Smart Enhanced Performance
Intervention ROVs
Control system advancements for marine work-class
ROV manipulators

Satja Sivčev

Supervisors: Prof Daniel Toal
Dr Gerard Dooly
Dr Joseph Coleman

Department of Electronic & Computer Engineering
University of Limerick

Submitted to the University of Limerick for the degree of
Doctor of Philosophy

May 2018
I would like to dedicate this thesis to my loving parents Ljiljana & Goran or simply
*Mama* i *Tata.*
Declaration

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

Satja Sivčev
May 2018
Acknowledgements

I would like to express my gratitude to my supervisors Prof Daniel Toal, Dr Gerard Dooly, and Dr Joseph Coleman for giving me the opportunity to conduct this research and for their support and encouragement throughout my PhD studies. I would also like to acknowledge the assistance and support from my colleague researchers Dr Edin Omerdić, Matija Rossi, Petar Trslić, and Stefano Pierantozzi. Significant teamwork that included system integration and field testing was required to complete this research work. Therefore, I would like to thank the whole MMRRC/CRIS team for their support and assistance, but also for creating an enjoyable working environment.

For their encouragement, motivation, and support and for believing in me, I would like to thank my mother Ljiljana, father Goran, aunt Verica (Lela), grandmother Ljubica (Buba), grandfather Milan, and especially my cousin Dr Nina Dinjaški who has always been my role model in education. Finally, I would like to thank my girlfriend Dunja for her support and more importantly for her love and very much needed distraction from research life.

This material is based upon works supported by Science Foundation Ireland (SFI) under the Research Centres Award 2012, SFI Centre for Marine & Renewable Energy Research (12/RC/2302 and 14/SP/2740). The MaREI project is also supported by the following industrial partners: Resolve Marine Group, Shannon Foynes Port Company, Teledyne Blueview, Teledyne Reason and The Commissioners of Irish Lights and I sincerely thank them for their project support. The technology developed in this project has significant application potential in their sectors of marine operations.
Abstract

This thesis describes a body of research work focused on the domain of marine robotics and underwater manipulator control systems in particular. Work-class marine ROVs equipped with hydraulic manipulators are widely used in offshore industry for undersea inspection and intervention operations. In standard subsea operations setup, a human pilot employs telemanipulation technology to operate both vehicle and manipulators based on the worksite visual feedback provided by camera and sonar systems. In the emerging marine renewable energy industry, target devices are in motion due to their location in challenging environments of high energy winds, currents, and waves offshore. In such high energy sites, current commercial ROV technology capabilities are not sufficient for inspection, repair, and maintenance operations and utilising a traditional teleoperation approach is likely to fail even with very skilled pilots. This research describes the development of robotic manipulator control systems beyond the state-of-the-art capable of executing tasks in challenging conditions of the dynamic wave/current environment. A particular focus is on automation solutions that can be easily retrofitted to existing off-the-shelf underwater manipulator systems on the global fleet of commercial work-class ROVs without any hardware or software modifications. The thesis describes a developed kinematics control engine that allows human pilots to operate ROV manipulators with auto-assist, utilising enhanced manual, semi-automatic, and fully automatic (visual) servo control modes of operation while addressing stationary and moving targets. A developed collision detection and avoidance algorithm for subsea manipulators for safe, reliable, and efficient operations is also described. The control solutions for automated manipulation have been developed, implemented, and verified in simulation, dry laboratory experimental tests, and through subsea trials on a commercial work-class ROV with industry standard hydraulic manipulators. The developed control systems have a potential to reduce the task load, operational time, and costs of subsea inspection and intervention operations and significantly extend the window of operating conditions for marine ROV inspection, repair, and maintenance in the marine renewable energy and other sectors.
# Table of contents

List of figures                                                                                                               xv
List of tables                                                                  xxvii
Nomenclature                                                                   xxix

1 Introduction                                                                  1
   1.1 Background                                                               1
   1.2 Motivation                                                              2
   1.3 Aims and objectives                                                    4
   1.4 Contributions                                                           5
   1.5 Overview of chapter contents                                          6

2 Robotic arms background                                                    9
   2.1 Introduction                                                            9
   2.2 Industrial robot arms                                                   10
   2.3 Motion planning and control                                            14
   2.4 Visual servoing                                                        17
   2.5 ANFIS                                                                  23
   2.6 Concluding remarks                                                     24

3 Literature review on underwater manipulators                                  27
   3.1 Introduction                                                            27
   3.2 Definition and application                                              28
   3.3 Mechanical design                                                       29
   3.4 Actuation                                                              36
      3.4.1 Hydraulic manipulators                                             36
      3.4.2 Electric manipulators                                              38
   3.5 Commercial underwater manipulator control systems                    43
      3.5.1 Rate control                                                     43
## Table of contents

3.5.2 Position control ............................................. 44  
3.5.3 Force feedback control ..................................... 45  
3.5.4 Gripper control ............................................. 46  
3.6 Underwater manipulator motion control ......................... 47  
3.6.1 Decentralised control ....................................... 48  
3.6.2 Centralised control ....................................... 49  
3.6.3 Neural and fuzzy control .................................... 50  
3.7 UVMS motion control ........................................... 51  
3.7.1 ROV compensation for coupling effects ..................... 52  
3.7.2 UVMS motion control ....................................... 54  
3.7.3 Manipulator compensation for coupling effects .......... 57  
3.8 Kinematic control and motion planning ........................ 58  
3.8.1 Redundancy .................................................. 61  
3.9 Force control .................................................. 63  
3.10 Collision avoidance ........................................... 64  
3.11 Visual control ................................................ 66  
3.12 Implementational issues ...................................... 69  
3.13 Concluding remarks .......................................... 71  

4 Design and development of advanced control systems for underwater manipulators ........................................... 75  
4.1 Introduction .................................................... 75  
4.2 Subsea manipulation task analysis .............................. 76  
4.3 Kinematics control engine ...................................... 82  
4.3.1 Kinematic modelling ........................................ 82  
4.3.2 Forward kinematics ........................................ 82  
4.3.3 Inverse kinematics ......................................... 83  
4.3.4 Analytical inverse kinematics solution ..................... 84  
4.3.5 Numerical inverse kinematics solution .................... 91  
4.3.6 Actuator kinematics ........................................ 94  
4.4 Visual-based motion control ................................... 99  
4.4.1 PBVS algorithm — stationary targets ..................... 101  
4.4.2 Enhanced PBVS algorithm — moving targets ............ 109  
4.5 Collision detection ........................................... 115  
4.5.1 Voxel map modelling — static obstacles .................. 116  
4.5.2 Voxel map modelling — manipulators ..................... 118  
4.5.3 Voxel map modelling — manipulators’ workspaces ........ 121  
4.5.4 The collision avoidance algorithm ......................... 122
## Table of contents

4.6 Concluding remarks ................................................. 123

5 Simulation and experimental test results .......................... 125
  5.1 Introduction ...................................................... 125
  5.2 Kinematic modelling and control ................................ 126
    5.2.1 Simulations with industrial robot arms and subsea manipulators 126
    5.2.2 Experiments with industrial robot arms ........................ 128
    5.2.3 Design and development of pilot control software for subsea manipulators ........................................ 129
    5.2.4 Experiments with subsea manipulators ........................ 132
  5.3 Visual-based motion control ...................................... 139
    5.3.1 Implementation of machine vision algorithms for pose estimation ........................................ 139
    5.3.2 Simulations with industrial robot arms and a real camera ... 141
    5.3.3 Experiments of the PBVS algorithm with an industrial robot arm addressing a stationary target .............. 145
    5.3.4 Experiments of the PBVS algorithm with an underwater manipulator addressing a stationary target ............ 164
    5.3.5 Motion analysis and simulations of OC3-Hywind floating wind turbine for IRM assessment ...................... 178
    5.3.6 Experiments on replication of the OC3-Hywind floating wind turbine motion under the influence of sea waves ... 186
    5.3.7 Dry laboratory experiments with an industrial robot arm utilising PBVS algorithm to address a moving target .... 193
    5.3.8 Dry laboratory experiments with an underwater manipulator utilising PBVS algorithm to address a moving target ... 198
  5.4 Collision detection simulations and experiments ................ 218
    5.4.1 Simulations with subsea manipulators and an ROV model ... 219
    5.4.2 Dry laboratory experiment with an underwater manipulator .................................................. 223
  5.5 Concluding remarks ................................................ 225

6 Discussion, conclusion, and future work .......................... 227
  6.1 Introduction ...................................................... 227
  6.2 Summary of the thesis ............................................ 227
  6.3 Realisation of aims and objectives ................................ 229
  6.4 Contributions .................................................... 231
  6.5 Future work ....................................................... 233

References ........................................................................ 237
<table>
<thead>
<tr>
<th>Appendix</th>
<th>Description</th>
<th>Page</th>
</tr>
</thead>
<tbody>
<tr>
<td>A</td>
<td>Staubli TX60 inverse kinematics — position</td>
<td>265</td>
</tr>
<tr>
<td>B</td>
<td>Staubli TX60 inverse kinematics — orientation</td>
<td>271</td>
</tr>
<tr>
<td>C</td>
<td>Schilling Orion 7P inverse kinematics — orientation</td>
<td>277</td>
</tr>
<tr>
<td>D</td>
<td>Schilling Orion 7P actuator kinematics</td>
<td>283</td>
</tr>
<tr>
<td>E</td>
<td>Schilling Orion 7P joint offset values</td>
<td>289</td>
</tr>
<tr>
<td>F</td>
<td>Models of robotic manipulators</td>
<td>291</td>
</tr>
<tr>
<td>G</td>
<td>Key published papers</td>
<td>295</td>
</tr>
</tbody>
</table>
# List of figures

1.1 Difficulty matrix classification with increasing current strength, sea state and challenges of robotic target applications .................. 5
2.1 Relationship between forward kinematics and inverse kinematics 11
2.2 Block scheme of open-loop vision-based control with *look then move* structure. .................................................. 18
2.3 Classification of visual servoing systems by camera configuration. 20
2.4 Block scheme of closed loop position based vision control with *dynamic look and move* structure. ............................................. 23
2.5 Block scheme of closed loop image based vision control with *dynamic look and move* structure. ............................................. 23
2.6 Block scheme of closed loop position based vision control with *direct visual servo* structure. .................................................. 23
2.7 Block scheme of closed loop image based vision control with *direct visual servo* structure. .................................................. 23
2.8 ANFIS network framework architecture .................................................. 24
3.1 Factors affecting underwater manipulator performance .................. 29
3.2 Kraft Raptor, Copyright © 2017 by Kraft TeleRobotics, Inc. ........... 38
3.3 Schilling Titan 4, Copyright © 2017 by TechnipFMC plc .................. 38
3.4 Cybernetix Maestro, Copyright © 2016 by Ifremer ......................... 38
3.5 Hydro-Lek 40500R, Copyright © 2017 by Hydro-Lek ..................... 38
3.6 Eca Robotics 7E, Copyright © 2017 by ECA Group ......................... 40
3.7 Ansaldo Maris 7080, Copyright © 2006 by Autonomous Systems Laboratory .............................................................. 40
3.8 Graal Tech UMA, Copyright © 2015 by Graal Tech S.r.l. ................ 40
3.9 Hydro-Lek Pilot Console, Copyright © 2017 by Hydro-Lek ............. 44
3.10 Perry Pilot Console, Copyright © 2017 by Forum Energy Technologies 44
3.11 Schilling Rate Hand Controller – ”Bear Claw“, Copyright © 2017 by TechnipFMC plc .............................................................. 44
List of figures

3.12 Kraft Master Controller, Copyright © 2017 by Kraft TeleRobotics, Inc. ................................................................. 45
3.13 Schilling Master Controller, Copyright © 2017 by TechnipFMC plc ................................................................. 45
3.14 Perry Master Controller, Copyright © 2017 by Forum Energy Technologies ................................................................. 45
3.15 Papers referenced in this thesis that are related to subsea manipulator control systems and report results in: a) concept and theory, b) simulation, c) hardware-in-the-loop tests, d) dry laboratory tests, e) wet tank tests, f) subsea field trials in calm water, and g) subsea field trials in challenging conditions ................................................................. 73
4.1 Factors that affect the complexity of ROV manipulation tasks ................................................................. 79
4.2 ROV Holland I equipped with two Schilling Orion 7P manipulators ................................................................. 80
4.3 ROV Hercules performing sediment core sampling ................................................................. 80
4.4 Kinematic models of robotic manipulators with Denavit-Hartenberg convention coordinate frame assignment ................................................................. 83
4.5 Projection of the wrist centre on the $x_0y_0$-plane; two configurations for a given wrist centre ................................................................. 86
4.6 Projection of the wrist centre on the plane formed by links 2 and 3, four configurations for a given wrist position ................................................................. 87
4.7 A triangle showing the inscribed circle and the partitioning of the sides into segments $s - a$, $s - b$, and $s - c$ ................................................................. 89
4.8 Geometrical relationship between the linear actuator and the revolute joint ($q_1$) of the Schilling Orion 7P ................................................................. 96
4.9 Geometrical relationship between the linear actuator and the revolute joint ($q_2$) of the Schilling Orion 7P ................................................................. 96
4.10 Geometrical relationship between the linear actuator and the revolute joint ($q_3$) of the Schilling Orion 7P ................................................................. 97
4.11 Geometrical relationship between the linear actuator and the revolute joint ($q_5$) of the Schilling Orion 7P ................................................................. 97
4.12 Geometrical relationship between the linear actuator and the revolute joint ................................................................. 98
4.13 Steps to acquire joint offset values for Schilling Orion 7P manipulator model ................................................................. 99
4.14 Acquiring joint offset values for Schilling Orion 7P manipulator model ................................................................. 99
4.15 Developed PBVS algorithm scheme ................................................................. 101
4.16 Fiducial marker ................................................................. 102
4.17 Relevant homogeneous transformations for pose estimation ................................................................. 102
## List of figures

4.18 Relevant homogeneous transformations to estimate the desired end-effector motion ........................................... 105
4.19 Concept of target approaching with variable percentage of path ................................................................. 108
4.20 Relevant homogeneous transformations for target pose estimation and motion prediction ............................. 110
4.21 Block scheme of the enhanced PBVS algorithm for addressing targets in motion ........................................ 111
4.22 Block diagram of a dual manipulator control system encapsulating the proposed collision detection algorithm .................................................. 115
4.23 Meshing an ROV camera CAD model into a point cloud using SolidWorks ................................................. 117
4.24 Regular 3D voxel grid — unoccupied ........................................ 118

5.1 Staubli TX60 end-effector Cartesian position during the Scenario 1 simulation ................................................. 128
5.2 Visualisation of the virtual scene with two Staubli TX60 robots during the Scenario 1 simulation ......................... 128
5.3 Trajectory trace after the Scenario 1 simulation .................................................................................................. 128
5.4 Visualisation of the virtual environment with Staubli TX60 and Schilling Titan 2 robot models during the Scenario 1 simulation .................................................. 129
5.5 Photos taken during the Scenario 1 experiment ................................................................................................ 129
5.6 User interface of the developed LabVIEW application for control of subsea manipulators .............................. 130
5.7 End-effector position during the experiment of utilising semi-automatic Cartesian motion functions to generate a cube shaped path .................................................. 133
5.8 3D trajectory trace during the experiment of utilising semi-automatic Cartesian motion functions to generate a cube shaped path .................................................. 134
5.9 End-effector position coordinates during the experiment of utilising semi-automatic Cartesian motion functions to generate a circular path in end-effector frame .................................................................................................. 135
5.10 Trace of the 3D trajectory during the experiment of utilising semi-automatic Cartesian motion functions to generate a circular path in end-effector frame .................................................................................................. 135
5.11 End-effector position coordinates during the experiment of utilising semi-automatic Cartesian motion functions to generate a circular path in robot base frame .................................................................................................. 136
<table>
<thead>
<tr>
<th>Figure</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>5.12</td>
<td>Trace of the 3D trajectory during the experiment of utilising semi-automatic Cartesian motion functions to generate a circular path in robot base frame</td>
</tr>
<tr>
<td>5.13</td>
<td>Photos taken during the manipulation tasks of (a) relocating a metal frame and (b) removing a rope for the real world validation of manual and semi-automatic Cartesian motion functions</td>
</tr>
<tr>
<td>5.14</td>
<td>Cartesian coordinates of the reference trajectory during the rope removal manipulation task</td>
</tr>
<tr>
<td>5.15</td>
<td>Joint space reference trajectory during the rope removal manipulation task</td>
</tr>
<tr>
<td>5.16</td>
<td>3D trace of the reference trajectory during the rope removal manipulation task</td>
</tr>
<tr>
<td>5.17</td>
<td>Point Grey Bumblebee 2 stereo camera</td>
</tr>
<tr>
<td>5.18</td>
<td>Point Grey BlackFly camera with Kowa lens and Sexton subsea housing</td>
</tr>
<tr>
<td>5.19</td>
<td>Camera image of a chessboard (a) before and (b) after the calibration</td>
</tr>
<tr>
<td>5.20</td>
<td>Camera image of a fiducial marker (a) before and (b) after the detection</td>
</tr>
<tr>
<td>5.21</td>
<td>Coordinate transformations required to orient the end-effector mounted camera to point towards the fiducial marker</td>
</tr>
<tr>
<td>5.22</td>
<td>Simulations of generating reference manipulator configurations to orient the end-effector mounted camera to point towards the fiducial marker</td>
</tr>
<tr>
<td>5.23</td>
<td>Simulations of generating reference manipulator configurations to orient the end-effector mounted camera parallel to the fiducial marker</td>
</tr>
<tr>
<td>5.24</td>
<td>Simulations of generating reference manipulator configurations to position and orient the end-effector mounted camera relative to the fiducial marker</td>
</tr>
<tr>
<td>5.25</td>
<td>Stages of the manipulation task execution utilising the open-loop PBVS algorithm with <em>look then move</em> structure</td>
</tr>
<tr>
<td>5.26</td>
<td>Partitioned execution time of the open-loop PBVS algorithm with <em>look then move</em> structure and analytical inverse kinematics</td>
</tr>
<tr>
<td>5.27</td>
<td>Partitioned execution time of the open-loop PBVS algorithm with <em>look then move</em> structure and numerical inverse kinematics</td>
</tr>
<tr>
<td>5.28</td>
<td>Family of normalised distance functions depending on the PP parameter</td>
</tr>
<tr>
<td>5.29</td>
<td>Duration of the visual servoing manipulation task as a function of PP and IP parameters</td>
</tr>
<tr>
<td>5.30</td>
<td>Stages of the manipulation task execution utilising the close-loop PBVS algorithm with dynamic <em>look and move</em> structure</td>
</tr>
</tbody>
</table>
List of figures

5.31 Camera view of the manipulation task execution stages utilising the close-loop PBVS algorithm with dynamic *look and move* structure  . 156
5.32 Partitioned execution time of the close-loop PBVS algorithm with *look and move* structure and numerical inverse kinematics implementation .................................................. 157
5.33 Position error during the PBVS experiment (PP = 60% and IP = 14) 157
5.34 Orientation error during the PBVS experiment (PP = 60% and IP = 14) 157
5.35 *Robot 2* Virtual TCP motion during the PBVS experiment (PP = 60% and IP = 14) .............. 158
5.36 Target position during during the PBVS experiment (PP = 60% and IP = 14) ......................... 158
5.37 Target position error during the PBVS experiment (PP = 60% and IP = 14) .......................... 158
5.38 Joint positions during the PBVS experiment (PP = 60% and IP = 14) ................................. 158
5.39 Norm of position error during the visual servoing task with different initial positions (IP1,...,IP8) ...................................................... 160
5.40 Visual servoing experiments with different initial orientations (IO1, IO2, IO3) ..................... 161
5.41 Unsuitable camera poses in visual servoing experiment with different initial orientations (IO4) ................................................................. 161
5.42 Visual servoing experiment with different initial orientations (IO4) ................................. 161
5.43 Simulation of the camera and end-effector reference trajectories ................................. 162
5.44 Suitable camera poses in visual servoing experiment with different initial orientations (IO4) ................................................................. 163
5.45 Visual servoing experiment with different initial orientations (IO4) ................................. 163
5.46 Stages of pick and place manipulation task utilising the developed PBVS algorithm ......................... 165
5.47 Pick and place task utilising PBVS algorithm (PP = 10% and IP = 14) ............................... 166
5.48 Virtual TCP motion in pick and place task utilising PBVS algorithm (PP = 10% and IP = 14) ........................................................................ 166
5.49 Pick and place task utilising PBVS algorithm (PP = 80% and IP = 14) ............................... 166
5.50 Virtual TCP motion in pick and place task utilising PBVS algorithm (PP = 80% and IP = 14) ........................................................................ 166
5.51 Camera view during the execution of the Plugging a connector task .................................... 170
5.52 Fiducial marker (a) position and (b) orientation during the Plugging a connector task experiment ........................................................................ 170
5.53 Joint angle reference positions during the Plugging a connector task experiment ................................. 171
5.54 Camera view during the execution of the *Turning a valve* intervention task ........................................... 172
5.55 Error vector during the execution of the *Turning a valve* intervention task ........................................... 172
5.56 Convergence of the TCP towards the target during the execution of the *Turning a valve* intervention task ........ 173
5.57 End-effector position error (b) as a consequence of the joint position error (a) during the execution of the *Turning a valve* intervention task 173
5.58 Partitioned execution time of the PBVS algorithm during the visual servoing task .................................... 174
5.59 Norm of position error during the PBVS validation task. Six experiments with different Initial Positions (IP1,...,IP6) . 175
5.60 Camera view during the execution of the *Grabbing a tool* task ................................................................. 176
5.61 Image taken by the ROV mounted camera during the execution of the PBVS validation task with the target in motion, carried by Schilling Orion 7P manipulator ............................................................. 177
5.62 Position error vector during the execution of the PBVS validation task with the target in motion ......................... 178
5.63 Convergence of the TCP towards the moving target during the execution of the PBVS validation task ............... 178
5.64 OC3-Hywind floating wind turbine spar buoy (O—origin, M—mooring, B—bottom) ........................................ 180
5.65 OC3-Hywind Surge RAO ......................................................... 183
5.66 OC3-Hywind Surge Phase Shift ........................................... 183
5.67 OC3-Hywind Heave RAO ..................................................... 183
5.68 OC3-Hywind Heave Phase Shift ......................................... 183
5.69 OC3-Hywind Pitch RAO ..................................................... 183
5.70 OC3-Hywind Pitch Phase Shift ........................................... 183
5.71 OC3-Hywind (SS = 4, WP = 6 s, WH = 2.5 m): Position of the (a) origin point (O), (b) mooring fairlead point (M), and (c) bottom point (B) ................................................................. 185
5.72 OC3-Hywind (SS = 4, WP = 6 s, WH = 2.5 m): Trajectory in *xy*-plane of the (a) origin point (O), (b) mooring fairlead point (M), and (c) bottom point (B) .................................................. 185
5.73 OC3-Hywind simulation result (SS = 3, WP = 4 s, WH = 1.25 m): (M) Mooring fairlead point (a) position, (b) velocity, and (c) acceleration .................................................. 186
5.74 OC3-Hywind simulation result (SS = 3, WP = 9 s, WH = 1.25 m): (M) Mooring fairlead point (a) position, (b) velocity, and (c) acceleration ........................................ 186
5.75 OC3-Hywind simulation result (SS = 4, WP = 4 s, WH = 2.5 m): (M) Mooring fairlead point (a) position, (b) velocity, and (c) acceleration 187
5.76 OC3-Hywind simulation result (SS = 4, WP = 10.5 s, WH = 2.5 m): (M) Mooring fairlead point (a) position, (b) velocity, and (c) acceleration 187
5.77 OC3-Hywind simulation result (SS = 5, WP = 6 s, WH = 4 m): (M) Mooring fairlead point (a) position, (b) velocity, and (c) acceleration 187
5.78 OC3-Hywind simulation result (SS = 5, WP = 10.5 s, WH = 4 m): (M) Mooring fairlead point (a) position, (b) velocity, and (c) acceleration 188
5.79 OC3-Hywind simulation result (SS = 6, WP = 8 s, WH = 6 m): (M) Mooring fairlead point (a) position, (b) velocity, and (c) acceleration 188
5.80 OC3-Hywind simulation result (SS = 6, WP = 10.5 s, WH = 6 m): (M) Mooring fairlead point (a) position, (b) velocity, and (c) acceleration 188
5.81 OC3-Hywind simulation result (SS = 7, WP = 9 s, WH = 8.25 m): (M) Mooring fairlead point (a) position, (b) velocity, and (c) acceleration 189
5.82 OC3-Hywind simulation result (SS = 7, WP = 10.5 s, WH = 8.25 m): (M) Mooring fairlead point (a) position, (b) velocity, and (c) acceleration 189
5.83 OC3-Hywind tracking / motion matching experiment result (SS = 3, WP = 4 s, WH = 1.25 m): (M) Mooring fairlead point (a) position, (b) velocity, and (c) acceleration 190
5.84 OC3-Hywind tracking / motion matching experiment result (SS = 3, WP = 9 s, WH = 1.25 m): (M) Mooring fairlead point (a) position, (b) velocity, and (c) acceleration 191
5.85 OC3-Hywind tracking / motion matching experiment result (SS = 4, WP = 4 s, WH = 2.5 m): (M) Mooring fairlead point (a) position, (b) velocity, and (c) acceleration 191
5.86 OC3-Hywind tracking / motion matching experiment result (SS = 4, WP = 10.5 s, WH = 2.5 m): (M) Mooring fairlead point (a) position, (b) velocity, and (c) acceleration 191
List of figures

5.87 OC3-Hywind tracking / motion matching experiment result (SS = 5, WP = 6 s, WH = 4 m): (M) Mooring fairlead point (a) position, (b) velocity, and (c) acceleration ........................................... 192
5.88 OC3-Hywind tracking / motion matching experiment result (SS = 5, WP = 10.5 s, WH = 4 m): (M) Mooring fairlead point (a) position, (b) velocity, and (c) acceleration ........................................... 192
5.89 OC3-Hywind tracking / motion matching experiment result (SS = 6, WP = 8 s, WH = 6 m): (M) Mooring fairlead point (a) position, (b) velocity, and (c) acceleration ........................................... 192
5.90 OC3-Hywind tracking / motion matching experiment result (SS = 6, WP = 10.5 s, WH = 6 m): (M) Mooring fairlead point (a) position, (b) velocity, and (c) acceleration ........................................... 193
5.91 Robot arm following a moving target (fiducial marker) utilising the developed ANFIS enhanced PBVS algorithm ........................................... 196
5.92 Robot arm TCP following the target ........................................... 197
5.93 Absolute values of position and orientation errors ........................................... 197
5.94 Experimental results using PBVS algorithm (a) without motion prediction, (b) with numerical integration based prediction, and (c) with ANFIS based prediction. The reference target trajectory simulates (M) mooring fairlead point of OC3-Hywind floating wind turbine (SS = 3, WP = 4 s, WH = 1.25 m) ........................................... 198
5.95 Experimental results using PBVS algorithm (a) without motion prediction, (b) with numerical integration based prediction, and (c) with ANFIS based prediction. The reference target trajectory simulates (M) mooring fairlead point of OC3-Hywind floating wind turbine (SS = 3, WP = 9 s, WH = 1.25 m) ........................................... 199
5.96 Experimental results using PBVS algorithm (a) without motion prediction, (b) with numerical integration based prediction, and (c) with ANFIS based prediction. The reference target trajectory simulates (M) mooring fairlead point of OC3-Hywind floating wind turbine (SS = 4, WP = 4 s, WH = 2.5 m) ........................................... 199
5.97 Experimental results using PBVS algorithm (a) without motion prediction, (b) with numerical integration based prediction, and (c) with ANFIS based prediction. The reference target trajectory simulates (M) mooring fairlead point of OC3-Hywind floating wind turbine (SS = 4, WP = 10.5 s, WH = 2.5 m) ........................................... 199
5.98 Experimental results using PBVS algorithm (a) without motion prediction, (b) with numerical integration based prediction, and (c) with ANFIS based prediction. The reference target trajectory simulates (M) mooring fairlead point of OC3-Hywind floating wind turbine (SS = 5, WP = 6 s, WH = 4 m) ................................. 200
5.99 Experimental results using PBVS algorithm (a) without motion prediction, (b) with numerical integration based prediction, and (c) with ANFIS based prediction. The reference target trajectory simulates (M) mooring fairlead point of OC3-Hywind floating wind turbine (SS = 5, WP = 10.5 s, WH = 4 m) ................................. 200
5.100 Experimental results using PBVS algorithm (a) without motion prediction, (b) with numerical integration based prediction, and (c) with ANFIS based prediction. The reference target trajectory simulates (M) mooring fairlead point of OC3-Hywind floating wind turbine (SS = 6, WP = 8 s, WH = 6 m) ................................. 200
5.101 Experimental results using PBVS algorithm (a) without motion prediction, (b) with numerical integration based prediction, and (c) with ANFIS based prediction. The reference target trajectory simulates (M) mooring fairlead point of OC3-Hywind floating wind turbine (SS = 6, WP = 9 s, WH = 6 m) ................................. 201
5.102 Schilling Titan 2 manipulator following a moving target (fiducial marker) ......................................................... 203
5.103 Subsea manipulator TCP following the target .................. 204
5.104 Absolute values of position and orientation errors ........... 204
5.105 Histogram and normal distribution of the position error norm ................................................................. 205
5.106 Position error norm for a single experiment without ANFIS and ten experiments with ANFIS .......................... 205
5.107 Prediction error of ANFIS network trained with 50 epochs and 128 position measurements. Error of separate position coordinates and position error norm .................................................. 206
5.108 Position error norm for a single experiment without ANFIS and four experiments with ANFIS, each predicting a different future consecutive target position vector ..................................... 207
5.109 Histogram and normal distribution of the target position error for the task of following a moving target by utilising: (a) PBVS algorithm not implementing any motion prediction method; PBVS algorithm with the implementation of ANFIS network for the prediction of (b) 1st, (c) 2nd, (d) 3rd, (e) 4th future consecutive target position vector 208
List of figures

5.110 Position error norm and histogram with normal distribution during the target following task, with ROV facing the target in the (a) x-axis direction, (b) y-axis direction, (c) direction between x and y axes at approximately 45°. ................................................. 211

5.111 Target position error norm, histogram and normal distribution for the target following task using PBVS algorithm (a) without motion prediction, and (b) with ANFIS predictor. The target reference trajectory simulates the (M) mooring fairlead of the OC3-Hywind floating wind turbine (SS = 3, WP = 9 s, WH = 1.25 m) ............. 213

5.112 Target position error norm, histogram and normal distribution for the target following task using PBVS algorithm (a) without motion prediction, and (b) with ANFIS predictor. The target reference trajectory simulates the (M) mooring fairlead of the OC3-Hywind floating wind turbine (SS = 4, WP = 4 s, WH = 2.5 m) ............. 213

5.113 Target position error norm, histogram and normal distribution for the target following task using PBVS algorithm (a) without motion prediction, and (b) with ANFIS predictor. The target reference trajectory simulates the (M) mooring fairlead of the OC3-Hywind floating wind turbine (SS = 4, WP = 10.5 s, WH = 2.5 m) ........... 214

5.114 Target position error norm, histogram and normal distribution for the target following task using PBVS algorithm (a) without motion prediction, and (b) with ANFIS predictor. The target reference trajectory simulates the (M) mooring fairlead of the OC3-Hywind floating wind turbine (SS = 5, WP = 6 s, WH = 4 m) ................. 214

5.115 Target position error norm, histogram and normal distribution for the target following task using PBVS algorithm (a) without motion prediction, and (b) with ANFIS predictor. The target reference trajectory simulates the (M) mooring fairlead of the OC3-Hywind floating wind turbine (SS = 5, WP = 10.5 s, WH = 4 m) ............ 215

5.116 Target position error norm, histogram and normal distribution for the target following task using PBVS algorithm (a) without motion prediction, and (b) with ANFIS predictor. The target reference trajectory simulates the (M) mooring fairlead of the OC3-Hywind floating wind turbine (SS = 6, WP = 8 s, WH = 6 m) ............... 215

5.117 Visual inspection experiment test panel ........................................... 216
5.118 Target position error norm during the visual inspection experiment with: (a) static target, 9 points, and 50 mm error threshold; (b) static target, 27 points, and 25 mm error threshold; (c) dynamic target, 9 points, and 50 mm error threshold; (b) dynamic target, 27 points, and 25 mm error threshold. ........................................ 217

5.119 University of Limerick MRE ROV—ROV Étain .................. 218

5.120 MRE ROV modelled with voxels of different size: (a) 100 mm; (b) 33 mm, and (c) 10 mm ................................. 219

5.121 Reference input and collision-free output trajectories in joint space . 220

5.122 Reference input and collision-free output trajectories in Cartesian space ................................................... 220

5.123 Simulation of the collision detection between each manipulator and the ROV ................................................. 221

5.124 Simulation of the collision detection between two manipulators . . 222

5.125 Collision detection experimental setup with the floor as an obstacle . 223

5.126 Reference input and collision-free output trajectories from the experiment ........................................................ 224
List of tables

<table>
<thead>
<tr>
<th>Table</th>
<th>Description</th>
<th>Page</th>
</tr>
</thead>
<tbody>
<tr>
<td>3.1</td>
<td>Specifications of existing commercial underwater manipulators</td>
<td>32</td>
</tr>
<tr>
<td>3.2</td>
<td>Relative manipulator-to-vehicle weight for commercial heavy (H), medium (M) and light (L) work class ROVs</td>
<td>35</td>
</tr>
<tr>
<td>3.3</td>
<td>Actuation force comparison between hydraulic and electric ROV manipulators</td>
<td>41</td>
</tr>
<tr>
<td>3.4</td>
<td>Summary of typical ROV operated tools</td>
<td>42</td>
</tr>
<tr>
<td>4.1</td>
<td>DH parameters for Staubli TX60</td>
<td>83</td>
</tr>
<tr>
<td>4.2</td>
<td>DH parameters for Schilling Titan 2</td>
<td>83</td>
</tr>
<tr>
<td>4.3</td>
<td>DH parameters for Schilling Orion 7P</td>
<td>83</td>
</tr>
<tr>
<td>5.1</td>
<td>Organisation of sections in chapter 5 related to the specific simulation and experimental test results</td>
<td>126</td>
</tr>
<tr>
<td>5.2</td>
<td>Comparative performance analysis of the closed-loop PBVS algorithm with <em>look and move</em> structure with different combinations of IP and PP parameter values</td>
<td>151</td>
</tr>
<tr>
<td>5.3</td>
<td>Partitioned execution time of the close-loop PBVS algorithm with <em>look and move</em> structure and numerical inverse kinematics implementation</td>
<td>154</td>
</tr>
<tr>
<td>5.4</td>
<td>Irish Weather Buoy M3: Sea State occurrence (%) for the period between May 2016 and May 2017</td>
<td>184</td>
</tr>
<tr>
<td>5.5</td>
<td>Experimental results implementing ANFIS with different number of training epochs with the PBVS algorithm for the task of following a moving target</td>
<td>206</td>
</tr>
<tr>
<td>5.6</td>
<td>Robustness analysis of the ANFIS enhanced visual servoing algorithm to the change in target motion amplitude and period</td>
<td>209</td>
</tr>
<tr>
<td>5.7</td>
<td>Analysis of ANFIS enhanced visual servoing performance for potential integration of ANFIS retraining in the PBVS control loop</td>
<td>210</td>
</tr>
<tr>
<td>5.8</td>
<td>Computational load analysis of the collision detection algorithm</td>
<td>223</td>
</tr>
</tbody>
</table>
Nomenclature

Acronyms / Abbreviations

ABA  Articulated-Body Algorithm
ANFIS  Adaptive Neuro-Fuzzy Inference System
AROV  Autonomous Remotely Operated Vehicle
AUV  Autonomous Underwater Vehicle
BLDC  Brushless DC
CAD  Computer-Aided Design
CRIS  Centre for Robotics and Intelligent Systems
C – space  Configuration space
DH  Denavit–Hartenberg
DLL  Dynamic-Link Library
DOF  Degree of Freedom
GPU  Graphics Processing Unit
HPU  Hydraulic Power Unit
I – AUV  Intervention Autonomous Underwater Vehicle
IBVS  Image-Based Visual Servo
IET  International Electrotechnical Commission
INS  Inertial Navigation System
IRM  Inspection Repair and Maintenance
**Nomenclature**

<table>
<thead>
<tr>
<th>Term</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>LVDT</td>
<td>Linear Variable Differential Transformer</td>
</tr>
<tr>
<td>MaREI</td>
<td>Marine and Renewable Energy Ireland</td>
</tr>
<tr>
<td>MCU</td>
<td>Master Control Unit</td>
</tr>
<tr>
<td>MHK</td>
<td>Marine Hydrokinetic Devices</td>
</tr>
<tr>
<td>MMRRC</td>
<td>Mobile and Marine Robotics Research Centre</td>
</tr>
<tr>
<td>MRAC</td>
<td>Model Reference Adaptive Control</td>
</tr>
<tr>
<td>MRE</td>
<td>Marine Renewable Energy</td>
</tr>
<tr>
<td>msw</td>
<td>meters of sea water</td>
</tr>
<tr>
<td>NREL</td>
<td>National Wind Technology Center</td>
</tr>
<tr>
<td>O&amp;M</td>
<td>Operations &amp; Maintenance</td>
</tr>
<tr>
<td>P</td>
<td>Proportional</td>
</tr>
<tr>
<td>PBVS</td>
<td>Position-Based Visual Servo</td>
</tr>
<tr>
<td>PD</td>
<td>Proportional-Derivative</td>
</tr>
<tr>
<td>PI</td>
<td>Proportional-Integral</td>
</tr>
<tr>
<td>PID</td>
<td>Proportional-Integral-Derivative</td>
</tr>
<tr>
<td>PoE</td>
<td>Power over Ethernet</td>
</tr>
<tr>
<td>PTP</td>
<td>Point-to-Point</td>
</tr>
<tr>
<td>ROT</td>
<td>Remotely Operated Tool</td>
</tr>
<tr>
<td>ROV</td>
<td>Remotely Operated Vehicle</td>
</tr>
<tr>
<td>RSIFLC</td>
<td>Robust Single Input Fuzzy Logic Controller</td>
</tr>
<tr>
<td>SCU</td>
<td>Slave Control Unit</td>
</tr>
<tr>
<td>SDK</td>
<td>Software Development Kit</td>
</tr>
<tr>
<td>SFI</td>
<td>Science Foundation Ireland</td>
</tr>
<tr>
<td>SISO</td>
<td>Single-Input/Single-Output</td>
</tr>
</tbody>
</table>

xxx
Nomenclature

SMC Sliding Mode Controller

SPS Subsea Production Systems

TCP Tool Centre Point

UAV Unmanned Aerial Vehicle

UUV Unmanned Underwater Vehicle

UVMS Underwater Vehicle Manipulator System
Chapter 1

Introduction

1.1 Background

This thesis is a partial contribution to a large scale project carried out at the Marine and Renewable Energy Ireland (MaREI) research, development, and innovation centre supported by Science Foundation Ireland (SFI). The research work reported by this thesis comes under the umbrella of field robotics research area associated with observations and operations at marine energy sites. The field robotics related research within MaREI is carried out at the Mobile and Marine Robotics Research Centre (MMRRC)—recently renamed to Centre for Robotics and Intelligent Systems (CRIS)—at the University of Limerick. Since high energy environments prohibit the use of divers to maintain energy infrastructure, developing underwater robots and remotely operated vehicles is necessary to exploit these high-energy, hostile environments safely, reliably, and cost-effectively. The challenges addressed within the field robotics research scope include design and development of automatic and semi-automatic stabilisation and navigation solutions for underwater robots with minimal pilot input, ensuring device control, redundancy, and intuitive pilot-focused information. The core research areas of field robotics within MaREI include:

- Development of ROV, AUV, UAV, and airborne inspection technologies.
- Sensor-based approaches to monitoring and inspection, utilising acoustic and camera-based imaging for registration, positioning, navigation, and visualisation systems to facilitate ROV inspection.
- Remote presence/communications and high-speed data security provision for remote monitoring and control of robotics intervention.
Introduction

- Development of control strategies for “intervention in motion” utilising real-time video and high-resolution sonar servoing systems in the control of manipulator system and base robotic vehicles.

This thesis falls within the scope of the last core research area mentioned above and has a particular focus on control systems for manipulators on work-class ROVs.

1.2 Motivation

Ireland has exceptional Marine Renewable Energy (MRE) resources due to its convenient location and climate. Therefore, ocean energy technologies play an important role in the Irish government’s energy strategy, which targets all renewables to generate 33% of Irish electricity by 2020 (Rourke et al., 2009). While the same trend exists globally, a particular focus on MRE exists in other parts of northern Europe in which these natural resources are widespread. Offshore wind energy is already significantly contributing to the national grid in Denmark and UK (Wolfram, 2006). Additionally, the wave and tidal stream resources around the UK present an enormous potential and form crucial components of energy resources recognised by the UK government to be exploited and meet the targets for renewable energy set for 2020 and beyond.

Development of robotic capability is necessary to support large-scale MRE operations for construction/roll out, Inspection Repair and Maintenance (IRM), monitoring and control of MRE installations. Such MRE devices are by design located in the dynamic real-world environment of high energy winds, currents and waves offshore. Service robots are essential to allow the nascent MRE sector to develop and grow. Subsea robots such as advanced ROV systems and AUVs are needed to address these challenges by providing high precision surveys and stationkeeping intervention operations in high-current harsh environments (Elvander and Hawkes, 2012; Li and Bachmayer, 2013).

Current commercial intervention ROV technology capabilities as used in other sectors are not sufficient for operating in waves and currents. For example, the offshore oil and gas, marine civil engineering, and marine science sectors have utilised work-class ROVs for subsea operations for decades (Bachmayer et al., 1998; Gerwick Jr, 2007; Shukla and Karki, 2016). However, in these sectors ROV operations are in general not performed in the top 20 m to 40 m splash zone but rather the ROVs punch through the splash zone to operate in the relatively quiescent conditions below on the seabed.
1.2 Motivation

By contrast, Marine Hydrokinetic Devices (MHK) and offshore wind installations to be both viable and, ideally, cost-effective, are located in environments where wind, waves, tides, and currents are consistently energetic enough to generate electricity. While beneficial from an energy perspective, these locations can be extremely challenging from an installation and maintenance standpoint (Beaudoin et al., 2010). MRE energy farm plant may by definition be located in the splash zone in challenging wave wind tidal environments. Target plant may be in motion, and the robotic vehicles themselves must deal with the disturbances of this environment while performing intervention and inspection tasks. The IRM operational conditions for MRE will under many circumstances be above operating limits of current ROV platform technology (O’Connor et al., 2013; Omerdic et al., 2010).

The cost of Operations and Maintenance (O&M) represents a significant share of the build-up of overall offshore energy cost (Toal et al., 2011). On the other hand, learning from oil and gas sector, money lost through production losses far outweighs the costs associated with maintenance. Therefore, the majority of the cost reduction for offshore oil and gas industry is in the prevention of failure in one of the production arteries (downhole tubing, pipelines and production vessels). Thus, the development of tools to assist in the operation and IRM of ocean energy farms has been identified as a research priority. The major obstacles restricting the development of wave and tidal energy devices include high deployment and maintenance costs, due to the harsh nature of locations where ocean energy devices are deployed. The cost reduction can be realised in two ways: making offshore operations—including ROV operations—easier and more efficient, and saving in expensive support vessel time during deployment, ongoing servicing, and IRM on ocean energy installations.

The motivation thus is to research and develop ROV systems for IRM operations in currents and wave regimes of increasing strength and specifically deal with challenges in the performance of ROVs at high energy MRE sites (Toal et al., 2011). Developing such smart ROV technologies can lead to meaningful savings in O&M time and costs if they become available for wide-scale commercial and industrial use (Omerdic and Toal, 2012). Manipulators for intervention on work-class ROVs use standard, hydraulic systems and are entirely reliant on a pilot in the loop for direct teleoperation control based on scene feedback through camera and sonar systems with little/motion/disturbance of the ROV or the target infrastructure (Yuh and West, 2001). As a human operator can react only after the change has already happened, even an experienced operator is likely to fail at performing IRM operations in challenging conditions of MRE sites. With current state-of-the-art commercial ROV control systems, simple tasks from an industrial robotics perspective can become difficult even for a very skilled pilot/operator due to difficulties such as poor
visibility, poor 3D perception based on the 2D image presented on screen, and pilot fatigue. This makes subsea operations time consuming and therefore very expensive. Referring to Fig. 1.1, it is necessary to develop robot control capability to move away from the origin in the 3D plot moving along each of the axes. ROV systems ought to be more powerful, flexible, user-friendly, with improved autonomy of the vehicle and better control of subsea manipulators to allow the operator to perform complex tasks (Zhang et al., 2015). Such capability will have application in other challenging subsea applications besides MRE IRM. In fact, the number of offshore oil and gas installations, estimated to 4000 (Hegde et al., 2015), is increasing on the global level. Also, there is a trend in an Autonomous ROV (AROV) which is to reside in designated subsea docking areas and autonomously perform manipulation tasks with automatic remotely operated tools systems requiring limited operator control (Hegde et al., 2018; Schjølberg and Utne, 2015). These operations, to be performed on new Subsea Production Systems (SPS) technologies such as subsea compressors, storage, and garages, increase the need for safe, reliable, and efficient IMR systems due to increased job complexity and uncertainty, therefore, requiring a high level of autonomy and human-machine interaction in IRM (Schjølberg et al., 2016). Additionally, as pointed out by Yuh et al. (2011), autonomous subsea intervention robots would pave the way for diverse new areas of application, such as deep-ocean and under-ice exploration, tasks in hazardous and disastrous environments, both natural and man-made, automated search and surveillance missions, etc. Future robots with such advanced capabilities will allow conquering the extremes in all marine regions of the planet with fast sea currents and large waves, increasing the O&M weather window to hurricane and typhoon level (Zhang et al., 2015).

1.3 Aims and objectives

The aim is research and development of advanced ROV manipulator control systems beyond the current state-of-the-art in work-class ROVs to enable them to address the challenging conditions encountered in emerging sectors such as MRE (offshore wind, floating wind, wave energy conversion, and tidal energy conversion). Semi-automatic and fully automatic control solutions are to be designed for manipulators on intervention marine robots with vision-based servo control and integrated ROV platform and robot arm control systems capable of being operated by pilots with auto-assist in the dynamic wave/current environment. As the purpose is to apply the developed control systems in the offshore industry, these systems must be suitable
1.4 Contributions

The main contributions of the research of this thesis are in the field of marine robotics, more specifically in robot modelling, kinematic control, visual servoing, and collision detection. The main contributions are listed as follows:

- Explore existing control systems for the underwater ROV manipulators.
- Explore existing control solutions common in the industrial robotics sector.
- Identify which of these solutions are suitable and meet the defined requirements for application in ROV manipulator systems.
- Design and develop new advanced manipulation control systems for ROVs.
- Test the developed control solutions in simulation.
- Perform laboratory experiments to evaluate the developed control systems.
- Verify the performance of the developed control systems in subsea field trials.
Introduction

- Identification of a gap in the available ROV technology capability for performing close inspection and intervention operations with manipulators in the challenging, dynamic conditions of MRE sites.

- Implementation and technology transfer of industrial robotics capabilities to industry standard underwater robotic manipulators for ROVs.

- Design, development, and experimental validation of control software for ROV manipulators that enables human pilots to assume a supervisory position and utilise advanced control approaches for manipulators including enhanced manual, semi-automatic, and fully-automatic modes of operation.

- Design, development, and experimental validation of a novel visual servoing algorithm that allows performing fully-automatic inspection and intervention operations using industry standard hydraulic ROV manipulators on stationary and moving targets; therefore, applicable for IRM of MRE devices.

- Design, development, and experimental validation of a collision detection algorithm for subsea ROV manipulators that increases the safety and facilitates intervention operations.

1.5 Overview of chapter contents

Chapter 2, “Robotic arms background”, provides an introduction of fundamentals in the area of industrial robotics due to the similarity of robot arms used in manufacturing with underwater manipulator systems. In essence, this chapter represents the theoretical foundation on which the research contributions of this thesis have been built. It provides an overview of robot kinematics modelling, motion planning, and control methods, and also includes a review of visual servoing techniques and some machine learning algorithms relevant to this thesis.

Chapter 3, “Literature review on underwater manipulators”, provides a comprehensive survey of the state-of-the-art in the ROV manipulation technology. It begins with an overview of background topics related to commercially available subsea manipulator systems, and traditional and modern control approaches present in the global realm of ROVs. This chapter identifies the gaps in the capability of ROV manipulator systems with particular focus on comparison with industrial robot arms and highlights vital technology challenge areas to be addressed in research and development. The chapter also outlines previous work and recent advances in the field of autonomous underwater manipulation, addressing pertinent control topics on
1.5 Overview of chapter contents

various levels. It concludes with the discussion of the achievements in the academic sector, recognising the deficiency of adequate control solutions for industry standard hydraulic manipulator systems on ROVs to enable fully automatic inspection and intervention. The majority of the ideas for the development described in chapter 4 are a result of this review.

Chapter 4, “Design and development of advanced control systems for underwater manipulators”, by its title, gives a strong indication of its contents. This chapter describes the development of a kinematics control engine that brings underwater manipulators a few steps closer to their terrestrial counterparts. It details the implemented modelling approaches and software development for advanced pilot control of manipulators utilising enhanced manual and semi-automatic modes of operation. This chapter also proposes a novel control approach for subsea ROV manipulators to enable fully automatic visual-based intervention operations on stationary targets addressing typical tasks for ROVs. The proposed solution is extended using machine learning methods to enable ROV manipulators to autonomously address moving targets, which is vital for IRM operations in challenging conditions of the emerging MRE sector. Safety-critical concepts related to collision detection are also introduced in this chapter that can improve the performance of ROV interventions missions.

Chapter 5, “Simulations and experimental test results”, describes the steps undertaken to validate the proposed control solutions for ROV manipulator systems and evaluate their performance in performing IRM tasks typical for oil and gas, MRE sector, and other fields of application. The chapter describes the control software for industry standard hydraulic manipulators for ROVs developed as a result of this thesis, which encapsulates all control techniques proposed in chapter 4. It reports the results obtained through simulations, laboratory experiments and underwater field trials, employing the developed control algorithms on real work-class ROV systems equipped with heavy-duty hydraulic manipulators. The proposed control solutions, backed up with satisfactory experimental test results described in this chapter are the subject of a number of conference and journal publications listed in Appendix G.

Chapter 6, “Discussion, conclusion, and future work”, summarises the thesis, lists and outlines the key research contributions, and suggests further research in the related robotics area.

This thesis is supplemented with a number of appendices listed below, the subject of which is apparent from their titles.

Appendix A Staubli TX60 inverse kinematics — position
Appendix B Staubli TX60 inverse kinematics — orientation
Appendix C Schilling Orion 7P inverse kinematics — orientation
Appendix D Schilling Orion 7P actuator kinematics
Introduction

Appendix E Schilling Orion 7P joint offset values
Appendix F Models of robotic manipulators
Appendix G Key published papers
Chapter 2

Robotic arms background

2.1 Introduction

To consider that underwater manipulators are robots may be a matter of opinion and definition. However, the fact is that the serial chain kinematic structure of subsea manipulators is identical to standard robot arms used in the manufacturing industry. Therefore, to conduct research and development in the area of advanced ROV manipulation, it is worth analysing industrial robot arms as advanced systems that are, if not the same, at the least most similar to underwater manipulators.

This chapter aims to provide an essential mathematical background of robot arm systems as well as an introductory overview of robot modelling, planning, and control methodologies relevant to the research presented later in the following chapters. Any reader who is already familiar with the basics of robotics engineering may chose to skip this chapter. For detailed analysis of particular robotic fields of study such as rigid transformations, kinematic and dynamic modelling, forward and inverse kinematics, instantaneous kinematics, path planning, trajectory tracking, motion and force control, design, actuation, sensing, vision-based control, and robot programming, the reader is directed to various textbooks that address these areas (Corke, 2017; Siciliano and Khatib, 2016; Siciliano et al., 2009; Spong et al., 2006).

Two Staubli TX60 and one KUKA LBR iiwa all electric robot arms have been employed for the experimental evaluation of the developed control systems within the scope of the research work reported in this thesis.

This chapter is organized as follows: Section 2.2 provides an introduction to modelling techniques standard for industrial robot arms. Section 2.3 describes typical motion planning and control strategies, and their application. Section 2.4 analyses and classifies visual-based motion control schemes. Section 2.5 briefly describes
Robotic arms background

Adaptive Neuro-Fuzzy Inference System (ANFIS). Finally, section 2.6 summarises the robotics modelling and control techniques covered in this chapter.

2.2 Industrial robot arms

A robotic arm is an anthropomorphic\(^1\) programmable mechanical device composed of a sequence of rigid links interconnected by revolute or prismatic joints, and a gripper or other interchangeable tool attached at the end-effector. The robot arm base is considered to be stationary, fixed at a specific location in the robot cell; otherwise, it is a mobile robotic arm.

The portion of the robot’s surrounding that the end-effector can access is called the workspace. The geometrical structure of the manipulator and mechanical constraints of its joints characterise the volume and the shape of the workspace. It is often classified into a reachable workspace and a dexterous workspace; the former is a geometric locus of Cartesian points the end-effector can reach with at least one orientation and the latter with an arbitrary orientation.

The configuration of a robot arm represents the complete knowledge of the location of all points on a robot arm in a single posture, and the configuration space the set of all configurations a robot arm can assume. The number of joints of a robotic manipulator determines the number of independent parameters, known as Degrees of Freedom (DOF), that define its configuration. The configuration space or joint space of an \(n\)-DOF robotic manipulator is defined by the vector of \(n\) generalised coordinates (joint variables):

\[
q = \begin{bmatrix} q_1 & q_2 & \ldots & q_n \end{bmatrix}^T, \quad q \in \mathbb{R}^n
\] (2.1)

Industrial robot arms typically have six DOFs as that is the minimal number that can enable the end-effector to assume an arbitrary pose in its workspace (Spong et al., 2006). However, robots with seven degrees of freedom became quite common in recent years, especially in the research sector. Robot arms with more than six DOFs are inherently kinematically redundant, which means they possess a feature that can be exploited for addressing a secondary objective such as obstacle avoidance (Siciliano et al., 2009) as well as controlling the end-effector position and orientation at the same time.

Robot tasks are by definition performed by the end-effector and are, therefore, most commonly described in the operational (Cartesian) space; this is also the most intuitive approach from a human perspective. One way to define the end-effector

---

\(^1\)Anthropomorphic means having human characteristics.
2.2 Industrial robot arms

Pose in the operational space is with the following vector:

\[
x_e = \begin{bmatrix} p_e \\ \phi_e \end{bmatrix}
\]  

(2.2)

where \( p_e \) represents the \( 3 \times 1 \) end-effector position vector given in Cartesian coordinates, and \( \phi_e \) the \( 3 \times 1 \) end-effector orientation vector given by the minimal representation such as Euler angles \( \phi = [\varphi \ \theta \ \psi]^T \), both with respect to the robot base frame. Other common methods for the parametrisation of rotations are the roll-pitch-yaw \( \phi = [\varphi \ \theta \ \psi]^T \) representation, the angle-axis \((\theta, r)\) representation, and the unit quaternion \( Q = \{\eta, \epsilon\} \) representation. For computational practicality, robotics extensively uses matrix notation to express rigid motion. Therefore, the end-effector pose is often defined by the \( 4 \times 4 \) homogeneous transformation matrix, a unified mathematical description of translational and rotational displacement, given by:

\[
H_e = \begin{bmatrix} R_e & p_e \\ 0^T & 1 \end{bmatrix}
\]

(2.3)

where \( R_e \) represents the \( 3 \times 3 \) rotation matrix describing the end-effector orientation with respect to the robot base frame.

Determining the end-effector pose can be achieved by solving the forward kinematics equation, which is a function of individual joint variables, available from position sensors placed in the joints. However, solving the forward kinematics problem requires a priori knowledge of robot’s spatial geometry which is utilised to form a mathematical model of the robot. Denavit–Hartenberg convention (DH) is a relatively standard method for robot modelling that represents a systematic procedure for assigning coordinate frames to robot links, and defining methods to compute coordinate transformations between them in matrix form. Even though it is a straightforward method characterised by a distinctly defined set of rules, it does not necessarily yield a unique robot model. Some intuitive freedom in decision making is left to the one who is doing the modelling, such as choosing the origins

![Fig. 2.1 Relationship between forward kinematics and inverse kinematics](image-url)
for specific coordinate frames and selecting the orientation of some axes. The DH convention assumes assigning one coordinate frame rigidly attached to each link of the manipulator which moves with the link, and one attached to the robot’s base, which is stationary. For each robot arm link/joint, four DH parameters, describing the geometrical relations between the assigned coordinate frames, are specified; those are as follows: link length, link twist, link offset, and joint angle. For a given link, three DH parameters are constant, and one, joint angle for a revolute joint and link offset for a prismatic joint, denotes the joint variable.

The forward kinematics equation for the \( n \)-DOF robot arm modelled based on the DH convention, as a function of joint position variables \( q_i \), is given as a matrix multiplication by:

\[
H_e = \prod_{i=1}^{n} A_i^{-1}(q_i)
\]

where \( A_i^{-1} \) represents the homogeneous transformation matrix describing the pose of the coordinate frame rigidly attached to the \( i \)th link with reference to the coordinate frame of the \((i-1)\)st link. Regardless of possible differences in robot modelling, forward kinematics equation always yields a unique solution for the single robot arm.

The inverse kinematics is concerned with mapping the end-effector pose, given by the homogeneous transformation \( H_e \), to the joint position variables \( q \), and is in general, a much more complicated problem to solve compared to the forward kinematics. Robotic tasks are generally described by the desired end-effector motion in Cartesian space. To control the actuators that are driving the joints requires deriving the joint space motion that corresponds to the desired end-effector motion. Therefore, solving the inverse kinematics problem is of fundamental importance. However, inverse kinematics is a complex problem since it requires finding a solution to a system of twelve nonlinear equations with \( n \) unknown variables. It is possible that no solutions exist, e.g. if the desired end-effector pose is out of the workspace. If it is in the workspace though, there might be a single solution or multiple solutions; however, some solutions might be inadmissible due to mechanical limits of the joints. For some poses there may be infinite solutions, e.g. in cases where two or more axes of motion are aligned. In such cases, the manipulator is said to be at a kinematic singularity. In cases where solutions do exist, it is not always possible to find them analytically in closed-form. For a \( n \)-DOF manipulator, it is possible to find the closed-form inverse kinematics solution if it has a spherical wrist, which means that the last three revolute joint axes intersect at a common point. Algebraic (Stifter, 1994) and geometric methods (Lee and Ziegler, 1984) are the most common approaches for deriving closed-form analytical inverse kinematics.
solutions. The former is concerned with finding the significant equations containing unknown joint variables and rearranging them to a form that can yield a solution. On the other hand, geometric methods seek to identify those significant points on the manipulator’s structure relative to which it is possible to express position and/or orientation as a function of a reduced number of unknown joint variables. If the manipulator has a spherical wrist, this often leads to decomposing the inverse kinematics problem into two separate simplistic problems, the inverse position and the inverse orientation. One of the benefits of closed-form solutions is that they are computationally inexpensive. For some robot applications where the end-effector Cartesian trajectory is not predetermined but generated online by an external machine vision system and fed to the robot controller, this is important as the inverse kinematics equation has to be solved very fast. Additionally, if the closed-form kinematics equation yields multiple solutions, an automatic decision-making scheme can be designed based on some criteria to resolve which solution is applicable for given conditions. The disadvantage is that the closed-form inverse kinematics algorithms are derived based on the specific robot model and are, therefore, not applicable to other robots.

In those cases where the closed-form solution is impossible to derive, an alternative approach to solve the inverse kinematics problem is by utilising numerical techniques. Unlike closed-form inverse kinematics algorithms, derived for the particular robot model, numerical methods are robot model independent and applicable to any robotic manipulator; however, they are in general much more computationally intensive. The most common numerical inverse kinematics methods are Jacobian based, such as the inverse and pseudo-inverse methods (Siciliano et al., 2009, p. 133), the transpose method (Siciliano et al., 2009, p. 133), the Damped Least Squares methods (Aristidou and Lasenby, 2009; Buss and Kim, 2005; Chiaverini et al., 1994), and the feedback inverse kinematics method (Pechev, 2008). Some other existing methods are optimization approaches (Chin et al., 1997; Courty and Arnaud, 2008; Zhao and Badler, 1994), heuristic algorithms (Song and Hu, 2011), and artificial neural network approaches (Aghajarian and Kiani, 2011; Almusawi et al., 2016; Pérez-Rodríguez et al., 2012; Wei et al., 2003).

Differential kinematics is concerned with instantaneous transformations between the joint velocities and the corresponding linear and angular velocity of the end-effector. The instantaneous forward kinematics equation involves mapping the vector of joint velocities onto the vector of linear and angular end-effector velocity; it is derived as a Jacobian of the forward kinematics function, given by:

\[ v_e = J(q)\dot{q} \]  

(2.5)
where
\[ \mathbf{v}_e = \begin{bmatrix} v_x & v_y & v_z & \omega_x & \omega_y & \omega_z \end{bmatrix}^T \] (2.6)
is the spatial velocity of the end-effector expressed in the robot base frame, \( \dot{q} \) is the vector of joint velocities, and \( \mathbf{J}(q) \) the manipulator Jacobian, also called the geometric Jacobian. Computing the manipulator Jacobian for a given joint configuration is straightforward once the forward kinematics equation is derived. It is important to note that the spatial velocity vector \( \mathbf{v}_e \) is not the time derivative of the end-effector pose vector \( \mathbf{x}_e \). Mapping the joint velocities to the time derivative of the end-effector pose expressed with a minimal representation is done by differentiating the forward kinematics equation with respect to joint variables, which yields the analytical Jacobian, and is given by:

\[ \dot{\mathbf{x}}_e = \mathbf{J}_A(q) \dot{q} \] (2.7)

The analytical Jacobian is different from the geometrical Jacobian; however, it is possible to compute one from another (Spong et al., 2005, p. 140).

Often robot motion requires controlling the end-effector to move smoothly between the initial and the desired pose. This requires mapping the desired end-effector velocity profile to joint space velocities, which is the inverse instantaneous kinematics problem that can be simply solved by inverting the manipulator Jacobian:

\[ \dot{q} = \mathbf{J}^{-1}(q) \mathbf{v}_e \] (2.8)

However, the inverse solution of the Jacobian does not necessarily exist, i.e. if it is singular or not square, which is always true for a redundant robot arm. In those cases, alternative methods such as pseudo-inverse are used. If the inverse Jacobian solution does exist, the numerical inversion computation can be intensive.

### 2.3 Motion planning and control

Robot arms utilise various motion planning and control approaches depending on the complexity of the tasks they address. Motion planning is concerned with generating the reference inputs for the motion controller which drives the joint actuators ensuring that the manipulator arm moves in the planned manner.

The most straightforward motion planning approach is the Point-to-Point (PTP) method which is concerned only with the manipulator attaining the desired posture in a specified time, disregarding the locus of points the end-effector sweeps while in the transient motion. The desired robot pose can be specified either in the joint
space or the Cartesian space; with the latter employing the inverse kinematics algorithm to find the corresponding joint variables, the problem gets reduced to the former. The joint position of each actuator is then individually regulated to reach and hold the desired setpoint, or controlled to reach it while moving along a smooth trajectory. The joint motion is most often synchronised so that every robot actuator starts and stops moving at the same time. The trajectories that the joints are employed to track are usually constructed with the emphasis on minimising the actuator energy dissipation while taking into consideration the maximal velocities and accelerations the actuators can achieve, although simple position regulation using basic techniques such as PD and PID are also used. Some of the typical trajectories are the ones with smooth continuous derivatives such as cubic polynomial or quintic polynomial. A standard industry practice is to use linear segments with parabolic blends which has a trapezoidal velocity profile; it includes a constant acceleration segment to achieve the desired velocity, a segment with constant velocity, and lastly a constant deceleration segment just before reaching the desired setpoint. Finally, minimum time trajectories are employed characterised by maximum achievable constant acceleration. Rather than the specified time interval for the PTP motion, robot programming environments most often require specifying a percentage of the maximum achievable or allowable velocity and acceleration of the robot arm as a whole. These parameters are then used to generate the motion trajectory profiles for the individual joints. A typical application for PTP motion planning is spot-welding, or pick and place task, manipulating target objects with a vacuum suction cup or a mechanical gripper in an obstacle-free worksite. Often, a human operator uses a teaching pendant to manipulate the robot arm to the desired locations and store the corresponding Cartesian end-effector pose or joint variables, which are afterwards used as input reference for the robot controller in automatic mode. Storing the Cartesian position and orientation rather than the joint position values may result in manipulator attaining a configuration different to the one used in the teaching stage. Hence, some robot systems allow complementing the stored end-effector pose with one or more user permitted configurations. For applications in complex environments in which robot arms operate, robot programming suites are used which offer detailed CAD modelled robot cell environment and full simulation of robot operations. The desired setpoints are hence determined from the visualised 3D scene of the robot’s work area. When the locations of the target objects for manipulation are not known, external sensors and camera systems are used to estimate them.

Storing a certain number of robot arm configurations in a sequence is the standard approach to enhance the PTP motion planning, primarily to ensure that the manipulator does not collide with the obstacles in its environment. These via points
Robotic arms background

(Spong et al., 2005, p. 196) can include a number of intermediate configurations between the initial and the desired pose or can form a densely constructed path with many configurations the robot arm has to attain sequentially. Spline techniques are used to blend the motion of the low polynomial trajectories employed between every two adjacent postures.

Some advanced robot tasks such as arc welding or spray painting require continuous end-effector motion, so PTP motion planning may not be suitable for them. Emphasising the difference between a path and a trajectory is essential. A path is a purely geometric locus of points through which the robot end-effector has to pass with or without the possibility of stopping. The trajectory, on the other hand, is parametrised by time, which means it is a path constrained with position, velocity, and acceleration functions of time.

A sequence of end-effector poses often specifies the path that robot has to follow. End-effector motion along a straight line or an arc are the most common applications of continuous path motion planning. Additionally, combining multiple arcs and straight line paths allows constructing complex curved motion paths. The majority of industrial robot arms support recording the path by moving the manipulator with a teaching pendant, while some advanced collaborative robots allow the manipulator to be physically hand guided by the operator for the same purpose.

Trajectory tracking is the most advanced approach used for higher level manipulation tasks, and addressing it requires the most sophisticated motion control.

The task of a robot arm controller is to make sure that the joint driving actuators produce adequate torques/forces to satisfy the reference motion planning inputs. In general, a robotic manipulator is a multibody system and the dynamic equations that describe it are nonlinear and multivariable. A controller has to be capable of tracking the reference motion and dealing with the external disturbances at the same time. Various segments of the system introduce complexity to designing a controller; the type of drive system, for instance, can effect the system’s dynamics. Robot arms with direct drive actuators are highly nonlinear systems due to the significant kinematic and dynamic coupling occurring between the joints due to the time-varying manipulator configuration. On the other hand, joint actuators with high gear reduction ratios linearise manipulator system dynamics and decouple the joints, reducing the nonlinearities. However, the downside is the introduction of undesired effects such as joint friction, elasticity, and backlash. Two strategies for developing control laws applicable to manipulator systems exist. The first approach, known as the decentralised scheme, regards the manipulator as a mechanical system composed of multiple independent systems defined by the number of its joint actuators. Each joint axis is controlled individually, and the coupling effects between the joints
are treated as disturbances each joint controller has to deal with. Examples of decentralised control laws that are capable of tracking time-varying trajectories are the feedforward compensation control law and the computed torque control law. On the other hand, centralised control strategies take dynamic coupling effects between the joints into consideration for controller design. PD with gravity compensation control and inverse dynamics control are some of the centralised control laws that are suitable for trajectory tracking applications. However, all of these mentioned control laws require accurate knowledge of the parameters of the system dynamics. Targeting advanced tasks that may include the manipulator tracking a reference motion and lifting and manipulating unknown loads at the same time, requires advanced control algorithms such as robust and adaptive control algorithms.

The above has covered the essential segments of manipulator modelling, planning, and control from the control engineering point of view. However, a robotic system is comprised of many other components besides the manipulator arm and the control unit. It also includes an external power source, end-effector tools, internal and external sensors, user interface including computer and teaching pendant, and finally the programming software. These are all critical parts that effect the performance of the robot arm system. Rather than review each of these robotic subsystems, the following sections review material in robotics that is particularly pertinent to the research of this thesis.

2.4 Visual servoing

Cameras are in general considered universal sensors, which is why they are so widespread in control applications. Fundamentals related to camera imaging geometry and mathematical modelling, as well as other computer vision basics such as image formation, image processing, feature extraction and segmentation, stereo vision, etc. are available in (Hartley and Zisserman, 2004; Szeliski, 2010), while its application in robotics is comprehensively introduced in (Corke, 2017). This section provides an introduction and overview of visual-based control techniques for robot arms. The general visual servoing problem is defined, followed by various classifications of visual-based control algorithms.

Vision-guided robot control has been an active research topic for more than four decades. Shirai and Inoue (1973) are among the first to propose using a vision system to estimate the end-effector positioning error and reduce it by feeding it back to the robot arm motion controller. The technical development achieved over the years enabled excellent research progress as well as commercialisation in this sector.
Robotic arms background

![Block scheme of open-loop vision-based control with look then move structure.](image)

Today, machine vision cameras are underlying sensors used in vision-based control systems in the manufacturing industry. These imaging systems provide “eyesight” to robot arms, enabling them to determine the position and orientation of work parts arriving for example on a conveyer, and thus perform gripping, relocating, reorienting, sorting, palletising, depalletising, and other actions.

The majority of industrial applications utilise vision system measurements to generate a “one-off” set command for the robot arm motion controller without real-time closed-loop feedback control. The performance of such open-loop control systems with *look then move* structure (Fig. 2.2) (Hutchinson et al., 1996) is purely a function of the sensing accuracy of the vision system and the positioning accuracy of the robot arm. However, visual servoing by definition assumes closing the loop with visual feedback to control the pose of the end-effector of the robotic manipulator relative to a target object or a set of target features (Corke, 1993).

Visual feedback control or vision-based control utilises the ability of the imaging system to collect geometrical and qualitative information from the robot’s environment without physical interaction and processes it accordingly to drive the end-effector to approach, follow, and perform specified operations on stationary or moving targets. It encapsulates sensing, motion planning, and control of a robot arm.

The direct measurements a vision system provides is a set of parameters in the image coordinate system called pixels; a matrix of light intensity values for a monochrome camera, and a matrix of a triplet of RGB values for a colour camera. Robot arm tasks are usually described in the operational space by a constant or time-varying end-effector pose the robot arm has to attain relative to the observed target object. The camera image containing the target, represented by pixels, is a function of that relative pose. Therefore, to control the motion of the robotic manipulator in the desired manner, a relevant and robust set of parameters which describe the object of interest in the scene, called visual features, has to be extracted from the image and transformed into adequate motion planning parameters using image processing and computer vision algorithms. Doing this requires establishing a relationship between the parameters in the image space and the geometrical parameters of the robot’s environment.

There are many different ways to classify vision-based robot control systems. Some of these include classification based on the type of vision system, the number
of visual sensors used, and their location. Others are concerned with the structural design of the control system, image processing algorithms, feature extraction and interpretation.

Machine vision cameras are widely used vision systems in robotics and automation. In essence, these cameras are hardware wise very similar to the standard consumer cameras. They often have identical sensors, although machine vision cameras tend to have lower resolution. Depending on the industrial conditions and the task itself, these cameras can come with a specialised housing, low light lens, specified water resistance, G-force resistance, and other requirements. The main difference is that machine vision cameras integrate a dedicated microcontroller programmed to execute specific computer vision algorithms and provide useful outputs including the image, according to a standardised industrial communication protocol. Moreover, machine vision cameras usually come with a Software Development Kit (SDK) that enables integrating them with other parts of the automated system and allows an external program to control the camera.

Single monocular camera vision systems are the most affordable and straightforward to integrate, and therefore, the most often used. However, multiple camera systems also exist and are used for specific applications. As for the type, besides monocular cameras, stereo vision systems are also used. A stereo camera is, in fact, a system of two monocular cameras in a stereo configuration, with the known baseline between them, usually acquired through the calibration process, and with an overlapping field of view. The main advantage of stereo vision is the possibility to use two images captured by different cameras at the same time to estimate the pose of the target object relative to the camera without the knowledge of the object’s dimensions. In effect, in combination with appropriate software, stereo cameras can be considered as 3D vision sensors. The same result can be achieved with a single monocular camera only if the observed object’s dimensions are known, or if it has unique features of known shape and size. However, there are methods to estimate the full pose of pixels on the image with a single camera such as structure from motion, but these are complex, computationally expensive, and require multiple sequential images acquired (Saputra et al., 2018).

When it comes to the location of the camera in the visual servoing system, two distinct configurations are standard, “eye-in-hand” and “eye-to-hand” configurations (Corke, 2017). The former configuration, also found in the literature under the names “in-hand”, “mobile”, and “endpoint-closed-loop”, assumes that the camera is mounted on the manipulator and moves with it. During the visual servoing, the camera observes only target objects. Most commonly, it is mounted on the end-effector, but it can also be fixed to other links, frequently to the one preceding the wrist. The pose
of end-effector relative to the camera is usually constant (end-effector mounted) and has to be known. It can be approximated from the assembly CAD model of the whole system or acquired by a “hand-eye” calibration process. During the manipulator motion, the targets in the field of view of the camera change significantly, which leads to high variability in the accuracy of measurements acquired by the vision system. However, as the camera approaches the observed object the measurement accuracy increases, which can be useful for some applications. Another advantage of this configuration is the absence of potential object occlusions by the manipulator.

On the other hand, “eye-to-hand” systems, which are also known in the literature by terms “out-hand”, ”fixed”, “endpoint-open-loop”, and “stand-alone”, have a stationary camera located on a known location in the robot’s environment, oriented for observing the workspace of the robot arm. In this configuration, the pose of the camera relative to the manipulator’s base is constant. Both target objects and end-effector are observed during the visual servoing. The field of view of the camera does not change during the motion of the manipulator, which implies that the accuracy of image measurements is constant throughout the task execution. However, the downside of this configuration is that the robot arm may occlude the view of the observing objects by moving between them and the camera. This is one of the reasons for considering having multiple “fixed” cameras for observing the work scene from different locations so that at least one can detect the target object at any time regardless of the posture of the robot arm.

There also exist hybrid solutions that consist of one camera in eye-in-hand and one in eye-to-hand configuration. Such a configuration exploits the benefits of both systems, close distance accuracy of the former and global sight of the scene of the latter (Flandin et al., 2000; Lippiello et al., 2005). There are also redundant multi-camera systems (Kragic et al., 2002) comprised of three and more cameras in any configuration (Weber and Kühnlenz, 2010). Fig. 2.3 gives an overview of possible combinations regarding the number of cameras used and their configuration in the visual servoing system.
2.4 Visual servoing

According to the taxonomy of visual servoing systems (Sanderson and Weiss, 1980), there are two main categories classified based on the control scheme architecture. These are vision-based control systems with the dynamic look and move structure and with the direct visual servo structure (Corke, 1993), which are also known as indirect visual servoing systems and direct visual servoing systems, respectively (Malis, 2002).

The main characteristic of the direct visual servoing systems is the direct utilisation of visual measurements in the feedback control loop. These systems do not make use of joint position feedback, the internal joint servo control is removed, and machine vision is utilised directly to stabilise the manipulator (Alepu et al., 2016). The visual features are extracted from the image and transformed into joint torques or forces and after that mapped onto reference control input signals for power amplifiers driving joint actuators (Reyes and Kelly, 1998). This control design is suitable for high-speed robot tasks (Pomares et al., 2011) as visual servoing has to be performed at a rapid update rate, at least every $10 \text{ ms}$ (Malis, 2002).

On the other hand, vision-based control systems with dynamic look and move structure have a hierarchical architecture and utilise joint position feedback to stabilise the robot according to the reference inputs provided on a higher level by the machine vision system. These systems are characterised by a rapid sampling rate for internal motion control and a significantly lower sampling rate for external visual control. This control design assumes that the inner motion controller has idealised axis dynamics, considering the manipulator as an ideal positioning device (Corke, 1993).

The majority of vision-based control solutions for robotic manipulators proposed in the literature have dynamic look and move structure. One of the main reasons is that off-the-shelf manipulators already have a low-level motion controller with the interface accepting velocity or position reference inputs, in Cartesian or joint space (Hutchinson et al., 1996), which is handy for the development of a visual servoing system. Another important reason is that computer vision algorithms are computationally intensive and the principal cause of latency introduced in the visual-based control systems, which makes the machine vision the weakest point of the visual servoing system (Castelli et al., 2017). Poor update rates available from the vision system make designing direct visual servoing systems a very challenging control problem because of complex nonlinear dynamics. Significant delays are unacceptable for robot arms, which are inherently real-time systems. This machine vision shortfall in the systems with the look and move structure is therefore compensated by the inner loop motion controller with a rapid update rate.
Robotic arms background

Even though the majority of visual-based control systems proposed in the literature drop into the *dynamic look and move* category, the term *visual servoing* is generally used for any robotic system that utilises machine vision in a closed loop control system (Corke, 1993).

Another significant classification of visual servoing systems is based on whether the control is performed in Cartesian space or image space.

The classical and probably the most intuitive way from a roboticist’s standpoint to solve the problem of controlling a manipulator to align its end-effector with a target at an unknown location is first to estimate the target’s pose relative to the camera and then control the robot arm utilising traditional methods. This approach comes under the first visual-based control system classification category, the Position-Based Visual Servo (PBVS) control (Lippiello et al., 2007; Martinet and Gallice, 1999). In PBVS control the extracted visual features from the image are used in combination with the observed object’s geometrical model to estimate the pose of the target relative to the end-effector. Once the pose is estimated, it is used together with the specified desired pose to define the operational (Cartesian) space control variable, and consequently, implement the control law in Cartesian space to control the motion of the manipulator so that its end-effector moves from the initial to the desired pose. PBVS most often requires a calibrated camera, knowledge of targets geometry, and observed image plane features. It is sensitive to the calibration errors. If the camera calibration is not perfect or if there are errors in the target object’s 3D model, this will influence the pose estimation accuracy. The drawback of PBVS is that the image features used in the pose estimation might leave the camera field of view since there is no control in the image space, leading to the opening of visual feedback loop because of the lack of visual measurements and eventually to servoing failure. However, since PBVS acts directly on operational space variables, directly controlling the camera’s trajectory in Cartesian space, with appropriate trajectory planning this risk may be avoided.

The second category of classification is called the Image-Based Visual Servo (IBVS) control. In IBVS control the pose estimation step is skipped, and the extracted visual features such as coordinates of points of interests, parameters of straight or curved lines, circles, or ellipses, and regions are used directly to control the end-effector motion to align it with the observing target object (Kragic et al., 2002). Hence, the control law is implemented in the image space with the error variable constructed based on the comparison of the current image feature parameters to the desired image feature parameters.

Both PBVS and IBVS methods can be designed with a *dynamic look and move* or a *direct visual servo* control structure (Figs. 2.4 to 2.7). There also exist hybrid
visual servoing schemes (Deng, 2004; Hafez et al., 2008; Kermorgant and Chaumette, 2011), which combine characteristics common to both PBVS and IBVS.

Fig. 2.4 Block scheme of closed loop position based vision control with *dynamic look and move* structure.

Fig. 2.5 Block scheme of closed loop image based vision control with *dynamic look and move* structure.

Fig. 2.6 Block scheme of closed loop position based vision control with *direct visual servo* structure.

Fig. 2.7 Block scheme of closed loop image based vision control with *direct visual servo* structure.

### 2.5 ANFIS

The research that this thesis encapsulates makes use of artificial intelligence systems, specifically of ANFIS; it is not involved in further advancements in this area. For that
Robotic arms background

reason, this section presents only a brief introduction to ANFIS. ANFIS represents the adaptive neural network framework designed to be functionally equivalent to a fuzzy inference system (Jang, 1993). Some basics on neural networks can be found in (Müller et al., 2012), and on fuzzy control in (Michels et al., 2007). Its advantage is that, through a hybrid learning algorithm common for neural networks, ANFIS can automatically identify a set of fuzzy inference system if-then rules and appropriately tune the membership function parameters from an existing data set that consists of input-output pairs. Fig. 2.8 illustrates the ANFIS network architecture that consists of five layers, \( m \) inputs (X), each assigned with \( n \) fuzzy membership functions described with linguistic labels (A), constituting \( r \) rules (R). In this figure, adaptive nodes are represented by circles and fixed nodes by squares. The first layer is called the “fuzzification” layer. Each node in this layer specifies the degree to which a given input satisfies the fuzzy membership function associated with that node. Parameters in this layer are called premise parameters. The second layer involves determining the firing strength for each rule. The nodes in this layer are labelled with \( \pi \), indicating that they usually perform as a simple multiplier. The third layer involves normalising the firing strengths produced by the previous layer. The fourth layer is the so-called “defuzzification” layer; it involves computing the weighted consequent value for each given rule. Parameters in this layer are called consequent parameters. Lastly, the node in the last layer performs the summation of all incoming signals. ANFIS uses a hybrid learning algorithm to identify and tune consequent and premise parameters.

2.6 Concluding remarks

This chapter has provided the fundamental mathematical background of industrial robotic arm systems and a review of robot modelling, planning, and (vision) control.
2.6 Concluding remarks

methodologies. The addressed industrial robotics techniques are relevant and repre-
sent the basis of the underwater manipulation research and development, which is
presented later in this thesis in chapter 4 and 5. The next chapter, takes a detailed
look of the manipulator systems employed in underwater robotics.
Chapter 3

Literature review on underwater manipulators

3.1 Introduction

A brief overview on subsea manipulators can be found in the underwater robots review paper by Yuh and West (2001), and more recently in an ocean engineering book chapter by Kim et al. (2016). Antonelli (2014) provided a solid theoretical background for underwater manipulators from the modelling and control point of view. However, a complete survey article encapsulating relevant practical and theoretical knowledge, state-of-the-art in ROV manipulation technology as well as up to date research done in this area cannot be found in the literature. Therefore, this chapter aims to provide a review of the state-of-the-art in underwater manipulator technology covering all relevant aspects.

The chapter is organized as follows: Section 3.2 provides an introduction to underwater manipulator systems and their application. Section 3.3 describes mechanical design features and capabilities of existing underwater manipulators and compares them. Section 3.4 analyses underwater manipulator actuation methods. Section 3.5 describes control systems of commercially available subsea manipulators. Sections 3.6 and 3.7 cover academic research achievements in the area of motion control for underwater manipulators and underwater vehicle-manipulator systems respectively. Section 3.8 covers kinematic control and motion planning algorithms, while section 3.9 focuses on force control algorithms. Section 3.10 describes collision detection and avoidance algorithms. Visual based motion control algorithms are covered in section 3.11. Section 3.12 analyses the implementation issues of various solutions within real ROV manipulator systems. Finally, section
3.13 summarises the state-of-the-art reviewed in this chapter and presents the areas for further development.

3.2 Definition and application

A robotic manipulator is considered to be the most suitable tool for executing subsea intervention operations. Hence, Unmanned Underwater Vehicles (UUVs) such as ROVs and in some cases, AUVs are equipped with one or more underwater manipulators. UUVs with manipulators are often called Underwater Vehicle Manipulator Systems (UVMS). The majority of existing underwater manipulators used on UUVs are anthropomorphic. These mechanical devices are comprised of a sequence of rigid links interconnected by revolute joints with a suitable angular displacement between them and have grippers or other interchangeable tools attached at the end-effector. For the observation of their surroundings, underwater manipulators are accompanied with additional equipment including one or more cameras and spotlights mounted on the subsea base vehicle and on the manipulator itself.

Underwater manipulators are used for a variety of subsea intervention tasks in different applications within offshore oil and gas, maritime defence, marine civil engineering, marine science and recently in the emerging marine renewable energy sector (Capocci et al., 2018; Toal et al., 2011; Yuh et al., 2011). A wide range of subsea tasks undertaken by ROVs is done using underwater manipulators, including pipe inspection (Christ and Wernli, 2014), salvage of sunken objects (Chang et al., 2004), mine disposal (Djapic et al., 2013; Fletcher, 2000), surface cleaning (Davey et al., 1999), valve operating (Hopper, 1990), drilling (Crook, 2010; Crowhurst and Lowe, 2011), rope cutting (Christ and Wernli, 2014), weld inspection (Jeppesen et al., 2005; Martin, 2011), cable laying and repair, clearing debris and fishing nets, biological (Jones, 2009) and geological sampling (Noé et al., 2006), archaeological work (Coleman et al., 2003), etc. A variety of subsea manipulator design implementations exists due to their use in a wide range of applications. There are manipulators with limited mobility equipped with grippers for lifting large, heavy objects, manipulators used for fixing a detachable gripper to a selected, sunken object. Then, there exist grabber manipulators equipped with grippers or vacuum cups for fixing an underwater vehicle to submerged structures or near flat walls to stabilise the base robot during the operation, manipulators equipped with inspection devices (Asokan et al., 2005), dexterous light manipulators with grippers that can carry diver tools (Gancet et al., 2016). The ones that do most of the heavy work are intervention manipulators with grippers that can handle different ROV operated tools for repair
3.3 Mechanical design

and maintenance operations on submerged structures. The majority of work-class ROVs are equipped with one advanced, dexterous seven-function manipulator and one simple, powerful five-function grabber arm. The supporting manipulator is used to anchor the ROV onto the hydroengineering structure or wreck on which the intervention operation is to take place while the other manipulator performs the actual intervention task. However, some work-class ROVs have a second, advanced seven-function manipulator instead of a grabber arm. In general, the manipulators are located at the front side of UUVs, but this is not always the case, e.g. there are vehicles with a manipulator located at the bottom side (Ribas et al., 2012). The complete manipulator system, in addition to the robotic arm itself, consists of many other subsystems which affect the performance of the manipulation. These factors, outlined in the Fig. 3.1, are expanded upon in detail within the following sections.

3.3 Mechanical design

To be able to operate in deep waters and cope with the harsh conditions of the subsea environment, the construction of underwater manipulators utilises specialised materials. Additionally, depending on the task for which they are designed, under-
Literature review on underwater manipulators

Underwater manipulators have to meet relevant requirements regarding the size of the workspace in which they are to operate, lifting capacity, wrist torque, etc. Table 3.1 lists specifications of the majority of existing commercial underwater manipulators.

The most common materials used in the construction of underwater manipulators are metal alloys such as titanium Ti 6-4, anodised aluminium alloys (5083, 6082 T6, 6061 T6, 7075 T6, A356), stainless steel alloys (316, 630, 660), as well as some plastics (Polyethylene). The properties of these materials are relatively high strength and corrosion resistance and excellent machinability. Some experiments of using buoyant materials on underwater manipulators have been done to reduce the weight in the water and minimise the actuator load (Ishimi et al., 1991). Typically, commercially available underwater manipulators are rated between 3000 and 6500 meters of seawater (msw); however, some manipulators can operate at depths up to 7000 msw, e.g. Schilling Robotics Titan 4 and a prototype manipulator developed by (Zhang et al., 2014). Additionally, there are some systems designed for full ocean depth (11 000 msw). Woods Hole Oceanographic Institute in collaboration with Kraft Robotics designed one such manipulator for the Mariana Trench exploration mission (Bowen et al., 2008). Others include “Magnum 7”, a product of ISE Ltd. and, “The ARM” and “MK-37” developed by Western Space and Marine, Inc. A parameter called “reach” typically defines the size of underwater manipulators; it stands for the length of the whole manipulator kinematic chain. Together with the range of motion of joints, it determines the size of the workspace of a manipulator, a set of points that can be reached by its end-effector (Cao et al., 2011). The reach of commercial underwater manipulators ranges from 0.5 m for grabber arms up to 2.4 m for heavy duty manipulators. Maximum wrist torque which underwater manipulators are capable of producing spans from 8 N m to 250 N m. According to the petroleum and natural gas industries 13628-8:2002 (2002) standard, rotary low torque ROV interfaces on SPS, which are typical for subsea tree needle valves, are rated to maximum 75 N m. The lifting/carrying (payload) capacity for underwater manipulators ranges from 5 kg up to 500 kg. Manufacturers often provide different parameters for manipulator lift capacity (“max. nominal”, “at full extension”, “at rated speed”, “through envelope”, etc.) which makes the comparison non-trivial as the carrying capacity is not a constant value but depends on the posture of the manipulator. Underwater manipulators can be equipped with various types of grippers on the end-effector. The commercial ones come with different interchangeable grippers each of which has their specific purpose. A common gripper type is one with parallel acting jaws which includes a slot for a standard T-bar handle (13628-8:2002, 2002), and its primary function is grasping different objects and tools in a variety of subsea operations. Therefore, tooling is designed with a T-bar precisely for this purpose. Another very
3.3 Mechanical design

common grippers for ROV manipulators are three and four finger intermeshing jaws, which are also standardised for handling ROV operated tools. Other grippers include two/three finger floating jaws, scissor jaws, suction feet, etc. Gripper actuators are usually hydraulic and the gripping force of commercially available grippers ranges from 35 kgf (343 N) to 625 kgf (6129 N). For certain intervention operations, an ROV manipulator has to handle several tools. Therefore, a subsea manipulator tool changer exists that enables swapping ROV operated tools subsea (McCoy Jr, 2014, 2015; Williams, 2017). It has a standardised tool latch, where standardised tools have adequate latch receiver. Subsea manipulators weigh (in air) between 6 kg and 150 kg; however, their weight in water is more important, as it determines the buoyancy needed on the base vehicle to compensate for the manipulator. The weight and size are critical factors as they are directly responsible for the amount of dynamic coupling introduced between the manipulator and the underwater base vehicle on which it is mounted and can thus influence the performance of the whole system. To be able to exploit manipulator characteristics fully, the manipulator weight should be a low enough percentage of the whole underwater vehicle weight. In that case, the dynamic coupling can be neglected or taken into account as an external disturbance that can be dealt with by the underwater vehicle dynamic positioning system, if it exists. Higher weight and bigger size bring about higher demands concerning the robustness of underwater vehicle thruster system to the disturbance caused by the dynamic coupling. Table 3.2 presents relative manipulator-to-vehicle weight for the typical commercial heavy, medium and light work class ROVs. It can be seen that this ratio is significantly low even for the light work-class commercial vehicles.
<table>
<thead>
<tr>
<th>Manufacturer</th>
<th>Model</th>
<th>Actuation</th>
<th>DOF</th>
<th>Weight in air (kg)</th>
<th>Weight in water (kg)</th>
<th>Lift capacity max nom. (full ext.) (kgf)</th>
<th>Wrist torque (Nm)</th>
<th>Grip force (kgf)</th>
<th>Depth rating (m)</th>
<th>Max. reach (m)</th>
<th>Power Source</th>
<th>Material</th>
<th>Actuators</th>
<th>Sensors</th>
<th>Control System</th>
<th>Price ($)</th>
</tr>
</thead>
<tbody>
<tr>
<td>Ansaldo</td>
<td>MARIS 7080</td>
<td>Electric</td>
<td>7</td>
<td>65</td>
<td>45</td>
<td>85 (96)</td>
<td>190</td>
<td>150</td>
<td>100</td>
<td>2.4</td>
<td>4500</td>
<td>Al</td>
<td>BLDC</td>
<td>Resolvers, Semi-Automatic</td>
<td>F/T</td>
<td>~110k</td>
</tr>
<tr>
<td>Cybernetix</td>
<td>Maestro</td>
<td>Hydraulic</td>
<td>6</td>
<td>85</td>
<td>65</td>
<td>100 (96)</td>
<td>190</td>
<td>150</td>
<td>6000</td>
<td>2.4</td>
<td>4500</td>
<td>Al</td>
<td>BLDC</td>
<td>Resolvers, Semi-Automatic</td>
<td>F/T</td>
<td>~110k</td>
</tr>
<tr>
<td>Ec-Hytec</td>
<td>Arm TE</td>
<td>Electric</td>
<td>6</td>
<td>69</td>
<td>49.2</td>
<td>40 (40)</td>
<td>25</td>
<td>30</td>
<td>20.4</td>
<td>6000</td>
<td>1.4</td>
<td>72VDC</td>
<td>Al BLDC</td>
<td>BLDC in oil</td>
<td>/</td>
<td>/</td>
</tr>
<tr>
<td>Ec-Hytec</td>
<td>Arm SE Mini</td>
<td>Electric</td>
<td>6</td>
<td>51</td>
<td>30</td>
<td>25 (25)</td>
<td>25</td>
<td>30</td>
<td>20.4</td>
<td>6000</td>
<td>1.4</td>
<td>72VDC</td>
<td>Al BLDC</td>
<td>BLDC in oil</td>
<td>/</td>
<td>/</td>
</tr>
<tr>
<td>Ec-Hytec</td>
<td>Arm SE Micro</td>
<td>Electric</td>
<td>4</td>
<td>27</td>
<td>18.5</td>
<td>25 (25)</td>
<td>10</td>
<td>10</td>
<td>20.4</td>
<td>6000</td>
<td>1.4</td>
<td>72VDC</td>
<td>Al BLDC</td>
<td>BLDC in oil</td>
<td>/</td>
<td>/</td>
</tr>
<tr>
<td>Fusion Perry</td>
<td>TAD0</td>
<td>Hydraulic</td>
<td>4</td>
<td>98</td>
<td>65</td>
<td>125 (210)</td>
<td>150</td>
<td>50</td>
<td>11000</td>
<td>2</td>
<td>No electrical Al SS Cylinders, rot.</td>
<td>Cylinders, rot.</td>
<td>240VDC</td>
<td>/</td>
<td></td>
<td></td>
</tr>
<tr>
<td>Forum Perry</td>
<td>TA60</td>
<td>Hydraulic</td>
<td>4</td>
<td>82</td>
<td>60</td>
<td>380 (300)</td>
<td>250</td>
<td>50</td>
<td>11000</td>
<td>1.44</td>
<td>No electrical Al SS Cylinders, rot.</td>
<td>Cylinders, rot.</td>
<td>Hybrid fb.</td>
<td>/</td>
<td></td>
<td></td>
</tr>
<tr>
<td>Forum Perry</td>
<td>TA60J</td>
<td>Hydraulic</td>
<td>4</td>
<td>76</td>
<td>51</td>
<td>380 (300)</td>
<td>250</td>
<td>50</td>
<td>11000</td>
<td>1.38</td>
<td>No electrical Al SS Cylinders, rot.</td>
<td>Cylinders, rot.</td>
<td>Hybrid fb.</td>
<td>/</td>
<td></td>
<td></td>
</tr>
<tr>
<td>Forum Perry</td>
<td>TAD8</td>
<td>Hydraulic</td>
<td>4</td>
<td>50</td>
<td>40</td>
<td>147 (102)</td>
<td>108</td>
<td>26</td>
<td>11000</td>
<td>1.06</td>
<td>No electrical Al SS Cylinders, rot.</td>
<td>Cylinders, rot.</td>
<td>Hybrid fb.</td>
<td>/</td>
<td></td>
<td></td>
</tr>
<tr>
<td>Fusion Perry</td>
<td>TAD10</td>
<td>Hydraulic</td>
<td>4</td>
<td>28</td>
<td>14</td>
<td>10 (10)</td>
<td>/</td>
<td>/</td>
<td>11000</td>
<td>1.06</td>
<td>No electrical Al SS Cylinders, rot.</td>
<td>Cylinders, rot.</td>
<td>Hybrid fb.</td>
<td>/</td>
<td></td>
<td></td>
</tr>
<tr>
<td>Fusion Perry</td>
<td>TAD16</td>
<td>Hydraulic</td>
<td>4</td>
<td>25</td>
<td>12</td>
<td>100 (102)</td>
<td>250</td>
<td>50</td>
<td>11000</td>
<td>1.06</td>
<td>No electrical Al SS Cylinders, rot.</td>
<td>Cylinders, rot.</td>
<td>Hybrid fb.</td>
<td>/</td>
<td></td>
<td></td>
</tr>
<tr>
<td>Fusion Perry</td>
<td>TAD20</td>
<td>Hydraulic</td>
<td>4</td>
<td>25</td>
<td>15</td>
<td>100 (102)</td>
<td>250</td>
<td>50</td>
<td>11000</td>
<td>1.06</td>
<td>No electrical Al SS Cylinders, rot.</td>
<td>Cylinders, rot.</td>
<td>Hybrid fb.</td>
<td>/</td>
<td></td>
<td></td>
</tr>
<tr>
<td>Hydro-Lek</td>
<td>40400</td>
<td>Hydraulic</td>
<td>4</td>
<td>45</td>
<td>30</td>
<td>150 (210)</td>
<td>75</td>
<td>/</td>
<td>11000</td>
<td>1.42</td>
<td>SS 316-Al cylinders &amp; 210bar</td>
<td>Cylinders &amp; 210bar</td>
<td>Hybrid fb.</td>
<td>/</td>
<td></td>
<td></td>
</tr>
<tr>
<td>Hydro-Lek</td>
<td>40500 (R)</td>
<td>Hydraulic</td>
<td>4</td>
<td>59</td>
<td>40</td>
<td>150 (210)</td>
<td>75</td>
<td>/</td>
<td>11000</td>
<td>1.5</td>
<td>SS 316-Al cylinders &amp; 210bar</td>
<td>Cylinders &amp; 210bar</td>
<td>Hybrid fb.</td>
<td>/</td>
<td></td>
<td></td>
</tr>
</tbody>
</table>

Table 3.1 Specifications of existing commercial underwater manipulators
<table>
<thead>
<tr>
<th>Model</th>
<th>Type</th>
<th>Hydraulic</th>
<th>Displacement (L)</th>
<th>Flow (lpm)</th>
<th>Voltage (V)</th>
<th>Current (A)</th>
<th>Frequency (Hz)</th>
<th>Phase</th>
<th>Pressure (bar)</th>
<th>Material</th>
<th>Encoder Type</th>
<th>Position &amp; Force Feedback</th>
<th>Cost (~$)</th>
</tr>
</thead>
<tbody>
<tr>
<td>Hydro-Lek 43000</td>
<td>Hydraulic</td>
<td>4</td>
<td>6</td>
<td>4</td>
<td>10 (20)</td>
<td>8</td>
<td>/</td>
<td>11000</td>
<td>0.53</td>
<td>SS 316, PE</td>
<td>Cylinders &amp; gerotor</td>
<td>No</td>
<td>Rate ~4k</td>
</tr>
<tr>
<td>Hydro-Lek CRA6</td>
<td>Hydraulic</td>
<td>5</td>
<td>28</td>
<td>14.5</td>
<td>32 (32)</td>
<td>38</td>
<td>/</td>
<td>11000</td>
<td>1.5</td>
<td>SS 316, Al HE30, PE</td>
<td>Cylinders &amp; gerotor</td>
<td>No</td>
<td>Rate ~12k</td>
</tr>
<tr>
<td>Hydro-Lek EH5</td>
<td>Hydraulic</td>
<td>4</td>
<td>12</td>
<td>9.5</td>
<td>25 (25)</td>
<td>14</td>
<td>/</td>
<td>11000</td>
<td>0.8</td>
<td>Al E30, SS 316</td>
<td>Cylinders &amp; gerotor</td>
<td>No</td>
<td>Rate ~7k</td>
</tr>
<tr>
<td>Hydro-Lek HD5</td>
<td>Hydraulic</td>
<td>4</td>
<td>21.5</td>
<td>16.5</td>
<td>40 (40)</td>
<td>38</td>
<td>/</td>
<td>11000</td>
<td>0.819</td>
<td>SS 316, Al HE30, PE</td>
<td>Cylinders &amp; gerotor</td>
<td>No</td>
<td>Rate ~9k</td>
</tr>
<tr>
<td>Hydro-Lek HD6W HD6R</td>
<td>Hydraulic</td>
<td>5</td>
<td>29</td>
<td>21</td>
<td>40 (40)</td>
<td>38</td>
<td>/</td>
<td>11000</td>
<td>1.12</td>
<td>SS 316, Al HE30, PE</td>
<td>Cylinders &amp; gerotor</td>
<td>No</td>
<td>Rate ~12k</td>
</tr>
<tr>
<td>Hydro-Lek MB4</td>
<td>Hydraulic</td>
<td>3</td>
<td>13.3</td>
<td>11</td>
<td>40 (40)</td>
<td>40</td>
<td>/</td>
<td>11000</td>
<td>0.63</td>
<td>SS 316, Al HE30, PE</td>
<td>Cylinders &amp; gerotor</td>
<td>No</td>
<td>Rate ~5k</td>
</tr>
<tr>
<td>Hydro-Lek RHD5(W)</td>
<td>Hydraulic</td>
<td>4</td>
<td>30</td>
<td>20</td>
<td>80 (80)</td>
<td>38</td>
<td>/</td>
<td>11000</td>
<td>0.95</td>
<td>SS 316, Al HE30, PE</td>
<td>Cylinders &amp; gerotor</td>
<td>No</td>
<td>Rate ~11k</td>
</tr>
<tr>
<td>ISE Ltd. Magnum 7</td>
<td>Hydraulic</td>
<td>6</td>
<td>63.5</td>
<td>30</td>
<td>454 (295)</td>
<td>108</td>
<td>205</td>
<td>11000</td>
<td>1.5</td>
<td>Al, SS</td>
<td>Cylinders Potentiometers</td>
<td>Pos./Rate ~59k</td>
<td>~59k</td>
</tr>
<tr>
<td>ISE Ltd. Magnum 5</td>
<td>Hydraulic</td>
<td>4</td>
<td>50</td>
<td>27</td>
<td>454 (295)</td>
<td>108</td>
<td>160</td>
<td>5000</td>
<td>1.16</td>
<td>Al, SS</td>
<td>Cylinders Potentiometers</td>
<td>Pos./Rate ~52k</td>
<td>~52k</td>
</tr>
<tr>
<td>ISE Ltd. Magnum 6</td>
<td>Hydraulic</td>
<td>5</td>
<td>57</td>
<td>30.6</td>
<td>454 (317)</td>
<td>108</td>
<td>160</td>
<td>5000</td>
<td>0.96</td>
<td>Al, SS</td>
<td>Cylinders</td>
<td>No</td>
<td>~56k</td>
</tr>
<tr>
<td>ISE Ltd. Magnum 5</td>
<td>Hydraulic</td>
<td>4</td>
<td>34</td>
<td>24</td>
<td>68 (23)</td>
<td>14</td>
<td>35</td>
<td>5000</td>
<td>0.71</td>
<td>Al, SS</td>
<td>Cylinders</td>
<td>No</td>
<td>~46k</td>
</tr>
<tr>
<td>KKR Systems Inc. HYDRA UW3</td>
<td>Hydraulic</td>
<td>6</td>
<td>130</td>
<td>/</td>
<td>300 (121)</td>
<td>350</td>
<td>300</td>
<td>500</td>
<td>2.035</td>
<td>Al, SS, Ti</td>
<td>Cylinders &amp; rotary vane Encoders</td>
<td>Position ~175k</td>
<td>~175k</td>
</tr>
<tr>
<td>Kraft Predator</td>
<td>Hydraulic</td>
<td>6</td>
<td>80</td>
<td>51</td>
<td>227 (91)</td>
<td>135</td>
<td>135</td>
<td>6500</td>
<td>1.64</td>
<td>Al, SS</td>
<td>Cylinders, rot. vane &amp; gerotor</td>
<td>Yes Pos. &amp; force fb. ~200k</td>
<td>~200k</td>
</tr>
<tr>
<td>Kraft Raptor</td>
<td>Hydraulic</td>
<td>6</td>
<td>75</td>
<td>44</td>
<td>227 (91)</td>
<td>135</td>
<td>135</td>
<td>6500</td>
<td>1.52</td>
<td>Al, SS</td>
<td>Cylinders, rot. vane &amp; gerotor</td>
<td>Yes Pos. &amp; force fb. ~200k</td>
<td>~200k</td>
</tr>
<tr>
<td>Kraft Grips</td>
<td>Hydraulic</td>
<td>6</td>
<td>59</td>
<td>41</td>
<td>82 (45)</td>
<td>20</td>
<td>90</td>
<td>3000</td>
<td>1.556</td>
<td>Al, SS</td>
<td>Cylinders, rot. vane &amp; gerotor</td>
<td>Yes Pos. &amp; force fb. ~200k</td>
<td>~200k</td>
</tr>
<tr>
<td>Ocean Innovation System BE5-500</td>
<td>Electric</td>
<td>4</td>
<td>15</td>
<td>8 (16)</td>
<td>1.6</td>
<td>100</td>
<td>500</td>
<td>0.7</td>
<td>24 VDC</td>
<td>Al 5083, PE</td>
<td>BLDC Hall</td>
<td>Rate ~30k</td>
<td>~30k</td>
</tr>
<tr>
<td>Manufacturer</td>
<td>Model</td>
<td>Series</td>
<td>HP</td>
<td>Torque (Nm)</td>
<td>Max. Speed (rpm)</td>
<td>Pressure</td>
<td>Flow Rate (L/min)</td>
<td>Material(s)</td>
<td>Cylinders Type</td>
<td>Feedback Type</td>
<td>Notes</td>
<td></td>
<td></td>
</tr>
<tr>
<td>------------------------------</td>
<td>----------------</td>
<td>--------</td>
<td>----</td>
<td>-------------</td>
<td>------------------</td>
<td>----------</td>
<td>-------------------</td>
<td>-------------</td>
<td>----------------</td>
<td>---------------</td>
<td>----------------</td>
<td></td>
<td></td>
</tr>
<tr>
<td>Oceaneering</td>
<td>Atlas Hybrid</td>
<td>6</td>
<td>73</td>
<td>50</td>
<td>454 (250)</td>
<td>205</td>
<td>454</td>
<td>Al, SS</td>
<td>Cylinders &amp; rot</td>
<td>Available</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>Schilling</td>
<td>Titan 2</td>
<td>6</td>
<td>80</td>
<td>61</td>
<td>/ (109)</td>
<td>68</td>
<td>136</td>
<td>Ti</td>
<td>Cylinders, rot, vane &amp; gerotor</td>
<td>Available</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>Schilling</td>
<td>Titan 3</td>
<td>6</td>
<td>/</td>
<td>/</td>
<td>/ (1)</td>
<td>/</td>
<td>/</td>
<td>Ti</td>
<td>Cylinders, rot, vane &amp; gerotor</td>
<td>Available</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>Schilling</td>
<td>Titan 4</td>
<td>6</td>
<td>100</td>
<td>78</td>
<td>454 (122)</td>
<td>170</td>
<td>417</td>
<td>Ti</td>
<td>Cylinders, rot, vane &amp; gerotor</td>
<td>Available</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>Schilling</td>
<td>Conan 7P</td>
<td>6</td>
<td>107</td>
<td>73</td>
<td>273 (159)</td>
<td>205</td>
<td>454</td>
<td>Al, SS</td>
<td>Cylinders, rot, vane &amp; gerotor</td>
<td>Available</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>Schilling</td>
<td>Orion 7P/7R</td>
<td>6</td>
<td>54</td>
<td>38</td>
<td>250 (68)</td>
<td>205</td>
<td>454</td>
<td>Al, SS</td>
<td>Cylinders, rot, vane &amp; gerotor</td>
<td>Available</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>Schilling</td>
<td>Atlas 7R</td>
<td>6</td>
<td>73</td>
<td>50</td>
<td>500 (250)</td>
<td>205</td>
<td>454</td>
<td>Al, SS</td>
<td>Cylinders, rot, vane &amp; gerotor</td>
<td>Available</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>Schilling</td>
<td>RigMaster</td>
<td>4</td>
<td>64</td>
<td>48</td>
<td>270 (181)</td>
<td>205</td>
<td>454</td>
<td>Ti, Al, SS 316</td>
<td>Cylinders, rot, vane &amp; gerotor</td>
<td>Available</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>Schilling</td>
<td>Orion 4R</td>
<td>3</td>
<td>30</td>
<td>21</td>
<td>136 (7)</td>
<td>205</td>
<td>454</td>
<td>Al, SS</td>
<td>Cylinders, rot, vane &amp; gerotor</td>
<td>Available</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>Seamor</td>
<td>7F-H-ARM</td>
<td>6</td>
<td>32</td>
<td>/</td>
<td>/ (5)</td>
<td>/</td>
<td>300</td>
<td>300V 39bar 4.5lpm</td>
<td>Cylinders, rot, vane &amp; gerotor</td>
<td>Available</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>TitanRob</td>
<td>M700</td>
<td>6</td>
<td>30</td>
<td>20</td>
<td>50 (40)</td>
<td>45</td>
<td>80</td>
<td>Ti, SS 316</td>
<td>Cylinders, rot, vane &amp; gerotor</td>
<td>Available</td>
<td>~45k</td>
<td></td>
<td></td>
</tr>
<tr>
<td>TitanRob</td>
<td>G500</td>
<td>4</td>
<td>20</td>
<td>15</td>
<td>100 (80)</td>
<td>80</td>
<td>250</td>
<td>Ti, SS 316</td>
<td>Cylinders &amp; rotary</td>
<td>Available</td>
<td>~40k</td>
<td></td>
<td></td>
</tr>
<tr>
<td>TitanRob</td>
<td>M501</td>
<td>4</td>
<td>14</td>
<td>11</td>
<td>50 (40)</td>
<td>45</td>
<td>80</td>
<td>Ti, SS 316</td>
<td>Cylinders &amp; rotary</td>
<td>Available</td>
<td>~35k</td>
<td></td>
<td></td>
</tr>
<tr>
<td>Western Space &amp; Marine</td>
<td>The ARM</td>
<td>6</td>
<td>145</td>
<td>97</td>
<td>45.4 (29.5)</td>
<td>/</td>
<td>11000</td>
<td>24VDC 204bar 7.6lpm</td>
<td>Cylinders, rot, vane &amp; gerotor</td>
<td>Available</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>Western Space &amp; Marine</td>
<td>MK37</td>
<td>6</td>
<td>43</td>
<td>16</td>
<td>23 (7)</td>
<td>/</td>
<td>114</td>
<td>24(±15)VDC 204bar 4.5lpm</td>
<td>Cylinders, rot, vane &amp; gerotor</td>
<td>Available</td>
<td></td>
<td></td>
<td></td>
</tr>
</tbody>
</table>

**Note:** / - Information not available; Pos. - Position; Prop. - Proportional; Fb. - Feedback.
3.3 Mechanical design

Table 3.2 Relative manipulator-to-vehicle weight for commercial heavy (H), medium (M) and light (L) work class ROVs

<table>
<thead>
<tr>
<th>ROV Class</th>
<th>ROV weight in air (kg)</th>
<th>Manipulator</th>
<th>Manipulator weight in air (kg)</th>
<th>Manipulator-ROV relative weight (%)</th>
</tr>
</thead>
<tbody>
<tr>
<td>Oceaneering Nexus H</td>
<td>4700</td>
<td>Schilling Atlas</td>
<td>73</td>
<td>1.5</td>
</tr>
<tr>
<td>Perry XLX-Evo H</td>
<td>5500</td>
<td>Schilling Titan 4</td>
<td>100</td>
<td>1.8</td>
</tr>
<tr>
<td>Oceaneering Millennium H</td>
<td>4000</td>
<td>Schilling Titan 4</td>
<td>100</td>
<td>2.5</td>
</tr>
<tr>
<td>Oceaneering Magnum H</td>
<td>3000</td>
<td>Schilling Titan 4</td>
<td>100</td>
<td>3.3</td>
</tr>
<tr>
<td>Perry XLX-C H</td>
<td>3000</td>
<td>Schilling Titan 4</td>
<td>100</td>
<td>3.3</td>
</tr>
<tr>
<td>Saab Seaeye Leopard M</td>
<td>1200</td>
<td>Schilling Orion</td>
<td>54</td>
<td>4.5</td>
</tr>
<tr>
<td>Sub-Atlantic Comanche M</td>
<td>1130</td>
<td>Schilling Orion</td>
<td>54</td>
<td>4.7</td>
</tr>
<tr>
<td>Saab Seaeye Panther-XT Plus L</td>
<td>800</td>
<td>Schilling Orion</td>
<td>54</td>
<td>6.7</td>
</tr>
<tr>
<td>Saab Seaeye Cougar-XT L</td>
<td>580</td>
<td>Hydro-Lek HD5</td>
<td>21.5</td>
<td>3.7</td>
</tr>
<tr>
<td>Saab Seaeye Panther-XT L</td>
<td>500</td>
<td>Hydro-Lek HD6R</td>
<td>21.5</td>
<td>4.3</td>
</tr>
<tr>
<td>Oceaneering Spectrum L</td>
<td>415</td>
<td>Hydro-Lek HD5</td>
<td>21.5</td>
<td>5.2</td>
</tr>
<tr>
<td>Sub-Atlantic Mohawk II L</td>
<td>395</td>
<td>Hydro-Lek HD5</td>
<td>21.5</td>
<td>5.4</td>
</tr>
</tbody>
</table>

Note. H - Heavy Work Class; M - Medium Work Class; L - Light Work Class

Depending on the nature of the task underwater manipulators are designed for, they come with a different number of DOFs. Most commercial and experimental underwater manipulators are composed of three to six DOFs without taking gripper’s mobility into account. The reason for this is that three DOFs are sufficient for achieving an arbitrary position and six DOFs for achieving both arbitrary position and orientation of the end-effector or tool in the workspace (Spong et al., 2006). The term “n-function” is often used in the technical literature. It describes the number of single joint actuators contained in a manipulator including the gripper’s mobility, e.g. a seven function manipulator has six actuators responsible for manipulator motion that provide six true DOFs plus one actuator for jaw movement. Underwater manipulators with seven or more DOFs (without gripper mobility) are not very common, but they do exist. Automating such subsea manipulators can be useful since it is possible to exploit their redundancy for secondary objectives such as obstacle avoidance and optimal load distribution between the joints (Siciliano, 1990). Marani et al. (2009) and Ribas et al. (2015) reported research with seven DOF underwater manipulators, and Greig and Broome (1994) with an eight DOF manipulator. Some authors propose a multi-stage manipulator—a micro-macro manipulator concept that includes a large manipulator responsible for coarse positioning that carries a smaller manipulator responsible for the fine end-effector positioning (Asokan et al., 2003; Ishimi et al., 1991).

Any robotics application to be carried out with underwater manipulators requires applying kinematic modelling, planning, and control algorithms. Underwater manipulators have a serial-chain mechanical structure similar to industrial robotic arms. There is much robotics literature on the algorithms common for robotic arms that can be applied to underwater manipulators, some of which can be found in (Corke,
Literature review on underwater manipulators

2017; Siciliano et al., 2009; Spong et al., 2006). Additional literature related to vehicle-manipulator systems can be found in From et al. (2014), and more specific literature related to underwater vehicle-manipulator systems in Antonelli (2014).

3.4 Actuation

In the early 90’s a few authors proposed and experimented on seawater driven actuators for subsea manipulators (Ishimi et al., 1991; Yoshinada et al., 1991). Some of the benefits of water hydraulics are low viscosity, high power density, non-flammable properties, and zero environmental impact (Krutz and Chua, 2004). However, this actuation approach has been abandoned over the years due to its various disadvantages such as corrosive and abrasive properties, unsuitable working temperature range, lubrication and sealing issues. Today all the existing commercially available subsea manipulators and most of the experimental/prototype subsea manipulators developed for research purposes run on either hydraulic oil or electric power, both of which have their advantages and disadvantages. Denket (2006) proposes a hybrid power structure using both hydraulic and electric actuators for a single manipulator to have the benefit of both. However, this actuation method has not yet found use in the commercial sector. Biodegradable fluids have begun to be introduced to minimise the impact on the environment of fluid leaks.

3.4.1 Hydraulic manipulators

In general, hydraulic actuators are capable of producing an output force/torque much larger than the force applied on the input without the use of mechanical transmission components such as gears and levers (direct drive), which is a very common necessity for the implementation with electric actuators. Thus, the payload capacity of hydraulic systems is relatively high due to high power to weight ratio, which goes up to the order of three for the existing commercial hydraulic underwater manipulators, whereas that ratio is one or less for the electrical ones. For this reason and because they require fewer parts, hydraulic systems are more compact for the same carrying capacity. Additionally, hydraulic systems are inherently pressurised, i.e. the internal pressure is higher than the ambient pressure, so they are not as susceptible to the sea water ingress as are their electric counterparts. Another advantage of hydraulic systems is that they possess inbuilt protection against overload.

Due to the benefits outlined above, it is hydraulic oil that drives the vast majority of commercial manipulators operating underwater. Typically, actuators with limited
motion such as piston cylinders and rotary vane actuators drive the joints of a manipulator. However, hydraulic motors with continuous motion, are sometimes used for wrist joint actuation. The medium used for power transmission, pressurised hydraulic fluid is conveyed from a reservoir to actuators through flexible hoses or rigid pipes by a Hydraulic Power Unit (HPU), which consists of an electrically driven hydraulic pump, a pressure regulator, and accompanying equipment. Regulating the hydraulic fluid flow controls the motion of hydraulic systems, and the devices that are in charge of the flow control are various electro-hydraulic valves such as directional control valves (“switching” / “bang-bang”), proportional valves, or servo valves. These valves are located either in an external valve pack or are integrated into the body of the manipulator along with hoses/pipes and electronic components and circuits. In the latter case, inner manipulator chambers that contain valves and accompanying electronic circuits are oil filled and pressurised by a pressure compensator, which maintains the internal pressure slightly above the external pressure. Such manipulators have three external hydraulic hose connectors (pressure, return, and compensation pressure) and a single electrical connector. On the other side, if the valves and electronics are located in the valve pack independent from the manipulator, the external valve pack is oil compensated instead of the manipulator. Such systems have more hydraulic lines, i.e. a pair of pressure and return lines for each actuator (joint).

Leading commercial manufacturers of underwater hydraulic manipulators are Kraft Telerobotics, Schilling Robotics, Cybernetix, Hydro-lek, among others. Some of the mentioned manipulators are presented in Figs. 3.2-3.5. Apart from industrial manufacturers, some research groups reported work on the design and development of experimental/prototype hydraulic underwater manipulators. Yao et al. (2009) and Zhang et al. (2014) present the development of manipulators with specifications and robot configurations similar to the commercially available seven-function manipulators on work class ROVs. Conversely, Zuyao et al. (2011) focus on atypical configuration with fewer DOFs. Sonoda et al. (2017) demonstrate a 5 DOF lightweight hydraulic underwater manipulator that can be fitted on an AUV.

Despite their numerous advantages, hydraulically driven manipulators do have drawbacks. Unlike their electrical equivalents, they can feature poor positioning accuracy and are not suited for implementation of fine control of the interaction force with the environment during contact tasks (Terribile et al., 1993). These limitations are not substantial in the conventional master-slave teleoperation; however, in the case of implementation of automatic robotic functions, their significance may be of great concern. Another disadvantage of hydraulic systems is leakage of a minor amount of hydraulic fluid which is almost impossible to prevent, and the necessity to
protect fluids from contamination, both of which bring about demands for the highest quality standards and materials for manufacturing of components, making hydraulic systems more expensive. Moreover, hydraulic manipulators require complementary equipment such as a hydraulic pump, a reservoir, filters, regulators, valves, etc. Hydraulically actuated thrusters propel the majority of work-class ROVs and, therefore, hydraulic equipment and power are already available on the base vehicles. However, this is not always the case, e.g. if the available power can be insufficient to run both thrusters and the manipulator at the same time, or if ROV has electrically actuated thrusters it is less likely that the vehicle may have an HPU. On the other hand, electrical power is the only additional requirement for an electrical manipulator.

### 3.4.2 Electric manipulators

Electric underwater manipulators are less frequent in commercial use but are often custom made as prototypes for research purposes. Commonly used actuators are
3.4 Actuation

Brushless DC (BLDC) electric motors with harmonic drive gears featuring low backlash and large reduction ratio. For the prevention of water ingress, the actuators are often oil filled, which also provides lubrication and cooling. Frequently, to avoid having external cables or possible entanglement, power and signal cables are fed through the same hoses used for pressure compensation (Terribile et al., 1994). Ishitsuka and Ishii (2007a) and Ishitsuka and Ishii (2007b) report experimental prototypes using magnetic coupling mechanisms for transferring torque into joints as an alternative approach for watertightness. The main advantage of electrically driven manipulators is the capability for precise motion and force/torque control as they are inherently equivalent to the industrial robot arms.

The leading commercial manufacturer of underwater electric manipulators is Eca Robotics, whose manipulator 7E is presented in Fig. 3.6.

Yoerger et al. (1991) are among the first to address the design of an electrically driven underwater manipulator—a three DOF robotic arm developed for the Woods Hole Oceanographic Institution’s ROV JASON. Terribile et al. (1994) reported a six DOF electrical manipulator developed by Tecnomare and Ansaldo in Italy, which has 2.1 m reach with a maximum payload of 30 kg. Another example of the early work can be found in Smith et al. (1994), where a five DOF manipulator called “Poseidon” is developed, consisting of 1 m reach, operational depth of up to 100 m and lifting capacity of 5 kg. Collaborating with the Autonomous Systems Laboratory of the University of Hawaii for the SAUVIM AUV project, Ansaldo developed a seven DOF manipulator called “MARIS 7080” (Fig. 3.7) (Marini et al., 2009; Yuh et al., 1998). Rated for 6000 m depth, it has 1.4 m reach and 6 kg payload at full extension. Two seven DOF manipulators also developed by Ansaldo were used within the AMADEUS project (Casalino et al., 2001; Lane et al., 1997) for cooperative sampling. In 2007, the Space System Laboratory at the University of Maryland teamed with the Woods Hole Oceanographic Institute and developed a six DOF Subsea Arctic Manipulator for Underwater Retrieval and Autonomous Interventions (“SAMURAI“) (Lewandowski et al., 2008). Some more recent work on the development of experimental electrical underwater manipulators is reported by Pandian and Sakagami (2010) who present a three DOF manipulator developed for validation of control algorithms. Cobos-Guzman et al. (2013) developed a three DOF manipulator called LAFMIA-UMI-I which was to be mounted on a small submarine. Sheikhbahaee et al. (2014) developed a four DOF manipulator called “Kavosh-4” for use in a towing tank. Xu et al. (2010) proposed a design for an electrical three DOF manipulator and reported its development and testing (Shen et al., 2011; Xiao et al., 2011). Within the RAUVI project, Fernandez et al. (2013) reported modifying a commercially available electric manipulator, ARM 5E, a product of CSIP (now
Literature review on underwater manipulators

ECA). As a result of collaborative work of IRS-Lab from The Jaume I University in Spain and CSIP company, this manipulator was rebuilt to reduce the dimensions and weight—so it can fit on the existing underwater vehicle GIRONA 500—and to reduce the dynamic coupling with it. Another recent custom made electrical underwater manipulator is UMA (Fig. 3.8), developed by Graal Tech SRL in Italy for the TRIDENT project, and which subsequently is currently commercially available (Ribas et al., 2015). Its unique characteristic is that it is made by modular joints with a compatible electromechanical interface, which allows building a customised manipulator according to the desired user kinematics without doing a dedicated design. Tang et al. (2017) proposed an electrically driven lightweight underwater manipulator with a unique mechanical design comprised of a tendon based actuation system where all servos are placed in the manipulator base.

Depending on the nature of the task, underwater electric manipulators can find use in subsea intervention operations. However, for most industrial intervention subsea tasks, they often do not meet the speed, reliability and strength or force/torque requirements (Hildebrandt, Kerdels, Albiez and Kirchner, 2009; Schjølberg et al., 2016). Electric manipulators are inherently not designed or available with sufficient power for work-class ROV intervention work as specified by ocean engineering contractor requirements. For example, the manipulator-operated torque tool, which uses the wrist rotate function of the manipulator to generate the required torque is used to operate ISO 13628 Class 1 (67 N m) and 2 (271 N m) (13628-8:2002, 2002) interfaces without the need of a hydraulically operated torque tool (Christ and Wernli, 2014). Table 3.4 presents actuation forces of subsea hydraulic and
3.4 Actuation

electric manipulators, and Table 3.3 summarises the weight of typical ROV operated tools used in the offshore oil and gas industry. Analysis of these two tables leads to the conclusion that the majority of electrical manipulators would struggle even to lift, let alone intervene with most of the tools. One of the few research groups that have been working with a commercial underwater manipulator (Schilling Orion 7P) is DFKI-Lab Bremen who experimented on automated plugging of a deep-sea connector in a wet laboratory testbed within the CManipulator project (Hildebrandt, Kerdels, Albiez and Kirchner, 2009).

As outlined above, the majority of academic research experiments in the field of autonomous underwater manipulation have been carried out on electro-mechanical robotic arms which are either prototypes or recently commercialised. Additionally, all those advanced subsea autonomous manipulation solutions found in the literature (Cieslak et al., 2015; Evans et al., 2003; Marani and Yuh, 2014) are related to intervention AUVs, which are not industry standard but rather a concept in development and besides, the base vehicle (AUVs) are considerably power constrained. Electric robotic manipulators cannot perform all subsea intervention operations which is why these prototype manipulators are not ready for adoption in the offshore industry. There are strong reasons why all work-class ROVs use hydraulic manipulators (depth rating, very high carrying capacity and torque, straightforward field maintenance, etc.).

<table>
<thead>
<tr>
<th>Manufacturer</th>
<th>Model</th>
<th>Actuation</th>
<th>Lift capacity max nom. (kg)</th>
<th>Lift capacity full ext. (kg)</th>
<th>Wrist torque (Nm)</th>
<th>Grip force (kgf)</th>
</tr>
</thead>
<tbody>
<tr>
<td>ISE Ltd.</td>
<td>Magnum 7</td>
<td>Hydraulic</td>
<td>454</td>
<td>295</td>
<td>108</td>
<td>205</td>
</tr>
<tr>
<td>Schilling</td>
<td>Titan 4</td>
<td>Hydraulic</td>
<td>454</td>
<td>122</td>
<td>170</td>
<td>170</td>
</tr>
<tr>
<td>KNR Systems Inc.</td>
<td>HYDRA UW3</td>
<td>Hydraulic</td>
<td>300</td>
<td>121</td>
<td>350</td>
<td>300</td>
</tr>
<tr>
<td>Profound Technology</td>
<td>M1P</td>
<td>Hydraulic</td>
<td>275</td>
<td>250</td>
<td>175</td>
<td>652</td>
</tr>
<tr>
<td>Schilling</td>
<td>Orion 7P /7R</td>
<td>Hydraulic</td>
<td>250</td>
<td>68</td>
<td>205</td>
<td>454</td>
</tr>
<tr>
<td>Kraft</td>
<td>Predator</td>
<td>Hydraulic</td>
<td>227</td>
<td>91</td>
<td>135</td>
<td>135</td>
</tr>
<tr>
<td>Hydro-Lek</td>
<td>40400</td>
<td>Hydraulic</td>
<td>150</td>
<td>210</td>
<td>75</td>
<td>/</td>
</tr>
<tr>
<td>Forum Perry</td>
<td>TA40</td>
<td>Hydraulic</td>
<td>125</td>
<td>250</td>
<td>150</td>
<td>509</td>
</tr>
<tr>
<td>Cybernetix</td>
<td>Maestro</td>
<td>Hydraulic</td>
<td>100</td>
<td>96</td>
<td>190</td>
<td>150</td>
</tr>
<tr>
<td>Eca Hytec</td>
<td>Arm 7E</td>
<td>Electric</td>
<td>40</td>
<td>40</td>
<td>25</td>
<td>80</td>
</tr>
<tr>
<td>Eca Hytec</td>
<td>Arm 7E Mini</td>
<td>Electric</td>
<td>25</td>
<td>25</td>
<td>25</td>
<td>50</td>
</tr>
<tr>
<td>Eca Hytec</td>
<td>Arm 5E</td>
<td>Electric</td>
<td>25</td>
<td>25</td>
<td>25</td>
<td>50</td>
</tr>
<tr>
<td>Eca Hytec</td>
<td>Arm 5E Micro</td>
<td>Electric</td>
<td>10</td>
<td>10</td>
<td>10</td>
<td>50</td>
</tr>
<tr>
<td>Graal Tech</td>
<td>UMA</td>
<td>Electric</td>
<td>10</td>
<td>/</td>
<td>/</td>
<td>/</td>
</tr>
<tr>
<td>Ocean Innovation System</td>
<td>BE5-500</td>
<td>Electric</td>
<td>/</td>
<td>16</td>
<td>1.6</td>
<td>100</td>
</tr>
<tr>
<td>Ansaldo</td>
<td>MARIS 70800</td>
<td>Electric</td>
<td>8</td>
<td>/</td>
<td>/</td>
<td>20.4</td>
</tr>
</tbody>
</table>
Table 3.4 Summary of typical ROV operated tools

<table>
<thead>
<tr>
<th>Tool type</th>
<th>Manufacturer</th>
<th>W.i.a. (kg)</th>
<th>W.i.w. (kg)</th>
<th>Tool type</th>
<th>Manufacturer</th>
<th>W.i.a. (kg)</th>
<th>W.i.w. (kg)</th>
</tr>
</thead>
<tbody>
<tr>
<td>Torque Tool Class 1-2</td>
<td>FET</td>
<td>26</td>
<td>18</td>
<td>Hub Cleaning Tool</td>
<td>J2 Subsea</td>
<td>21.8</td>
<td>/</td>
</tr>
<tr>
<td>Torque Tool Class 1-2</td>
<td>J2 Subsea</td>
<td>30.8</td>
<td>25.9</td>
<td>Hub Cleaning Tool</td>
<td>Fugro</td>
<td>46.9</td>
<td>27.5</td>
</tr>
<tr>
<td>Torque Tool Class 1-4</td>
<td>Zetechtics</td>
<td>48.9</td>
<td>37.1</td>
<td>Hub Cleaning Tool</td>
<td>IKM</td>
<td>58</td>
<td>35</td>
</tr>
<tr>
<td>Torque Tool Class 1-4</td>
<td>Oceanering</td>
<td>39</td>
<td>30</td>
<td>Hub Cleaning Tool</td>
<td>Oceanering</td>
<td>60</td>
<td>40</td>
</tr>
<tr>
<td>Torque Tool Class 1-4</td>
<td>Oceanering</td>
<td>43</td>
<td>32</td>
<td>Cleaning Brush Tool</td>
<td>J2 Subsea</td>
<td>7</td>
<td>7</td>
</tr>
<tr>
<td>Torque Tool Class 1-4</td>
<td>Fugro</td>
<td>47</td>
<td>38</td>
<td>Cleaning Brush Tool</td>
<td>FET</td>
<td>10</td>
<td>7</td>
</tr>
<tr>
<td>Torque Tool Class 1-4</td>
<td>FET</td>
<td>45</td>
<td>35</td>
<td>Cleaning Brush Tool</td>
<td>Rovquip</td>
<td>11.3</td>
<td>7.9</td>
</tr>
<tr>
<td>Torque Tool Class 1-4</td>
<td>Oceanering</td>
<td>64</td>
<td>43</td>
<td>Cleaning Brush Tool</td>
<td>Fugro</td>
<td>18</td>
<td>18</td>
</tr>
<tr>
<td>Torque Tool Class 5</td>
<td>FET</td>
<td>54</td>
<td>41</td>
<td>Cleaning Brush Tool</td>
<td>Oceanering</td>
<td>25</td>
<td>21</td>
</tr>
<tr>
<td>Torque Tool Class 5</td>
<td>Zetechtics</td>
<td>85.5</td>
<td>68.4</td>
<td>Cleaning Brush Tool</td>
<td>IKM</td>
<td>43</td>
<td>31</td>
</tr>
<tr>
<td>Torque Tool Class 5</td>
<td>Fugro</td>
<td>90</td>
<td>69</td>
<td>Gasket Removal Tool</td>
<td>J2 Subsea</td>
<td>15</td>
<td>/</td>
</tr>
<tr>
<td>Torque Tool Class 6</td>
<td>Fugro</td>
<td>107</td>
<td>111</td>
<td>Gasket Removal Tool</td>
<td>Fugro</td>
<td>15.3</td>
<td>10.4</td>
</tr>
<tr>
<td>Torque Tool Class 6</td>
<td>Fugro</td>
<td>123.1</td>
<td>82.6</td>
<td>Gasket Removal Tool</td>
<td>Fugro</td>
<td>20.8</td>
<td>15.6</td>
</tr>
<tr>
<td>Gear Adap. Class 4-5</td>
<td>Oceanering</td>
<td>36</td>
<td>28</td>
<td>Gasket Removal Tool</td>
<td>Fugro</td>
<td>27.8</td>
<td>19.2</td>
</tr>
<tr>
<td>Gear Adap. Class 4-6</td>
<td>Oceanering</td>
<td>63</td>
<td>46</td>
<td>Diamond Wire Saw</td>
<td>IKM</td>
<td>42</td>
<td>26.9</td>
</tr>
<tr>
<td>Gear Adap. Class 4-7</td>
<td>Oceanering</td>
<td>71</td>
<td>54</td>
<td>Diamond Wire Saw</td>
<td>Oceanering</td>
<td>71</td>
<td>50</td>
</tr>
<tr>
<td>Gear Adap. Class 4-7</td>
<td>Oceanering</td>
<td>81.6</td>
<td>77.1</td>
<td>Diamond Wire Saw</td>
<td>IKM</td>
<td>155</td>
<td>99</td>
</tr>
<tr>
<td>Norm Inspection Tool</td>
<td>Oceanering</td>
<td>13.3</td>
<td>/</td>
<td>Diamond Wire Saw</td>
<td>IKM</td>
<td>239</td>
<td>159</td>
</tr>
<tr>
<td>Linear Override Tool</td>
<td>Fugro</td>
<td>41</td>
<td>32</td>
<td>Diamond Wire Saw</td>
<td>Mirage</td>
<td>413</td>
<td>235</td>
</tr>
<tr>
<td>Linear Override Tool</td>
<td>Fugro</td>
<td>49</td>
<td>38</td>
<td>Diamond Wire Saw</td>
<td>Mirage</td>
<td>566</td>
<td>269</td>
</tr>
<tr>
<td>Linear Override Tool</td>
<td>i-Tech</td>
<td>55.4</td>
<td>44.5</td>
<td>Diamond Wire Saw</td>
<td>Mirage</td>
<td>/</td>
<td></td>
</tr>
<tr>
<td>Linear Override Tool</td>
<td>Oceanering</td>
<td>81.6</td>
<td>77.1</td>
<td>Subsea Drill</td>
<td>IKM</td>
<td>140</td>
<td>110</td>
</tr>
<tr>
<td>PH Probe</td>
<td>Oceanering</td>
<td>30</td>
<td>17</td>
<td>Wire Rope Cutter</td>
<td>Fugro</td>
<td>19</td>
<td>16</td>
</tr>
<tr>
<td>PH Probe</td>
<td>IKM</td>
<td>15</td>
<td>15</td>
<td>Wire Rope Cutter</td>
<td>Rovquip</td>
<td>21</td>
<td>/</td>
</tr>
<tr>
<td>Soft Rope Cutter</td>
<td>FET</td>
<td>9.7</td>
<td>/</td>
<td>Wire Rope Cutter</td>
<td>FET</td>
<td>22</td>
<td>/</td>
</tr>
<tr>
<td>Soft Rope Cutter</td>
<td>Fugro</td>
<td>10</td>
<td>7</td>
<td>Wire Rope Cutter</td>
<td>J2 Subsea</td>
<td>22</td>
<td>19</td>
</tr>
<tr>
<td>Soft Rope Cutter</td>
<td>J2 Subsea</td>
<td>10</td>
<td>7</td>
<td>Wire Rope Cutter</td>
<td>FET</td>
<td>43</td>
<td>/</td>
</tr>
<tr>
<td>Soft Rope Cutter</td>
<td>J2 Subsea</td>
<td>15</td>
<td>10</td>
<td>Wire Rope Cutter</td>
<td>J2 Subsea</td>
<td>45</td>
<td>/</td>
</tr>
<tr>
<td>Soft Rope Cutter</td>
<td>Fugro</td>
<td>15</td>
<td>10.5</td>
<td>Wire Rope Cutter</td>
<td>Fugro</td>
<td>45</td>
<td>32</td>
</tr>
<tr>
<td>Soft Rope Cutter</td>
<td>Fugro</td>
<td>34.5</td>
<td>23.5</td>
<td>Wire Rope Cutter</td>
<td>Fugro</td>
<td>120</td>
<td>105</td>
</tr>
<tr>
<td>Soft Rope Cutter</td>
<td>Webtool</td>
<td>34.5</td>
<td>23.5</td>
<td>Wire Rope Cutter</td>
<td>FET</td>
<td>125</td>
<td>/</td>
</tr>
</tbody>
</table>

Note. W.i.a. - Weight in air; W.i.w. - Weight in water.
3.5 Commercial underwater manipulator control systems

Commercial underwater manipulators mounted on ROVs are controlled by teleoperation systems and are completely reliant on the pilot/operator in the loop who is located on the surface vessel. The pilot observes the scene through feedback from camera and sonar systems and simultaneously takes decisions regarding the motion and remotely operates the underwater manipulator using one of a few alternate input devices. Different control methods are utilised depending on the technical capabilities of the underwater manipulators.

3.5.1 Rate control

Hydraulic underwater manipulators without joint position sensors most commonly make use of joint rate (speed) control mode. In this mode of operation, a valve pack fitted with solenoid directional control valves and/or proportional valves controls the motion of manipulator’s joint actuators. The most basic control approach with minimum equipment is achieved with directional control valves, often called switching valves because they “switch” the fluid passing through the valve from the source of flow to one of the actuator ports (Walters, 2013). Using a pilot console (Fig. 3.9) equipped with a set of 3-position ON/OFF/ON switches, the operator controls the valves in the pack, and consequently the motion of the manipulator. The size of the valve orifice determines the flow rate of the passing fluid, and thus limits the achievable joint speed. Additional adjustable flow control valves are used to regulate the flow rate and the actuator/joint speed. Achieving active actuator fluid flow control, i.e. joint speed control is possible by using proportional valves; hence the name proportional control. These valves allow infinite positioning of spools and thus provide infinitely adjustable flow. Some of them are designed to provide directional control functions as well as flow/speed control all in one valve, instead of requiring separate valves for direction and speed. Varying the level of electrical signals that control the valves enables various required flow rates, and regulating rate of change of these electrical signals enables smooth actuator acceleration and deceleration. The described switch set pilot console is the most basic input device on the pilot side and using it provides the poorest manipulation performance. A more sophisticated and intuitive input device that is used is a type of joystick which is often called a rate hand controller or a “bear-claw” (Fig. 3.11). Pushing buttons integrated on this input device achieves some hydraulic control functions and other functions by twisting it, rocking it from side to side or forth and back. Even though
it is more intuitive than the method using a switch set, it still requires quite a skilled and experienced pilot for safe, reliable, and efficient operation. Such systems utilise no joint sensors and are examples of open-loop control systems. It is the pilot with the camera view of the manipulator and the scene who closes the loop to achieve position control.

3.5.2 Position control

Some advanced underwater manipulators are equipped with position sensors in each joint such as potentiometers, analogue resolvers, digital optical encoders or solid-state linear position sensors. In case of hydraulic manipulators, these sensors combined with hydraulic servo valves allow the realisation of closed loop joint position control (set point regulation). Servo valves are electrically operated, continuously acting valves that control how hydraulic fluid is ported to an actuator. Low voltage signals for the servo valve control are amplified to accommodate sufficient power to alter the valve’s position, and the valve then delivers the required fluid power to the actuator (Dunnigan et al., 1996). The position sensor on each joint returns an electrical signal to the servo amplifier which, based on the comparison with the command signal, conditions the strength of the servo valve control voltage. Manipulator joint servo position control is most commonly achieved in a so-called master-slave configuration, with the use of a miniature master arm (Figs. 3.12 - 3.14) as an input device with similar kinematics to the slave arm (underwater manipulator). The operator physically manoeuvres the master arm whose motion is then copied to the slave arm, while simultaneously observing the slave arm’s response through a video system feedback. Each motion action of the miniature master arm is converted into varying electrical signals by position sensors, usually potentiometers placed in
3.5 Commercial underwater manipulator control systems

Fig. 3.12 Kraft Master Controller, Copyright © 2017 by Kraft TeleRobotics, Inc.
Fig. 3.13 Schilling Master Controller, Copyright © 2017 by TechnipFMC plc
Fig. 3.14 Perry Master Controller, Copyright © 2017 by Forum Energy Technologies

each joint of the master arm. Simultaneously, position sensors in the slave arm feed back varying electrical signals corresponding to the actual slave arm joint positions. These signals are compared, and any difference (error signal) initiates control signals for the servo valve, causing it to release a certain flow of fluid to the appropriate port of its hydraulic actuator, resulting in the actuator/slave arm joint moving towards the commanded position. Most master arms have several push buttons integrated for some specific functions. The most common one is “freeze” button which disables the master-slave mode and disconnects the master arm from the control loop, leaving the slave arm in the last configuration in which it was at the moment of pressing the button. This gives the operator time to rest or reconfigure the master arm posture before continuing with the operation. Other buttons can be used for jaw functions, stowing functions, etc. Master arms are integrated into the master controller which provides additional user interface functions via function keys, display, etc.

3.5.3 Force feedback control

Some master-slave underwater manipulator control systems have force feedback, which enables the pilot to sense reaction forces generated by the underwater manipulator and therefore make remote operation smoother and more intuitive. This mode of operation is referred to in the technical literature as bilateral control while the operation without it is unilateral control (Niemeyer et al., 2008). In order to provide force feedback, master arms joints integrate small electric actuators in addition to position sensors. In the case of hydraulic manipulators, the forces/torques acting on the individual joints of an underwater manipulator are measured either directly or indirectly. The former makes use of six-axis force/torque sensor located on the wrist of the manipulator, while the latter is done by measuring the pressure in the actuator supply lines and converting this information into a force/torque estimation.
In both cases, the force/torque information is then used to condition the strength of the control signals for the electric actuators located in the master arm individual joints, which results in force feedback to the operator. In addition to the improved telepresence, the compliant nature of a force feedback system considerably reduces the risk of accidental damage that the manipulator may inflict on the work site and to itself. Even with all the benefits provided with force feedback, pilots have to be highly trained to operate subsea manipulators successfully. Such adequate skill is particularly important for the use of manipulators in delicate sites such as archaeological sampling (Scaradozzi et al., 2013; Søreide and Jasinski, 2008). Additionally, if there is a push for resident ROV teleoperation of manipulators, i.e. manipulation from shore through network infrastructure, then, again pilot task load is increased, and there is a significant dependence on pilot skill and network quality. The resident ROV teleoperation concept has recently been introduced in the industry by IKM Subsea where a permanently deployed ROV system is remotely operated from shore (Offshore Engineer, 2016). Chevron adopted a similar field resident concept for AUVs (Gilmour et al., 2012). DexROV is an academic project which focuses on the same resident ROV concept with the addition of enabling the continuation of intervention operations even with significant satellite communication delays (Gancet et al., 2015).

With an increase of motion disturbances affecting the underwater vehicle and the manipulator, the task execution with a pilot in the loop becomes more difficult, more time consuming, and eventually impossible, especially in the case where the target infrastructure is in motion and the ROV for some reason, cannot clamp onto to it. The human operator can react only after the change has already happened, and therefore, even an experienced operator utilising traditional teleoperation control is likely to fail if the conditions for task execution are difficult.

### 3.5.4 Gripper control

The same miniature master arm in master-slave teleoperation mode that controls the actuators of an underwater manipulator also controls its gripper. The majority of grippers are hydraulically actuated and rate controlled in open-loop. The pilot predefined the gripper opening and closing speed in master controller settings and activates the grip function by squeezing the textured bands on the master arm wrist. Some underwater manipulators have a closed-loop servo position controlled gripper which is achieved with sensor feedback, usually a Linear Variable Differential Transformer (LVDT). A minority of suppliers offer grip force control and force feedback.
3.6 Underwater manipulator motion control

Underwater manipulators are multibody dynamic systems, and therefore two main control techniques can be implemented. The first approach regards the manipulator as a mechanical system formed by multiple independent systems determined by the number of joints it has. Each joint axis is controlled individually as a Single-Input/Single-Output (SISO) system and the coupling effects between the joints, induced during the motion due to the changing manipulator configuration, are treated as a disturbance. This type of manipulator control strategy is known in the literature as a decentralised control scheme (Siciliano et al., 2009). The opposite strategy is a centralised control scheme which takes dynamic interaction effects between the joints into account for the controller design.

To be able to design an adequate controller for an underwater manipulator, some additional factors need to be analysed such as the type of drive system used to actuate the manipulator’s joints.

Manipulators actuated with electric motors usually have high ratio gears which tend to linearise manipulator system dynamics and thus significantly reduce nonlinearity effects caused by coupling effects between the joints. However, this comes at the price of introducing significant joint friction, elasticity and backlash effects. On the other hand, manipulators actuated with direct drives such as hydraulic actuators have to deal with kinematic and dynamic coupling between the joints which is a result of configuration-dependent inertia forces, Coriolis and centrifugal forces. Hydraulic systems introduce high nonlinearities due to laminar and turbulent flow, channel geometry and friction. Additionally, hydraulic system parameters are considerably variable, dependent on the oil viscosity and the relationship between flow and pressure (Yao and Wang, 2012).

Hydrodynamic effects influencing an underwater manipulator such as buoyancy, added mass, dissipative drag and lift forces, as well as external disturbances (current, waves, tides), all add nonlinearities and uncertainties to the dynamics. Thus, modelling and control become even more complicated as the precise estimation of the hydrodynamic coefficients is impossible because they vary according to the temperature, depth, salinity, etc. (Antonelli, 2006). When a body accelerates through a fluid, some of the surrounding fluid also accelerates with the body, which creates additional inertia added to the system. This phenomenon is known as the added mass effect, and it makes dynamics model parameters become variable and uncertain (Fossen, 2011). Drag and lift forces have a similar influence on the dynamic model parameters. These forces act on a vehicle due to its movement through a viscous fluid, and since the density of sea water is significant, the magnitude of these forces
Literature review on underwater manipulators

can be significant as well. Buoyancy forces work against gravity and are dependent on the density of the fluid and the volume of the fluid displaced by the manipulator (McMillan et al., 1995). Waves, sea currents and tides cause fluid accelerations, and therefore external motion disturbances as well as forced oscillations and loads on the manipulator and the vehicle (Lapierre et al., 1998). Finally, the presence of substantial kinematic and dynamic coupling effects can occur between the base vehicle and the underwater manipulator.

Any successful control scheme applied to the underwater manipulator must be able to cope with such highly nonlinear, time-varying and uncertain dynamics. There has been an abundance of control schemes proposed for underwater manipulators over more than two decades; however, the majority of research has been done on a theoretical level with control performance validation done through simulations. Some control strategies have been tested on real experimental underwater manipulators but the work done on commercial underwater manipulators is scarce. The lack of research on the application of control systems onto standard ROV manipulator systems is not surprising as the state-of-the-art commercially available manipulators are quite expensive and often have integrated motion controllers with limited access to the control implementation. Any attempt to validate control approaches using commercial manipulators requires a significant amount of modification on the manipulator hardware.

3.6.1 Decentralised control

Control loop feedback mechanisms which integrate proportional (P), integral (I) and derivative (D) terms in different variations for basic controllers (P, PI, PD, PID, etc.) as set-point regulators, have been present in the industry for decades (Choi and Chung, 2004). Utilising these PID type control laws for underwater manipulators within a decentralised joint space control strategy offers the simplicity of implementation and low computational cost. The trajectory planning for commercial underwater manipulators falls under the joint space PTP method (initial to final joint configuration) due to its master-slave teleoperation approach. However, PID based control laws provide poor dynamic accuracy when trajectory tracking comes into play, and the dynamic performance of the manipulator varies according to its configuration (Khalil and Dombre, 2004).

Regardless of its limitations, a number of authors including Smith et al. (1994) report utilising PID based joint control laws in a decentralised control scheme for underwater manipulators. Dunnigan et al. (1996) incorporated a fixed-gain PID controller for each manipulator’s joint, and realising that the control performance de-
3.6 Underwater manipulator motion control

grades when the manipulator operates at different points in the workspace, proposed investigating a self-tuning adaptive control approach. The outcome was an adaptive SISO self-tuning pole-placement joint angle controller which provides benefits over a fixed gain PI/PID controller for a range of different operating conditions (Clegg et al., 2001). Ishimi et al. (1991) propose another adaptive PID controller with automatic gain tuning in accordance with arm posture changes and with feedforward compensation for gravity, buoyancy, speed and acceleration. Zhang et al. (2013) also report analysing the performance disadvantages of traditional PID control and propose a PI algorithm with variable gains. Xu, Sakagami, Pandian and Petry (2005) propose utilising fuzzy logic theory for adaptive PD controller gain tuning and Yao and Wang (2012) propose a Model Reference Adaptive Control (MRAC) scheme for individual joint control where the interaction between other joints and hydrodynamics influence is considered as external disturbance and controller parameters are adjusted on-line in real time.

3.6.2 Centralised control

Other authors have focused on nonlinear centralised control schemes which take advantage of the knowledge of an underwater manipulator dynamics model to compensate for the nonlinearities by eliminating them rather than reducing the effect induced by them and therefore enhance the trajectory tracking performance (Siciliano et al., 2009).

Liceaga-Castro et al. (1991) reports one of the early works on the investigation of underwater manipulator model-based control and proposes a nonlinear model matching controller. The manipulator dynamics model includes some hydrodynamic effects which are calculated according to Morison’s equations (Morison et al., 1950). Schjølberg and Fossen (1994) derived an underwater manipulator dynamics model with most dominating hydrodynamic forces included using an iterative Newton-Euler algorithm and propose an inverse dynamics control approach. This approach uses the feedback linearisation method to completely linearise a nonlinear system leaving it linear and decoupled so that a much more straightforward stabilising linear controller can be utilised for trajectory tracking. However, this approach assumes an exact knowledge of the dynamics model which is impossible to measure or estimate. The model is in reality known with a degree of uncertainty, and imperfect cancellation of dynamics terms is guaranteed. Therefore, this approach does not have adequate robustness as it is sensitive to time-varying and uncertain model parameters and external disturbances (Siciliano et al., 2009).
To design a controller that can counteract the effects of imperfect compensation and thus deal to some extent with variable parameters and disturbances and therefore relax the unrealistic assumption of the accurate knowledge of the underwater manipulator dynamics model parameters, some researchers have investigated integrating robust and adaptive control strategies. The former counteracts the effects of the model approximation, and the latter adapts the model parameters to those of the real underwater manipulator dynamics model (Siciliano et al., 2009).

Lee and Choi (2000) propose a robust controller designed by combining a computed torque controller and a Sliding Mode Controller (SMC) with a multi-layer neural network controller which acts as a compensator, maintaining the control performance when the initial uncertainty assumptions cease to be valid. Another robust control scheme is presented by Yuh et al. (2001) which consist of a disturbance observer controller, which transforms a nonlinear underwater manipulator system with uncertainties into a simple model with disturbance error, and a non-regressor based adaptive controller designed according to the simplified model. Some authors propose robust trajectory tracking controllers for underwater manipulators based on the sliding mode control strategy and the dynamics model for estimating uncertainty bounds (Kwon et al., 2000; Xu et al., 2006, 2007). Some authors propose using fuzzy logic heuristics for sliding mode controller adaptive gain tuning (Xu, Pandian and Petry, 2005). Mohan (2011) proposes an observer-based PD backstepping robust nonlinear control technique for underwater manipulators. Esfahani et al. (2013) presents a control scheme where an artificial immune system algorithm with wavelet mutation is used to derive optimal parameters for the conventional sliding mode controller. A modified sliding mode control scheme, namely terminal SMC is proposed by Venkatesan et al. (2014) with a disturbance observer integrated for dealing with disturbances and uncertainties. Mohan and Kim (2015b) present another robust controller with uncertainty/disturbance estimator. This controller integrates approximated inverse dynamic model output as a model-base portion of the controller, uses a feed-forward term to enhance the control activity, estimates a perturbed term to compensate for the external disturbances and unmodelled dynamics, and has a decoupled nonlinear PID as a feedback portion to enhance closed-loop stability and account for the estimation error of uncertainties.

3.6.3 Neural and fuzzy control

Some authors have been investigating neural networks and fuzzy logic theory for designing control strategies capable of resolving the nonlinear control problem without taking into account any knowledge of the underwater manipulator dynamics.
Wang et al. (2008) propose a hybrid control method based on the integration of fuzzy logic control with a cerebellar model articulation controller, which is a neural network type. Suboh et al. (2009) present another fuzzy hybrid control scheme where a Takagi-Sugeno fuzzy controller is merged with a model reference adaptive controller equipped with PI adjustment mechanism based on the previous work of Golea et al. (2002). Pandian and Sakagami (2010) present a neuro-fuzzy PD control scheme for underwater manipulators where a fuzzy gain tuning method is utilised for adaptation under uncertainties and disturbances while the neural network is used to approximate the dynamics of the underwater manipulator and to add a feedforward compensation input to the PD fuzzy controller.

3.7 UVMS motion control

Underwater manipulators are usually not standalone systems, as they are mounted on underwater vehicles, in most cases work-class ROVs. These systems are often referred to as UVMS in the literature. Work-class ROV intervention operations which include using underwater manipulators require at least two highly skilled operators. One operator to pilot the ROV, trying to keep it as stable as possible by compensating for external motion disturbances (sea current, waves, tides) and ROV motion induced by manipulator’s reaction forces/moments, while the other operator performs the actual teleoperated manipulation task. This mode of operation is required in the case the ROV is not operating on the seabed and when there is no possibility to clamp it onto the underwater structure. Such situation can occur if the surrounding environment does not provide adequate conditions for a safe and secure connection or if the underwater structure is not designed to be clamped onto. The significant disadvantage of teleoperated control becomes prominent in harsh sea conditions because even very skilled pilots can react only after the disturbance event has already happened, which induces significant delays in the system. Due to many handicaps that contribute to the task performance complexity, pilots eventually get fatigued which leads to significant reduction of task effectiveness (Cooke, 2006). Therefore, a plausible approach to solving these problems is to implement semi-automated or fully automated UVMS control methods.

Since underwater vehicle and manipulator motion are coordinated separately, for ROVs in current use, the straightforward approach for UVMS control implementation is to decouple the vehicle and the manipulator and regard the whole system as two systems and control them independently. Conventional station keeping algorithms can be utilised for underwater vehicle control while the underwater manipulator
motion control can be utilised as if it were on a fixed base, having the automated manipulator system carry out the prescribed tasks through arm motion alone. Vehicle position set point can be replanned when the target is out of the workspace of the manipulator. This approach simplifies the manipulator control but places a heavy burden onto the underwater vehicle control system as it needs to have a drive system with sufficient dynamic capabilities and precision navigation so that the overall system can achieve adequate manipulation performance. Articles describing ROV precision navigation and motion control are presented by Omerdic and Toal (2012); Toal et al. (2011), while sole manipulator control algorithms have been outlined in the previous section. The problem of this approach is that it does not take into account the dynamic and kinematic coupling that occurs between the manipulator and its base vehicle, which can significantly degrade the control effectiveness of the whole UVMS. The dynamic coupling arises as the manipulator, while in motion, transmits forces and moments which are variable in magnitude and direction to its base vehicle. These forces and moments alter the pose of the base vehicle and hence the manipulator end-effector position and orientation, which is regarded as kinematic coupling (Dunnigan and Russell, 1994). The factors that cause these coupling effects are the relative size, weight and shape of the manipulator compared to its base vehicle. The lower the manipulator’s weight and size compared to the underwater vehicle the better, as coupling can then be either neglected or taken into account and dealt with appropriately. Table 3.2 presents relative manipulator-to-vehicle weight ratios for the existing typical heavy, medium and light work class ROVs. When coupling effects are significant, advanced control approaches have to be adopted.

### 3.7.1 ROV compensation for coupling effects

One solution to deal with coupling effects while the manipulator is operating is to decouple the UVMS and run a separate control loop for the underwater base vehicle alone, keeping it in a fixed pose using the propulsion system. This approach has been investigated by a number of researchers who propose advanced station keeping algorithms for the underwater vehicle, taking dynamic and kinematic coupling effects into account.

Koval (1994) proposes an automatic vehicle stabilisation method where the manipulator caused vehicle motion is compensated for by feedforward terms based on manipulator kinematics and simplified dynamics (without hydrodynamic forces). Dunnigan and Russell (1994) demonstrated the effect of dynamic coupling between manipulator and ROV through numerical simulations and propose a scheme to reduce it using a variable structure control law where forces/torques affecting the ROV are
3.7 UVMS motion control

deduced from the equations of motion which include simplified hydrodynamic terms. McLain et al. (1996) experimentally demonstrate the significance of hydrodynamic coupling between the single-link manipulator and its base vehicle when no vehicle control is applied and propose a coordinated control approach for the UVMS by incorporating model-based hydrodynamic coupling information into the vehicle control law.

An alternative approach for ROV stabilisation due to the motion induced by an operating manipulator and external disturbances is investigated by Kato and Lane (1995) where multiple smaller arms, other than the main manipulator, are used as paddles for motion compensation.

Interaction forces that occur between the manipulator and its base vehicle can be measured using a six-axis force/torque sensor mounted at the manipulator base. Some authors propose utilising controllers for vehicle station keeping that use these measurements to compensate for coupling effects by adjusting thruster commands to correct the position of the vehicle (Fraisse et al., 2000; Lapierre et al., 1998).

In case the force/torque sensor is unavailable, Ryu et al. (2001) propose a controller which is based on the developed disturbance observer (Geffard et al., 2000) for the estimation of interaction forces between the ROV and the manipulator. Vossoughi et al. (2004) present a similar approach where forces/moments are estimated based on the dynamic model and used as a feedforward portion of the ROV controller. To predict dynamic coupling forces/torques, Soylu et al. (2005) utilise an Articulated-Body Algorithm (ABA) which is based on the UVMS feedback states and a sliding mode controller for UVMS station keeping. Soylu et al. (2009) present using the same ABA algorithm along with a combined H-infinity-Sliding-Mode control scheme for underwater vehicle station keeping.

Lynch and Ellery (2014) propose an alternative vehicle stabilisation method where the focus is on the control of the UVMS barycenter rather than on the vehicle position. Antonelli and Cataldi (2014) reports designing an adaptive and recursive low-level controller for the vehicle assuming that the manipulator is controlled independently by a joint-based controller and using the information of desired manipulator trajectories within the vehicle controller. In the ROV case, apart from vehicle-manipulator dynamic coupling effects, hydrodynamic and gravitational forces acting on the tether also create internal forces on the ROV. Soylu et al. (2010a) reports addressing this problem and performing numerical simulations which reveal the extent to which the tether affects the manipulator dynamics; the same author proposes utilising a model-based sliding mode controller for ROV station keeping capable of dealing with dynamic coupling effects caused by the tether.
3.7.2 UVMS motion control

Instead of decoupling it into two separate systems, a UVMS can be addressed as a single system which can utilise different control approaches capable of dealing with coupling effects occurring between the manipulator and the base vehicle.

The primary focus for a number of researchers investigating this approach is on the UVMS set-point regulation. Lizarralde et al. (1995) propose a velocity-less PD control set-point regulator which drives the vehicle and the manipulator to the desired attitude and position. Another example of UVMS set-point regulation is presented by Antonelli and Chiaverini (1998a), where a robust sliding mode based control approach is used, and by Sun and Cheah (2004), with the use of a generalised adaptive saturated PD controller with gravity regressor for gravity and buoyancy compensation.

Other authors focused on deriving the detailed UVMS dynamics model and propose the implementation of conventional model-based control schemes which can be applied for trajectory tracking. Utilising a Newton-Euler approach, Fossen (1991) derived equations of motion for a UVMS considered as a micro-macro manipulator, a specific combination of parallel and serial mechanical structures, where the manipulator gives fast and accurate end-effector motion and the vehicle is the slower positioning part; the same author proposes an adaptive controller for the UVMS, which is based on the work of Slotine and Weiping (1988). Mahesh et al. (1991) derive equations of combined UVMS motion using a vector-dyadic method and have designed a coordinated adaptive control strategy where parameters of the linearised coupled model are estimated on-line, and used by a discrete-time adaptive velocity controller for self-tuning. McMillan et al. (1995, 1996) report on the developed efficient UVMS dynamics simulation algorithm which includes major hydrodynamic effects which can provide aid in the design of control algorithms. Tarn et al. (1996) use Kane’s method for the development of a dynamics model of an underwater vehicle equipped with an n-DOF manipulator also including major hydrodynamic terms and thus provide a solid background for the design of UVMS control algorithms.

Schjølberg and Fossen (1994) propose a feedback linearisation approach followed by a derivation of the detailed dynamic model including the most significant hydrodynamic terms. A similar approach based on feedback linearisation has been investigated by some other researchers. Schjølberg and Egeland (1996) utilise two different spacecraft-manipulator system control schemes and apply them on the UVMS; Tarn and Yang (1997) address a multiple manipulator UVMS model which includes major hydrodynamic forces; Wilson et al. (2011) propose a computed
torque controller for UVMS; Mohan (2013) present a model reference UVMS control scheme, and Korkmaz et al. (2013) present an inverse dynamics control method for UVMS trajectory tracking where separate tasks are assigned for the end-effector and the vehicle. Mohan and Kim (2012) propose a nonlinear control scheme based on the feedback linearisation using indirect knowledge of the system dynamics and external disturbances via an extended Kalman filter. The same authors in (Mohan and Kim, 2015a) propose a coordinated motion control scheme using a disturbance observer in task space. Londhe et al. (2017) propose a Robust Single Input Fuzzy Logic Controller (RSIFLC) applied for task-space trajectory control that consists of feedback linearisation and feed-forward controllers along with a single input fuzzy controller and an uncertainty estimator.

The main drawback of the feedback linearisation approach is that it assumes the exact knowledge of the system dynamics or at least a close estimation which is unrealistic and does not guarantee robustness to model parameter variation and uncertainties. Dunnigan and Russell (1998) emphasise the significance of dynamic coupling through computer simulations on a six DOF vehicle equipped with a three DOF manipulator and integrate closed-form manipulator disturbance expressions neglecting hydrodynamic terms into a sliding controller which is capable of dealing with parameter uncertainties to some extent. For improving dynamic coupling modelling accuracy and achieve better UVMS coordinated motion control, Leabourne and Rock (1998) present research on an empirically determined hydrodynamic manipulator model, acquired by real experiments with a two-link manipulator mounted on a free-floating vehicle. De Wit et al. (1998a) emphasise that manipulator-to-vehicle coupling effects are dominant over the vehicle-to-manipulator ones and that the feedback compensation is only needed to overcome the coupling effects from the manipulator. The UVMS is divided into two subsystems, one of which is fully independent of the other (manipulator) while the other (vehicle) is perturbed by the first subsystem (manipulator). An approach based on a singular perturbation is proposed, as an alternative to existing approaches requiring full model knowledge to compensate for vehicle/manipulator coupling and their nonlinear effects, by a partial linearising decoupled controller. Based on the same decoupling approach, De Wit et al. (1998b) present a robust non-model-based controller for the UVMS which consists of a linear PD controller for the manipulator and a robust nonlinear controller for the underwater vehicle based on the work reported by Williamson and De Wit (1995). A comparative study of the proposed controller neglecting previous assumptions such as that there is no saturation in the thrusters, infinitely accurate sensors, infinitely small sampling time and absence of thruster nonlinearities is presented by Diaz et al. (1998). Based on the same singular perturbed model, De Wit et al.
(2000) propose robust nonlinear feedback control for the UVMS with composite dynamics, which offers a considerable compromise between control complexity and closed-loop performance.

A modular approach for UVMS control is presented by Antonelli, Caccavale and Chiaverini (1999); Antonelli et al. (2004) where an adaptive tracking controller with virtual decomposition (Zhu et al., 1997) is adopted. The proposed control approach exploits the serial-chain structure of the UVMS to decompose the overall motion control problem into separate simple rigid body control problems, i.e. manipulator links and the vehicle. Sarkar and Podder (2001) use a quasi-Lagrange method to derive UVMS dynamic equations of motion which include thruster dynamics and utilise a computed torque control law for UVMS trajectory tracking. Unlike most control methods based on the computed torque, Ishitsuka et al. (2004) propose a resolved acceleration control method for the UVMS. The proposed control method is verified by numerical simulations reported by Ishitsuka and Ishii (2005, 2006). An extension of this algorithm which includes compensation for disturbances caused by hydrodynamic forces acting on the vehicle along with the experimental verification can be found in Yatoh and Sagara (2008). Sagara et al. (2010) present another addition to the algorithm where the disturbance compensation is utilised both for the vehicle and the manipulator followed with the experimental validation. Finally, the same method is enriched for a dual manipulator control scheme (Sagara and Ambar, 2014). Xu, Abe, Sakagami and Pandian (2005) present a non-adaptive model-based sliding mode controller for UVMS based on the decentralised dynamics with simulation validations showing that the proposed sliding mode controller provides accurate and robust tracking performance, superior to that obtained with a traditional PD controller. Periasamy et al. (2008) present the development of a UVMS dynamic model using the bond graph modelling technique where major hydrodynamic effects and coupled dynamics are included, and they have designed a PD plus buoyancy compensation control for the UVMS end-effector trajectory tracking. Han et al. (2011) propose a robust UVMS control approach where a nonlinear H-infinity optimal control is utilised as an external tracking control loop and a disturbance observer as an internal disturbance compensation loop.

During the execution of the UVMS intervention task, the manipulator motion causes the shift of the orientation equilibrium of the whole UVMS, which causes restoring moments and forces to change as well and this eventually leads to additional spontaneous end-effector motion. To exploit this phenomenon, Han and Chung (2014) propose a robust adaptive control scheme that uses variant restoring moments to control the UVMS actively.

56
3.7 UVMS motion control

Most of the existing UVMS control approaches neglect the existence of nonlinear thruster dynamics. Taira et al. (2010) present a regressor-based adaptive controller for UVMS with thruster dynamics where adaptive control inputs are composed of adaptive feedforward signals including regressors of dynamic system models and error feedback signals. For simplification of the complex structure due to regressors, the same authors propose an adaptive controller in Taira et al. (2012) that uses radial basis function networks (Haykin, 1998) instead of the regressors, while in Taira et al. (2011) they propose a robust controller where the feedforward term is entirely removed. The performance of the robust controller is improved with pre-compensators including integral actions (Taira et al., 2014).

To avoid the unrealistic assumption of exact knowledge of UVMS dynamics and the complexity of estimating close to the real dynamic model, some authors have resorted to adaptive control methods that are independent of any model knowledge. Lee et al. (2000); Lee and Yuh (1999) report on a non-regressor based adaptive control scheme for UVMS trajectory tracking based on a bound estimation method and a parameter adaptation algorithm for adjusting the controller gains based on the performance of the system rather than the knowledge of the dynamic model. A sliding mode type controller with fuzzy logic implementation for adaptive gain tuning is proposed by Xu, Pandian and Petry (2005). Using fuzzy logic heuristics for decentralised PD type controller gain tuning is also presented in Xu, Sakagami, Pandian and Petry (2005). Sakagami (2009) propose using an iterative learning control approach to deal with the manipulator-to-vehicle coupling effects which assumes that the motion of the manipulator that is to be compensated for is known in advance.

3.7.3 Manipulator compensation for coupling effects

Some authors address the end-effector trajectory tracking control problem of UVMS in a different manner where the focus is on developing advanced manipulator control methods which alone compensate for the movement introduced by coupling effects between underwater vehicle and manipulator and the external disturbances.

One of the early references on compensation of the underwater vehicle motion as a result of external (tidal) disturbances is by Ishimi et al. (1991), where a sway compensation controller is developed which uses the Inertial Navigation System (INS) signal, transforms it into a displacement of the manipulator and adds it to the position command signal as a feedforward term, thus cancelling out the sway of the vehicle with manipulator motion alone. Chung et al. (2000) propose modelling a UVMS as a class of underactuated robotic system with the assumption that the
Literature review on underwater manipulators

vehicle is free floating and proposed a robust control algorithm where the ROV is modelled as a passive joint, and manipulator joints are modelled as active joints. A robust control method is realised with a nonlinear feedback disturbance observer plus PD control scheme and applied to each active joint. Kim et al. (2003) propose an active damping two-time scale control approach where the vehicle is passive or controlled by a simple P-type controller and the manipulator is controlled with a two-time composite scale (fast and slow) controller designed according to a partial decoupling approach. Hildebrandt, Christensen, Kerdels, Albiez and Kirchner (2009) propose an algorithm implemented on the manipulator system that can actively compensate for the ROV motion based on the model-based vehicle motion prediction algorithm.

Some authors have investigated compensation schemes such that the manipulator control law is a function of the vehicle velocity. In the TRIDENT project, Simetti et al. (2014) propose a dynamic programming approach which allows an optimal manipulator movement based on the current measured vehicle velocity. Simetti et al. (2017) report experimental results of the MARIS project including a parallel task-priority inverse kinematics solution for manipulator control also based on vehicle velocity.

Some authors propose using vehicle motion to complement the capabilities of the manipulator by providing extra DOFs to achieve manipulator end-effector trajectory tracking, rather than having the underwater vehicle passive or controlled for station keeping (Antonelli et al., 2004, 2000; Antonelli and Chiaverini, 2003a,b). However, using thrusters for manipulator end-effector positioning is inefficient from an energy consumption perspective. This results due to the complexity of controlling the underwater vehicle in hovering and lateral directions as well as because of significant difference in vehicle and manipulator inertias (Mohan et al., 2012).

3.8 Kinematic control and motion planning

To improve the overall performance of underwater manipulation, many researchers utilised kinematic level enhancements. Some focus on the improvement of conventional pilot teleoperation techniques while others resort to the utilisation of semi-autonomous and fully autonomous control techniques in the hope of transferring the operator from direct control to a supervisory position. Some of the early research on teleoperation control enhancements includes the development of a human-machine interface where a choice of possible remote control input devices is available, including keyboard, mouse, master arm or joystick, each of which
provides benefits for different tasks (Larkum and Broome, 1994). Another important approach that has been investigated by many researchers includes real-time motion simulations and virtual graphical reconstruction of the manipulator workspace including workpiece solid models presented on the monitor display. Such tools provide supplementary visual aids in the traditional teleoperation by helping the operator to perceive the posture of the underwater manipulator and possibly avoid collision with the environment. Relevant pioneering research on this topic is reported by Broome et al. (1995); Ishimi et al. (1991), while more recent research work can be found in Hildebrandt, Kerdels, Albiez and Kirchner (2008); Jun et al. (2009); Zhang et al. (2003).

A number of researchers have investigated Cartesian space teleoperation schemes as such an approach is more intuitive from the human perspective, and advantages that it offers for various underwater manipulation tasks are quite clear, especially for the “peg-in-a-hole” type tasks. Operational space teleoperation techniques implemented and tested on work class ROV are reported in Jun et al. (2008, 2004, 2009). A workspace control approach which is based on the implementation of the differential inverse kinematics algorithm is developed where the operator uses an input device to control the manipulator end-effector velocity and hence its position and orientation in Cartesian space. Hildebrandt, Albiez and Kirchner (2008) present optimal direct kinematics and closed-form analytical inverse kinematics solutions with particular respect to computational simplicity while maintaining high numerical precision. This algorithm has been applied to control a commercial underwater manipulator Schilling Orion 7P. Fernandez et al. (2013) also address Cartesian space control with implementation on a modified version of the ECA ARM 5E. Huo et al. (2013) utilise an analytical inverse kinematics solution for the six DOF underwater manipulator developed by the Shenyang Institute of Automation of the Chinese Academy of Sciences. Implementing inverse kinematics algorithms and developing Cartesian control methods opens a lot of new possibilities for semi-automated supervisory control schemes and fully-automated manipulation solutions. In the former, the operator selects a task via a user interface and then observes and monitors while the task is being executed, while in the latter the task is sensor triggered. Moreover, this approach brings underwater manipulation closer to totally automated industrial (terrestrial) robotics. This is because the majority of industrial robot arms have kinematics control (end-effector motion trajectory planning) carried out in the operational (Cartesian) space, followed by inverse kinematics implementation and finally low-level motion control (motion trajectory tracking control) performed in the joint space. On the other hand, commercial underwater manipulators are teleoperated, and therefore kinematics set point control is carried out in joint space.
since the master arm motion is recorded in joint space and then copied directly to
the slave arm.

One of the earliest references on the implementation of supervisory control for
underwater manipulators is by Ishimi et al. (1991) who describe how after an operator
has chosen a task, motion planning is utilised automatically based on the knowledge
of the environment, work tools, platform, etc., and then transformed into data for ma-
nipulator motion control. Broome et al. (1995) also discuss semi-automatic and fully
automatic modes backed up with motion planning algorithms and inverse kinematics
implementation. Zhang et al. (2003) propose semi-autonomous manipulator control
under operator’s supervision with a possibility of using a visualisation display as a
tool for outlining tasks. Jun et al. (2009) present a preprogrammed waypoint task
control approach where the manipulator can execute a task for which a path was
taught by the operator in advance. The efficiency of this approach has been experi-
mentally tested on ROV mounted Schilling Orion 7P manipulators performing a soil
coring task. Hildebrandt, Kerdels, Albiez and Kirchner (2008) propose integrating
additional features to the online virtual reality 3D visualisation that could help the
operator during the supervisory control such as visualisation of end-effector paths
or previewing possible manipulator configurations before the actual movement as a
result of motion planning algorithms. This is achieved by a secondary 3D visualisa-
tion of the manipulator projected as a semi-transparent overlay. Another feature that
can aid the pilot in teleoperated mode is proposed by Albiez et al. (2009) based on
the workspace analysis of the manipulator. A computer is continuously calculating
the distance to the workspace border, enabling the operator to get information about
current dexterity of the underwater manipulator, the ranges where the dexterity does
not exceed a given limit, and the possibility for automatic motion.

Some authors propose integrating collision avoidance algorithms into supervisory
and completely autonomous control schemes (Broome et al., 1995). Ishibashi et al.
(2001, 2002) use genetic algorithms for obstacle avoidance of both stationary and
dynamic objects where both the end-effector position and posture of the manipu-
lator is utilised for motion planning. David et al. (2007) propose integrating an
active collision avoidance feature based on the MARGITTE 3D graphic supervisor
(Gravez et al., 2003) into the enhanced teleoperation scheme by coupling the real
robot manipulator with a simulated one to generate movement for both of them.
Since the simulated manipulator has integrated algorithm for automatic obstacle
avoidance in the virtual reality, the operator can handle the real slave manipulator
safely without taking care of real environment obstacles, assuming perfect knowl-
edge and modelling of obstacles. This control scheme was developed for use in the
3.8 Kinematic control and motion planning

nuclear environment but was tested on a Cybernetix Maestro hydraulic underwater manipulator that could be applied for subsea applications.

Some authors address kinematic control and motion planning algorithms on a UVMS level. Podder and Sarkar (2000) propose a dynamics-based trajectory planning approach capable of generating both kinematically admissible and dynamically feasible joint space trajectories for systems composed of heterogeneous dynamics such as UVMS, characterised by a much slower dynamic response of the underwater vehicle compared to that of the manipulator. In the proposed approach, task-space trajectory is represented in terms of Fourier series and various frequency components from the series are used to generate reference joint-space trajectories based on natural frequencies of the respective subsystems. Shim et al. (2013) address UVMS motion planning and control based on ROV position estimation using extended Kalman filter and propose a precise dynamic workspace control method where the manipulator is controlled to move in a straight line while the ROV is assumed to be floating. A method for UVMS global motion planning, capable of generating feasible and obstacle-free task paths based on the 4D bump surface concept is presented by Sotiropoulos et al. (2013). The PANDORA project has explored learning the trajectory of the vehicle and the end-effector by demonstration (Carrera et al., 2014) to accomplish the valve turning, with experiments in a tank environment. In the MER-BOTS project Youakim et al. (2017) have used the motion planning ROS package MoveIt! to compute reference trajectories for the UVMS. Utilising machine learning for semi-autonomous manipulation has been the focus of DexROV project (Gancet et al., 2016; Havoutis and Calinon, 2018; Tanwani, 2018). Bae et al. (2018) propose a method based on genetic algorithm to optimise the configuration of an UVMS with a dual-arm manipulation system for the valve handling task ensuring the maximum dynamic manipulability through a desired end-effector path. Additional advanced control problems considering dual arm UVMS are presented in (Ambar et al., 2015; Farivarnejad and Moosavian, 2014; Simetti and Casalino, 2015). Even more complex problems, such as cooperative UVMS are addressed in (Conti et al., 2015; Simetti and Casalino, 2017). The Ocean One project developed a bimanual force controlled “humanoid” robot that affords immediate and intuitive haptic interaction in oceanic environments (Khatib et al., 2016).

3.8.1 Redundancy

If an underwater vehicle is station keeping or fixed, e.g. sitting on the seabed or clamped onto a structure, achieving arbitrary position and orientation with end-effector within its workspace is possible only if the manipulator has at least six
DOFs. Manipulators with more DOFs are in a kinematic sense considered inherently redundant (Siciliano et al., 2009), and their redundant DOFs can be advantageously exploited in different ways for achieving secondary objectives such as avoiding obstacles, minimising energy consumption, etc. However, since a free-floating vehicle has its own six DOFs, those can be exploited together with manipulator DOFs for achieving arbitrary end-effector position and orientation as well as for redundancy resolution.

Some authors addressed this and investigated approaches which take UVMS redundancies into account to generate trajectories corresponding to given tasks, while the extra DOFs are used to assign additional motion without impeding the end-effector’s performance. Antonelli and Chiaverini (1998b) propose a UVMS task-priority inverse kinematics approach for redundancy resolution with robustness to the occurrence of algorithmic singularities where redundancy is exploited for reducing power consumption and increasing the manipulability of the system. Sarkar and Podder (1999, 2001) address the kinematic redundancy resolution with the minimisation of the total drag on the UVMS. Antonelli and Chiaverini (2003a,b) present an approach where redundancy is exploited to find an optimal posture for a manipulator by obtaining either maximum manipulability, staying away from the joint limits or minimising the vehicle’s roll and pitch angles. Jun et al. (2008, 2004) report redundancy resolution algorithms with the focus on obtaining an optimal manipulator posture. Ismail and Dunnigan (2009) propose a redundancy resolution to minimise gravity and buoyancy loading of the UVMS. Soylu et al. (2010b) address redundancy resolution with different secondary objectives such as avoiding joint limits, singularities and high joint velocities, keeping the end-effector in sight of the onboard camera, minimising the ROV motion and minimising the drag forces on the ROV. Han and Chung (2014); Han et al. (2011) focus on the optimisation of UVMS’s restoring moments. A fuzzy redundancy resolution approach is investigated by Antonelli and Chiaverini (2003a), Antonelli and Chiaverini (2003b) and dos Santos et al. (2006). Casalino et al. (2012) and Simetti et al. (2013) propose alternative prioritisation of tasks for the coordinated motion control of a UVMS. Within this strategy, end-effector position and orientation are given a secondary priority objective while the high priority tasks keep each joint within its range of motion, keep the manipulability measure above the given positive threshold, maintain horizontal vehicle attitude in the desired range and keep the vehicle pose in the desired range. Mohan and Kim (2015a) propose a task space coordination control scheme that is able to track the given desired path and also perform power efficient trajectories. In the PANDORA project, Cieslak et al. (2015) propose an approach that combines learning and task priority and present experimental trials. Kang et al. (2017) propose
3.9 Force control

Subsea intervention tasks carried out with underwater manipulators often demand extensive contact with the environment (connector plugging, valve turning, etc.). Therefore, end-effector trajectory tracking control has to be backed up with the implementation of interaction force-torque control of the end-effector with the environment. This is because, under pure manipulator position control, slight deviations of the end-effector from the planned trajectory can cause the manipulator to either lose contact with the surface or press too strongly on the surface and possibly result in disastrous consequences by generating extremely large interaction forces (Spong et al., 2006).

Force control has been broadly investigated for fixed base industrial robot arms. The most common control schemes in the literature are passive and active compliance control, impedance control and hybrid impedance control, which can be found in Siciliano et al. (2009); Siciliano and Villani (2012); Spong et al. (2006).

One of the first to address this control problem for an underwater manipulator are Dunnigan et al. (1996) who propose a hybrid position/force control scheme and present the results achieved through practical tests on the Slingsby TA9 hydraulic manipulator. In the DexROV project, Lillo et al. (2016) demonstrate how force regulation can be accomplished at a kinematic level both for simple tasks such as closing a valve and pushing a button, or for more complex intervention such as pipeline weld inspection activities that require contact with the target and force regulation at the same time (Casalino et al., 2017). Barbalata et al. (2018) propose a variable structure model dynamic controller for an underwater manipulator performing motion and interaction tasks in test tank with an electrical prototype manipulator. Additional research on underwater manipulator force control algorithms with validation through extensive numerical simulations can be found in Antonelli, Sarkar and Chiaverini (1999); Antonelli et al. (2002); Cataldi and Antonelli (2015); Cui et al. (1999); Cui and Sarkar (2000); Kajita and Kosuge (1997); Lapierre et al. (1998); Lemieux et al. (2006).


3.10 Collision avoidance

ROVs usually integrate multiple sensing devices including camera systems, forward-looking sonars, and other sensors and tools (Capocci et al., 2017). A lot of the equipment is installed on the front side of the ROV, therefore inside the manipulators’ workspace. ROV pilots must be very cautious during teleoperation not to damage the valuable equipment, the ROV’s body, the targeted structure, and the manipulators themselves. Reduced visibility due to water turbidity and poor 3D perception due to the 2D video feedback only adds to the complexity of telemanipulation. As a result, tasks that might seem straightforward can become very difficult and tiresome even for very skilled operators, significantly affecting their performance. Moreover, there is a trend towards resident ROV teleoperation of manipulators, i.e. manipulation from shore through telecommunication network infrastructure (Gancet et al., 2016; Offshore Engineer, 2016). Such a setup increases the operator’s task load and emphasises the importance of pilot skills and of the network quality, which might introduce delays in control and sensory feedback. Due to the complexities mentioned above, subsea operations are time-consuming, and therefore, costly.

Most advanced commercial subsea ROV manipulator systems have an integrated software function to limit the range of motion of the manipulator’s joint axes. This is often done to prohibit the access to certain areas on the base vehicle and protect the equipment. However, limiting the manipulator’s motion in joint space is not efficient as it enormously limits the manipulator’s operational workspace. Moreover, it does not prevent the two manipulators from colliding, which is an important issue as these manipulators are capable of exerting considerable forces that may cause severe mechanical damage. As two manipulators with overlapping workspaces simultaneously operate in the same working area, a real-time collision avoidance algorithm is required, capable of detecting and prohibiting motion commands which would result in a collision and allowing only collision-free motion. Each manipulator represents a dynamic obstacle to the other manipulator. Therefore it is necessary to address the collision between the two as well as between each of them and other obstacles.

Over the last three decades, various researchers have been investigating collision detection and avoidance methods for robotic arms, chiefly as part of collision-free motion planning algorithms (Chang et al., 1994; Lee and Lee, 1987; Onda et al., 1990). Much of the research focus has been on developing algorithms and evaluating them through simulations and experimental laboratory tests using electro-mechanical autonomous industrial robot arms typically used in manufacturing industry. Many of the proposed approaches are off-line and designed for pre-programmable robot
motion planning, and therefore, not suitable for commercial ROV manipulator systems, which are controlled in teleoperation mode and utilise PTP motion planning, where the full path cannot be known in advance. For this reason, any collision avoidance implementation for the teleoperated ROV manipulators has to work online. There is a research trend towards automating ROV intervention operations (Schjølberg and Utne, 2015), and in the case the ROV industry adopts it, off-line collision avoidance methods might become suitable. However, this is still in its early research and development stage, and fully automated manipulation systems do not exist yet in the global fleet of work-class ROVs.

Numerous on-line collision detection methods based on various geometrical modelling approaches have been proposed. Czarnecki (1994) proposes an approach that includes discretising the Cartesian space into cuboids and forming a collision map based on obstacle-unaware trajectories. Greenspan and Burtnyk (1996) describe a model-based real-time collision avoidance method in which the manipulator links are modelled as sets of spheres and obstacles as a weighted voxel map. Henrich et al. (1998) propose an implicit and discretised configuration space (C-space) based approach where collision detection is performed in the Cartesian workspace. Fei et al. (2004) present a similar C-space obstacle boundary approach based on the reachable manifold and contact manifold theories. Freund and Rossman (2003) describe a collision avoidance in real-time environments method where the points on the robot’s surface endangered by obstacles are assigned collision avoidance points. Some authors have adopted geometrical modelling methods where the manipulator links are modelled with spherical shells, volumes formed by moving a sphere with a certain radius on a specified primitive such as a point, line, or rectangle (Afaghani and Aiyama, 2013, 2015; Bosscher and Hedman, 2009; Spencer et al., 2008). Smith et al. (2012) present a survey that includes collision detection and avoidance approaches for dual-arm robots, which are kinematically identical to work-class ROVs; they are equipped with two manipulators with overlapping workspaces. Other non-geometrical model-based approaches have also been investigated. Lee and Song (2016) propose a collision detection algorithm based on an external torque observer and friction model identification. This approach necessitates monitoring the electric current of the manipulator’s joint motors and is applicable only for electrically driven robot arms, which are rare in the ROV industry. Force feedback based collision detection methods have also been proposed (García et al., 2003). The problem of the last two approaches is that the collision can be detected solely after the contact has been made, and since ROV equipment includes multiple cameras with glass domes and other delicate devices, any contact is hugely undesirable. Lumelsky and Cheung (1991) experimented on whole-sensitive manipulator arms whose entire
body is fitted with sensitive skin capable of detecting collision with other objects. Besides the contact issue, this approach may prove to be complicated for underwater implementation as the sensor skin would have to be waterproof. Various researchers have investigated machine vision methods for collision detection (Flacco et al., 2012; Khatib, 1986; Morikawa et al., 2007); however, low visibility in ROV worksites is not uncommon, and such methods would, therefore, be condition dependent. Moreover, multiple cameras might be required to encircle the manipulators’ environment and ensure any impeding collision is detected. One of the few publications on collision detection for subsea ROV manipulators is by Agba (1995) within the “SeaMaster” ROV-manipulator system simulator. This method assumes modelling manipulator links using the super-ellipsoid equation, while the collision detection is identified by checking whether a point on the surface of an object lies within the inside surface of a link model. Another recent publication is by Rydén et al. (2013), who propose a method based on a 3D vision sensor for the rendering of spherical forbidden region and guidance virtual fixtures. This method can be used to guide the operator, who is using a haptic feedback device in advanced telemanipulation mode, towards the desired tool configuration while avoiding unintended collisions with the environment. Some authors have addressed collision detection indirectly, like Farivarnejad and Moosavian (2014), who present impedance controller design that can efficiently accomplish manipulation tasks even in the presence of unpredictable disturbances such as sudden collision with the unstructured underwater environment. Despite the significant achievements in academia, including many publications on autonomous subsea manipulation (Conti et al., 2017, 2015; Cui and Sarkar, 2000; Sarkar and Podder, 2001), collision-free manipulation has not been developed and integrated on work-class ROVs.

### 3.11 Visual control

Visual-based motion control, i.e. utilising machine vision directly in the control loop and developing fully autonomous or semi-autonomous visual servoing algorithms, is an approach that has a high potential to enhance existing sub-sea manipulation systems. Such approach have been focused on in the academic research community for more than twenty years.

One of the early references is reported by Smith et al. (1994), who describe the philosophy of utilising a 3D vision system capable of providing full target position and orientation by using laser triangulation and applying it for the open loop control of an underwater electrical manipulator. Other pioneering achievements in ROV au-
3.11 Visual control

tonomous intervention are reported within the UNION project in which coordinated control and sensing techniques have been validated in simulation conditions with the first complete vehicle-manipulator system for autonomous manipulation (Rigaud, 1995; Rigaud et al., 1998). More early work on autonomous intervention with AUVs has been done within the ODIN (Choi et al., 1994) and the OTTER (Wang et al., 1995) projects. Since than, many researchers have experimented with electrically driven subsea manipulators. Ishibashi et al. (2000) report performing experiments in test tank using a manipulator to autonomously insert a bolt into a hole, and propose a calibration method to overcome the pose estimation issues in water due to refraction. Huster and Rock (2001) suggest a method for pose estimation that merges monocular camera vision and inertial rate sensors, and present experimental results on a task to press a button using a general purpose manipulator in dry laboratory conditions. Marchand et al. (2001) propose a closed-loop visual servoing system based on an eye-to-hand approach and present experiments with a Cartesian robot arm. Lin et al. (2001) propose using stereo vision aided with ultrasonic and laser distance measurement systems in an open-loop control scheme and present experiments with a subsea electrical manipulator. Within the ALIVE project, Evans et al. (2003) propose a method for automatic AUV docking using sonar imaging to guide the vehicle close to the docking panel and using video system utilising edge and circle detection for pose estimation during the final docking phase. Some authors have investigated a measurement system for underwater manipulation applications which use ultrasonic, laser, and stereo camera systems for 3D pose estimation (Chang et al., 2004; Wu et al., 2004). Yann et al. (2005) have implemented a stereo vision based visual servoing system and tested it on an AUV in test tank for catching and retrieving a ball with a vacuum pump fitted on a 2 DOF parallel mechanism manipulator. A pose estimation method based on stereo vision system has been reported by Ishibashi (2009), along with in air experiments to validate a camera calibration method that can be performed with the manipulator, underwater. Xiao et al. (2010) have investigated using an ultra-sonic probe array and an underwater camera to estimate the pose and grasp an object lying on the test tank floor. Within the RAUVI and TRIDENT projects, visual servoing has been reported using a light-weight commercial electrical underwater manipulator on an intervention AUV (Fernandez et al., 2013; Prats et al., 2011; Ribas et al., 2012). Autonomous intervention AUV (I-AUV) docking and fixed-base manipulation for valve handling and connector mating, as well as free-floating manipulation, has been demonstrated within the TRIDENT project in the test tank and sea trials (Peñalver et al., 2015; Prats et al., 2012; Simetti et al., 2013). The SAUVIM project has demonstrated recovering a priori known objects from the seabed autonomously with a 7 DOF electrical manipulator in a free-floating
Literature review on underwater manipulators

mode (Marani et al., 2009). Kawamura et al. (2015) report experimenting on visual servoing using camera imaging to estimate the target’s pose and also to enhance the end-effector pose estimation by combining joint sensor measurements with visual feedback from detectable features placed on the manipulator. Within the MAROB and PANDORA projects, implementation of cognitive learning by demonstration for stereo visual based autonomous AUV intervention has been investigated and tested in the wet tank (Carrera et al., 2015). Khatib et al. (2016) report developing a subsea “humanoid” with two torque controlled manipulators and stereo vision to target dexterous intervention tasks commonly performed by human divers. Krupiński et al. (2017) propose a nonlinear inertial-aided IBVS control approach for an intervention AUV validated in wet tank experiments. Youakim et al. (2017) present a software framework that provides a broad range of functions applicable for intervention AUVs, including kinematics modelling and motion planning techniques enabled through various third-party integrated libraries. Conti et al. (2017) report some simulation results within the ARROWS project related to I-AUV manipulation while the vehicle moves at a relevant speed. Rizzini et al. (2017) report experimenting on stereo vision based autonomous manipulation within the MARIS project with the focus on pipe handling tasks in various light conditions and with partially observable objects. Palomer et al. (2018) propose using an underwater 3D laser scanner for underwater manipulation and have demonstrated its capabilities in underwater test tank trials.

One of the few research groups that have been working with a commercial underwater ROV manipulator (Schilling Orion 7P) is DFKI-Lab Bremen where automated plugging of a deep-sea connector and object retrieval in a wet laboratory testbed has been conducted within the CManipulator project (Hildebrandt, Kerdels, Albiez and Kirchner, 2008, 2009). The same authors, in (Christensen et al., 2009), report the development of the hardware facility and the software to control it for the simulation of realistic ROV movement in the test tank. Such a laboratory rig enables investigating visual servoing with relative motion between the target and the base vehicle.

Some academic research that included experimental subsea trials in the field environment or at least in test tanks has been done within the ALIVE (Rigaud et al., 2004), the SAUVIM (Marani and Yuh, 2014), the MARIS (Casalino et al., 2014), the TRIDENT (Sanz, Ridao, Oliver, Melchiorri, Casalino, Silvestre, Petillot and Turetta, 2010), the TRITON (Palomeras et al., 2014), the RAUVI (Sanz, Prats, Ridao, Ribas, Oliver and Ortiz, 2010) the PANDORA (Carrera et al., 2012), and the CManipulator (Spenneberg et al., 2007) projects. Only the last one addressed commercial hydraulic ROV manipulator systems, while the remaining ones focused on intervention AUVs with electrical prototype or recently commercialised manipulators. Some of the
ongoing projects that are involved in relevant underwater manipulation experiments are as follows:

- **ROBUST** ([http://eu-robust.eu/](http://eu-robust.eu/))
- **MERBOTS** ([http://www.irs.uji.es/project/merbots](http://www.irs.uji.es/project/merbots))
- **DexROV** ([http://www.dexrov.eu/](http://www.dexrov.eu/))
- **ARROWS** ([http://www.arrowsproject.eu/](http://www.arrowsproject.eu/))
- **Ocean One** ([https://cs.stanford.edu/groups/manips/ocean-one.html#media](https://cs.stanford.edu/groups/manips/ocean-one.html#media))
- **Operations Support Engineering spoke project under MaREI** ([http://www.mmmrc.ul.ie/](http://www.mmmrc.ul.ie/))

The last project is the one addressed by this thesis, and also the only one of the projects mentioned above that makes use of industry standard hydraulic ROV manipulators; other projects use electrical manipulators on AUVs.

As outlined, the majority of academic research experiments in the field of autonomous underwater manipulation have been carried out on electric robotic arms which are either prototypes or recently commercialised. Additionally, all those advanced subsea autonomous manipulation solutions found in the literature are related to intervention AUVs, which are not industry standard but rather a concept in development and are also considerably power constrained. Electric arms cannot perform all subsea operations which is why these prototype manipulators are not ready for adoption in the offshore industry. There are sound reasons why all work-class ROVs use hydraulic manipulators (depth rating, very high carrying capacity and torque, straightforward field maintenance, etc.). Despite the significant progress made in the academia over the years, the autonomous manipulation approach has not been implemented in the commercial ocean engineering sector which still employs traditional telemanipulation approaches with a human pilot in the loop for work-class ROV IRM operations.

### 3.12 Implementational issues

Investigating the application of different control techniques on commercial underwater manipulators reveals some of the implementation issues.

There are no torque controllable manipulators available commercially (Table 3.1). Therefore, many of the proposed low-level control algorithms are not applicable to commercial systems and even to most of the prototypes.
The state-of-the-art commercially available position servo controllers for underwater manipulators work on the basis of joint space setpoint regulation. For this reason, the implementation of any trajectory tracking with detailed information about the position, velocity and acceleration either in joint or Cartesian space is impossible without excessive hardware and software modifications on the low-level motion controller of the underwater manipulator. This means that any high-level kinematics control and motion planning implementation on commercial underwater manipulators has to utilise joint space PTP motion planning with a path described as a sequence of set points (Spenneberg et al., 2007). While investigating high-level control implementation on a commercial hydraulic underwater manipulator (Schilling Orion 7P), Hildebrandt, Albiez and Kirchner (2008) emphasise the importance of the type of ramp function to be used within the joint angle reference signal as part of the described path to avoid “jerky” and high strain movements. The same authors propose using Bézier-like curves computed using De Casteljau’s algorithm (Boehm and Müller, 1999) which are at least twice continuously differentiable.

Another issue is a nonlinearity occurrence that comes with the use of a linear cylinder as the underwater manipulator actuator, which can be compensated for by finding a relationship between the linear actuator stroke and the joint angle (Hildebrandt, Albiez and Kirchner, 2008; Jun et al., 2009).

The low position accuracy present in low-level motion controllers of underwater manipulators is another issue that can be of great concern. To cope with this drawback, Hildebrandt, Kerdels, Albiez and Kirchner (2009) propose a multi-layered controller, which provides increased precision without any modification of the manipulator’s hardware, as a combination of an adaptive speed control layer and a second sub-degree position control layer.

Since underwater manipulators are built for human in the loop teleoperation, the control command (set-point update) frequency on the top side is rather low, e.g. a Slingsby manipulator features a 50 Hz manipulator control update while the top side pilot command update is 5 Hz (Larkum and Broome, 1994). Analysing the communication protocols of Schilling Orion 7P manipulator, Hildebrandt, Albiez and Kirchner (2008) report on the low 12.5 Hz control frequency and emphasises the possibility to boost it up to 62.5 Hz only by modifying protocols. However, this is not the case with some electrical manipulators such as Graal Tech SRL UMA manipulator which can provide a global interface for sending commands and receiving feedback from joints at a rate of up to 200 Hz (Ribas et al., 2015).
3.13 Concluding remarks

The state-of-the-art of practical and theoretical knowledge about underwater manipulator systems has been comprehensively summarised and has given a broad overview of the existing bibliography and research results. This chapter has provided a survey of the use of manipulation technology for a variety of subsea intervention and inspection operations within different offshore areas of application. Both commercially available underwater manipulator solutions and prototype systems have been analysed. Relevant topics have been discussed, including manipulator technical specifications, mechanical design, actuation, robot modelling (kinematics and dynamics), control approaches and algorithms (motion control, kinematic control, motion planning), and a detailed comparison has been presented highlighting advantages and disadvantages of different solutions present in the underwater manipulation technology. A realistic picture of the existing technology and its limitations have been presented and have provided a useful background source for research in the field of underwater robotics and manipulation. Critical factors limiting the performance of underwater manipulators have crystallised from the comprehensive review of the state-of-the-art. The designers of future underwater manipulator systems ought to consider these factors.

Before conclusion, the major problems in the present technology are outlined, and areas for research are proposed. Some of the main issues of the existing commercial systems are low control capabilities and lack of automation. Even though the top of the range manipulator systems consist of high-quality sensors and drives, the capabilities of its control systems are insufficient. They feature low accuracy, low repeatability, and low control loop update rate. These limitations are the consequence of the design approach taken historically, which considers subsea manipulators as remote teleoperated devices rather than robot arms. Despite the significant advances achieved by the academic community over the years, the autonomous approach has not been adopted by the commercial ocean engineering sector, which still employs traditional telemanipulation approaches with the human pilot in the loop for work-class ROVs. Performing subsea operations in this fashion is well established and often preferred by ROV pilots. Since commercial work-class ROVs are equipped as standard with hydraulic manipulators, which are considerably underdeveloped in the sense of autonomy compared to stationary industrial robot arms used in factories, the author believes it is necessary to develop advanced control systems that can be employed on these robotic arms with little to no hardware modification. The approach to achieving this lies in exploiting the knowledge of the industrial robot arm (visual) servo control approaches used for typical manufacturing applications.
and adapting and transferring these techniques to challenging underwater robotics tasks. The key for underwater manipulation technology is in achieving industrial robot arm capabilities while keeping the operator in the control loop to some extent, either directly or as a supervisor. The goal is not to replace commercial underwater manipulators, which are hydro-mechanically well designed, with new industrial-class robots, but to automate as many tasks as possible with the existing equipment, and thus, decrease the task execution time and lower the burden on the operator. Additionally, replacing the existing teleoperation systems with advanced, totally autonomous robot arms would require extensive reliability testing, also, many subsea operations are too complex to automate, and human pilots are therefore likely to stay in the loop. By introducing automation to ROV systems, some processes can become totally automated or semi-automated while others would remain remotely operated. Another possibility is to develop systems which would simultaneously utilise both autonomous functions and teleoperation. i.e. the major motion would be controlled automatically while the operator would be able to apply additional motion corrections based on visual feedback due to insufficient positioning accuracy and other inherent limitations. Significant improvements can be made by developing advanced, high-level control systems and implementing them on top of the existing hydraulic underwater manipulator systems without modifying these systems hardware or software wise. The huge potential benefit of this approach is the ease of integration on existing vehicles. That way, the ROV pilot would have an option to chose between the traditional or advanced manipulation control system. However, this approach is not without limitations, and further advancements can only be made by retrofitting the control systems of the existing underwater manipulators.

Collision avoidance, or rather, controlled collision is another aspect that has not been developed and integrated on work-class ROVs, but can prove to be quite useful in manipulation technology. Implementing a system that inherently prevents the manipulator from colliding with other equipment on the ROV base vehicle and with the other manipulator, if there are two on the vehicle, would provide safer operations enabling the pilot to operate with ease. Additionally, the existence of such system would increase the confidence in implementing automated manipulation solutions.

Some specific tasks, especially the repetitive ones could benefit from teaching by demonstration methods. The operator would perform a teleoperation task which would be recorded with the possibility to be repeated in an autonomous regime. This feature would allow an operator who is not familiar with the robot programming to utilise autonomous manipulation functions.

Another potential field of study that builds on the automated manipulation is the combined vehicle-manipulator control approach. An interesting approach to
investigate is the development of systems that use vehicle navigation and positioning data for the manipulator control to compensate for the vehicle motion. In such system, the end-effector position and/or orientation would be unaffected by the motion of the vehicle. This would simplify the advanced teleoperation and semi-autonomous control, and increase the effectiveness of manipulation tasks. The ideas of combined vehicle-manipulator control approaches are not a novelty in the literature, but there are very few practical implementations, especially such that can easily be integrated on existing systems, which the author believes is crucial if it is taken up for application in the subsea industry.

While there have been many advanced subsea manipulator control approaches reported in the literature, most have not been taken through comprehensive and full design test validation and field test cycles (Fig. 3.15). To make use of many published approaches in the literature would require reimplementing the described systems and completing the full cycle of development and testing to a high technology readiness level. Such work would be prohibitively costly both financially and in time effort. In the following chapters a pragmatic approach has been taken to build on the research in the field and combine the research and development work on rugged technology that is already pervasive in the offshore operations support sectors for oil and gas, pipelines, cables, wind, etc.
Chapter 4

Design and development of advanced control systems for underwater manipulators

4.1 Introduction

Chapter 3 thoroughly reviews the current state-of-the-art in marine work-class ROV manipulation technology and highlights critical challenges in offshore inspection and intervention operations that can be resolved by improving existing manipulator control systems. This chapter proposes an advanced kinematics control engine for industry standard subsea hydraulic ROV manipulators capable of overcoming some of the related problems in IRM operations, increasing the manipulation performance in general, and providing novel solutions for the emerging MRE sector. The fundamental principles of robotics described in chapter 2 represent the backbone of the developed control strategies. The proposed solution includes a number of different control approaches, including enhanced manual, semi-automatic, and fully automatic (visual) servo control; it is also augmented with a collision detection algorithm.

This chapter is organised as follows: Section 4.2 analyses potential scenarios for implementation of advanced control approaches for underwater manipulation on work-class ROVs to facilitate typical intervention tasks. Section 4.3 describes robot modelling techniques and kinematics level control algorithms applied for underwater robotic manipulators. Section 4.4 proposes visual-based motion control strategies to enable hydraulic subsea manipulators to address stationary and moving targets in an entirely automated regime; to facilitate potential commercialisation the developed control systems are designed to be easily interchangeable with the existing control systems in the global fleet of work-class ROVs. Section 4.5 presents the collision
Design and development of advanced control systems for underwater manipulators

detection algorithm developed for underwater manipulators to facilitate teleoperation as well as advanced manipulation approaches. Lastly, section 4.6 includes concluding remarks.

4.2 Subsea manipulation task analysis

This section presents the analysis of possible scenarios for implementation of servo control approaches including vision-based control strategies for underwater robot manipulation on ROVs equipped with manipulators.

Firstly, a description of some of the typical operations performed by ROVs follows. Tjomsland and Lie (2017) provide a good overview of inspection and intervention operations in SPS, while Schjølberg et al. (2016) analyse future aspects of IRM operations in the same sector with increased robotic capability. In general, subsea intervention and inspection operations performed with ROVs are categorised into inspection, repair, and maintenance operations.

Inspection operations are the least complex of the three. These operations represent scheduled condition monitoring of offshore oil and gas production facilities (wellheads, Christmas trees, pipelines, etc.) and other submerged equipment such as moorings and anchors for detection of abnormalities and defects. Most often inspection falls under video and still photography or sonar imaging. However, other monitoring methods that do or do not include contact with targets of inspection also exist, not limited to cathodic measurements and eddy current inspection of structures and pipes. In general, inspection is performed to check for: broken or bent members, cracks, strain, stress, corrosion, pitting, accumulation and impacts of marine debris, pollution, marine growth accumulation, movement of hardware, condition, operability, etc.

Corrective maintenance operations are performed in case any irregularities are detected during the inspection operations or if a performance decrease in SPS is reported. Also, it is customary to conduct preventive maintenance which involves substituting equipment before failure. Typically, maintenance operations are limited to replacement of different units such as chokes, pumps, control modules, jumpers, flying leads, seals, cables, hoses, sensors, gauges, gaskets, etc. Additionally, maintenance tasks can include cleaning and high-pressure water jetting of subsea systems, mitigation, resolution, and removal of entanglement, and other similar activities.

Repair operations are the most complex ones and can be executed on caisson or conductor pipes, template hatches, locks, hinges, cables, etc. It can also include leak repair, pipeline integrity solution, corrosion repair, coating repair and mechanical operations.
reinforcement, trenching of cables, flexible umbilical and pipelines, as well as light construction. In general, repair operations make use of customised tools, the use of which is not that common for maintenance operations (Uyiomendo and Markeset, 2010).

Other ROV operations include support related to the installation of SPS structures. Such operation can span from the ROV acting as a remote camera to providing support during Remotely Operated Tool (ROT) missions and being directly involved in tie-in assembly operations of SPS connection systems. From the manipulation task standpoint, ROV manipulators handle valves, torque tools, override tools, measurement tools, cutting tools, clamping tools, hot stabs, and other rigid equipment. e.g. hooks, shackles, soft slings, etc.

Subsea mechanical interface standards represent the foundation for efficient IRM operations. Hegde et al. (2015) analyse the ROV standards used in the offshore oil and gas industry and highlight the gaps required to enable automated intervention in IRM. Capability gaps include, but are not limited to, highly manoeuvrable and station keeping capable vehicles, reliable command and control link to the operator, autonomous homing and docking onto stationary passive structures, autonomous manipulation, advanced vehicle navigation and localisation, advanced imaging, and increased reliability of subsea vehicle systems (McLeod, 2010). The International Electrotechnical Commission (IEC) has been formed in 2008 with 26 member countries to address the development of standards pertinent to MRE—specifically, wave tidal, and current energy conversion (Alawa et al., 2009). They recognise application resemblance of ISO standards for use in offshore oil and gas industry among which are ISO 13628-8 ROV interfaces on subsea production systems and ISO 13628-9 ROV intervention systems.

Different factors add to the complexity of manipulation tasks for subsea intervention. To gain a better insight into the problem of tasks and understand the difficulties for automation, the author proposes it is necessary to answer the following questions related to intervention operations:

- Is the target for manipulation stationary or in motion?
- If the target is stationary, is it positioned at a known location?
- If the target’s location is unknown, can it be obtained by calculations based on geometric relations?
- If the geometrical calculations are not sufficient, can the target’s location be estimated using an augmenting tool such as a vision system?
Design and development of advanced control systems for underwater manipulators

- If the target is in motion, is the motion known?
- If the motion of the target is unknown, can it be predicted?
- Is the ROV parked on the seabed or is it suspended in the water column?
- If the ROV is in the water column, is it fixed to the object for manipulation, hovering, or moving?
- If the ROV clamps onto the object for manipulation, is the grip firm with no relative motion between the base vehicle and the target?
- If the ROV hovers, is the station keeping system efficient so that the vehicle can be considered static?
- If both the ROV and the target are in motion, can the ROV motion match the motion of the target to partially or entirely reduce the relative motion between them?

Fig. 4.1 represents a tree diagram constructed from the given questions which can aid in categorising intervention operations based on execution complexity. Crucial information the answers to the questions above can provide relates to the existence of relative motion between the target and the ROV. In essence, this is determined by the combinations of static and dynamic cases for the target and the base vehicle. Two possible combinations yield no relative motion. The first one, if both the target and the ROV are stationary (static block in the figure), and the second one, if they are both in motion (dynamic block in the figure) with the ROV motion matching the motion of the target. On the other hand, three possible combinations result in no relative motion between the target and the ROV—at least the target or the ROV has to be in motion (dynamic block in the figure). However, additional questions have to be answered related to how advanced the addressed manipulator system is itself, i.e. how many DOFs there are in the robotic manipulator, whether it has joint position sensors, whether servo control is implemented, etc. With all the necessary information it is possible to completely perceive the problems that need to be solved to enable addressing subsea intervention operations with fully automatic or semi-automatic robotic control implemented on work-class ROVs.

An example of such ROV is Holland I, an Irish Marine Institute owned work-class ROV (see Fig. 4.2) equipped with two seven function Schilling Orion 7P manipulators, which is often used for scientific missions. One of the typical intervention tasks for an ROV in the marine science sector is collecting a sample from the seabed and putting it in a sample container which can be fixed on the vehicle or held by the
4.2 Subsea manipulation task analysis

Fig. 4.1 Factors that affect the complexity of ROV manipulation tasks

other manipulator. Fig. 4.3 illustrates the worksite scene of such an intervention operation where another scientific ROV owned by Ocean Exploration Trust called Hercules is performing sediment core sampling. Detailed analysis of the addressed task, regardless of its field of application, uncovers the similarity of the individual task stages to those encompassed in tasks typical for other marine sectors, including offshore oil and gas and marine renewables. The description of the task scenarios that follows provides detailed information about separate task stages, highlights their similarities in different fields of application, and suggests how to automate them. The scenarios being investigated are outlined below and start with relatively straightforward tasks, advancing to more challenging applications.

The addressed scenarios are:

Scenario 1: An ROV is equipped with two advanced seven-function manipulators both of which are assumed to have angular position sensors in each joint such as resolvers. The task stage that is addressed is autonomously placing a grasped sample in a sample container using one of the manipulators. The sample container is either fixed on the vehicle or held by the other manipulator. Whether the ROV is stationary or in motion is irrelevant for this task stage as the object is already grasped and the target is located on the base vehicle. A plausible solution for this scenario is to develop a kinematics engine using standard forward and inverse kinematics techniques, which are common in industrial robotics. Taking such an approach allows utilising a kinematics level algorithm to provide an end-effector trajectory in either Cartesian or joint space that enables the completion of the addressed task stage. Desired trajectories can be generated based on the information provided by joint position sensors, along with the known relative pose between the sample container
Design and development of advanced control systems for underwater manipulators

and the manipulator base, and the relative pose between two manipulator bases, both of which can be measured or obtained from the ROV geometric model. An identical task scenario typical for subsea IRM operations is tool handling. The task stage can be placing a tool that is held by the manipulator in the dedicated tool case located on the ROV, or the opposite, taking a previously grasped tool out of its case, provided that the location of the tool (case) is known.

Scenario 2: This scenario is identical to the first one except that the manipulator holding the sample container is assumed not to be as advanced, i.e. it does not have angular position sensors integrated into each joint, which is not uncommon for subsea manipulators. Therefore, the location of the target is unknown, and it is impossible to calculate it straightforwardly. One way to compensate for this shortfall is by adding a vision system in the control loop in the form of a camera system mounted either on the robot manipulator or the ROV. Thus, by the aid of visual servoing techniques, it is possible to develop algorithms which can generate the desired end-effector motion based on the capability of the machine vision system to locate the target. Such a solution for the described scenario is universal regarding the location of the target as long as it can successfully locate the target. Returning to the domain of subsea IRM operations, the presented solution is suitable for addressing the tool retrieving task where a tool case can be located anywhere on the ROV. Another example is the approach phase of the tool acquisition task where the end-effector needs to assume a grabbing position and where the tool does not necessarily have to be accurately placed in the tool case.
4.2 Subsea manipulation task analysis

Scenario 3: Unlike the previous scenarios which deal only with the automation of the pick and place task stages, this scenario also takes the target object acquisition/grasping into consideration. Moreover, it deals with the interaction of the seven-function arm with a static target object independent of the ROV base platform. Thus, the addressed scenario could relate to a scientific intervention operation such as the task stage of grasping a sample from the seabed. Similar complexity subsea industry tasks include using an ROV manipulator to grab and rotate a valve handle or plug a subsea connector in a socket. The base vehicle in this scenario is considered to be parked on the seabed or rigidly fixed to the object of manipulation, which is common when an ROV is operating on an oil well Christmas tree—an assembly of valves, spools, and fittings.

Scenario 4: Further complication is introduced to the previous scenario if the ROV has to hover during the task execution. An example of such scenario in an ocean science mission is grasping a sample from a subsea cliff wall, or in subsea industry, an IRM task where the ROV cannot clamp onto the object on which the manipulation is to take place. Alternatively, clamping on the object structure could be the task itself. The difference between the current and the previous scenarios is that the motion of the manipulator can affect the base vehicle station keeping.

Scenario 5: This scenario assumes that there is relative motion between the ROV base vehicle and the target for manipulation. An example of such a situation is when the ROV is loosely clamped onto the underwater structure that contains the target object for manipulation and which is in motion due to the effects of waves, tide, or subsea current. Alternatively, an ROV with insufficiently powerful and accurate station keeping system operating in hovering mode can be in motion due to the same effects. Finally, both the ROV and the target object can be in motion which is a common case of IRM operations at MRE sites with floating structures. Such scenarios require advanced control systems capable of addressing a target in motion. Perhaps a most basic task in such conditions is to perform a close visual inspection with a camera mounted on the manipulator end-effector. The target could be a mooring chain that is heaving in the water column. On the other hand, the most difficult scenarios in such conditions include intervention operation task stages that include physical contact between the manipulator and the target such as grasping a valve handle or mating a connector.
Design and development of advanced control systems for underwater manipulators

4.3 Kinematics control engine

This section describes robot modelling techniques and kinematics control algorithm design applied to underwater robotic manipulators. Firstly, mathematical methods for constructing kinematic models of subsea manipulators are described in subsection 4.3.1. This is followed by forward kinematics solutions in subsection 4.3.2, and inverse kinematics solutions specific for subsea manipulators in subsection 4.3.3. Lastly, subsection 4.3.6 provides the derivation of actuator kinematics, which is necessary for the development of a full kinematics engine for a robotic manipulator.

4.3.1 Kinematic modelling

All robotic arms addressed within the scope of this research project have six DOFs. These are two Staubli TX60 industrial robot arms and two Schilling Robotics underwater manipulators, Titan 2 and Orion 7P. Each robotic arm is mathematically modelled as an open kinematic chain of seven rigid links connected by six revolute joints using a DH convention. Fig. 4.4 shows the coordinate frames \((O_{x_0y_0z_0} \mathrm{to} \ O_{x_6y_6z_6})\) assigned to manipulators’ joints. The DH parameters for each robot model are acquired from the technical documentation and presented in Tables 4.1 to 4.3.

4.3.2 Forward kinematics

With the knowledge of robot arm DH parameters, individual coordinate transformations, each of which describes the pose of the current coordinate frame relative to the preceding coordinate frame, can be formulated as functions of joint variables in the homogeneous matrix form using the following expression (Spong et al., 2006):

\[
A_i^{i-1}(q_i) = \begin{bmatrix} 
\cos q_i & -\sin q_i \cos \alpha_i & \sin q_i \sin \alpha_i & a_i \cos q_i \\
\sin q_i & \cos q_i \cos \alpha_i & -\cos q_i \sin \alpha_i & a_i \sin q_i \\
0 & \sin \alpha_i & \cos \alpha_i & d_i \\
0 & 0 & 0 & 1 
\end{bmatrix} \tag{4.1}
\]

These homogeneous transformations can thus be used to construct the direct kinematics function, a vital robotics kinematic tool that allows determining the pose of the end-effector in the manipulator base coordinate frame; that is:

\[
T_6^0(q_1, q_2 \ldots q_6) = A_1^0(q_1)A_2^1(q_2) \cdots A_6^5(q_6) \tag{4.2}
\]
4.3 Kinematics control engine

(a) Staubli TX60
(b) Schilling Titan 2
(c) Schilling Orion 7P

Fig. 4.4 Kinematic models of robotic manipulators with Denavit-Hartenberg convention coordinate frame assignment

4.3.3 Inverse kinematics

The problem of inverse kinematics can be defined as the determination of joint variables corresponding to a given end-effector position and orientation, i.e. given a $4 \times 4$ homogeneous transformation defined by equation (2.3), find a solution, or multiple solutions if possible, of the following equation:

$$T_6^0(q_1, q_2 \ldots q_6) = H_e$$

(4.3)

First, an analytical solution is described used for the Staubli TX60 robot arm and the Schilling Orion 7P manipulator, and then a numerical solution used for the Schilling Titan 2 manipulator.

<table>
<thead>
<tr>
<th>Table 4.1 DH parameters for Staubli TX60</th>
<th>Table 4.2 DH parameters for Schilling Titan 2</th>
<th>Table 4.3 DH parameters for Schilling Orion 7P</th>
</tr>
</thead>
<tbody>
<tr>
<td>$a_i$ [m], $\alpha_i$ [°], $d_i$ [m], $q_i$ [°]</td>
<td>$a_i$ [m], $\alpha_i$ [°], $d_i$ [m], $q_i$ [°]</td>
<td>$a_i$ [m], $\alpha_i$ [°], $d_i$ [m], $q_i$ [°]</td>
</tr>
<tr>
<td>0, −90, 0.375, $q_1$</td>
<td>0.121, −90, 0.195, $q_1$</td>
<td>0.070, −90, 0.079, $q_1$</td>
</tr>
<tr>
<td>0.290, 0, 0.020, $q_2$</td>
<td>0.851, 0, 0, $q_2$</td>
<td>0.567, 0, 0, $q_2$</td>
</tr>
<tr>
<td>0, 90, 0, $q_3$</td>
<td>0.483, 0, 0, $q_3$</td>
<td>0.135, −90, 0, $q_3$</td>
</tr>
<tr>
<td>0, −90, 0.310, $q_4$</td>
<td>0.133, 90, 0, $q_4$</td>
<td>0, 90, 0, $q_4$</td>
</tr>
<tr>
<td>0, 90, 0, $q_5$</td>
<td>0, 0, 0.377, $q_5$</td>
<td>0, −90, 0, $q_5$</td>
</tr>
<tr>
<td>0, 0, 0.070, $q_6$</td>
<td>0, 0, 0.377, $q_6$</td>
<td>0, 0, 0.411, $q_6$</td>
</tr>
</tbody>
</table>
4.3.4 Analytical inverse kinematics solution

Since the Staubli TX60 and Schilling Orion 7P robotic arms have a spherical wrist—the last three joints axes \((z_3, z_4, z_5)\) intersect at a single point, see Figs. 4.4a and 4.4c—it is possible to find an analytical closed-form inverse kinematics solution. The solution in such form can be obtained through kinematic decoupling, i.e. separating the inverse kinematics problem into two simpler problems: inverse position kinematics and inverse orientation kinematics (Spong et al., 2005, p. 96). First, the inverse position is solved for the TX60 using an algebraic approach, and then for the Orion 7P using a geometric approach. After that, the inverse orientation is solved for both.

**Algebraic inverse position (Staubli TX60)**

An algebraic approach is used to solve the position component of the inverse kinematics problem for the Staubli TX60 robot arm. The complete computation is performed using Wolfram Mathematica software and is presented in Appendix A, while the main steps are outlined as follows. For more information, the reader is referred to the appendix which contains a detailed derivation. The first step is the computation of the forward kinematics equation for the position of the wrist; that is:

\[
\begin{bmatrix}
    R^0_4 & p_w \\
    0^T & 1
\end{bmatrix} = T^0_4(q_1, q_2, q_3, q_4)
\]  

(4.4)

Extracting the three equations that define the wrist position yields:

\[
x_w = -d_2 \sin q_1 + r_1 \cos q_1
\]

(4.5)

\[
y_w = d_2 \cos q_1 + r_1 \sin q_1
\]

(4.6)

\[
z_w = d_1 + d_4 \cos(q_2 + q_3) - a_2 \sin q_2
\]

(4.7)

where \(r\) is:

\[
r_1 = a_2 \cos q_2 + d_4 \sin(q_2 + q_3)
\]

(4.8)

To solve the set of these three equations for \(q_1\) it is worth squaring and summing equations (4.5) and (4.6), and solving the resulting expression for \(r_1\), which yields:

\[
r_1 = \pm \sqrt{-d_2^2 + x_w^2 + y_w^2}
\]

(4.9)

Multiplying equation (4.5) with \((d_2 \cos q_1 + r \sin q_1)\) and equation (4.6) with \((-d_2 \sin q_1 + r \cos q_1)\), subtracting the two equations, and solving the resulting expression for \(q_1\),
4.3 Kinematics control engine

yields:

\[ q_1 = \text{atan2}
\left(-d_2x_w + r_1y_w, r_1x_w + d_2y_w\right) + k\pi \quad k \in \mathbb{Z} \quad (4.10) \]

After relocating the term \( d_1 \) to the left side, equation (4.7) is squared and summed with the result of squaring and summing equations (4.5) and (4.6). The resulting equation is solved for \( \sin q_3 \), yielding:

\[ q_3 = \text{atan2}
\left(r_2, r_3 \sqrt{1 - \frac{r_2^2}{r_3^2}} \right) + k\pi \quad k \in \mathbb{Z} \quad (4.11) \]

where

\[ r_2 = -a_2^2 + (d_1 - z_w)^2 - d_2^2 - d_4^2 + x_w^2 + y_w^2 \quad (4.12) \]

and

\[ r_3 = 2a_2d_4 \quad (4.13) \]

Finally, equation (4.8) is solved for \( q_2 \); that is:

\[ q_2 = \text{atan2}
\left((r_1r_4^2 + r_5r_6)/(r_4r_7), (r_1r_5 - r_6)/r_7\right) + k\pi \quad k \in \mathbb{Z} \quad (4.14) \]

where

\[ r_6 = \sqrt{r_4^2(-r_2^2 + r_5^2) + r_7^2} \quad (4.15) \]

\[ r_7 = r_4^2 + r_5^2 \quad (4.16) \]

\[ r_5 = a_2 + d_4 \sin q_3 \quad (4.17) \]

\[ r_4 = d_4 \cos q_3 \quad (4.18) \]

**Geometric inverse position (Schilling Orion 7P)**

A geometric approach, an alternative method to solve the inverse position problem, is utilised for the Schilling Orion 7P robotic manipulator. The first step of this approach is the computation of the wrist position vector, by performing a backward translation from the end-effector Tool Centre Point (TCP) (origin of \( O_{x_6y_6z_6} \) frame) along the \( z_5 \)-axis for distance \( d_6 \), as:

\[
p_w = \begin{bmatrix} x_w \\ y_w \\ z_w \end{bmatrix} = p_e - d_6R_e \begin{bmatrix} 0 \\ 0 \\ 1 \end{bmatrix} \quad (4.19)
\]

Projecting the wrist centre \( p_w \) onto the \( x_0y_0 \)-plane (Fig. 4.5a) is helpful for identifying the method to compute the first joint angle variable. In general, there exist two
valid solutions for the first joint variable, one for the shoulder-front configuration:

\[ q_1 = \text{atan2}(y_w, x_w) \]  

(4.20)

and one for the shoulder-back configuration:

\[ q_1 = \pi + \text{atan2}(y_w, x_w) \]  

(4.21)

where \( \text{atan2}(x, y) \) represents the two argument arctangent function which, unlike the standard single argument arctangent, takes two input arguments and returns a unique angle \( q \) in the appropriate quadrant based on their signs. The \( \text{atan2} \) function is defined for all \( (x, y) \neq (0, 0) \). Two different solutions for the first joint angle affect the upcoming computation of the second and third joint angle values. The question may arise regarding whether \( (x_w, y_w) = (0, 0) \) can occur, and if it does what it means. The answer is that the robotic manipulator is at a singularity, and therefore, it is impossible to find the inverse kinematic solution as there are infinitely many solutions for \( q_0 \). In the physical sense, this means that the centre point of the spherical wrist lies on the \( z_0 \)-axis, above or below the manipulator base. By analysing the industrial robot arm Staubli TX60, it can be noticed that such a posture, in which the first joint causes the singularity, cannot be attained. This is due to the existing shoulder offset \( d_2 \), which can be recognised from Fig. 4.4a, that the wrist centre point cannot intersect the \( z_0 \)-axis, and thus cause the singularity. Effectively,
4.3 Kinematics control engine

(a) shoulder-front, elbow-up  
(b) shoulder-front, elbow-down

(c) shoulder-back, elbow-up  
(d) shoulder-back, elbow-down

Fig. 4.6 Projection of the wrist centre on the plane formed by links 2 and 3, four configurations for a given wrist position

this offset creates a volume in the shape of a cylinder of radius $d_2$ around the $z_0$-axis that is unattainable for the end-effector/wrist centre point. This feature, in the first instance may seem insignificant, but can make a huge difference, as is outlined above. The reason it exists in industrial robotic arms is that those devices are designed with consideration on potential issues that may occur in the mathematical modelling process, including solving inverse kinematics. On the other hand, subsea manipulators are not designed for autonomous robotic tasks, and therefore, such aspects are not taken into consideration.

Finding solutions to compute $q_2$ and $q_3$ is not as straightforward as is the case with $q_1$, and requires a more in-depth trigonometry analysis of the manipulator kinematic chain. The plane to analyse is the one formed by the second and third links, illustrated in Figs. 4.6a to 4.6d. Four different figures depict four possible manipulator configurations for a given wrist position. These configurations are not always attainable due to mechanical constraints and joint range limits. For the elbow-up configuration, depicted in the Figs. 4.6a and 4.6c, the solutions for the
Design and development of advanced control systems for underwater manipulators

second and third joint angle variables are expressed as:

\[ q_2 = -\epsilon - \alpha \] (4.22)

\[ q_3 = \pi - \gamma - \delta \] (4.23)

while the alternative solutions corresponding to the elbow-down configuration, depicted in the Figs. 4.6b and 4.6d, these are expressed as:

\[ q_2 = -\epsilon + \alpha \] (4.24)

\[ q_3 = -\pi + \gamma - \delta \] (4.25)

Computing the joint angle values requires determining the values of four auxiliary angles: \( \epsilon, \alpha, \gamma, \) and \( \delta \). The angle \( \delta \) is constant and can be calculated as:

\[ \delta = \text{atan2}(d_4, a_3) \] (4.26)

The angle \( \epsilon \) is straightforward to calculate; that is:

\[ \epsilon = \text{atan2}(z_w - d_1, P - a_1) \] (4.27)

for the shoulder-front configuration, and as:

\[ \epsilon = \text{atan2}(z_w - d_1, P + a_1) \] (4.28)

for the shoulder-up configuration, where \( P \) is calculated as:

\[ P = \sqrt{x_w^2 + y_w^2} \] (4.29)

Obtaining the values of angles \( \alpha \) and \( \gamma \) is not that simple. It requires the aid of the Heron’s formula, applied on the triangle formed by the angles \( \alpha, \beta, \) and \( \gamma \), which states that the following expression gives the area of a triangle:

\[ A = \sqrt{s(s - a)(s - b)(s - c)} \] (4.30)

where \( a, b, \) and \( c \) are the sides, and \( s \) the semiperimeter of the triangle; that is:

\[ s = \frac{a + b + c}{2} \] (4.31)
The area of the same triangle can be calculated as:

\[ A = \frac{1}{2}ra + \frac{1}{2}rb + \frac{1}{2}rc = rs \]  \hspace{1cm} (4.32)

where \( r \) is the radius of the inscribed triangle. Combining equations (4.30) and (4.32) yields:

\[ r = \sqrt{\frac{(s-a)(s-b)(s-c)}{s}} \]  \hspace{1cm} (4.33)

From Fig. 4.6, the triangle side \( c \) can be calculated as:

\[ c = \sqrt{(z_w - d_1)^2 + (P - a_1)^2} \]  \hspace{1cm} (4.34)

for the shoulder-front configuration, and as:

\[ c = \sqrt{(z_w - d_1)^2 + (P + a_1)^2} \]  \hspace{1cm} (4.35)

for the shoulder-back configuration, while the sides \( a \) and \( b \) are constant; that is:

\[ a = \sqrt{a_3^2 + d_4^2} \]  \hspace{1cm} (4.36)

\[ b \equiv a_2 \]  \hspace{1cm} (4.37)

Finally, it is possible to calculate the angles \( \alpha \) and \( \gamma \) (see Fig. 4.7):

\[ \alpha = 2 \text{atan2}(r, s-a) \]  \hspace{1cm} (4.38)

\[ \gamma = 2 \text{atan2}(r, s-c) \]  \hspace{1cm} (4.39)

Having determined \( \epsilon, \alpha, \gamma, \) and \( \delta \), allows the computation of \( q_2 \) and \( q_3 \) from equations
Design and development of advanced control systems for underwater manipulators

(4.22) and (4.23) for the elbow-up configuration, and equations (4.24) and (4.25) for the elbow-down configuration.

Inverse orientation (Staubli TX60 and Schilling Orion 7P)

The next step is the inverse orientation part of the algorithm. As the values for the first three joint angles are determined, it is possible to obtain $R_0^3$, by extracting it from the homogeneous transformation matrix that represents the pose of the frame $O_{x_3y_3z_3}$ relative to the base frame, which is calculated by the following expression:

$$T_0^3(q_1, q_2, q_3) = A_1^0A_2^1A_3^2$$  \(4.40\)

The rotation matrix that describes the end-effector orientation relative to the frame $O_{x_3y_3z_3}$ is computed by:

$$R_0^3 = R_0^3T R_e$$  \(4.41\)

The next step is finding joint variables $q_4$, $q_5$, and $q_6$ corresponding to the rotation matrix $R_6^3$. The forward kinematics equation to express the end-effector pose in the $O_{x_3y_3z_3}$ reference frame is given by:

$$T_6^3(q_4, q_5, q_6) = A_4^3A_5^4A_6^5$$  \(4.42\)

This equation enables the extraction of the rotation matrix $R_6^3$ in analytical form. As the joint angles $q_4$, $q_5$, $q_6$ constitute the ZYZ set of Euler angles with respect to the frame $O_{x_3y_3z_3}$ (Siciliano et al., 2009, p. 76), given the:

$$R_6^3 = \begin{bmatrix} n_x & s_x & a_x \\ n_y & s_y & a_y \\ n_z & s_z & a_z \end{bmatrix}$$  \(4.43\)

it is possible to compute:

$$q_4 = \text{atan2}(a_y, a_x)$$
$$q_5 = \text{atan2}(\sqrt{(a_x)^2 + (a_y)^2}, a_z)$$  \(4.44\)
$$q_6 = \text{atan2}(s_z, -n_z)$$
for $q_5 \in (0, \pi)$, and:

\[
q_4 = \text{atan2}(-a_y, -a_z) \\
q_5 = \text{atan2}(-\sqrt{(a_x)^2 + (a_y)^2}, a_z) \\
q_6 = \text{atan2}(-s_z, n_z)
\] (4.45)

for $q_5 \in (-\pi, 0)$. For the TX60 manipulator, relevant elements from equation (4.42) are as follows:

\[
a_x = r_{13} \cos q_1 \cos (q_2 + q_3) + r_{23} \sin q_1 \cos (q_2 + q_3) - r_{33} \sin (q_2 + q_3) \\
a_y = r_{23} \cos q_1 - r_{13} \sin q_1 \\
a_z = r_{33} \cos (q_2 + q_3) + \sin (q_2 + q_3) (r_{13} \cos q_1 + r_{23} \sin q_1) \\
s_z = r_{32} \cos (q_2 + q_3) + \sin (q_2 + q_3) (r_{12} \cos q_1 + r_{22} \sin q_1) \\
n_z = r_{31} \cos (q_2 + q_3) + \sin (q_2 + q_3) (r_{11} \cos q_1 + r_{21} \sin q_1)
\] (4.46-4.55)

For the Orion 7P manipulator, relevant elements from equation (4.42) are as follows:

\[
a_x = r_{13} \cos q_1 \cos (q_2 + q_3) + r_{23} \sin q_1 \cos (q_2 + q_3) - r_{33} \sin (q_2 + q_3) \\
a_y = -r_{23} \cos q_1 + r_{13} \sin q_1 \\
a_z = -r_{33} \cos (q_2 + q_3) - \sin (q_2 + q_3) (r_{13} \cos q_1 + r_{23} \sin q_1) \\
s_z = -r_{32} \cos (q_2 + q_3) - \sin (q_2 + q_3) (r_{12} \cos q_1 + r_{22} \sin q_1) \\
n_z = -r_{31} \cos (q_2 + q_3) - \sin (q_2 + q_3) (r_{11} \cos q_1 + r_{21} \sin q_1)
\] (4.51-4.55)

This encapsulates the whole inverse kinematics solution for the two manipulators that have a spherical wrist. Complete derivations of inverse orientation using Wolfram Mathematica software are given in Appendix B for Staubli TX60 and Appendix C for Schilling Orion 7P.

### 4.3.5 Numerical inverse kinematics solution

The Schilling Titan 2 manipulator does not have a spherical wrist (Fig. 4.4b) and therefore finding a closed-form analytical solution is impossible. This may be another evidence that subsea manipulators, unlike the industrial robot arms, are designed without considerations on features that can simplify mathematical modelling and
Design and development of advanced control systems for underwater manipulators

control. For that reason, inverse kinematics problem for the Titan 2 manipulator is solved using numerical methods. The algorithm employed is the closed-loop second-order inverse kinematics algorithm with pseudo-inverse Jacobian (Siciliano et al., 2009, p. 133)

The well-known differential kinematics equation which represents the relationship between the joint space velocities and the task space velocities is given by equation (2.5). Derivation of this equation yields the second order differential kinematics equation:

\[ \dot{v}_e = J(q)\ddot{q} + \dot{J}(q, \dot{q})\dot{q} \]  (4.56)

Rearranging the terms and inverting this equation yields a solution in terms of joint accelerations, given by:

\[ \ddot{q} = J^{-1}(q)(\dot{v}_e - \dot{J}(q, \dot{q})\dot{q}) \]  (4.57)

with the assumption that the Jacobian matrix \( J \) is non-singular and square. By integrating this equation over time, joint velocities \( \dot{q} \) and positions \( q \) can be re-constructed. However, open loop solutions through numerical integration methods suffer from solution drift and thus lead to operational space errors. This shortfall can be overcome by using a closed-loop inverse kinematics algorithm. Utilising this algorithm requires defining a six-dimensional operational space error vector as the difference between the desired and actual end-effector position and orientation:

\[ e(t) = \begin{bmatrix} e_p(t) \\ e_o(t) \end{bmatrix} = x_d(t) - x_e(t), \]  (4.58)

its time derivative which is effectively the difference between the desired and actual end-effector velocity:

\[ \dot{e}(t) = \dot{x}_d(t) - \dot{x}_e(t), \]  (4.59)

and its second derivative:

\[ \ddot{e}(t) = \ddot{x}_d(t) - \ddot{x}_e(t) \]  (4.60)

The position error \( e_p(t) \) from equation (4.58) is calculated straightforwardly as the difference between the desired and the actual end-effector position:

\[ e_p(t) = p_d(t) - p_e(t) \]  (4.61)
As end-effector orientation is represented with unit quaternions, the orientation error $e_o(t)$ from equation (4.58) is calculated by the expression:

$$e_o(t) = \eta_o(q)\epsilon_d - \eta_d(q)\epsilon_e(q) - S(\epsilon_d)\epsilon_e(q)$$

(4.62)

where $Q = \{\eta_d, \epsilon_d\}$ and $Q = \{\eta_e, \epsilon_e\}$ are assumed to denote, respectively quaternions associated with desired and actual end-effector orientation, and $S(\cdot)$ is a skew-symmetric operator:

$$h = \begin{bmatrix} 0 & -h_3 & h_2 \\ h_3 & 0 & -h_1 \\ -h_2 & h_1 & 0 \end{bmatrix}$$

(4.63)

The actual end-effector velocity from equation (4.59) is obtained by:

$$v_e(t) = J(q)\dot{q}$$

(4.64)

where $J$ represents the geometric Jacobian. Using (4.56) in (4.60) yields

$$\ddot{e} = \dot{v}_d - J(q)\dot{q} - J(q, \dot{q})\dot{q}$$

(4.65)

By choosing the joint acceleration vector as:

$$\ddot{q} = J^\dagger(q)(\dot{v}_d + K_D\dot{e} + K_Pe - J(q, \dot{q})\dot{q})$$

(4.66)

we get the equivalent linear error system:

$$\ddot{e} + K_D\dot{e} + K_Pe = 0$$

(4.67)

which under the assumption that $K_P$ and $K_D$ are positive definite matrices is asymptotically stable, i.e. the error tends to zero along the trajectory; $J^\dagger$ is the pseudo-inverse of the geometric Jacobian $J$, and it is used when the manipulator is redundant for a given task. This solution minimises the norm of joint accelerations. Integrating equation (4.66) over time leads to the reconstruction of joint velocities $\dot{q}(t)$ and positions $q(t)$ assuming the initial conditions $\dot{q}(0)$ and positions $q(0)$ are known. The integration is done in discrete time using the Euler integration method. Namely, knowing the joint positions, velocities and accelerations at time $t_k$, joint velocities and positions at time $t_{k+1}$ can be computed as:

$$\dot{q}(t_{k+1}) = \dot{q}(t_k) + \dot{q}(t_k)\Delta t$$

(4.68)
Design and development of advanced control systems for underwater manipulators

\[ q(t_{k+1}) = q(t_k) + \dot{q}(t_k) \Delta t \] (4.69)

where \( \Delta t \) is a given integration interval.

The described numerical closed loop inverse kinematics algorithm has been developed and implemented for both the Staubli TX60 and the Schilling Titan 2 robotic arms.

4.3.6 Actuator kinematics

Utilising the forward kinematics function is meaningful only if the joint angle values are available. However, measurements acquired by joint position sensors do not necessarily represent the angular joint position data in the direct form. For example, rotary actuators drive two joints \((q_4 \text{ and } q_6)\) of the Schilling Orion manipulator, and the rotary potentiometers placed in the joints acquire position measurements for these joints. Establishing the relationship between these joint sensor readings and joint angle values is straightforward using linear interpolation:

\[ q = q_{\text{min}} + \frac{(q_{\text{max}} - q_{\text{min}})}{(s_{\text{max}} - s_{\text{min}})}(s - s_{\text{min}}) = q_{\text{min}} + \frac{\Delta q}{\Delta s}(s - s_{\text{min}}) \] (4.70)

where the variable \( s \), and constants \( s_{\text{min}} \) and \( s_{\text{max}} \) represent the rotary potentiometer sensor current, and lower and upper bound values, while \( q_{\text{min}} \) and \( q_{\text{max}} \) are the lower and upper bound values corresponding to the \( s_{\text{min}} \) and \( s_{\text{max}} \) values. It is important not to be misled by the actual values when establishing the correlation defined by equation (4.70) as the sensor measurements and the joint angle values can be either directly or inversely proportional, depending on the orientation of the coordinate frame related to that joint. Except for the shoulder joint, every joint of the Schilling Titan 2 manipulator is equipped with a resolver—a rotary position sensor—and driven by a rotary hydraulic actuator. When it comes to the Staubli TX60 robot arm, electric motors drive the joints, while absolute encoders acquire angular position measurements. Therefore, the link between the sensor readings and the joint angle values for these joints can be established using equation (4.70).

As rotary actuators act directly on the joints, the rotation angle of the actuator shaft is equivalent to the rotation angle of the joint driven by that actuator.

On the other hand, if a linear actuator, such as a hydraulic cylinder drives a revolute joint, the position sensor placed in the joint measures the rotational angle of that joint rather than the cylinder’s piston rod extension. Therefore, it is necessary to identify the mapping between the position sensor readings, extension of linear actuators, and the joint angle values. Knowledge of this relationship allows employing inverse kinematics function on the actuator level, i.e. it is possible to compute the position
variables for all joint actuators, rotary or linear, given the desired end-effector pose. Rotary potentiometers acquire the position measurements for the cylinder driven joints ($q_1$, $q_2$, $q_3$, and $q_5$) of the Orion 7P manipulator. Similarly, a rotary resolver measures the position of the cylinder driven shoulder joint ($q_2$) of the Schilling Titan 2 manipulator. A linear relationship established between the rotary sensor readings and the joint angle is given by equation (4.70). The next step is identifying the non-linear mapping between the extraction lengths of prismatic piston rods and the angular values of revolute joints. For the Schilling Orion 7P manipulator, this relationship exists for the joints $q_1$, $q_2$, $q_3$, and $q_5$, and can be identified from Figs. 4.8 - 4.11. From these figures, it can be seen that the joint angles can be obtained as:

\[
q_1 = \gamma + \frac{\pi}{2} - \beta - \alpha \tag{4.71}
\]

\[
q_2 = \gamma + \frac{\pi}{2} - \beta - \alpha \tag{4.72}
\]

\[
q_3 = -\gamma + \frac{\pi}{2} + \beta - \alpha \tag{4.73}
\]

\[
q_5 = \gamma - \frac{\pi}{2} - \beta + \alpha \tag{4.74}
\]

The angle $\beta$ is constant, determined by the mechanical design of the actuator. The angle $\gamma$ is not constant; however, a single value of this angle that corresponds to a specific posture of the manipulator, often the one given in the technical manual of the robot, can be calculated and used as a constant benchmark value for the conversion. The angle $\alpha$ is a variable directly linked with the piston rod extension $a$, and the relationship between these two variables can be obtained from the cosine theorem:

\[
a^2 = b^2 + c^2 - 2bc \cos \alpha \tag{4.75}
\]

where the values of $b$ and $c$ are constant, known from the actuator design, see Figs. 4.8 - 4.11. Fig. 4.12 is used to obtain the relationship between the angle value of the Schilling Titan 2 shoulder joint ($q_2$) and the extraction of the cylinder driving that joint; that is:

\[
q_2 = \gamma + \frac{\pi}{2} - \beta - \alpha \tag{4.76}
\]

Complete computation of the described kinematic correlations for the Schilling Orion 7P and Titan 2 manipulators using MATLAB is given in Appendix D. The described kinematic correlation between sensor values / joint angles and lengths of the linear actuator moving elements is based on a comparison between the current manipulator posture and the benchmark manipulator posture. It is essential to keep in mind that the benchmark posture that is used to determine the angle $\gamma$ may not
Design and development of advanced control systems for underwater manipulators

Fig. 4.8 Geometrical relationship between the linear actuator and the revolute joint ($q_1$) of the Schilling Orion 7P

Fig. 4.9 Geometrical relationship between the linear actuator and the revolute joint ($q_2$) of the Schilling Orion 7P
Fig. 4.10 Geometrical relationship between the linear actuator and the revolute joint \((\theta_3)\) of the Schilling Orion 7P

Fig. 4.11 Geometrical relationship between the linear actuator and the revolute joint \((\theta_5)\) of the Schilling Orion 7P
be the robot posture in which the values of joint angles are zero according to the DH convention manipulator model. The joint angle values $q$ in equations 4.71 to 4.76 are given relative to the joint angle values $\hat{q}$ that define the benchmark posture. Therefore, it is necessary to identify the relative joint offsets between the benchmark posture and the zero joint angle posture, and include them in the robot model by adding each to its corresponding joint variable. For example, if converting the given cylinder piston rod extension to a corresponding joint angle, and the resulting angle is zero, the actual joint angle is equal to the joint offset that defines the benchmark posture. If the resulting angle has a non zero value, the actual joint angle is the sum of the resulting joint angle and the joint offset with the appropriate sign. For the single Schilling Titan 2 manipulator joint that is driven by a cylinder, the joint angle offset does not exist as the manipulator is given in the posture in which the joint angle value is zero (see Fig. 4.12). The same is true for the first joint of Schilling Orion 7P manipulator (see Fig. 4.8). However, the case with identifying the offset values for the remaining Orion 7P cylinder driven joints is not that straightforward and requires a more in-depth trigonometry analysis. The method used to identify the joint offset values for the Orion 7P robotic manipulator is illustrated in Fig. 4.13. This step by step approach includes angular transformations required to reconfigure the manipulator from the DH convention model posture that corresponds to the null joint angle values (black contour in the left image) to the benchmark posture (blue contour in the right image), which is given in the manipulator’s technical manual. The angle $\alpha$ is constant and is calculated as:

$$
\alpha = \text{atan2}(a_3, d_4)
$$

(4.77)

while the angles $\beta$ and $\gamma$ are easily computed from the triangle formed by the axes of the joints $q_2$, $q_3$, and $q_5$, see Fig. 4.14. The computation of these angles is given in Appendix E. Finally, the angular joint offsets are formed and summed with the joint
4.4 Visual-based motion control

Advanced vision system development plays a vital role in the enhancement of robotic manipulation in the undersea environment. Driven by the intention to deal with various problems occurring underwater with camera vision systems, such as reduced and limited visibility, high turbidity, low contrast, blurring, and others, the development of a novel vision system based on the fusion of camera and sonar imaging is proposed by colleague researchers Rossi et al. (2015). Design of vision-based motion control laws for robotic manipulators has to be compatible with such future trends in vision system development. Additionally, the manipulator control software has to be readily interchangeable with existing teleoperation control systems for hydraulic ROV manipulators. Therefore, the algorithm design has to cope with
Design and development of advanced control systems for underwater manipulators

an imposed constraint that the solution has to be implementable on the existing off-the-shelf subsea manipulator systems without any modifications of the systems’ hardware or software. The targeted underwater manipulator systems have low-level joint position servo motion control integrated into the Slave Control Unit (SCU). Therefore, a purely kinematic visual servoing scheme is proposed, treating the robotic manipulator as an ideal positional device. It provides kinematic parameters as output signals which are forwarded as input reference signals to the existing manipulator SCU. Implementation of various solutions reported in the literature was considered in the scope of this research including PBVS, IBVS, and hybrid methods, all of which have been covered in section 2.4. Analysis of these techniques resulted in a conclusion that the most convenient approach is choosing and implementing PBVS type algorithms. One of the reasons is that forward-looking sonars are capable of providing raw position information; 2D sonars provide the position of points in the polar coordinate system specified by a radial distance and a polar angle, and 3D sonars in the spherical coordinate system with one additional polar angle. Therefore, integrating vision systems based on sonar and camera imaging fusion may be more straightforward if using PBVS compared to other methods, as it would involve using measurements in a similar form. Additionally, the benefit of applying PBVS algorithms, especially in the early development stage, is the possibility to separate the motion planning and control problem from the imaging and pose estimation problem. Such an approach enables developing the two systems independently of each other and leaves room for modularity—a PBVS motion control algorithm can utilise any pose estimation (vision) software provided it outputs the position information in a suitable form. Most importantly, the PBVS approach is suitable for integration with industry standard low-level actuator controllers which are inseparable from the majority of commercial subsea robotic manipulators. Additionally, eye-in-hand camera configuration is adopted mainly because such a solution gives increased measurement resolution the closer the imaging system is to the target and this might be essential in low visibility subsea conditions. However, as PBVS is adopted, additional vision systems such as cameras mounted on the ROV base vehicle can be easily integrated as aiding tools, e.g. if the target for manipulation is not in the field of view of the manipulator’s wrist-mounted camera.

Subsection 4.4.1 below describes the developed PBVS algorithm that can enable subsea ROV manipulators to address stationary targets completely automatically and provides discussion. Subsequently, improvements to this algorithm by including motion prediction methods to enable addressing targets in motion are given in subsection 4.4.2.
4.4 Visual-based motion control

4.4.1 PBVS algorithm — stationary targets

The scheme of the developed PBVS algorithm with dynamic look and move structure is shown in Fig. 4.15. This algorithm can be separated in two main components, which are pose estimation and motion control.

Pose Estimation

The pose estimation part of the PBVS algorithm is in charge of processing images acquired by a vision system in real time and providing the estimation of target object’s position and orientation. As this research is focused on the development of robot control software rather than machine vision algorithms, to simplify the target object detection and pose estimation, it was decided to use fiducial markers (Fig. 4.16), which are a well-established pose estimation tool (Lepetit et al., 2005). Using a calibrated camera (Zhang, 2000) and having the information of the exact geometry of the fiducial marker, it is possible to estimate its pose relative to the camera by employing a planar homography based algorithm (Agarwal et al., 2005). The accuracy of the
pose estimation strongly depends on the adequacy of the camera calibration and the replication reliability of the fiducial marker model. This pose estimation algorithm continuously takes images captured by the camera and removes distortion from them using the intrinsic parameters acquired from the camera calibration process. Subsequently, it attempts to detect the fiducial marker on the undistorted images. Finally, if the detection is successful, the algorithm provides the estimated pose of the fiducial marker in the camera reference frame ($H^C_M$) as an output (Fig. 4.17) based on the extrinsic parameters, which are also acquired from the camera calibration process (Garrido-Jurado et al., 2014). Utilising such a pose estimation approach assumes that the target object for manipulation is equipped with a fiducial marker. Alternatively, if the target object is of known geometry, other advanced markerless methods capable of estimating the position and orientation of the target are applicable. An advanced real-time 3D reconstruction method that performs camera tracking without relying on specific features is currently under development (Rossi et al., 2015). A dense surface model of workspace objects obtained with this method can be used to identify a target (either manually or automatically, e.g. from a CAD model) and compute its pose relative to the camera without the requirement for a fiducial marker.

**Dynamic Trajectory Planning**

Fixing a fiducial marker rigidly in the vicinity of the target, so that the relative pose between the target and the marker ($H^M_T$) is known, enables to straightforwardly determine the pose of the target in the camera reference frame ($H^C_T$), by the following
expression:

\[ H^C_T = H^C_M \cdot H^M_T \]  

(4.81)

Additionally, assuming that the relative pose between the TCP and camera (\( H^C_{TCP} \)) frames is known, it is possible to calculate the relative pose between the target and the TCP (\( H^T_{TCP} \)) from the expression:

\[ H^C_{TCP} \cdot H^T_{TCP} = H^C_T \]  

(4.82)

as:

\[ H^T_{TCP} = (H^C_{TCP})^{-1} \cdot H^C_T = (H^C_{TCP})^{-1} \cdot H^C_M \cdot H^M_T \]  

(4.83)

Defining \( H^T_{TCP} \) as an operational Cartesian space control variable and utilising it to control the motion of the manipulator might seem to be an intuitive approach as it is evident that the convergence of this homogeneous transformation to the identity matrix leads to the TCP reaching the target. The homogeneous transformation \( H^T_{TCP} \) is obtained indirectly through the target to marker (\( H^C_{M} \)), TCP to camera (\( H^C_{TCP} \)) and camera to end-effector (\( H^C_{EE} \)) homogeneous transformations. Therefore, any errors introduced in \( H^T_{TCP} \) by the inaccuracy of any of these homogeneous transformations may significantly reduce the effectiveness of the visual servoing algorithm. Having geometrically reliable CAD models of the target object equipped with a fiducial marker, camera, gripper, mounts and other relevant parts might provide sufficiently accurate homogeneous transformations. If these are unavailable, referring to additional measurement techniques might be necessary. The homogeneous transformations that are difficult to determine accurately are \( H^C_{EE} \) and \( H^C_{TCP} \); both are defined relative to the coordinate frame in the camera sensor placed inside the camera housing, and its location is often, if not always, unavailable from the technical documentation. A possible solution to overcome this shortfall is to resort to so-called eye-hand calibration techniques for estimating the homogeneous transformation \( H^C_{EE} \) such as Tsai’s method (Tsai and Lenz, 1989), which include repositioning a robot arm to different configurations and taking images of an object of known geometry from these poses. Having determined the homogeneous transformation \( H^C_{EE} \) it is possible to calculate the relative pose between the TCP and camera frames as:

\[ H^C_{TCP} = (H^C_{EE})^{-1} \cdot H^C_{EE} \]  

(4.84)

provided that a reliable gripper CAD model is available, which often is. An alternative simple but effective solution exists that can compensate for the potential imperfections caused by inexact knowledge of the relevant homogeneous transforma-
Design and development of advanced control systems for underwater manipulators

That is a “teach by showing” method that includes manually positioning the robotic manipulator to attain the desired posture, with the TCP positioned accurately for the grasp operation, and recording the value of the homogeneous transformation obtained as the output of the pose estimation algorithm. Thus, this value can be assigned to the homogeneous transformation referred to as the desired marker pose relative to the camera reference frame ($H_{desM}$). The operational space control variable can then be defined as the “difference” between the actual and the desired marker pose relative to the camera frame, given in matrix form by:

$$H_{error} = H_{M}^{C} \cdot (H_{desM}^{C})^{-1}$$  \hspace{1cm} (4.85)$$

Controlling this error variable so that it converges to the identity matrix leads to the TCP reaching the target. Having chosen the described approach, the next step to design an appropriate controller is to determine the desired end-effector pose relative to the base frame ($H_{desEE2}$) that corresponds to the homogeneous transformation $H_{desC}^{C}$. It might be compelling at this point to record this value as well by using the described “teach by showing” method. However, this would work just for the case where the target (fiducial marker) is stationary, fixed on the ROV, and its pose relative to the robot base frame is therefore unchanging. In a more general case, this homogeneous transformation is unknown and variable as the target can be anywhere in the workspace of the manipulator. On the other hand, the value of the homogeneous matrix $H_{desC}^{C}$ is constant and independent of the location of the target in the workspace of the robot if it is allowed to assume that there is only one way to grasp an object correctly. To find the corresponding homogeneous transformation $H_{desEE2}$, it is necessary to determine the relative motion ($H_{\Delta}$) the end-effector is required to perform so that the actual pose of the marker relative to the camera frame becomes identical to the desired pose (Fig. 4.18). Another way to describe $H_{\Delta}$ is the relative pose between the end-effector in the desired pose and the end-effector in the initial pose expressed in the reference frame of the end-effector in the initial pose. This relationship can be determined from the observed expression:

$$H_{C}^{EE2} \cdot H_{M}^{C} = H_{\Delta} \cdot H_{C}^{EE2} \cdot H_{desC}^{C}$$  \hspace{1cm} (4.86)$$

where $H_{C}^{EE2}$ is constant and known from the CAD model, $H_{M}^{C}$ is acquired from the pose estimation algorithm, and $H_{desC}^{C}$ is the desired value described earlier. Using simple matrix operations on this expression leads to:

$$H_{\Delta} = H_{C}^{EE2} \cdot H_{M}^{C} \cdot (H_{desM}^{C})^{-1} \cdot (H_{C}^{EE2})^{-1}$$  \hspace{1cm} (4.87)$$

104
Fig. 4.18 Relevant homogeneous transformations to estimate the desired end-effector motion

where $H \Delta$ is in a homogeneous transformation matrix form. Finally, the desired end-effector pose in the base frame can be calculated by:

$$H_{des}^{B_2} = H_{EE_2}^{B_2} \cdot H \Delta$$  \hspace{1cm} (4.88)

where $H_{EE_2}^{B_2}$ is the pose of the end-effector in the robot base frame calculated using forward kinematics with the values of the joint positions corresponding to the moment of capturing a camera image for the pose estimation algorithm. This value represents the reference for the motion control in Cartesian space as it would be traditionally defined. At this point, the motion control part of the visual servoing algorithm is described.

It is important to emphasise that the motion control component of the PBVS algorithm deals only with kinematics, i.e. generates reference motion parameters that are to be forwarded to the existing commercial low-level joint space positioning motion controller. To find the joint variables corresponding to the desired pose of end-effector in the robot base frame ($H_{des}^{B_2}$), it is necessary to solve the inverse kinematics problem. Developed inverse kinematics solutions applicable to robotic manipulators addressed in this research have been covered earlier in section 4.3.3. As the Schilling Titan 2 manipulator does not have a spherical wrist, a closed form analytical solution does not exist, and therefore, a numerical inverse kinematics method is used. That is, the closed-loop second-order inverse kinematics algorithm with pseudo-inverse Jacobian where the value of the desired end-effector pose relative to the robot base frame ($H_{des}^{B_2}$) is assigned to $x_d(t)$, see equation (4.58). On the other hand, analytical closed-form solutions are developed for the Schilling Orion 7P manipulator which have also been covered earlier in this chapter. Kinematic
Design and development of advanced control systems for underwater manipulators

modelling was essential for the algorithm development and it included the previously derived forward and inverse kinematics solutions which have already been covered in section 4.3.1 and published by the author, see Appendix G.

With the assumption that the camera is well calibrated and that the errors introduced with other modelling imperfections are negligible, an open-loop control scheme often referred in the literature as the look then move method in which the pose is estimated just once (Corke, 1996) are applicable. In such case, the inverse kinematics numerical algorithm can be designed to run in as many iterative loops as required until the solution converges to the predefined error threshold. Apart from neglecting the possibility of errors introduced in the modelling which lead to inaccuracy, the disadvantage of this method is that the time required for the execution of the sole inverse kinematics algorithm is variable and unknown in advance and could take too long. This method might prove to be sufficiently accurate for a stationary target, and only assuming that the modelling is close to perfect, but it is most likely to fail while addressing targets in motion due to the variable relative pose between the camera and the target. Therefore, closing the loop is necessary, due to inevitable modelling errors and due to the plan to address targets in motion, which is discussed later in section 4.4.2. Such requirements lead towards adopting the dynamic look and move visual servoing scheme (Sanderson and Weiss, 1980) for the PBVS algorithm. Closing the loop according to the described scheme yields the PBVS algorithm that consists of two loops, the outer loop which is in charge of pose estimation and reference setting, and the inner loop which is in charge of inverse kinematics.

Discussion of the PBVS algorithm

This subsection presents a discussion on relevant parameters that affect the efficiency of the proposed PBVS controller. As highlighted above, the developed PBVS algorithm utilises a numerical integration method in discrete time to obtain inverse kinematics solution. The value of the integration interval affects the efficiency of the inverse kinematics algorithm and subsequently the PBVS algorithm. One way to improve the algorithm performance is to have an integration interval that is variable and adaptively modified. An alternative method that renders the same result is to preconfigure the integration interval to a certain constant value and design a discrete inverse kinematics algorithm with a variable number of iterations that is adaptively modified by a Cartesian controller based on some strategy. The greater the number of iterations is the more accurate the output of the inverse kinematics is as it gradually converges to the given desired reference value. However, processing
4.4 Visual-based motion control

Each inverse kinematics loop requires a certain amount of time, and therefore, the number of iterations determines the frequency of the external control loop. In other words, it determines how often the target pose is re-estimated. If addressing a stationary target, this is not too relevant, and this parameter defining the number of iterations does not have an upper boundary. Therefore, it can be set to any value that yields the shortest total time required for the task to be executed successfully. However, as soon as the target in motion comes into play, this parameter gets an upper boundary. The reason is that increasing the number of iterations over a certain value inevitably leads to the execution of the inverse kinematics algorithm taking too much time resulting in the object leaving the field of view of the camera before the acquisition of the following image for the pose estimation. This occurrence causes the opening of visual feedback because of the lack of visual measurements and eventually to servoing failure. Reducing the number of iterations may prove to be a valid means to keep the object in the field of view of the camera during the task execution. However, an insufficient number of iterations may increase the total number of external loop iterations due to a greater mismatch between the inverse kinematics output and the given desired value and consequently increase the total task execution time. Therefore, led by the goal to minimise the total task execution time while keeping stability in mind, it is essential to find an appropriate value for the parameter determining the number of iterations of the inverse kinematics algorithm. Moreover, an algorithm capable of adaptively changing the number of inverse kinematics iterations based on some law and relevant inputs (distance to the target, target velocity, etc.) may also prove to be useful.

One method to reduce the risk of the target object leaving the camera field of view has been explained above. As the PBVS algorithm acts directly on operational space variables, with appropriate path planning algorithm the trajectory of the camera can be controlled directly in Cartesian space and this problem may be prevented. A potential method to realise such an algorithm is by further alterations on the \( H_{\Delta} \) homogeneous transformation, which represents the estimated relative motion that the end-effector needs to perform so that the gripper reaches the target. By adaptively modifying this value it is possible to control how the gripper approaches the target, rather than just making it reach it. Therefore, instead of having the homogeneous transformation matrix \( H_{\Delta} \) as a fixed value equal to the value obtained through the pose estimation result, it is possible to modify it to the desired needs. A simple method that can reduce the risk of the target leaving the field of view of the camera is to form a parameter that represents the percentage of the path along the straight line in Cartesian space between the initial and the desired pose (Fig. 4.19). Using this parameter, the value of the desired relative pose \( H_{\Delta} \) and interpolation techniques
Design and development of advanced control systems for underwater manipulators

(a) “One shot”  (b) Fixed  (c) Variable

Fig. 4.19 Concept of target approaching with variable percentage of path

it is possible to find the corresponding relative pose. Position coordinates can be computed using standard linear interpolation, and the orientation parameters by the spherical linear interpolation method (Dam et al., 1998) for Quaternions. By adopting this method, the inverse kinematics algorithm now has as a desired value an intermediate point in Cartesian space between the initial and the final pose. Utilising such an approach eventually leads to the manipulator reaching the target, but moving in steps, covering a certain amount of the path towards the target rather than at once. The value of this parameter determines the size of the steps taken in the target approach phase of the PBVS algorithm. Consequently, the approach velocity can be controlled using this method. The described approach may come with the price of an increase in the total time required for the task execution time; nonetheless, it is useful as it plays an essential role in preventing the target object leaving the field of view of the camera. Another benefit is that it introduces a safety factor. If the target is in motion, there unquestionably exists a possibility of collision. If the target motion is such that it moves towards the manipulator end-effector, the larger the intermediate steps in the visual servoing loop are, the bigger the chance is that a collision may occur. On the other hand, if the steps are small enough, approaching the target object problem can be solved with proper motion planning and the manipulation task executed without collision. Although the proposed method cannot guarantee that the target object stays in the field of view of the camera throughout the manipulation, the experiments with the stationary target have shown that this method does reduce the risk of this happening and consequently of the collision as well, as will be shown in the following chapter. However, addressing target objects with significant motion might entail the need to implement robust path planning methods for visual servoing, some of which are available in the literature (Baumann et al., 2010; Chesi et al., 2004; Kazemi et al., 2009; Thuilot et al., 2002).

The two described parameters, the number of iterations of the inverse kinematics algorithm and the parameter determining the percentage of the path between the current and the desired end-effector pose, affect the efficiency and the speed of the task execution. It is important to emphasise that these two parameters are mutually dependent. Therefore, maximising the performance of the visual servoing algorithm requires addressing their effect in combination rather than independently.
4.4 Visual-based motion control

Additionally, other relevant information should be taken into consideration when addressing the described parameters. One of them is the velocity of the target in motion which can be estimated, analysed and taken into account. Depending on the target velocity, the relevant parameters can be adaptively modified. If the target is slow, a more significant portion of $H\Delta$ in combination with a greater number of iterations should be suitable. Contrary to this, if the target is fast, smaller path steps should be taken in combination with less inverse kinematics iterations. Another critical factor is the distance between the camera and the target. The closer the target object is to the camera, the larger it is in the camera image, and it can, therefore, more quickly leave the image plane, especially if it is moving relatively fast. Therefore, the closer the gripper is to the target, the number of iterations should be reduced and the portion of the motion increased, to reach the target promptly in this terminal phase.

4.4.2 Enhanced PBVS algorithm — moving targets

This subsection describes additional modifications and improvements of the developed visual-based motion control algorithm, presented in subsection 4.4.1, to enable addressing targets in motion with an underwater manipulator. The ultimate goal is to develop a control solution capable of driving a subsea manipulator to approach, follow, and perform an intervention operation on a moving target.

Defining the error variable as the difference between the estimated ($H^C_M$) and the desired ($H_{des}^C_M$) pose of the target (fiducial marker) relative to the camera frame is the most straightforward approach for the PBVS algorithm. To accomplish the end-effector target following task successfully the motion of the manipulator has to be controlled so that the error variable converges and stays equal to the identity matrix. Doing this requires deriving the desired manipulator pose $H_{des}^B_E$ that corresponds to the desired target pose $H_{des}^C_M$ and controlling the manipulator to attain that pose; the described process is already covered earlier in this chapter with the emphasis on addressing stationary targets, see equations (4.85) to (4.88). The issue with this approach is that by the time the manipulator moves from the initial to the desired pose, the target will have already moved to a different location, see Fig. 4.20. To compensate for the target motion, an additional variable $H_{mov}$ is introduced in the proposed PBVS algorithm which is referred to as the motion prediction variable. It represents the relative pose between the location at which the target was at the time the camera captured the image for the pose estimation and the location at which the target will be when the manipulator reaches the desired pose. Methods utilised to estimate the target motion with sufficient accuracy and compensate for it represent
Design and development of advanced control systems for underwater manipulators

the core of this section, and their further description follows. Including the described motion prediction in the control scheme design (Fig. 4.21), the control error variable in the form of a homogeneous transformation matrix is defined by the following expression:

\[
H_{\text{error}} = H^C_M \cdot (H_{\text{des}}^C_M)^{-1} \cdot H_{\text{mov}}
\] (4.89)

Finding the manipulator pose \(H_{\text{des}}^B_E\) that corresponds to the desired target to camera pose \((H_{\text{des}}^C_M)\) is done by determining the relative motion \((H\Delta)\) the end-effector has to perform to attain the desired pose, which is given in equation (4.87). With the motion prediction variable included, this relationship is given with the following expression:

\[
H\Delta = H^E_C \cdot H^C_M \cdot H_{\text{mov}} \cdot (H_{\text{des}}^C_M)^{-1} \cdot (H^E_C)^{-1}
\] (4.90)

where the homogenous transformation \(H^E_C\) represents the pose of the camera with respect to the end-effector frame, which is constant and can be approximated from the CAD model. Finally, the desired end-effector pose in the manipulator base frame is calculated by equation (4.88) and the steps that follow are the same as in the PBVS algorithm described earlier in this chapter.

**Motion prediction based on numerical integration**

The first of two methods proposed for the estimation of the motion prediction variable \((H_{\text{mov}})\), defined in equation (4.89), is presented. The described approach is based on the Euler’s numerical integration method. Its derivation is a result of the analysis of the problem of tracking objects located on a conveyor for pick and place robots (Park and Lee, 1992). The velocity of the target object moving on a conveyor is given in

![Fig. 4.20 Relevant homogeneous transformations for target pose estimation and motion prediction](image)
the discrete time form by the following expression:

\[ \dot{x}[n] = \frac{x[n] - x[n-1]}{\Delta t} \] (4.91)

where \( n \) is the discrete time index, \( x[n] \) and \( x[n-1] \) are target position measurements, and \( \Delta t \) is the sampling time period. Since the velocity of the objects moving on a conveyor is close to constant, it is assumed that the change of position between two equal consecutive time intervals is the same. Hence, the target position in the first consecutive future point in time is computed via numerical integration as:

\[ x[n+1] = x[n] + \dot{x}[n] \cdot \Delta t \] (4.92)

Extracting the second term from equation (4.92) and substituting equation (4.91) into it yields:

\[ \Delta x = \dot{x}[n] \cdot \Delta t = x[n] - x[n-1] \] (4.93)

which represents the estimated displacement based on the assumption that the velocity is unchanged, and can be associated with the motion prediction variable. Constructing this term while keeping consistent with the adopted notation is described as follows. Firstly, the difference between the target position vector at time \( n - 1 \) and the same vector at time \( n \), with reference to the manipulator base frame, is
Design and development of advanced control systems for underwater manipulators

determined as:


Both position vectors are extracted from the corresponding homogeneous transformation calculated as:

\[ H_M^B[k] = H_E^B[k] \cdot H_C^E \cdot H_M^C[k] \quad \text{with} \quad k \in \{n, n - 1\} \]  (4.95)

The estimated displacement vector defined by equation (4.94) is then expressed with reference to the target frame at time \( n - 1 \) as follows:

\[ \Delta H_{M_{n-1}}^M = (H_M^B[n - 1])^{-1} \cdot \text{Trans}(\Delta p_M^B) \cdot H_M^B[n - 1] \]  (4.96)

\[ \Delta p_{M_{n-1}}^M = \text{Trans}(\Delta H_{M_{n-1}}^M) \]  (4.97)

where \( \text{Trans}() \) operator represents the operation of converting a position vector into a corresponding translation matrix in the homogeneous form, or extracting the position vector from a given homogeneous transformation matrix. Finally, the value to be assigned to the motion prediction variable is the translation matrix calculated as:

\[ H_{\text{mov}} = \text{Trans}(\Delta p_{M_{n-1}}^M) \]  (4.98)

It is essential to indicate that the described motion estimation approach takes only the change in position into consideration. The orientation of the target is assumed to be unchanged between times \( n \) and \( n + 1 \). This can be seen from equation (4.88) where the current (at time \( n \)) estimated camera pose \( H_M^C \) is post-multiplied by the motion prediction variable \( H_{\text{mov}} \), which is, in this case, a translation operation. The described motion estimation method might prove to be sufficient if the target object is either moving with a constant velocity or if it is occasionally accelerating with relatively low acceleration and for relatively short periods of time. However, if the acceleration is significant or if it lasts for a substantial amount of time, the motion estimation error is likely to be high. A similar method that may prove to be useful in dealing with the issues discussed above is derived which is also based on numerical integration but with the assumption that the acceleration is relatively constant. This leads to computing the target position in the future consecutive point in time as:

\[ x[n + 1] = x[n] + \dot{x}[n] \cdot \Delta t + \ddot{x}[n] \cdot \Delta t^2 \]  (4.99)
The acceleration at time $n + 1$ is assumed to be equal to the acceleration at time $n$ and is calculated as:

$$\ddot{x}[n] = \frac{\dot{x}[n] - \dot{x}[n - 1]}{\Delta t}$$ (4.100)

Extracting the last two terms from equation (4.99), substituting the acceleration term with equation (4.100), and all velocity terms with appropriately indexed equation (4.91), yields the following expression, representing the estimated displacement:

$$\Delta x = \dot{x}[n] \cdot \Delta t + \ddot{x}[n] \cdot \Delta t^2 = 2x[n] - 3x[n - 1] + x[n - 2]$$ (4.101)

As can be noticed, this approach includes three previous position measurements. Similarly to equation (4.94), the estimated target displacement vector expressed in the manipulator base frame is given by:


Repeating the previously described process that spans between equations (4.94) and (4.98) leads to acquiring the motion prediction variable $H_{mov}$. This approach is likely to be efficient if the target is accelerating with a constant acceleration as well as if the target is moving with a constant velocity. In case the acceleration is highly variable, which is undoubtedly the case with a sinusoidal wave motion change of position, velocity, and acceleration, it might be compelling to include one or more previous position measurements into the designed motion estimation method. However, this is not investigated within the scope of this research and is left for future work.

**Motion prediction based on ANFIS**

A novel visual based motion control algorithm is developed for subsea manipulation to enable addressing a moving target. This algorithm is characterised by the implementation of ANFIS, described in section 2.5, for the motion prediction part of the previously developed PBVS algorithm.

Various authors have reported utilising fuzzy heuristics for nonlinear motion prediction of moving targets resulting in performance improvement of the motion tracking system (de Costa Sousa and Setnes, 1999; Kaawaase and Chi, 2011; Rajpurohit and Pai, 2011). On the other hand, ANFIS proves to be exceptional when employed for chaotic dynamics prediction (Jang, 1993). Therefore, comparable performance was anticipated if ANFIS was to be utilised for the prediction of repetitive motion such as MRE device motion response subject to sea waves. Hence, the decision was made to investigate its application within the visual servoing algorithm.
ANFIS is employed for the task of using a time series of the position vector $p_B^M$ coordinate values up to time $n$ to predict the coordinate values of the same position vector at time $n + P$ in the future. This is done by creating a dataset of $D$ known position vector values of the time series, with the space of $\Delta$ between every two successive values, given by:

$$(p_B^M[n − (D − 1)\Delta], \ldots, p_B^M[n − \Delta], p_B^M[n])$$

(4.103)

and mapping this dataset to predicted future coordinate values of the position vector $p_B^M[n + P]$. Each position vector is extracted from the homogeneous transformation matrix acquired as the result of the computation of the appropriately indexed equation (4.95). Since ANFIS is traditionally a single output system, each of the coordinates $(x_M, y_M, z_M)$ of the position vector is addressed with a separate, but initially identical ANFIS framework. For the purpose of ANFIS network training, a preparation stage is introduced in the visual servoing algorithm. This stage includes teleoperating the camera-equipped manipulator to attain such pose that the target is in the field of view of the camera during the target motion cycles, which is followed by continuous estimation of target’s position utilising pose estimation software described previously in this chapter. Once this is finished, a dataset is formed, and ANFIS network training starts. It is common to use a certain fraction of recorded data for training, and a remaining fraction of data for validation by comparing the predicted values with the actual ones. Constructing a reliable and efficient ANFIS network for a given task depends on adopting appropriate values for different process variables involved, such as $D$, $P$, $\Delta$, the size of the input-output dataset, the number of membership functions per input, the number of training epochs, etc. Alternatively, extensive trial and error experiments with the aim to investigate the effect of these parameters can be performed. In case such an approach is impossible, a method for selecting an optimal number of dataset pairs can be used (Buragohain and Mahanta, 2008). Once the ANFIS network is trained it can estimate the target position vector $p_B^M[n + P]$ based on a specified number of previous measurements. Constructing the estimated target displacement vector in the following form:

$$\Delta p_B^M = p_B^M[n + 1] − p_B^M[n]$$

(4.104)

is beneficial as it allows continuing from equation (4.96) up to equation (4.98) in order to construct the motion prediction variable in a form ready to be incorporated with the rest of the algorithm.
4.5 Collision detection

This section describes the collision detection algorithm developed for underwater manipulators beyond the current state-of-the-art in work-class ROV technology. Fig. 4.22 shows a block diagram of the dual manipulator ROV control system architecture which includes the developed collision avoidance algorithm. The yellow blocks represent the software developed within this thesis and the white blocks represent the existing systems. The developed software has to be readily integrable with existing subsea hydraulic manipulator systems without any hardware or software modifications. Therefore, the proposed collision detection algorithm runs on a standard PC, located between the manipulator Master Controller Unit (MCU) and the low-level joint position servo controller. Additionally, a computer control software for subsea manipulators is proposed as an alternative to the traditional MCU, which is described in section 5.2.3. This software includes a program switch that allows the ROV pilot to select which of the two systems is used to control the manipulator, and to switch the control from one to the other during operation.

The proposed collision-free manipulation algorithm is an on-line method based on a voxel map—a representation of Cartesian space discretised into a regular grid. The work of Czarnecki (1994) inspired the core idea of the algorithm. The proposed solution is based on a purely kinematic method that processes kinematic parameters—joint positions—provided by a command control system and returns collision-free kinematic parameters in the same form, which become the input commands for the existing low-level joint space motion controller of the manipulator. Kinematic

---

Fig. 4.22 Block diagram of a dual manipulator control system encapsulating the proposed collision detection algorithm
Design and development of advanced control systems for underwater manipulators

modelling is essential for the algorithm development; it has been covered already in section 4.3. The algorithm is based on the discretisation of Cartesian space occupied by manipulators and obstacles into a regular grid of cubic voxels with the desired spatial resolution, determined by the voxel size. This process is referred to as voxeling in the literature. The smaller the voxels are, the more accurate the modelling is. The maximum total number of voxels \( n \) in the grid is inversely proportional to the voxel size \( s \) and grows according to the cubic law in (4.105).

\[
    n = \frac{1}{s^3} \quad \text{for} \quad s \in (0, 1]
\]  

(4.105)

A higher voxel grid resolution for a given volume leads to a higher computational load for the collision avoidance algorithm, which used to be a relevant factor in the past, but with the processing power currently available it is not anymore, as will be seen from the simulation and experimental results in section 5.4. Voxel maps are formed by highlighting those voxels that are occupied by worksite objects. Voxel maps that represent static obstacles only need to be computed once, during the initial stage of the algorithm. On the other hand, voxel maps that represent manipulators and other dynamic obstacles have to be recomputed in each control loop iteration. It is assumed that the geometry of worksite objects is known, i.e. that the CAD models of obstacles and manipulators are available. Each static obstacle’s pose, as well as manipulator poses, are also assumed to be known. The common reference frame, in this case, is a fixed coordinate frame on the ROV base vehicle. Collision detection is performed by checking whether more than one object occupies the same voxel at a given point in time. Subsections 4.5.1 and 4.5.2 describe the processes to construct voxel maps for static and dynamic obstacles, respectively. Subsection 4.5.3 describes a process to optimise workspace voxel maps, and lastly, subsection 4.5.4 describes the algorithm for sensing and prohibiting impeding collisions.

### 4.5.1 Voxel map modelling — static obstacles

This subsection describes a method to create a voxel map that represents static obstacles. For each workspace object, a separate voxel map can be generated based on its CAD model. An alternative way is to build multiple object CAD assemblies and create a voxel map for each one or form a single assembly that contains all static obstacles and transform it into a voxel map. For each mission an ROV can be equipped with a slightly different set of devices; additionally, the same devices can be installed in different locations. Having an independent voxel map for each device, or groups of devices that are often used together, it is straightforward to construct a
4.5 Collision detection

Fig. 4.23 Meshing an ROV camera CAD model into a point cloud using SolidWorks

single final voxel map for each mission as a union of separate voxel maps. Therefore,
for the sake of modularity, it is preferable to address each object separately when
constructing voxel maps.

The first step is transforming an obstacle’s CAD model into a point cloud, which
can be achieved using most 3D CAD tools. Using SolidWorks for example, it is
possible to generate a mesh of points with a user-specified mesh density for each
surface of the CAD model (Fig. 4.23). Selecting an appropriate mesh density is
crucial for avoiding gaps in voxel maps—the Euclidean distance between any two
points in the mesh should be at least an order of magnitude smaller than the voxel
size. This method does not consider the object’s volume, but only its external surface.
Therefore, the resulting point cloud is a shell of the shape of the object. The method
used to compensate for the loss of information about the interior of the obstacle is
performed on the voxel map level and is described later in this section. The resulting
point cloud is a set of \( m \) data points \( P \), defined with Cartesian coordinates:

\[
P = \{ P_i(x_i, y_i, z_i) \in \mathbb{R}^3 | i = 1, \ldots, m \}\]  

(4.106)

Choosing the point cloud reference frame is essential—it has to be either the ROV
base frame \( O_{xyz} \) or any frame \( ^hO_{xyz} \) whose pose \( H_h \) relative to the base is known or
possible to measure. In the latter case, expressing the point cloud in the ROV base
frame is straightforward:

\[
P = H_h^h P
\]  

(4.107)

The next step is mapping the generated point cloud to voxels in the Cartesian grid.
Voxels occupied by at least one point are highlighted and included into the voxel
map. This is done by finding the closest voxel to each point, i.e. the voxel with the
smallest Euclidean distance from the point. A regular voxel grid \( V \) (Fig. 4.24) is
given by:

\[
V = \{ V_i(x_i, y_i, z_i) \in \mathbb{R}^3 | i = 1, \ldots, n \}\]  

(4.108)
where each voxel is defined with Cartesian coordinates representing its volumetric centroid. Mapping each point in the point cloud $P$ to its nearest voxel is performed by a rounding function, given by:

$$
\hat{P}_i = \left\lfloor \left( \frac{P_i + \frac{s}{2}}{s} \right) \right\rfloor s \quad \text{for} \quad i = 1, \ldots, m
$$

(4.109)

As a result, each point in the newly formed point cloud $\hat{P}$ gets pushed to its nearest voxel. The coordinates of these points are used to highlight voxels in the grid and form a voxel map that represents the object’s shell. The next step is to compensate for the lost information on the object’s interior by filling the cavities. This is done by identifying all internal voxels and marking them as occupied. For each grid axis, the algorithm seeks for unoccupied voxels and checks that these voxels are located between occupied voxels. For a detailed implementation, see Algorithm 1. The result is a voxel map that models the entire volume of the object. The same process is repeated for all other objects/obstacles. Finally, a single voxel map representing all static worksite objects is created as a union of separate obstacle voxel maps. Since this voxel map is constant, it has to be constructed only once before the execution of the collision avoidance algorithm.

### 4.5.2 Voxel map modelling — manipulators

The method for creating a voxel map that represents a manipulator is slightly different due to its moving parts. The manipulator consists of a base and links. Since the base is fixed, it is modelled as a static object and incorporated in the static obstacle voxel
Algorithm 1 Algorithm for supplementing an object shell voxel map with internal volume voxels

1: procedure FillInternalVolume(V)
2: for all z-axis do
3: cnt ← 0
4: for all x-axis do
5: for all y-axis do
6: if V(x, y, z) = 1 then
7: cnt ← cnt + 1
8: if cnt = 1 then
9: V1 ← V(x, y, z)
10: y1 ← y
11: else if cnt = 2 then
12: if y - y1 = Vsize then
13: V1 ← V(x, y, z)
14: y1 ← y
15: else
16: V2 ← V1
17: V1 ← V(x, y, z)
18: cnt ← cnt + 1
19: for y ← y1, y2 do
20: V(x, y, z) ← 1
21: end for
22: end if
23: end if
24: else if cnt ≥ 2 then
25: cnt ← 0
26: end if
27: end if
28: end for
29: end for
30: cnt ← 0
31: for all y-axis do
32: for all x-axis do
33: if V(x, y, z) = 1 then
34: cnt ← cnt + 1
35: if cnt = 1 then
36: V1 ← V(x, y, z)
37: y1 ← y
38: else if cnt = 2 then
39: if x - x1 = Vsize then
40: V1 ← V(x, y, z)
41: x1 ← y
42: cnt ← cnt - 1
43: else
44: V2 ← V1
45: V1 ← V(x, y, z)
46: cnt ← cnt + 1
47: for x ← x1, x2 do
48: V(x, y, z) ← 1
49: end for
50: end if
51: end if
52: else if cnt ≥ 2 then
53: cnt ← 0
54: end if
55: end for
56: end for
57: end for
58: return V
59: end procedure

map. The situation with links is different as they move in space. Determining the volume in space occupied by each link requires a kinematic model of the manipulator, a CAD model of each link, and the manipulator’s latest angular joint positions acquired in each control loop. This volume can thus be modelled as a point cloud and transformed into a voxel map. A more efficient way is to precompute voxel maps for each manipulator links from their CAD models, and then iteratively, using the kinematic model and angular joint positions, remap the voxel maps appropriately.

The robotic manipulator kinematic model is derived according to the DH convention for attaching reference frames to the links of a manipulator and is given in section 4.3.1. It is assumed that the manipulator consists of six joints numbered from 1 to 6, and seven links numbered from 0 to 6, starting from the base. Each link has a coordinate frame rigidly attached to it; the DH convention determines its location. The first step in creating a manipulator voxel map is to create a voxel map for each link by using the technique for a separate static object described in section 4.5.1. However, in the step of transforming a CAD model into a point cloud, it is essential to choose the appropriate reference coordinate frame for each link. These frames have to be the coordinate frames rigidly attached to the links according to the DH
Design and development of advanced control systems for underwater manipulators

convention (Figs. 4.4). Thus, the resulting voxel map $^kL$ for the $k$th link $(k = 1, \ldots, 6)$ expressed in the coordinate frame $^kO_{xyz}$, is given by:

$$^kL = \{^kL_i(x_i, y_i, z_i) \in \mathbb{R}^3 | i = 1, \ldots, k_n \}$$  \hspace{1cm} (4.110)

where $^k_n$ is the number of voxels that describe the $k$th link. The pose of the coordinate frame $^kO_{xyz}$ can be expressed in the manipulator base frame $^0O_{xyz}$, as a homogeneous transformation calculated with the standard forward kinematics equation using the appropriate joint position values $\mathbf{q}$, given by (Siciliano et al., 2009):

$$H_0^k = \prod_{i=1}^{k} T_i^{-1}(q_i)$$  \hspace{1cm} (4.111)

This resulting homogeneous transformation can be expressed in the ROV base frame:

$$H_k = H_0^k H_k^0$$  \hspace{1cm} (4.112)

where $H_0$ is the pose of the manipulator base in the ROV frame. Finally, a voxel map for the $k$th link expressed in the ROV base coordinate frame $O_{xyz}$ is acquired by multiplying each voxel from the voxel map given in (4.110) with the homogeneous transformation given in (4.112):

$$^0L_{ki} = H_k^i L_{ki} \hspace{.3cm} \text{for} \hspace{.3cm} i = 1, \ldots, k_n$$  \hspace{1cm} (4.113)

The same procedure is repeated for each of the manipulator’s links, excluding the base. Subsequently, a single manipulator voxel map is created as the union of the individual links’ voxel maps. The voxel map derived in this way determines which voxels the manipulator occupies for any given joint configuration.

Obstacles independent of the ROV base vehicle are not addressed at this stage. Examples of such obstacles related to ROV operations are the sea floor, a dock wall, or any offshore infrastructure an ROV is not supposed to collide with while operating in its vicinity. Regardless of whether they are stationary or moving, these can be considered dynamic since the ROV is in motion; unless ideal station keeping is assumed. Computer vision might be a potential solution to identifying these obstacles and generating corresponding point clouds. Having the point cloud, transferring it into a voxel map is a straightforward process. Alternatively, if the obstacle structure is known, a method similar to that used for manipulator links can be applied. In this case, the missing component, to be potentially implemented leveraging computer vision, would be the estimation of the obstacle’s pose relative to the ROV. Software
solution developed to date in this work is applicable only for detecting collisions between the moving manipulators and the static workspace obstacles.

4.5.3 Voxel map modelling — manipulators’ workspaces

The static obstacle voxel map described in section 4.5.1 is formed based on the CAD models of all objects with which manipulators are not supposed to collide, without considering the manipulators’ workspace size, and therefore regardless of whether the manipulators can reach the potential obstacles. If static obstacles are out of the manipulators’ reachable workspace, there is no reason to include them in the voxel map. However, as the two manipulators’ workspaces do not overlap entirely, some obstacles are reachable by both manipulators and some only by one of them. Instead of using a single voxel map comprising of all obstacles, two separate voxel maps are used, one for each manipulator, each of which contains only the objects that are reachable by that manipulator. This section describes the procedure to generate these two voxel maps.

The Cartesian space defining a manipulator’s reachable workspace has to be transformed into a voxel map. The first step in creating a manipulator workspace voxel map is discretising the manipulator’s configuration space (C-space) and transforming it into the Cartesian space. The C-space represents the set of all allowable transformations of the manipulator; for a 6 DOF manipulator it forms a 6-dimensional manifold (Kavraki and LaValle, 2016). For each DOF of the manipulator \(k = 1, \ldots, 6\), a number of intervals along the generalised joint coordinate \(q_k\) is specified as:

\[
N_k = \left\lfloor \frac{q_k^{\text{max}} - q_k^{\text{min}}}{\Delta q_k} \right\rfloor \quad (4.114)
\]

where \(q_k^{\text{min}}\) and \(q_k^{\text{max}}\) are the physical limits of the \(k\)th joint motion, and \(\Delta q_k\) is the discretisation resolution of the \(k\)th joint. The most straightforward discretisation method is uniform discretisation — fixing \(\Delta q_k\) to a constant value used for all joints. The deficiency of this approach is that the Cartesian points it generates are not equidistant. This can be improved using advanced discretisation methods such as heuristic and optimal discretisation (Henrich et al., 1998). However, creating a workspace voxel map does not require having a dense point cloud of equidistant reachable end-effector Cartesian points throughout the entire workspace volume. Nevertheless, accurate and sufficiently dense modelling of the outer shell of the working space is required. As the C-space discretisation takes place only once, the computational time for this step is not of importance, and therefore the uniform discretisation method with high enough resolution is sufficient. Discretisation of the
Design and development of advanced control systems for underwater manipulators

C-space generates $K$ joint configuration vectors $q$, where $K = \prod_{k=1}^{6} N_k$, which are transformed into the Cartesian space using standard forward kinematics equation (Siciliano et al., 2009), resulting in a point cloud expressed in the manipulator’s base frame:

$$^0W = \left\{ ^0W_j = \prod_{i=1}^{6} T_i^{i-1}(q_i) \mid j = 1, \ldots, K \right\}$$  (4.115)

Using the same process that was used for generating the static obstacles map, this point cloud is transformed into a voxel map and expressed in the ROV base frame ($W$), see Equations (4.106-4.109). Since the resulting manipulator workspace voxel map might contain gaps within the volume, the missing voxels are added using Algorithm 1. Finally, the resulting voxel map represents the entire volume of the manipulator’s workspace. The intersection between this voxel map and the static obstacle voxel map then results in a map $O_1$ which is comprised only of obstacles reachable by that manipulator. The same procedure is repeated for the other manipulator, yielding voxel map $O_2$. These newly formed static obstacle voxel maps are smaller in size which is convenient for storage and computation.

4.5.4 The collision avoidance algorithm

Regardless of the manipulator control input device, mode of operation (manual, semi-automatic or fully automatic), and operational space (joint or Cartesian), the output kinematic parameters to be issued to the low-level manipulator motion controller are assumed to be angular joint positions. Additionally, it is considered that the desired joint position is a set of a specified number of joint positions that start from the initial joint position and incrementally go the desired joint position. The algorithm requires having access to desired motion commands and current joint position sensor measurements for both manipulators in each control loop. After issuing the manipulator motion commands, the desired joint position vectors are processed by the collision avoidance algorithm based on the procedure described in this section.

The first step in implementing the algorithm is forming a path between initial and desired joint configurations. Since the difference between the corresponding values of desired and initial joint positions is assumed to be relatively small, the number of steps forming this path does not have to be large. This provides a sequence of specific manipulator poses in space-time and for each of these poses a manipulator voxel map is created, using the technique described in sections 4.5.1, 4.5.2, and 4.5.3. The resulting voxel maps are merged into a single map, as a union between them. The newly formed voxel map ($M_{D1}$) represents the entire volume that the
4.6 Concluding remarks

The development work of this research on advanced robot control solutions beyond the current state-of-the-art in work-class ROV technology has been described in this chapter. The presented control strategies have been designed to enable enhanced manual, semi-automatic, and fully automatic (visual) manipulation on ROVs equipped with manipulators, and consequently facilitate IRM operations in various marine applications including, but not limited to, offshore oil and gas and MRE. Robot modelling and kinematics control techniques have been adopted from the industrial robotics sector, modified, and adapted for use on hydraulic underwater manipulators, enabling full-scale automation in marine intervention operations with ROVs. Visual-based servo control algorithms have been proposed to enable carrying out IRM operations automatically without a human teleoperator in the loop. These vision-guided control solutions have been designed to address both stationary targets and targets in motion, neither of which have been realised until now with the existing industry standard hydraulic subsea manipulators on the global fleet of ROVs. Lastly,
Design and development of advanced control systems for underwater manipulators

A collision detection algorithm has been developed for underwater manipulator systems as a safety factor that can enhance undersea intervention. The following chapter presents simulations and experimental results in laboratory and underwater in field trials to evaluate the designed advanced kinematics control engine for underwater ROV manipulator systems.
Chapter 5

Simulation and experimental test results

5.1 Introduction

An advanced kinematics control engine for underwater manipulator ROV systems is proposed in chapter 4. The system is comprised of enhanced manual, semi-automatic, and fully automatic (visual) servo control functions reinforced with collision detection. This chapter presents the development and highlights the features of the proposed control system for industry standard hydraulic manipulators capable of facilitating subsea inspection and intervention operations. Additionally, this chapter evaluates the performance of advanced control solutions developed for commercial work-class marine ROV systems through simulation, laboratory experiments and subsea field trials, and presents the results.

This chapter is organised as follows: Section 5.2 presents the design and development of the manipulation control software. The same section also provides simulation and experimental test results to validate the robot modelling and kinematic control techniques. The addressed experiments have been performed with industrial robot arms and underwater manipulators in dry laboratory conditions as well as subsea in field trials. Section 5.3 presents the implementation of the proposed PBVS algorithm in the pilot control software. It also provides results of the evaluation of the developed visual servoing control system for stationary targets in simulation and experiments with industrial robot arms and underwater manipulators, both in dry laboratory and subsea environments. Additionally, this section presents a motion analysis study of an MRE device followed by simulations and experiments using an industrial robot arm to replicate its motion. This section also describes experimental tests using industrial robot arms and subsea manipulators to perform
Simulation and experimental test results

Table 5.1 Organisation of sections in chapter 5 related to the specific simulation and experimental test results.

<table>
<thead>
<tr>
<th>Section</th>
<th>Kinematic control</th>
<th>PBVS</th>
<th>ANFIS PBVS</th>
<th>Collision detection</th>
</tr>
</thead>
<tbody>
<tr>
<td>Simulation tests</td>
<td>5.2.1</td>
<td>5.3.2</td>
<td>/</td>
<td>5.4.1</td>
</tr>
<tr>
<td>Industrial robot arm dry tests</td>
<td>5.2.2</td>
<td>5.3.3</td>
<td>5.3.7</td>
<td>/</td>
</tr>
<tr>
<td>Subsea manipulator dry tests</td>
<td>/</td>
<td>/</td>
<td>5.3.8</td>
<td>5.4.2</td>
</tr>
<tr>
<td>Subsea manipulator wet tests</td>
<td>5.2.4</td>
<td>5.3.4</td>
<td>/</td>
<td>/</td>
</tr>
</tbody>
</table>

This chapter presents simulation and experimental test results related to the specific kinematic control, PBVS, ANFIS PBVS, and collision detection algorithms. The chapter is divided into sections as follows: 5.2 Kinematic modelling and control, 5.3.2 Simulation tests, 5.3.3/5.3.7 Industrial robot arm dry tests, 5.3.8 Subsea manipulator dry tests, 5.4.1 Collision detection, 5.4.2 Subsea manipulator wet tests.

Section 5.2 Kinematic modelling and control

This section presents simulation and experimental results related to robot kinematic modelling and control. The designed control software developed by the author utilises kinematics algorithms for subsea manipulators and is implemented in MATLAB and later in LabVIEW. As well as continued in this thesis a description of the developed systems has been published; please see a list of publications from the body of research in Appendix G. The use of the proposed control solution for typical subsea intervention tasks is evaluated by addressing the manipulation scenario 1 described in section 4.2 automatically and with enhanced manual and semi-automatic functions. Firstly, simulations to validate robot modelling and control techniques with industrial robot arms and subsea manipulators are described in subsection 5.2.1. Subsection 5.2.2 presents experiments with real industrial robot arms to verify the same algorithms. Subsection 5.2.3 presents the design and development of pilot control software for subsea manipulators that encapsulates the developed kinematic motion control functions. Finally, subsection 5.2.4 presents subsea environment experimental tests to validate the developed pilot control software with an underwater manipulator.

5.2.1 Simulations with industrial robot arms and subsea manipulators

The first simulation test case includes two identical models of Staubli TX60 industrial robot arms (Appendix F) derived from DH parameters which are given in Table 4.1.
5.2 Kinematic modelling and control

in section 4.3.2. These two physical manipulators exist in the robotics laboratory at the University of Limerick, and are labelled as Robot 1 and Robot 2 below. In this simulation, Robot 1 represents a subsea manipulator that is holding a sample container and Robot 2 another subsea manipulator that is placing a sample into the container. The simulation scenario is described as follows. The relative pose between the bases of two robotic manipulators is known; it is measured from the laboratory experimental setup and expressed in the homogeneous transformation matrix form. Both robots start from an initial position defined in joint space. Robot 1 then moves to an arbitrary but predefined position moving along a joint space trajectory and simulating the action of placing the sample container in the workspace of Robot 2. After that, Robot 2 approaches Robot 1, also moving along a trajectory defined in joint space, and attains a pose in which the end-effectors of the two robots are coaxial and separated from each other by a specified distance. Robot 2 then moves in a straight line in Cartesian space to reach the TCP of Robot 1, with its TCP simulating the action of placing the sample into the container. Finally, Robot 2 pulls back moving in a straight line, and after returns to the initial position moving in joint space. The simulation includes utilising the forward kinematics equation (equation 4.2, section 4.3.2) to determine the Cartesian position of the end-effector of Robot 1, expressing the resulting position in the base frame of Robot 2, and generating the required trajectories using the developed inverse kinematics algorithms described in section 4.3.3. All joint space trajectories are constructed based on 5th order polynomials and all Cartesian space trajectories are constructed based on the trapezoidal velocity profile. Robotics Toolbox for MATLAB (Corke, 2015) is used as a base for designing the related kinematics engine algorithms. Figs. 5.1a and 5.1b represent Cartesian position coordinates plotted against time for Robot 1 and Robot 2 respectively, and show that Robot 2 reached the desired point after 20s, while Fig. 5.2 illustrates the virtual animation of the scene during the simulation, realised with the Virtual Reality Modelling Language (VRML).

The second simulation test case is identical to the first one except that it uses a Schilling Titan 2 model (Appendix F) for the simulation of the robotic manipulator that is placing a sample into the container. The DH parameters that are used to design the robotic manipulator model are given in Table 4.2 in section 4.3.2. Fig. 5.3 illustrates the trace of trajectories after the simulation and Fig. 5.4 shows screenshots of the VRML animation.
Simulation and experimental test results

Fig. 5.1 Staubli TX60 end-effector Cartesian position during the Scenario 1 simulation

Fig. 5.2 Visualisation of the virtual scene with two Staubli TX60 robots during the Scenario 1 simulation

Fig. 5.3 Trajectory trace after the Scenario 1 simulation

5.2.2 Experiments with industrial robot arms

The same algorithms used in the simulations outlined above are used in real experiments to control two Staubli TX60 robot arms in a physical simulation of Scenario 1 described in section 4.2. The main robot software runs on a standard PC and communicates with two Staubli TX60 CS8C commercial controllers over Ethernet. The standard programming language for Staubli Robots called VAL3 is used to preprogram the two robot controllers for the experiment. Robot 1 controller is preprogrammed to move the robot to the position defined by the user at the start of the experiment and after that to continuously send its joint positions to the PC. Robot 2 controller is preprogrammed to move the robot by utilising straight-line Cartesian trajectories to the Cartesian position that is continuously received from the PC. Fig. 5.5 shows the photos taken during the experiment. For safety reasons, a specific distance offset is left between the robots’ end-effectors, and that is why there exists a gap between them, as can be seen from the far right image on this figure.
5.2 Kinematic modelling and control

Fig. 5.4 Visualisation of the virtual environment with Staubli TX60 and Schilling Titan 2 robot models during the Scenario 1 simulation

Fig. 5.5 Photos taken during the Scenario 1 experiment

5.2.3 Design and development of pilot control software for subsea manipulators

Successful realisation and validation of the kinematics control engine in simulation and experiments using industrial robot arms, described in sections 5.2.1 and 5.2.2, lead to the decision to develop control software for an underwater manipulator using the same algorithms. To this end, a LabVIEW application is developed that integrates forward and inverse kinematics solutions to enable specific robot functions, and its key kinematic control features are described in this subsection. Fig. 5.6 shows the user interface of the developed application. The designed pilot interface can be used for Schilling Robotics manipulators Titan 2 and Orion 7P. Both of these manipulators are industry standard hydraulic devices that are found in the global fleet of work-class ROVs and used daily in the offshore oil and gas industry and other marine-related sectors. The developed software encapsulates the majority of algorithms designed within the scope of the research project that this thesis reports. Manual and semi-
Simulation and experimental test results

Fig. 5.6 User interface of the developed LabVIEW application for control of subsea manipulators
5.2 Kinematic modelling and control

automatic functions that are related to subsea manipulator kinematic control are described as follows; other functions are covered later in this chapter.

Regarding joint space motion, the user can manually control each joint of the manipulator separately by moving sliders. Alternatively, the user can set a final position in joint space (six angles), select the arbitrary duration of the trajectory and the number of intermediate positions and on a click of a button, generate a smooth trajectory from the initial to the desired position. The modelled joint space trajectory is a quintic polynomial with zero boundary conditions for velocity and acceleration. It is also possible to record a certain number of described joint space configurations (stow, deploy, tool 1, tool 2, etc.) and command the manipulator to attain these predefined poses at a click of a mouse.

Regarding motion in Cartesian space, by moving sliders, the user can manually control the manipulator, forcing its end-effector to move along a straight line in one of three directions \((x, y, z)\) defined in the end-effector coordinate frame. Similarly, another three sliders exist for controlling the end-effector roll, pitch, and yaw angles. These functions cause the manipulator to rotate around the specified axis of the end-effector frame while holding the end-effector position. An alternative coordinate frame can be specified by defining its coordinate transformation relative to the end-effector frame and can then be used as a reference for the described Cartesian motion. In this manner, it is possible to address the motion of any tool held by the manipulator by controlling the position and orientation of the actual TCP frame in Cartesian space. Another function related to Cartesian space motion control allows the user to set the desired pose and generate a smooth trajectory moving the manipulator end-effector in a straight line to attain that pose. The first of two options for defining the desired pose is by a homogeneous transformation matrix. The second option is by defining the position with Cartesian coordinates and the orientation by using either roll-pitch-yaw or angle-axis representation. There is also the possibility of maintaining the initial position or the initial orientation during the motion. Regardless of the method used to define the desired pose, it is possible also to select the base or the end-effector frame as a reference. The modelled Cartesian trajectory is a sequence of homogeneous transformations that follow a trapezoidal velocity profile along the path, where the user specifies the number of intermediate transformations and the duration of the trajectory. The user interface of the developed software integrates real-time virtual animation of manipulator motion. The manipulator motion simulated in the virtual environment can be generated based on the motion commands issued by the user or based on the joint sensor feedback resulting from these motion commands. A function for circular motion in Cartesian space also exists where the user can specify a circle radius, number of intermediate points, and relevant coordinate frame—base
Simulation and experimental test results

or end-effector—and generate a trajectory that forces the TCP to move in a circle upon a virtual plane in space.

Besides the described user interface, a novel input device is integrated with the developed pilot control software. This device is a 3D mouse called SpaceMouse, manufactured by 3Dconnexion. Namely, by manipulating the mouse’s controller cap, the software generates Cartesian space trajectories (straight line and angular motion), with a possibility to enable or disable any number of end-effector DOFs of motion to avoid spurious inputs. Regardless of the way in which the user generates the manipulator motion commands, the output of the software is a sequence of joint angles mapped to the input of the commercial low-level actuator motion controller of the Schilling manipulator. Therefore, the generated motion does not represent a genuine trajectory with detailed time-ordered information about the position, velocity, and acceleration. However, with an adequate refresh rate, this is the best that can be achieved considering the constraints imposed by the commercial controller which does not allow any other motion control scheme other than the continuous PTP motion planning.

An alternative approach that would allow a more sophisticated motion control scheme such as trajectory tracking would require developing an entirely new low-level actuator driver for Schilling manipulators that could replace the existing ones, which is beyond the scope of this research project.

The described control software was initially developed to replace the Schilling MCU completely. However, to provide the possibility for the pilot to choose either traditional teleoperation mode using the miniature master controller or the automatic and semi-automatic functions described above, the overall system—hardware and software—was modified to include a software switch enabling one of the two input devices, with dual operation modes available. The benefit of this is that it eases the installation on the existing manipulator system and facilitates a smooth transition for pilots, maintaining backward compatibility.

5.2.4 Experiments with subsea manipulators

This section presents the experimental test results from some of the manual and semi-automatic functions encapsulated in the developed pilot control software described in section 5.2.3. The designed kinematics control engine algorithms were validated in a real-world underwater environment of a flooded quarry in Portroe (Ireland) using Schilling Titan 2 manipulator mounted on the Marine Institute owned work-class ROV—Holland 1.
First, a cube shaped 3D path is addressed, realised by implementing a combination of manual and semi-automatic Cartesian motion trajectories. It is constructed as a set of sequential straight line motion trajectories defined in the end-effector reference frame and mapped adequately to provide reference motion commands for the manipulator SCU. Fig. 5.7 shows the desired and actual Cartesian position coordinates of the manipulator end-effector during the motion and Fig. 5.8 illustrates the corresponding trajectory trace in 3D space. Additionally, circular motion was implemented in both end-effector (Figs. 5.9 and 5.10) and world (Figs. 5.11 and 5.12) reference frames. The presented results of Cartesian trajectory motion expose apparent positioning inaccuracy. This significant imperfection was revealed to be the result of the insufficient absolute accuracy of the integrated Schilling Titan 2 low-level joint position controller of the SCU. It was discovered that for a given position command after the joints move to the desired position, there remains an absolute error. Reading joint position values from the resolvers after the motion revealed that this error is significant and for specific joints reaches up to 1.5°. It was determined that the position error is different for each joint and that it depends on the direction from which the joint moves to the desired position, i.e. hysteresis is present. Moreover, the error value is different for the different desired positions through the range of motion. However, in the vicinity of the specified desired position (≈ ±20° for each joint), the error value turned out to be more-or-less repeatable. An additional function to calibrate the joints of the manipulator is designed and integrated into the pilot control software to battle this SCU controller defect. By performing certain motions relative to the nominal manipulator pose, it is possible to obtain a set of errors between the commanded and the actual joint positions. Based on the resulting errors, the joint calibration algorithm can then calculate the command offsets for each joint that may reduce the absolute positioning error. Unfortunately, the Cartesian motion trajectory tests were not repeated in the same conditions as the ones presented in this subsection to reveal the effectiveness of the described joint
Simulation and experimental test results

Fig. 5.8 3D trajectory trace during the experiment of utilising semi-automatic Cartesian motion functions to generate a cube shaped path

calibration algorithm. However, the joint calibration algorithm is validated for a different experimental test, presented later in section 5.3.4, reducing absolute joint position error values below 0.7°.

Despite the presence of an absolute positioning error, the developed Cartesian trajectory motion control functions proved to be useful. During the field trials held in the Portroe flooded quarry, two unexpected manipulation tasks arose that had to be carried out. The first task was to relocate and rotate a metal construction that was used for experiments and the other to move a rope that was in the way and interfered with the experiments, behind the metal construction. Instead of teleoperating the manipulator in the traditional manner to perform these tasks, it was decided to use these circumstances to test the developed manipulator control schemes. Guided by the visual feedback of the scene provided by the camera, an inexperienced ROV manipulator pilot (the author) utilised the Cartesian motion functions to operate the manipulator and accomplished the given tasks successfully, as can be seen from Figs. 5.13a and 5.13b. Fig. 5.14 shows the Cartesian trajectory generated during the rope removal manipulation task, Fig. 5.14 the corresponding joint space trajectories, and Fig. 5.16 the trace of the resulting 3D reference trajectory. The red trace represents motion until the rope was gripped, which occurred after 50 s; the blue trace, motion to relocate the rope, which occurred at 193 s, and the green trace, the pulling back motion. Due to the common problem of inadequate 3D perception based on the 2D
5.2 Kinematic modelling and control

Fig. 5.9 End-effector position coordinates during the experiment of utilising semi-automatic Cartesian motion functions to generate a circular path in end-effector frame

Fig. 5.10 Trace of the 3D trajectory during the experiment of utilising semi-automatic Cartesian motion functions to generate a circular path in end-effector frame

image displayed on the screen, it took three attempts to grasp the target and two attempts to position it at a suitable location. A visual-based motion control algorithm designed to tackle this issue is addressed later in section 5.3.

Manual and semi-automatic Cartesian functions encapsulated in the developed pilot control software were utilised to control the manipulator within another experi-
Simulation and experimental test results

![Graph showing end-effector position coordinates during the experiment.](image)

Fig. 5.11 End-effector position coordinates during the experiment of utilising semi-automatic Cartesian motion functions to generate a circular path in robot base frame.

![Graph showing 3D trajectory trace.](image)

Fig. 5.12 Trace of the 3D trajectory during the experiment of utilising semi-automatic Cartesian motion functions to generate a circular path in robot base frame.

mental research test to perform 3D reconstruction of a subsea environment. Straight line and angular motion of the camera mounted on the end-effector of the manipulator was required to provide proper coverage of the scene. A researcher colleague, Matija Rossi, who developed the 3D reconstruction software is summarising the results and is soon to submit them for publication.

**Concluding remarks for section 5.2**

As outlined above, kinematics control engine is developed for subsea robotic manipulators using techniques common for industrial robotics. The developed software is capable of generating reference kinematics parameters for manipulator motion both in joint space and in Cartesian space. Utilising the proposed robotic modelling tools, various subsea intervention tasks that are typically performed by a human teleoperator can be automated. Therefore, the developed software may prove to be useful for ROV pilots, enabling them to execute a certain number of tasks “at the click of a button”, and hence, preserve their energy and concentration for other
5.2 Kinematic modelling and control

Fig. 5.13 Photos taken during the manipulation tasks of (a) relocating a metal frame and (b) removing a rope for the real world validation of manual and semi-automatic Cartesian motion functions

Fig. 5.14 Cartesian coordinates of the reference trajectory during the rope removal manipulation task

delicate tasks. Additionally, the designed visualisation tools can be integrated into the control software and aid ROV pilots to gain better spatial awareness through the virtual real-time animation when the camera systems are insufficient, e.g. in reduced visibility conditions, and manipulate the robotic arms to accomplish tasks in challenging conditions. Consequently, performing subsea intervention operations would be more comfortable for pilots, safer, faster, and therefore, more cost-effective.
Simulation and experimental test results

Fig. 5.15 Joint space reference trajectory during the rope removal manipulation task

Fig. 5.16 3D trace of the reference trajectory during the rope removal manipulation task
5.3 Visual-based motion control

This section presents experimental test results using the developed visual-based motion control algorithms to address manipulation tasks from scenarios 2 to 5 described in section 4.2. The designed visual servoing system for subsea manipulation is extensively tested with different industrial and underwater manipulators in dry laboratory conditions and field trials in the real subsea environment. Multiple test cases are presented, utilising the developed PBVS algorithm, described in section 4.4. The designed control software is implemented in MATLAB for initial testing and then in LabVIEW, encapsulated in the developed pilot control software described in section 5.2.3. Firstly, the implementation of machine vision algorithms for pose estimation is described in subsection 5.3.1. Subsection 5.3.2 presents simulations with industrial robot models and a real camera system in the loop to validate the developed pose estimation software. Dry environment laboratory tests addressing a stationary target with an industrial robot arm are presented in subsection 5.3.3. Subsection 5.3.4 presents experimental results addressing a stationary target with an underwater manipulator in dry laboratory conditions and subsea field trials, and initial results addressing a target in motion. Subsection 5.3.5 describes a motion analysis study with simulations of an MRE device, performed to comprehend how the target objects for subsea manipulation move in the water column. The motion resulting from the simulations is then replicated with an industrial robot arm with the results presented in subsection 5.3.6. Finally, extensive experimental tests with a moving target are presented in subsection 5.3.7 for an industrial robot arm and in subsection 5.3.8 for an underwater manipulator.

5.3.1 Implementation of machine vision algorithms for pose estimation

Two vision systems are used within this research project, those are a Point Gray Bumblebee 2 stereo camera (Fig. 5.17) and a Point Gray BlackFly monocular camera (Fig. 5.18). The use of stereo vision was considered at the initial stage of this project as various pose estimation methods using stereo vision exist in the literature, some of which are covered in section 2.4. However, this project focuses on the development of vision-guided motion control algorithms for robotic manipulators rather than on the development of advanced machine vision algorithms. Moreover, another researcher from the same research group is engaged in the development of advanced monocular vision systems, i.e. real-time visual-based 3D reconstruction and navigation, and there is a plan to fuse the two systems once they reach a sufficient
Simulation and experimental test results

level of development. Therefore, to avoid unnecessary additional development work, it was decided to abandon stereo vision within this project and simplify the camera imaging for target pose estimation by using available algorithms for monocular cameras that utilise fiducial markers. At the time this decision was made, the Bumblebee 2 stereo camera was already acquired, so it was used in some experiments, only in a monocular mode. Additionally, there are technical issues that make integration of Bumblebee 2 camera on the ROV system complicated because it uses a Firewire interface standard for data transfer, which is very sensitive to cable length and noise, and no subsea standard connectors and cables exist to facilitate its application underwater. Moreover, besides getting signals from the camera mounted on the manipulator end-effector to the ROV electronics bottle, there exists the problem of converting and transferring data signals from the ROV to the surface PC over a fibre-optic communication network. No standard industrial equipment exists that can enable such (Firewire) communication. The manipulator control software, or at least the camera imaging part may be modified to run on the bottom side to avoid this communication constraint; however, this would again require additional development on the technical side. For the reasons outlined above, a standard monocular computer vision camera (BlackFly) that utilises Power over Ethernet (PoE) for data transfer was obtained and integrated with a suitable subsea housing and used in underwater experiments. Machine vision software performing fiducial marker pose estimation is developed in the C++ programming language using different open source libraries (OpenCV, ArUco, FlyCapture, and Triclops). The developed program is encapsulated in the form of a dynamic-Link Library (DLL) so that it can be used within other software. It includes a camera calibration algorithm that utilises standard computer vision techniques based on planar homography and a chessboard (Fig. 5.19) as a calibration rig (Bouguet, 2015; Zhang, 2000) for the
5.3 Visual-based motion control

Fig. 5.19 Camera image of a chessboard (a) before and (b) after the calibration

Fig. 5.20 Camera image of a fiducial marker (a) before and (b) after the detection

estimation of distortion coefficients, intrinsic parameters, and extrinsic parameters. Running the pose estimation software requires a calibrated camera; it works as follows: the camera continuously captures images; algorithm removes the distortion using intrinsic parameters; detects the fiducial marker on the undistorted image, and using the extrinsic parameters calculates the fiducial marker pose in the camera reference frame ($H_{CAM}^M$). The fiducial marker detection on the image (Fig. 5.20) is done using the ArUco library developed by (Garrido-Jurado et al., 2014).

5.3.2 Simulations with industrial robot arms and a real camera

This section presents a hardware-in-the-loop simulation with kinematic models of industrial robot arms and a real camera. The goal of this simulation is twofold: to derive and validate coordinate transformations necessary for the implementation of the PBVS algorithm and to test the pose estimation software. The designed simulation software is implemented in MATLAB, and it includes two Staubli TX60 industrial robot arm models (Appendix F) derived from DH parameters available in Table 4.1 in section 4.3.2. These two robotic manipulators exist in the robotics
Simulation and experimental test results

laboratory at the University of Limerick. The simulation utilises the developed
kinematics control engine to determine the reference motion that the manipulator
has to perform to address a target based on the pose estimation software input. The
simulation is based on the following assumptions. The first robot arm, labelled
as Robot 1, is equipped with a fiducial marker, placed directly on its end-effector.
The camera is mounted on the end-effector of the other manipulator, labelled Robot
2. The relative pose between the two robot manipulators is known. At the start
of the simulation, the manipulators are in such initial configuration that their end-
effectors point at each other and there is a certain distance between them. During
the simulation / experiment, the camera (Robot 2) remains stationary while a human
participant physically moves the fiducial marker, keeping it in the camera field of
view. With the known joint position of Robot 2 and the information provided by
the pose estimation software, it is possible to calculate the fiducial marker pose
with reference to Robot 2 base frame. Additionally, the knowledge of the relative
pose between the robots’ bases, allows expressing the fiducial marker pose in the
Robot 1 base frame. Finally, with the aid of the inverse kinematics algorithm, it is
possible to calculate joint positions and visualise Robot 1 in a configuration such
that its end-effector is at the location of the physical fiducial marker. As the fiducial
marker moves so does the model of Robot 1 in the simulation. Observing that
the fiducial marker motion induced by a human participant is correctly mapped in
the virtual environment provides the acknowledgement that the derived coordinate
transformations are valid. The other part of the simulation is related to determining
the desired Robot 2 pose that positions its end-effector relative to the target in a
particular way suitable for performing a visual servoing task. Three test cases are
addressed, each with a differently defined desired end-effector to target alignment.

In the first test case, the task of Robot 2 is to orient its end-effector so that z-axis
points at the centre of the fiducial marker throughout its motion but without changing
the end-effector Cartesian position. The estimated fiducial marker position vector
contained in the homogeneous transformation matrix provided by the pose estimation
software represents the direction from the camera to the target. The given orientation
problem, illustrated in Fig. 5.21, is to find the rotation matrix that aligns the z-axis
of the current camera coordinate frame \( O_{xyz} \) with the fiducial marker position vector,
that is, with the \( z' \)-axis of the desired camera coordinate frame that is to be formed.
Additionally, the plane formed by the \( z' \)-axis and the \( y' \)-axis has to be perpendicular
to the \( xy \)-plane; this ensures that the fiducial marker is upright in the camera image.
The desired rotation matrix is formed by the following equation:

\[
R_{des} = R_z\left(\beta - \frac{\pi}{2}\right) \cdot R_x(-\alpha) \tag{5.1}
\]
Fig. 5.21 Coordinate transformations required to orient the end-effector mounted camera to point towards the fiducial marker

where the angles $\alpha$ and $\beta$ are determined via standard vector transformations, as follows:

$$\beta = \arctan2 \left( \frac{z' \times k \cdot z'}{\|z'\|}, \frac{z' \cdot k}{\|z'\|} \right)$$  \hspace{1cm} (5.2)$$

$$z'_{xy} = z' - (z' \cdot k) \cdot k$$  \hspace{1cm} (5.3)$$

$$\alpha = \arctan2 \left( \frac{z'_{xy} \cdot j \cdot z'_{xy} \cdot i}{\|z'_{xy}\|}, \frac{z'_{xy} \cdot i}{\|z'_{xy}\|} \right)$$  \hspace{1cm} (5.4)$$

The resulting rotation matrix is then combined with the current end-effector position vector to construct the homogeneous transformation matrix and utilise the inverse kinematics algorithm to obtain the corresponding joint position. Fig. 5.22 shows simulation screenshots of the visual environment to validate the resulting robot configurations. As the simulation is designed based on perfect models, if the coordinate transformations are performed correctly there exists no possibility for error occurrence; therefore, there is no other way to verify the results except visually.

In the second case, as in the previous one, the end-effector of Robot 2 keeps its Cartesian position unchanged; however, instead of pointing with $z$-axis at the fiducial marker, it remains parallel to it, i.e. to the axis that points out from the fiducial marker (Fig. 5.23). The homogeneous transformation to achieve such posture can is obtained by combining the current end-effector position vector with the rotation matrix that is contained in the fiducial marker pose.

The last test case addresses both position and orientation where the end-effector of Robot 2 follows the fiducial marker, keeping a specified pose relative to it; in this particular simulation, the end-effector and target $z$-axes are collinear with a fixed
Fig. 5.22 Simulations of generating reference manipulator configurations to orient the end-effector mounted camera to point towards the fiducial marker.

Fig. 5.23 Simulations of generating reference manipulator configurations to orient the end-effector mounted camera parallel to the fiducial marker.
5.3 Visual-based motion control

distance between the origins of their coordinate frames, and their $y$-axes are parallel (Fig. 5.24). The desired end-effector pose in this test case is the combination of poses from the previous two test cases as it meets the conditions from both, i.e. the camera points at the fiducial marker and is parallel to it. Post-multiplying the fiducial marker pose with the inverse of the desired camera-to-marker pose results in the homogeneous transformation matrix that describes the motion the end-effector needs to perform to achieve the desired posture.

5.3.3 Experiments of the PBVS algorithm with an industrial robot arm addressing a stationary target

This subsection describes experimental tests using Staubli TX60 industrial robot arms to validate the developed PBVS algorithm, described in section 4.4.1. Different variations of the designed vision-guided control scheme are utilised in multiple test scenarios to drive the robotic manipulator and position the camera mounted on its end-effector in a desired pose relative to the target. The PBVS algorithm is realised in open-loop and closed-loop configurations with analytical and numerical inverse kinematics implementations and is employed for the task of locating, collecting, manipulating and returning a target object to its initial location. In the addressed experiments, the target object for manipulation is a box with a fiducial marker attached to its surface. In some experiments, the target lies on the surface of the workbench; in others, it is attached to the end-effector of Robot 1. The reason for mounting the fiducial marker on the manipulator’s end-effector is that such arrangement allows the calculation of the pose of the target relative to the robot base frame, and this information can be adopted as the ground truth for the validation of
Simulation and experimental test results

the pose estimation software. As these experiments address visual servoing with stationary targets, Robot 1 does not move during them. This setup also allows addressing target in motion, which is discussed later in subsections 5.3.5–5.3.8. At the beginning of each experiment, Robot 1 is positioned so that the target on its end-effector is located in the workspace of the other robot arm, which is the main visual servoing robot arm equipped with a camera. Robot 2 is equipped with a vacuum gripper for target object acquisition. At the beginning of the experiment, it is positioned in such configuration that the target is in the field of view of the camera. The manipulation task addressed in these experiments consist of two approach phases. In the 1st approach phase the goal of Robot 2 is to position the camera mounted on its end-effector right in front of the fiducial marker at a predefined distance from it. Such relative distance then simplifies the 2nd approach phase, which is the main approach phase in which the robot gripper reaches the target. In the 1st approach phase, the homogeneous transformation matrix that represents the reference for the motion controller \( H_{des}^{B_2}_{EE_2} \), is adjusted to include a specified \( z \)-axis offset; in this experiment it is 250 mm. This adjustment forces the manipulator to attain such pose that the gripper is 250 mm in front of the target and looking straight at it. The 1st approach phase is active until the control error variable, formed as the Euclidean distance between the desired and actual Cartesian position vectors, drops below the specified threshold value. The error threshold value is relaxed in the 1st approach phase as the accuracy is not of much importance at this stage, i.e. the goal is to acquire satisfactory initial conditions for the 2nd approach phase. For this particular experiment, the specified error threshold for the 1st approach phase is 8 mm, which means that once the end-effector comes this close to the desired Cartesian point, the 2nd approach phase triggers. The reference homogeneous transformation matrix \( H_{des}^{B_2}_{EE_2} \) is not modified at all in the 2nd approach phase, which forces the manipulator to attain a posture that is suitable for gripping. Instead of forming a single Euclidian distance error variable, the control error is in the 2nd approach phase addressed for each Cartesian coordinate separately, and the control error threshold values are set on much lower values. For this particular experiment, the threshold values are 1 mm for \( x \) and \( y \) coordinates, and 0.5 mm for the \( z \) coordinate. In the event of the fiducial marker not being detected during any of the two approach phases, a special phase called the search phase is triggered. This phase includes such robot motion that the end-effector pitch and yaw angles increase alternately in each direction for a specified angle until the target is detected or until the predefined angle limits are reached. The resulting end-effector trajectory is a spiral such that the camera points further away from the axis of the spiral in each turn. If the fiducial marker is not detected throughout the whole search phase, it is
assumed that the target is not in the work scene and the visual servoing algorithm stops. The search phase also occurs if the target is not detected at the beginning of the experiment. As soon as the target is detected, the search phase stops, and the phase which was previously interrupted becomes the active phase. After the 2nd approach phase, the manipulator moves in a straight line towards the target object without visual feedback, reaching its surface with vacuum gripper suction cups. This phase is referred to as the grip phase and includes robot motion only due to the use of a vacuum gripper. If the robot arm has a mechanical gripper, no robot motion is required during this phase; rather, it is necessary to stop the motion until the gripper closes. After completing the grip phase, the vacuum gripper activates, and the manipulator then pulls back holding the target object.

**Open-loop PBVS algorithm experiments**

The experimental test results in which the open-loop PBVS algorithm operates with a look then move structure is utilised for the manipulation task described below. The developed visual servoing control scheme is applied with two distinct implementations of inverse kinematics solutions—analytical and numerical—and their performance is analysed and compared. Robot 2 accomplished the manipulation task successfully utilising both of the algorithms (Fig. 5.25). It completed the 1st approach phase in one go, positioning the end-effector in the desired pose with sufficient accuracy. However, this is not the case for the 2nd approach phase which required an additional control loop iteration to correct the end-effector pose and align the camera with the fiducial marker. For both approach phases, the PBVS
Simulation and experimental test results

The control loop algorithm incorporates four distinct segments: the acquisition of joint angle positions (T1), which consists of a pair of messages—request and response—parsed and transmitted over Ethernet between the PC and the Staubli controller; the pose estimation (T2), which includes image acquisition and processing, and planar homography computation; the segment for the coordinate transformation and inverse kinematics computation (T3), and the last segment (T4), comprised of a position command issued over Ethernet by the PC, robot arm motion to attain the commanded position, and a verification response that the manipulator completed the motion. In the grip phase, the coordinate transformation and inverse kinematics computation are joined with the last segment; the Staubli controller performs it. The duration of individual segments throughout the experiment is given for the PBVS algorithm with analytical and numerical inverse kinematics implementations in Fig. 5.26 and Fig. 5.27 respectively. In line with expectations, the PBVS algorithm with the analytical inverse kinematics solution implementation required less time, albeit not much less, to reach the target and execute the manipulation task. The reason is that the numerical computation of the inverse kinematics solution takes more than double the time compared to the computation of the analytical solution due to iterative nature. Regardless, the segment that took the longest time in both cases is T4, which includes the robot motion. Regarding the failure of the robot arm to complete the 2nd approach phase in a single iteration, the reason is most likely a combination of modelling errors and errors introduced by the pose estimation software, both of which are increasingly expressed with distance to the target, and not due to the positioning accuracy of the Staubli TX60 robot arm, which is submillimeter as per technical documentation. Owing to such positioning accuracy, the robot arm can be used as the ground truth to estimate the measurement accuracy of the pose estimation software. Modelling errors can be reduced by performing eye-to-hand calibration. However, none of these approaches was addressed at this stage. If ameliorated, the PBVS realised in the described open-loop configuration might save less than one second which is not of utmost importance for industrial subsea applications. Instead, the goal was to continue towards the development of a PBVS algorithm that is capable of addressing both stationary and moving targets, and the first step in doing that was closing the loop with the dynamic look and move configuration.

Closed-loop PBVS algorithm experiments

The remainder of this subsection presents the experimental evaluation of the developed closed-loop PBVS algorithm with a dynamic look and move structure, detailed in subsection 4.4.1. The goal of the experiments was to use an industrial robot arm
5.3 Visual-based motion control

Fig. 5.26 Partitioned execution time of the open-loop PBVS algorithm with *look then move* structure and analytical inverse kinematics

Fig. 5.27 Partitioned execution time of the open-loop PBVS algorithm with *look then move* structure and numerical inverse kinematics

to validate the control scheme that is developed to be implementable on Schilling Titan 2 subsea manipulators. Therefore, the addressed PBVS algorithm encapsulates numerical derivation of inverse kinematics; a closed-form analytical inverse kinematics solution for Titan 2 does not exist due to its mechanical configuration, i.e. absence of a spherical wrist. The visual servoing manipulation task is identical to the one described in the previous subsection; the only exception in this experimental scenario is that the fiducial marker is located on the end-effector of *Robot 1*. The developed PBVS algorithm contains two parameters: the number of iterations of the internal control loop inverse kinematics algorithm (IP), and the parameter that defines the portion of the path from the initial to the desired pose along which the end-effector moves in each external control loop, i.e. the path percentage (PP) parameter. The higher the value of the IP parameter, the more accurate the inverse kinematics solution, but at the expense of a longer inner PBVS control loop duration.
Simulation and experimental test results

In contrast, the PP parameter does not affect the duration of either the inner or the outer loop, but it affects the total number of outer loops/iterations required as it determines the nominal rate of convergence of the camera towards the target. Fig. 5.28 illustrates the relationship between the PP parameter, assuming it is constant, and the normalised distance from the camera to the target, i.e. this figure shows the number of theoretical PBVS iterations required for the manipulator to reach the target for a given PP parameter. To avoid confusion, the value “1” on the vertical axis represents the whole distance travelled.

**Experiments with different PBVS parameter values**

Multiple sets of experiments were conducted to analyse different performance aspects of the developed PBVS algorithm. The first set of experiments aimed to investigate the effect of different IP and PP parameter combinations on the overall efficiency of the PBVS algorithm. Identical experiments with the task to autonomously locate and approach a stationary target with positioning accuracy adequate for grasping were repeated multiple times with different constant values assigned to these parameters. PP parameter values from 5% to 100% were addressed in combination with IP parameter values from 3 to 20. The focus was to identify the total time and number of control loops needed to complete the task, as well as the duration of each segment of the control loop (Table 5.2). The results presented in this table enable analysis of the PBVS algorithm performance with various combinations of PP and IP parameter values. The total time required to perform the visual servoing task is one of the most
5.3 Visual-based motion control

Table 5.2 Comparative performance analysis of the closed-loop PBV algorithm with *look and move* structure with different combinations of IP and PP parameter values

<table>
<thead>
<tr>
<th>IP</th>
<th>PP</th>
<th>AVG (ms)</th>
<th>MIN (ms)</th>
<th>MAX (ms)</th>
<th>AVG (ms)</th>
<th>MIN (ms)</th>
<th>MAX (ms)</th>
<th>AVG (ms)</th>
<th>MIN (ms)</th>
<th>MAX (ms)</th>
<th>AVG (ms)</th>
<th>MIN (ms)</th>
<th>MAX (ms)</th>
<th>AVG (ms)</th>
<th>MIN (ms)</th>
<th>MAX (ms)</th>
</tr>
</thead>
<tbody>
<tr>
<td>5</td>
<td>Cam</td>
<td>82</td>
<td>64</td>
<td>234</td>
<td>81</td>
<td>64</td>
<td>103</td>
<td>80</td>
<td>64</td>
<td>94</td>
<td>82</td>
<td>64</td>
<td>94</td>
<td>84</td>
<td>67</td>
<td>104</td>
</tr>
<tr>
<td></td>
<td>Move</td>
<td>121</td>
<td>101</td>
<td>181</td>
<td>126</td>
<td>102</td>
<td>169</td>
<td>130</td>
<td>103</td>
<td>192</td>
<td>134</td>
<td>102</td>
<td>190</td>
<td>134</td>
<td>102</td>
<td>190</td>
</tr>
<tr>
<td></td>
<td>Cam</td>
<td>0.220</td>
<td>738</td>
<td>109</td>
<td>0.230</td>
<td>354</td>
<td>84.2</td>
<td>0.249</td>
<td>192</td>
<td>48.8</td>
<td>0.277</td>
<td>130</td>
<td>43.0</td>
<td>0.295</td>
<td>153</td>
<td>46.8</td>
</tr>
<tr>
<td></td>
<td>Move</td>
<td>17</td>
<td>16</td>
<td>118</td>
<td>24</td>
<td>23</td>
<td>123</td>
<td>40</td>
<td>39</td>
<td>138</td>
<td>60</td>
<td>57</td>
<td>156</td>
<td>83</td>
<td>80</td>
<td>182</td>
</tr>
<tr>
<td></td>
<td>Cam</td>
<td>0.224</td>
<td>381</td>
<td>88.8</td>
<td>0.236</td>
<td>171</td>
<td>42.1</td>
<td>0.257</td>
<td>93</td>
<td>25.5</td>
<td>0.285</td>
<td>231</td>
<td>21.7</td>
<td>0.305</td>
<td>75</td>
<td>23.8</td>
</tr>
<tr>
<td></td>
<td>Move</td>
<td>17</td>
<td>16</td>
<td>115</td>
<td>25</td>
<td>23</td>
<td>126</td>
<td>42</td>
<td>39</td>
<td>139</td>
<td>61</td>
<td>57</td>
<td>156</td>
<td>84</td>
<td>80</td>
<td>182</td>
</tr>
<tr>
<td></td>
<td>Cam</td>
<td>0.227</td>
<td>187</td>
<td>43.4</td>
<td>0.247</td>
<td>84</td>
<td>22.4</td>
<td>0.272</td>
<td>43</td>
<td>12.0</td>
<td>0.301</td>
<td>34</td>
<td>10.8</td>
<td>0.324</td>
<td>35</td>
<td>12.0</td>
</tr>
<tr>
<td></td>
<td>Move</td>
<td>81</td>
<td>65</td>
<td>95</td>
<td>80</td>
<td>63</td>
<td>95</td>
<td>78</td>
<td>63</td>
<td>92</td>
<td>82</td>
<td>66</td>
<td>96</td>
<td>80</td>
<td>66</td>
<td>93</td>
</tr>
<tr>
<td></td>
<td>Cam</td>
<td>0.234</td>
<td>119</td>
<td>28.5</td>
<td>0.250</td>
<td>59</td>
<td>15.4</td>
<td>0.280</td>
<td>30</td>
<td>8.7</td>
<td>0.315</td>
<td>22</td>
<td>7.7</td>
<td>0.340</td>
<td>22</td>
<td>8.1</td>
</tr>
<tr>
<td></td>
<td>Move</td>
<td>80</td>
<td>65</td>
<td>97</td>
<td>80</td>
<td>65</td>
<td>94</td>
<td>77</td>
<td>65</td>
<td>95</td>
<td>81</td>
<td>73</td>
<td>92</td>
<td>79</td>
<td>66</td>
<td>95</td>
</tr>
<tr>
<td></td>
<td>Cam</td>
<td>0.238</td>
<td>85</td>
<td>28.9</td>
<td>0.256</td>
<td>44</td>
<td>11.6</td>
<td>0.264</td>
<td>24</td>
<td>7.4</td>
<td>0.331</td>
<td>15</td>
<td>5.3</td>
<td>0.347</td>
<td>16</td>
<td>5.9</td>
</tr>
<tr>
<td></td>
<td>Move</td>
<td>18</td>
<td>16</td>
<td>117</td>
<td>27</td>
<td>23</td>
<td>125</td>
<td>45</td>
<td>39</td>
<td>139</td>
<td>68</td>
<td>57</td>
<td>156</td>
<td>88</td>
<td>80</td>
<td>180</td>
</tr>
<tr>
<td></td>
<td>Cam</td>
<td>0.241</td>
<td>71</td>
<td>18.0</td>
<td>0.269</td>
<td>33</td>
<td>9.1</td>
<td>0.305</td>
<td>18</td>
<td>5.8</td>
<td>0.354</td>
<td>10</td>
<td>3.9</td>
<td>0.357</td>
<td>13</td>
<td>5.0</td>
</tr>
<tr>
<td></td>
<td>Move</td>
<td>80</td>
<td>66</td>
<td>98</td>
<td>83</td>
<td>69</td>
<td>95</td>
<td>79</td>
<td>64</td>
<td>92</td>
<td>81</td>
<td>68</td>
<td>95</td>
<td>82</td>
<td>66</td>
<td>95</td>
</tr>
<tr>
<td></td>
<td>Cam</td>
<td>1.64</td>
<td>88</td>
<td>34</td>
<td>77</td>
<td>66</td>
<td>88</td>
<td>81</td>
<td>65</td>
<td>94</td>
<td>81</td>
<td>71</td>
<td>92</td>
<td>83</td>
<td>69</td>
<td>96</td>
</tr>
<tr>
<td></td>
<td>Move</td>
<td>81</td>
<td>16</td>
<td>117</td>
<td>27</td>
<td>23</td>
<td>124</td>
<td>47</td>
<td>39</td>
<td>141</td>
<td>69</td>
<td>57</td>
<td>162</td>
<td>91</td>
<td>80</td>
<td>178</td>
</tr>
<tr>
<td></td>
<td>Cam</td>
<td>0.246</td>
<td>58</td>
<td>14.9</td>
<td>0.265</td>
<td>28</td>
<td>7.8</td>
<td>0.307</td>
<td>14</td>
<td>4.6</td>
<td>0.349</td>
<td>9</td>
<td>3.5</td>
<td>0.372</td>
<td>10</td>
<td>4.1</td>
</tr>
<tr>
<td></td>
<td>Move</td>
<td>81</td>
<td>65</td>
<td>102</td>
<td>79</td>
<td>68</td>
<td>95</td>
<td>79</td>
<td>66</td>
<td>91</td>
<td>85</td>
<td>77</td>
<td>96</td>
<td>80</td>
<td>67</td>
<td>92</td>
</tr>
<tr>
<td></td>
<td>Cam</td>
<td>0.245</td>
<td>52</td>
<td>13.1</td>
<td>0.271</td>
<td>24</td>
<td>6.8</td>
<td>0.336</td>
<td>13</td>
<td>4.7</td>
<td>0.358</td>
<td>8</td>
<td>3.2</td>
<td>0.390</td>
<td>7</td>
<td>3.1</td>
</tr>
<tr>
<td></td>
<td>Move</td>
<td>79</td>
<td>64</td>
<td>98</td>
<td>83</td>
<td>66</td>
<td>117</td>
<td>81</td>
<td>68</td>
<td>99</td>
<td>82</td>
<td>70</td>
<td>96</td>
<td>82</td>
<td>69</td>
<td>96</td>
</tr>
<tr>
<td></td>
<td>Cam</td>
<td>0.248</td>
<td>43</td>
<td>11.0</td>
<td>0.279</td>
<td>23</td>
<td>6.7</td>
<td>0.331</td>
<td>10</td>
<td>3.9</td>
<td>0.368</td>
<td>3</td>
<td>2.2</td>
<td>0.438</td>
<td>5</td>
<td>2.5</td>
</tr>
<tr>
<td></td>
<td>Move</td>
<td>81</td>
<td>64</td>
<td>95</td>
<td>80</td>
<td>65</td>
<td>90</td>
<td>79</td>
<td>65</td>
<td>87</td>
<td>82</td>
<td>68</td>
<td>92</td>
<td>82</td>
<td>71</td>
<td>92</td>
</tr>
<tr>
<td></td>
<td>Cam</td>
<td>0.257</td>
<td>55</td>
<td>9.4</td>
<td>0.281</td>
<td>18</td>
<td>5.4</td>
<td>0.332</td>
<td>11</td>
<td>4.6</td>
<td>0.365</td>
<td>10</td>
<td>4.5</td>
<td>0.403</td>
<td>9</td>
<td>3.9</td>
</tr>
<tr>
<td></td>
<td>Move</td>
<td>79</td>
<td>67</td>
<td>96</td>
<td>80</td>
<td>63</td>
<td>96</td>
<td>81</td>
<td>66</td>
<td>95</td>
<td>82</td>
<td>67</td>
<td>96</td>
<td>80</td>
<td>66</td>
<td>96</td>
</tr>
<tr>
<td></td>
<td>Cam</td>
<td>0.257</td>
<td>31</td>
<td>8.3</td>
<td>0.290</td>
<td>15</td>
<td>4.9</td>
<td>0.330</td>
<td>32</td>
<td>10.9</td>
<td>0.360</td>
<td>33</td>
<td>12.2</td>
<td>0.399</td>
<td>28</td>
<td>12.0</td>
</tr>
</tbody>
</table>

important criteria. For this set of experiments, total time is given as a function of PP and IP parameters in Fig. 5.29. For adequate visualisation, the z-axis is displayed in logarithmic scale. In contrast to the expectations, the shortest task execution time is not achieved with the PP parameter value of 100% for an arbitrary IP parameter, but only for the IP values up to 5. Instead, for IP parameter values of 9 and higher, the results show that the shortest time required to complete the task is achieved with the PP parameter value of 80%. Moreover, the experiment with the shortest total task execution time (2.5 s) is achieved for the IP parameter value of 20, and it took 5 visual servoing iterations. The reason that the combinations with PP parameter of 80% outperform the ones with PP parameter over 90% is that in the latter cases there is a repetitive overshoot that causes the end-effector to oscillate around the
Simulation and experimental test results

Fig. 5.29 Duration of the visual servoing manipulation task as a function of PP and IP parameters

desired pose, which leads to a higher number of visual servoing iterations, and consequently longer task execution time. Another interesting observation is that for a fixed PP parameter value in 5% - 60% range, experiments with IP parameter of 14, outperformed the experiments with lower and higher IP parameter values. On the other hand, for the experiments with the PP parameter over 70%, the higher the IP the greater the performance. This observation implies that if the goal is not to arrive at the desired location in one go but in a certain number of iterations, then the inverse kinematics accuracy of intermediate steps is not that relevant and too many iterations of the inverse kinematics algorithm might be not only unnecessary but also counterproductive. Additional experimental tests could be performed varying the PP parameter around the value of 80% and the IP parameter around the value of 14 to find the optimal combination. Returning to Table 5.2, observations can be made for the time required for each stage of the PBVS algorithm. Technical implementation of the PBVS algorithm addressed for the experiments just described is slightly improved compared to the one of the open-loop PBVS algorithm. The difference is that one pair of communication messages is removed from the control
loop by fusing the segment in which joint positions are acquired with the segment in which robot motion commands are issued. This is possible to achieve because the robot arm, after attaining the desired configuration, now sends a response which includes the current joint position. As a result, there are just three segments in the addressed PBVS algorithm: pose estimation (T1), inverse kinematics computation (T2), and motion command (T3). The time required for the pose estimation is more or less constant in all experiments with the average value of 81 ms. The minimum recorded time required for this algorithm stage is 63 ms and the maximum 234 ms. Such a long maximum duration for pose estimation is caused solely because of the whole PBVS algorithm implementation in MATLAB. The reason is that MATLAB needs some time to get up to speed, which is why the maximum recorded values are always in the first few visual servoing iterations. Regardless of that, the pose estimation software itself can be further optimised, and with that, the time for the vision part of the PBVS algorithm reduced. Multithreading is also an option, with the pose estimation software running in a separate thread, performing image acquisition, processing, and other computations unrelated to the PBVS algorithm, which would save time significantly. The time required to calculate the inverse kinematics obviously depends on the number of iterations (IP). The results show that the average time for a single iteration is 5 ms. The maximum recorded time for the inverse kinematics computation with 20 iterations was 182 ms. The time it takes for this segment of the PBVS algorithm may prove to be too long for a single visual servoing iteration when targets in motion come into play. A simple solution that may speed up this process is to implement the numerical inverse kinematics algorithm in C++ instead of MATLAB. This should not be difficult, as, besides the Jacobian derivation, the remainder is comprised of basic mathematical operations. The time required for the robot to move to a given pose depends solely on its specifications. For the addressed experiments, Staubli TX60 robot arm was set to move with highest possible velocity and acceleration. The results show that this segment of the PBVS algorithm took from 100 ms to 410 ms. Underwater manipulators are in general much slower devices than industrial robot arms, so the duration of this segment may be significantly longer. The question may arise as to why the robot arm has to attain the desired pose set by the PBVS algorithm before it sends back its joint position required for the subsequent control loop. For the developed algorithm to work correctly, it is of great importance that joint position reading occurs at the same time as the camera image acquisition. The reason is that the computation of the desired manipulator pose is performed based on the combination of current target pose, acquired by the camera, and current manipulator pose, acquired through joint position sensor readings. If the acquisition does not take place at the same
Simulation and experimental test results

Table 5.3 Partitioned execution time of the close-loop PBVS algorithm with look and move structure and numerical inverse kinematics implementation

<table>
<thead>
<tr>
<th>Iter</th>
<th>1</th>
<th>2</th>
<th>3</th>
<th>4</th>
<th>5</th>
<th>6</th>
<th>7</th>
<th>8</th>
<th>9</th>
<th>AVG</th>
<th>MIN</th>
<th>MAX</th>
<th>SUM</th>
</tr>
</thead>
<tbody>
<tr>
<td>Cam (ms)</td>
<td>78</td>
<td>76</td>
<td>74</td>
<td>82</td>
<td>89</td>
<td>73</td>
<td>90</td>
<td>71</td>
<td>92</td>
<td>81</td>
<td>71</td>
<td>92</td>
<td>727</td>
</tr>
<tr>
<td>Ikine (ms)</td>
<td>162</td>
<td>60</td>
<td>57</td>
<td>57</td>
<td>57</td>
<td>57</td>
<td>57</td>
<td>57</td>
<td>57</td>
<td>57</td>
<td>69</td>
<td>57</td>
<td>162</td>
</tr>
<tr>
<td>Move (ms)</td>
<td>350</td>
<td>220</td>
<td>187</td>
<td>258</td>
<td>195</td>
<td>148</td>
<td>150</td>
<td>130</td>
<td>150</td>
<td>199</td>
<td>130</td>
<td>350</td>
<td>1789</td>
</tr>
<tr>
<td>Total (ms)</td>
<td>590</td>
<td>356</td>
<td>319</td>
<td>398</td>
<td>341</td>
<td>278</td>
<td>298</td>
<td>259</td>
<td>300</td>
<td>349</td>
<td>259</td>
<td>300</td>
<td>3140</td>
</tr>
</tbody>
</table>

time and the robot is in motion, it is for certain that these two readings will not correspond to each other. The synchronisation problem is of a technical nature. Namely, Staubli TX60 does not have a continuous stream of joint positions available. Moreover, it does not allow actual joint position reading at all, but only the last joint positions the robot arm was commanded to attain. The situation is similar with Schilling underwater manipulators which also do not have a continuous stream of joint position data available, but require a serial communication request message to acquire joint position readings. Therefore, since it is impossible to synchronise the acquisition of the camera image and joint angles using the available hardware, a simple workaround is to perform the image acquisition at the very beginning of the control loop, immediately after receiving the information that the manipulator has reached the pose specified in the previous control loop.

Detailed analysis of a single PBVS experiment

The results of one specific experiment test case (PP = 60% and IP = 14) are presented here and analysed in greater detail. For this combination of parameters, it took 3.5 ms and 10 iterations of the PBVS algorithm for the robot arm to complete the manipulation task. Fig. 5.30 shows the distinct steps in the execution of the task, and Fig. 5.31 the same steps but from the camera point of view. Duration of the individual segments throughout the experiment is given in Table 5.3, while Fig. 5.32 shows this data plotted against time for comparative analysis. It can be noticed that the sum of durations of individual stages (3.14 s) differs from the total time required for the task execution (3.5 s). The reason is the absence of the last stage in the sum where the end-effector moves in a straight line for 10 mm without visual feedback to establish contact between the vacuum gripper suction cups and the target. The duration of each visual servoing iteration varies between 259 ms and 590 ms with the average value of 349 ms. The robot motion takes most of the time, especially in the initial iterations, since the distance the manipulator has to traverse reduces exponentially with iterations (Fig. 5.28). Figs. 5.33 and 5.34 show the position and orientation errors during the experiment; the vertical dashed line depicts the moment the 2nd approach phase triggers. The position error is formed by extracting
Fig. 5.30 Stages of the manipulation task execution utilising the close-loop PBVS algorithm with dynamic *look and move* structure
Simulation and experimental test results

Fig. 5.31 Camera view of the manipulation task execution stages utilising the close-loop PBVS algorithm with dynamic *look and move* structure
5.3 Visual-based motion control

![Partitioned execution time of the close-loop PBVS algorithm with look and move structure and numerical inverse kinematics implementation](image)

Fig. 5.32 Partitioned execution time of the close-loop PBVS algorithm with *look and move* structure and numerical inverse kinematics implementation

![Position error during the PBVS experiment (PP = 60% and IP = 14)](image)

Fig. 5.33 Position error during the PBVS experiment (PP = 60% and IP = 14)

![Orientation error during the PBVS experiment (PP = 60% and IP = 14)](image)

Fig. 5.34 Orientation error during the PBVS experiment (PP = 60% and IP = 14)

the translation vector from the homogeneous transformation $H_{\text{error}}$, see equation (4.84), and the orientation error by extracting the rotation matrix from the same homogeneous transformation, and converting it into the axis-angle representation (Siciliano et al., 2009, p. 52). Fig. 5.35 represents the estimated target position and the way the TCP converges towards it. Fig. 5.36 depicts the real and the estimated position of the target; the former is computed from joint position readings of Robot 1, which has the target on its end-effector, and the latter is acquired via camera pose estimation software. As it is fixed on the end-effector of Robot 1, the real position of the target is computed based on the forward kinematics equation and the robot’s joint position readings, and expressed in the Robot 2 base frame based on the measured relative pose between the robots’ bases. The estimated target position is obtained
Simulation and experimental test results

Fig. 5.35 Robot 2 Virtual TCP motion during the PBVS experiment (PP = 60% and IP = 14)

Fig. 5.36 Target position during the PBVS experiment (PP = 60% and IP = 14)

Fig. 5.37 Target position error during the PBVS experiment (PP = 60% and IP = 14)
5.3 Visual-based motion control

Fig. 5.38 Joint positions during the PBVS experiment (PP = 60% and IP = 14)

by the machine vision pose estimation software and expressed in the Robot 2 base frame through the forward kinematics equation and Robot 2 joint position readings. Fig. 5.37 shows the error between the real and the estimated target position plotted against time during the experiment. It can be noticed that the estimation error is not constant, i.e. it decreases with the distance to the target, and has a value of 21 mm at the end of the experiment, which is a significant error. Sometime after the experiments, it was determined that estimation error was caused in no small extent by the inconsistency between the dimensions of the real camera mount and its model, because the mount was deformed, bent downwards from the weight of the camera. This did not affect the performance of the PBVS algorithm considerably because the desired pose for visual servoing was acquired with the same error as well. Finally, Fig. 5.38 shows the joint positions which are the output of the developed PBVS software and the reference for the low-level motion controller.

Experiments with different initial positions

The following set of experiments was conducted to validate the performance of the developed closed-loop PBVS algorithm with dynamic look and move structure subject to different initial position of the camera relative to the target. The addressed PBVS algorithm was employed with the following parameters: PP = 60% and IP = 9. The robot arm equipped with a camera began the experiments positioned in different locations, such that the fiducial marker appeared at a different outer limit of the camera image at the beginning of each experiment (i.e. top, bottom, left, right, top-right, top-left, bottom-left, bottom-right). All experiments were accomplished successfully with the end-effector converging to the target, as can be seen from Fig. 5.39.
Simulation and experimental test results

Experiments with different initial orientations

In the previous set of experiments described above, Robot 2 started the visual servoing task from different initial positions, but with relatively small initial orientation differences, i.e. the camera was upright with the image plane more or less parallel with the fiducial marker. Therefore, another set of experiments was conducted to test the performance of the developed closed-loop PBVS algorithm with dynamic look and move structure subject to different initial orientation of the camera relative to the target. The manipulator equipped with a camera began each experiment with the end-effector positioned at the same location, but rotated around the TCP z-axis for one of the following angles: 12°, 55°, 88°, and 120°. The PBVS algorithm employed in the experiments had the following parameters: PP = 60% and IP = 14. Tests with the initial orientation error of 88° and lower were performed successfully, as can be seen from Fig. 5.40. For safety reasons, to be able to prevent potential collisions, the robot arm speed was reduced, which is the reason for a prolonged task execution time of 12 s. The first problem in accomplishing the visual servoing task appeared in the experiment with the initial orientation error of 120°. Namely, during the first iteration of the 1st approach phase, the fiducial marker left the field of view of the camera. This triggered the search phase, during which the target was detected again; however, the orientation error even increased. The 1st approach phase restarted, but the target again left the field of view of the camera. Fig. 5.41 shows the manipulator at the beginning of the experiment, after the first iteration of the 1st approach phase, and after the first search phase. After a couple of such repetitions, the robot arm managed to accomplish the 1st approach phase, and subsequently the task itself. However, the task execution took significantly longer time; 300 s was required instead of 12 s, as can be seen from Fig. 5.42. The reason the fiducial marker kept leaving the field of view of the camera is now explained. In each PBVS iteration,

![Fig. 5.39 Norm of position error during the visual servoing task with different initial positions (IP1,...,IP8)](image-url)
5.3 Visual-based motion control

Fig. 5.40 Visual servoing experiments with different initial orientations (IO1, IO2, IO3)

Fig. 5.41 Unsuitable camera poses in visual servoing experiment with different initial orientations (IO4)

Fig. 5.42 Visual servoing experiment with different initial orientations (IO4)
the end-effector traverses a portion of the path towards the target defined by the PP parameter, translating along the line that connects the current and desired end-effector Cartesian points. The rotational part of the end-effector motion is determined by the spherical linear interpolation method for quaternions (Dam et al., 1998). The resulting straight line motion trajectory is generated for the end-effector. However, since the z-axis of the camera frame is not collinear with the z-axis of the end-effector frame, i.e. there are offsets in x and y coordinates between the two frames, the camera trajectory resulting from the interpolation technique mentioned above is not a Cartesian space straight line. This can be seen from Fig. 5.43 which shows a trajectory obtained by the same interpolation method between two coordinate frames. The end-effector trajectory, represented by green-blue colour is a straight line, and the camera trajectory, represented by yellow-pink colour is a curved line. Returning to the experiment, what happens in the 1st approach phase is that the robot performs 60% of the translatory and rotary reference motion combined, which results in fairly unsuitable camera pose for the given initial orientation. A simple solution to prevent this from happening is to change the PP parameter to 100% for the first few iterations of the 1st approach phase. The same experiment was repeated with the described
5.3 Visual-based motion control

Fig. 5.44 Suitable camera poses in visual servoing experiment with different initial orientations (IO4)

![Image of suitable camera poses]

(a) Position error norm  
(b) Orientation error

Fig. 5.45 Visual servoing experiment with different initial orientations (IO4)

change of parameter settings, which led to a successful and improved execution of the manipulation task. Fig. 5.41 shows the manipulator at the beginning of the experiment, and after the first two iterations of the 1st approach phase, while Fig. 5.45 shows the position and orientation errors during the experiment, which again lasted around 12 s. Also, the IP parameter was set to 40 for the first iteration, which provided increased positioning accuracy at the price of increased computational load.

This experimental test shows the importance of selecting the appropriate parameters of the PBVS algorithm for achieving successful and fast task execution without losing the target from the camera field of view. Some tips can be derived, related to the initial steps in the development of an algorithm capable of actively modifying PBVS parameter values during the visual servoing task based on relative position and orientation between the camera and the target. Another possible improvement is to separate the translatory from the rotary motion within the reference trajectory, i.e. have two PP parameters, one for the position and one for the orientation. In that way, the camera frame could be aligned with the target frame rapidly, while keeping a different, slower approach pace. Additionally, the PBVS algorithm can be improved if the reference trajectory derived by the interpolation method, described above, is generated for the camera instead of the end-effector. The origin of the camera frame would then move in a straight line, which would reduce the possibility of losing the target. Depending on the relative pose between the camera, end-effector, and
Simulation and experimental test results

target, finding the end-effector trajectory that corresponds to the desired straight-line camera trajectory may not be possible in all conditions due to the physical limitations of the manipulator; regardless, it is worth further research.

Pick and place experiment

The described closed-loop PBVS algorithm with dynamic look and move structure was applied for a pick and place task. The target object for manipulation (a small box) is velcroed on a plastic holder which has a fiducial marker attached to its surface. The target holder is mounted on the end-effector of Robot 1. A vacuum gripper is used to pick the box up and place it back on the holder. The 1st approach phase, 2nd approach phase, and grip phase are identical to those of the previously described experiments. In the following phase, the manipulator pulls back with the object held by the gripper to a predefined pose relative to the target holder. This phase also utilises the PBVS algorithm. After this, the 1st approach phase, 2nd approach phase, and grip phase are repeated to return the target object to its place. Finally, the vacuum gripper deactivates, and the robot pulls back empty-handed. The described pick and place experiment was conducted using the PBVS algorithm with two combinations of parameter values (PP = 10% and IP = 14, and PP = 80% and IP = 14), both of which provided satisfactory results. Fig. 5.46 depicts the separate stages of the experiment, and Figs. 5.47–5.50 the position and orientation errors and the convergence of the virtual TCP towards the target for experiments with both parameter combinations.

This subsection presented experimental validation of the developed PBVS algorithm using industrial robot arms in laboratory conditions. Successful execution of the pick and place task demonstrates the possibility of addressing typical subsea ROV manipulation tasks (sample placing in a sample bin, tool acquisition and retrieval) utilising the proposed visual servoing control scheme. The designed PBVS algorithm is capable of generating reference kinematics parameters for motion in joints space in a form suitable for application to underwater manipulator systems. Integration of the developed control system on real underwater manipulators and further experimental testing is addressed in the following subsection.

5.3.4 Experiments of the PBVS algorithm with an underwater manipulator addressing a stationary target

The developed PBVS algorithm for hydraulic subsea ROV manipulators is described in section 4.4.1 and validated with industrial robot arms in section 5.3.3. Having
Fig. 5.46 Stages of pick and place manipulation task utilising the developed PBVS algorithm
Simulation and experimental test results

(a) Position error

(b) Orientation error

Fig. 5.47 Pick and place task utilising PBVS algorithm (PP = 10% and IP = 14)

Fig. 5.48 Virtual TCP motion in pick and place task utilising PBVS algorithm (PP = 10% and IP = 14)

(a) Position error

(b) Orientation error

Fig. 5.49 Pick and place task utilising PBVS algorithm (PP = 80% and IP = 14)

Fig. 5.50 Virtual TCP motion in pick and place task utilising PBVS algorithm (PP = 80% and IP = 14)
proved to be satisfactory, the author modified the visual servoing system for use on Schilling Titan 2 manipulator with a wrist-mounted Point Grey Blackfly camera. In this configuration, it was tested both in dry laboratory experiments as well as in subsea field trials in a flooded quarry in Portroe (Ireland) with the manipulator installed on the Irish Marine Institute owned commercial work-class ROV Holland I. This subsection summarises the experimental results, some of which are already published by the author, see Appendix G.

The developed visual servoing control algorithms are encapsulated in the pilot control software described in section 5.2.3, which replaced the Titan 2 MCU. The pilot control software runs on a dedicated topside PC and communicates over an available serial communication channel with the Titan 2 SCU, located on the ROV. The algorithms integrated into the software are purely kinematic, i.e. they provide kinematic output parameters which are effectively the input reference signals for the existing manipulator SCU. This implies that the robotic manipulator is treated as an ideal positioning device as the author believes that off-the-shelf manipulators such as Schilling Titan and Orion series have sufficient capabilities to utilise this approach. The designed fully automatic vision-guided control scheme is utilised in multiple tests addressing different typical subsea intervention tasks to cover scenarios 2 and 3 described in section 4.2. The addressed experimental setup involves various test panels equipped with fiducial markers and standard T-bar handles which simulate target devices for different intervention tasks that are performed with ROV manipulators. A T-bar handle is the most common standardised mechanical interface between manipulator jaws and subsea tooling equipment and infrastructure (13628-8:2002, 2002). Some experiments include a supporting Schilling Orion 7P manipulator for cooperative manipulation tests. The automated manipulation task experiments which are carried out are:

1. **PBVS validation** — in which the manipulator automatically moves in the desired vicinity of the target and stays there for a designated amount of time.

2. **Grabbing a tool** — in which the manipulator automatically locates the T-bar handle representing a subsea tool, approaches it, grabs it, and pulls it out of the tool holder.

3. **Turning a valve** — in which the manipulator automatically locates, grabs, and rotates the T-bar handle representing a valve handle.

4. **Plugging a connector** — in which the manipulator automatically plugs the T-bar which is already held in its jaws in a hole, simulating the process of plugging a subsea connector.
All tasks described above were performed in dry laboratory experiments in controlled conditions with a stationary target. The same tasks except for the one that simulates connector plugging were also carried out in the underwater environment. Additionally, the PBVS algorithm developed for stationary targets was tested subsea with the target in motion. All visual servoing manipulation tasks were tested with various parameter settings and with different initial target locations. The base of the manipulator was stationary in all experiments. In subsea trials, the ROV was parked on the seabed. Marine work-class ROVs often use secondary grabber manipulator to fix the ROV to the underwater structure on which the job is to be carried out. This case is substantially similar with the case where the ROV is parked on the sea floor as there is no relative motion between the target and the base of the manipulator.

**Laboratory experiment of the Plugging a connector task**

The results of the laboratory experiment addressing the *Plugging a connector* intervention task are presented as follows. This task is carried out in the following phases:

1. **Preparation phase** — in which the pilot positions the robotic manipulator in a suitable posture for performing the task; a typical deployed/ready posture with the camera pointing away from the ROV.

2. **1st approach phase** — in which the developed PBVS algorithm aligns and positions the camera right in front of the fiducial marker at a predefined distance from it. The threshold value for the position error is relaxed in this phase as the accuracy is not too relevant; the goal is to obtain satisfactory initial conditions for the subsequent phase in a timely manner.

3. **2nd approach phase** — in which the developed PBVS algorithm controls the manipulator to approach the target so that the tip of the T-bar handle that simulates the connector is just outside the hole on the panel that simulates the socket. The threshold values are set to relatively low values to ensure sufficient positioning accuracy which is critical for this phase.

4. **Connection phase** — in which the desired camera to fiducial marker distance, which is a part of the PBVS algorithm control variable, and the actual distance at the beginning of this phase, decreases at a predefined pace to a threshold value, thereby making the camera approach the marker and the T-bar handle to enter the hole.
5. **End phase** — in which the manipulator releases the T-bar handle having successfully plugged the connector and pulls back, moving the end-effector in a straight line retreat without visual feedback.

The described intervention task was performed successfully, as can be seen from Fig. 5.51. Fig. 5.52 shows the position and orientation of the fiducial marker in the camera frame during the experiment; orientation is expressed with angle-axis representation. It can be clearly seen that the **Connection phase** starts after approximately 40 s during which the $x$ and $y$ position coordinates stay close to zero and the $z$ coordinate reduces over time. Fig. 5.53 depicts joint position reference signals that the developed PBVS algorithm generates. Of all addressed manipulation tasks, this one is by far the most challenging. This is because it requires sufficient positioning accuracy not only at one final pose but also throughout the whole **Connection phase**. Unlike with other intervention tasks, difficulties may arise while executing the **Plugging a connector** task because the PBVS algorithm generates reference motion for the manipulator as a sequence of joint positions and the low-level commercial SCU controller does not synchronise the motion of the joints in the desired manner, i.e. Cartesian end-effector velocity control does not exist. However, provided that via points forming the path for T-bar insertion consists of a relatively large number of configurations with a relatively small difference between every two consecutive configurations, and that these reference commands are issued with a relatively slow update rate, the task can be completed successfully. On the contrary, if the reference configurations are sparse or issued too fast one after another, the end-effector motion becomes notably jerky and considerably deviates from the straight line, inevitably causing the T-bar connector to get stuck. Nevertheless, implementing the preventive measures outlined above allows execution of this task automatically in sufficiently short time, i.e. less than 1.5 min.

**Field subsea experiment of the Turning a valve task**

The detailed results of the **Turning a valve** intervention task carried out in subsea field trials is now presented. This task is performed in the following phases:

1. **Preparation phase** — in which the pilot positions the robotic manipulator in a suitable posture for performing the task; a typical deploy posture with the camera pointing away from the ROV.

2. **1st approach phase** — in which the developed PBVS algorithm aligns and positions the camera right in front of the fiducial marker on a predefined distance from it. The threshold value for the position error is relaxed in this
Simulation and experimental test results

Fig. 5.51 Camera view during the execution of the *Plugging a connector* task

Fig. 5.52 Fiducial marker (a) position and (b) orientation during the *Plugging a connector* task experiment
5.3 Visual-based motion control

Fig. 5.53 Joint angle reference positions during the *Plugging a connector* task experiment

phase as the accuracy is not too relevant; the goal is to obtain satisfactory initial conditions for the subsequent phase in a timely manner.

3. **2nd approach phase** — in which the developed PBVS algorithm controls the manipulator to approach the target completely and attain a pose convenient for gripping the T-bar. The threshold values are set to relatively low values to ensure sufficient positioning accuracy which is critical for this phase.

4. **End phase** — in which the manipulator grabs the T-bar handle by closing the jaws, rotates the wrist through $90^\circ$, opens the jaws, and finally pulls back, moving the end-effector in a straight line without visual feedback.

Fig. 5.54a shows images taken by the wrist-mounted camera during the execution of this task; the same images are used for the target pose estimation within the PBVS algorithm. Fig. 5.54b shows the images from the camera mounted on the ROV taken in the same time. The total time required to complete the task was 30 s, where the *1st approach phase* was accomplished in 5 s in a single iteration, *2nd approach phase* in 14 s in 4 iterations, and the *end phase* in 11 s. Fig. 5.55 shows the position and orientation error during the visual servoing phases. The position error is formed by extracting the translation vector from the homogeneous transformation $H_{\text{error}}$, given in equation (4.84), and the orientation error by extracting the rotation matrix from the same homogeneous transformation, and converting it into the axis-angle representation (Siciliano et al., 2009, p. 52). The position error threshold in the *2nd approach phase* was set to 15 mm for this experiment. Fig. 5.56 represents the estimated target pose and TCP converging towards it. Prior to the execution of the task, joints were calibrated using the developed procedure described in section 5.2.4. Even though this procedure did reduce the error between the joint positions commands which are the output of the PBVS algorithm and the joint positions sensed by the resolvers, certain differences were still present as can
Simulation and experimental test results

![Camera views](a) Wrist camera  (b) ROV camera

*Fig. 5.54 Camera view during the execution of the *Turning a valve* intervention task*

![Error vectors](a) Position error  (b) Orientation Error

*Fig. 5.55 Error vector during the execution of the *Turning a valve* intervention task*

be seen from Fig. 5.57a. Fig. 5.57b shows the effect of such position error in the joints on the end-effector position error, constructed via forward kinematics. The intervention task was completed successfully with satisfactory performance despite the notable influence of the absolute inaccuracy of the joint controller on Cartesian space error variable. One might notice that the state feedback is sparse and hence that the experiment consists of very few data points with noticeably low control update rate. This is due to the limitations of the commercial Schilling manipulator SCU. Nonetheless, the task was performed with satisfactory performance. Fig. 5.58 shows the partitioned execution time of the PBVS algorithm during the experiment. Each PBVS control loop incorporates four distinct segments. The first one is the pose estimation segment (T1) which encapsulates image acquisition, processing, and planar homography computation and which took approximately 72 ms. The
5.3 Visual-based motion control

Fig. 5.56 Convergence of the TCP towards the target during the execution of the *Turning a valve* intervention task

![Graph showing TCP convergence](image)

Fig. 5.57 End-effector position error (b) as a consequence of the joint position error (a) during the execution of the *Turning a valve* intervention task

![Graphs showing joint and end-effector errors](image)

The following segment includes the derivation of coordinate transformations and inverse kinematics numerical computation (T2), the duration of which is negligible compared to the other ones. It is worth noting that the computation of the numerical solution for inverse kinematics implemented in LabVIEW takes enormously less time compared to earlier versions of the software implemented in MATLAB for the same number of iterations (compiled languages are faster than scripting languages). The subsequent segment includes the acquisition of angular joint position readings (T3). As is already mentioned, Titan 2 low-level motion controller does not support a continuous stream of angular joint position data from the sensors, and to acquire this data it is necessary to issue a serial communication request and await a reply, which takes more than 100 ms. The last segment (T4) incorporates another pair of serial communication messages, a command to move to the desired position and a confirmation that the command message is received, and the time it takes for the manipulator to attain the desired pose. The duration of this segment is variable, depending on the distance the manipulator’s end-effector is required to traverse. The existing commercial...
Simulation and experimental test results

control system of Schilling Titan 2 underwater manipulator has significant limitations that make achieving outstanding PBVS algorithm performance difficult. The first one is the requirement to exchange four serial communication messages in each PBVS control loop, and the second, the impossibility to continuously read position sensor data and broadcast position commands. Nonetheless, despite these technical shortcomings, the developed visual control approach works sufficiently well as the tasks can be accomplished automatically and still in less time than it might take for a human pilot to execute the same tasks.

Field subsea experiments of the PBVS validation task with different initial positions

The developed visual servoing algorithm is evaluated for multiple initial positions by performing the PBVS validation task. This task has only three phases which are the same as the ones from the Turning a valve task. The only difference is that the
PBVS algorithm does not stop if the position error drops below a specified threshold value but after a certain predefined time from the 2nd approach phase. Fig. 5.59 shows the resulting position error for six different experiments.

**Field subsea experiment of the Grabbing a tool task**

The Grabbing a tool task was also tested multiple times with satisfactory results. This task is performed in the same phases as is the Turning a valve task, with the exception in the End phase in which the jaw rotation and opening are omitted, i.e. the manipulator pulls back without visual feedback right after grabbing the T-bar handle. Fig. 5.60 shows the execution of this intervention task. It can be seen that the experiment started with the absence of the target in the field of view of the camera. For this reason, the manipulator entered the search phase and after few iterations detected the fiducial marker and located the target. This enabled the task continuation and eventually successful accomplishment.

**Field subsea experiments of the PBVS validation task with the target in motion**

Additionally, the visual servoing control scheme developed for stationary targets was tested experimentally in performing the PBVS validation task with the target in motion. The presented experiment represents the first attempt at addressing the target in motion, and the obtained results lead to further development of the PBVS algorithm to enhance the performance. The test panel with the fiducial marker was mounted on the second underwater manipulator, Schilling Orion 7P. The ROV pilot teleoperated this manipulator, continuously moving its end-effector up to 100 mm up and down, forth and back with relatively slow speed (see Fig. 5.61). The primary manipulator for visual servoing utilised the developed PBVS algorithm with the goal

![Graph](image-url)
Simulation and experimental test results

Fig. 5.60 Camera view during the execution of the *Grabbing a tool* task

to see how well the manipulator can follow the moving target and how fast it can respond to its movement. The experiment lasted 70 s during which the manipulator managed to track the target, although with noticeable delays between the movements. From Fig. 5.62, it can be seen that every time the target moves the position error increases but eventually converges to zero after approximately 2 s. This tracking delay can also be noticed from Fig. 5.63 which presents the estimated target pose and the TCP converging towards it.

The intervention task *Turning a valve* with the stationary target was also successfully performed using two manipulators. During the experiment, Orion 7P manipulator was stationary holding the target while Titan 2 manipulator utilised the PBVS algorithm to perform the given task automatically. This experiment validated the possibility of implementing simple visual guided cooperative tasks using two subsea manipulators.
Concluding remarks for section 5.3.4

The laboratory experiments and subsea field trials presented in this section demonstrated a successful adoption and modification of algorithms common for industrial robotics for the development of a visual servoing control scheme for existing commercial subsea hydraulic manipulators for ROVs. The achieved experimental results, some of which are already published (see Appendix G), validated the field-tested control solution that works with the global fleet of industry standard commercial work-class ROV systems that employ Schilling Titan and Orion series manipulators as a software upgrade rather than a hardware replacement for the first time. Encouraging results acquired through unique real-world underwater experiments demonstrated the ability of the developed PBVS algorithm. Viewed from an industrial robotics perspective, the proposed solution may be somewhat simple as it only considers kinematics; however, successful field trials prove that it is an entirely suitable approach and represents a very significant advancement in commercial subsea manipulator capabilities. The presented control system is capable of replicating, entirely automatically, what an ROV pilot does by traditional means of teleoperation for typical tasks addressing stationary targets, using unmodified ROV manipulator systems that are industry standard. Using the developed control system, the author believes that ROV pilots would be able to perform typical underwater manipulation
Simulation and experimental test results

Fig. 5.62 Position error vector during the execution of the PBVS validation task with the target in motion

Fig. 5.63 Convergence of the TCP towards the moving target during the execution of the PBVS validation task

tasks in a supervisory role with greater ease, faster, and with a reduced cognitive load. This could provide significant cost savings for subsea inspection and intervention operations and significantly reduce pilot fatigue and associated errors. The author believes that excellent initial results are achieved with very promising applications and potential for uptake in the field of semi-automated IRM of oil and gas, and MRE installations.

5.3.5 Motion analysis and simulations of OC3-Hywind floating wind turbine for IRM assessment

The need for ROV operations in offshore oil and gas industry is typically on or near the seafloor where there is no heave motion. However, this research is concerned about the emerging MRE sector where more and more devices will be on or near the surface and in the splash zone, first 20 m - 40 m depth, in which there is significant
motion disturbance present in the water column that will perturb motion in 6 DOF of MRE devices. Understanding the motion of these devices in the addressed challenging conditions is a critical prerequisite for developing state-of-the-art technology for IRM operations in the MRE industry. To that end, this subsection describes the motion analysis study for the OC3-Hywind floating wind turbine, details the modelling methods used and presents simulation results. Hywind floating wind turbine is the world leading floating wind turbine system. Such systems show an immediate requirement for subsea IRM tasks to be performed on floating structures. The OC3-Hywind floating platform represents a 130 m long spar buoy with a draft of 120 m (Fig. 5.64). The submerged part of the floating platform consists of two cylindrical parts of different diameters (9.4 m and 6.5 m) with a tapered region in between. The distance from the sea water level to the top of the taper is 4 m and the distance to the bottom of the taper is 12 m. The spar buoy has three catenary mooring lines installed at an angle of 120° between each two. In the modelling setup, the distance from the sea water level to the mooring fairleads is 70 m and to the anchors, which is the total water depth, is 320 m. Equations of motion are derived for the reference frame whose origin lies at the point of the intersection of the platform centreline with the sea water level (Fig. 5.64). The $x$-axis of this frame is parallel to the nacelle axis and together with the $y$-axis forms a plane coincident with the sea surface. The $z$-axis is oriented upwards along the platform centreline. The system has six DOFs: three translations, one along each axis, i.e. surge ($x$), sway ($y$), and heave ($z$), and three rotations around the same set of axes, i.e. roll ($\theta_x$), pitch ($\theta_y$), and yaw ($\theta_z$). Equations of motion are derived based on the following simplifying assumptions:

- The floating platform is considered to be a rigid body.
- Motion is induced by sea waves only, not including effects of wind and sea currents.
- Sea waves are regular, harmonic sinusoidal waves defined by their amplitude, frequency and length.
- Sea waves propagate in the $x$ direction.
- The system is linear, which means that the motion response of the platform induced by regular sinusoidal waves is also sinusoidal but with the difference in amplitude and phase.
Simulation and experimental test results

- Second-order wave drift forces are neglected, which means that the system oscillates about the equilibrium position—undisplaced platform floating in the calm sea.

Si et al. (2013) reported a more in-depth OC3-Hywind modelling that includes flexibility of the structure, and (Ramachandran et al., 2013) modelling that includes wind effects. Additional literature that covers irregular waves and second-order wave drift forces can be found in (Journée and Massie, 2001, pg.5-29) and (Journée and Massie, 2001, pg.6-27), respectively. However, for the purpose of this research, very accurate modelling is not necessary. The goal is to approximately model and physically simulate the motion of a floating structure and enable executing feasibility tests of the visual servoing algorithms for subsea manipulators. Based on Newton’s second law, the six equations of motion are given by:

$$\sum_{j=1}^{6} m_{i,j} \cdot \ddot{x}_j = \sum_{j=1}^{6} \left\{ -a_{i,j}(\omega) \cdot \dot{x}_j(\omega,t) - b_{i,j}(\omega) \cdot \dot{x}_j(\omega,t) - c_{i,j}(\omega) \cdot x_j(\omega,t) \right\} + F_{\omega a_i}(\omega) \cdot \cos(\omega t + \epsilon_i(\omega))$$  

(5.5)

The term on the left side of the equation represents the sum of forces/moments expressed as a product of solid mass and inertia coefficients ($m_{i,j}$) and body acceleration ($\ddot{x}_j$). The first two terms on the right side of the equation are hydrodynamic forces and moments expressed in terms with hydrodynamic added mass ($a_{i,j}$) and damping coefficients ($b_{i,j}$). The third term represents hydrostatic restoring forces.
and moments expressed in a term with a spring coefficient \((c_{i,j})\), and the last term represents wave excitation forces and moments \((F_{\omega a_i})\) as a function of incident wave frequency \((\omega)\) with a phase shift \((\varepsilon_i)\). After some reordering, these equations of motion can be expressed in the frequency domain:

\[
\sum_{j=1}^{6} \left\{ (m_{i,j} + a_{i,j}(\omega)) \cdot \ddot{x}_j(\omega,t) + b_{i,j}(\omega) \cdot \dot{x}_j(\omega,t) + c_{i,j}(\omega) \cdot x_j(\omega,t) \right\} = F_{\omega a_i}(\omega) \cdot \cos(\omega t + \varepsilon_i(\omega))
\]

Furthermore, these equations of motion can be expressed in the matrix form as follows:

\[
(M + A(\omega)) \ddot{X} + B(\omega) \dot{X} - C \bar{X} = \bar{F}(\omega)
\]

Using Euler’s complex analysis formula and reordering the terms yields:

\[
-\omega^2(M + A(\omega)) \bar{X} e^{j\omega t} + jB(\omega) \bar{X} e^{j\omega t} - C \bar{X} e^{j\omega t} = \bar{F} e^{j\omega t}
\]

\[
(-\omega^2(M + A(\omega)) + jB(\omega) - C) \bar{X} = \bar{F}
\]

\[
\bar{X} = (-\omega^2(M + A(\omega)) + jB(\omega) - C)^{-1} \bar{F}
\]

\[
\bar{X} = H \bar{F}
\]

where

\[
\bar{F} = |\bar{F}(\omega)| e^{j\varphi(\omega)} = |\bar{F}(\omega)| \cos(\varphi(\omega)) + j|\bar{F}(\omega)| \sin(\varphi(\omega))
\]

represents wave excitation forces and moments normalised with the amplitude of the inducing waves, \(H\) represents the force RAO, which is the force-to-motion transfer function, and \(\bar{X}\) represents the motion RAO (Fossen, 2011).

The U.S. National Wind Technology Center (NREL) graciously supplied the dynamic parameters used within their OC3-Hywind project. Having these parameters enabled derivation of motion RAOs, along with the phase shift graphs, which are described later. The dynamic parameters include: the aggregate mass inertia matrix of the total system (platform, ballast, tower, nacelle and rotor, not including the moorings), calculated using the MSC ADAMS software; the hydrostatic restoring matrix calculated by the WAMIT software; the hydrodynamic added mass and damping calculated by the WAMIT software as a solution to the radiation problem; the wave excitation forces calculated using the WAMIT software as a solution to the diffraction problem, and the moorings restoring matrix provided by Statoil. A detailed explanation of the modelling methods NREL implemented for obtaining these dynamic parameters can be found in (Jonkman, 2010). To complete the modelling the author derived the gravity restoring matrix as this is not included in the WAMIT hydrostatic restoring matrix. The mathematical model of the OC3-Hywind
floating platform is developed using the MATLAB software, and RAO and phase shift graphs are computed for each DOF. Figs. 5.65–5.70 show RAOs and phase angles for the dominant DOFs—surge, heave and pitch. RAOs for other DOFs are of few orders of magnitude smaller and are, therefore, neglected. To be able to interpret and extract useful information from RAO graphs it is necessary to know what periods and significant heights of sea waves occur in nature. Not all combinations of periods and heights are possible, and some occur more often than others. For this reason, the author analysed the sea wave measurements obtained by the M3 Irish weather buoy, which are available online on the Irish Marine Institute’s website (Marine Institute, 2017b). The M3 buoy is located in the Atlantic Ocean 389 km off the Cork coast. The water depth on its location is between 100 m and 300 m (Marine Institute, 2017a). As this depth range is suitable for the Hywind floating turbine, the M3 weather buoy measurements are appropriate for the analysis. The author analysed annual measurements acquired by the M3 weather buoy for the period between May 2016 and May 2017. Table 5.4 depicts the results that the percentage occurrence of different combinations of significant wave heights and periods. The table is divided into regions according to the Douglas Sea Scale (Journée and Massie, 2001, pg.5-54). Based on the boundary wave periods for each sea state and the motion RAO graphs it is possible to characterise the floating wind turbine motion for any sea state. For explanation purposes, surge RAO is analysed as follows. Let us assume that the sea state is of degree 4 with a significant wave height of 2.5 m, that is amplitude ($\zeta_a$) of 1.25 m, and a period of 6 s, that is frequency of 0.16 Hz. Fig. 5.65 shows that for the given wave frequency the surge RAO is 0.2. The motion RAO represents the ratio between the amplitude of the wave elevation and the amplitude of the motion in the specified DOF for a given wave frequency. This means that in the given wave conditions the platform surges from the equilibrium position in both directions for:

$$x_a = RAO_{surge} \cdot \zeta_a = 0.2 \cdot 1.25 \text{ m} = 0.25 \text{ m}$$

(5.13)

Additionally, Fig. 5.66 shows that the surge motion lags the wave elevation by 100°. The same approach can be used to determine heave and pitch motion. RAO and phase shift graphs thus provide information for the origin point (O) of the floating wind turbine. Using simple geometrical transformations, it is possible to find the motion of any other point on the floating wind turbine (Journée and Massie, 2001, pg.8-22). A particular point of interest for motion analysis is the location where the mooring lines are fastened to the spar buoy as the moorings commonly require a visual inspection for structural degradation (McLeod et al., 2012). For reference, this point is labelled as the mooring fairlead point (M). Another point of interest is the bottom end point (B) of the spar buoy. Being able to derive the motion of these
three points (O, M and B) is beneficial for understanding the nature of motion of the structure as a whole. Therefore, the author conducted motion simulations for these points under the same sea conditions; the presented simulation assumes 2.5 m wave height and 6 s wave period. Fig. 5.71 shows the resulting motion of the floating platform origin point (O), bottom end point (B) and mooring fairlead point (M) as plots of position versus time, while Fig. 5.72 shows the same trajectories in the $xz$-plane. These figures show that the heave amplitude is similar for all three points ($\sim 0.02$ m), while the surge amplitude varies significantly: 0.23 m for the origin point; 0.07 m for the mooring fairlead point, and 0.05 m for the bottom end point.
**Simulation and experimental test results**

Table 5.4 Irish Weather Buoy M3: Sea State occurrence (%) for the period between May 2016 and May 2017

<table>
<thead>
<tr>
<th>Sea State</th>
<th>Wave Period [s]</th>
<th>Wave Height [m]</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td>3.5-4.5</td>
<td>4.5-5.5</td>
</tr>
<tr>
<td>7</td>
<td>0.00</td>
<td>0.00</td>
</tr>
<tr>
<td>7.25-7.75</td>
<td>0.00</td>
<td>0.00</td>
</tr>
<tr>
<td>7.75-8.25</td>
<td>0.00</td>
<td>0.00</td>
</tr>
<tr>
<td>8.25-8.75</td>
<td>0.00</td>
<td>0.00</td>
</tr>
<tr>
<td>8.75-9.25</td>
<td>0.00</td>
<td>0.00</td>
</tr>
<tr>
<td>9.25-9.75</td>
<td>0.00</td>
<td>0.00</td>
</tr>
<tr>
<td>9.75-10.25</td>
<td>0.00</td>
<td>0.00</td>
</tr>
<tr>
<td>10.25-10.75</td>
<td>0.00</td>
<td>0.00</td>
</tr>
<tr>
<td>10.75-11.25</td>
<td>0.00</td>
<td>0.00</td>
</tr>
<tr>
<td>11.25-11.75</td>
<td>0.00</td>
<td>0.00</td>
</tr>
</tbody>
</table>

Pitch motion of the floating platform heavily influences the surge motion of any point on the platform. However, the phase shift between surge and pitch motion is such that the two cancel each other to a greater extent the deeper the point of interest is on the platform. As a result, surge motion dominates near the surface while its influence significantly decreases towards the bottom of the floating platform. Consequently, the bottom end point of the floating platform has the least motion of the three point considered. This is the case in any sea condition. The floating platform motion was simulated multiple times to cover essential sea conditions depicted in Table 5.4. For each sea state (3–7), two boundary cases were addressed: a combination of the minimum period and the maximum wave height, and a combination of the maximum period and the maximum wave height. Motion in sea state 2 is minimal, so this case was disregarded. The mooring fairlead point (M) on the floating platform was selected for the simulations. Figs. 5.73–5.82 show the resulting position, velocity, and acceleration graphs plotted versus time. In the context of all plausible sea conditions (3–7), the peak-to-peak motion amplitude varies from 0.004 m to 0.8 m for the heave DOF, and from 0.04 m to 1.6 m for the surge DOF. The peak of the sinusoidal velocity goes from 15 mm s$^{-1}$ to 500 mm s$^{-1}$ for the surge motion, and from 2.5 mm s$^{-1}$ to 250 mm s$^{-1}$ for the heave motion, while the peak of the acceleration goes from 25 mm s$^{-2}$ to 300 mm s$^{-2}$ for the surge motion, and from 4 mm s$^{-2}$ to 150 mm s$^{-2}$ for the heave motion. Per expectations, higher sea waves
5.3 Visual-based motion control

Fig. 5.71 OC3-Hywind (SS = 4, WP = 6 s, WH = 2.5 m): Position of the (a) origin point (O), (b) mooring fairlead point (M), and (c) bottom point (B)

Fig. 5.72 OC3-Hywind (SS = 4, WP = 6 s, WH = 2.5 m): Trajectory in xy-plane of the (a) origin point (O), (b) mooring fairlead point (M), and (c) bottom point (B)

affect the motion of the floating platform more significantly than lower waves. The same can be said for the waves with higher periods compared to the ones with lower periods. An interesting observation is the significant effect of the sea waves with large periods on inducing motion in the floating platform. For example, sea waves of 1.25 m and 9 s period induce similar motion as sea waves of 4 m and 6 s period. Furthermore, for a given wave height, the induced surge motion varies 3 to 12.5 times depending on the wave period. The effect of sea waves with higher periods on the velocity and acceleration is the same as their effect on the range of motion.

The presented motion analysis assists in comprehending the nature of motion of the MRE device as a whole and thus identifies the typical conditions for performing inspection and intervention operations. Moreover, it serves as a tool for describing the motion of any particular point or area of interest on the floating platform at any given sea conditions. This is useful since it enables determining the conditions of the worst case scenarios for each sea state. The described method can be used to estimate whether sea conditions—real or forecasted—are satisfactory for the inspection. However, this also requires identifying the capabilities of the underwater vehicle-manipulator system that is to perform the IRM operation. This can be done experimentally by applying the motion trajectories that result from the presented motion analysis as reference trajectories for the moving target, and addressing this target with a manipulator. Provided that the vehicle-manipulator system can motion...
Simulation and experimental test results

match the motion of the target, performing operations in given conditions can be considered possible. A more comprehensive study would require understanding the ROV motion due to the waves only if in the free mid-water column and the ROV motion as a result of local strong disturbance washing around the large cylinder platform surging, heaving, pitching, etc. The trajectory required for a manipulator to track a moving target would then be the difference between the target motion and the ROV motion. However, such detailed analysis is left for future research and at this stage it is assumed that the ROV base vehicle can station keep in the vicinity of the target for inspection/intervention.

![Position, Velocity, Acceleration Graphs](image)

Fig. 5.73 OC3-Hywind simulation result (SS = 3, WP = 4 s, WH = 1.25 m): (M) Mooring fairlead point (a) position, (b) velocity, and (c) acceleration

![Position, Velocity, Acceleration Graphs](image)

Fig. 5.74 OC3-Hywind simulation result (SS = 3, WP = 9 s, WH = 1.25 m): (M) Mooring fairlead point (a) position, (b) velocity, and (c) acceleration

5.3.6 Experiments on replication of the OC3-Hywind floating wind turbine motion under the influence of sea waves

This subsection presents the results of experiments in which an industrial robot arm physically simulates the motion of a floating wind turbine, based on the simulation results presented in section 5.3.5. The idea of the experiments is to replicate the motion of a particular point on the OC3-Hywind spar buoy with the end-effector of
5.3 Visual-based motion control

Fig. 5.75 OC3-Hywind simulation result (SS = 4, WP = 4 s, WH = 2.5 m): (M) Mooring fairlead point (a) position, (b) velocity, and (c) acceleration

Fig. 5.76 OC3-Hywind simulation result (SS = 4, WP = 10.5 s, WH = 2.5 m): (M) Mooring fairlead point (a) position, (b) velocity, and (c) acceleration

Fig. 5.77 OC3-Hywind simulation result (SS = 5, WP = 6 s, WH = 4 m): (M) Mooring fairlead point (a) position, (b) velocity, and (c) acceleration
Simulation and experimental test results

Fig. 5.78 OC3-Hywind simulation result (SS = 5, WP = 10.5 s, WH = 4 m): (M) Mooring fairlead point (a) position, (b) velocity, and (c) acceleration

Fig. 5.79 OC3-Hywind simulation result (SS = 6, WP = 8 s, WH = 6 m): (M) Mooring fairlead point (a) position, (b) velocity, and (c) acceleration

Fig. 5.80 OC3-Hywind simulation result (SS = 6, WP = 10.5 s, WH = 6 m): (M) Mooring fairlead point (a) position, (b) velocity, and (c) acceleration
5.3 Visual-based motion control

Fig. 5.81 OC3-Hywind simulation result (SS = 7, WP = 9 s, WH = 8.25 m): (M) Mooring fairlead point (a) position, (b) velocity, and (c) acceleration

Fig. 5.82 OC3-Hywind simulation result (SS = 7, WP = 10.5 s, WH = 8.25 m): (M) Mooring fairlead point (a) position, (b) velocity, and (c) acceleration

the robot arm. This includes matching a trajectory with detailed position, velocity, and acceleration. The experiments employ Staubli TX60 robot arm, and therefore, the software for motion replication is implemented using VAL 3, the standard Staubli robot programming language. The developed software allows a user to specify desired parameters: wave period, motion amplitudes in surge ($x$) and heave ($z$) directions, and phase shift between surge and heave motions. Running the program drives the arm, replicating the motion of the floating platform. To be able to verify the experimental results, a fiducial marker was mounted on the robot arm end-effector and the developed pose estimation software, described in section 5.3.1, was used to track and estimate its motion. The camera filming the motion of the fiducial marker was installed on the second industrial robot arm which was stationary during the experiments. One of the reasons to mount a camera on a robot arm is because it can accurately position the camera at a suitable location, far enough so that the fiducial marker stays in the field of view of the camera during the experiments, but close enough so that it can be detected. The other advantage is that the relative pose between the robot bases ($H_{B2}^{B1}$) is known, i.e. measured from the experimental setup. Based on this pose, the pose of the camera relative to the end-effector carrying it ($H_{CAM}^{EE2}$), the pose of that end-effector relative to its base ($H_{EE2}^{B2}$), and the initial
Simulation and experimental test results

Pose of the end-effector of the robot arm that carries the fiducial marker \( \mathbf{H}_{\text{init}B_1}^{EE_1} \), it is possible to express the estimated motion of the fiducial marker in the reference frame about which it oscillates:

\[
\mathbf{H}_M = (\mathbf{H}_{\text{init}EE_1}^{B_1} )^{-1} \cdot (\mathbf{H}_{B_1}^{B_2} )^{-1} \cdot \mathbf{H}_{EE_1}^{B_2} \cdot \mathbf{H}_{CAM}^{EE_2} \cdot \mathbf{H}_{M}^{CAM} \quad (5.14)
\]

This reference frame is the equivalent frame relative to which the particular point of interest on the floating platform oscillates. Therefore, the experimental results can be compared directly with the desired trajectories acquired from the floating platform motion simulations, described in the section 5.3.5. Multiple experiments were carried out replicating motion of the OC3-Hywind floating wind turbine by implementing these trajectories. However, it was not possible to replicate all trajectories due to the limited robot arm workspace. In total, eight trajectories were addressed representing the motion of the floating platform mooring fairlead point (M) subject to different sea conditions. This includes a pair of trajectories representing boundary cases for each sea state from 3 to 5. The trajectory corresponding to the lower boundary of the sea state 6 is also replicated; however, the motion of the trajectory corresponding to the higher boundary is too large in position for the Staubli robot arm and was omitted. Therefore, a case from the same sea state with the same wave height (6 m) but a lower wave period (9.5 s) replaced the unachievable one. This is also the case with the maximum motion that Staubli robot arm can replicate. Figs. 5.83–5.90 present the resulting target trajectories described with position, velocity and acceleration graphs plotted versus time. The position of the target is directly estimated with the camera system, while velocity and acceleration are derived from the position data. These figures show that the motion matching in position is satisfactory, i.e., despite the discontinuities existing in the velocity and acceleration, it is clear that there exists a sinusoidal rate of change. Therefore, a tolerable matching level for velocity and acceleration is achieved as well.

![Fig. 5.83 OC3-Hywind tracking / motion matching experiment result (SS = 3, WP = 4 s, WH = 1.25 m): (M) Mooring fairlead point (a) position, (b) velocity, and (c) acceleration](image)

190
5.3 Visual-based motion control

Fig. 5.84 OC3-Hywind tracking / motion matching experiment result (SS = 3, WP = 9 s, WH = 1.25 m): (M) Mooring fairlead point (a) position, (b) velocity, and (c) acceleration

Fig. 5.85 OC3-Hywind tracking / motion matching experiment result (SS = 4, WP = 4 s, WH = 2.5 m): (M) Mooring fairlead point (a) position, (b) velocity, and (c) acceleration

Fig. 5.86 OC3-Hywind tracking / motion matching experiment result (SS = 4, WP = 10.5 s, WH = 2.5 m): (M) Mooring fairlead point (a) position, (b) velocity, and (c) acceleration
Simulation and experimental test results

Fig. 5.87 OC3-Hywind tracking / motion matching experiment result (SS = 5, WP = 6 s, WH = 4 m): (M) Mooring fairlead point (a) position, (b) velocity, and (c) acceleration

Fig. 5.88 OC3-Hywind tracking / motion matching experiment result (SS = 5, WP = 10.5 s, WH = 4 m): (M) Mooring fairlead point (a) position, (b) velocity, and (c) acceleration

Fig. 5.89 OC3-Hywind tracking / motion matching experiment result (SS = 6, WP = 8 s, WH = 6 m): (M) Mooring fairlead point (a) position, (b) velocity, and (c) acceleration
Section 5.3.5 above demonstrates the motion analysis study applicable for MRE devices, which includes hydrodynamic modelling and simulations of the OC3-Hywind floating wind turbine. This is beneficial for understanding the nature of motion of MRE devices and identifying the conditions weather windows for performing IRM operations. This section presented laboratory experiments replicating the motion of OC3-Hywind floating platform with an industrial robot arm based on the obtained simulation results mentioned above. The developed laboratory rig is used for the development and dry testing of control systems and algorithms for automated visual inspection and intervention operations on MRE devices. The following section presents results of laboratory experiments employing visual servoing algorithms for industrial robot arms to address a moving target, whose motion is described in this section.

5.3.7 Dry laboratory experiments with an industrial robot arm utilising PBVS algorithm to address a moving target

Section 4.4.2 describes the improved version of the developed PBVS algorithm which utilises ANFIS network framework for addressing targets in motion. This section presents experimental tests using Staubli TX60 industrial robot arms to validate the proposed vision-based control solution for moving targets. The robot software that generates target motion is standalone software also developed by the author, see section 5.3.6. It is designed to enable a Staubli robot arm to physically simulate the motion of OC3-Hywind floating wind turbine subject to regular sinusoidal sea waves, i.e. robot’s end-effector to replicate the motion of a specific point of interest on the floating platform. The resulting end-effector motion is a repetitive quasi-elliptical
planar trajectory whose heave and surge position coordinates vary sinusoidally. Heave and surge motion have the same period, which is the period of the incident sea waves that are causing the motion; however, heave and surge amplitudes are different and phase shifted. The implemented reference trajectories for the target are acquired from hydrodynamic modelling and simulations described in section 5.3.5. To run the target-motion robot software, a user specifies the following parameters: the wave period, the heave amplitude, the surge amplitude and the phase shift, after which the robot starts moving in the desired repetitive manner. The robot control software employed for target following is an upgraded version of the software for addressing stationary targets with industrial robot arms, described in section 5.3.3. It incorporates two parts: the first one runs on the PC and utilises the developed enhanced PBVS algorithm, and the second one runs on the Staubli robot controller and is in charge of robot arm low-level motion control; these two software units communicate over an Ethernet network. Following a moving target is a component of a typical task for ROV manipulators, e.g. visual inspection of mooring chain links with a wrist-mounted camera, water jet cleaning of moving targets, or taking measurements with a laser or similar tool, e.g. cathodic protection probe mounted on the end-effector. Such ROV manipulation tasks could clearly be aided and facilitated with a capability to track a moving target. A mooring chain connected to a floating buoy or a surface energy converter device will be in motion and motion tracking aids inspection or intervention on such a target. Another example can be any intervention task for subsea manipulators, such as handling a valve or plugging a connector, and if target devices necessitating such intervention actions are in motion, the manipulator must be able to follow them.

Three identical sets of experiments were carried out with robot arms for the same task of following a moving target; each experiment implemented a different variation of the visual servoing algorithm. The first of the three implemented algorithms is the PBVS algorithm that does not include any target motion prediction method. This algorithm as such is the one addressed in sections 5.3.4 and 5.3.3. The second algorithm utilises a motion prediction method based on numerical interpolation, and the third algorithm utilises an ANFIS predictor for the same purpose; both methods are detailed in section 4.4.2. Each set consists of eight experiments that have different trajectories implemented for the moving target. These selected trajectories cover the essential sea conditions: for each sea state (3–6) two boundary cases are addressed, one of which represents the worst sea conditions for a given sea state. The sea states 1 and 2 are omitted because the motion induced in such conditions is negligible. In the addressed trajectories, the peak-to-peak motion amplitude ranges from 0.04 m to 1.6 m for the surge motion, and from 0.004 m to 0.8 m for the heave motion. The
peak of the sinusoidal velocity ranges from $15 \text{ mm s}^{-1}$ to $500 \text{ mm s}^{-1}$ for the surge motion, and from $2.5 \text{ mm s}^{-1}$ to $250 \text{ mm s}^{-1}$ for the heave motion, while the peak of the acceleration ranges from $25 \text{ mm s}^{-2}$ to $300 \text{ mm s}^{-2}$ for the surge motion, and from $4 \text{ mm s}^{-2}$ to $150 \text{ mm s}^{-2}$ for the heave motion.

Firstly, the results of one particular experiment are analysed in detail, while the results of the experiments in general are presented and discussed afterwards. The algorithm that utilises ANFIS for target motion prediction was implemented for this particular experiment. The selected scenario for this particular experiment is of sea state 4 with the wave period of $10.5 \text{ s}$ and wave height of $2.5 \text{ m}$. For this scenario, the corresponding target motion parameters are as follows: the peak-to-peak heave amplitude of $0.25 \text{ m}$, the peak heave velocity of $153 \text{ mm s}^{-1}$, the peak heave acceleration of $92 \text{ mm s}^{-2}$, the peak-to-peak surge amplitude of $0.51 \text{ m}$, the peak surge velocity of $74 \text{ mm s}^{-1}$, the peak surge acceleration of $46 \text{ mm s}^{-2}$ and the $92.8^\circ$ phase shift. A preparation phase is introduced in the experiment to train the ANFIS network. In this phase, the experiment participant manually moves the camera-equipped robot so that the target is in the camera field of view and runs the target-motion robot software with the adequate motion parameters. After that, the training software acquires 200 target position measurements during the preparation phase, of which the first 190 are used for the training and the remaining 10 for the validation. In this particular experiment, the ANFIS network was trained for predicting the first future constitutive position vector ($P = 1$) based on four previous consecutive measurements ($D = 4$, $\Delta = 1$). The training lasted for $0.3 \text{ s}$ and included 10 epochs. After the completion of the training phase, the target following phase started, lasting $100 \text{ s}$. During this phase the robot arm attained the desired pose and kept moving and following the target while maintaining a distance of $230 \text{ mm}$ between the camera and the target (Fig. 5.91). Fig. 5.92 shows the estimated position of the moving target and the position of the TCP following it. In this particular experiment, the TCP is an imaginary point placed $230 \text{ mm}$ in front of the camera. Fig. 5.93 shows the position and orientation errors during the experiment. From this figure it can be seen that in the steady state the position error is below $10 \text{ mm}$, $5 \text{ mm}$, and $2 \text{ mm}$ for the $x$, $y$, and $z$ coordinates, respectively, while the orientation angle error (angle-axis representation) is below $5^\circ$. The presented results demonstrate that the robot arm successfully followed the target and accomplished the given task.

As indicated previously, multiple experiments such as the one just described were carried out, implementing different visual servoing algorithm variations with different targets motion. Figs. 5.94–5.101 show the results of the sets of three experiments; each set of experiments is related to the implementation of three different algorithms for the same target motion. As expected, the visual servoing algorithm that does
Simulation and experimental test results

![Robot arm following a moving target (fiducial marker) utilising the developed ANFIS enhanced PBVS algorithm](image)

Not include target motion prediction showed the weakest performance. Moreover, during one of the experiments the fiducial marker left the camera field of view, which resulted in robot arm failing to follow the target (Fig. 5.101a). When the fiducial marker re-entered the field of view of the camera, the following phase continued; however, this target “escape” occurred several times during the experiment—the manipulator was reacting too slow. For this reason, this particular experiment was repeated multiple times, and the outcome was the same. The visual servoing algorithm utilising the numerical integration based method for target motion prediction showed improved performance; the error norm is significantly lower for all experiments, as can be seen from (b) parts of Figs. 5.94–5.101. Additionally, by utilising this algorithm, the robot arm followed the target successfully in the same conditions where the previous algorithm had failed. The visual servoing algorithm utilising ANFIS for target motion prediction showed the best performance of the three, as it resulted with a considerably lower error norm for all the experiments, which can be seen from the (c) parts of Figs. 5.94–5.101. From the addressed experiments, it is concluded that the error norm increases with the increase of target displacement and velocity, regardless of the algorithm implemented. Nevertheless, the experimental results are satisfactory in general, as the robot arm successfully followed the target, especially employing the proposed ANFIS based visual servoing algorithm. The proposed solution reduces the positioning error during the visual servoing task to a certain degree. The author believes that this reduction is sufficiently low for specific tasks carried out with ROV manipulators as they do not require very high accuracy, i.e. for visual inspection, only keeping the object in the camera field of view is probably accurate enough.
5.3 Visual-based motion control

Fig. 5.92 Robot arm TCP following the target

Fig. 5.93 Absolute values of position and orientation errors
Concluding remarks for section 5.3.6

The proposed visual servoing algorithm for ROV manipulators that can be implemented to perform subsea visual inspection and light intervention tasks automatically on marine energy devices and other subsea offshore structures is experimentally tested and the results are presented in this section. The developed software is designed so that it can be easily interchangeable with the existing ROV manipulators and control systems, which the author believes is a necessity for implementation on the global fleet of work-class ROVs. The vision-based control algorithm is developed with the emphasis on the ability to follow a moving target with the robotic manipulator. As the developed software is to be applied in MRE sites, this is of great importance due to the inevitable motion disturbance of the ROV and target infrastructure induced by sea waves, sea currents, tides, and wind. The solution to compensate for the induced motion is proposed by integrating the visual servoing algorithm with ANFIS network, which is capable of predicting the target motion, and by doing so, improving the controller. The effectiveness of the proposed algorithm is experimentally validated in a set of experiments carried out in the dry laboratory conditions using industrial robot arms, with the performance compared between two other algorithms. Having achieved satisfactory results, the proposed algorithm is implemented on the commercial hydraulic underwater manipulator system and tested in dry laboratory conditions, which is covered in the following subsection.

Fig. 5.94 Experimental results using PBVS algorithm (a) without motion prediction, (b) with numerical integration based prediction, and (c) with ANFIS based prediction. The reference target trajectory simulates (M) mooring fairlead point of OC3-Hywind floating wind turbine (SS = 3, WP = 4 s, WH = 1.25 m)

5.3.8 Dry laboratory experiments with an underwater manipulator utilising PBVS algorithm to address a moving target

The PBVS algorithm for commercial subsea ROV manipulators improved with ANFIS motion prediction is developed and described in section 4.4.2. Section 5.3.7 above presents the experimental validation of the proposed control solution with
5.3 Visual-based motion control

Fig. 5.95 Experimental results using PBVS algorithm (a) without motion prediction, (b) with numerical integration based prediction, and (c) with ANFIS based prediction. The reference target trajectory simulates (M) mooring fairlead point of OC3-Hywind floating wind turbine (SS = 3, WP = 9 s, WH = 1.25 m)

Fig. 5.96 Experimental results using PBVS algorithm (a) without motion prediction, (b) with numerical integration based prediction, and (c) with ANFIS based prediction. The reference target trajectory simulates (M) mooring fairlead point of OC3-Hywind floating wind turbine (SS = 4, WP = 4 s, WH = 2.5 m)

Fig. 5.97 Experimental results using PBVS algorithm (a) without motion prediction, (b) with numerical integration based prediction, and (c) with ANFIS based prediction. The reference target trajectory simulates (M) mooring fairlead point of OC3-Hywind floating wind turbine (SS = 4, WP = 10.5 s, WH = 2.5 m)
Simulation and experimental test results

Fig. 5.98 Experimental results using PBVS algorithm (a) without motion prediction, (b) with numerical integration based prediction, and (c) with ANFIS based prediction. The reference target trajectory simulates (M) mooring fairlead point of OC3-Hywind floating wind turbine (SS = 5, WP = 6 s, WH = 4 m)

Fig. 5.99 Experimental results using PBVS algorithm (a) without motion prediction, (b) with numerical integration based prediction, and (c) with ANFIS based prediction. The reference target trajectory simulates (M) mooring fairlead point of OC3-Hywind floating wind turbine (SS = 5, WP = 10.5 s, WH = 4 m)

Fig. 5.100 Experimental results using PBVS algorithm (a) without motion prediction, (b) with numerical integration based prediction, and (c) with ANFIS based prediction. The reference target trajectory simulates (M) mooring fairlead point of OC3-Hywind floating wind turbine (SS = 6, WP = 8 s, WH = 6 m)
5.3 Visual-based motion control

Fig. 5.101 Experimental results using PBVS algorithm (a) without motion prediction, (b) with numerical integration based prediction, and (c) with ANFIS based prediction. The reference target trajectory simulates (M) mooring fairlead point of OC3-Hywind floating wind turbine (SS = 6, WP = 9 s, WH = 6 m)

industrial robot arms addressing a target in motion. After achieving satisfactory results, the author encapsulated the proposed visual servoing control algorithm into the developed pilot control software for Schilling manipulators, described in section 5.2.3. This subsection describes the experimental evaluation of the proposed ANFIS enhanced PBVS algorithm in dry laboratory conditions using a hydraulic subsea robotic manipulator (Titan 2) to perform typical ROV tasks on moving targets.

The experiments described in this subsection included an LBR iiwa robot arm which physically simulated target motion by employing a separate robot software, developed by a colleague researcher David Adley, using KUKA Sunrise, a Java programming software specialised for KUKA robots. This target-motion software allows a KUKA robot arm to replicate the motion response of OC3-Hywind floating wind turbine subject to regular sinusoidal sea waves; it is the equivalent KUKA version of the software described in section 5.3.7 above. The only difference in this case is that the Schilling SCU performs low-level motion manipulator control.

Multiple series of experiments with the task of following a moving target with a robotic manipulator were carried out. Different modifications of the ANFIS network were tested to understand how various parameters and settings influence the effectiveness of the visual servoing algorithm. The robustness of the algorithm to the difference between the target motion used for training, and for validation, was also examined. Additionally, the algorithm was tested in addressing various target motion, covering multiple sea states from relatively calm sea to very high waves, and facing the target from different sides. Lastly, the task of visually inspecting a moving mooring chain with a wrist-mounted manipulator camera was physically simulated.

Firstly, a single experiment is described to highlight the advantage of utilising ANFIS enhanced visual servoing by comparing its results with the results of the PBVS algorithm that does not integrate any motion prediction method. The target motion selected for this experiment corresponds to the motion response of OC3-Hywind floating wind turbine subject to sea state 4, with a wave period of 10.5 s and
Simulation and experimental test results

wave height of 2.5 m. More precisely, this is the motion of a specific point (M) where the catenary mooring lines are attached to the floating platform. The sinusoidal motion parameters of the point M in the described sea conditions are as follows: the peak-to-peak heave amplitude of 250 mm; the peak heave velocity of \(153 \text{ mm s}^{-1}\); the peak heave acceleration of \(92 \text{ mm s}^{-2}\); the peak-to-peak surge amplitude of 510 mm; the peak surge velocity of \(74 \text{ mm s}^{-1}\); the peak heave acceleration of \(46 \text{ mm s}^{-2}\), and with a 92.8° phase shift. Depending on the direction in which the ROV faces the floating platform, the target’s motion projects differently in the reference frame of the wrist-mounted camera. In this particular experiment, it is assumed that the ROV is positioned between \(x\) and \(y\) axes at an angle of 45°, which means that the target is moving along a 3D trajectory from the camera point of view, oscillating in all directions (up-down, left-right, and front-back). The preparation phase of the experiment included teleoperating the camera-equipped manipulator so that the target is in the field of view of the camera, and running the target-motion KUKA robot software with appropriate motion parameters to train the ANFIS network. During the preparation stage, the training software acquired 107 target position measurements, which include 5 target wave motion cycles. The acquired data is then used to train the ANFIS network for predicting the third future consecutive position vector of the target \((P = 3)\) based on four previous consecutive measurements \((D = 4, \Delta = 1)\). The training lasted for 0.116 s and included 5 epochs. The phase where the manipulator approaches a moving target from afar is not addressed, and LBR iiwa robot carrying the target started moving only after the Titan 2 manipulator approached it. Subsequently, the target following phase was triggered, which lasted 46 s and included 4 motion cycles. During this phase, the Titan 2 manipulator converged to the desired pose and kept moving and following the target while maintaining the distance of 150 mm between it and the camera (Fig. 5.102). Fig. 5.103 shows the estimated position of the moving target and the position of the manipulator TCP following it. The TCP is an imaginary point placed 150 mm in front of the camera. Fig. 5.104 shows the position and orientation errors during the experiment. The orientation error is computed from the axis-angle representation of the rotation as the difference between desired and actual angle values. Fig. 5.105 represents a histogram with normal distribution of the position error norm which allows identification of the range of the position error as well as its frequency of occurrence during the experiment.

This experiment was repeated multiple times, and it was successful every time as the manipulator followed the target throughout the experiment. The benefit of the proposed ANFIS enhanced visual servoing algorithm can be identified from Fig. 5.106 by comparing the results of the ten identical experiments in which ANFIS
prediction was used with the result of the experiment utilising the PBVS algorithm that does not include any target motion prediction method. The experimental results demonstrate not only that the underwater manipulator successfully followed the target using the proposed ANFIS enhanced visual servoing algorithm but also that the position error in this particular case scenario was significantly reduced, by approximately 75% compared to no prediction.

The same target motion was addressed in another series of experiments with the goal to investigate the influence of the number of epochs used for ANFIS training on the effectiveness of the visual servoing algorithm. Five tests were conducted, each with a different number of epochs. The effectiveness of the sole ANFIS training was evaluated by employing the trained ANFIS network to predict the target position from the data set formed as a combination of position measurements used for the training and additional points used only for the validation. Fig. 5.107 depicts prediction errors for each position coordinate and position error norm for a single case of ANFIS training with 50 epochs; these are the results of sole ANFIS training, not of its implementation within the visual servoing algorithm. The results for other addressed cases are included in Table 5.5, along with the results of the implementation of trained ANFIS networks using different numbers of epochs within the visual servoing algorithm for the task of following a moving target. As was expected, ANFIS trained with the largest number of epochs provided the best results (see mean and median position errors) but for the price of much longer training time. The table shows that the position error during the visual servoing algorithm is much larger (approximately ten times) than the ANFIS prediction error. The reason is the influence of other parts of the algorithm on the overall visual servoing performance, such as unideal accuracy of the low-level manipulator joint position controller, errors
Simulation and experimental test results

Fig. 5.103 Subsea manipulator TCP following the target

Fig. 5.104 Absolute values of position and orientation errors
5.3 Visual-based motion control

introduced in the manipulator modelling, the time it takes for the manipulator to move between two consecutive set points, etc. An interesting observation is that even with a few training epochs and short training time, ANFIS prediction provides useful results, e.g. for the case of 5 epochs the mean position error of visual servoing is larger by only 6% than the same error in the case with 300 training epochs. However, the training is 54 times faster.

Another series of experiments was performed to investigate the influence of selecting a different future target position vector ($P$) that the ANFIS network is trained to predict, on the performance of the visual servoing algorithm. We have already seen that sole ANFIS prediction works very well. If it were possible to control the time it takes from the moment the controller issues a position command to the manipulator to the moment the manipulator reaches the desired pose, or to control the end-effector linear velocity, the overall visual servoing performance would have been independent of $P$. However, since the commercial manipulator SCU does not have this possibility, the approach taken is to experimentally identify which future prediction position vector provides the best performance. To that
Simulation and experimental test results

Fig. 5.107 Prediction error of ANFIS network trained with 50 epochs and 128 position measurements. Error of separate position coordinates and position error norm

dend, the ANFIS network was trained for the prediction of the first four consecutive future target position vectors. The results of visual servoing experiments utilising each trained ANFIS network are compared with the results of experiments utilising the PBVS algorithm without any motion prediction implementation. Fig. 5.108 depicts the position error norm during the execution of the target following task. Additionally, Fig. 5.109 presents a histogram and normal distribution of the position error norm for each case. Based on mean error ($\mu$) and standard deviation ($\sigma$) values,

Table 5.5 Experimental results implementing ANFIS with different number of training epochs with the PBVS algorithm for the task of following a moving target

<table>
<thead>
<tr>
<th>Epochs</th>
<th>ANFIS Training</th>
<th>ANFIS Visual Servoing</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td>Training Data = 128; Validation Data = $T \times D + 10$</td>
<td>Validation Data = 50</td>
</tr>
<tr>
<td>Time (s)</td>
<td>Position Error Norm (mm)</td>
<td>Position Error Norm (mm)</td>
</tr>
<tr>
<td>Mean</td>
<td>Median</td>
<td>Min</td>
</tr>
<tr>
<td>300</td>
<td>7.170</td>
<td>4.1</td>
</tr>
<tr>
<td>100</td>
<td>4.3</td>
<td>4.0</td>
</tr>
<tr>
<td>50</td>
<td>1.209</td>
<td>4.6</td>
</tr>
<tr>
<td>10</td>
<td>0.251</td>
<td>5.2</td>
</tr>
<tr>
<td>5</td>
<td>0.133</td>
<td>5.5</td>
</tr>
</tbody>
</table>
5.3 Visual-based motion control

Fig. 5.108 Position error norm for a single experiment without ANFIS and four experiments with ANFIS, each predicting a different future consecutive target position vector.

The ANFIS network trained to predict the third future target position vector \((P = 3)\) achieved the best visual servoing performance.

The proposed algorithm proves to be effective if employed on the same target motion that is used for training the ANFIS network. Another series of experiments was performed to investigate the robustness of the algorithm to a deviation in target motion in relation to the nominal one. Three series of experiments were conducted addressing both surge and heave motion amplitudes and periods; the results are given in Table 5.6. In the first series of experiments, the period of target surge and heave motion is identical to the period used for the training but the amplitudes of both heave and surge motion are increased and decreased by 10% steps. The 40% amplitude increase was the largest that still yielded successful target following. Decreasing the motion amplitudes eventually leads to the static target case and therefore, all such target following experiments were successful, however with noticeable target position error degradation, as can be seen from the table.

In the second series of experiments the target surge and heave motion amplitudes are fixed to values the same as the ones used for the training but the period is increased and decreased by 10% steps. Increasing the period results in slower target motion and is therefore a less interesting case as the failure of target following is less likely to happen. The period was thus increased only up to 20% to observe how it affects the position error. On the other hand, decreasing the motion period results in faster target motion which is much more difficult for the visual servoing algorithm to handle. The target following task was performed successfully up to a 15% decrease in the period; with any larger change, the target moved too fast causing the marker to leave the field of view of the camera. The last robustness testing series of experiments includes simultaneously changing target surge and heave motion amplitudes and period. The largest increase in amplitude and decrease...
Simulation and experimental test results

Fig. 5.109 Histogram and normal distribution of the target position error for the task of following a moving target by utilising: (a) PBVS algorithm not implementing any motion prediction method; PBVS algorithm with the implementation of ANFIS network for the prediction of (b) 1st, (c) 2nd, (d) 3rd, (e) 4th future consecutive target position vector.
5.3 Visual-based motion control

Table 5.6 Robustness analysis of the ANFIS enhanced visual servoing algorithm to the change in target motion amplitude and period.

<table>
<thead>
<tr>
<th>Error</th>
<th>Without ANFIS</th>
<th>ANFIS</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td>A = 100%</td>
<td>A = 110%</td>
</tr>
<tr>
<td></td>
<td>P = 100%</td>
<td>Period = 100%</td>
</tr>
<tr>
<td>Mean</td>
<td>131.1</td>
<td>54.3</td>
</tr>
<tr>
<td>Median</td>
<td>132.1</td>
<td>54.3</td>
</tr>
<tr>
<td>Min</td>
<td>69.9</td>
<td>18.7</td>
</tr>
<tr>
<td>Max</td>
<td>169.5</td>
<td>97.4</td>
</tr>
<tr>
<td>Sigma</td>
<td>22.9</td>
<td>17.7</td>
</tr>
<tr>
<td>N</td>
<td>89</td>
<td>90</td>
</tr>
</tbody>
</table>

in period with successful target following was 15%, while the largest increase in period and decrease in amplitude was 20%. Whether the proposed ANFIS enhanced visual servoing algorithm is robust enough to follow a moving target depends on which motion parameters are changed and whether they are increased or decreased. However, based on the presented experimental results, a rough estimate can be made that the target following utilising the proposed algorithm is likely to be successful within the range of 15% target motion change; beyond this, retraining is necessary.

Having investigated the influence of the number of training epochs and the selection of future consecutive target position vector ANFIS predicts on the overall performance of the visual servoing algorithm, the author noticed a possibility to investigate integrating ANFIS retraining in the visual servoing controller loop. Such solution would eliminate the necessity to perform preparation phase and data acquisition phase before employing the visual servoing algorithm. The visual servoing algorithm could thus start without ANFIS, and during the target following, once enough data measurements are acquired, the ANFIS network would get trained and continuously retrained in every controller loop or less frequently. The algorithm implemented in such a way would be more robust due to the possibility to take any difference in target motion into account in the ANFIS (re)training. To integrate the training into the control loop of the visual servoing algorithm, the training time needs to be short enough not to endanger the controller’s performance. As the training time is directly proportional to the number of epochs and number of position measurements, the aim was to investigate what combination of these parameters provides the best visual servoing performance while keeping the training time relatively low,
Simulation and experimental test results

Table 5.7 Analysis of ANFIS enhanced visual servoing performance for potential integration of ANFIS retraining in the PBVS control loop

<table>
<thead>
<tr>
<th>Epcs</th>
<th>Cycles</th>
<th>Trng. Data</th>
<th>Trng. Time (s)</th>
<th>Position Error Norm (mm)</th>
<th>Valid. Data</th>
<th>ANFIS Training</th>
<th>ANFIS Visual Servoing</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td></td>
<td></td>
<td></td>
<td>Mean</td>
<td>Median</td>
<td>Min</td>
<td>Max</td>
</tr>
<tr>
<td>5</td>
<td>6</td>
<td>128</td>
<td>0.133</td>
<td>5.5</td>
<td>4.7</td>
<td>0.9</td>
<td>16.2</td>
</tr>
<tr>
<td>5</td>
<td>5</td>
<td>107</td>
<td>0.116</td>
<td>5.8</td>
<td>4.9</td>
<td>1.0</td>
<td>17.9</td>
</tr>
<tr>
<td>8</td>
<td>4</td>
<td>86</td>
<td>0.145</td>
<td>6.2</td>
<td>5.1</td>
<td>0.7</td>
<td>21.0</td>
</tr>
<tr>
<td>10</td>
<td>3</td>
<td>65</td>
<td>0.137</td>
<td>15.4</td>
<td>15.2</td>
<td>0.0</td>
<td>40.6</td>
</tr>
<tr>
<td>15</td>
<td>2</td>
<td>44</td>
<td>0.142</td>
<td>15.6</td>
<td>15.2</td>
<td>0.0</td>
<td>39.6</td>
</tr>
<tr>
<td>25</td>
<td>1</td>
<td>23</td>
<td>0.133</td>
<td>16.6</td>
<td>16.8</td>
<td>0.0</td>
<td>40.7</td>
</tr>
</tbody>
</table>

Note. Epcs. – Epochs; Cycles. – Cycles; Valid. – Validation; F – Fail.

i.e. approximately between 100 ms and 150 ms. Numbers of data measurements were selected so that each training data set corresponds to a finite number of target motion cycles, i.e. six training data sets with one to six target motion cycles. For each training data set, a number of ANFIS training epochs was selected so that the training takes less than 150 ms. Table 5.7 encapsulates the results of ANFIS training and its implementation within the visual servoing algorithm for target following. The results show that the ANFIS training with a greater number of target motion cycles yields better motion prediction (see mean error, median error, and standard deviation). Additionally, the mean position error norm is approximately three times larger in the cases with three and fewer motion cycles than in the case with four and more motion cycles. Considering only ANFIS training validation, it might be assumed that the best potential combination would be one with six target motion cycles and 5 epochs. However, the results of the actual visual servoing show that this is not necessarily true, depending on the best performance metrics definition. For example, mean position error, maximal position error, and standard deviation were the smallest in the case with four target motion cycles and 8 training epochs. The values of standard deviation and max position error may be essential for tasks following a moving target. It might be better to have a larger mean error with narrower normal distribution than a smaller mean error with wider normal distribution because the probability the target might leave the field of view of the camera is higher. Visual servoing utilising the ANFIS network trained with data from a single target motion cycle was unsuccessful, with the manipulator failing to follow the target. This particular experiment was repeated multiple times, yielding the same outcome. Based on the experimental results it can be concluded that ANFIS training requires a training data set with at least two target motion cycles.

As is already mentioned, the direction in which the ROV and the manipulator face the target MRE device is essential as surge motion projects differently in the camera
5.3 Visual-based motion control

reference frame. If the ROV faces the floating platform in the \( y \)-axis direction (Fig. 5.64), the surge motion projects as horizontal left-right motion. The other boundary case is with the ROV aligned with the \( x \)-axis of the MRE device, in which the target moves towards and from the ROV. In any other alignment, the surge motion of the floating platform is split between the two horizontal axes in the camera reference frame. A test goal was thus to examine the behaviour of visual servoing subject

Fig. 5.110 Position error norm and histogram with normal distribution during the target following task, with ROV facing the target in the (a) \( x \)-axis direction, (b) \( y \)-axis direction, (c) direction between \( x \) and \( y \) axes at approximately 45°.
Simulation and experimental test results

to different directions in which the ROV faces the target. Three experiments were performed addressing the two mentioned boundary cases and one intermediate case with the ROV facing the target approximately at an angle of 45° between $x$ and $y$ axes (see Fig. 5.64). All three target following experiments were successful, but the algorithm performance differed. Fig. 5.110 presents the results including target position error norm and histogram with the normal distribution for all three case scenarios. The most challenging case scenario was the one where the ROV faces the target in the $x$-axis direction.

The proposed ANFIS enhanced visual servoing algorithm was tested subject to diverse target motion trajectories, selected to cover the essential sea conditions. For each sea state (3–5), two boundary cases are addressed, one of which represents the worst sea conditions for a given sea state, and a single case for sea state 6. Since the induced motion in sea states 1 and 2 is negligible, these cases are omitted. Visual servoing target following with and without ANFIS implementation for motion prediction was employed for each target motion. Figs. 5.111–5.116 present the results of the experiments. As expected, the experiment results demonstrate that the increase of target displacement and velocity increases the error norm, regardless of the algorithm implemented. Nonetheless, the visual servoing algorithm utilising ANFIS motion prediction showed improved performance compared to tracking without any motion prediction as it resulted in a considerably lower target position error norm for all experiments. Depending on the target motion, ANFIS implementation reduced the position error from 2% up to 80%. The results show that the benefit of using ANFIS is more noticeable if the target moves slower, i.e. with larger motion periods. Finally, experiments simulating a typical subsea visual inspection task were performed utilising the proposed algorithm. A test panel with the image of a mooring chain link and the fiducial marker was mounted on the KUKA robot arm end-effector (Fig. 5.117). The idea of visual inspection experiment is that the manipulator moves in such a way that the wrist passes over the whole body of the chain while keeping a predefined close distance to it. The desired path defined relative to the fiducial marker along which the camera was to move consists of a certain number of set points. To complete the visual inspection the manipulator mounted camera must visit all the points one by one in a specified order. Two inspection paths were addressed: the first one comprised of 9 points, represented by yellow circles in Fig. 5.117, and the second one comprised of 27 points, represented by green circles in the same figure. The threshold below which the position error norm had to drop so that the camera moves towards the next setpoint was 50 mm for the first path and 25 mm for the second path. The inspection task was performed successfully with a stationary target utilising the visual servoing algorithm without any motion prediction method,
5.3 Visual-based motion control

Fig. 5.111 Target position error norm, histogram and normal distribution for the target following task using PBVS algorithm (a) without motion prediction, and (b) with ANFIS predictor. The target reference trajectory simulates the (M) mooring fairlead of the OC3-Hywind floating wind turbine (SS = 3, WP = 9 s, WH = 1.25 m).

Fig. 5.112 Target position error norm, histogram and normal distribution for the target following task using PBVS algorithm (a) without motion prediction, and (b) with ANFIS predictor. The target reference trajectory simulates the (M) mooring fairlead of the OC3-Hywind floating wind turbine (SS = 4, WP = 4 s, WH = 2.5 m).
Simulation and experimental test results

Fig. 5.113 Target position error norm, histogram and normal distribution for the target following task using PBVS algorithm (a) without motion prediction, and (b) with ANFIS predictor. The target reference trajectory simulates the (M) mooring fairlead of the OC3-Hywind floating wind turbine (SS = 4, WP = 10.5 s, WH = 2.5 m)

Fig. 5.114 Target position error norm, histogram and normal distribution for the target following task using PBVS algorithm (a) without motion prediction, and (b) with ANFIS predictor. The target reference trajectory simulates the (M) mooring fairlead of the OC3-Hywind floating wind turbine (SS = 5, WP = 6 s, WH = 4 m)
5.3 Visual-based motion control

Fig. 5.115 Target position error norm, histogram and normal distribution for the target following task using PBVS algorithm (a) without motion prediction, and (b) with ANFIS predictor. The target reference trajectory simulates the (M) mooring fairlead of the OC3-Hywind floating wind turbine (SS = 5, WP = 10.5 s, WH = 4 m)

Fig. 5.116 Target position error norm, histogram and normal distribution for the target following task using PBVS algorithm (a) without motion prediction, and (b) with ANFIS predictor. The target reference trajectory simulates the (M) mooring fairlead of the OC3-Hywind floating wind turbine (SS = 6, WP = 8 s, WH = 6 m)
Simulation and experimental test results

![Visual inspection experiment test panel](image)

Fig. 5.117 Visual inspection experiment test panel

and with the proposed ANFIS enhanced algorithm addressing the moving target. Fig. 5.118 presents the position error norm during the experiments. The red dash-dot lines represent position error norm thresholds, and the green dash-dot lines represent moments of reaching set points. As was expected, the visual inspection tasks addressing the static target took less time than the ones addressing the target in motion. Attempts to address target in motion without a motion prediction method integrated into the visual servoing algorithm were unsuccessful; the selected error threshold was too large and thus prevented the visual inspection task completion, unlike the proposed ANFIS enhanced algorithm which proved to be successful in doing so.

**Concluding remarks for section 5.3.8**

The author proposed a visual servoing algorithm for ROV manipulators that can be implemented to perform subsea visual inspection and light intervention tasks automatically on marine energy devices and other subsea offshore structures. The developed software was designed so that it could be easily retrofitted to existing underwater manipulator control systems, which the author believes is a necessity for implementation of automation on the global fleet of commercial work-class ROVs with Schilling or similar manipulators. The vision-based control algorithm is developed with emphasis on the ability to follow a moving target with the robotic manipulator and this is experimentally evaluated in this section. As the developed software is to be applied in MRE sites, this is of the utmost importance due to the inevitable motion disturbance of the ROV and target infrastructure induced by sea waves, sea currents, tides, wind, etc. The solution to compensate for the induced motion is proposed by integrating the visual servoing algorithm with ANFIS network which can predict the target motion and feed it to the controller. The effectiveness of the proposed algorithm was experimentally validated through multiple series of experiments carried out in the dry laboratory conditions using a commercial hydraulic underwater manipulator. The proposed algorithm demonstrated positive
5.3 Visual-based motion control

Fig. 5.118 Target position error norm during the visual inspection experiment with:
(a) static target, 9 points, and 50 mm error threshold; (b) static target, 27 points, and
25 mm error threshold; (c) dynamic target, 9 points, and 50 mm error threshold; (d) dynamic target, 27 points, and 25 mm error threshold.
Simulation and experimental test results

performance, and the author believes that its utilisation and uptake could automate a significant number of typical subsea operations, enabling ROV pilots to perform inspection and intervention tasks easier and faster, which could provide significant cost savings. The next research step is integrating it into a commercial ROV system and testing it in the real underwater environment, addressing different inspection as well as light intervention tasks such as surface cleaning over an extended trial period. Additionally, the control strategy can be expanded from the sole manipulator control to the control of the vehicle-manipulator system where the inspection and intervention tasks might be addressed with the ROV motion as well.

5.4 Collision detection simulations and experiments

This section presents simulation and experimental results related to the proposed collision detection algorithm for ROV manipulators described in section 4.5. The designed software is implemented in MATLAB for initial simulation testing and then in LabVIEW for both simulations and experiments, encapsulated in the developed pilot control software described in section 5.2.3. Firstly, an implementation scenario for the proposed collision-free manipulation algorithm on an ROV equipped with two manipulators is analysed. One such robotic vehicle is the MRE ROV, named ROV Étaín (Fig. 5.119), a University of Limerick owned ROV which is mainly used for research purposes. This ROV is equipped with two seven function Schilling Orion

Fig. 5.119 University of Limerick MRE ROV—ROV Étaín
5.4 Collision detection simulations and experiments

7P manipulators, both of which have position sensors in each joint. The information provided by these sensors along with the known CAD model of the whole ROV, including the relative pose between the robotic manipulator bases, are sufficient to implement the proposed collision detection algorithm. The collision detection algorithm is developed in C++ in the form of an independent DLL by a colleague researcher Matija Rossi, and encapsulated in the robotic manipulation pilot control software, described in section 5.2.3, and journal published, see Appendix G.

5.4.1 Simulations with subsea manipulators and an ROV model

The simulation scenario for the validation of the developed software, including the proposed collision avoidance algorithm, consists of a mathematical model of the real MRE ROV and the two accompanying manipulators. Two cases are addressed in simulation: the collision between each manipulator and the base vehicle, and the collision between the two manipulators. Voxels of different sizes ranging from 10 mm to 100 mm were used for the modelling of all the rigid objects that are part of the ROV, including the manipulators (Fig. 5.120). Such variation enabled analysis of how the algorithm behaves with the increase in computational load due to the voxel map size, i.e. how voxel size affects the algorithm execution time.

The simulation scenario runs as follows. Both robotic manipulators start from predefined collision-free initial configurations defined in joint space; joint space trajectories are then generated for both manipulators simulating the motion command issued by a human operator or a computer program. Each trajectory is a predefined sequence of joint space configurations such that it first causes a collision between the manipulator and the body of the ROV base vehicle and afterwards a collision between the two manipulators. These reference trajectories are inputs of the developed

Fig. 5.120 MRE ROV modelled with voxels of different size: (a) 100 mm; (b) 33 mm, and (c) 10 mm
algorithm which acts as a collision filter, sensing and prohibiting any motion that causes a collision and allowing only collision-free motion. The reference input trajectories and resulting output trajectories are given in Fig. 5.121 in joint space and in Fig. 5.122 in Cartesian space; voxel size is 33 mm. Figs. 5.123 and 5.124 illustrate the visualisation of the collision detection algorithm for the described simulation scenario. The red-yellow and blue-green manipulators on the middle images represent the collision-free output motion of the proposed algorithm, while the red and blue manipulators represent the discarded reference motion. In each control loop, the collision detection algorithm checks not only whether the single desired configuration of the reference trajectory leads to a collision, but also the configurations in between, i.e. the whole volume the manipulator would sweep if it was to move from the current to the desired configuration, which is illustrated by
Fig. 5.123 Simulation of the collision detection between each manipulator and the ROV
Simulation and experimental test results

Fig. 5.124 Simulation of the collision detection between two manipulators
5.4 Collision detection simulations and experiments

Table 5.8 Computational load analysis of the collision detection algorithm

<table>
<thead>
<tr>
<th>Voxel size (mm)</th>
<th>Manip. voxels</th>
<th>Obstacle voxels</th>
<th>Intersection operations</th>
<th>Manip. voxeling (ms)</th>
<th>Intersection (ms)</th>
<th>Total loop (ms)</th>
</tr>
</thead>
<tbody>
<tr>
<td>100</td>
<td>1,120</td>
<td>710</td>
<td>795,200</td>
<td>3.4</td>
<td>0.6</td>
<td>8</td>
</tr>
<tr>
<td>66</td>
<td>2,500</td>
<td>1,688</td>
<td>4,220,000</td>
<td>3.6</td>
<td>1.5</td>
<td>10.2</td>
</tr>
<tr>
<td>33</td>
<td>9,850</td>
<td>5,016</td>
<td>49,407,600</td>
<td>3.8</td>
<td>5.6</td>
<td>18.8</td>
</tr>
<tr>
<td>22</td>
<td>18,530</td>
<td>7,399</td>
<td>137,103,470</td>
<td>4.5</td>
<td>10.7</td>
<td>30.4</td>
</tr>
<tr>
<td>15</td>
<td>47,610</td>
<td>9,964</td>
<td>474,386,040</td>
<td>5.8</td>
<td>26.9</td>
<td>65.4</td>
</tr>
<tr>
<td>10</td>
<td>102,030</td>
<td>13,530</td>
<td>1,380,465,900</td>
<td>8</td>
<td>58.8</td>
<td>133.6</td>
</tr>
</tbody>
</table>

the yellow voxels in the figures. Each control loop executes the collision detection algorithm twice to check the reference motion for each manipulator. Therefore, the green voxels represent the other, passive manipulator, which together with the ROV body (represented by light blue voxels) forms the obstacle voxel map for that iteration. Finally, red voxels represent the imminent collision sensed by the developed algorithm. Table 5.8 shows the computational load of the collision detection algorithm for different voxel sizes, numbers of voxels, and numbers of intersection operations; as well the time required for different algorithm phases.

5.4.2 Dry laboratory experiment with an underwater manipulator

Additionally, the developed collision detection algorithm was tested in a real-world experimental setup in dry laboratory conditions using a Schilling Titan 2 manipulator. In the addressed scenario, the manipulator was intentionally commanded to collide with the floor, which the collision detection algorithm successfully prohibited. Fig. 5.125 shows a photo of this experiment, while Fig. 5.126 shows the utilised trajectories.
Simulation and experimental test results

Fig. 5.126 Reference input and collision-free output trajectories from the experiment

Concluding remarks for section 5.4

The proposed collision-free motion algorithm for marine robotic manipulation is described and successfully evaluated in simulation and experimental setup using a real underwater manipulator. The developed solution can be easily integrated as a software upgrade into the control and hardware systems that are present in the global fleet of industry standard work-class ROVs. The author believes that the presented collision detection algorithm has a potential to be a useful add-on for ROV pilots enabling them to execute typical IRM tasks with greater ease and speed, reducing their fatigue and human factor errors, and eventually reducing the cost of subsea IRM operations in oil and gas, the MRE sector, and other fields of application.

Ongoing work is integrating the developed algorithm into the MRE ROV control software and testing it in offshore subsea trials. Additionally, further algorithm development is planned in order to address aspects such as software optimisation, detecting potential manipulator self-collision, expanding the model library by includ-
5.5 Concluding remarks

Advanced kinematic control solutions for subsea ROV manipulator systems have been proposed in chapter 4. In this chapter, the development and testing of the proposed robot control solutions have been described. The contributions in marine robotics covered in this chapter have been made in three areas: kinematic control, visual servoing, and collision detection and avoidance.

Regarding kinematic control, advanced functions, which are not present in existing commercial ROV manipulator control systems, have been developed and implemented in the form of control software for underwater manipulators. Of a number of realised functions, perhaps the most useful one is enhanced manual and semi-automatic Cartesian control which allows a pilot to drive the manipulator so that the end-effector or wrist-held tool moves in a straight line or rotates around a specified axis. Such manipulation assistance aids dramatically in the execution of intervention tasks with a manipulator, and while this is standard for any robot arm in the manufacturing industry, it does not exist as a part of ROV control systems where old-fashioned telemanipulation with a mini-master arm remains the predominantly supported option within the offshore sector. Additionally, the developed controls enable the automation of a certain number of subsea task to be carried out with an underwater manipulator. The developed control software has been experimentally verified, employed for multiple intervention tasks subsea, and the results have been presented in this chapter.

Regarding visual servoing, for the first time, a solution that works with standard commercial systems already employed in the industry and the global fleet of workclass ROVs has been proposed and experimentally validated in subsea field trials. The proposed visual-based motion control approach has been utilised for fully automatic manipulation on stationary targets addressing typical intervention operations, such as handling valves, tools, and connectors. Additionally, the developed vision-guided control system is capable of performing close visual inspection and light intervention operations on moving targets in challenging conditions of MRE sites. Excellent
Simulation and experimental test results

Initial results have been achieved with very promising applications that have the potential to facilitate IRM operations in the emerging MRE sector.

Lastly, the developed control systems for commercial hydraulic subsea manipulators have been reinforced with a collision detection and avoidance algorithm. The implementation of the proposed algorithm for collision-free manipulation has also been verified in experiments, the results of which have been presented in this chapter. Even though it represents a relatively simple method, this safety feature is novel for the ROV manipulation sector, it can significantly facilitate and speed up intervention operations while significantly reducing risk. As it practically eliminates the possibility of physical damage induced to and by manipulators, it grants the opportunity to the pilot to operate at ease and to concentrate fully on the task itself, which is especially beneficial in challenging conditions with poor visibility.

The work presented in this chapter has been published in a paper at the IEEE Oceans 2015 conference, as a part of the Advances in Intelligent Systems and Computing book chapter, and in two journal articles, in the Control Engineering Practice and MDPI Sensors, see Appendix G.
Chapter 6

Discussion, conclusion, and future work

6.1 Introduction

This chapter summarises the research covered by this thesis, lists and analyses the main contributions, and suggests directions for further research and development in the field of ROV manipulator systems.

6.2 Summary of the thesis

Chapter 1, “Introduction”, has provided background and motivation for research work described in the thesis. This chapter has identified aims and objectives, listed main contributions, and given an overview of the thesis.

Chapter 2, “Robotic arms background”, is an introductory chapter which has presented fundamentals in the area of industrial manufacturing robotics, as the robot arms used in this sector are similar with underwater manipulator systems. This chapter has described the theoretical base which has been essential to producing the research contributions reported in this thesis. It has summarised robot kinematics modelling, motion planning, and control methods, and has also presented a review of visual servoing techniques and some machine learning algorithms relevant to this thesis.

Chapter 3, “Literature review on underwater manipulators”, has presented a comprehensive review of the state-of-the-art in ROV manipulation technology. It has summarised background topics related to commercially available subsea manipulator systems, and traditional and modern control approaches existing in the global fleet of ROVs. This chapter has also identified the gaps in the ROV manipulator systems’
Discussion, conclusion, and future work

capabilities with a particular focus on comparison with industrial robot arms and has highlighted essential technology challenge areas to be addressed in research and development. Additionally, this chapter has outlined previous work and recent progress in the field of autonomous underwater manipulation, addressing relevant control topics on various levels. It has concluded with the analysis of the achievements in academia, recognising the deficiency of adequate control solutions for industry standard hydraulic manipulator systems on ROVs to enable fully automatic inspection and intervention. The majority of the ideas for the development reported in this thesis are a result of the review described in this chapter.

The contents of Chapter 4, “Design and development of advanced control systems for underwater manipulators”, is strongly indicated by its title. This chapter has presented the development of a kinematics control engine that brings underwater manipulators a few steps closer to industrial manufacturing robot arms in the sense of automation capabilities. It has described the implemented modelling approaches and software development for advanced pilot control of manipulators employing enhanced manual and semi-automatic regimes. This chapter has also proposed a novel control solution for subsea ROV manipulators to enable fully automatic visual-based intervention operations on stationary targets addressing typical tasks for ROVs. The proposed solution is extended using machine learning methods to enable ROV manipulators to autonomously address moving targets, which is essential for IRM operations in challenging conditions of the emerging MRE sector. This chapter has also included safety-critical concepts related to collision detection which can improve the performance of ROV intervention operations.

Chapter 5, “Simulations and experimental test results”, has detailed the work done to validate the proposed control solutions for ROV manipulator systems and evaluate their performance in performing IRM tasks typical for the oil and gas sector, the MRE sector, and other fields of application. This chapter has presented the control software for industry standard hydraulic manipulators for ROVs developed as a result of this thesis, which encapsulates all control techniques that Chapter 4 has proposed. It has also described the results achieved in simulation, laboratory experiments and subsea field trials, employing the developed control algorithms on real work-class ROV systems equipped with heavy-duty hydraulic manipulators. The proposed control solutions, backed up with satisfactory experimental test results that this chapter has reported are the subject of a number of conference and journal publications listed in Appendix G.
6.3 Realisation of aims and objectives

This section summarises accomplishments related to realisation of aims and objectives, given in section 1.3.

The aim of the project this thesis reports has been the research and development of advanced ROV manipulator control systems beyond the current state-of-the-art in work-class ROV systems to enable them to address the challenging conditions encountered in the emerging MRE sectors, including offshore wind, floating wind, wave energy conversion, and tidal energy conversion.

The author has proposed an advanced kinematics control engine for manipulator systems on marine work-class ROVs. The developed control system allows control of industry standard hydraulic manipulators in enhanced manual, semi-automatic, and fully automatic (visual) servo control regimes, and is reinforced with collision detection and avoidance. For the first time, a manipulation solution has been realised that is capable of addressing stationary and moving targets in challenging conditions and that works with standard commercial systems already employed in the industry and the global fleet of work-class ROVs. The developed manipulator control software is designed to be easily interchangeable with the existing underwater manipulator control systems that are present in the vast majority of commercial work-class ROVs. This is a huge benefit as the proposed control system can upgrade the existing ones without any modifications to the ROV hardware or software, other than the addition of add-on system blocks, which the author believes is very important or even inevitable for potential commercialisation and entrance to the subsea industry sector. The performance of the proposed control solutions has been verified through extensive simulations and experimental tests in the laboratory and subsea in field trials. Utilising the developed robotic manipulation tools, various subsea intervention tasks that are typically performed by a human pilot teleoperating subsea manipulators have been automated, which significantly facilitates ROV missions. This enables ROV pilots to transfer from direct control to a supervisory position and carry out intervention and inspection operations “at the click of a button”. Pilots can operate with greater ease and speed, reducing their fatigue and preserving energy and concentration. Consequently, performing subsea IRM operations in oil and gas, the MRE sector, and other offshore fields of application have been facilitated to become more comfortable, faster, and therefore, more cost-effective.

- Explore existing control systems for underwater ROV manipulators. A comprehensive literature review related to underwater manipulator systems has been presented in Chapter 3. This chapter covers all relevant aspects of
Discussion, conclusion, and future work

Subsea manipulation technology, including traditional industry standard control approaches as well as state-of-the-art research results from the academic sector.

- **Explore existing control solutions common in industrial robotics sector.** A background research related to control systems for industrial robot arms has been presented in Chapter 2. This chapter covers standard robot modelling, motion planning and control approaches, with special focus on visual servoing control solutions applicable for ROV subsea manipulation.

- **Identify which of these solutions are suitable and meet the defined requirements for application in ROV manipulator systems.** Specific requirements, implementation issues, and reflections on further development, as detailed in section 4.4, imposed certain limitations on choosing an appropriate control approach. Based on the analysis of these restrictions, a specific visual-based control approach, namely, Position-based visual servoing control, has been chosen. This control solution can be implemented on commercial ROV hydraulic manipulator systems without any modifications on the existing hardware and software. Enhanced with ANFIS motion prediction, the developed control system allows improved performance with moving targets. With collision detection and avoidance added, it has provided safer operations. Also, as it is based on pose estimation, it simplifies further development related to the fusion of camera and sonar systems.

- **Design and develop advanced manipulation control systems.** A kinematic control engine for subsea ROV manipulators has been designed and developed, and encapsulated in a new pilot control computer software. It is comprised of basic robotic functionalities as well as advanced control systems to enable fully automatic and semi-automatic execution of typical ROV manipulation tasks on stationary and moving targets. Chapter 4 has provided a detailed description of the developed systems.

- **Test the developed control solutions in simulation.** Chapter 5 has detailed simulations that have been developed and carried out using MATLAB and LabVIEW software to verify robot modelling, motion planning and (vision) control solutions. This included testing different functionalities related to joint space and Cartesian space computer control of ROV manipulators, hardware-in-the-loop simulations with a real camera system, and validation of the developed collision detection algorithm.
6.4 Contributions

- **Perform laboratory experiments to evaluate the developed control systems.** Laboratory experiments have been carried out using industrial robot arms and commercial hydraulic subsea manipulators to evaluate the designed control systems encapsulated within the developed pilot control software. These experiments covered tests of all the main algorithms developed and implemented within the scope of this thesis, i.e. testing the functionality of the kinematic control engine, visual servoing on static and moving targets, and collision detection. The results have been given in Chapter 5.

- **Verify the performance of the developed control systems in subsea field trials.** The majority of control solutions designed and developed within the scope of this thesis have been integrated on a real commercial ROV manipulator system, SMD Quasar work-class ROV called Holland 1 equipped with Schilling Titan 2 hydraulic manipulator, and tested in the real-world subsea environment in a flooded quarry in Portroe, Ireland. The results of field trials presented in Chapter 5 show successful performance and prove the validity and readiness for integration of the developed control systems into the global fleet of ROV systems for improved IRM operations. Very recently the developed systems have been integrated on a commercial Forum Energy Technologies Comanche work-class ROV, equipped with Schilling Orion 7P manipulators and tested in offshore field tests in Galway Bay.

---

6.4 Contributions

This section summarises the main contributions of the research carried out within the scope this thesis.

- **Identification of a gap in the available ROV technology capability for performing close inspection and intervention operations with manipulators in the challenging, dynamic conditions of MRE sites.** Deploying work-class ROVs in challenging MRE environment with high energy winds, currents, and waves offshore for IRM operations requires the development of robotic capability beyond the current state-of-the-art teleoperation technology that is present in the global fleet of commercial ROVs. Development of fully automatic and semi-automatic control systems that assume a supervisory role for the human pilot may prove to be a plausible solution that can compensate for the shortcomings imposed by the traditional approaches with direct human-in-the-loop control. A comprehensive review of existing ROV manipulator
systems and their limitations has been described in Chapter 3, and is the subject of a journal paper in review (Sivčev, Coleman, Omerdić, Dooly and Toal, 2018) (see Appendix G).

• **Implementation and technology transfer of industrial robotics capabilities to industry standard underwater robotic manipulators for ROVs.** Robotic functionality that is standard for industrial robot arms, including robot modelling techniques and joint space and Cartesian space motion planning and control approaches has been analysed, modified, and implemented for commercial ROV manipulator systems. The fundamental robotics principles, described in Chapter 2, have been utilised and implemented on subsea industry-standard ROV manipulator systems, and presented in section 4.3. Some initial development work on this topic has been published in a conference paper (see Appendix G) (Sivčev et al., 2015).

• **Design, development, and experimental validation of control software for ROV manipulators that enables human pilots to assume a supervisory position and utilise advanced control approaches for manipulators including enhanced manual, semi-automatic, and fully-automatic modes of operation.** Computer software that encapsulates fundamental robotics techniques for enhanced subsea manipulation has been designed and developed using LabVIEW. The results of its testing, both in simulation as well as in dry laboratory and real-world subsea environment, has been presented in section 5.2. Some initial software development has been published in a conference paper, while the experimental results of subsea field trials have been published in a journal paper (see Appendix G) (Sivčev, Rossi, Coleman, Dooly, Omerdić and Toal, 2018; Sivčev et al., 2015).

• **Design, development, and experimental validation of a novel visual servoing algorithm that allows performing fully-automatic inspection and intervention operations using industry standard hydraulic ROV manipulators on stationary and moving targets; therefore, applicable for IRM of MRE devices.** A Position-Based Visual Servoing control solution that enables carrying out typical IRM tasks on stationary targets in a fully-automatic regime has been developed. Enhanced with ANFIS motion prediction, the proposed control solution can address moving targets with increased performance, which is a small but essential step for the development of service robots capable of performing IRM operations in MRE sites. This control system, which can be readily implemented on the global fleet of ROV systems without signifi-
6.5 Future work

The research described in this thesis has been conducted within the MMRRC, recently renamed to CRIS. Some months ago, this research group acquired a brand new commercial work-class ROV—a Comanche vehicle model, manufactured by the Forum Energy Technologies. We have named her Étaín, after a heroine of Irish
Discussion, conclusion, and future work

mythology who is known as the “horse rider” (Freeman, 2017). In the Irish language (Gaeilge), white-crested sea waves are called ‘capaill bhána’ which literary means ‘white horses’. Since the acquisition of this state-of-the-art commercial system, CRIS has put significant effort to integrate vehicle navigation and control systems developed over the years to enable ROV Étaín to “ride the waves”, and thus become the most suitable ROV for MRE sites. To justify this also from the intervention operations standpoint, the author has been integrating the control systems presented in this thesis onto the ROV Étaín’s two Schilling Orion 7P manipulator systems. This path towards the commercialisation is still work in progress, and much remains to be done to have the described control approaches implemented, tested, and ready for operation on both these manipulators at the same time.

The robotic functionalities enabled by the kinematics control engine described in this thesis are to be further utilised for the automation of some typical tasks that ROV Étaín may be carrying out in different missions. Additionally, the capabilities of the developed control software are to be expanded to a dual manipulator system level to enable addressing intervention tasks that require cooperation of two manipulators. The current visualisation integrated into the software, which is limited to the manipulator, will be expanded to include the whole scene. This will allow an improved perception of the surrounding environment, which is especially significant in the impaired visibility conditions. In-house research on real-time 3D reconstruction based on camera and sonar imaging is ongoing to achieve this. The goal is to investigate carrying out operations in a virtual environment without having to observe the actual feedback of the scene from the cameras and sonars.

One of the primary future development activities related to the visual servoing system for ROV manipulators is to replace the current pose estimation algorithm that uses fiducial markers with a markerless algorithm. Such algorithm, a product of another project in CRIS, is currently in the final development stage; it also works with a monocular camera and can provide position and orientation of the target object in real time. Further ongoing development is on fusing camera and sonar imaging to compensate for low visibility issues and enable automatic manipulation in such conditions. As the system is intended for use on ROV systems, there is no reason not to investigate making use of any other possible equipment on the ROV to try to improve the visual servoing. Therefore, for redundancy, and potentially improved positioning accuracy, use of additional ROV mounted “fixed” cameras in a hybrid configuration should be considered and tested.

From the application point of view, we plan to acquire typical offshore industry ROV operated tools and replace the current mock-up valve panels with real industry standard ones and test the developed control systems in the experimental scenario
identical to the industrial one. Also, different inspection tasks utilising close video photography are to be addressed for detection of abnormalities and defects on submerged equipment. There are also other monitoring methods that do or do not include contact with targets of inspection, and that can make use of the proposed visual servoing control algorithm, not limited to cathodic measurements and eddy current inspection of structures. Some light intervention tasks such as surface cleaning and high-pressure water jetting are to be addressed as well.

There are further development and testing plans related to the manipulation control system for addressing moving targets. ANFIS enhanced visual servoing algorithm is to be tested offshore integrated on the commercial ROV system. Ongoing development on the same algorithm is on the implementation that includes ANFIS retraining in the visual servoing controller loop and testing this control strategy subsea. Different concepts on addressing moving targets are to be examined as well. Some target objects for IRM operations in the MRE sector are rigid body man-made structures. Therefore, it is worth investigating the possibility to actually measure the motion of such a device by onboard sensors and communicate that information wirelessly to the ROV to improve the automatic manipulation performance.

Additionally, the control strategy is to be expanded from the sole manipulator control to the control of the vehicle-manipulator system where the inspection and intervention tasks might be addressed with the ROV motion compensation as well. Approaching target objects from a distance in an automatic regime is another strategy that is to be implemented.

One of the principal directives that guided this research was to develop systems that can be implemented on commercial ROVs without any modifications of existing hardware or software—as such an approach has the potential to be accepted in the ROV industry. It might be worth investigating the possibility to retain the same approach but on a slightly lower level, that is, develop a low-level motion controller with improved positioning accuracy that is readily interchangeable with the existing commercial manipulator controllers of the same type. This may involve dynamics modelling and manipulator control on the hydraulic servo valve level.

Ongoing work involves integrating the developed collision detection algorithm on the ROV Étaín and performing tests at sea. This algorithm is to be tested in traditional teleoperation mode, as well as using the developed computer control software. Different modes of operation are to be tested including enhanced manual, semi-automatic, and fully-automatic regime utilising the developed visual servoing techniques. Subsea experiments will include physical simulation of intervention operations with various mock-up test panels and tool skids. Specific manipulation tasks will be repeated multiple times with and without the implementation of the
Discussion, conclusion, and future work

collision detection algorithm, where an ROV manipulator operator will focus on executing tasks with increased speed. Measuring the time required to complete the task and number of collisions during the process, and comparing them to the traditional method will reveal the actual performance of the proposed collision detection. Additionally, further collision detection algorithm development is planned to address the following aspects. Further software optimisation to reduce the execution time is to be investigated which may involve code optimisation techniques, multi-threading, and implementation on the GPU technology. Real-time visualisation as a pilot assisting tool is to be realised, which can be useful in turbid and low visibility environments; this may also involve GPU programming. Furthermore, the collision detection algorithm is to be expanded to include the use of different ROV operated tools and the detection of manipulator self-collision. On the practical side, we plan to investigate alternative methods to acquire obstacle point clouds, including camera imaging and laser scanning techniques.

There is so much still to do but for now and for this thesis—the work is finished!
References


References


References


References


References


References


Deng, L. (2004), Comparison of image-based and position-based robot visual servoing methods and improvements, University of Waterloo.


References


References


References


References


Jeppesen, L., Dam, J. and Olsen, H. O. (2005), Subsea automated ultrasonic inspection, in ‘Middle East Nondestructive Testing Conference and Exhibition’.


References


References


References


References


References


References


Appendix A

Staubli TX60 inverse kinematics — position

This appendix contains the derivation of inverse position for the Staubli TX60 robot arm using Wolfram Mathematica software.
\[ T(q_1, a_1, a_2, d_1) := \begin{bmatrix} \cos(q_1), \ -\sin(q_1), \ \sin(q_1) \cdot \cos(q_1), \ a \cdot \cos(q_1) \\ \sin(q_1), \ \cos(q_1) \cdot \cos(q_2), \ \cos(q_2) \cdot \sin(q_1), \ \{q_1, \ \sin(q_2), \ \cos(q_2), \ d_1\} \end{bmatrix}, \{0, 0, 0, 1\} \]

\[ a_1 = -P1/2; \]
\[ a_1 = 0; \]
\[ T01 = T(q_1, a_1, a_2, d_1) \]

\[ a_2 = 0; \]
\[ T12 = T(q_2, a_2, a_2, d_2) \]

\[ a_3 = P1/2; \]
\[ a_3 = 0; \]
\[ d_3 = 0; \]
\[ T23 = T(q_3, a_3, a_3, d_3) \]

\[ a_4 = P1/2; \]
\[ a_4 = 0; \]
\[ d_4 = 0; \]
\[ T34 = T(q_4, a_4, a_4, d_4) \]

\[ a_5 = 0; \]
\[ a_5 = 0; \]
\[ d_5 = 0; \]
\[ T45 = T(q_5, a_5, a_5, d_5) \]

\[ a_6 = 0; \]
\[ a_6 = 0; \]
\[ T56 = T(q_6, a_6, a_6, d_6) \]

\[ \begin{bmatrix} \cos(q_1), \ -\sin(q_1), \ \sin(q_1) \cdot \cos(q_1), \ \{0, -1, 0, d_1\} \end{bmatrix}, \{0, 0, 0, 1\} \]

\[ \begin{bmatrix} \cos(q_1), \ -\sin(q_1), \ \cos(q_1) \cdot a_2, \ \sin(q_1) \cdot \cos(q_1), \ \{0, 0, 1, d_2\} \end{bmatrix}, \{0, 0, 0, 1\} \]

\[ \begin{bmatrix} \cos(q_1), \ \sin(q_1), \ \cos(q_1) \cdot \cos(q_1), \ \{0, 0, 1, 0\} \end{bmatrix}, \{0, 0, 0, 1\} \]

\[ \begin{bmatrix} \cos(q_1), \ -\sin(q_1), \ \cos(q_1) \cdot \sin(q_1), \ \{0, 0, 1, d_3\} \end{bmatrix}, \{0, 0, 0, 1\} \]

\[ \begin{bmatrix} \cos(q_1), \ \sin(q_1), \ \cos(q_1) \cdot \sin(q_1), \ \{0, 0, 0, d_4\} \end{bmatrix}, \{0, 0, 0, 1\} \]

\[ \begin{bmatrix} \cos(q_1), \ -\sin(q_1), \ \cos(q_1) \cdot \sin(q_1), \ \{0, 0, 0, d_5\} \end{bmatrix}, \{0, 0, 0, 1\} \]

\[ \begin{bmatrix} \cos(q_1), \ \sin(q_1), \ \cos(q_1) \cdot \sin(q_1), \ \{0, 0, 0, d_6\} \end{bmatrix}, \{0, 0, 0, 1\} \]

\[ T02 = T01 \cdot T12 \]

\[ T03 = Simplify[T02, T23] \]

\[ T04 = Simplify[T03, T34] \]

\[ T05 = Simplify[T04, T45] \]
Simplify 

\[
\begin{align*}
&\{ \sin[\theta_1] (\cos[\theta_2] \cos[\theta_3] \sin[\theta_4] + \cos[\theta_4] \sin[\theta_5]) + \\
&\quad \cos[\theta_7] (\cos[\theta_8] \sin[\theta_9] + \cos[\theta_9] \cos[\theta_10] \sin[\theta_11] - \sin[\theta_11] \sin[\theta_12]) \}
\end{align*}
\]
Expand[eq1]
\[
\cos(q_1) d_2 xw + \sin(q_1) r_1 xw = -\sin(q_1) d_2 yw + \cos(q_1) r_1 yw
\]
eq1 = Collect[(d_2 xw \cos(q_1) - y_1 r_1, \cos(q_1)), \cos(q_1)] = Collect[[-x_0 r_1 \sin(q_1) - d_2 yw \sin(q_1)], \sin(q_1)]

\[
\cos(q_1) (d_2 xw - t_1 yw) = \sin(q_1) (-r_1 xw - d_2 yw)
\]
eq1 = \sin(q_1)/\cos(q_1) = (d_2 xw - y_1 r_1) / (-d_2 yw - x_0 r_1)

\[
\tan(q_1) = -\frac{d_2 xw - t_1 yw}{r_1 xw - d_2 yw}
\]
Solve[eq1, q_1] = ConditionalExpression[ArcTan[-d_2 xw + r_1 yw / r_1 xw + d_2 yw] \in \pi [1, \cos(1), \cos(1) \in \text{Integers}]]

eq1 = x_0 = T04[[1, 4]]
eq2 = y_0 = T04[[2, 4]]
eq3 = z_0 = T04[[3, 4]]
x_w = \cos(q_1) \cos(q_2) a_2 + \sin(q_1) d_2 + \cos(q_1) \sin(q_1 + q_3) d_4
y_w = \cos(q_1) \sin(q_1) a_2 + \cos(q_1) d_2 + \sin(q_1) \sin(q_1 + q_3) d_4
z_w = -\sin(q_1) a_2 + d_2 + \cos(q_1 + q_3) d_4

eq1 = (x_0)^2 + (y_0)^2 = Expand[(d_2 \sin(q_1) + \cos(q_1) (a_2 \cos(q_1) + d_4 \sin(q_1 + q_3)))^2 + (d_2 \cos(q_1) + \sin(q_1) (a_2 \sin(q_1) + d_4 \sin(q_1 + q_3)))^2]

x_0^2 + y_0^2 = \cos(q_2) a_2^2 + \cos(q_1)^2 d_2^2 + 2 \cos(q_1) \sin(q_1 + q_3) a_2 d_4 + \sin(q_1 + q_3)^2 d_4^2 + x_w^2 + y_w^2

(-d_1 + z_0)^2 = \sin(q_1)^2 a_2 + 2 \cos(q_2 + q_3) \sin(q_1 + q_3) a_2 d_4 + \cos(q_1 + q_3)^2 d_4^2

eq1 = \sin(q_1)^2 a_2 + d_2^2 + 2 a_2 \cos(q_2)^2 + 2 a_2 d_4 \cos(q_2) \sin(q_2 + q_3) + d_4^2 \sin(q_2 + q_3)^2 + (d_4 \cos(q_2 + q_3) - a_2 \sin(q_2))^2

x_0^2 + y_0^2 + (-d_1 + z_0)^2 = d_2^2 + a_2^2 \cos(q_2)^2 + 2 a_2 d_4 \cos(q_2) \sin(q_2 + q_3) + d_4^2 \sin(q_2 + q_3)^2 + (d_4 \cos(q_2 + q_3) - a_2 \sin(q_2))^2

\[
\text{FullSimplify}[\text{Solve}[\text{eq1}, \sin(q_1)]]
\]
\[
\left\{ \left[ \sin(q_1) \rightarrow \frac{-a_2^2 - d_2^2 - d_4^2 + x_0^2 + y_0^2 + (d_1 - z_0)^2}{2 a_2 d_4} \right] \right\}
\]

S3 = \sqrt{1 - (-\text{S3})^2}

\[
-\frac{a_2^2 - d_2^2 - d_4^2 + x_0^2 + y_0^2 + (d_1 - z_0)^2}{2 a_2 d_4}
\]
C3 = \sqrt{1 - (S3)^2}

\[
\text{eq1} = \sin(q_1)/\cos(q_1) = \text{S3}/\text{C3}
\]
\[
\tan(q_1) = \frac{-a_2^2 - d_2^2 - d_4^2 + x_0^2 + y_0^2 + (d_1 - z_0)^2}{2 a_2 d_4 \sqrt{1 - \frac{a_2^2 - d_2^2 - d_4^2 + x_0^2 + y_0^2 + (d_1 - z_0)^2}{4 a_2 d_4}}}
\]
Solve[equ1, q3] 

\[
\left\{ \begin{array}{l}
q_3 \rightarrow \text{ConditionalExpression}
\end{array} \right.
\]

\[
\text{ArcTan} \left[ \frac{a_2}{2 a_2 d_4 \sqrt{1 - \frac{-a_2^2 d_4^2 + x_2^2 + y_2^2}{4 d_4^2}}} \right] + \frac{d_2^2}{2 a_2 d_4 \sqrt{1 - \frac{-a_2^2 d_4^2 + x_2^2 + y_2^2}{4 d_4^2}}} + \frac{d_2^2}{2 a_2 d_4 \sqrt{1 - \frac{-a_2^2 d_4^2 + x_2^2 + y_2^2}{4 d_4^2}}} + \frac{d_2^2}{2 a_2 d_4 \sqrt{1 - \frac{-a_2^2 d_4^2 + x_2^2 + y_2^2}{4 d_4^2}}} + \cdots
\right]
\]

\[
s_3 = r_2 / r_1
\]

\[
c_3 = \sqrt{(1 - (s_3)^2)^2}
\]

equ1 = \sin[q_3] / \cos[q_3] = s_3 / c_3

\[
r_2, r_3
\]

\[
\frac{1 - \frac{r_2}{r_3}}{r_3}
\]

\[
\tan[q_3] = \frac{r_2}{1 - \frac{r_2}{r_3}}
\]

Solve[equ1, q3] 

\[
\left\{ \begin{array}{l}
q_3 \rightarrow \text{ConditionalExpression}
\end{array} \right.
\]

\[
\frac{\text{ArcTan} \left[ \frac{r_2}{1 - \frac{r_2}{r_3}} \right]}{\pi \left[ 1, C[1] \in \text{Integers} \right]}
\]

\[
r_2 = \text{FullSimplify} \left[ -a_2^2 - d_2^2 - d_4^2 + x_2^2 + y_2^2 - 2 \, d_1 \, z_4 + z_2^2 \right]
\]

\[
r_3 = 2 \, a_2 \, d_4
\]

\[
-a_2^2 - d_2^2 - d_4^2 + x_2^2 + y_2^2 + (d_1 - z_4)^2
\]

\[
2 \, a_2 \, d_4
\]

equ4 = r_2 \rightarrow \text{TrigExpand} \left[ a_2 \, \cos[q_3] + d_4 \, \sin[q_2] + q_3 \right]

\[
r_1 = \cos[q_3] \, a_2 + \cos[q_3] \, \sin[q_2] \, d_4 + \cos[q_3] \, \sin[q_2] \, d_4
\]

\[
equ4 = \text{Collect} \left[ \left[ r_2 = a_2 \, \cos[q_3] + d_4 \, \cos[q_3] \, \sin[q_2] + d_4 \, \cos[q_3] \, \sin[q_2] \right], \cos[q_3] \right]
\]

\[
r_1 = \cos[q_3] \, \sin[q_2] \, d_4 + \cos[q_3] \, \left[ a_2 + \sin[q_3] \, d_4 \right]
\]

Solve[equ4, q2] 

\[
\left\{ \begin{array}{l}
q_3 \rightarrow \text{ConditionalExpression}
\end{array} \right.
\]

\[
\text{ArcTan} \left[ \left( a_2 \, r_1 + \sin[q_3] \, d_4 \, r_1 - \sqrt{\cos[q_3]^2 \, a_2^2 \, d_4^2 + 2 \, \cos[q_3]^2 \, \sin[q_3] \, a_2 \, d_4 + \cos[q_3]^4 \, d_4^2 + \cos[q_3]^2 \, \sin[q_3]^2 \, d_4^2 - \cos[q_3]^2 \, d_4^2 \, r_1^2} \right) \right] /
\]

\[
\left( a_2^2 + 2 \, \sin[q_3] \, a_2 \, d_4 + \cos[q_3]^2 \, d_4^2 + \sin[q_3]^2 \, d_4^2 \right) \, \frac{\sec[q_3]}{\left( \left( a_2^2 + 2 \, \sin[q_3] \, a_2 \, d_4 + \cos[q_3]^2 \, d_4^2 + \sin[q_3]^2 \, d_4^2 \right) \right)} / 
\left( a_2^2 + 2 \, \sin[q_3] \, a_2 \, d_4 + \cos[q_3]^2 \, d_4^2 + \sin[q_3]^2 \, d_4^2 \right)
\]

\[
2 \, \sin[q_3] \, a_2 \, d_4 \, r_1
\]

\[
\left( a_2^2 + 2 \, \sin[q_3] \, a_2 \, d_4 + \cos[q_3]^2 \, d_4^2 + \sin[q_3]^2 \, d_4^2 \right) / 
\left( a_2^2 + 2 \, \sin[q_3] \, a_2 \, d_4 + \cos[q_3]^2 \, d_4^2 + \sin[q_3]^2 \, d_4^2 \right)
\]

\[
\left( a_2^2 + 2 \, \sin[q_3] \, a_2 \, d_4 + \cos[q_3]^2 \, d_4^2 + \sin[q_3]^2 \, d_4^2 \right) / 
\left( a_2^2 + 2 \, \sin[q_3] \, a_2 \, d_4 + \cos[q_3]^2 \, d_4^2 + \sin[q_3]^2 \, d_4^2 \right)
\]

\[
2 \, \pi \left[ 1, C[1] \in \text{Integers} \right], \left[ q_2 \rightarrow \text{ConditionalExpression} \right]
\]

\[
\text{ArcTan} \left[ \left( a_2 \, r_1 + \sin[q_3] \, d_4 \, r_1 + \sqrt{\cos[q_3]^2 \, a_2^2 \, d_4^2 + 2 \, \cos[q_3]^2 \, \sin[q_3] \, a_2 \, d_4 + \cos[q_3]^4 \, d_4^2 + \cos[q_3]^2 \, \sin[q_3]^2 \, d_4^2 - \cos[q_3]^2 \, d_4^2 \, r_1^2} \right) \right] /
\]

\[
\left( a_2^2 + 2 \, \sin[q_3] \, a_2 \, d_4 + \cos[q_3]^2 \, d_4^2 + \sin[q_3]^2 \, d_4^2 \right) \, \frac{\sec[q_3]}{\left( \left( a_2^2 + 2 \, \sin[q_3] \, a_2 \, d_4 + \cos[q_3]^2 \, d_4^2 + \sin[q_3]^2 \, d_4^2 \right) \right)} / 
\left( a_2^2 + 2 \, \sin[q_3] \, a_2 \, d_4 + \cos[q_3]^2 \, d_4^2 + \sin[q_3]^2 \, d_4^2 \right)
\]

\[
2 \, \pi \left[ 1, C[1] \in \text{Integers} \right], \left[ q_2 \rightarrow \text{ConditionalExpression} \right]
\]

\[
\text{ArcTan} \left[ \left( a_2 \, r_1 + \sin[q_3] \, d_4 \, r_1 + \sqrt{\cos[q_3]^2 \, a_2^2 \, d_4^2 + 2 \, \cos[q_3]^2 \, \sin[q_3] \, a_2 \, d_4 + \cos[q_3]^4 \, d_4^2 + \cos[q_3]^2 \, \sin[q_3]^2 \, d_4^2 - \cos[q_3]^2 \, d_4^2 \, r_1^2} \right) \right] /
\]

\[
\left( a_2^2 + 2 \, \sin[q_3] \, a_2 \, d_4 + \cos[q_3]^2 \, d_4^2 + \sin[q_3]^2 \, d_4^2 \right) \, \frac{\sec[q_3]}{\left( \left( a_2^2 + 2 \, \sin[q_3] \, a_2 \, d_4 + \cos[q_3]^2 \, d_4^2 + \sin[q_3]^2 \, d_4^2 \right) \right)} / 
\left( a_2^2 + 2 \, \sin[q_3] \, a_2 \, d_4 + \cos[q_3]^2 \, d_4^2 + \sin[q_3]^2 \, d_4^2 \right)
\]

\[
2 \, \pi \left[ 1, C[1] \in \text{Integers} \right], \left[ q_2 \rightarrow \text{ConditionalExpression} \right]
\]
\[ \text{equ4} = r_1 = r_4 \sin(q_2) + r_5 \cos(q_2) \]
\[ \text{Solve[equ4, q_2]} \]
\[ r_1 = \sin(q_2) \]
\[ r_4 = \cos(q_2) \]
\[ r_5 \]
\[
\left\{ \begin{array}{l}
q_2 \rightarrow \text{ConditionalExpression} \left[ \text{ArcTan}\left( \frac{r_1 r_5 - \sqrt{-r_4^2 r_5^2 + r_4^4 + r_5^4}}{r_4^2 + r_5^2} \right) \right] + 2 \pi C[1], C[1] \in \text{Integers} \right\},
\end{array} \right.
\]
\[
\left\{ \begin{array}{l}
q_2 \rightarrow \text{ConditionalExpression} \left[ \text{ArcTan}\left( \frac{r_1 r_5 - \sqrt{-r_4^2 r_5^2 + r_4^4 + r_5^4}}{r_4^2 + r_5^2} \right) \right] + 2 \pi C[1], C[1] \in \text{Integers} \right\}
\end{array} \right.
\]
\[
\text{Simplify} \left[ \frac{r_1}{r_4} \right]
\]
\[
\text{Simplify} \left[ \frac{r_4}{r_5} \right]
\]
\[ r_1 r_5 - \sqrt{r_4^2 \left( r_1^2 - r_4^2 + r_5^2 \right)} \]
\[ r_1^2 + r_5^2 \]
\[ \sqrt{r_4^2 \left( -r_1^2 + r_4^2 + r_5^2 \right)} \]
\[ r_4^2 + r_5^2 \]
\[ r_1^2 r_4^2 + r_5^2 r_6 \]
\[ r_4 \]
\[ r_3 \]
\[ r_7 \]
\[ r_4 \]
\[ r_1^2 r_4^2 + r_5^2 r_6 \]
\[ r_3 \]
\[ r_7 \]
\[ r_4 \]
\[ r_5 = \sqrt{r_4^2 \left( -r_1^2 + r_4^2 + r_5^2 \right)} \]
\[ r_4^2 + r_5^2 \]
\[ r_4 = \cos(q_1) \]
\[ r_4 = \cos(q_1) \]
\[ r_5 = a_2 + \sin(q_2) \]
\[ r_5 = a_2 + \sin(q_2) \]
Appendix B

Staubli TX60 inverse kinematics — orientation

This appendix contains the derivation of inverse orientation for the Staubli TX60 robot arm using Wolfram Mathematica software.
The document contains mathematical expressions and calculations. The expressions involve trigonometric functions and coefficients. The text appears to be discussing a set of equations or a mathematical model, possibly related to robotics or mechanical systems, given the context of the symbols used.
Simplify \[ R36d \]

\[
\begin{align*}
&\{r11 \cos(q1) \cos(q2 + q3) + 2r12 \cos(q2 + q3) \sin(q1) - 3r13 \sin(q2 + q3), \\
&2r12 \cos(q1) \cos(q2 + q3) + 2r22 \cos(q2 + q3) \sin(q1) - 3r23 \sin(q2 + q3), \\
&2r12 \cos(q1) \cos(q2 + q3) + 2r22 \cos(q2 + q3) \sin(q1) - 3r23 \sin(q2 + q3), \\
&r11 \sin(q2 + q3), \\
&r22 \sin(q1) - 2r13 \sin(q1), \\
&r22 \sin(q1) - 2r13 \sin(q1), \\
&r33 \cos(q2 + q3) + 2r33 \sin(q2 + q3) + 2r33 \sin(q2 + q3)
\end{align*}
\]

\[
\begin{align*}
\{\cos(q4) \cos(q5) \cos(q6) - \sin(q5) \sin(q6), & -\cos(q4) \cos(q5) \sin(q6) - \sin(q4) \cos(q5) q6, \\
\cos(q4) \cos(q5) \sin(q6) - \sin(q4) \cos(q5) q6, & -\cos(q4) \cos(q5) \sin(q6) - \sin(q4) \cos(q5) q6,
\end{align*}
\]

\[
\begin{align*}
&-\sin(q4) \cos(q5) q6, -\sin(q4) \cos(q5) q6,
\end{align*}
\]

\[
\begin{align*}
r22 \sin(q1) - 2r13 \sin(q1) - 3r33 \sin(q2 + q3)
\end{align*}
\]

\[
\begin{align*}
r33 \cos(q2 + q3) + 2r33 \sin(q2 + q3) + 2r33 \sin(q2 + q3)
\end{align*}
\]

R36eul = \[
\begin{align*}
&\{\cos(q4) \cos(q5) \cos(q6) - \sin(q5) \sin(q6), \cos(q4) \cos(q5) \sin(q6) - \sin(q4) \cos(q5) q6, \\
&\cos(q4) \cos(q5) \sin(q6) - \sin(q4) \cos(q5) q6, \cos(q4) \cos(q5) \sin(q6) - \sin(q4) \cos(q5) q6,
\end{align*}
\]

\[
\begin{align*}
&-\sin(q4) \cos(q5) q6, -\sin(q4) \cos(q5) q6,
\end{align*}
\]

equil = R36d[[1, 3]]
equil2 = R36d[[2, 3]]
r13 \cos(q2 + q3) + 2r23 \cos(q2 + q3) \sin(q1) - 3r33 \sin(q2 + q3) = \cos(q4) \sin(q6)
r23 \cos(q2 + q3) - r13 \sin(q1) = \sin(q4) \sin(q6)

equil = \[
\begin{align*}
&\{r23 \cos(q2 + q3) - r13 \sin(q1)\}/\{r23 \cos(q2 + q3) + 2r23 \cos(q2 + q3) \sin(q1) - 3r33 \sin(q2 + q3)\} = \{\cos(q4) \sin(q6)\}/\{\cos(q4) \sin(q6)\}
\end{align*}
\]

\[
\begin{align*}
r13 \cos(q2 + q3) + 2r23 \cos(q2 + q3) \sin(q1) - 3r33 \sin(q2 + q3)
\end{align*}
\]

tan(q6) = \[
\begin{align*}
&\{r23 \cos(q2 + q3) - r13 \sin(q1)\}/\{r23 \cos(q2 + q3) + 2r23 \cos(q2 + q3) \sin(q1) - 3r33 \sin(q2 + q3)\}
\end{align*}
\]

Solve[equil, q6]

\[
\begin{align*}
&\{q5 = \text{ConditionalExpression}, \text{ArcTan}[r23 \cos(q2 + q3) - r13 \sin(q1), r23 \cos(q2 + q3) + 2r23 \cos(q2 + q3) \sin(q1) - 3r33 \sin(q2 + q3)] + \pi C[1], C[1] \in \text{Integers} \}
\end{align*}
\]

equil = R36eul[[1, 3]]
equil2 = R36eul[[2, 3]]
r13 \cos(q2 + q3) + 2r23 \cos(q2 + q3) \sin(q1) - 3r33 \sin(q2 + q3) = \cos(q4) \sin(q6)
r23 \cos(q2 + q3) - r13 \sin(q1) = \sin(q4) \sin(q6)

equil = \[
\begin{align*}
&\{(r23 \cos(q2 + q3) - r13 \sin(q1))^2 + (\sin(q4) \sin(q6))^2\}^2 = \\
&(r13 \cos(q1) \cos(q2 + q3) + 2r23 \cos(q2 + q3) \sin(q1) - 3r33 \sin(q2 + q3))^2 + (r23 \cos(q2 + q3) - r13 \sin(q1))^2
\end{align*}
\]

\[
\begin{align*}
&\sin(q5)^2 = (r23 \cos(q2 + q3) - r13 \sin(q1))^2 + (r13 \cos(q1) \cos(q2 + q3) + 2r23 \cos(q2 + q3) \sin(q1) - 3r33 \sin(q2 + q3))^2
\end{align*}
\]

\[
\begin{align*}
&\sin(q6)^2 = (r23 \cos(q2 + q3) - r13 \sin(q1))^2 + (r13 \cos(q1) \cos(q2 + q3) + 2r23 \cos(q2 + q3) \sin(q1) - 3r33 \sin(q2 + q3))^2
\end{align*}
\]

\[
\begin{align*}
&\cos(q5) = \sqrt{1 - (r23 \cos(q2 + q3) - r13 \sin(q1))^2}, \cos(q6) = \sqrt{1 - (r23 \cos(q2 + q3) - r13 \sin(q1))^2} \\
&\cos(q5)/\cos(q6) = \sqrt{\{(r23 \cos(q2 + q3) - r13 \sin(q1))^2 + (r13 \cos(q1) \cos(q2 + q3) + 2r23 \cos(q2 + q3) \sin(q1) - 3r33 \sin(q2 + q3))^2\}/
\]

\[
\begin{align*}
&\sqrt{1 - (r23 \cos(q2 + q3) - r13 \sin(q1))^2), \tan(q6) = \sqrt{\{(r23 \cos(q2 + q3) - r13 \sin(q1))^2 + (r13 \cos(q1) \cos(q2 + q3) + 2r23 \cos(q2 + q3) \sin(q1) - 3r33 \sin(q2 + q3))^2\}/
\]

\[
\begin{align*}
&\sqrt{1 - (r23 \cos(q2 + q3) - r13 \sin(q1))^2, \tan(q6) = \sqrt{\{(r23 \cos(q2 + q3) - r13 \sin(q1))^2 + (r13 \cos(q1) \cos(q2 + q3) + 2r23 \cos(q2 + q3) \sin(q1) - 3r33 \sin(q2 + q3))^2\}}}
\end{align*}
\]

Solve[equil, q6]

\[
\begin{align*}
&\{q5 = \text{ConditionalExpression}, \text{ArcTan}[\{(r23 \cos(q2 + q3) - r13 \sin(q1))^2 + (r13 \cos(q1) \cos(q2 + q3) + 2r23 \cos(q2 + q3) \sin(q1) - 3r33 \sin(q2 + q3))^2\}/
\]

\[
\begin{align*}
&\{(r23 \cos(q2 + q3) - r13 \sin(q1))^2 + (r13 \cos(q1) \cos(q2 + q3) + 2r23 \cos(q2 + q3) \sin(q1) - 3r33 \sin(q2 + q3))^2\}] + \pi C[1], C[1] \in \text{Integers} \}
\end{align*}
\]

equil = R36d[[3, 1]]
equil2 = R36d[[3, 2]]
r31 \cos(q2 + q3) + r11 \cos(q4) \sin(q5) + 2r21 \sin(q5) \sin(q6) - \cos(q4) \sin(q6)
r32 \cos(q2 + q3) + r12 \cos(q4) \sin(q5) + 2r22 \sin(q5) \sin(q6) - \sin(q4) \sin(q6)
\[
equ1 = \left\{ \begin{array}{l} r_{32} \cos(q_2 + q_3) + r_{12} \cos(q_1) \sin(q_2 + q_3) + r_{22} \sin(q_1) \sin(q_2 + q_3) \\
- \left( r_{32} \cos(q_2 + q_3) + r_{12} \cos(q_1) \sin(q_2 + q_3) + r_{22} \sin(q_1) \sin(q_2 + q_3) \right) / \left( \cos(q_6) \sin(q_5) \right) \end{array} \right. \\
- r_{32} \cos(q_2 + q_3) - r_{12} \cos(q_1) \sin(q_2 + q_3) - r_{22} \sin(q_1) \sin(q_2 + q_3) \rightarrow \tan(q_6)
\]

\[
\text{Solve}\left[\text{equ1, } q_6\right]
\]

\[
\left[ q_6 \rightarrow \text{ConditionalExpression}\left[ \text{ArcTan}\left( -r_{32} \cos(q_2 + q_3) - r_{12} \cos(q_1) \sin(q_2 + q_3) - r_{22} \sin(q_1) \sin(q_2 + q_3) \right) / \left( r_{32} \cos(q_2 + q_3) + r_{12} \cos(q_1) \sin(q_2 + q_3) + r_{22} \sin(q_1) \sin(q_2 + q_3) \right) + \pi C[1], C[1] \in \text{Integers} \right] \right]
\]

\[
R36d = \{ (nx, sx, ax), (ny, sy, ay), (nz, sz, az) \};
\]

\[
\text{Simplify}\left[R36d\right]
\]

\[
\text{R36eul} = \left\{ \left( \cos(q_6) \cos(q_5) \cos(q_4) - \sin(q_6) \sin(q_5), -\cos(q_6) \cos(q_5) \cos(q_4) - \sin(q_6) \sin(q_5) \right), \left( \sin(q_6) \cos(q_5) \cos(q_4) - \cos(q_6) \sin(q_5), -\sin(q_6) \cos(q_5) \cos(q_4) - \cos(q_6) \sin(q_5) \right), \left( -\sin(q_6) \cos(q_5) \cos(q_4) + \cos(q_6) \sin(q_5), \sin(q_6) \cos(q_5) \cos(q_4) + \cos(q_6) \sin(q_5) \right) \right\}
\]

\[
equ1 = R36d[\{1, 3\}] = R36eul[\{1, 3\}]
equ2 = R36d[\{2, 3\}] = R36eul[\{2, 3\}]
ax = \cos(q_4) \sin(q_5)
ay = \sin(q_4) \sin(q_5)
\]

\[
equ1 = ay / ax = \left( \sin(q_4) \sin(q_5) \right) / \left( \cos(q_4) \sin(q_5) \right)
\]

\[
y_4 = \tan(q_4)
ax
\]

\[
\text{Solve}\left[\text{equ1, } q_4\right]
\]

\[
\left[ q_4 \rightarrow \text{ConditionalExpression}\left[ \text{ArcTan}\left( ay / ax + \pi C[1], C[1] \in \text{Integers} \right) \right] \right]
\]

\[
equ1 = R36d[\{1, 3\}] = R36eul[\{1, 3\}]
equ2 = R36d[\{2, 3\}] = R36eul[\{2, 3\}]
ax = \cos(q_4) \sin(q_5)
ay = \sin(q_4) \sin(q_5)
\]

\[
equ1 = \text{Simplify}\left[ \left( \cos(q_6) \sin(q_5) \right)^2 + \left( \sin(q_6) \sin(q_5) \right)^2 \right] = \left( ax \right)^2 + \left( ay \right)^2
\]

\[
\sin(q_5)^2 = ax^2 + ay^2
\]

\[
\cos(q_5) = \sqrt{1 - \left( ax^2 + ay^2 \right)}
\]

\[
\begin{align*}
\text{equ1} & = \sin(q_5) / \cos(q_5) = \sqrt{ax^2 + ay^2} / \sqrt{1 - \left( ax^2 + ay^2 \right)} \\
\tan(q_5) & = \sqrt{ax^2 + ay^2} / \sqrt{1 - ax^2 - ay^2}
\end{align*}
\]

\[
\text{Solve}\left[\text{equ1, } q_5\right]
\]

\[
\left[ q_5 \rightarrow \text{ConditionalExpression}\left[ \text{ArcTan}\left( \sqrt{ax^2 + ay^2} / \sqrt{1 - ax^2 - ay^2} + \pi C[1], C[1] \in \text{Integers} \right) \right] \right]
\]

\[
equ1 = R36d[\{3, 1\}] = R36eul[\{3, 1\}]
equ2 = R36d[\{3, 2\}] = R36eul[\{3, 2\}]
nz = -\cos(q_6) \sin(q_5)
sz = \sin(q_6) \sin(q_5)
\]
equ1 = $sz / (-nz) = \{\sin[q_5] \sin[q_6]\} / \{\cos[q_6] \sin[q_6]\}$

\[
\frac{sz}{nz} = \tan[q_6]
\]

Solve[{equ1, q_6}]

\[
\{q_6 \rightarrow \text{ConditionalExpression}[-\text{ArcTan}[\frac{sz}{nz}] + n C[1], C[1] \in \text{Integers}]\}
\]
Appendix C

Schilling Orion 7P inverse kinematics — orientation

This appendix contains the derivation of inverse orientation for the Schilling Orion 7P robotic manipulator using Wolfram Mathematica software.
\[ T[q_1, q_2, q_3, a_1, a_2, d_1] := \{(\cos[q_1], -\sin[q_1] \cdot \cos[a_1], \sin[q_1] \cdot \sin[a_1], a_1 \cdot \cos[a_1]), \]
\[ (\sin[q_1], \cos[q_1] \cdot \cos[a_1], -\cos[q_1] \cdot \sin[a_1], a_2 \cdot \sin[q_1]), \}
\{\theta, \sin[q_1], \cos[q_1], d_1\}, \{0, 0, 0, 0, 1\}\]
\[ a_2 = -\pi/2; \]
\[ T_{01} = T(q_1, a_1, a_2, d_1) \]
\[ d_2 = 0; \]
\[ a_2 = 0; \]
\[ T_{12} = T(q_2, a_2, a_2, d_1) \]
\[ a_2 = -\pi/2; \]
\[ d_3 = 0; \]
\[ T_{23} = T(q_3, a_3, a_3, d_3) \]
\[ a_4 = \pi/2; \]
\[ a_4 = 0; \]
\[ T_{34} = T(q_4, a_4, a_4, d_4) \]
\[ a_5 = -\pi/2; \]
\[ a_5 = 0; \]
\[ d_5 = 0; \]
\[ T_{45} = T(q_5, a_5, a_5, d_5) \]
\[ a_6 = 0; \]
\[ a_6 = 0; \]
\[ T_{56} = T(q_6, a_6, a_6, d_4) \]
\[ \{(\cos[q_1], 0, -\sin[q_1] \cdot \cos[a_1], \sin[q_1] \cdot \cos[a_1]), (0, -1, 0, 0), \{0, 0, 0, 1\}\} \]
\[ \{(\cos[q_2], 0, -\sin[q_2] \cdot \cos[a_2], \sin[q_2] \cdot \cos[a_2]), (0, 0, 1, 0), \{0, 0, 0, 1\}\} \]
\[ \{(\cos[q_3], 0, -\sin[q_3] \cdot \cos[a_3], \sin[q_3] \cdot \cos[a_3]), (0, -1, 0, 0), \{0, 0, 0, 1\}\} \]
\[ \{(\cos[q_4], 0, -\sin[q_4] \cdot \cos[a_4], \sin[q_4] \cdot \cos[a_4]), (0, 1, 1, 0), \{0, 0, 1, 1\}\} \]
\[ \{(\cos[q_5], 0, -\sin[q_5] \cdot \cos[a_5], \sin[q_5] \cdot \cos[a_5]), (0, -1, 0, 0), \{0, 0, 1, 1\}\} \]
\[ \{(\cos[q_6], 0, -\sin[q_6] \cdot \cos[a_6], \sin[q_6] \cdot \cos[a_6]), (0, 1, 0, 0), \{0, 0, 1, 1\}\} \]
\[ T_{02} = T_{01} \cdot T_{12} \]
\[ T_{03} = \text{Simplify}[T_{02} \cdot T_{23}] \]
\[ R_{04} = (T_{03}[1, 1], T_{03}[1, 2], T_{03}[1, 3]), (T_{03}[2, 1], T_{03}[2, 2], T_{03}[2, 3]), (T_{03}[3, 1], T_{03}[3, 2], T_{03}[3, 3]) \]
\[ R_{05} = \text{Simplify}[\text{Inverse}[R_{03}]] \]
\[ R_{06} = \text{Simplify}[\text{Inverse}[R_{03}]] \]
\[ R_{07} = \{(r_{11}, r_{12}, r_{13}), (r_{21}, r_{22}, r_{23}), (r_{31}, r_{32}, r_{33})\} \]
\[ R_{08} = \{(r_{11}, r_{12}, r_{13}), (r_{21}, r_{22}, r_{23}), (r_{31}, r_{32}, r_{33})\} \]
\[r_{13} \cos(q_1) \cos(q_2) \sin(q_1) - r_{12} \cos(q_2) \sin(q_1) - r_{32} \cos(q_2) \sin(q_1) - (r_{13} \cos(q_1) + r_{23} \sin(q_1)) \sin(q_2) + q_3]\]
eq1 = (-r32 Cos[q2] + q3) - r12 Cos[q1] Sin[q1] + r22 Sin[q1] Sin[q2] + q3) / 
(r31 Cos[q2] + q3 + r11 Cos[q1] Sin[q2] + q1) Sin[q2] + q2) = (Sin[q2] Sin[q2]) / (Cos[q2] Sin[q2])

-r32 Cos[q2] + q3) - r12 Cos[q1] Sin[q2] + q1 - r22 Sin[q1] Sin[q2] + q2) = Tan[q2]

r31 Cos[q2] + q3] + r11 Cos[q1] Sin[q2] + q2) = Tan[q2]

Solve[eq1, q4]

Abs[eq1]


Solve[eq1, q4]

Abs[eq1]


Solve[eq1, q4]

Abs[eq1]


Solve[eq1, q4]

Abs[eq1]


Solve[eq1, q4]

Abs[eq1]


Solve[eq1, q4]

Abs[eq1]


Solve[eq1, q4]

Abs[eq1]


Solve[eq1, q4]

Abs[eq1]


Solve[eq1, q4]

Abs[eq1]


Solve[eq1, q4]

Abs[eq1]
\text{Solve}\{\text{equ1}, q_6\}

\{q_6 \rightarrow \text{ConditionalExpression}\left\{ -\text{ArcTan}\left(\frac{5}{n_{z}}\right) + n \cdot C[1], C[1] \in \text{Integers} \right\} \}
Appendix D

Schilling Orion 7P actuator kinematics

This appendix contains the computation of joint actuator kinematics for the Schilling Orion 7P robotic manipulator using MATLAB software.
Azimuth - Joint 1

% The lengths are expressed in inches
% The angles are expressed in degrees

A = [1.705 3.400];
B = [1.542 4.891];
C = [-7.025 3.200];

AC_x = abs(C(1) - A(1));
AC_y = abs(C(2) - A(2));

AB_x = abs(A(1) - B(1));
AB_y = abs(A(2) - B(2));

b = sqrt(AC_x^2 + AC_y^2)
c = sqrt(AB_x^2 + AB_y^2)

beta = acosd(AB_y/c)
gamma = acosd(AC_x/b)

b =
8.7323
c =
1.4999
beta =
6.2389
gamma =
1.3124

Published with MATLAB® R2014a
Shoulder - Joint 2

% The lengths are expressed in inches
% The angles are expressed in degrees

A = [4.455 3.101];
B = [5.237 1.125];
C = [14.511 4.978];

AC_x = abs(C(1) - A(1));
AC_y = abs(C(2) - A(2));

AB_x = abs(A(1) - B(1));
AB_y = abs(A(2) - B(2));

b = sqrt(AC_x^2 + AC_y^2)
c = sqrt(AB_x^2 + AB_y^2)

beta = acosd(AB_y/c)
gamma = acosd(AC_x/b)

b =
   10.2297

c =
   2.1251

beta =
   21.5911

gamma =
   10.5729

Published with MATLAB® R2014a
Elbow - Joint 3

% The lengths are expressed in inches
% The angles are expressed in degrees

A = [26.101 8.591];
B = [28.006 7.981];
C = [16.082 5.397];

AC_x = abs(C(1) - A(1));
AC_y = abs(C(2) - A(2));

AB_x = abs(A(1) - B(1));
AB_y = abs(A(2) - B(2));

b = sqrt(AC_x^2 + AC_y^2)
c = sqrt(AB_x^2 + AB_y^2)

beta = acosd(AB_y/c)
gamma = acosd(AC_x/b)

b =
   10.5158

c =
   2.0003

beta =
   72.2444

gamma =
   17.6820

Published with MATLAB® R2014a
Wrist Pitch - Joint 5

% The lengths are expressed in inches
% The angles are expressed in degrees

A = [45.765 3.101];
B = [46.515 4.400];
C = [38.321 7.672];

AC_x = abs(C(1) - A(1));
AC_y = abs(C(2) - A(2));

AB_x = abs(A(1) - B(1));
AB_y = abs(A(2) - B(2));

b = sqrt(AC_x^2 + AC_y^2)
c = sqrt(AB_x^2 + AB_y^2)

beta = acosd(AB_y/c)
gamma = acosd(AC_x/b)

b = 8.7354
c = 1.5000
beta = 30.0007
gamma = 31.5521

Published with MATLAB® R2014a
Appendix E

Schilling Orion 7P joint offset values

This appendix contains the computation of joint offset values for the Schilling Orion 7P robotic manipulator using MATLAB software.
% The lengths are expressed in inches
% The angles are expressed in degrees

A = [26.101 8.591];
B = [4.455 3.101];
C = [45.765 3.101];
d4 = 19.708;
a3 = 5.330;
b = 20.416;

a_x = abs(B(1) - C(1));
a_y = abs(B(2) - C(2));

c_x = abs(A(1) - B(1));
c_y = abs(A(2) - B(2));

a = sqrt(a_x^2+a_y^2);
c = sqrt(c_x^2+c_y^2);

alpha = atan2d(a3,d4)
gamma = acosd((b^2+a^2-c^2)/(2*b*a))
beta = acosd((c^2+a^2-b^2)/(2*c*a))

alpha = 
   15.1335

gamma = 
   15.5992

beta = 
   14.2316

Published with MATLAB® R2014a
Appendix F

Models of robotic manipulators

This appendix contains models of the Staubli TX60, Schilling Titan 2, and Schilling Orion 7P robotic manipulators constructed using MATLAB software.
Staubli TX60 manipulator model

%mdl_TX60 Create model of Staubli TX60 manipulator
%
%      mdl_TX60
%
% Script creates the workspace variable TX60 which describes the
% kinematic characteristics of a Staubli TX60 manipulator
% using standard DH conventions.

deg = pi/180;

L(1) = Revolute('d', 0.375, 'a', 0, 'alpha', -pi/2,...
     'qlim', [-180 180]*deg );

L(2) = Revolute('d', 0.02, 'a', 0.29, 'alpha', 0,...
     'qlim', [-127.5 127.5]*deg );

L(3) = Revolute('d', 0, 'a', 0, 'alpha', pi/2, ...
     'qlim', [-142.5 142.5]*deg );

L(4) = Revolute('d', 0.31, 'a', 0, 'alpha', -pi/2, ...
     'qlim', [-270 270]*deg );

L(5) = Revolute('d', 0, 'a', 0, 'alpha', pi/2, ...
     'qlim', [-122.5 133.5]*deg );

L(6) = Revolute('d', 0.07, 'a', 0, 'alpha', 0, ...
     'qlim', [-270 270]*deg );

TX60 = SerialLink(L, 'name', 'TX60', ...
     'manufacturer', 'Staubli');

TX60.model3d = 'Staubli/TX60';

TX60.tool = transl(0,0,0);
TX60.base = transl(0,0,0);

TX60.offset(2) = - pi/2;
TX60.offset(3) = pi/2;

Published with MATLAB® R2014a
Schilling Titan 2 manipulator model

%mdl_T2 Create model of Schilling Titan 2 manipulator
%
% mdl_T2
%
% Script creates the workspace variable T2 which describes the
% kinematic characteristics of a Schilling Titan 2 manipulator
% using standard DH conventions.

deg = pi/180;

L(1) = Revolute('d', 0.195, 'a', 0.121, 'alpha', -pi/2, ...
 'qlim', [-120 120]*deg);

L(2) = Revolute('d', 0, 'a', 0.851, 'alpha', 0, ...
 'qlim', [-78 42]*deg);

L(3) = Revolute('d', 0, 'a', 0.483, 'alpha', 0, ...
 'qlim', [-106 164]*deg);

L(4) = Revolute('d', 0, 'a', 0.133, 'alpha', pi/2, ...
 'qlim', [-90 90]*deg);

L(5) = Revolute('d', 0, 'a', 0, 'alpha', pi/2, ...
 'qlim', [-90 90]*deg);

L(6) = Revolute('d', 0.339, 'a', 0, 'alpha', 0, ...
 'qlim', [-180 180]*deg);

T2 = SerialLink(L, 'name', 'Titan 2', ...
 'manufacturer', 'Schilling');

T2.model3d = 'Schilling/T2';

T2.tool = transl(0,0,0);
T2.base = transl(0,0,0);

T2.offset(2) = 0;
T2.offset(5) = pi/2;

Published with MATLAB® R2014a
%mdl_O7P Create model of Schilling Orion 7P manipulator
%
mdl_O7P
%
% Script creates the workspace variable O7P which describes the
% kinematic characteristics of a Schilling Orion 7P manipulator
% using standard DH conventions.

deg = pi/180;
inch = 0.0254;

L(1) = Revolute('d', 3.101*inch, 'a', 2.750*inch, 'alpha', -pi/2, ...
          'qlim', [-60 60]*deg);

L(2) = Revolute('d', 0, 'a', 22.331*inch, 'alpha', 0, ...
          'qlim', [-90 32]*deg);

L(3) = Revolute('d', 0, 'a', 5.330*inch, 'alpha', -pi/2, ...
           'qlim', [-133 150]*deg);

L(4) = Revolute('d', 19.708*inch, 'a', 0, 'alpha', pi/2, ...
             'qlim', [-94 26]*deg);

L(5) = Revolute('d', 0, 'a', 0, 'alpha', -pi/2, ...
              'qlim', [-135 135]*deg);

L(6) = Revolute('d', 16.24*inch*inch, 'a', 0, 'alpha', 0, ...
             'qlim', [-180 180]*deg);

O7P = SerialLink(L, 'name', 'Orion 7P', ...
              'manufacturer', 'Schilling');

O7P.model3d = 'Schilling/O7P';

O7P.tool = transl(0,0,0);
O7P.base = transl(0,0,0);

O7P.offset(2) = -14.232*deg;
O7P.offset(3) = -44.964*deg;
O7P.offset(5) = -30.732*deg;

Published with MATLAB® R2014a
Appendix G

Key published papers

The papers listed below are the result of the research work described in this thesis. Papers 1 and 2 have been published in Conference Proceedings. Paper 2 has also been published in a book chapter. Papers 3–5 have been published in Journals. Papers 6 and 7 have been submitted for Journal publication and paper 8 for Conference Proceedings. Paper 9, a result of associated collaborative research work of the author with the National University of Ireland, Maynooth, is also submitted for Journal publication. The last five papers are currently in review.


Key published papers


