapping region to the right. In order to illustrate the relationship between safety margins and membership functions three cases were considered. Case MF-1 illustrated in Figure 6.45 has the smallest safety margin (overlapping region is interval $[1,2]$). Case MF-2 is characterised by higher safety margin (overlapping region is interval $[2,3]$, Figure 6.46). Finally, the biggest safety margin is obtained in case MF-3, as indicated in Figure 6.47 (overlapping region is interval $[3,4]$).

Figure 6.45 Case MF-1: (a) Vehicle trajectory– top view (b) Thrusters’ power consumption (c) Membership functions for the input variables (overlapping region $[1,2]$).
Figure 6.46 Case MF-2: (a) Vehicle trajectory—top view (b) Thrusters’ power consumption (c) Membership functions for the input variables (overlapping region [2,3]).

Figure 6.47 Case MF-3: (a) Vehicle trajectory—top view (b) Thrusters’ power consumption (c) Membership functions for the input variables (overlapping region [3,4]).
Table 6.9 provides comparison of power and energy consumption for cases MF-1, MF-2 and MF-3. It can be noticed that the energy consumption and the length of the mission is the highest for case MF-1 and the lowest for case MF-3. The average power is the smallest for case MF-1 and will increase through case MF-2, reaching the highest value for case MF-3. These results are expected, since higher safety margin means that the vehicle can travel faster during obstacle avoidance stage.

A complete, thorough analysis on the robustness of the fuzzy logic approach for obstacle avoidance is beyond the scope of this thesis. The above comparative experiments have been carried out in the absence of external disturbances (ocean currents). A study on the robustness of different fuzzy controllers for obstacle avoidance in the presence of external disturbances, and the development of dynamic path planning algorithms are subject of future work.

6.6 Concluding remarks

This chapter has evaluated the performance of the proposed GNC system for the Tethra AUV and highlighted its key features. Using computer simulations, the performance was evaluated through a series of representative test cases. These cases were chosen to represent real-world survey scenarios. Performance of the control scheme has been investigated in both still water and in the presence of currents. Simulation results presented in this chapter demonstrate the versatility and effectiveness of the proposed GNC system.
Chapter 6: Testing & Evaluation

For reasons of high cost, and long experimentation cycle time from mobilisation through in-water testing and demobilisation, full scale pool and lake tests of the comprehensive Guidance, Navigation and control system on the Tethra vehicle have not yet been performed. The basic behaviours have though been successfully tested in the diving pool.

Analysing the simulation results and partial real-world tests, the following conclusion can be drawn:

The influence of external disturbances can be only partially compensated, due to constraints imposed by thruster configuration of Tethra.
Chapter 7: Conclusions & Future work

7.1 Introduction

This chapter reviews and summarises the work described in the thesis, lists and discusses the main contributions and proposes suggestions for further work.

7.2 Review of the thesis

Chapter 1 is an introductory chapter, which provided background and motivation for the work described in the thesis, identified aims and objectives, lists the main contributions and gave an overview of chapter contents.

A comprehensive literature survey has been undertaken in chapter 2 in order to explore the existing methods for guidance, navigation and control for underwater vehicles. Research fields covered in this survey include the following topics: propulsion systems & control allocation, control architectures for mobile robots & AUVs and way-point guidance systems.

A hybrid control architecture, described in this chapter, was used in chapters 4 and 5 as a foundation to build a novel guidance, navigation and control system for open-frame, thrusters-propelled underwater vehicles.

Chapter 3 provided insight into Tethra design concepts, including the description of mechanical and electrical interface, actuators, power system, embedded microcontrollers, navigation sensors (initial and new design) and communication protocols.

A novel guidance, navigation and control (GNC) system for Tethra AUV is described in chapter 4. Feedback signals from the navigation system and reference signals from the guidance system are used by the control system to actuate thrusters in order to satisfy the control objective and perform mission tasks. The hybrid control architecture described in
Chapter 7: Conclusions & Future work

Chapter 2 was modified to accommodate the thruster configuration of Tethra. The main components of the control system include: Control Allocation, Virtual Joystick, Low-level controllers, Fuzzy controller for Obstacle Avoidance and Behaviour Coordinator/Arbitration. Chapter 5 describes the main features of the way-points guidance system, which represents one of the main components of the proposed GNC system for Tethra, presented in chapter 4. The way-points guidance system has been implemented through the mission planner and tracking way-points planner, which are realised by means of state machines. The mission planner includes typical AUV survey mission stages. The tracking way-points planner is realised as a state machine whose parent is the tracking way-points super-state in the mission planner state machine. Two basic low-level behaviours (“Rotate towards Target” and “Move to Target”) are executed in the loop, until the tracking is finished or the mission is aborted. Depending on criteria to be optimised, a number of strategies have been proposed for these basic behaviours. This basic set of behaviours was extended to handle missed way-points & smooth transition between transects. The feedforward control scheme was proposed to provide very accurate trajectory tracking in 3D space.

Chapter 6 evaluated the performance of the proposed GNC system for Tethra AUV and highlighted its key features. Using computer simulations, the performance was evaluated through a series of representative test cases. These cases were chosen to represent real-world underwater survey scenarios. Performance of the control scheme has been investigated in both still water and in the presence of currents. Simulation results presented in this chapter demonstrated the versatility and effectiveness of the proposed GNC system. Simulation results illustrated that the proposed GNC system exhibits excellent tracking performance in the absence of external disturbances for various underwater scenarios, including precise 3D
path following. Due to limitations of the existing thruster configuration of Tethra (the sway DOF is not directly controllable), relatively significant tracking errors appear on path segments perpendicular to the current direction in the presence of external disturbances (ocean currents). Unavoidable sideways drifts are partially compensated by the heading corrective action of the GNC system. The proposed GNC system automatically compensates for the non-symmetrical shape of propeller $T$-curves in an optimal way, making the command input space appear symmetrical. Preliminary test results of real-world application of basic behaviours were also presented.

7.3 **Realisation of aims & objectives**

This section discusses accomplishments in realisation of the aims and objectives, given in section 1.3.

7.3.1 **Realisation of aims**

The GNC system, proposed in chapters 4 and 5, fulfil the following requirements:

- For feasible command inputs optimal control allocation is guaranteed, since the inverse solution (4.29) is unique in that case.
- For unfeasible command inputs two feasible approximations of unfeasible control vectors are proposed: truncation (clipping) and scaling.
- In the absence of ocean currents the GNC system exhibits excellent tracking performance both for planar and full 3D path following.

Key features of Tethra that make challenging near seabed scenarios possible are highlighted in the following:
Chapter 7: Conclusions & Future work

- Multi thrusters design allows hovering and multivectored control, while torpedo shaped vehicles with flaps need forward velocity for control surface efficacy.

- Fault tolerance capabilities required for near seabed operation, e.g. thruster clogging in weed, net etc.; Tethra has better chance than vehicle without fault tolerance capabilities.

7.3.2 Accomplishments of objectives

- Explore the existing methods for guidance, navigation and control for underwater vehicles. An extensive literature survey was undertaken in chapter 2, covering the following topics: propulsion systems, control allocation methods, control architectures for mobile robots and AUVs, and way-point guidance systems.

- Identify methods that will be used as basis for novel GNC system to meet requirements of the research project. Affine thrusters model, control allocation, hybrid control architecture and way-point guidance approaches from chapter 2 were identified to be used as a foundation for development of the GNC system.

- Formulate and solve control allocation problem for Tethra AUV. The control allocation problem was formulated and solved in chapter 4.

- Design and develop GNC system architecture. The GNC system for Tethra AUV is described in chapters 4 and 5.

- Develop the real-time simulator to test algorithms. The real-time implementation of the GNC system is described in Appendix A. This simulator is used to evaluate the performance of the GNC system in different mission scenarios and simulation results are presented in chapter 6.
• **Verify the performance of the GNC system in real-world applications.** The preliminary results of real-world testing in the diving pool, swimming pool and Laugh Derg Lake were given in chapter 6. Due to time limits, high mobilisation costs and limited financial resources, comprehensive full-scale open water tests have not been performed.

### 7.4 Main contributions

This section discusses the main contributions of the work presented in the thesis.

• **Decomposition of control problem for UUVs into regulation and actuator selection tasks.** The control problem is decomposed into Regulation Task (which determines the necessary forces and moments to satisfy control objective) and actuator selection task (which maps the total control effort (forces and moments) onto individual actuators, i.e. input voltages to be applied to drive thrusters).

• **Full implementation of the hybrid control architecture with safety critical behaviours with highest priority.** The control system utilizes a hybrid control approach, which combines the advantages of a top-down traditional (hierarchical) AI approach and a bottom-up behaviour-based approach. The control architecture is structured in three layers: the deliberative layer, the control execution layer and the functional reactive layer. The deliberative layer transforms the mission into a set of tasks. The reactive layer takes care of the real-time issues related to the interactions with the environment. It calls a set of low-level controllers with auto-tuning capabilities, followed by a control allocator. The control execution layer interacts between the upper and lower layers, supervising the accomplishment of the tasks. Thus, the hybrid architecture takes advantage of the hierarchical planning aspects of
traditional AI and the reactive & real-time aspects of the behaviour-based approach.

The main components of the control system include: Control Allocation, Virtual Joystick, Low-level controllers, Fuzzy controller for Obstacle Avoidance and Behaviour Coordinator/Arbitration. Due to the under-actuated thruster configuration of Tethra, cross-tracking error compensation has been performed by different strategies, including heading-integrated and heading-separated approaches, described in chapter 5.

- **Formulation of the control allocation problem in normalised form and its solution, including clear geometrical interpretation.** The control allocation problem for Tethra AUV is fully described and solved, including clear geometrical interpretation and visualisation of the solution, in chapter 4.

- **Introducing the control buffer concept with clusters for virtual joystick and low-level controller clusters.** The control buffer concept has been developed to provide transparency and easy fusion of different task executor control demands. Each task executor has its own control cluster and mask inside the control buffer. The control cluster consists of Hand Control Unit (HCU) components (to simulate a virtual joystick), a set of settings for low-level controllers (set points and on/off switches to activate/deactivate individual controllers) and a priority value, reserved for future use in competitive arbitration within the reactive layer. A set of low-level controllers includes an Auto-Heave Controller (to maintain/track desired depth/altitude), an Auto-Heading (to maintain desired heading) and a Surge-Brake Controller (to stop surge motion in the absence of a demand to move forward/backward, like a mechanical damper). The Auto-Heave controller is enhanced with the feedforward
control path in order to cancel tracking error for time-varying depth/altitude input trajectories. The virtual joystick is introduced to mimic ROV pilot commands. In the ROV operation mode, the virtual joystick is driven by an ROV pilot. In the AUV operation mode, it is controlled by the Behaviour Coordinator/Arbitration component.

- **Development of Fuzzy Logic controller for 3D obstacle avoidance.** A three-dimensional algorithm for obstacle avoidance is presented in chapter 4. It utilises a Sugeno-type Fuzzy logic controller with 6 inputs, 3 outputs and 64 rules. The inputs are provided from altimeters mounted on the vehicle. The outputs are normalised forces & moments components, bundled in the AUV Obstacle Avoidance control cluster.

- **Realisation of mission & tracking way-point planners as state machines.** The way-points guidance system has been implemented through the mission planner and tracking way-points planner, which are realised by means of state machines. The mission planner includes typical AUV survey mission stages. The tracking way-points planner is realised as a state machine whose parent is the tracking way-points super-state in the mission planner state machine. Two basic low-level behaviours (“Rotate towards Target” and “Move to Target”) are executed in the loop, until the tracking is finished or the mission is aborted. Depending on criteria to be optimised, a number of strategies are proposed for these basic behaviours. This basic set of behaviours is extended to handle missed way-points & smooth transition between transects. The feedforward control scheme is proposed to provide very accurate trajectory tracking in 3D space. The open architecture of the proposed GNC system
enables easy augmentation with new strategies for objectives that are not covered in this thesis.

- **Fusion of task executors using arbitration.** Each control cluster is masked with a corresponding mask, depending on the state of the mission and navigation data (interaction with the real world). A mask is built from weights and logic gates, which are bundled into the same structure as the control cluster. Masking is performed by multiplying (ANDing) corresponding fields in the control cluster and mask. The priority level of the task executor is determined by the mask content. In this way it is possible to control the degree of co-operation and competition between different task executors. After masking, control clusters are bundled into the Winner Control Cluster, which has exclusive actuator control.

- **Design of a simulation model with virtual reality display.** A real-time simulator (see Appendix A), including AUV dynamics, wall reaction, sonar simulation and virtual reality display of the way-points inside the virtual pool has been developed to test and evaluate algorithms proposed in chapters 4 and 5. The simulator is the result of combined efforts of the author and other members of the research group. A virtual reality display raises the level of graphical presentation into a new dimension. Real-world situations and mission scenarios can be easily simulated, without the need to perform time-consuming and expensive trials.

### 7.5 Conclusion & future work

Further work will involve the following activities:

- Assessment and analysis of robustness of the proposed control scheme,
Chapter 7: Conclusions & Future work

- A study on the robustness of different fuzzy controllers for obstacle avoidance in the presence of external disturbances,
- The development of dynamic path planning algorithms,
- Improvements of low-level controllers by introduction of feedforward control path in the Auto-Heading controller,
- Development of fully autonomous auto-tuning features for low-level controllers,
- Full scale real-world tests of the proposed GNC system,
- Development of a Virtual Underwater lab with other members of the research group.
References


References


References


References


208


REFERENCES


References


212

Appendix A

Appendix A: Real-time Tethra simulator

Body-fixed and Earth-fixed coordinate frames for Tethra

Dynamic equation of motion in 6 DOF:
The following 6 DOF dynamic model of an UUV is used in order to simulate ocean currents and their effect on motion:

\[
\begin{align*}
&M^B \dot{v} + C_{RB}^B v + C_A^B v_r + D^B v_r + g^E \eta = \tau
\end{align*}
\]

- \(M\): inertia matrix (including added mass)
- \(C_{RB}^B v\): rigid body Coriolis and centripetal matrix
- \(C_A^B v_r\): added-mass Coriolis and centripetal matrix
- \(D^B v_r\): total hydrodynamic damping matrix
- \(g^E \eta\): vector of restoring (gravitational and buoyant) forces and moments
- \(\tau\): vector of propulsion forces and moments (exerted by the thrusters)
- \(\eta\): position and orientation of the vehicle in earth fixed frame
- \(v\): current velocity vector
- \(v_r = v - v_c\): relative velocity vector

It is assumed that \(v_c\) is slowly-varying, that is \(\dot{v}_c = 0\).
Appendix A

Kinematic equation of motion:

$$E\eta = \begin{bmatrix} E\eta_1 \\ E\eta_2 \end{bmatrix}$$

position and orientation vector relative to \(\{E\}\)

$$E\eta_1 = \begin{bmatrix} x \\ y \\ z \end{bmatrix}$$

position vector relative to \(\{E\}\)

$$E\eta_2 = \begin{bmatrix} \phi \\ \theta \\ \psi \end{bmatrix}$$

orientation vector relative to \(\{E\}\)

$$B\nu = \begin{bmatrix} B\nu_1 \\ B\nu_2 \end{bmatrix}$$

linear and angular velocity vector relative to \(\{B\}\)

$$B\nu_1 = \begin{bmatrix} u \\ v \\ w \end{bmatrix}$$

linear velocity vector relative to \(\{B\}\)

$$B\nu_2 = \begin{bmatrix} p \\ q \\ r \end{bmatrix}$$

angular velocity vector relative to \(\{B\}\)

Euler angles attitude representation:

$$\begin{bmatrix} E\eta_1 \\ E\eta_2 \end{bmatrix} = J_1(E\eta_2) \begin{bmatrix} 0_{3,3} \\ J_2(E\eta_2) \end{bmatrix} B\nu_1$$

$$\iff \quad E\eta = J(E\eta)^B \nu$$

$$J_1(E\eta_2) = \begin{bmatrix} \cos \psi \cos \theta & -\sin \psi & \cos \theta \cos \phi + \sin \psi \sin \theta \sin \phi & \sin \psi \sin \phi + \cos \psi \cos \phi \sin \theta \\ \sin \psi \cos \theta & \cos \psi & \cos \psi \cos \phi + \sin \psi \sin \theta \sin \phi & -\cos \psi \sin \phi + \sin \theta \sin \psi \cos \phi \\ -\sin \theta & 0 & \cos \theta \sin \phi & \cos \theta \cos \phi \\ 0 & \cos \theta & \cos \phi \sin \theta & \cos \phi \cos \theta \end{bmatrix}$$

$$J_2(E\eta_2) = \begin{bmatrix} 1 & \sin \phi \tan \theta & \cos \phi \tan \theta \\ 0 & \cos \phi & -\sin \phi \\ 0 & \sin \phi & \cos \phi \\ 0 & \cos \theta & \cos \theta \end{bmatrix}$$
Appendix A

3D Irrotational Current Model:
The current velocity is generated by using a 1st-order Gauss-Markov process:
\[ V_c + \mu V_c = w \]

- \( V_c \) current velocity
- \( \alpha_c \) angle of attack
- \( \beta_c \) sideslip angle
- \( w \) Gaussian white noise
- \( \mu > 0 \) constant value

Components of the current velocity \( {}^B v_c = [{}^B u_c, {}^B v_c, {}^B w_c, 0, 0, 0] \) are found from:

\[
{}^B V_c = {}^B R_{E}^{B}(\eta_2) {}^B R_{y, \alpha_c}^{T} {}^B R_{z, -\beta_c}^{T} [V_c]
\]

where the rotation matrices \( {}^B R_{E}^{B}(\eta_2) \), \( {}^B R_{y, \alpha_c}^{T} \) and \( {}^B R_{z, -\beta_c}^{T} \) are defined as:

\[
{}^B R_{E}^{B}(\eta_2) = \begin{bmatrix}
\cos \psi \cos \theta & \sin \psi \cos \theta & -\sin \theta \\
-\sin \psi \cos \phi + \cos \psi \sin \theta \sin \phi & \cos \psi \cos \phi + \sin \psi \sin \theta \sin \phi & \cos \theta \sin \phi \\
\sin \psi \sin \phi + \cos \psi \cos \phi \sin \theta & -\cos \psi \sin \phi + \sin \theta \sin \psi \cos \phi & \cos \theta \cos \phi
\end{bmatrix}
\]

\[
{}^B R_{y, \alpha_c} = \begin{bmatrix}
\cos \alpha_c & 0 & \sin \alpha_c \\
0 & 1 & 0 \\
-\sin \alpha_c & 0 & \cos \alpha_c
\end{bmatrix}
\]

\[
{}^B R_{z, -\beta_c} = \begin{bmatrix}
\cos \beta_c & \sin \beta_c & 0 \\
-\sin \beta_c & \cos \beta_c & 0 \\
0 & 0 & 1
\end{bmatrix}
\]
Simulation diagram for Tethra dynamics and kinematics using the Euler angles attitude representation

Simulink model to implement the dynamics and kinematics for Tethra
Appendix A

Real-time simulator – functional diagram

Control Centre

Ocean current

Propulsion System

ROV/AUV Dynamics & kinematics

Wall reaction

Navigation

Sonars

Virtual Reality

Interface \( \{ \mathbf{e} \} \rightarrow \{ \mathbf{vR} \} \)

\( \mathbf{\eta}, \mathbf{v}, sA, \ldots, sF \)

Tethra Simulator

\( \mathbf{\eta}, \mathbf{v}, sA, \ldots, sF \)

\( \mathbf{n}_f \)

\( \mathbf{n}_f \)
Illustration of the way-point database

Navigation data display
Appendix A

Mission supervision, way-point guidance strategy selection interface

Guidance and Navigation display
Appendix A

Low-level control display

Control allocation
Appendix A

Simulink configuration

Communication parameters
Way-point database generator

Pool geometry
Way-point database generator output
### Appendix B: Technical specifications of the Tethra AUV

<table>
<thead>
<tr>
<th>Dimension</th>
<th></th>
</tr>
</thead>
<tbody>
<tr>
<td>Height</td>
<td>1.2 m</td>
</tr>
<tr>
<td>Width</td>
<td>1.4 m</td>
</tr>
<tr>
<td>Length</td>
<td>2.0 m</td>
</tr>
<tr>
<td>Weight (in air)</td>
<td>180 kg (no payload)</td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>Navigation</th>
<th></th>
</tr>
</thead>
<tbody>
<tr>
<td>INS (Ixsea PHINS)</td>
<td>6 DI/DO, 2PI, 4AO</td>
</tr>
<tr>
<td>DVL (RDI)</td>
<td>2 DO (RS232)</td>
</tr>
<tr>
<td>Depth sensor</td>
<td>Digiquartz</td>
</tr>
<tr>
<td>6 Altimeters (Tritech)</td>
<td>0.2-10 m (5 units)</td>
</tr>
<tr>
<td></td>
<td>1-100 m (1 unit)</td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>Bathymetry</th>
<th></th>
</tr>
</thead>
<tbody>
<tr>
<td>Multi-beam (RESON)</td>
<td>7125</td>
</tr>
<tr>
<td>Sound Velocity Probe</td>
<td>SVP-24</td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>Communication</th>
<th></th>
</tr>
</thead>
<tbody>
<tr>
<td>Ethernet (UDP protocol)</td>
<td>Speed: 100 Mbps</td>
</tr>
<tr>
<td>Radio Control on Surface</td>
<td>format: Binary</td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>Onboard System 1</th>
<th></th>
</tr>
</thead>
<tbody>
<tr>
<td>Embedded PC104</td>
<td>166 MHz Pentium, 32 Mb RAM, 8 Mb flash memory</td>
</tr>
<tr>
<td>I/O Expansion Boards</td>
<td>3×AI, DO, counter, timer (PIC16F788A)</td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>Onboard System 2</th>
<th></th>
</tr>
</thead>
<tbody>
<tr>
<td>Embedded PC104</td>
<td>300 MHz Pentium, 128 Mb RAM, 2 Gb IDE flash drive</td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>Propulsion</th>
<th></th>
</tr>
</thead>
<tbody>
<tr>
<td>4 Thrusters (Minnkota)</td>
<td>2 Horizontal,</td>
</tr>
<tr>
<td></td>
<td>2 Vertical</td>
</tr>
<tr>
<td>4 Amplifiers (Msonic)</td>
<td>PWM 2.3 KHz</td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>Buoyancy</th>
<th></th>
</tr>
</thead>
<tbody>
<tr>
<td>Passive</td>
<td>2 fixed &amp; 2 adaptive tubes</td>
</tr>
<tr>
<td>Active</td>
<td>4×2 L ballast tanks</td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>Operation Mode</th>
<th></th>
</tr>
</thead>
<tbody>
<tr>
<td>ROV Mode</td>
<td>Aid of low-level controllers</td>
</tr>
<tr>
<td>AUV Mode</td>
<td>Tracking way-points, Obstacle avoidance</td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>Control System Architecture</th>
<th></th>
</tr>
</thead>
<tbody>
<tr>
<td>Hybrid</td>
<td>High-level hierarchical &amp; Low-level behaviour based architecture with Control allocation</td>
</tr>
</tbody>
</table>
Appendix C: Technical specifications of PHINS INS, RDI DVL and Microbath depth/altitude sensor

IxSea PHINS INS

---

PHINS INS documentation (http://www.ixsea.com/downloads/sales_documentation/uk_Phins_v0503.pdf)
## Appendix C

### Performance

<table>
<thead>
<tr>
<th>Description</th>
<th>Value</th>
</tr>
</thead>
<tbody>
<tr>
<td><strong>Position accuracy</strong></td>
<td></td>
</tr>
<tr>
<td>With stand-alone GPS aiding</td>
<td>5 to 15 m</td>
</tr>
<tr>
<td>With differential GPS aiding</td>
<td>0.5 to 3 m</td>
</tr>
<tr>
<td>With RTK differential GPS aiding</td>
<td>2 to 5 cm</td>
</tr>
<tr>
<td>With DVL aiding</td>
<td>$10^{-6}$ of traveled distance / [3 m/hr at 2 knots]</td>
</tr>
<tr>
<td>No aiding for 2 minutes</td>
<td>3 m CEP&lt;sup&gt;28&lt;/sup&gt;</td>
</tr>
<tr>
<td>No aiding for 5 minutes</td>
<td>20 m CEP&lt;sup&gt;29&lt;/sup&gt;</td>
</tr>
<tr>
<td>Pure inertial mode</td>
<td>0.6 Nm/hr</td>
</tr>
<tr>
<td><strong>Heading accuracy</strong></td>
<td></td>
</tr>
<tr>
<td>With GPS aiding</td>
<td>0.01 deg secant latitude&lt;sup&gt;30&lt;/sup&gt;</td>
</tr>
<tr>
<td>With DVL aiding only</td>
<td>0.02 deg secant latitude&lt;sup&gt;31&lt;/sup&gt;</td>
</tr>
<tr>
<td>No aiding</td>
<td>0.05 deg secant latitude&lt;sup&gt;31&lt;/sup&gt;</td>
</tr>
<tr>
<td><strong>Roll and Pitch accuracy</strong></td>
<td></td>
</tr>
<tr>
<td></td>
<td>0.01 deg</td>
</tr>
<tr>
<td><strong>Heave accuracy</strong></td>
<td></td>
</tr>
<tr>
<td></td>
<td>5 cm or 5% (whichever is highest)</td>
</tr>
<tr>
<td>Set-up free: SAFE HEAVE&lt;sup&gt;TM&lt;/sup&gt;</td>
<td></td>
</tr>
</tbody>
</table>

### Physical Characteristics

<table>
<thead>
<tr>
<th>Description</th>
<th>Value</th>
</tr>
</thead>
<tbody>
<tr>
<td>Dimensions</td>
<td>16 x 16 x16 cm&lt;sup&gt;3&lt;/sup&gt;</td>
</tr>
<tr>
<td>Weight in air</td>
<td>4.5 kg</td>
</tr>
<tr>
<td>Material</td>
<td>Aluminium</td>
</tr>
<tr>
<td></td>
<td>subsea version also available</td>
</tr>
</tbody>
</table>

### Operating / Environment

<table>
<thead>
<tr>
<th>Description</th>
<th>Value</th>
</tr>
</thead>
<tbody>
<tr>
<td>Temperature operating / Storage</td>
<td>-10 to 50 °C / -40 to 80 °C</td>
</tr>
<tr>
<td>Calibration interval</td>
<td>None required</td>
</tr>
<tr>
<td>Power consumption</td>
<td>&lt; 12 watts</td>
</tr>
<tr>
<td>Power supply</td>
<td>24 V nominal (from 20 to 30 V)</td>
</tr>
<tr>
<td>MTBF</td>
<td>30,000 hours</td>
</tr>
<tr>
<td>Angular dynamic range</td>
<td>&gt; 500 deg/s</td>
</tr>
<tr>
<td>Acceleration dynamic range</td>
<td>+/- 5 g</td>
</tr>
<tr>
<td>Attitude range</td>
<td>No limitation</td>
</tr>
</tbody>
</table>

### Interfaces

<table>
<thead>
<tr>
<th>Description</th>
<th>Value</th>
</tr>
</thead>
<tbody>
<tr>
<td>Data output rate</td>
<td>100 Hz</td>
</tr>
<tr>
<td>RS 232/ RS 422</td>
<td>7 inputs, 7 outputs</td>
</tr>
<tr>
<td>Output format</td>
<td>Based on industry standards (NMEA0183, ASCII, hexa or binary)</td>
</tr>
</tbody>
</table>

<sup>1</sup> CEP: Circular Error Probability

<sup>2</sup> Heading, Roll, Pitch figures are RMS values

<sup>3</sup> Secant latitude = 1 / cosine latitude
Appendix C

600kHz WorkHorse Navigator RDI DVL

<table>
<thead>
<tr>
<th>Model</th>
<th>390</th>
<th>600</th>
<th>1200</th>
</tr>
</thead>
<tbody>
<tr>
<td><strong>Bottom Velocity</strong></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>Single Ping Precision</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>Std dev at 1m</td>
<td>±3mm/s</td>
<td>±3mm/s</td>
<td>±3mm/s</td>
</tr>
<tr>
<td>Std dev at 5m</td>
<td>±6mm/s</td>
<td>±6mm/s</td>
<td>±6mm/s</td>
</tr>
<tr>
<td>Long term Accuracy</td>
<td>±0.5°</td>
<td>±0.5°</td>
<td>±0.5°</td>
</tr>
<tr>
<td>Minimum typical Altitude</td>
<td>1m</td>
<td>0.2m</td>
<td>0.5m</td>
</tr>
<tr>
<td>Maximum Altitude at 0°C</td>
<td>200m</td>
<td>90m</td>
<td>30m</td>
</tr>
<tr>
<td><strong>Temperature</strong></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>Velocity Range</td>
<td>±120m/s</td>
<td>±200m/s</td>
<td>±300m/s</td>
</tr>
<tr>
<td>Velocity Resolution</td>
<td>±3m/s</td>
<td>±5m/s</td>
<td>±10m/s</td>
</tr>
<tr>
<td>Power Max</td>
<td>7.5W Max</td>
<td>7.5W Max</td>
<td>11W Max</td>
</tr>
<tr>
<td><strong>Water Reference Velocity</strong></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>Accuracy</td>
<td>±0.4% ±0.1mm/s</td>
<td>±0.7% ±0.1mm/s</td>
<td>±0.7% ±0.1mm/s</td>
</tr>
<tr>
<td>Latitude</td>
<td>Selectable</td>
<td>Selectable</td>
<td>Selectable</td>
</tr>
<tr>
<td>Minimum Range</td>
<td>1m</td>
<td>0.2m</td>
<td>0.5m</td>
</tr>
<tr>
<td>Maximum Range</td>
<td>110m</td>
<td>50m</td>
<td>110m</td>
</tr>
<tr>
<td><strong>Environmental</strong></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>Operating Temperature</td>
<td>-5°C to 50°C</td>
<td>-5°C to 50°C</td>
<td>-5°C to 40°C</td>
</tr>
<tr>
<td>Storage Temperature</td>
<td>30°C to 75°C</td>
<td>30°C to 75°C</td>
<td>30°C to 75°C</td>
</tr>
<tr>
<td>Weight Max</td>
<td>1.1kg</td>
<td>1.1kg</td>
<td>6.4kg</td>
</tr>
<tr>
<td>Weight Min</td>
<td>1.1kg</td>
<td>1.1kg</td>
<td>6.4kg</td>
</tr>
<tr>
<td>DC Input</td>
<td>24 - 42V DC, external supply (40V DC typical)</td>
<td></td>
<td></td>
</tr>
<tr>
<td>Current</td>
<td>0.5A maximum power supply capacity</td>
<td></td>
<td></td>
</tr>
<tr>
<td>External D32 cables and power/serial talk</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>Peak Power @ 24VDC</td>
<td>66W</td>
<td>21W</td>
<td>8W</td>
</tr>
<tr>
<td>Average Power (typical)</td>
<td>8W</td>
<td>3W</td>
<td>3W</td>
</tr>
</tbody>
</table>

Note: Maximum before turning range may be reduced by these limits.

Navx 0Dp 9.06
Tilt: 8° up to 35°
Temperature: -5°C to 45°C

Hardware
- Configurations: 4 beam, 5 beam, 11 beam, 16 beam
- Communications: NMEA0183, ASCI, or simultaneous transducer, 30° beam angle
- Power outputs at 100W - 115W, serial port selectable: Serial port module switch selectable for RS232 or RS422
- Trigger inputs: 0-5V (or 0-10V) (Low-Latency)

3 Navigator ADCP/DVL User’s Guide
Teledyne RDInstruments ADCP/DVL datasheet (http://www.rdinstruments.com/navigator.html)

229
Appendix C

CDL Microbath⁴

⁴ CDLt MicroBath Manual v1.01
CDL MicroBath brochure (http://www.cdltd.net/products/sales_brochures/microbath_tech_brochure.pdf)
Appendix C

Altimeter Performance:

Operating Frequency: 200kHz
Beam Width: 10° Cone Nominal
Maximum Resolution: 2.5mm (0.1") / 16000 samples
Maximum Range: 500m, 300m usable (1,640ft, 985ft usable)
Minimum Range: Adjustable, Limited by pulse width
Transmit Pulse Width: Adjustable, 20 to 1000 mS
Repetition Rate: Adjustable, Limited by range

Depth Performance (Digiquartz):

Pressure Range: 0-2000psia, 0-3000psia, 0-6000psia
Repeatability: ±0.01% FSD
Hysteresis: ±0.01% FSD

Telemetry Interface: (CDL MiniSAM bi-directional RS232):
Depth Rating: 3000 meters (9845 ft)

Control Module Dimensions: 106 dia x 225mm 4.17” dia x 8.86”
Weight: 2.4kg (dry) 0.41kg (wet) 5.29 lb (dry) 0.91 lb (wet)
Altimeter Module Dimensions: 106 dia x 157mm 4.17” dia x 6.18”
Weight: 2kg (dry) 0.62kg (wet) 4.4 lb (dry) 1.36 lb (wet)
DigiQuartz Module Dimensions: 106 dia x149mm 4.17” dia x 5.87”
Weight: 1.5kg (dry) 0.19kg (wet) 3.31 lb (dry) 0.41 lb (dry)
Appendix D

Appendix D: Kalman filter (Grewal, Weill et al., 2001)

Kalman filtering is an extremely effective and versatile procedure for combining noisy sensor outputs to estimate the state of a system with uncertain dynamics.

- The noisy sensors may include position sensors and inertial sensors, but may also include speed sensors and time sensors;
- The system state in question may include the position, velocity, acceleration, attitude, and attitude rate of a vehicle, but the system state may include ancillary “nuisance variables” for modeling correlated noise sources and time-varying parameters of the sensors, such as scale factor, output bias or frequency;
- Uncertain dynamics includes unpredictable disturbances of the host vehicle, whether caused by a human operator or by the medium (e.g. winds, currents), but it may also include unpredictable changes in the sensor parameters;

The Kalman filter maintains two types of variables:

1. **Estimated State Vector.** The components of the estimated state vector include the following:
   - The variable of interest (position or velocity);
   - “Nuisance variables” that are of no intrinsic interest but may be necessary to the estimation process;
   - The Kalman filter state variables for a specific application must include all those system dynamic variables that are measurable by the sensors used in that application;
2. **A Covariance Matrix: a Measure of Estimation Uncertainty.** The equations used to propagate the covariance matrix model and manage uncertainty, taking into account how sensor noise and dynamic uncertainty contribute to uncertainty about the estimated system state.
### Appendix D

#### Predictor (Time Updates)
- Predicted state vector:
  \[ \hat{x}_k(\cdot) = \Phi_k \hat{x}_{k-1}(\cdot) \]
- Predicted covariance matrix:
  \[ P_k(\cdot) = \Phi_k P_{k-1}(\cdot) \Phi_k^T + Q_{k-1} \]

#### Corrector (Measurement Updates)
- Kalman gain:
  \[ K_k = P_k(\cdot)H_k^T(H_k P_k(\cdot)H_k^T + R_k)^{-1} \]
- Corrected state estimate:
  \[ \hat{x}_k(\cdot) = \hat{x}_k(\cdot) + K_k(z_k - H_k \hat{x}_k(\cdot)) \]
- Corrected covariance matrix:
  \[ P_k(\cdot) = P_k(\cdot) - K_k H_k P_k(\cdot) \]

The following are the names commonly used for the symbols:

- **H** is the measurement sensitivity matrix or observation matrix
- **H\hat{x}_k(\cdot)** is the predicted measurement
- **z - H\hat{x}_k(\cdot)** the difference between the measurement vector and the predicted measurement, is the innovations vector
- **\overline{K}** is the Kalman gain
- **P_k(\cdot)** is the predicted or a priori value of estimation covariance
- **P_k(\cdot)** is the corrected or a posteriori value of estimation covariance
- **Q_k** is the covariance of dynamic disturbance noise
- **R** is the covariance of sensor noise or measurement uncertainty
- **\hat{x}_k(\cdot)** is the predicted or a priori value of the estimated state vector
- **\hat{x}_k(\cdot)** is the corrected or a posteriori value of the estimated state vector
- **z** is the measurement vector or observation vector
Appendix E

Appendix E: Publications

Conference papers:


Journal papers submitted for review:


Submitted to: IEEE Journal of Oceanic Engineering  
Submission date: November 9, 2005


Submitted to: Special Issue of International Journal of Control  
Submission date: April 30, 2006