Department of Mechanical, Materials and Manufacturing Engineering

Faculty of Engineering

Control of Navigation and Feeding Actions of a Continuum Arm for In-Situ Repair

By

David Palmer
Master of Cybernetics

Thesis submitted to the University of Nottingham

for the degree of Doctor of Philosophy

January 2016

Abstract

Maintenance and repair of large or complex structures is often an afterthought post-production. Conventional restoration techniques involve removing components and performing operation in a workshop. For many of these structures, it is not feasible. As such, an EU Project was setup to create a generic in-situ repair tool. This was comprised of two individual robots. This study follows the development of the control algorithms for one of the systems, devised as Snake Arm, which is a sub-set of the field of hyper-redundant manipulators. It consists of 12 sections, which means there are 25 degrees of freedom including the feed-in mechanism. The robot has been demonstrated for two main scenarios: the inspection and repair of a gas turbine engine whilst it is still attached to an aeroplane wing, and inspection of an RPV head inside a nuclear reactor while mounted on the other robot.

Through this work, two algorithms are created to overcome the challenges faced. Both are designed in a way that the kinematic and behaviour of the robot can be interchanged for different systems. To navigate the Snake Arm, an algorithm is required to manage the shape as the tip follows a path. The aim being to prevent deviation from this guided route while continuing the advancement. Previous work by others was specific to a particular design or relied on environmental data in their approach. This is combined with another algorithm to provide a feed-in movement by uncoiling the Snake Arm from a rotating drum. Not only does this require a smooth action to translate the pose from coiled to straight, but the handover point must remain on the expected trajectory for the navigation algorithm. No prior work was found on an active, three-dimensional uncoiling process.

Acknowledgements

The work entailed in this thesis would not have been feasible without the team involved at University of Nottingham. Especially Mr. Xin Dong, who is responsible for the design of the Snake Arm.

Prof Dragos Axinte, my principal supervisor, has provided drive, care and advice throughout the project. The robot would never have been completed without his support. Rolls-Royce is represented by Dr James Kell, who has been supportive and understanding through the research approach and provided several of the 3D printed parts; including the engine model.

My family have been to a great aid during the last month of writing, not only with regular tea deliveries; my brother, Dr Tim Palmer, deserves special mention as he has spent many hours helping proof read the content in the final hours.

Mr. Stewart Lowth has been a constant approachable ear to listen to my concerns and problems during the project; offering advice and his experience when he could. Dr Christopher Windmill has been a valued friend throughout the process. Providing insight and multiple reminders to stop and relax.

Finally mention goes to my long suffering Fiancée, Tanya, who has put up with endless nights of me being locked away on my computer; only to have me pass out without a word.

Publications Arising from the Thesis

Paper:

Real-Time Method for Tip Following Navigation of Continuum Snake Arm Robots, David Palmer, Salvador Cobos-Guzman, Dragos Axinte, Journal of Robotics and Autonomous Systems, vol. 62, pp. 1478-1485, 2014

Conference Workshop:

MiRoR workshop at the *2015 IEEE/RSJ International Conference on Intelligent Robots and Systems* in Hamburg, Germany, 28th September 2015.

Table of Contents

Declaration of Originality..... **Error! Bookmark not defined.**

Abstract..... ii

Acknowledgements iii

Publications Arising from the Thesis iv

List of Figures..... xii

List of Tables xviii

1 Introduction 1

1.1 Background of field and problem 1

1.2 Problem Definition..... 3

1.1.1 Repair of Gas Turbine Engine 3

1.1.2 Inspection within Nuclear Power Plant 4

1.3 Aims and Objectives 5

1.4 Terminology and Nomenclature..... 6

2 Literature Survey 10

2.1 What are Hyper-Redundant Manipulators?..... 10

2.1.1 Types of Usage: Elephant Trunks versus Snake Arms..... 11

2.1.2 Design Architecture: Discrete, Serpentine or Continuum 13

2.1.3 Methods and Mechanisms for Actuation 14

2.2 Noteworthy Examples 17

2.2.1 Gregory S. Chirikjian 17

2.2.2 The Elephant Trunk and OctArm 17

2.2.3 OC Robotics 19
2.3 Control of movement 21
2.3.1 Navigation 21
2.3.2 Feed-In 23
2.4 User Interface 26
2.5 Chapter Summary 27
2.6 Academic Challenges 27
3 Research Approach 29
3.1 Low-Level Control 29
3.2 Tip Following 30
3.3 Feed-In 30
3.4 Combined System 31
3.5 Prototyping 31
3.5.1 Prototype 1: 3-Cable, Two-Section Snake Arm 31
3.5.2 Prototype 2: 4-Cable, Twelve-Section Snake Arm 32
3.5.3 Prototype 3: Three-Section Snake Arm 33
3.5.4 Prototype 4: Twelve-Section Snake Arm with Rotating Drum 34
3.6 Chapter Summary 35
4 Cable Kinematics and Low Level Control 36
4.1 Cable Kinematics 36
4.1.1 3-Cables 37
4.1.2 4-Cables 39

4.1.3 Dealing with Multiple Sections 40

4.1.4 Pros and Cons of 3 Cables or 4 Cables 43

4.2 Cable Kinematics for Different Designs 44

4.2.1 Partially Rigid Continuum Spine 44

4.2.2 Segregated Bending for Final Prototype 47

4.3 Observations in Cable Kinematics 48

4.4 Hardware for Actuation 49

4.4.1 Cable Actuation 50

4.4.2 Main Controller 54

4.5 Control Strategy 55

4.5.1 Bang-Bang 55

4.5.2 Proportional Control 56

4.5.3 Proportional Control with Pseudo-Derivative 58

4.6 Manual Control 59

4.6.1 Graphical Representation for User Interpretation 59

4.6.2 Direct Control 60

4.6.3 Implementation 62

4.7 Accuracy and Repeatability Trials on Third Prototype. 64

4.7.1 Methodology 64

4.7.2 Accuracy Trials 65

4.7.3 Repeatability Trials. 67

4.8 Chapter Summary 68

5 Tip Following 70

5.1 Introduction 70

5.2 Geometric Kinematics 71

5.2.1 Representing a Single Section 71

5.2.2 Recursive Representation of the Multi-Section Snake Arm 74

5.3 General Method and Alternative Approaches 75

5.3.1 Follow the Leader 76

5.3.2 Jacobian 77

5.4 Optimisation Operation 79

5.4.1 Constraints 80

5.5 Objective Function 81

5.5.1 Possible Methods for Comparing the Path 81

5.5.2 Implementing Many-to-One 84

5.6 Streamlining the Algorithm 85

5.6.1 Feed-In Approximation 85

5.6.2 Dealing with variable feed-rates 85

5.6.3 Performance Weighting 86

5.7 Simulation Tests and Results 86

5.7.1 Generic Profiles Tests of Tip Following 87

5.7.2 Full-Length Trials 92

5.8 Physical Trials with Second Prototype 98

5.9 Chapter Summary 101

6 Feed-In 102
6.1 Introduction 102
6.2 Specification 102
6.2.1 Coiled Profile 103
6.2.2 Optimising the Profile 105
6.3 Proposed Uncoiling Methods 105
6.3.1 Fixed Drum 106
6.3.2 Active Drum 107
6.4 Planning the movement 110
6.4.1 Initial bend 110
6.4.2 Vertical movement 111
6.4.3 Actions required 113
6.5 Optimising the Movement 114
6.5.1 Objective Function 114
6.5.2 Setup 115
6.5.3 Fixed Step Validation 116
6.6 Non-Continuous Step Function 117
6.7 Results 119
6.7.1 Non-Continuous Function Performance 120
6.7.2 Simulation 121
6.7.3 Physical Trials 123
6.8 Chapter Summary 125

7 Combined System 127

7.1 Introduction 127

7.1.1 Hardware 127

7.1.2 Software 128

7.2 Combining the Feed-In with Tip Following 128

7.2.1 Linear Stage 129

7.3 Communicating with High-Level Control 131

7.3.1 Commanding the Low-Level and Tip Following 131

7.3.2 Features of High-Level Controller 132

7.4 User Interaction with Tip Following 134

7.4.1 Endoscope Mode 134

7.4.2 “GoHome” Mode 135

7.4.3 Branching Mode 136

7.5 Including the Machining Capabilities 137

7.6 Results 138

7.6.1 Simulation of Tip Following with HLC 138

7.6.2 Tip Following through Engine Model for Defect Inspection 145

7.6.3 Inspection of RPV head co-operating with walking hexapod 148

7.6.4 Machining Trials 149

7.7 Chapter Summary 150

8 Conclusions 152

8.1 Low-Level Control 152

8.2 Navigation 152

8.3 Coiling Process 153

8.4 Final Prototype 153

8.5 Future Work 154

8.5.1 Mechanical & Control 154

8.5.2 Tip Following 154

8.5.3 Feed-In 155

8.5.4 Combined 155

9 Bibliography 152

List of Figures

Figure 1-1: Concept art from the MiRoR project proposal 2

Figure 1-2: Common terminology used to describe parts of the system 8

Figure 2-1: Examples of Different Kinds of Manipulator A) Underactuated: Segway i2 Personal Transporter [5] B) Critically Actuated: A Simple 3-Axis CNC Machine [6] C) Redundantly Actuated: 5-Axis Robotic Arm by ST Robotics [7] 10

Figure 2-2: Examples of A) an Octopus Arm Grasping and Stacking Cones [9] and B) a Snake Arm Exploring and Inspecting a Makeshift Environment [10] 11

Figure 2-3: The ACM-R5 is an Impressive Example of a Snake Robot. Created by HiBot, this robot can Self-Propel on Both Land and Water [17] 12

Figure 2-4: Examples of different backbone designs 13

Figure 2-5: Examples of Cable Driven Arms A) Two-Section Discrete Backbone [26] B) Two-Section Continuum Backbone [25] 15

Figure 2-6: Examples of Pneumatically Driven Manipulators A) Cut-Away of OctArm VI Design Concept B) Annotated Picture of OctArm VI [33] C) Festo's Bionic Handling Assistant [32] 16

Figure 2-7: Alternate Actuation Methods A) Manipulator Using Shape Memory Alloys [37] B) an Active Cannula [38] 16

Figure 2-8: Work at Clemson University A) Elephant Trunk Robot in Hooked Configuration [11] B) OctArm VI Grasping a Spherical Ball [9] 18

Figure 2-9: Explorer Range from OC Robotics [10] A) Snake Arm Moving with Actuation Pack Exposed B) Close Up of Snake Arm Joints C) Examples of Available Tooling 19

Figure 2-10: OC Robotics Bespoke Solutions [10] A) LaserSnake Cutting through Steel Pipes B) Test Firing of JetSnake C) SAFIRE Remote Vehicle in its Packed State 20

Figure 2-11: Diagrams of tunnelling technique A) Tensor Manipulator Constrained by a System of Tunnels [45] B) Continuum arm with discreet sensors [73] 21

Figure 2-12: Examples of Feed-In Techniques using Independent Hardware A) Tracked Robotic Chassis [9] B) Linear Track [10] C) Industrial Robotic Arm [10] 24

Figure 2-13: Examples of Feed-In with a Static Base Position using A) a Loop [45] or B) a Coiled Stack [75] *24*

Figure 2-14: Examples of Coiled Snake Arms by OC Robotics [10] A) an Internal Coiled Snake B) A Planar Snake for Inspection *25*

Figure 2-15: Other Work of Benefit to Feed-In Research A) Tree-Climbing Snake [16] B) Tendril Robot [77] *26*

Figure 3-1: Infographic to explain the dependencies of each element of the final system and their hierarchal relationship within *29*

Figure 3-2: Prototype 1 A) the complete system B) electronics of demonstrator, a NI 9606 sbRIRIO and 6 stepper motor drivers C) a simple Two-Section Snake Arm consisting of a central backbone and deformed by steel cables *32*

Figure 3-3: Second Prototype, electronic in actuation pack (left) last four sections (right) *33*

Figure 3-4: Third Prototype, actuation pack electronics (top) and machining spindle (bottom) *34*

Figure 3-5: Final Prototype *35*

Figure 4-1: Graphical representation to show why guide discs are required *37*

Figure 4-2: A simple continuum backbone design using 3-Cable actuation A) illustrative profile B) top down view of the cable arrangement and assignment *37*

Figure 4-3: Kinematics of a single section *38*

Figure 4-4: A simple four cable design A) Trimmetric view B) Cable assignment *39*

Figure 4-5: Cables run through their preceding sections, meaning what action is taken must be counteracted by subsequent sections *41*

Figure 4-6: Cables cannot run through each other, each section requires its own set of holes shifted by one of the methods shown *42*

Figure 4-7: Pseudo code to calculate the combined length change for each section, where i is the index of the section, and j is the index for the phase and radii settings. f is the kinematic function and xL, yL are arrays of the combined length changes. *43*

Figure 4-8: A Partially Rigid Backbone design A) Trimetric profile B) Side Cutaway *44*

Figure 4-9: geometric representation of the design to aid mathematical explanation 45

Figure 4-10: Comparison of change in cable length to validate the assumption in the trapezium kinematics 46

Figure 4-11: Graphical representation of the segregated design used in the final prototype .. 47

Figure 4-12: Exaggerated concept to demonstrate behaviour of the segmented design 47

Figure 4-13: Graph to demonstrate how the hole radius affects the force and the movement required to actuate a single section. 48

Figure 4-14: Layout of the electrical hardware for the final prototype 49

Figure 4-15: The misalignment within the first demonstrator A) Depiction of the linear actuation in operation B) Mathematical geometry of the behaviour 50

Figure 4-16: Different methods of spooling cable A) Horizontal Spiral B) Vertical Helix 51

Figure 4-17: A) Comparison of Polynomial Approximation for Spiral Behaviour, B) Function Applied to get the change in Rotation for the desired change in Length. 53

Figure 4-18: Illustration of how the cable spooled shifts between the extremes of the range. 53

Figure 4-19: Bang-Bang Control A) Position vs. Time B) Control Setup 56

Figure 4-20: Proportional Control A) Simulated Behaviour B) Control Loop 57

Figure 4-21: Proportion-Derivative Control A) Simulated Behaviour B) Control Loop 58

Figure 4-22: Comparison of how similar A) Compass Plot and (B) Top-Down view of single section are 59

Figure 4-23: A 5-section snake represented as A) a stacked series of compass plots B) the same snake simulated in 3D showing the front and side view 60

Figure 4-24: Possible control schemes for two sections, 1st axis in red and 2nd in blue A) Controls mapped to a joystick B) Controls mapped to a gamepad 61

Figure 4-25: Examples of non-conventional controllers that could be implemented for the control of snake arm: A) Novint Falcon B) 3Dconnexions SpaceMouse Pro C) Logitech G13 Gaming Keypad 61

Figure 4-26: Experimental setup of video gauge with the Third Prototype 64

Figure 4-27: Accuracy Trials, where red diamonds indicate the expected position and the asterisks represent the actual. *65*

Figure 4-28: Repeatability trials, graph showing the displacement from the average point. The black dotted circle represents a 0.5mm radius and the magenta circle is 2 standard deviations from the mean *67*

Figure 5-1: Kinematic parameters of a single section *72*

Figure 5-2: MATLAB code used to create an Arc *73*

Figure 5-3: Graph of a single section created using MATLAB code S = 100mm, ϑ = 45° and ϕ = 45° *74*

Figure 5-4: Graphical Representation of a Three Section Snake. S = [100mm 100mm 100mm], ϑ = [45° 45° 90°] and ϕ = [45° 45° 225°] *75*

Figure 5-5: Example scenario midway through navigation to illustrate the general requirements *76*

Figure 5-6: Graph Showing the Overshoot Caused by the Non-Linear Relationship *77*

Figure 5-7: Early tests on Jacobian method, initial tests were promising (a) but full length simulations created large quantities of drift (b) *79*

Figure 5-8: Graphical representation of search space available to one section assuming ϑ = 45° *80*

Figure 5-9: Illustration of Direct Comparison method and misalignment causing incorrect performance analysis *82*

Figure 5-10: Diagram to help explain the Many-to-One approach *83*

Figure 5-11: 90 degree bend profiles, in black, of different curvatures. Overlaid is every pose used in the Tip Following process to display the deviations. *89*

Figure 5-12: S-Bend profiles, in black, of different angles. Overlaid, in green, is all the poses used in the Tip Following process. *91*

Figure 5-13: Overlay of all poses used for Tip Following within a generic scenario. *92*

Figure 5-14: Performance data for 25 random runs of the proposed method *94*

Figure 5-15: Path Deviation Outliers at their most deviated step. *96*

Figure 5-16: Each tip error outlier at the step of greatest error 97

Figure 5-17: Time Series of Poses during Tip Following Trial 99

Figure 5-18: Extended bump path test at the step with the maximum deviation. Graphical comparison of tip following path (blue) and the physical robot against the trajectory in black. 100

Figure 6-1: The actuation pack for the Final Prototype, Concept (Left) Actual (Right) 103

Figure 6-2: Comparison of Forward Kinematic attempts to match coil profile 104

Figure 6-3: Optimised Pose to match the profile of the helical channel 105

Figure 6-4: Non-Coiled Method for Feed-In using Spare Length of Arm 106

Figure 6-5: Passive Uncoiling Methods 107

Figure 6-6: Uncoiling a) Vertically B) 45degrees C) At Pitch angle 108

Figure 6-7: Horizontal uncoiling at three different heights 109

Figure 6-8: Base section coiling on drum and the Lateral Error caused in the process 110

Figure 6-9: Side View of Initial Bend 111

Figure 6-10: Stages of the Vertical Bend A) Two-Section involved in making an S-Bend B) The current pair reached the split point, the next section (shown in cyan) is considered for future use C) First section begins straightening (orange), the other two sections make an extended curve. D) First Section is now straight, with the remaining sections in an equivalent state to A) 113

Figure 6-11: Formulation of the four criteria to evaluate the optimisation solution 115

Figure 6-12: Errors before and after the optimised S-Bend 116

Figure 6-13: TF_READY pose and the steps to pack the remaining sections A) Straighten all sections that have not been coiled B) Wrap remain sections into their coiled pose. 117

Figure 6-14: Flowchart for the Non-Continuous Step function 118

Figure 6-15: Computation time versus the requested position 120

Figure 6-16: Requested Position versus the Actual Position of handover point from the feed-in function 121

Figure 6-17: Timeseries of an uncoiling process. 122

Figure 6-18: Point clouds of the drift and angular errors after the optimisation process. *123*

Figure 6-19: Final Prototype coiled to the pose TF_READY *124*

Figure 6-20: Timeseries of uncoiling action performed on Final Demonstrator *124*

Figure 7-1: Hardware connection topography for the final prototype *127*

Figure 7-2: Software connections *128*

Figure 7-3: Flowchart of Tip Following algorithm, and the steps added to make the combined procedure *129*

Figure 7-4: Final Prototype mounted on Linear Stage *130*

Figure 7-5: Early experiment of the combined uncoiling and Tip Following algorithm *130*

Figure 7-6: Diagram of the effect the torus constraint has on the path planning *133*

Figure 7-7: CAD visualisation used for generating paths based on the goal position and environment data *133*

Figure 7-8: Interface for Endoscopic navigation and illustration of the process *135*

Figure 7-9: Branching Mode, rather than retreating to O the GoHome can be stopped at B and a new path can be planned to C *136*

Figure 7-10: Machining Process *138*

Figure 7-11: Side view and Top view of first action *139*

Figure 7-12: Commanded CAD movement in the visualization, planned motion (left) and resulting action (right) *139*

Figure 7-13: Result series after Endoscopic navigation *140*

Figure 7-14: Commanded Action and the resulting trajectory in the visualization *141*

Figure 7-15: Results after planning "GoHome" action *141*

Figure 7-16: Visualization of the execution interrupted (left) and the new path generated (right) *142*

Figure 7-17: Poses used to navigate the Branching action *142*

Figure 7-18: Combined overlay of poses for the final action of the simulation *143*

Figure 7-19: Summary of the Path Deviation and Tip Errors generated through the simulation *144*

Figure 7-20: Snake Arm in-situ of 1:1 scale model of a gas turbine engine. 145

Figure 7-21: End-effector point-of-view of the three defects. 146

Figure 7-22: Timeseries of the GoHome action from Defect 1 147

Figure 7-23: Snake Arm mounted on top of Walking Hex as it enters the mock-up environment 148

Figure 7-24: Before and after of a machining operation on the leading edge of IP1 149

Figure 7-25: Machining on titanium blade, mid-process 150

List of Tables

Table 1-1: Nomenclature 9

Table 4-1: Summary of position accuracy results 66

Table 4-2: Summary of Repeatability Trial 68

Table 5-1: Results Summary of Short Tests. 87

Table 5-2: Result Data from Random 10-Section Paths 95

1 Introduction

1.1 Background of field and problem

The manufacturing industry has been focused on robotic systems and machine tools for creating new products or for repetitive labour. However little work has been done in developing robotic tools for post-production tasks, such as maintenance or repair. There are several industry sectors that rely on post-production for safety critical operations. A few examples:

- Where continuous safe running is required in complex installations nuclear power plants, highly engineered products like jet engines or safety critical beams in construction
- Large costs accompanying down-time such as shutting down power plants, or grounding an aircraft
- Concerns with health and safety while performing post-production operations from working at height, cramped spaces or exposure to chemical and radiation hazards

Many of the conventional methods for maintenance and repair involve taking parts off-site to a workshop, but given the size and complexity in these sectors this is often not economically viable or in fact impossible. Current in-situ repair tends to use 'job-customised' equipment that are often lack flexibility and can be cumbersome to set up.

An EU project was formed to tackle these problems. The title for this collaborative consortium is: Miniaturised Robotic systems for holistic in-situ Repair and maintenance works in restrained and hazardous environments (MiRoR). Partners in the consortium include: Rolls-Royce Aerospace, Rolls-Royce Civil Nuclear, Intelligent NDT, Petrom, Acciona, ETHz, IPA and Tekniker.

The MiRoR project aims to provide a solution to these scenarios by designing and implementing a Miniature-Robotised Machine tool (Mini-RoboMach), which is to be in the form of a walking hexapod machine tool (WalkingHex) combined with a Semi Rigid robot Arm (SeRArm).

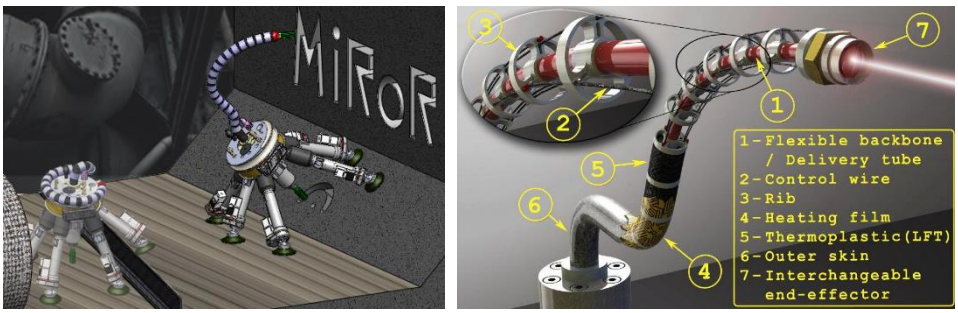


Figure 1-1: Concept art from the MiRoR project proposal

The research in this report is focussed on the control of the SeRArm and how it can be navigated within a restrictive environment. As the concept art in Figure 1-1 suggests, the robot arm considered is a repeating geometry. This type of arm is known as a hyper redundant manipulator and is explained in more detail in Chapter 2. The initial size and weight of SeRArm is dictated by the capacity of the WalkingHex. The mounting area has a diameter of 300mm with a maximum load capacity of 10kg.

1.2 Problem Definition

The system is implemented as a generic tool capable or a broad range of in-situ repair tasks. In this regard, the arm itself is designed to work with any end-effector that can be attached to the tip; provided that it is under 200g and cables can be supplied using an 8mm working channel throughout the length of the robot. There are two main scenarios to be demonstrated using SeRArm, both to be performed on 1:1 scale models of the environment to start with.

1.1.1 Repair of Gas Turbine Engine

Rolls-Royce Aerospace are interested in the SeRArm system for the repair of gas turbine engines [1], without the removing the engine from the plane. Unscheduled disassemble costs approximately $500,000, with the daily down-time of the aircraft estimated at $100,000 per day. The problem that leads to this event is foreign object damage to the blades within the engine. Small stones or ice, cause chips or cracks to the leading and trailing edges. On their own these defects are safe to fly, but they will act as stress features and propagate further into the body of the part over time, therefore causing the part to require replacement before the scheduled maintenance. Current techniques use a blending borescope to machine the edge into a smoother scallop shape. This will reduce the rate of propagation significantly.

However the tooling used at the moment is rigid and requires highly specialised training. The hardest area to reach the leading edge of IP1 and IP2 of the Intermediate Pressure (IP) compressor. This involves the removal of the front fan blades, so the technician can reach outstretched into the engine. Often this will take one or two complete shifts to complete, in a very arduous position.

SeRArm will need to be able to enter a gas turbine engine through the front fan and weave its way to the Intermediate Pressure (IP) compressor. Once at the target the robot will be used to inspect the defect on the blade. Once measured the feature will be blended out using a high-speed spindle mounted on the tip of the robot.

To be able to reach the target, SeRArm needs to be highly flexible with more flexibility towards the tip, because the geometry tightens significantly when approaching the IP. Through some sections there is only a 60mm gap which means the algorithm needs to deviate less than 25mm from an ideal path. Although it is envisioned that the robot will be operated by an intelligent controller capable of path planning, the system should also be usable by a human operator in real-time.

1.1.2 Inspection within Nuclear Power Plant

Co-operative interaction between SeRArm and the WalkingHex is to be demonstrated by performing an inspection on a Reactor Pressure Vessel (RPV) head. This part is a removable section at the top of the RPV, with several holes which the fuel rods are passed through. These holes need to be inspected regularly for oxidisation and the weld seams checked for defects.

Currently the head is removed and placed on an inspection platform, allowing workers to move around the part and make visual and non-destructive tests (NDT). However this room is irradiated. This means the time allowed for technicians to operate in the environment is limited and several precautions are required before they are allowed to enter.

The proposed substitute is to walk the combined Mini-RoboMach from the access point into the room and underneath the RPV head. Once in place, SeRArm can be used to perform visual inspections of the weld seams and look for evidence of oxidisation. The WalkingHex can also carry tooling for machining and NDT.

This does not limit the performance of SeRArm system by the magnitude of the gas turbine, apart from the limitation in placed by mounting on the WalkingHex. It does, however, require that the system can be packed during navigation of the hexapod, and capable of unpacking on its own once in-situ. Rather than walking with the arm outstretched, it is much easier if the body was coiled around the robot and therefore keeping all parts within the 300mm diameter. Uncoiling can then be performed from this state to provide the length of the arm as required to reach the target.

1.3 Aims and Objectives

The aim of the project is to create the robot described by SeRArm. This system has to be capable of navigating within a restrictive environment, such as the gas turbine engine. To achieve this, the arm must be able to manoeuvre through a complex trajectory without deviating too far from the path. The body then needs to be able to be packed within the envelope of the specification, and uncoiled to perform the required actions.

The work to be done can be split into a clear set of objectives:

- Based on the conceptual design of a hyper-redundant manipulator using a serpentine structure and cable actuation, develop the kinematics and methodologies that are the basis for the control strategies and user interface.

- Develop novel algorithm for Tip Following the robot through a restrictive environment that can be run on real-time and validate on a 24 degree of freedom continuum arm with a central spine.
- Produce novel method for feeding the continuum robot based on coiling and uncoiling around a helical grooved drum, including strategies to correct positioning. Demonstration of this method on 24 degree of freedom robot with segregated joints that avoid twisting under gravity.
- Combine the two algorithms, Tip Following and Feed-In, into a unitary navigation of a continuum arm with 24 degrees of freedom to simulate complex repair environments such as the gas turbine engine.
- Demonstrate the unitary system (high and low level control) on 1:1 scale model environments for both inspection and machining task in-situ. To be presented for the end-user application and interaction.

1.4 Terminology and Nomenclature

Within the field of hyper-redundant manipulators and snake arms there are many field-specific terms, with the aid of Figure 1-2 this section aims to clarify the main terminology to avoid confusion for the reader.

- **Guide Disc –** A rigid disc with several holes for the cables to run through, maintaining the arrangement and spacing as the arm is bent.

- **Segment** – The flexible gap or joint between two discs. On their own, segments can be considered passive and share the active movement of their section group.

- **Section** – An actuated length of the arm. Each section can bend with two degrees of freedom and is composed of several segments.

- **Stage** – Some continuum manipulators adopt a staggered design to help balance the forces within the arm. In this scenario, the collection of similar sized sections forms a stage. This is because each stage must be treated differently within the forward kinematics.

- **Tip** – The distal end of the arm, with the actuation pack considered as the base. Numbering convention for the sections starts at 1 from the base.

- **Actuation Pack** – To reduce the weight of the arm, it is common to keep the control mechanism extrinsic in an actuation pack. The size and shape of the pack depends on the design of the arm attached and the method of actuation used.

- **End-Effector** – The tool or combination of tools equipped at the tip of the arm. Possible tools include cameras for inspection, grippers for foreign object removal, and high-speed spindle or laser for in-situ repairs or machining.

- **Working Channel** – Opening within the core of the Snake Arm design to allow electrical or other connections from the base with the end-effector

- **Feed-In System** – An actuated base to advance the origin position, either linearly or rotationally.

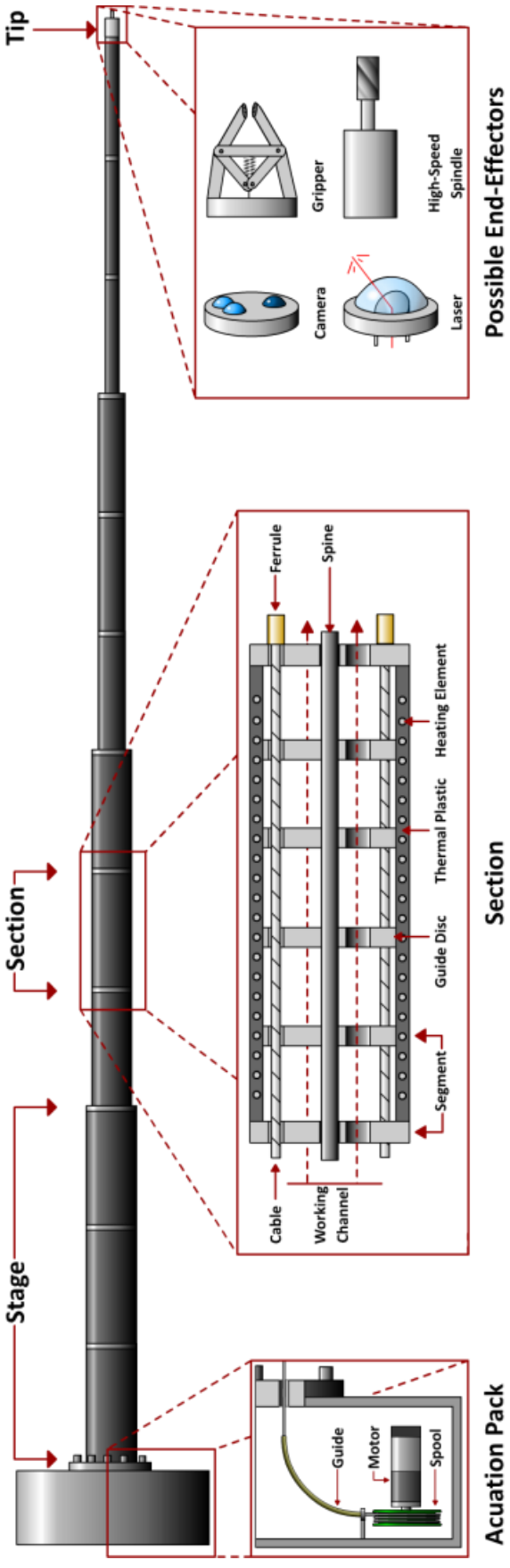


Figure 1-2: Common terminology used to describe parts of the system

The important symbols used are defined below in Table 1-1

Table 1-1: Nomenclature

Symbol	Name	Definition
θ	Bending Angle	The angle between the tip of the section and the z-axis
ϕ	Direction Angle	The angle of the bending plane, taken counter-clockwise from the x-axis.
S	Section Length	The length of the actuated section
R	Rotation Matrix	The matrix to convert from the origin frame to the co-ordinate frame at the tip
B	Base Position	The start of the snake, in the tip following algorithm this value is advanced by a small amount
P	Tip Position	The end of the snake, in the tip following algorithm this value is advanced by a small amount
L_i	Length	A length values, usually related to the actuation. i is sometimes 1-3 or x, y depending on appropriate use
n	Total Number	A variable value for the total number or final number in the set.
r	Radius	The distance from the central axis
α	Rotation	Number of rotations, used for the cable spools or the helical coiling
ρ	Phase	The angle of offset between sections
λ	Pitch	The difference in position per rotation of a spiral or helix.

2 Literature Survey

2.1 What are Hyper-Redundant Manipulators?

Any manipulator fits within one of three categories, as illustrated in Figure 2-1:

Underactuated manipulators are trivially described as a system with less actuators than degrees of freedom (DoF), therefore leaving some joints passive. However, well designed systems in this category can still be fully controlled such as the inverted pendulum [2] or an Acrobot [3].

Critically Actuated systems are the easiest to control, with only one motor per degree of freedom the kinematics are straightforward and, more importantly, so are the inverse kinematics. CNC machines are commonly implemented like this.

Redundantly Actuated manipulators have more active joints than degrees of freedom; either to increase the workspace of the resulting system or the combination of joints allows reaching the same point with different trajectories. The extra motions available make this type of manipulators the most popular and most common choice for industrial robotics, such as Industrial Arms [4].

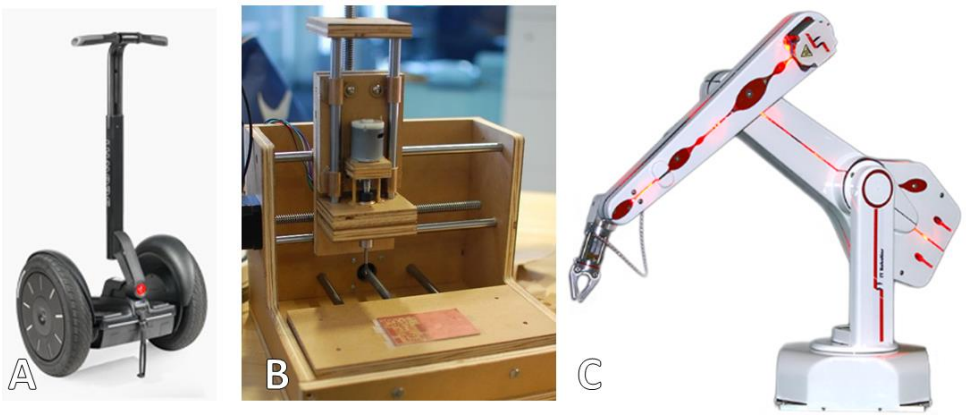


Figure 2-1: Examples of Different Kinds of Manipulator A) Underactuated: Segway i2 Personal Transporter [5] B) Critically Actuated: A Simple 3-Axis CNC Machine [6] C) Redundantly Actuated: 5-Axis Robotic Arm by ST Robotics [7]

Hyper-Redundant manipulators [8] are the extreme of the last category. The typical design approach features a repeating geometry of actuators working on the planes perpendicular to the axial through the centre of the manipulator. Using an amalgamation of these actuations allows the tip of the device to be defined and placed in Cartesian space using all 6 degrees of freedom (x, y, z, φ, θ, ψ). Naturally the complexity of these systems makes the control and inverse kinematics complicated, with the difficulty increasing exponentially with the number of actuations involved.

2.1.1 Types of Usage: Elephant Trunks versus Snake Arms

Given the flexibility of these manipulators, the method of moving from one pose to another is not a trivial task. If performed purely in the configuration space of the robot the resulting behaviour is likely to appear very chaotic and is not suitable in a restrictive environment. Instead there are two distinct behaviours used which are named after their biological morphology, these are encapsulated in Figure 2-2.

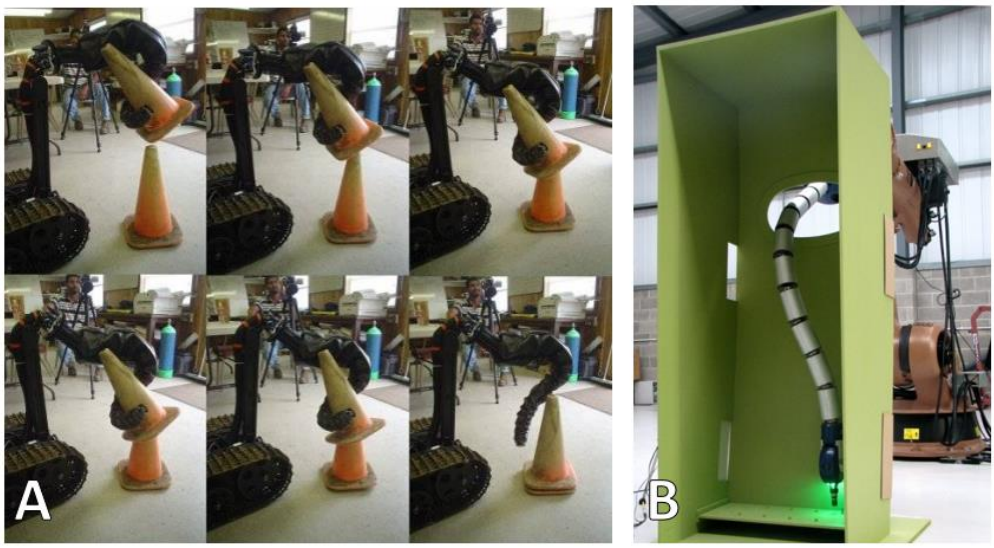


Figure 2-2: Examples of A) an Octopus Arm Grasping and Stacking Cones [9] and B) a Snake Arm Exploring and Inspecting a Makeshift Environment [10]

Elephant Trunks [11] or Tentacles [12] operate in sweeping motions the same way their namesake would act. Commonly these systems have a fixed base and swing and curl their body to avoid obstacles. The main purpose for this type of system is that they can use their bodies to grip and move objects [9].

Alternately, a Snake Arm uses linear advancement is along its axial length in the way a snake would explore the environment [13, 14]. These systems are used to investigate restrictive environments and perform tasks within areas humans cannot reach. Typically they require some additional mechanism to move the base of the manipulator to advance the system. With the scenario described in Chapter 1, it is clear that a Snake Arm approach is required as the environment is too restrictive for the project.

Snake Arms are not to be confused with snake robots, which are free agents and are able to replicate the translocation techniques used by their biological counterparts [15] [16]. However there is a lot of crossover between the fields since similar design choices are made in both the structure and actuation, as can be inferred from Figure 2-3. Both architectures use repeating geometry of joints. Leading to the actuation and behaviour to be very similar.

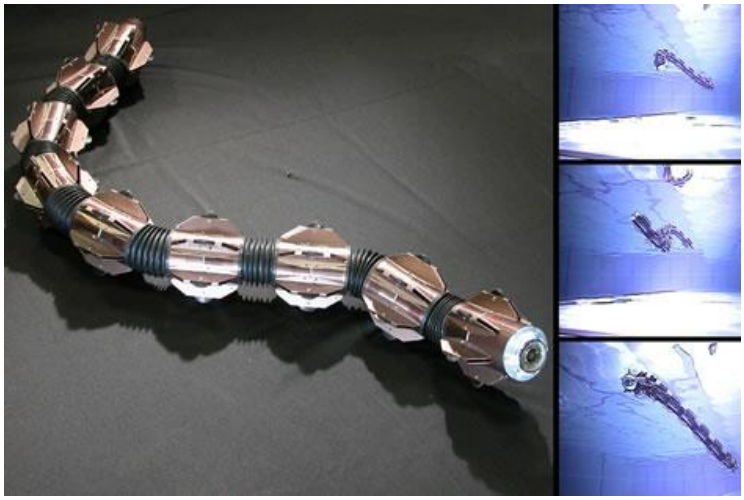


Figure 2-3: The ACM-R5 is an Impressive Example of a Snake Robot. Created by HiBot, this robot can Self-Propel on Both Land and Water [17]

2.1.2 Design Architecture: Discrete, Serpentine or Continuum

The backbone design can be encompassed by three different configurations, as illustrated in Figure 2-4. Firstly by discrete joints separated by solid links, allowing a large amount of reach for the system with the minimal number of joints; common in the early developments of hyper redundant manipulators when computation resources were limited [18, 19], but still used today in large modern industrial systems [10] that need superior stiffness and strength.

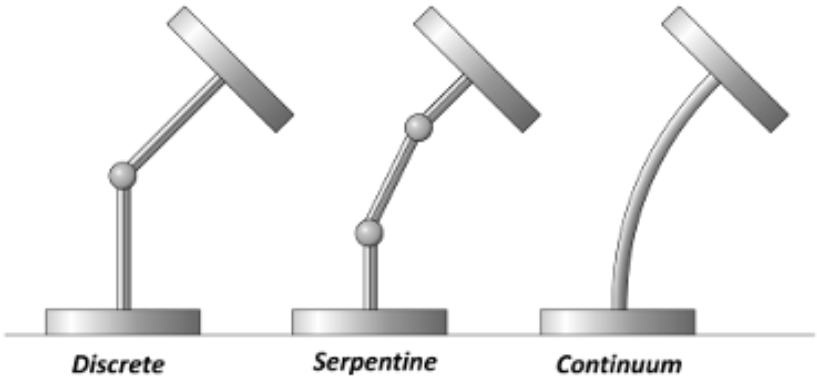


Figure 2-4: Examples of different backbone designs

Alternately, the design can have a continuous backbone that flexes to produce a smooth profile by using a continuum structure rather than a discrete joint [20, 21]. This gives much greater flexibility and is often implemented in medical applications where size and therefore weight is greatly reduced. However this architecture suffers from low stiffness and is well documented to produce a skewed curve rather than the uniform shape intended [22, 23]

By combining these two approaches it is possible to balance the stiffness of a discrete system while the flexibility of a continuum. This is achieved by having more discrete joints in one section, typically referred to as serpentine as these are analogous with the backbone of a serpent. Often this architecture is used in industrial systems that either require substantial bending or operate in more restrictive environments [11, 24].

For this project, the best architecture to use is the Serpentine. With this design a highly flexible Snake Arm can be produced that still has stiffness within the required scale of the project goal. Initial work tried to use a continuum structure, but it was found that the designs suffered severely from twisting about the axis, making them unsuitable.

2.1.3 Methods and Mechanisms for Actuation

In general hardware design, any joint is typically driven directly with immediate feedback sensors of the resulting action, in order to create the smallest closed-loop control possible. This provides a highly accurate system and allows the feedback to be used quickly within complex control algorithms. The joints can also be implemented with the simplest mechanics, rather than convoluted mechanisms, making forward kinematics easier to calculate and process.

Conversely, the architecture of a continuum system is similar to a cantilever. As such it is beneficial to keep the extended mass as light as possible to reduce the effects of gravity on the structure. With this in mind, most designs keep the actuation extrinsic from the outstretched construction. This configuration reduces the possible methods of actuation to techniques analogous with puppetry. However there are still multiple methods for implementing the active control.

The use of wires to deflect a joint at a distance is a well-established practise, and it is not surprising to find popular use within this field, Figure 2-5 [24-28]. The advantages with control cables are their strength, high precision of their extension, and intuitive to use. Also [29] has employed a method to stiffen sections of a tentacle by shortening their lengths when in place. However they suffer some drawbacks, most notable the path they take between guides will be the most direct which means any design will need several guide discs, as shown in Figure 2-5b. Given the strength and durability required of these guides, they are the significant factor for the mass of the

manipulator. Another problem is that cables can stretch, which leads to re-tensioning being required; Or over tension can cause them to snap, leaving the section underactuated.

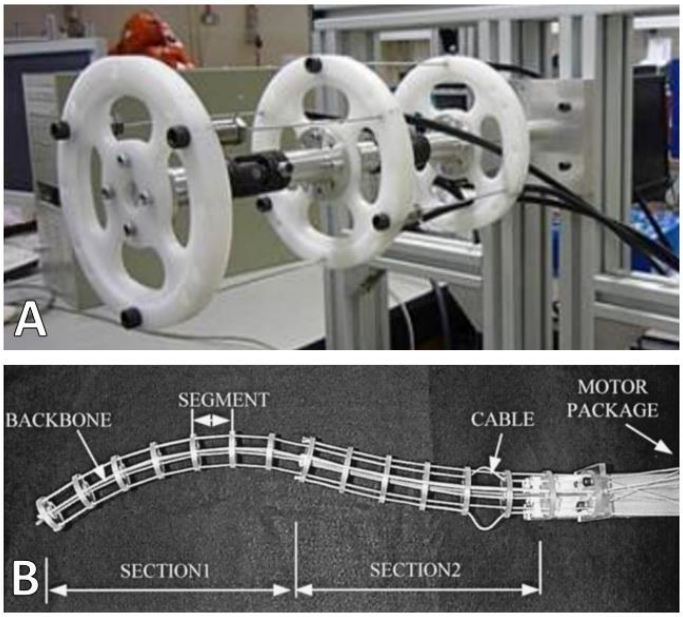


Figure 2-5: Examples of Cable Driven Arms A) Two-Section Discrete Backbone [26] B) Two-Section Continuum Backbone [25]

Another common way of controlling an action extrinsically is with the use of pneumatics or hydraulics, similar to those shown in Figure 2-6. Not only do they have incredible strength at high pressure, but air has almost negligible mass when compared to a steel wire. Another advantage is that they can be implemented as a soft-robot. A soft-robot is a system that can deform such that collisions can be used for gripping objects without danger of damage. This is extremely useful for the purpose of an octopus tentacle [30, 31]. Furthermore, FESTO [32] has demonstrated that the entire body of a manipulator using this method can be manufactured with a 3D printer as a single part. However this project would likely need 72+ micro valves (at least 6 valves per section; with 12 sections) to operate a manipulator with this type of actuation, making the system inside the arm overly intricate. There is also less precision in the use of pneumatics, without significant time investment into the mechanical behaviour.

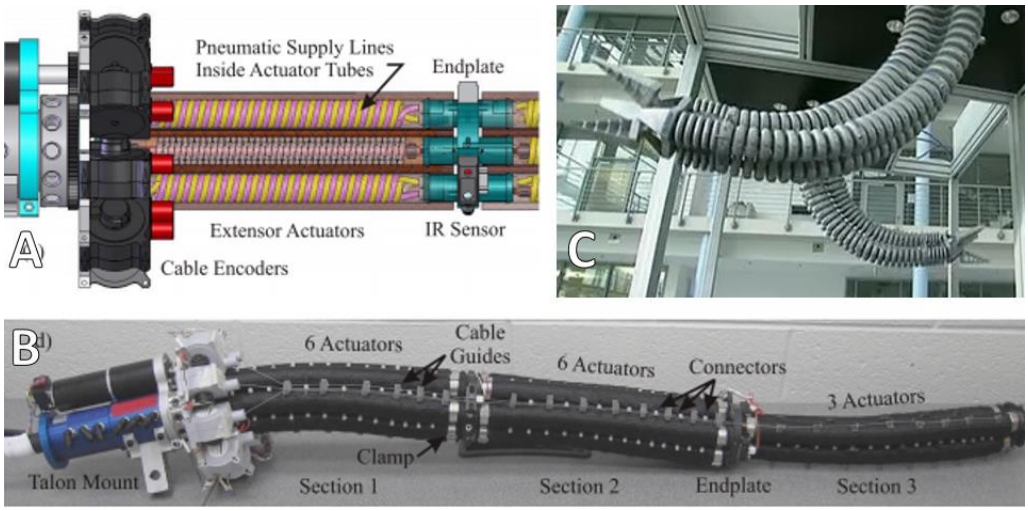


Figure 2-6: Examples of Pneumatically Driven Manipulators A) Cut-Away of OctArm VI Design Concept B) Annotated Picture of OctArm VI [33] C) Festo's Bionic Handling Assistant [32]

Other methods exist and have been exhibited in working systems, as pictured in Figure 2-7. A common method in medical manipulators is the use of Shape Memory Alloys (SMAs) [34]. These materials change between two pre-set shapes by changing the temperature of the medium, wires of SMA can be made to shorten as the temperature increases. Significant research is being performed on the accurate, precision control of such devices [35], but for the larger scale use they are too inaccurate for use in this project. Another example from the medical field uses Active Cannulas [36]. These use a nested series of curved tubes, which are rotated and extended in turn to reach the desired position and shape. It is unlikely this technique would work for the purposes of the project as would not be compatible with the machining operation.

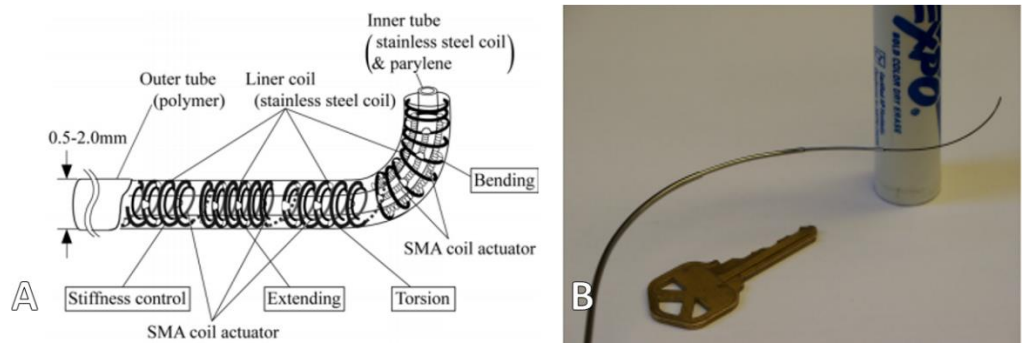


Figure 2-7: Alternate Actuation Methods A) Manipulator Using Shape Memory Alloys [37] B) an Active Cannula [38]

2.2 Noteworthy Examples

As with any field there are a number of well-known examples, within Hyper-Redundant manipulators there is the work by Chirikjian, the Elephant Trunk, the OctArm and the work of OC Robotics.

2.2.1 Gregory S. Chirikjian

Gregory S. Chirikjian completed his undergraduate degrees at Johns Hopkins University in 1988 and received a PhD at the California Institute of Technology. He has published at large amount of the early work in the field and is most notably credited with the creation of the term “Hyper Redundant Manipulators”. His work covers the potential uses [8, 39], the design [40], kinematics [41-44] and the navigation [45-49] of such devices.

2.2.2 The Elephant Trunk and OctArm

Ian Walker’s team at Clemson University has been working on hyper-redundant manipulators since 1999. The initial work led to the creation of the Elephant Trunk robot [50, 51]. This was the first multi-segmented backbone at the time, which means there are multiple joints to each active section. Note how the design shown in Figure 2-8 has springs along its length, these are required to provide resistive force such that when the cables are pulled the segments work together to make a smooth curve rather than collapsing into chaos.

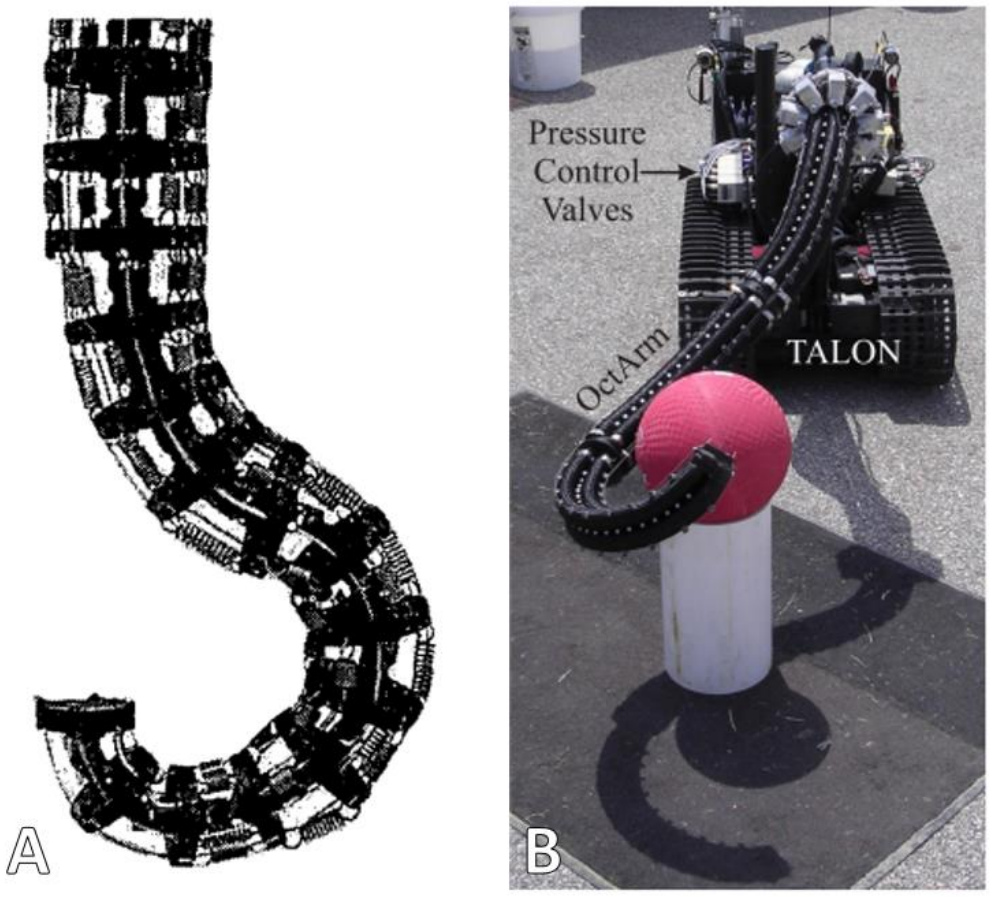


Figure 2-8: Work at Clemson University A) Elephant Trunk Robot in Hooked Configuration [11] B) OctArm VI Grasping a Spherical Ball [9]

Work on the Elephant Trunk led to a pneumatic soft robot known as the OctArm of which there have been several generations. This system has been the backbone for a large amount of the academic work published on the IEEE for the last decade, from design [21, 22, 33, 52-54] and kinematics [55-60] to advance control strategies [30, 61-64] and user interface [65].

The work on the OctArm has greatly influence the field, especially for Continuum Arm kinematics. The algorithms, in Chapter 5 and 6, are built from geometric kinematics published by the group [66]. However the majority of the research focusses on pneumatic actuation and the grasping action, which is not directly useful to this project.

2.2.3 OC Robotics

A discussion on Snake Arms is not complete without mentioning OC Robotics. This Bristol based company was founded in 1997 by Rob Buckingham and Andrew Graham and to date is the only commercially successful company selling Snake Arm solutions. While OC Robotics have produced no academic papers, they have provided several keynote talks [13, 67] as well as publishing several patents [68-70].

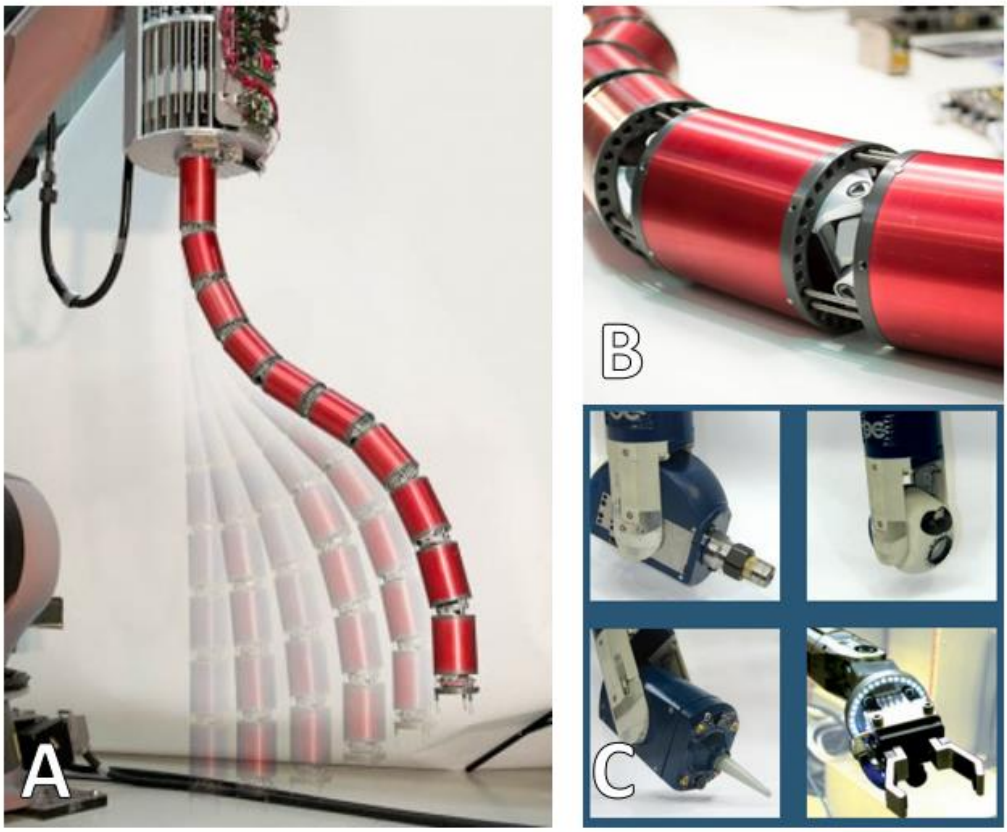


Figure 2-9: Explorer Range from OC Robotics [10] A) Snake Arm Moving with Actuation Pack Exposed B) Close Up of Snake Arm Joints C) Examples of Available Tooling

Their off-the-shelf Explorer Series ranges from 40mm diameter arms to 150mm, with lengths from 1m to 3.25m and payload capacities of up to 32kg depending on the previous parameters. The design uses discrete universal joints, controlled by steel cables, separated by hollow tubes. Cables are actuated by a conical series of linear motors housed in the actuation pack. The arms can support different tooling at the tip for inspection, cutting, welding or grabbing. Alongside this, they also offer an intuitive software package that includes several simulated scenarios for practice.

Alternately, OC Robotics has provided several bespoke solutions for different industries. Examples include LaserSnake, an Explorer arm fitted with a 5kw laser for cutting through thick walled steel pipes. JetSnake, a linear-track mounted Snake Arm to deliver high pressure water to clean tunnel boring machines of Bouygues Travaux Publics. Finally, the SAFIRE system for inspection of the Upper Feeder Cabinet (UFC) of CANDU reactors at Ontario Power Generation's Pickering site.

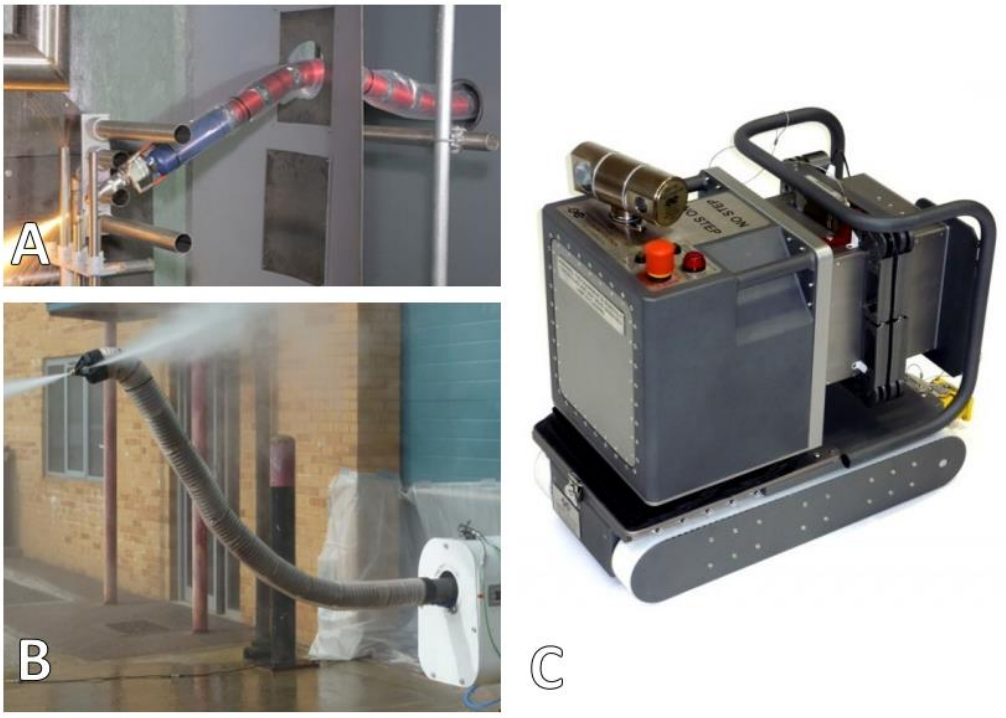


Figure 2-10: OC Robotics Bespoke Solutions [10] A) LaserSnake Cutting through Steel Pipes B) Test Firing of JetSnake C) SAFIRE Remote Vehicle in its Packed State

The conventional products offered by OC Robotics are much too large for use in the Gas Turbine scenario. Furthermore, each joint in the explorer design is limited to 20° of bending, with the entire arm limited to a summation of 180° of bend, meaning it would also not be feasible to navigate the environment. The thinnest bespoke option only has a length of 600mm, which is not long enough to get past the Low Pressure compressor, let alone reach the target area.

2.3 Control of movement

Hardware is not the only challenge faced in the creation of a hyper redundant manipulator. As briefly mentioned in Section 2.1.1 the movements required easily become complex and perplexing as more sections are added to the arm.

2.3.1 Navigation

Navigation is computationally the most difficult task for this type of manipulator. Not only is a Snake Arm extremely complex, but it is also compounded by the typical scenarios requiring accurate movement through tight gaps if used for navigating in geometrically complex environments. The approach often taken for this longitudinal navigation, is to apply already existing algorithms such as those for obstacle avoidance or path planning [71, 72].

Chirikjian et al. [45] demonstrates, Figure 2-11a, a planar obstacle avoidance algorithm with a "tunneling" technique (to constrain the geometry) to navigate through a known environment. The method uses equations of differential geometry to ensure the manipulator is constrained within the tunnels. Tunnels are located where the *interior* segments must avoid an obstacle, less focus is given to the *exterior* segments.

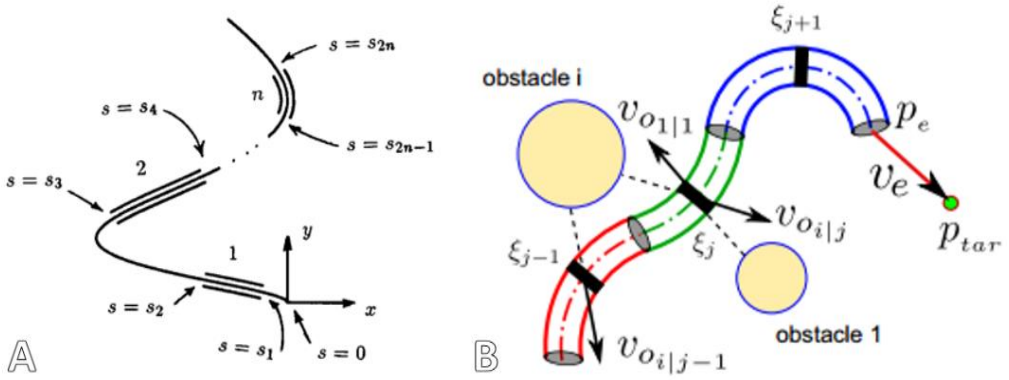


Figure 2-11: Diagrams of tunnelling technique A) Tensor Manipulator Constrained by a System of Tunnels [45] B) Continuum arm with discreet sensors [73]

An extension of this performed by Choset and Henning [74] uses sensor data to create Generalized Voronoi Graphs (GVG) to achieve a follow-the-leader approach. The GVGs create a map of the environment which provides an obstacle free path to the target. The head of the robot is navigated along the paths and the body fitted to the shape using techniques employed by Chirikjian.

Another optimization approach is presented by Lyons, Webster and Alterovitz [38] to navigate active cannulas to avoid pre-determined obstacles. A potential field is used to find a path that is furthest from all the obstacles, that is every point in the workspace can be given a value dependent on its closest obstacle.

An obstacle avoidance algorithm for rigid link robots was shown in [19] that is capable of avoiding dynamic obstacles. This is achieved by constantly reassessing the distance to the features, or more explicitly, not assuming the environment is static.

This was updated by Godage et al. [73] to show how it can be adapted to continuum manipulators, as in Figure 2-11b. Using of discrete sensors along the length of the arm to measure the proximity of the environment to create an actively measured potential field. The report goes on to explain how this method can compensate for lack of direction feedback or sensor omnipotence.

Most obstacle avoidance algorithms or path planners mentioned thus far depend on pre-known environment. They would struggle in tasks with free-moving elements that, although are known by their CAD information, their position is unknown within the navigation space, such as the rotatives within a jet engine that can move freely on a shaft; or for disaster sites where no environmental data exists. While the obstacles in these types of environment can be mapped by sensors, as suggested in [73]; however, sensors add extra weight which many designs could not carry due to limited payload

capacity. Also it is worth noting that some of these examples only consider manipulators with fewer sections, which means the inverse kinematics are simpler.

An alternate technique for navigating the Snake Arm is Tip Following; this process works on-the-fly by treating the tip of the arm as a free-moving point. As this point is moved in the desired direction, an algorithm is used to determine a new arrangement of the snake arm that advances the tip to the new position and minimizes the deviation from the path already travelled. This is similar to the method in [74], except it uses its path rather than a GVG to fit the body.

Recently, a new tip following method suitable for snaking has been reported [70]; this takes sections in pairs along the length and finds an arrangement that starts tangential to the path and ends tangential to the path. By following this process through the whole arm it is possible to find a completely new pose for small changes. However this method limits the freedom of the snake, as every other section end must be on the path, which may not be possible in complex poses with high bending angles.

2.3.2 Feed-In

Most previous manipulators have either been presented with a mobile platform for insertion. Techniques demonstrated, Figure 2-12, include attaching the arm to a bomb disposal chassis [9], using a linear track [13] or an industrial robot arm [10]. The implementation of these feed-in techniques relies on third-party hardware that is capable of maintaining the longitudinal motion, rather than any adaption being performed by the Snake Arm; in fact it is possible to claim the Snake Arm control is kept ignorant of how this process is achieved. Furthermore, these methods require significant space outside the scenario environment which is not always possible.

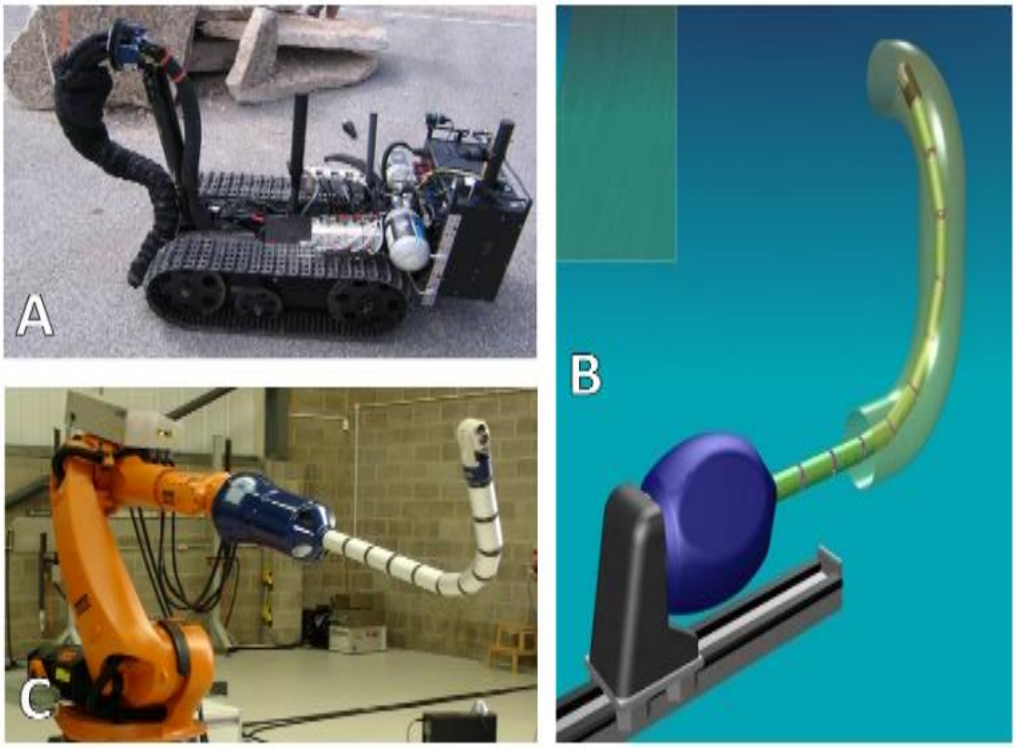


Figure 2-12: Examples of Feed-In Techniques using Independent Hardware A) Tracked Robotic Chassis [9] B) Linear Track [10] C) Industrial Robotic Arm [10]

Alternately, Snake Arms have been created with static base and use the arm to provide the linear motion [45, 75]. This is achieved by taking the extra length of the arm and creating a large loop or a coil stack, as the tip is advanced the loop is shrunk or the coil is unfolded to advance the body of the Snake; as shown in Figure 2-13. Once again this requires sufficient space outside of the environment entrance and can be computationally taxing.

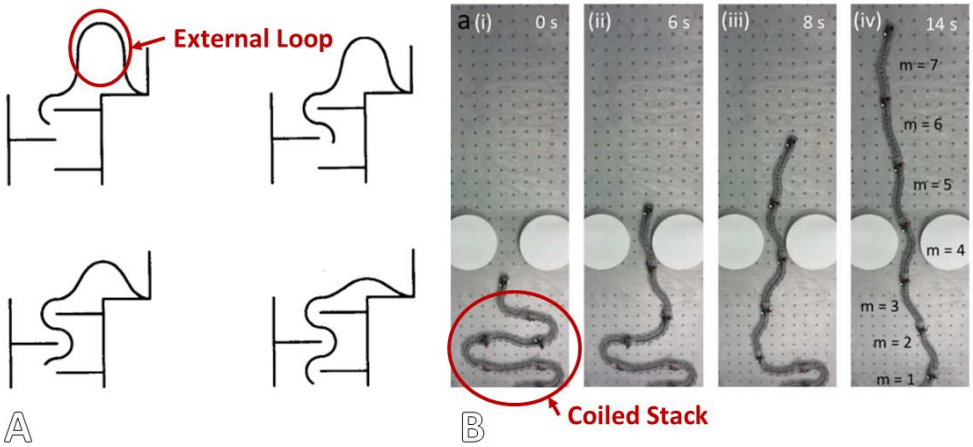


Figure 2-13: Examples of Feed-In with a Static Base Position using A) a Loop [45] or B) a Coiled Stack [75]

Alternately, the intended arm will be coiled around a rotating actuation pack. In this arrangement the manipulator takes the least amount of storage space and allows the final prototype to be initialised in any direction. A smaller, shorter, and internally coiled arm design is documented by OC Robotics [76], Figure 2-14a. However the coiling stage in this example appears to work while the sections are passive, i.e. the cables have been slackened. Another example from OC Robotics, Figure 2-14b, the SAFIRE Snake Arm is mounted vertically on a robotic chassis and can uncoil in any direction allowing the user to inspect the space above or below the arm. However this arm is planar, and is only designed to operate within very strict discrete curves.

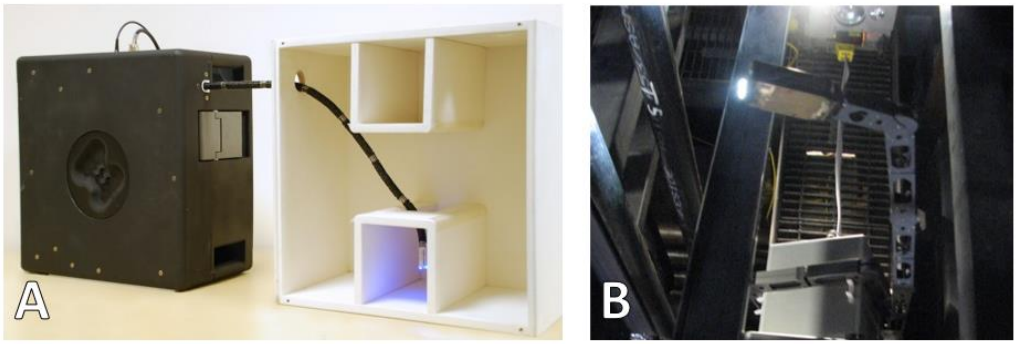


Figure 2-14: Examples of Coiled Snake Arms by OC Robotics [10] A) an Internal Coiled Snake B) A Planar Snake for Inspection

Other than the commercial implementations above, there has not been any academic reports on actively controlled coiling mechanisms for serpentine Snake Arms. However there have been publications in other fields that have some transferable knowledge, such as those in Figure 2-15. For example, Carnegie Mellon researchers have developed a robotic snake capable of climbing cylindrical objects, like trees, by applying a helical movement [16]. There is also the tendril robot developed by NASA [77]. This robotic endoscope operates similar to a snake arm, however the body is not fully actuated. Instead the tip is actively controlled to steer while the rest is passive. The system features an actuation pack for retracting and extending the tendril similar in design to the OC Robotics' suitcase snake.

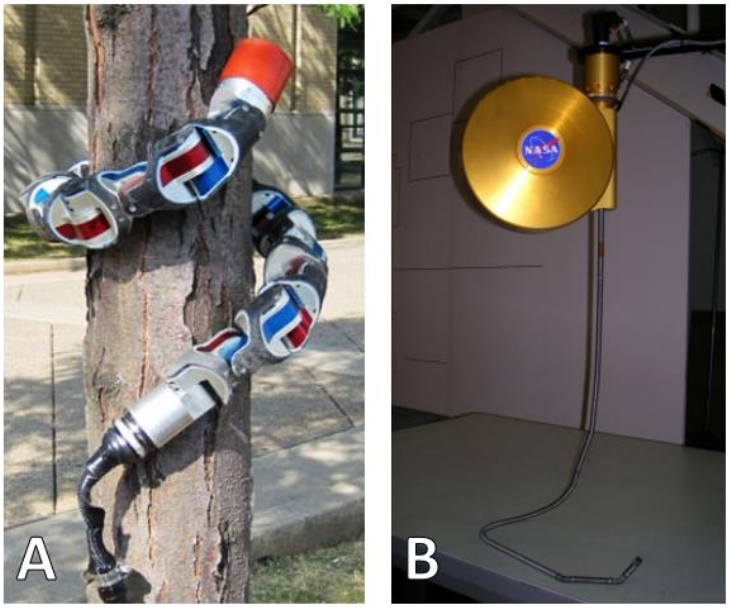


Figure 2-15: Other Work of Benefit to Feed-In Research A) Tree-Climbing Snake [16] B) Tendril Robot [77]

2.4 User Interface

The more developed systems have considered how the control of their system can be best operated by the user. End-users are typically less interested in the mathematics and the academic challenges. As such the human interaction needs to be simplified and intuitive. This in turn leads to faster and more efficient implementations.

Grasping with octopus tentacles [65] is a non-trivial task for to simplify the behaviour to a single joystick. Csencsits achieves this by creating multiple mappings which the user can switch between. Each mode achieving a different task, through either position or velocity.

As mentioned previously, Tip Following is a common method for reducing the degrees of freedom the user needs to control in Snake Arm architectures. OC Robotics patent [70] presents a mapping that can be used with a gamepad. It also shows another method for human interaction using a magnet ring. By creating sensing and a control algorithm that attempts to keep the robot central in the ring, the shape can be adjusted manually by moving the ring.

How the user interface can be best implemented is discussed further in Chapter 4.

2.5 Chapter Summary

This chapter has explained the field of hyper-redundant manipulators, including the common designs, actuation and usage styles. From the system description provided in Chapter 1, it is clear that a Snake Arm morphology is required. Given the scale required to navigate restrictive environments, such as the gas turbine engine scenario, a serpentine construction is to be used to provide enough stiffness with sufficient flexibility. The hardware design is discussed further in Chapter 4. It was apparent that pneumatics would not be suitable and alternate techniques too weak for long arms with this structure in mind, therefore it was decided to pursue cable actuation.

To achieve the navigation essential to the use of the Snake Arm morphology, an algorithm is need. Comparing the techniques used by others, it was found that many are specific to the behaviour of the robot used, and several are dependent on the environment being known prior to operation. While it was suggested sensors could provide this data on-the-fly, it would not be feasible to add the extra weight or space on the robot design.

For the coiling method required a small selection of prior art was found. These presented with a passive drum and a guide to calibrate the sections before release, or a planar discrete arm with a pre-set movement on a square drum, or a largely passive body.

2.6 Academic Challenges

From this investigation it was clear that a set of challenges need to be overcome to complete the objectives.

1. **Academic Challenge – Tip Following:**

To create a Tip Following algorithm capable of navigating the proposed structure of the Snake Arm as well as be easily re-addressed to alternate designs. This needs to not only maintain the path within the tolerance of the specification, but also computational efficient for the length and number of sections involved. Since the project is being setup for generic use, the algorithm should operate from the required path alone and not be reliant on the environmental data being provided

2. **Academic Challenge – Feed-In:**

   Develop an active method to coil the body of the robot around a drum. This can be aided by an extra degree of freedom to rotate the drum. From this the system needs to be able to uncoil in a manner that can be used by the Tip Following to advance the Snake Arm into a scenario, without detriment to the performance of the preceding algorithm.

3. **Academic Challenge – Combined System:**

   Combine both algorithms into a single process, such that the full Snake Arm configuration can be simulated. This is essential for using the outputs with the physical robot. It is also important for accuracy that the Tip Following uses the advancement created from the feed-in, rather than an assumed position. This will lead to the functional final prototype, which can run independently or mounted on the hexapod.

3 Research Approach

To create a functional prototype capable of achieving all the desired goals, the system has been broken down into smaller elements. Each with their own objectives, expected outcomes and demonstrable accomplishments. Figure 3-1 illustrates the different parts and their dependency throughout the project. In the following chapters the techniques conceived and implemented will be discussed.

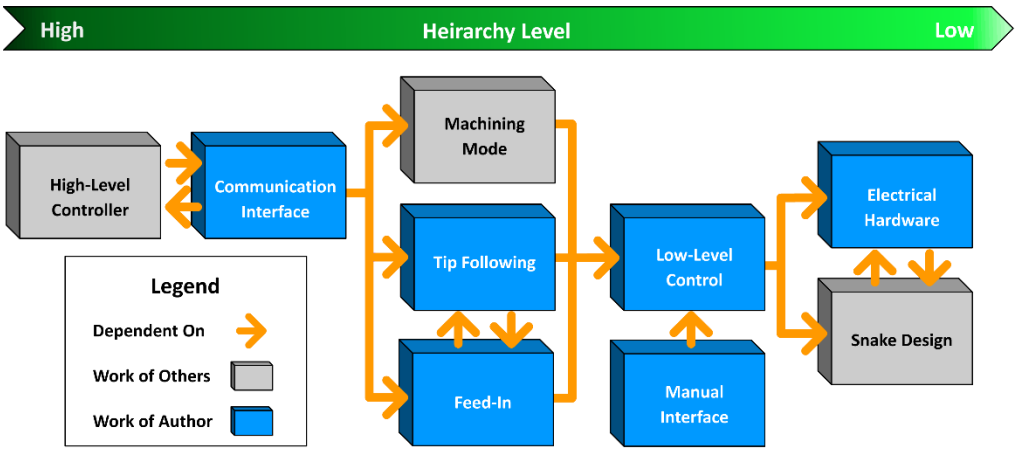


Figure 3-1: Infographic to explain the dependencies of each element of the final system and their hierarchal relationship within

3.1 Low-Level Control

At the lowest level of the hardware is the low-level control (LLC). These simple algorithms need to take the desired pose of the continuum arm and convert it into a resulting action equivalent to the demand. This is heavily dependent on both the electrical hardware and the design architecture of the continuum robot. These are both inter-dependent and determined by the design specification.

Chapter 4 covers how the actuating cables of the continuum robot can be used to deform the structure, the electronics and how they can be used to control the cables. For successful operation of the continuum robot the LLC needs to be reliable, accurate, repeatable and deterministic. A manual control interface is developed to command a simple two section demonstrator using an Xbox 360.

3.2 Tip Following

Tip following is the process used to organise the various poses of the Snake Arm requires during the navigation. In Chapter 5 alternate methods for achieving this are investigated. The implementation of this optimisation process is reliant on the objective function to which different techniques are covered to find best performance.

Testing this algorithm is a case of demonstrating the computational time, path deviation, and tip error. Through a mixed series of simulations it is shown that all these factors can be kept below the thresholds set by the design specification. Examples include a randomly specified selection of paths and a scenario within a generic environment.

3.3 Feed-In

The Tip Following development depends on the method used for the Feed-In mechanism, while the Feed-In mechanism needs to know the requirements of the Tip Following. The design has decided upon a rotating drum for the mechanism which has led to some unique problems that require a novel solution involving an active uncoiling motion.

Chapter 6 encompasses the steps taken to achieve the coiling movements and the correction needed for the drum. These include creating the coil profile, analysing the different uncoiled positions, assessing the errors, producing corrective actions and developing this into a single function that can be combined with the Tip Following. The final simulation shows the removal of the errors and the uncoiling function.

3.4 Combined System

Chapter 7, brings the different elements together into one cohesive system, by reviewing how the Tip Following and Feed-In algorithms are joined into a single implementation, how the results from the optimisation can be run on the hardware, how the machining mode can be included, how the user can interact with the system, and how an external high level controller can operate the Snake Arm.

With the completed system it is possible to demonstrate the capabilities of the prototype on a selection of physical environments. These tests include inspection and repair of a full scale model of a gas turbine engine, and the examination of an RPV head substitute while mounted on the walking hexapod.

3.5 Prototyping

Throughout the course of the project, different demonstrators have been created. Each demonstrator was designed for a specific purpose and are used for various examples within this thesis. A brief description of each system is provided below while the control is covered in detail in Chapter 4.

3.5.1 Prototype 1: 3-Cable, Two-Section Snake Arm

This quick and simple prototype was designed to give a perfunctory demo to show the basic operation and can be used as an educational tool during the project. The Snake Arm, Figure 3-2, in this system is a straightforward, central, continuous, Super-Elastic NiTi backbone with a few guide discs along its length. Meanwhile the actuation pack uses three cables per section, which are controlled linearly by six stepper motors.

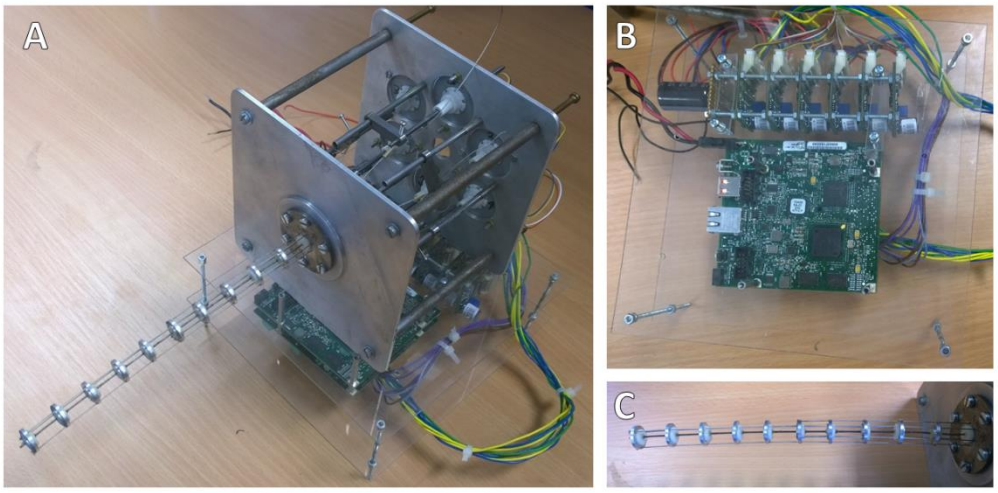


Figure 3-2: Prototype 1 A) the complete system B) electronics of demonstrator, a NI 9606 sbRIRIO and 6 stepper motor drivers C) a simple Two-Section Snake Arm consisting of a central backbone and deformed by steel cables

This demonstrator served several purposes, including experience with sbRIO hardware. The 3-Cable method was investigated, basic kinematics, manual control using an Xbox 360 Controller, and initial interface design. During its usage it helped identify the risks towards yielding the backbone which was solved for further designs by increasing the number of guide discs and the quirks of producing an efficient program within LabView.

3.5.2 Prototype 2: 4-Cable, Twelve-Section Snake Arm

The second demonstrator uses four cables to control each section rather than three, furthermore the Snake Arm design was extended to be the full length of the specification with 12 sections. To achieve this, the prototype uses 24 DC motors each controlled by an electronic speed controller (ESC) intended for radio control vehicles.

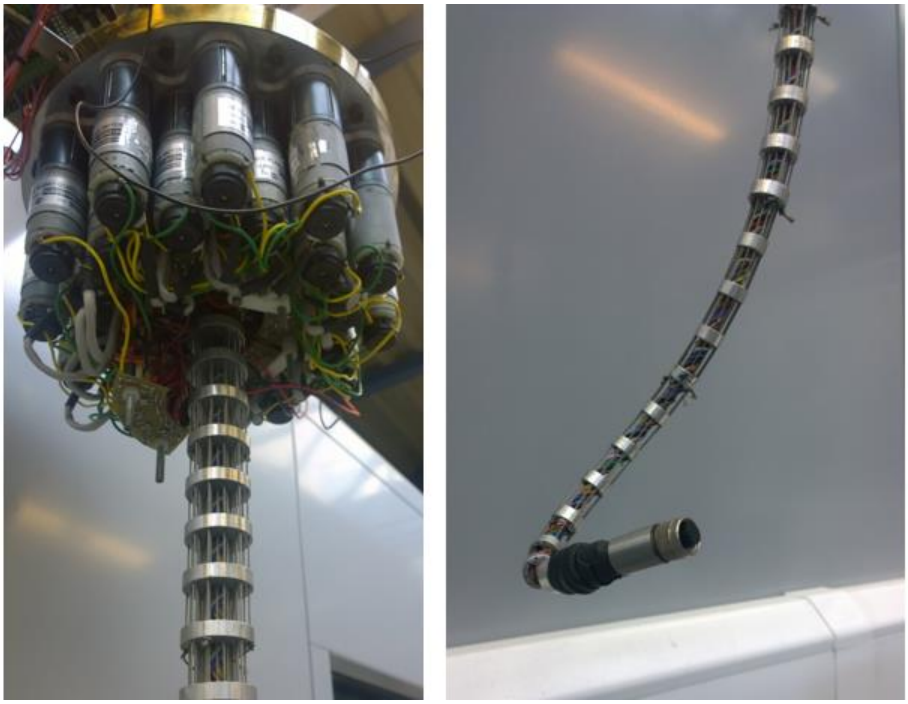


Figure 3-3: Second Prototype, electronic in actuation pack (left) last four sections (right)

The primary purpose for this system was to gain experience in the problems associated with building and operating at the complete complexity. At the same time this prototype allows experimentation with the 4-cable approach and the Tip Following algorithm. During its development a significant problem was identified, Snake Arms with a central backbone cannot resist an axial twisting motion, which leads to the mechanics not behaving as intended when combined with the effects of gravity.

3.5.3 Prototype 3: Three-Section Snake Arm

To test the solution to the design problems of Prototype 2, a new short, three section snake arm was created, using the all the hardware elements that were intended for the final prototype to confirm that they are adequate for the purpose. This includes the motors, speed controllers, spool design and the cable management.

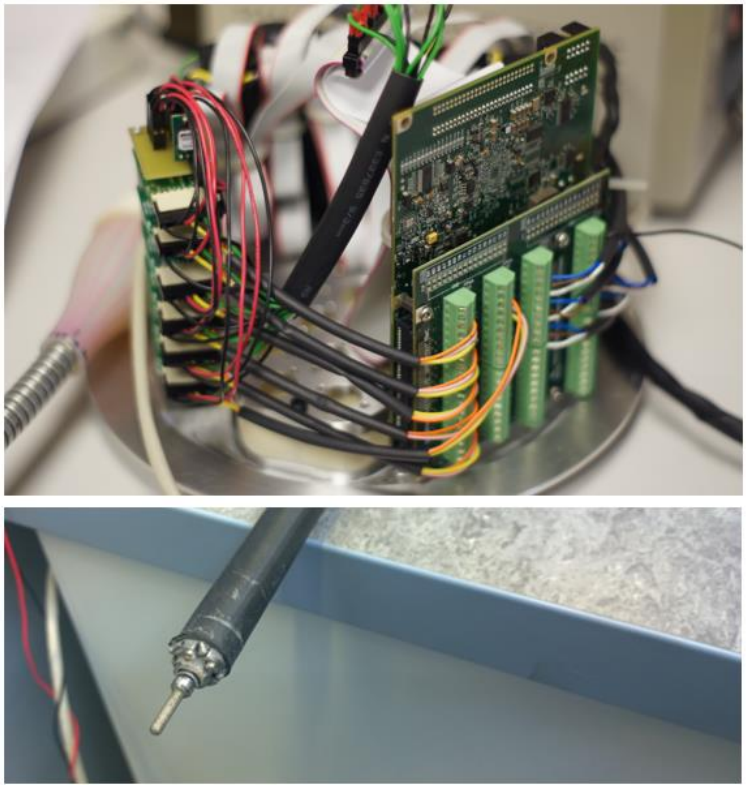


Figure 3-4: Third Prototype, actuation pack electronics (top) and machining spindle (bottom)

Given the enhancements in the hardware selection, it allows the LLC to be refined and improved. Through this a fast and repeatable system was presented that is used for accuracy tests, communication trials and demonstrating the machining mode. This system had only minor issues relating to the backlash within the cable spools.

3.5.4 Prototype 4: Twelve-Section Snake Arm with Rotating Drum

The final prototype, designed to fulfil the project brief with a tapered, twelve-section Snake Arm connected to a rotating drum. The rotating mechanism works by a single motor that engages with a ring gear that is equivalent to the size of the drum, in turn this rotates the whole actuation pack relative to the base plate.



Figure 3-5: Final Prototype

This system can Tip Follow, Feed-in using the rotation, and perform the combined algorithm of these actions. With the core functionality, the prototype can demonstrate inspections within restrictive environments and while mounted on the walking hexapod. The development suffered further problems with the backlash in the cables, but these were compensated for in the code.

3.6 Chapter Summary

In this chapter the sub-elements of the complete system and the prototypes developed along the course have been introduced, so that the research challenges to be addressed in this project can be understood. Through this, their dependencies and hierarchy have been discussed. With the description of the main tasks, their expectations and results are covered, along with where they can be found. With the overview of the prototypes, a clear evolution of the hardware is provided as well as some of the design decisions taken.

4 Cable Kinematics and Low Level Control

The low level control (LLC) of the Snake Arm is dependent on two main factors, the design of the body and the electronic hardware. Variations in the design change the kinematics required to achieve the desired result and the requirements of the actuations. Changes in the electronic hardware affects the method of control as different actuators have different requirements and behaviours

The aim is to develop a reliable, accurate and repeatable LLC that is capable to operate all the actuators simultaneously. For the final prototype this needs to be implemented on the smallest possible footprint within the actuation pack while still providing a determinant control loop at a high frequency. Determinism is required such that all the position loops operate at the same time, every iteration, therefore maintaining the synchronicity

4.1 Cable Kinematics

Early on in the design process it was decided to use cables rather than alternatives. Steel cables are pulled to bend the respective sections by the desired amount while at the same time their opposites are extended allowing the movement. To keep stiffness in the system, the cables are pre-loaded, and in an ideal system this tension is kept constant.

There are two standard methods for operating a hyper-redundant manipulator with cables: 3-cables or 4-cables. For 3-cables, any position is fully defined by the position by all three cables. In 4-cable systems, they are treated as independent X-Y pairs. Both will be discussed fully in sections 4.1.1 and 4.1.2 respectively. With both approaches, the Snake Arm design requires cable guides. As Figure 4-1 demonstrates, the cables will take the shortest route between points and when bending this can lead to the

cables colliding with the backbone. Instead extra discs are added which are passive to the system but will affect the kinematics.

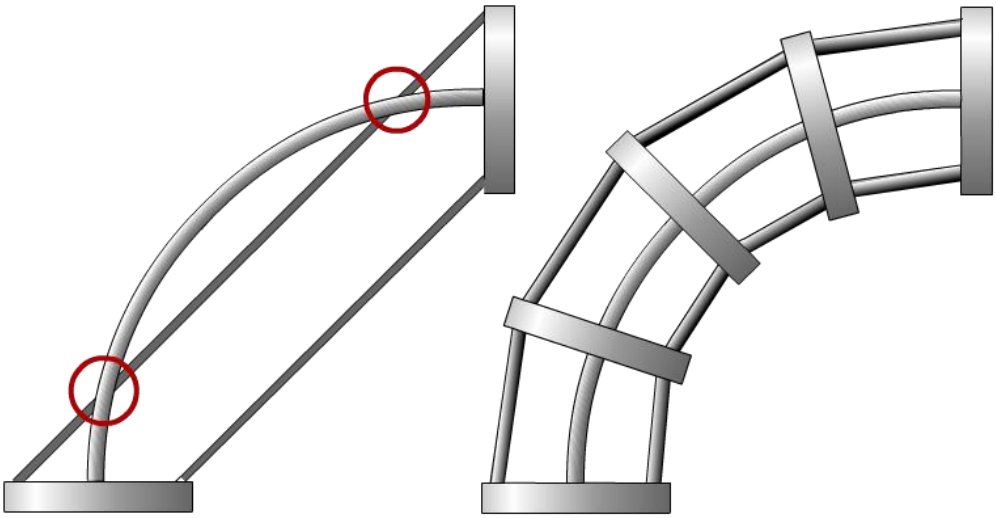


Figure 4-1: Graphical representation to show why guide discs are required

4.1.1 3-Cables

A paper from Clemson [55] by Jones et. al, provides a set of equations to calculate the lengths of the actuation need to be to reach the desired configuration for a basic continuum design. In this system, the holes are equally spaced and numbered such that the second cable hole is on the y-axis as described in Figure 4-2.

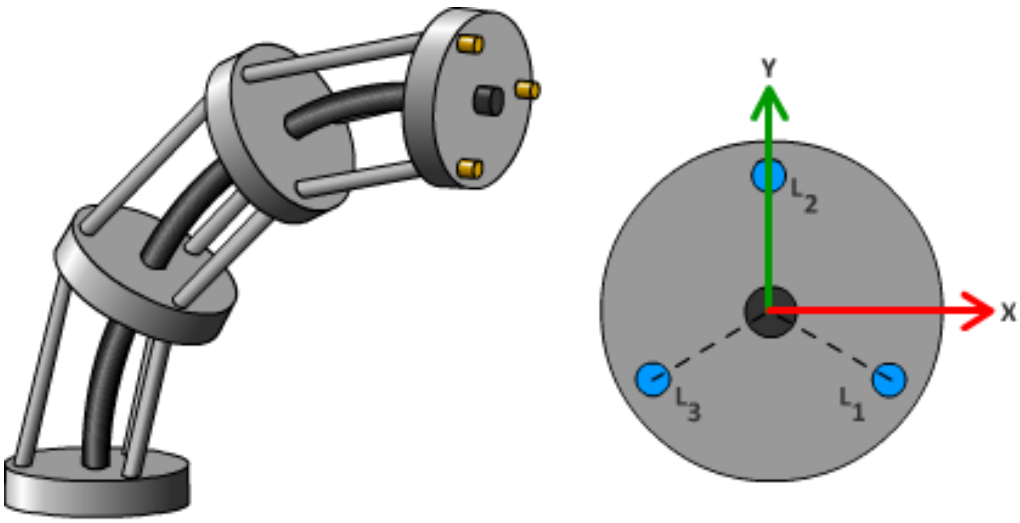


Figure 4-2: A simple continuum backbone design using 3-Cable actuation A) illustrative profile B) top down view of the cable arrangement and assignment

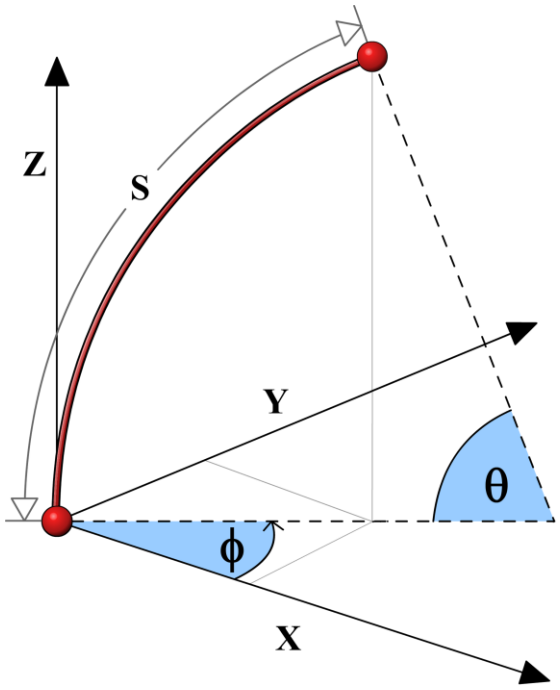


Figure 4-3: Kinematics of a single section

In the equations that follow, S is the axial length of a section, θ is the bending angle and ϕ is the direction as in Figure 4-3. For the section depicted, $L_{1\text{-}3}$ can be defined as

$$\begin{aligned} L_1 &= 2n \sin\left(\frac{\theta}{2n}\right)\left(\frac{S}{\theta} - r\cos(\pi/6+\phi)\right) \\ L_2 &= 2n \sin\left(\frac{\theta}{2n}\right)\left(\frac{S}{\theta} - r\sin(\phi)\right) \\ L_3 &= 2n \sin\left(\frac{\theta}{2n}\right)\left(\frac{S}{\theta} + r\sin(\pi/3+\phi)\right) \end{aligned} \tag{4.1}$$

where *n* is the number of guide discs and *r* for the radial distance of the hole pattern. In (4.1) $L_{1\text{-}3}$ are defined as the absolute lengths, but they can equally be described as the change in length by subtracting S from their values. Jones also provides the inverse equations, that is how to calculate S, θ and ϕ from the cable lengths.

$$
\begin{aligned}
S &= \frac{L_1 + L_2 + L_3}{3} \\
\theta &= \frac{r\left(L_1 + L_2 + L_3\right)}{2\sqrt{g}} \\
\phi &= \arctan\left(\frac{L_3 + L_2 - 2L_1}{\sqrt{3}\left(L_2 - L_3\right)}\right)
\end{aligned}
\tag{4.2}
$$

Where

$$
g = L_1^2 + L_2^2 + L_3^2 - L_1L_2 - L_2L_3 - L_1L_3
$$

4.1.2 4-Cables

Using 4 cables to control a section makes a more intuitive equation. Basically each pair of cables can be treated as the X or Y component of the desired motion. One cable of each pair is chosen to be dominant, this is used to set the polarity of the sign meaning which direction and extension or retraction relates to.

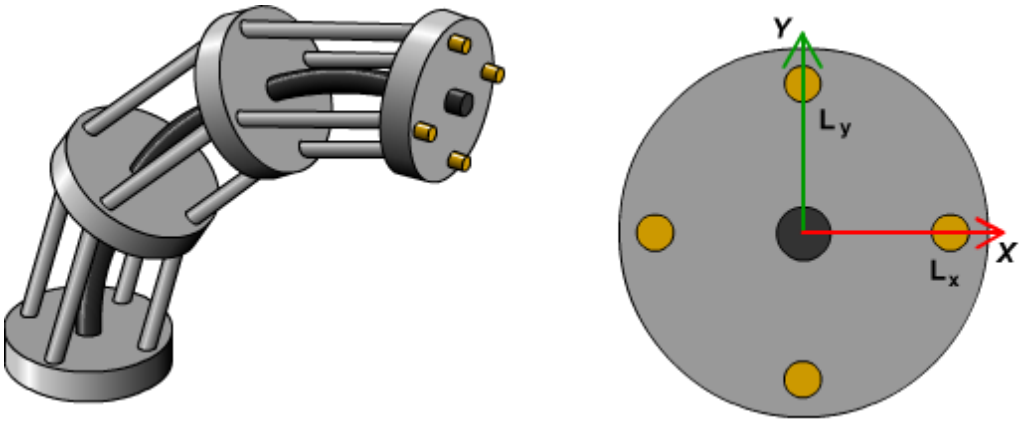


Figure 4-4: A simple four cable design A) Trimmetric view B) Cable assignment

For the simple continuum design, like that depicted in Figure 4-4, the equations for four cables are similar to (4.1) only the hole locations have changed

$$L_x = 2n \sin\left(\frac{\theta}{2n}\right)\left(\frac{S}{\theta} - r\cos(\phi)\right)$$
$$L_y = 2n \sin\left(\frac{\theta}{2n}\right)\left(\frac{S}{\theta} - r\sin(\phi)\right) \quad (4.3)$$

where L_x and L_y are the X and Y cable pairs respectively. Only two variables are needed as the opposite cables are reflections of their partners; that is the amount one side pulls is equal to the extra slack the opposite requires.

4.1.3 Dealing with Multiple Sections

The Snake Arm does not consist of one section. As such, it is important to understand how several sections are calculated. There are two main factors that need to be addressed. These factors can be solved the same way, no matter if the system is using the 3 cable or 4 cable method.

Firstly, due to the extrinsic nature of the design, the movement of one section affects the cables for every subsequent section. Therefore, each section needs to compensate for the position of its predecessors. This is easily achieved by repeating the calculations of previous sections and summating the final lengths with the configuration of the current section as Figure 4-5 depicts. For this reason it is easier to use the change in lengths rather than the absolute as it makes more sense within the LLC.

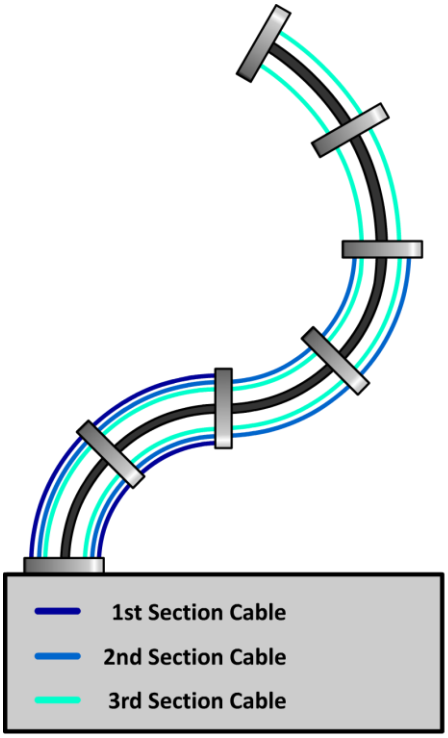


Figure 4-5: Cables run through their preceding sections, meaning what action is taken must be counteracted by subsequent sections

However, the equations so far have assumed that the cables run through the same holes. While this is physically possible with concentric tubes, it is entirely impractical for a snake the scale of the specification. As such, another set of holes will be used for the next section. Since this set will be shifted around the same axis all that is required is a simple correction. This shift can be a change of the pattern radius, an axial rotation or a combination of both, see Figure 4-6 for a diagram.

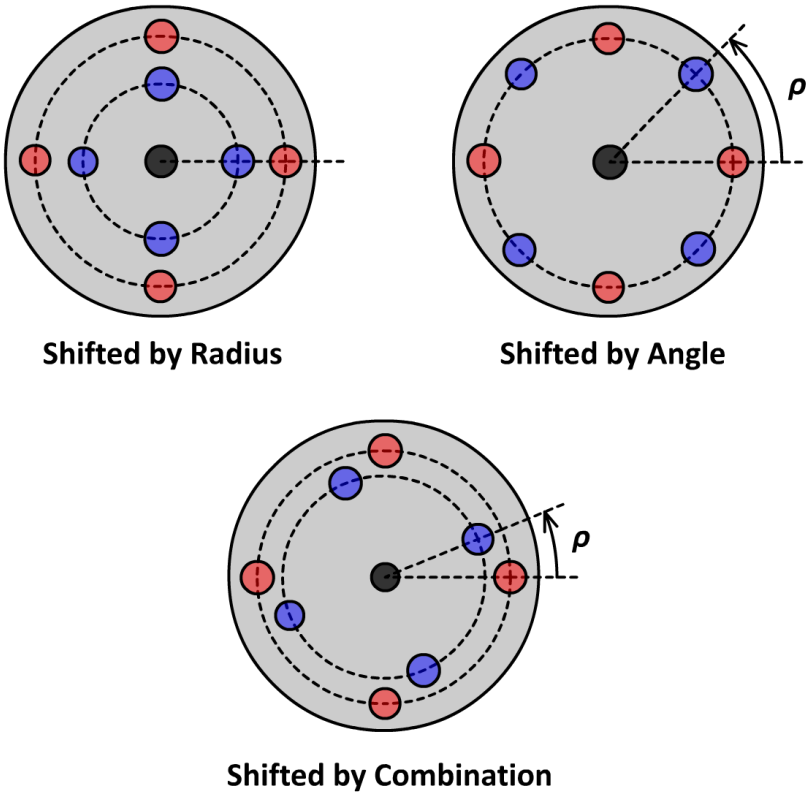


Figure 4-6: Cables cannot run through each other, each section requires its own set of holes shifted by one of the methods shown

Changing the radius is as trivial as using a different value for the variable in equations (4.1) or (4.2). For a rotation, the angle from the default arrangement is known as the phase, ρ. By taking this angle away from ϕ you treat the section as if the desired direction is slightly offset.

$$L_x = 2n \sin\left(\frac{\theta}{2n}\right)\left(\frac{S}{\theta} - r\cos(\phi - \rho)\right)$$
$$L_y = 2n \sin\left(\frac{\theta}{2n}\right)\left(\frac{S}{\theta} - r\sin(\phi - \rho)\right) \quad (4.4)$$

These two factors can be combined into a single function by creating a nested for loop pair that ignores sections once the iteration has passed, as the pseudo code in Figure 4-7 entails.

$$
\begin{aligned}
&for \quad i = 1 \rightarrow \text{number of sections} \\
&\quad for \quad j = i \rightarrow \text{number of sections} \\
&\qquad \left[l_x, l_y\right] = f\left(\theta(i), \phi(i), r(j), phase(j)\right) \\
&\qquad xL_j = xL_j + l_x \\
&\qquad yL_j = yL_j + l_y \\
&\quad end \\
&end
\end{aligned}
$$

Figure 4-7: Pseudo code to calculate the combined length change for each section, where i is the index of the section, and j is the index for the phase and radii settings. f is the kinematic function and xL, yL are arrays of the combined length changes.

The key point to notice, is the nested ***for*** loop starts from index *i*. So on the first iteration the base section is calculated for phase and radii of all the sections, then as the second iteration is computed except the phase and radius of the first section is ignored. Afterwards the values for the second section's phase and radius are summated.

4.1.4 Pros and Cons of 3 Cables or 4 Cables

There are clear advantages and disadvantages for both methods, furthermore these are perfectly aligned with each other. For example, the 3-Cable method requires a motor per cable while the 4-cable allows a pair of cables to be operated by one motor. However, having individual motors makes tensioning the system simple while the conjoined method is more complicated and can lead to only one side in tension. Again the countering argument is that due to the tensioning, in the 3-cable method all cables must move in sync to maintain the intended behaviour leading to a complex control problem, while with 4-Cables there are two simple and separate position control loops per section.

For the project the biggest constraints are the size and weight of the final system. As such it was decided that a 4-Cable method was to be used.

4.2 Cable Kinematics for Different Designs

So far the equations have only dealt with a simple continuum backbone design. With a different backbone architecture comes different behaviour. Through several complications, as discussed below, it was decided that the generic design was not suitable for the size or scope of the project.

4.2.1 Partially Rigid Continuum Spine

After problems with the super-elastic NiTi, used for the backbone, not creating a uniform curve when bending the design was altered to the one depicted in Figure 4-8. Rather than remove the central continuum backbone, small metal tubes were added to make a rigid length within the design.

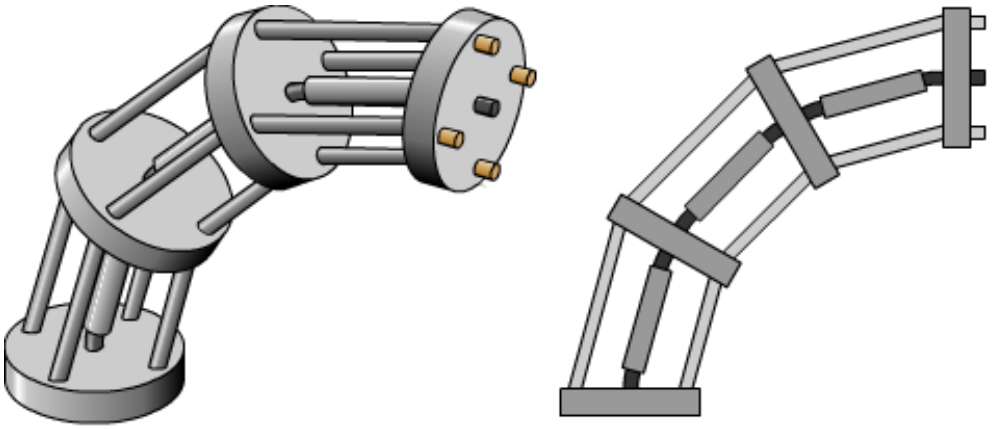


Figure 4-8: A Partially Rigid Backbone design A) Trimetric profile B) Side Cutaway

This means the backbone now acts similar to two spherical joints with a rigid link, rather than a curve. As such, the kinematics need to be changed. Using this approximation, it is possible to treat each sub-section as a trapezium as shown in Figure 4-9 assuming that all the sub-sections share the bending equally. Since the kinematics are only interested in the change in length the fixed sections can be ignored, such as the cable length across the thickness of the guide discs.

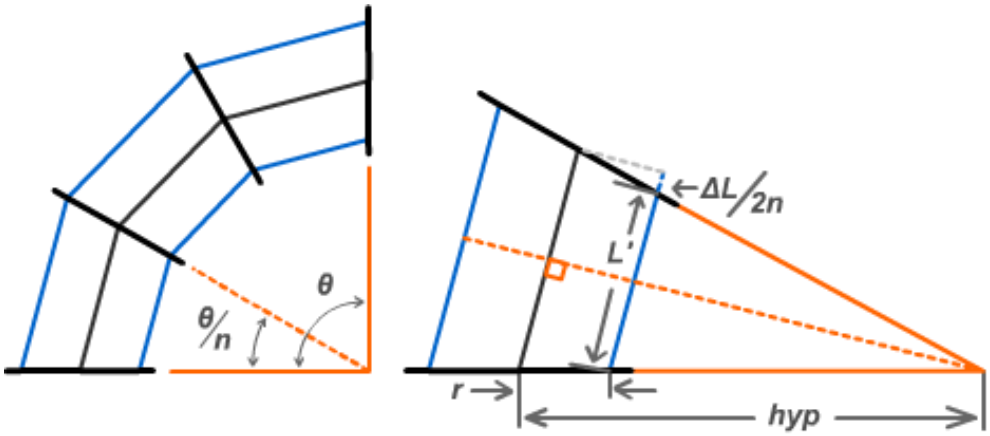


Figure 4-9: geometric representation of the design to aid mathematical explanation

By the geometry of a trapezium, it can be shown that the change of length is equal to the difference between the new cable length, L', and the central length. This can be calculated by trigonometry such that

$$\Delta L = 2n(hyp - r)\sin\left(\theta/_{2n}\right) - 2n(hyp)\sin\left(\theta/_{2n}\right)$$

which can be simplified to

$$\Delta L = -2rn\sin\left(\theta/_{2n}\right) \tag{4.5}$$

and since the four cable kinematics are just X and Y components the complete equations can be found to be

$$\begin{aligned} \Delta L_x &= -2nr\sin\left(\frac{\theta}{2n}\right)\cos(\phi) \\ \Delta L_y &= -2nr\sin\left(\frac{\theta}{2n}\right)\sin(\phi) \end{aligned} \tag{4.6}$$

The designer, Mr. Xin Dong, calculated the actual kinematics without the assumption that the joints behave in a discrete fashion. In comparison, the errors in using this method were found to be 0.5mm at most or 2% as evident by the graphs in Figure 4-10. Given the kinematics provided in [78] are more complex, it would be acceptable to use this approximation to save computational resources if required.

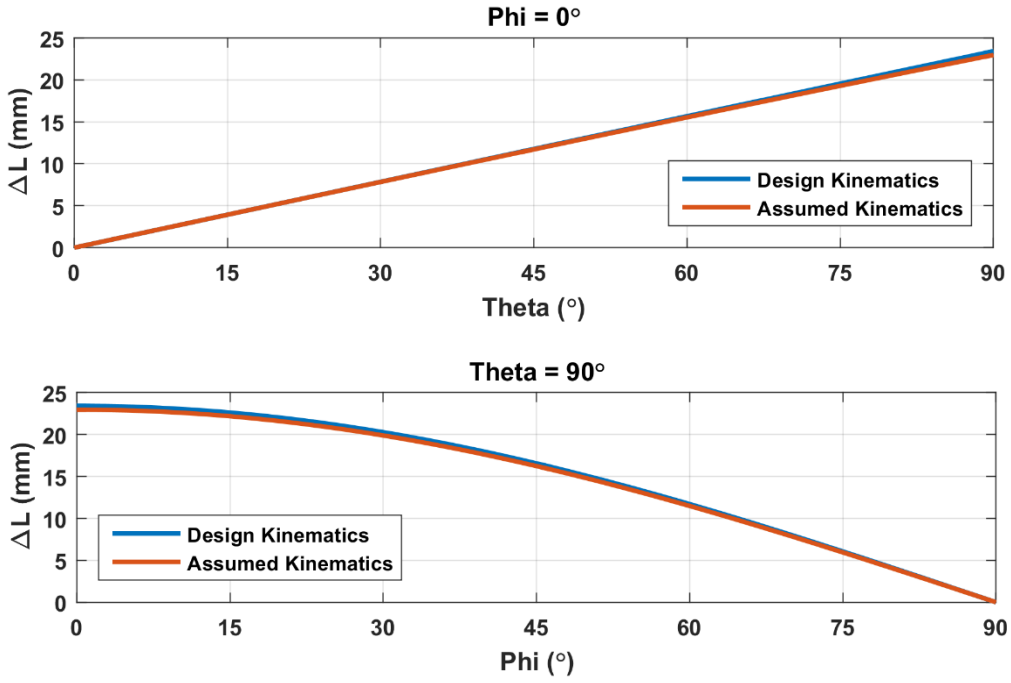


Figure 4-10: Comparison of change in cable length to validate the assumption in the trapezium kinematics

The inverse of the kinematics in (6) can be shown to be

$$\theta = 2n \arcsin\left(\frac{\sqrt{\Delta L_x^2 + \Delta L_y^2}}{2nr}\right)$$
$$\phi = \arccos\left(\frac{-\Delta L_x}{\sqrt{\Delta L_x^2 + \Delta L_y^2}}\right) \quad (4.7)$$

unless ΔL_x AND ΔL_y equal zero, then both θ and ϕ equal zero.

4.2.2 Segregated Bending for Final Prototype

The final prototype does not use a central backbone as this design suffers from a twisting motion. Basically the backbone has no resistance to rotation about the central axis, and the control is not implemented in a manner that can react against this motion. To prevent this, the design is instead segregated into more sub-sections, as represented in Figure 4-11, with small flat flexors which alternate in plane. By doing this each sub-section can only bend on one axis and cannot be twisted.

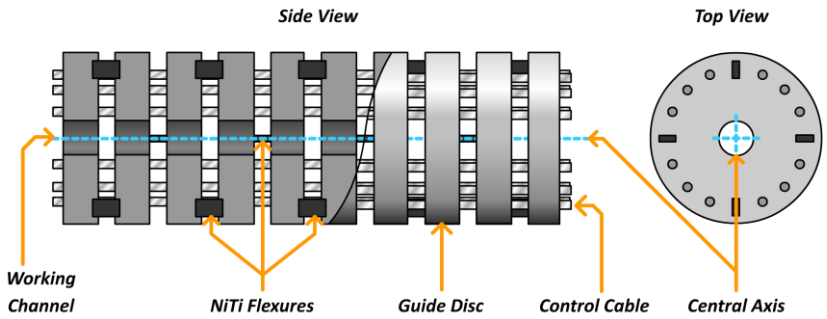


Figure 4-11: Graphical representation of the segregated design used in the final prototype

However, this does make the kinematics more convoluted. Since the movement will be a combination of different bends in the opposing flexors, as demonstrated in Figure 4-12. So while the system can be viewed similar to the trapezium approximation it would not make sense for all values without the system being split into X and Y components.

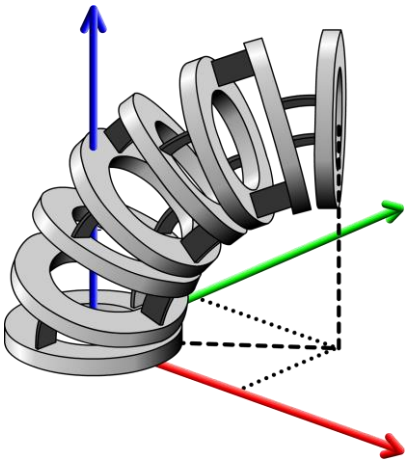


Figure 4-12: Exaggerated concept to demonstrate behaviour of the segmented design

4.3 Observations in Cable Kinematics

Through the investigation in the kinematics, some observations were made that not immediately obvious. Firstly, the change in cable length is not dependent on the section length. While (4.1), (4.3) and (4.4) use the section length, this is only because the equations are calculating absolute length. Instead the magnitude of the changes is dependent on the cable hole pattern radius. This is the radius varied in (4.4) and shown in Figure 4-6. As Figure 4-13 shows this relationship is linear, and increases at a gradient of 1.56 for a 90° bend. So it would seem the best solution to keep the actuation pack compact is to have a small diameter arm. However while this may decrease the change in cable length, the force requirement is increasing with an inverse exponential relationship. This leads to the design having to balance the force requirements with the thickness of the Snake Arm to match the performance of the motors with space requirements of the actuation pack.

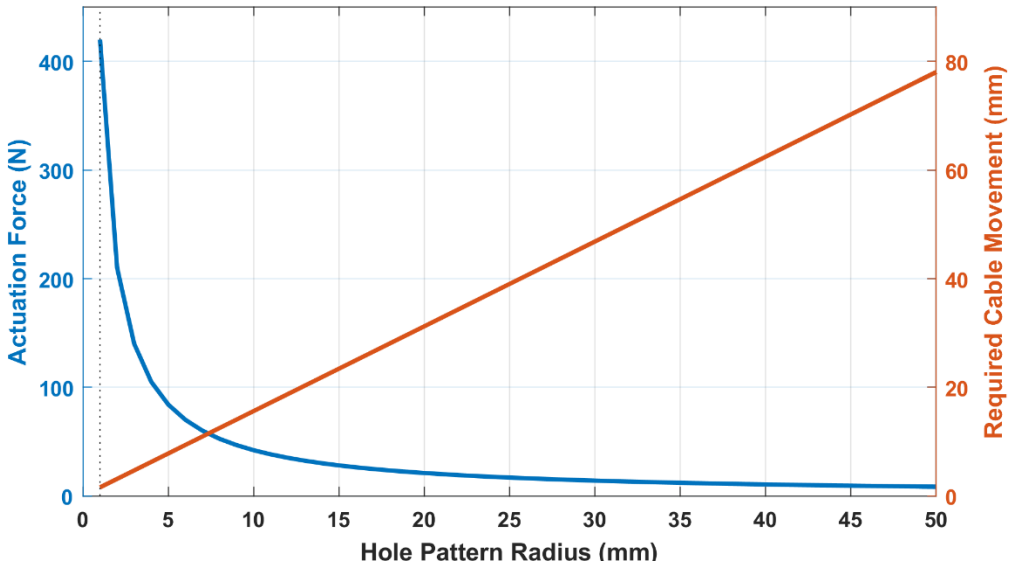


Figure 4-13: Graph to demonstrate how the hole radius affects the force and the movement required to actuate a single section.

4.4 Hardware for Actuation

To convert the result of the kinematics into a mechanical action it requires a combination of devices within the actuation pack, as illustrated by Figure 4-14. The cables are directly driven by a motor which, in turn, are controlled by its own speed controller. To know their position most motors include an encoder. All the signals run in and out of a main controller which generates the signals to operate the speed controllers and interprets the returned quadrature pulses. The Low Level Control (LLC) is implemented on the main controller.

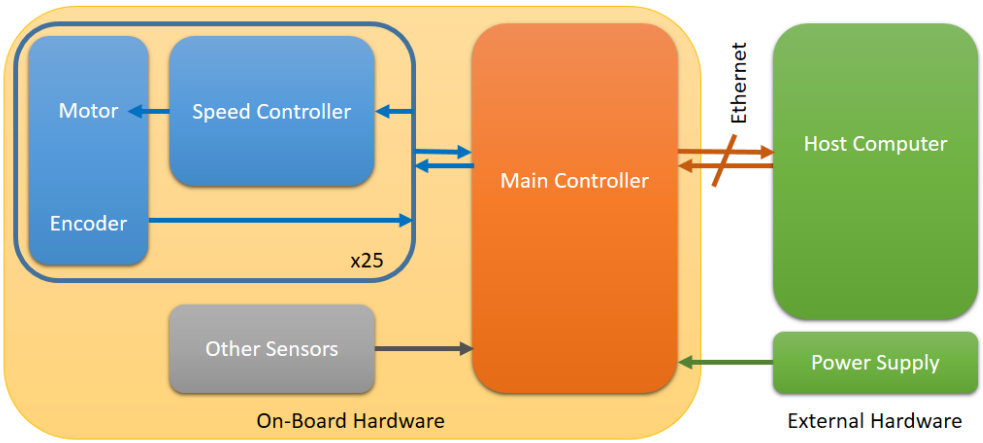


Figure 4-14: Layout of the electrical hardware for the final prototype

Throughout the project different motors and speed controllers were used. For the first prototype simple stepper motors were employed while in later systems DC motors were employed. For size and simplicity, the second demonstrator was implemented with speed controllers designed for hobbyist radio controlled vehicles, as such they respond to a variation of pulse width modulation which operates at 50 Hz. This is significantly slower than the last two prototypes that could function at 20 kHz, meaning a faster response time as well as less mechanical noise. Position of the motor is recorded in encoder pulses, which necessitates a conversion with millimetres for the kinematics.

4.4.1 Cable Actuation

There are multiple approaches that could be used for actuating the cables [79]. Three different mechanical interfaces were used during the project. The stepper motors that were implemented in the first prototype operated linearly and included a simple conversion from the manufacturer of 1 turn equals 1.27mm, however they were placed out of alignment leading to a diagonal movement by the cable as shown in Figure 4-15a.

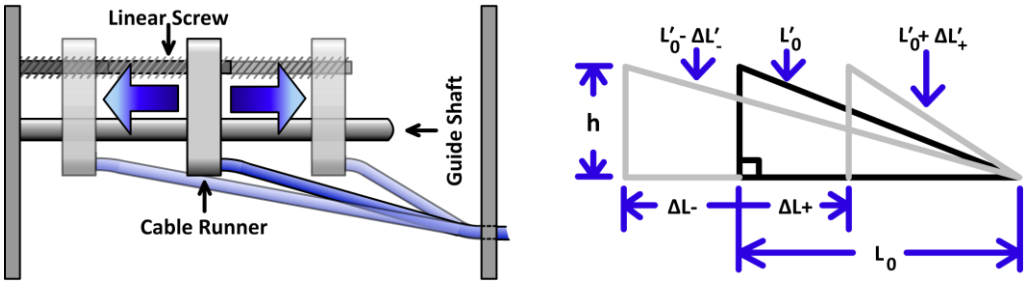


Figure 4-15: The misalignment within the first demonstrator A) Depiction of the linear actuation in operation B) Mathematical geometry of the behaviour

The cable is tensioned in a central position along the guide shaft, meaning the runner it is attached to is located at a distance of L_0 from the cable guide plate. Conversely, due to the misalignment of h, the actual length of cable is L'_0. By basic trigonometry, this is

$$L'_0 = \sqrt{L_0^2 + h^2} \tag{4.8}$$

which means that for any change the following is true

$$\Delta L \neq \Delta L'$$

and more importantly

$$\Delta L_+ \equiv \Delta L_- \quad but \quad \Delta L'_+ \not\equiv \Delta L'_-$$

as such, to get the desired cable movement, ΔL', the runner needs to be moved according to

$$\Delta L = \sqrt{\left(L_0' + \Delta L'\right)^2 - h^2} - L_0 \tag{4.9}$$

and since L' is unlikely to be known prior (4.8) can be substituted, making the full expression

$$\Delta L = \sqrt{\left(\sqrt{L_0^2 + h^2} + \Delta L'\right)^2 - h^2} - L_0 \tag{4.10}$$

All the DC motors in the other prototypes used spools for storing the cable due to size restrictions versus the amount of length change required. The second demonstrator had a flat spiral design, Figure 4-16a, while the final systems used a vertical helix method, Figure 4-16b. Both designs used the 4-cable basis that while one cable is pulled another is let out equally.

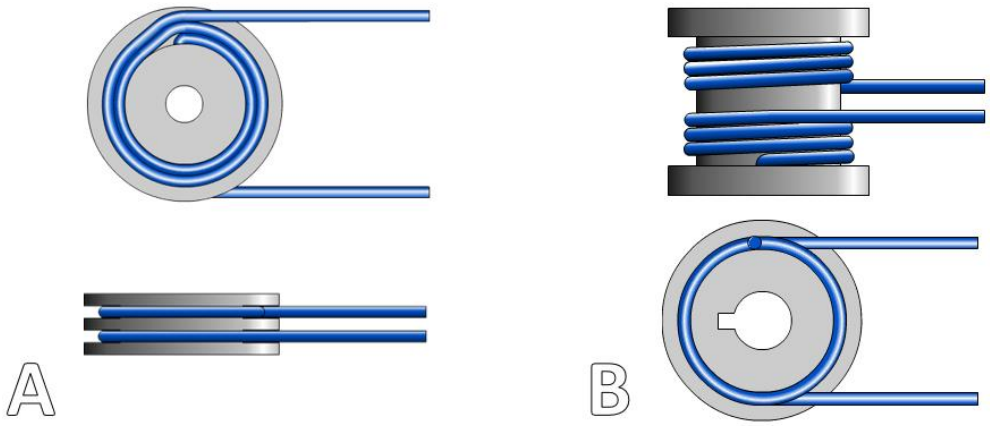


Figure 4-16: Different methods of spooling cable A) Horizontal Spiral B) Vertical Helix

For a spool design the conversion between the kinematics and the motor is

$$\Delta\alpha = \frac{\Delta L}{r_{spool}} \tag{4.11}$$

where Δα is the change in the rotation position for the desired change in cable length, ΔL. However as the spiral arrangement is used the radius varies and therefore the length change would not be consistent. To compensate, it was modelled as an Archimedes Spiral defined in polar form as

$$r = r_{\min} + \frac{\lambda}{2\pi}\theta \tag{4.12}$$

with λ being the pitch, and the arc length, *l*, using

$$l = \int_0^{\theta} \sqrt{r^2 + \left(\frac{dr}{d\theta}\right)^2}\, d\theta \tag{4.13}$$

and then simplified by fitting a polynomial to the curve, the comparison between these is displayed in Figure 4-17a. Taking the point where the number of turns on each side is equal as a zero point and then by assuming the tensioned side will always be the side with the higher radius, the curve can be reflected around this point to create the graph in Figure 4-17b. Given the imbalance between the tensions on the extending versus the retracting cable becomes relatively high, the vertical spool design was implemented instead so that this was avoided.

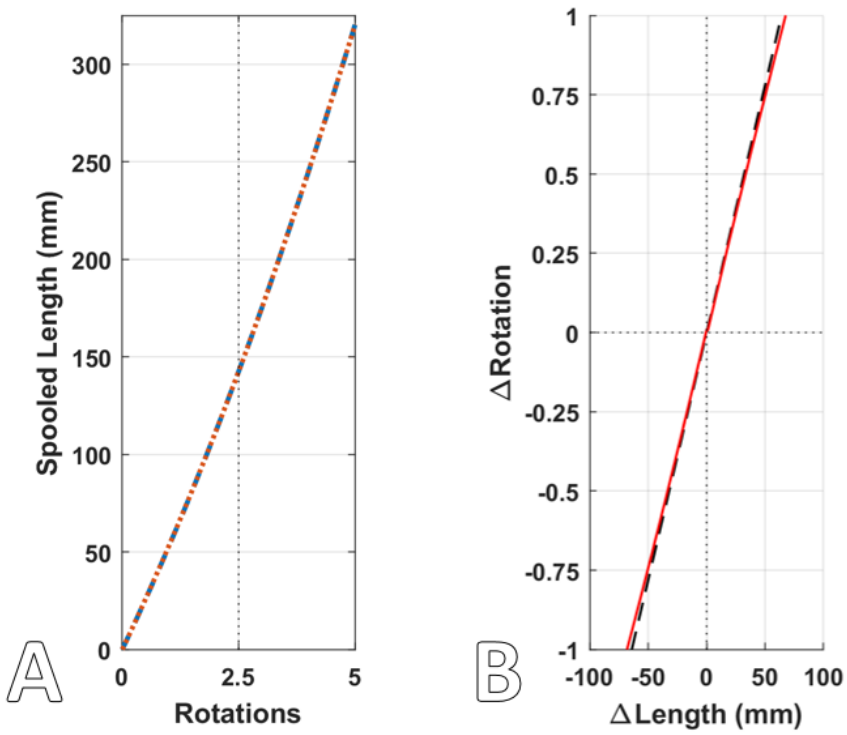


Figure 4-17: A) Comparison of Polynomial Approximation for Spiral Behaviour, B) Function Applied to get the change in Rotation for the desired change in Length.

The intention with the vertical spools is to have the equilibrium point in-line with the cable guide holes. That way the diagonal shift as the helix is unwound is equidistant to the similar shift on the complimentary side that is spooling up, as shown in Figure 4-18. This can then be compensated for in the same way as the linear case. Another advantage taken by the design is to reuse the space between the two coils, since when one is at its most coiled length the other will be at its least.

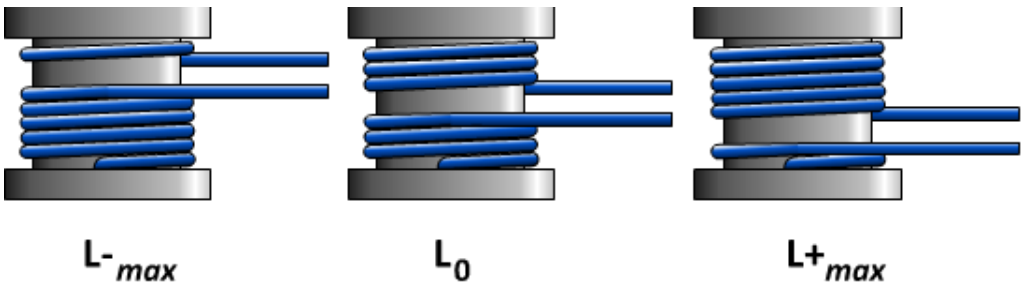


Figure 4-18: Illustration of how the cable spooled shifts between the extremes of the range.

To tension, the top half can be turned independently from the bottom half where a screw is used to lock the halves together. However in practice this tensioning method made the balanced position difficult to maintain as one side had to be turned by hand. While this has no direct effect on the cable movement, it does however, make the tension applied subjective to the person performing the task and the central position required recalibrating.

4.4.2 Main Controller

The decision of which hardware to implement the prototypes with came down to a few key areas. Firstly the size, the design specification is very clear on the available space to be used for the Snake Arm and its actuation pack. Numerous digital I/O lines are required to run all the motors, and other electronic devices on-board. The flexibility allowed for rapid testing of code during development. Finally, the computing power to support all the operations.

For these reasons, several common solutions were rejected. This includes the Arduino, the Raspberry Pi and PIC Microcontrollers as they cannot provide enough connections or computational power. It would have potentially have been possible to use the MATLAB Real Time Workshop with an xPC Target however there was not hardware found to match that fit the criteria.

The control board chosen to operate the hardware was the National Instruments (NI) single-board Reconfigurable Input / Output (sbRIO) 9626. This features a real-time operating system on a 400MHz processor combined with a Xilinx Spartan-6 LX45 Field Programmable Gate Array (FPGA). With this model there are 100 Digital I/0 lines, 16 Analogue inputs and 4 Analogue outputs. Two NI 9626 sbRIOs are required to control the final prototype despite the number of signals it can handle.

4.5 Control Strategy

The low-level control was implemented as a series of simple position control loops. By keeping the movements small, the system can be considered statically and avoid the additional complexity of dynamics. This ensured that more time was available to deal with the more critical aspects of the project, and decreed acceptable as the potential operating speed was sufficient to fulfil the scope.

Three methods were implemented during the course of the project, Bang-Bang, Proportional, and Proportional with a Psuedo-Derivative action. All were implemented on the FPGA. While the FPGA is powerful it does have a limit to the number of calculations that can be performed, this meant that to achieve all the desired tasks, namely 25 position control loops, certain operations had to be excluded to allow compilation.

Typically this is because the FPGA works in physically coded binary signals, such that floating point maths is impractical and fixed-point maths is possible but relatively time consuming. This means that actions on the FPGA are commonly implemented as integers, and operations like division need to be avoided.

4.5.1 Bang-Bang

A “bang-bang” method was used for the first and the second demonstrator. The first system was designed as a quick educational tool rather than a precision manipulator and did not require a complex control scheme. While the second prototype had several issues with cheap hardware. The hall-effect encoders provided by the motor manufacturer gave different pulse per revolution depending on the speed or direction. Even though they were replaced, a more effective control strategy was never implemented on this prototype.

This method consists simple of turning the motors on a set speed and stopping when the within a threshold of the desired position, as visualised in Figure 4-19a. As such the system is an extremely simple program, as conveyed in Figure 4-19b, consisting of calculating the difference between the desired and actual, determining the sign and moving the motor in that direction with the pre-defined speed unless within the threshold.

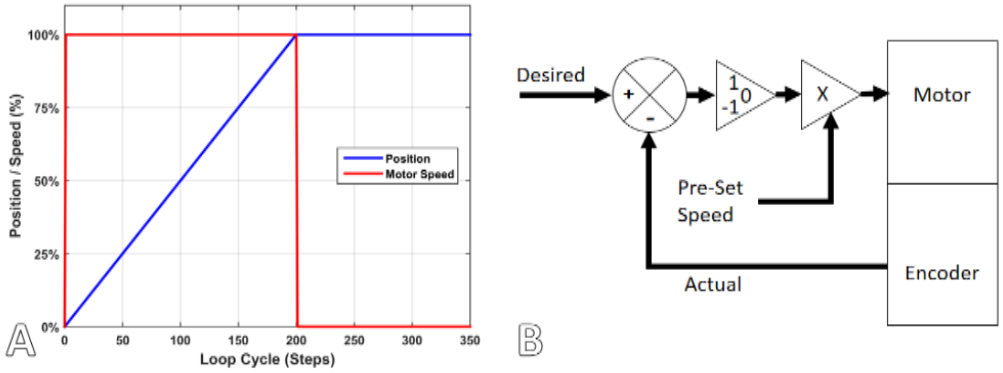


Figure 4-19: Bang-Bang Control A) Position vs. Time B) Control Setup

Naturally, there are two distinct flaws with this method which is why it is rarely implemented in precision robotics. Firstly it expects the motor to start and stop instantaneously which causes high current peaks in the system. To limit the effect of this and to prevent excessive mechanical wear, the speed is often set slow. Secondly the accuracy of the system is determined by the threshold, but if it is too small the system will oscillate as it repeatedly overshoots the target. To compensate for this there is only one option, run the motors slower.

4.5.2 Proportional Control

As expected, the first two demonstrators operated relative slowly. So a proper implementation was sought for the final prototype. The most common control technique is proportional gain. By taking the error between the desired position and the actual and scaling by a gain factor a corrective action can be created that

decelerates as it approaches the target position, Figure 4-20. By careful choice of the gain, the steady state error (SSE) can be reduced to fractions of a millimetre of cable pull.

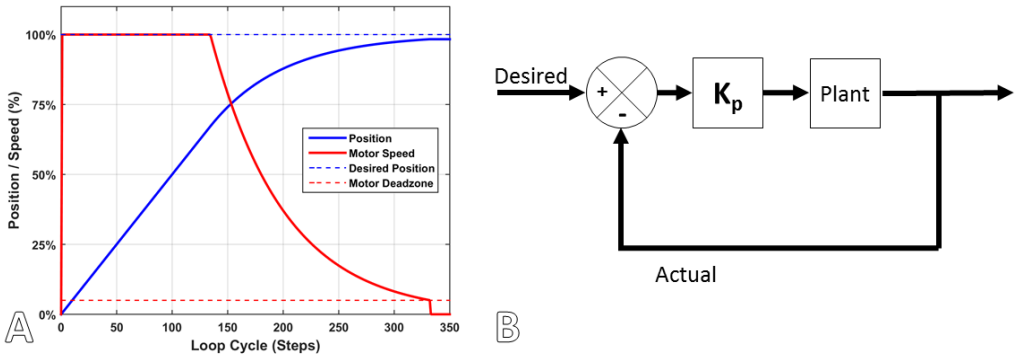


Figure 4-20: Proportional Control A) Simulated Behaviour B) Control Loop

Logically, given the commonality of this method, there exists functions to perform this task using the NI hardware selected. However, these sub-Vis are not optimised for a system running 25 motors and could not be implemented together and compiled. Therefore a minimalized implementation was required.

The code developed runs within the same while-loop that generates the pulse-width modulation (PWM) that controls the speed of its respective motor. After each period the proportional controller determines the next duty cycle. The gain in this instance also needs to scale the significant difference between the order of magnitude of the encoder count and motor duty. However a gain value around 0.03125 would traditionally require division, instead the value can be scaled by powers of two. This is because shifting a binary number to the left or right acts the same way as multiplying or dividing by two respectively. Therefore, since 0.03125 is equal to 2^{-5}, the value can be shifted 5 times.

4.5.3 Proportional Control with Pseudo-Derivative

The proportional control implemented provides a fast and accurate movement that fulfils all the requirements of the design scope. However, it still demands the motor starts instantaneously causing a current spike. For one motor this could be ignored, a 2A spike is not going to damage the hardware, but once combined with the other motors these spikes can exhibit peaks as high as 50A. Not only is it hard to support this amperage within the hardware, but it is also outside safe working practices.

To combat this, a derivative gain can be introduced to the applied action. This value is used to regulate the rate of change in the action as the graph in Figure 4-21a demonstrates. Through this, the acceleration of the motor will be constricted and therefore reducing the current spikes. As the loop in Figure 4-21b depicts, the derivative is implemented as a scale factor applied to the difference between the previous action and the planned action. This is then summated with the planned action and, due to the signage of the difference, reduces the result. This, in effect, works similarly to the practice of Cascade Compensation [80]. That is, two control loops in-line where the second loop's set point is the output of the first. In this case, the first and second loop are a proportional and a derivative respectively.

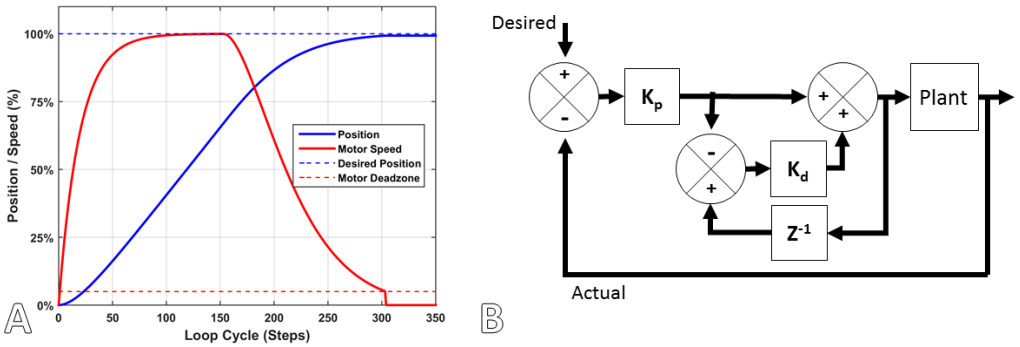


Figure 4-21: Proportion-Derivative Control A) Simulated Behaviour B) Control Loop

Unfortunately, the limitations of the hardware again prevent this from being implemented and compiled for 25 motors. Instead a pseudo-derivative is taken. This

is presented as a fixed limit on the change in speed, and stop it changing by more than 5% per each PWM period. Given this time factor, it still only takes 1ms to ramp the motor up to full speed, and thus has no noticeable effect on the behaviour other than removing the current spike.

4.6 Manual Control

For direct user control the Snake Arm parameters are not intuitive and not easily visualised. This comprehension becomes even harder as more sections are added. As such, a more intuitive interface is required.

4.6.1 Graphical Representation for User Interpretation

Rather than expressing the parameters numerically, they can be represented graphically using a polar co-ordinate plot like Figure 4-22. Using the standard notation for polar co-ordinates, r embodies the magnitude of the bend given by the θ of the snake arm, while the polar θ denotes the rotation characterized by the parameter φ. In comparison, the compass plot appears objectively the same as a top-down view of a single section.

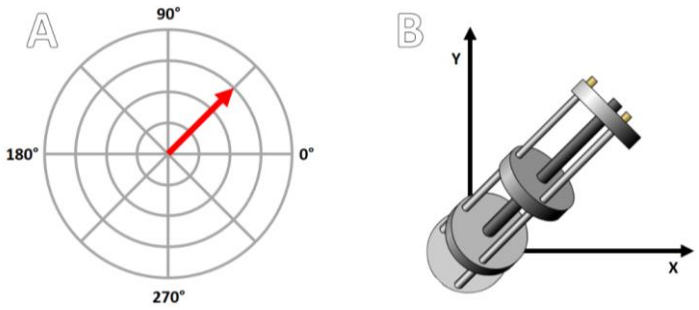


Figure 4-22: Comparison of how similar A) Compass Plot and (B) Top-Down view of single section are

While it is easy to depict two sections by this method, it gets confusing to remember the order with more lines. An alternative representation is shown in Figure 4-23a. This graph has several polar plots along a single axis, allowing the operator to determine the position of each section without having to guess the order. The benefit of this over

just depicting the 3D shape of the Snake is to save the user from re-sizing and rotating the graph to interpret the model on a 2D screen. This is made more apparent when comparing the graphs in Figure 4-23b, both show the same Snake Arm pose from different sides, but the bends in are completely indistinguishable in the other angle.

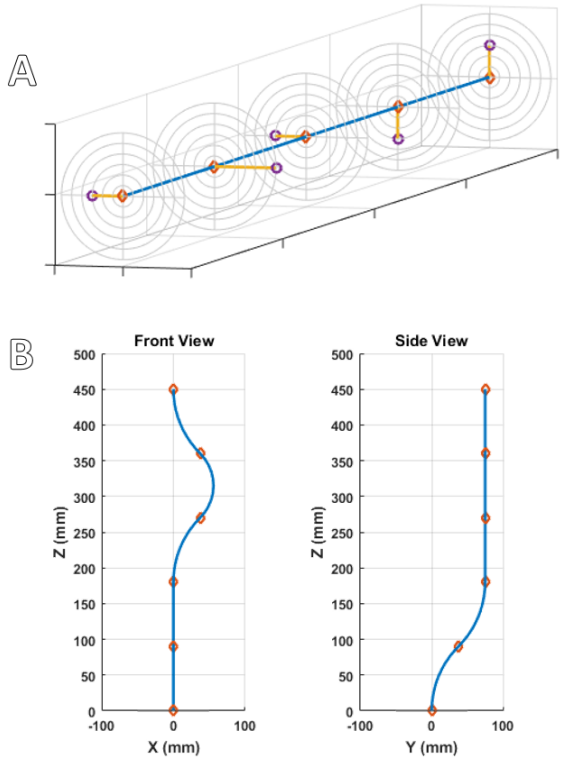


Figure 4-23: A 5-section snake represented as A) a stacked series of compass plots B) the same snake simulated in 3D showing the front and side view

4.6.2 Direct Control

To control the parameters, the user would prefer not to have to enter the parameters manually as this is not intuitive and very error prone. A common alternative is to use sliders, but with two variables per section this can become convoluted and time consuming. The user could drag the lines as they appear on the plots, but this still separates them from the action. As such, the delivered system needs a physical device to interact with.

Taking intimations from the graphical portrayals, there are a couple common control devices that provide similar two-axis movements, Joysticks and Gamepads. Most

joysticks have a single two-axis movement, but some include a mini-thumbstick on the handle to provide a secondary command. While Gamepads include two thumbsticks as standard as well as triggers, bumpers, a D-pad and four buttons. Possible control mappings are shown in Figure 4-24.

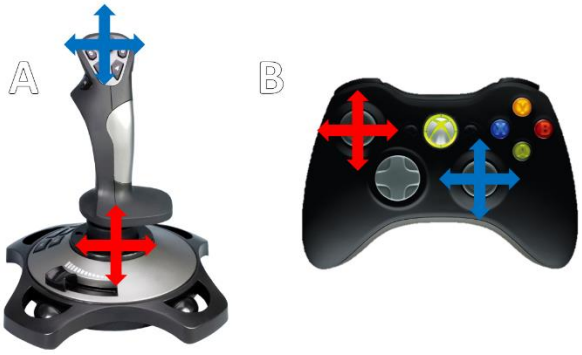


Figure 4-24: Possible control schemes for two sections, 1st axis in red and 2nd in blue A) Controls mapped to a joystick B) Controls mapped to a gamepad

Non-common devices, Figure 4-25, that could be implemented include haptic controller like the Novint Falcon or a Space Mouse developed by 3Dconnexions. Both of these are designed for 3D movement, but would be more useful if controlling the tip within the machining workspace rather than individual sections. Another alternative would be a gaming keypad such as the G13 made by Logitech, this would allow a large amount of commands to be mapped to the keys and information to be displayed on the screen but would limit the user to moving one section at a time.



Figure 4-25: Examples of non-conventional controllers that could be implemented for the control of snake arm: A) Novint Falcon B) 3Dconnexions SpaceMouse Pro C) Logitech G13 Gaming Keypad

Realistically, the user can only operate two-sections at a time. For ease of implementation and familiarity for the operator the gamepad was chosen. More specifically, the Xbox 360 Controller, as shown, since drivers and integration are already implemented under Windows operating systems.

4.6.3 Implementation

There are two methods of implementing the Gamepad to control the parameters. The simplest is to directly relate the two-axis movement to the polar co-ordinate position. Firstly the X and Y movement of the thumbstick is scaled to match the maximum bending angle. This is then converted into the polar co-ordinate form by the standard transformation

$$\begin{aligned} r &= \sqrt{x^2 + y^2} \\ \theta &= \arctan\left(y \middle/ x\right) \end{aligned} \tag{4.14}$$

However, because the Cartesian co-ordinates consider a square space while the polar is circular, values at the maximum bending angles x = 90° and y = 90° for example will result in an r value of 127.28°. As such, the r value must be coerced to not exceed the maximum bending angle. Note well that the conversion must take into account the quadrant of (x,y) or the value for θ will be wrong. This is simply

$$\begin{aligned} &if \quad x < 0 && elseif \quad y < 0 \\ &\theta = \theta + 180° && \theta = \theta + 360° \end{aligned} \tag{4.15}$$

After this the Snake Arm parameters can be extracted directly by the re-assignment given earlier.

The disadvantage of this method is that the change is commanded immediately. This means not only that the rate of change is dependent on the smoothness of the users

input, but they have to maintain the position on the thumbsticks to keep the current position. These issues can be combated with movement interpolation and push-to-move button respectively, but a more intuitive solution is the second possible implementation.

Rather than directly driving the polar position, the user drives the change in the position. That is, the controller position adjusts the rate of change of the movement. This way, when neutral the position is maintained, the speed and precision can be controlled by the operator.

To implement this, first the X/Y axis of the controller is scaled to represent the maximum rate of change desired per loop. The values are then added, this is done in Cartesian space to make the movement linear rather than curved. Once the new values are found they are converted back to polar co-ordinates, and the limits are applied as before. The combined equation can be shown to be

$$r_1 = \sqrt{\left(kx + r_0 \cos(\theta_0)\right)^2 + \left(ky + r_0 \sin(\theta_0)\right)^2}$$
$$\theta_1 = \arctan\frac{ky + r_0 \sin(\theta_0)}{kx + r_0 \cos(\theta_0)} \quad (4.16)$$

where k is the maximum rate of change, while x and y are the thumbstick inputs within the range -1 … 1. The same consideration in (4.15) needs to be include for (4.16).

To test this methodology and for educational demonstrations the manual control was implemented on the first demonstrator. This was done such that the control schemes could be switched to experience the different methods. The limits used allowed the Snake to move a maximum of 90° per section at a maximum rate of 0.5° per loop cycle of 10ms.

How this manual control is integrated with the tip following is discussed later in Chapter 7.

4.7 Accuracy and Repeatability Trials on Third Prototype.

With the development of the low-level position control loops, the accuracy and repeatability of the system needs to be validated, when combined with the cable actuation. The third demonstrator was used for these trials, since it is more uniform and made experimental measurement easier than the final prototype.

4.7.1 Methodology

For measurement a video gauge system was used to track a set of markers along the length of the Snake Arm. From this data it is possible to recompile the shape of the arm. This is then compared to the desired position to calculate the errors. Since the video gauge system works only in 2D, the movements of the Snake where performed on a single vertical plane.

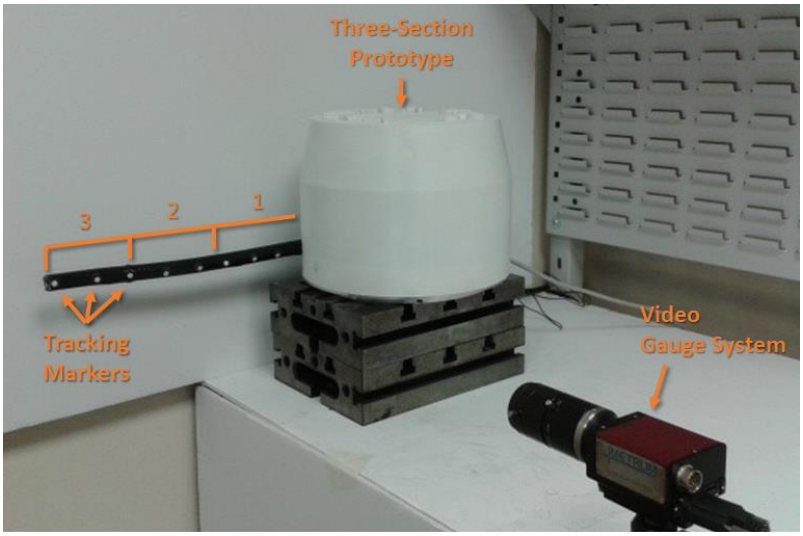


Figure 4-26: Experimental setup of video gauge with the Third Prototype

Accuracy was determined by comparing the end positions with their desired position. The angles to be tested were within a ±5° window of 30°, 45° and 60°. For repeatability,

the Snake was moved to a specified angle and returned to home continuously for several iterations, then the difference in the end position was calculated.

4.7.2 Accuracy Trials

The last section was actuated upwards to a set starting angle as a reference, from this point the section was moved to 6 different points within the ±5° window. The graphs that follow in Figure 4-27 display the measured results against the expected positions. These results are also summarised in Table 4-1.

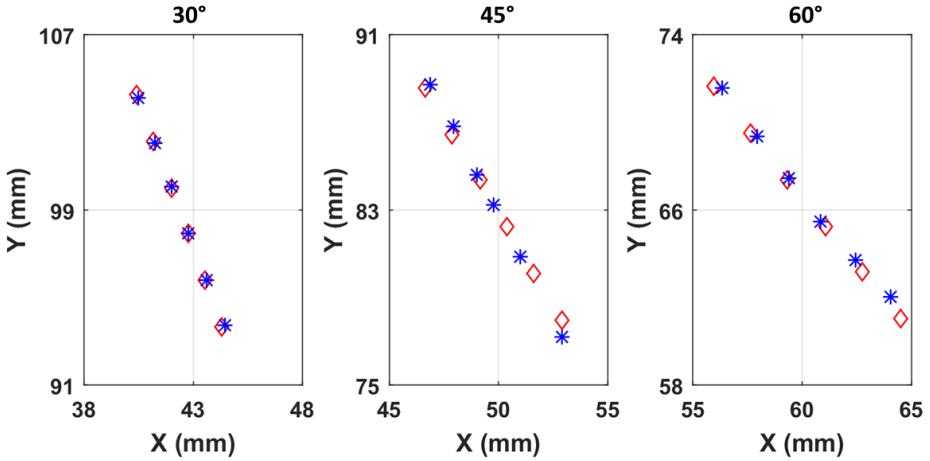


Figure 4-27: Accuracy Trials, where red diamonds indicate the expected position and the asterisks represent the actual.

Table 4-1: Summary of position accuracy results

Bending Angle	**Desired Position**		**Actual Position**		**Error**		
	X	**Y**	**X'**	**Y'**	**ΔX**	**ΔY**	**Euclidean**
	Starting Angle: 30°						
25°	40.41	104.23	40.51	104.1	0.10	0.13	0.16
27°	41.19	102.10	41.25	102.09	0.06	0.01	0.06
29°	41.97	99.97	41.98	100.04	0.01	0.07	0.07
31°	42.75	97.91	42.76	97.95	0.01	0.04	0.04
33°	43.53	95.78	43.58	95.79	0.05	0.01	0.05
35°	44.31	93.65	44.46	93.77	0.15	0.12	0.19
	Starting Angle: 45°						
40°	46.63	88.60	46.89	88.75	0.26	0.15	0.30
42°	47.88	86.47	47.92	86.78	0.04	0.31	0.31
44°	49.13	84.34	49.02	84.61	0.11	0.27	0.29
46°	50.37	82.20	49.78	83.20	0.59	1.00	1.16
48°	51.62	80.07	50.96	80.89	0.66	0.82	1.05
50°	52.87	77.94	52.90	77.20	0.03	-0.74	0.74
	Starting Angle: 60°						
55°	55.92	71.67	56.35	71.55	0.43	0.12	0.45
57°	57.64	69.54	57.92	69.38	0.28	0.16	0.32
59°	59.35	67.41	59.39	67.42	0.04	0.01	0.04
61°	61.07	65.27	60.85	65.50	0.22	0.23	0.32
63°	62.78	63.14	62.45	63.68	0.33	0.54	0.63
65°	64.50	61.01	64.04	62.01	0.46	1.00	1.10
				Average	**0.21**	**0.32**	**0.41**

N.B. All positions and errors are represented in millimetres

The requirements for the system state that it has to be as good as the current manual operation or better. Quantitatively this was defined as a maximum error of 2.5mm and an average around 1mm. Therefore, the results represented above are within the specification. However it would be preferable if the results for 46° and 48° were not a whole degree out.

4.7.3 Repeatability Trials.

For each angle, the end position of the first movement is taken as the comparative point. Then the end point of every iteration for this angle is compared to this initial point. The maximum drift can be determined from these results, as wells as the maximum drift from the average position. This can be represented graphically as depicted by Figure 4-28.

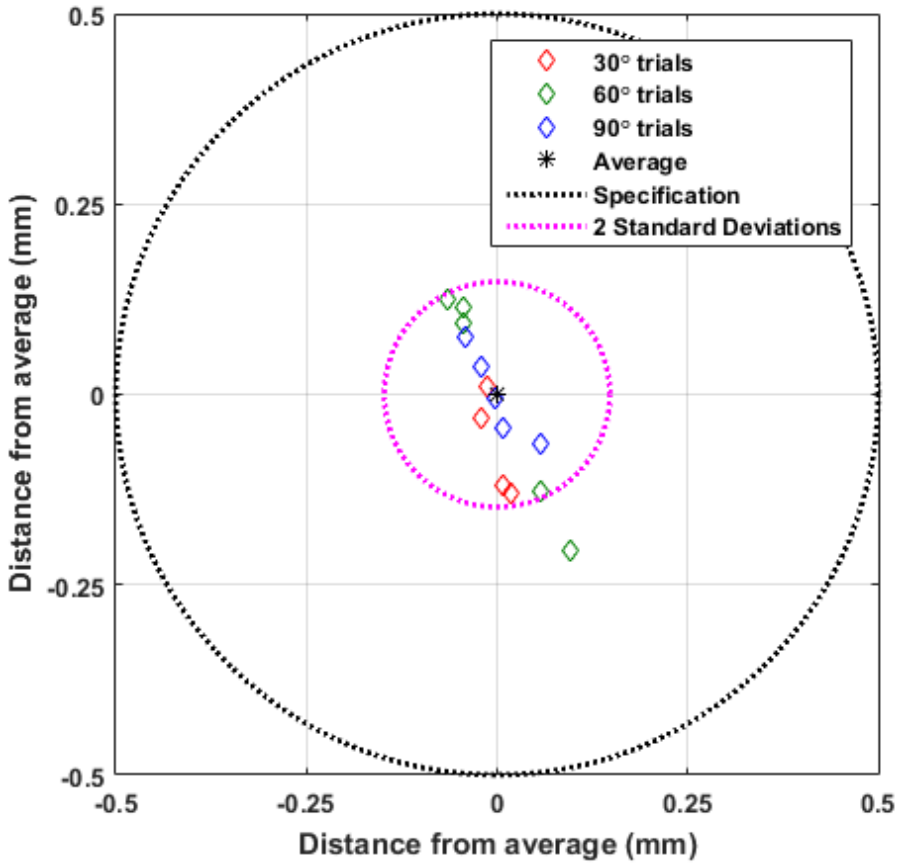


Figure 4-28: Repeatability trials, graph showing the displacement from the average point. The black dotted circle represents a 0.5mm radius and the magenta circle is 2 standard deviations from the mean

Table 4-2: Summary of Repeatability Trial

	30° Trial		60° Trial		90° Trial	
Iteration	**X**	**Y**	**X**	**Y**	**X**	**Y**
1	42.67	95.20	60.02	65.50	87.46	43.52
2	42.65	94.94	59.98	65.58	87.41	43.54
3	42.64	94.90	59.88	65.82	87.40	43.58
4	42.68	94.80	59.88	65.80	87.38	43.62
5	42.67	94.81	59.86	65.83	87.36	43.66
Average	**42.66**	**94.93**	**59.92**	**65.71**	**87.40**	**43.58**

Drift from mean	**Max**	0.27	***Drift from initial***	**Max**	0.40
	Average	0.10		**Average**	0.24
	Std. Deviation	0.07		**Std. Deviation**	0.13

N.B. All positions and errors represented in millimetres

As it can be seen in both the figure and Table 4-2, the first iteration starts further away and the system settles to an average overtime. This has been determined to be a symptom of the mechanical backlash that has been exhibited. As such, the movement is heavily dependent on the initial bias.

4.8 Chapter Summary

This chapter has explored the generic cable kinematics for 3-Cable and 4-Cable designs, how different Snake Arm structures, and how different actuation mechanisms effect the cable behaviour. From this, it was decided to use a 4-Cable system to save weight and to keep the size of the actuation pack down. The size was the same factor that determined a spool design was to be used to actuate the cables, and after learning the difficulties with a spiral concept a helical spool was chosen for the final prototype.

Various control schemes were discussed and tested. Through the requirements of the project the initial concept evolved to be as minimal as possible, such that is could be implemented on the chosen hardware, while providing a fast response that does not cause sharp current spikes.

User interaction was investigated, both how feedback can be displayed and how manual interaction can be commanded. Using similarities between the kinematics, compass plots are used to represent the action of the Snake Arm. A gamepad was implemented with the first prototype and the method to translate thumbstick movements into changes in the pose.

Finally the combined interaction of the kinematics, hardware and control has been tested within the third prototype. Through the experiments, the system has been shown that the snake arm to have a relatively good position accuracy, averaging 0.41mm. The hardware has also been proven to be extremely repeatable, with a standard deviation of 0.13mm from the initial run.

5 Tip Following

5.1 Introduction

Any navigation task can be split into what path does the robot need to take, and how is this path represented in the commands the robot comprehends. Generally, the planning is considered more complicated than the instruction list, as such these steps are usually combined. However this case is reversed within the realm of Snake Arms. Since for any movement a new arrangement of the arm must be found and with ten or more degrees of freedom (DoF) this becomes less trivial. Thus it is advantageous to split the path planning from the physical movement of robot.

As mentioned in the previous chapter, the simplest end-user interface is to operate the Snake Arm manually using a joystick or gamepad. Rather than giving control of all the DoF which can be confusing and non-intuitive, a tip following algorithm can be used to reposition the pose of the snake while the user only needs to be concerned with the tip position. A user could indicate their desired direction using an analogue input, with the aid of a camera feed from the end-effector, and command the advance and retreat of the snake with the available buttons.

In the more traditional path-planning symbiosis, the high level functions will be able to find a path through the environment that avoids obstacles within the tolerance of the tip following. Then the profile can be optimised such that it gives a smoothed curved path throughout. By discretising the movement into small steps, the tip following algorithm can provide a clean motion.

The remit of the tip following is defined as such. It is not aware of the environment, and is not expected to plan accordingly; instead it is focussed on maintaining the path travelled within the predicted performance capability. The arrangements sent are assumed to be executed correctly by the low-level control, the tip following is not responsible for compensating for gravity or inaccuracy in the positioning.

5.2 Geometric Kinematics

Before getting into how the proposed tip following algorithm works, it is important to understand how the Snake Arm can be represented within the computations. This can be expressed either as the parametric variables of the snakes pose, or the Cartesian points that denote its location and occupying space.

Many kinematic equations exist for hyper-redundant manipulators, each with their own advantages and disadvantages. For the purpose of the tip following, the kinematics by Neppalli [66] are to be used. These equations were chosen since they were applied to continuum arms, ignored gravity and not being computationally complex. However it is possible to use any kinematics that represent the behaviour of the robot to be used with the Tip Following algorithm, the principal is the same.

5.2.1 Representing a Single Section

For any given section of the snake, it is assumed that the body creates a continuous curve. As such the arc created can be represented by the section length, S, the bending angle, θ, and the direction angle, ϕ, as shown in Figure 5-1. While Neppalli prefers to use the curvature, it was deemed more intuitive with the bending angle. It is also important to note that the direction angle is counter-clockwise from the x-axis.

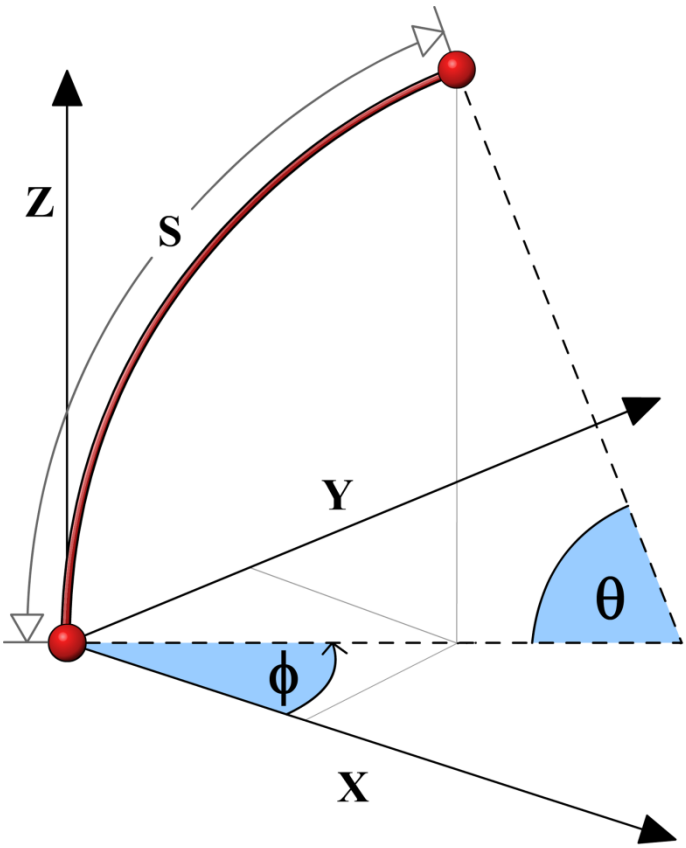


Figure 5-1: Kinematic parameters of a single section

From these parameters, the tip position can be found as

$$\begin{bmatrix} x \\ y \\ z \end{bmatrix} = \begin{bmatrix} \frac{S}{\theta}\cos(\phi)(1-\cos(\theta)) \\ \frac{S}{\theta}\sin(\phi)(1-\cos(\theta)) \\ \frac{S}{\theta}\sin\theta \end{bmatrix} \tag{5.1}$$

and the rotation matrix is between the original coordinate frame and the tip is

$$R = Rot_z(\phi)Rot_y(\theta)Rot_z(-\phi)$$

$$R = \begin{bmatrix} \cos(\phi) & -\sin(\phi) & 0 \\ \sin(\phi) & \cos(\phi) & 0 \\ 0 & 0 & 1 \end{bmatrix}\begin{bmatrix} \cos(\theta) & 0 & \sin(\theta) \\ 0 & 1 & 0 \\ -\sin(\theta) & 0 & \cos(\theta) \end{bmatrix}\begin{bmatrix} \cos(-\phi) & -\sin(-\phi) & 0 \\ \sin(-\phi) & \cos(-\phi) & 0 \\ 0 & 0 & 1 \end{bmatrix} \tag{5.2}$$

$$R = \begin{bmatrix} \cos^2(\phi)\cos(\theta)-\cos^2(\phi)+1 & \cos(\phi)\sin(\phi)(\cos(\theta)-1) & \cos(\phi)\sin(\theta) \\ \cos(\phi)\sin(\phi)(\cos(\theta)-1) & \cos^2(\phi)+\cos(\theta)\sin^2(\phi) & \sin(\phi)\sin(\theta) \\ -\cos(\phi)\sin(\theta) & -\sin(\phi)\sin(\theta) & \cos(\theta) \end{bmatrix}$$

The base and the tip can now be joined by an arc of angle θ with a radius of S/θ. This can be represented as an array of points using the MATLAB code in Figure 5-2. Firstly the end position calculated using (5.1) is rotated about the Z-axis -φ to place the resulting arc on the XZ-plane, making the maths simpler. Taking the parameters of the section, a polar co-ordinate can be found for the base position and the end position. Interpolating between these points creates an array of polar co-ordinate which can be converted back into Cartesian, translated and rotated back to the original plane. Using these equations and code the graph in Figure 5-3 can be plotted.

```
%If Th = 0, Path is straight line
if Th == 0
    x = zeros(1,n+1);
    y = zeros(1,n+1);
    z = linspace(0,S,n+1);

    Path = [x; y; z];

else %Path is curved

    radius = S/Th;

    %Consider Section is on XZ plane for ease
    dashP = Rot(-Ph,'Z') * endP;

    % Place start and finish in polar coordinates
    [TH1,~] = cart2pol(0 -radius, 0);
    [TH2,~] = cart2pol(dashP(1) -radius , dashP(3));

    % Interpolate between
    theta = linspace(TH1,TH2,n+1);
    rho = ones(1,n+1) * radius;

    % Return to Cartesian
    [x,z] = pol2cart(theta,rho);

    x = x + radius; %Offset X values
    y = zeros(1,n+1); % On XZ Plane so Y = 0
    %z = z;

    % Undo rotation,
    for i = 1:1:n+1
        Path(:,i) = Rot(Ph,'Z') * [x(i);y(i);z(i)];
    end
end
```

Figure 5-2: MATLAB code used to create an Arc

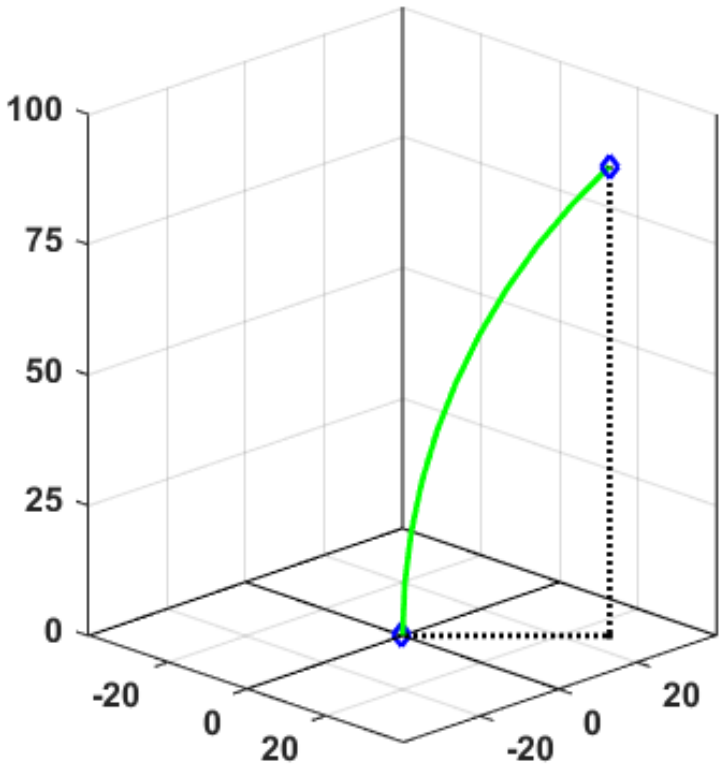


Figure 5-3: Graph of a single section created using MATLAB code
S = 100mm, ϑ = 45° and ϕ = 45°

5.2.2 Recursive Representation of the Multi-Section Snake Arm

Further sections can be added by moving and rotating the coordinates with respect to the previous section. Such that the computation of the next section occurs in the co-ordinate frame of the previous end. Then the result can be converted back to the global frame. Therefore adjusting (5.1) and (5.2) accordingly gives

$$\begin{bmatrix} x_n \\ y_n \\ z_n \end{bmatrix} = R_n \begin{bmatrix} \frac{S_n}{\theta_n}\cos(\phi_n)(1-\cos(\theta_n)) \\ \frac{S_n}{\theta_n}\sin(\phi_n)(1-\cos(\theta_n)) \\ \frac{S_n}{\theta_n}\sin(\theta_n) \end{bmatrix} + \begin{bmatrix} x_{n-1} \\ y_{n-1} \\ z_{n-1} \end{bmatrix} \tag{5.3}$$

and

$$R_n = R_{n-1}\ Rot_z(\phi_{n-1})Rot_y(\theta_{n-1})Rot_z(-\phi_{n-1}) \tag{5.4}$$

respectively, where (x_0,y_0,z_0) is the origin and R_0 is an identity matrix. By writing the equations in the form shown, it allows them to be implemented efficiently as a recursive function. In a similar manner to before, arcs can be calculated from base to tip of the section and then appended to make the total profile array. The resulting graph of a 3-section snake arm can be seen in Figure 5-4.

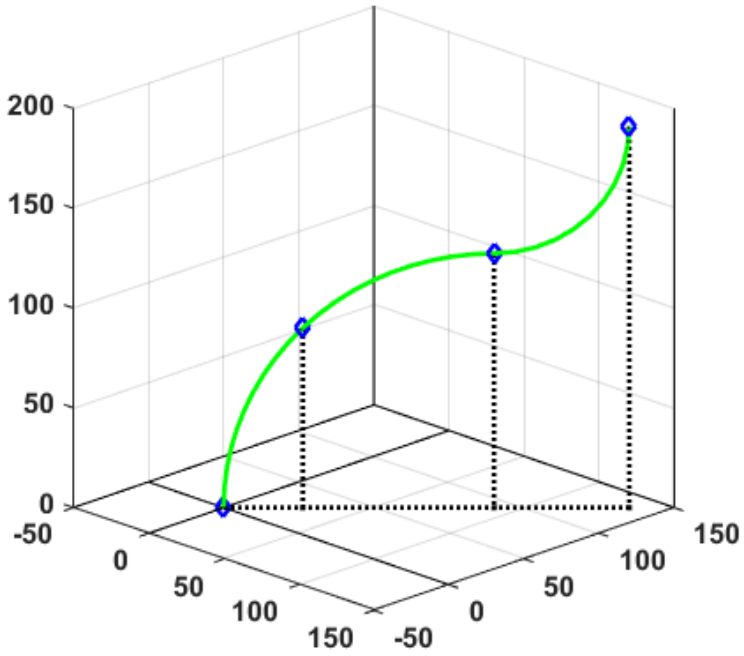


Figure 5-4: Graphical Representation of a Three Section Snake.
S = [100mm 100mm 100mm], ϑ = [45° 45° 90°] and ϕ = [45° 45° 225°]

5.3 General Method and Alternative Approaches

Whatever approach is used, the general method works on a simple principle. Consider the scenario illustrated in Figure 5-5, the snake is currently posed inside a restrictive environment and the user wishes to move forward. This movement can be represented as a discrete step. With a sufficiently small step the control can treat the resulting motion strictly as a position control problem and consider the dynamic response as negligible. At the same time the tip is advanced the base needs to be progressed as well to feed the remaining body of the snake into the environment.

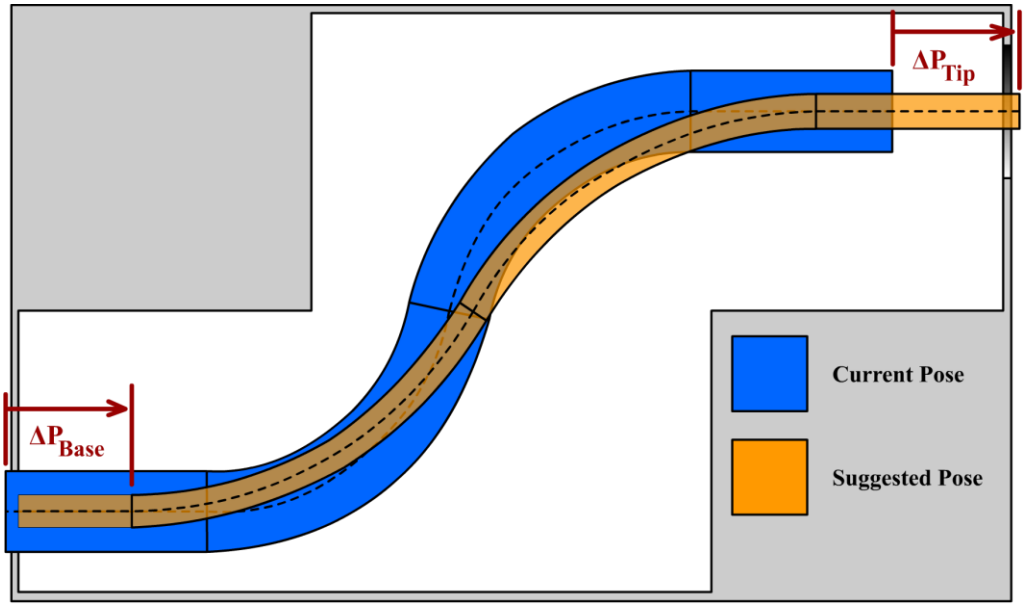


Figure 5-5: Example scenario midway through navigation to illustrate the general requirements

The algorithm can then use one of the approaches discussed below to calculate a new pose of the snake that reaches the new tip position while maintaining the path travelled. In the diagram this path is shown as the current pose, but is more likely to follow a series of steps used to reach the arrangement.

5.3.1 Follow the Leader

This approach follows the method that would be used in computer graphics, and indeed is for games such as the classic "Snake". By recording the positions used by the first section as it is advanced, they can be re-used by each following section as they reach the same point. This creates a fast and simple algorithm with only memory limits being the only computational overhead. This method is limited by the step size being constant once begun and all the sections are required to be same size.

However this approached is flawed, as shown in Figure 5-6, there is a non-linear relationship between the distance advanced at the base and the rate of curvature performed. For example: The snake consists of a section bent at 90° followed by a straight section, the user wishes to advance this arrangement fully until the first section is straight and the second is at 90°. To conserve the tip rotation, the two

sections must be at 45° at the same time and, given that both sections have the same length, this would occur half way through the operation assuming a linear relationship. However this causes a 13.6% overshoot as indicated on the figure.

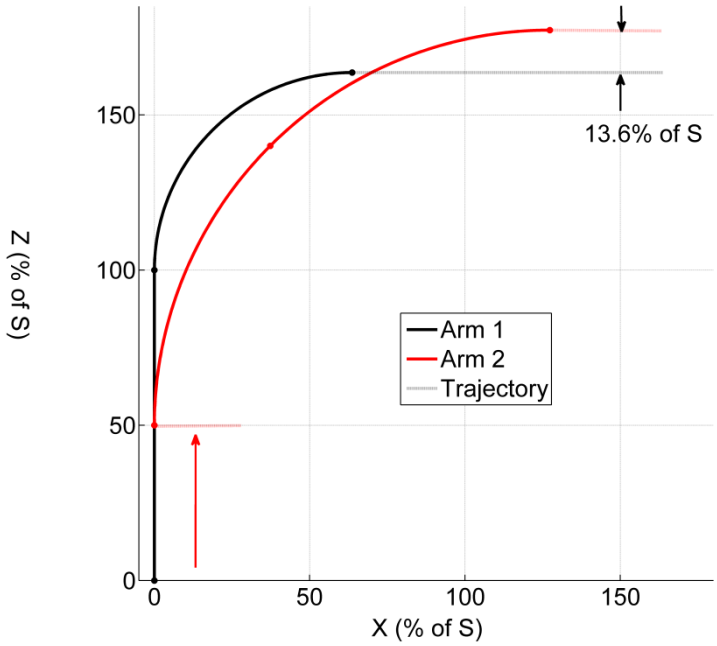


Figure 5-6: Graph Showing the Overshoot Caused by the Non-Linear Relationship

5.3.2 Jacobian

For a more formulised approach, the idea of solving the inverse kinematics was investigated. Given the large number of degrees of freedom many of the traditional approaches are impractical or unfeasible. However a method was found using the Taylor Series.

The Taylor Series is most commonly represented as

$$f(x) = f(a) + \frac{f'(a)}{1!}(x-a) + \frac{f''(a)}{2!}(x-a)^2 + \quad \cdots \quad + \frac{f^{(n)}(a)}{n!}(x-a)^n$$

but by taking

$$x_0 = a \quad \wedge \quad x - x_0 \equiv \Delta x, \quad \ni \quad x \equiv x_0 + \Delta x$$

it is possible to show it as

$$f(x_0 + \Delta x) = f(x_0) + \frac{f'(x_0)}{1!}\Delta x + \frac{f''(x_0)}{2!}\Delta x^2 + \quad \cdots \quad + \frac{f^{(n)}(x_0)}{n!}\Delta x^n$$

and therefore

$$f(x_0 + \Delta x) \approx f(x_0) + f'(x_0)\Delta x \tag{5.5}$$

The equation in (4.5) means that the result of a function for a known set of parameters and a small change is approximately equal to the function performed on the known parameters plus the first derivative of the function at the known parameters times the small change. Taking (4.3) from section 4.2.2 as the function, where the parameters are taken as $x_0 = [\theta_1, \theta_2, \dots, \theta_n, \phi_1, \phi_2, \dots, \phi_n]$, the resulting tip position can be represented as P. As such, for small values of Δx, the following can be true

$$\begin{aligned} P &= f(x_0) \\ P + \Delta P &= f(x_0 + \Delta x) \\ P + \Delta P &\approx f(x_0) + J(x_0)\Delta x \end{aligned} \tag{5.6}$$

$$\therefore \Delta P \approx J(x_0)\Delta x$$

Where J(x) is the first derivative of f(x), or the Jacobian since this is a vector-valued function and would be represented as a matrix of first order partial-derivatives. This means that by taking the inverse of J it would be possible to determine the Δx to reach a new position based on ΔP. By expanding f(x) to include (4.4) it is also possible to maintain the desired angle of approach.

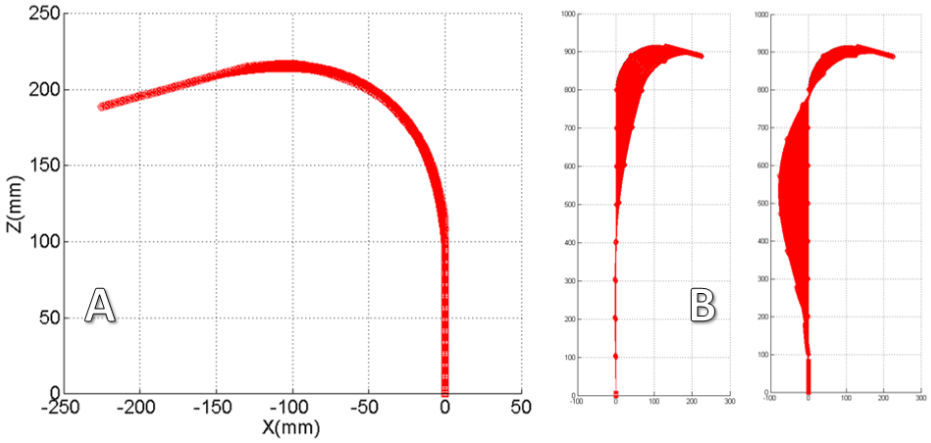


Figure 5-7: Early tests on Jacobian method, initial tests were promising (a) but full length simulations created large quantities of drift (b)

Using a pseudo-inversion, initial tests of this method proved encouraging as shown in Figure 5-7a. However when introduced with more sections the resulting motion tended to drift from the initialised path, Figure 5-7b. This was due to the approach having no constraints for maintaining the path and as such this method was scrapped for final implementation.

5.4 Optimisation Operation

The method chosen in the end is to use an optimisation process to find the best possible solution for the next arrangement. This process uses an algorithm to search for a minimum in the values returned by an objective function given the input criteria. The inputs into the algorithm are the initial values, the constraints, the ideal path and the current pose parameters.

Initial work was tested and developed in MATLAB, using the optimisation tool box. Given the behaviour of the system, only constrained non-linear methods are practical. In the toolbox these are interior point, sequential quadratic programming (SQP), active-set, and trust-region reflective. Reliable results were found with both SQP and active-set approaches, with SQP having slightly faster outcomes.

For the advantages of parallelisation and to keep the code in the same language, the algorithm was finally implemented in LabVIEW. The constrained nonlinear optimisation VI uses the SQP algorithm. The stopping criteria set for the function are: a function tolerance of 0.001, a parameter tolerance of 1E-6, the maximum number of iterations is 150, maximum function calls is 1500, the gradient tolerance is 1E-6 and the maximum time is 5 seconds.

5.4.1 Constraints

Since the optimisation process has no knowledge of the problem other than the response from the objective function, it is important to limit the search space that it can use. This will prevented it searching in orders of magnitude outside the range and eliminate a large number of undesirable local minima. In turn this will greatly decrease the computational time needed to settle on a solution. The search space for the given section in Figure 5-8 is enclosed by the blue circles.

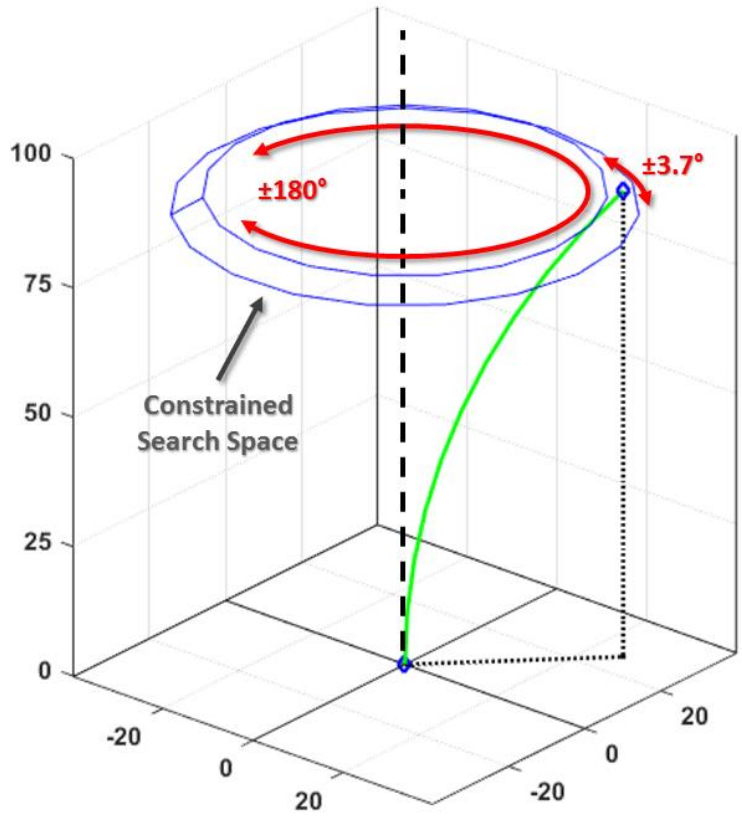


Figure 5-8: Graphical representation of search space available to one section assuming $\vartheta = 45°$

The constraints were chosen using a series of design criteria. Firstly the robot is designed to bend no further than 90° and as such this has to be a hard limit in the positioning. Since the feed-in movement is limited to a maximum of 2mm per step the optimal pose will exist close to the current arrangement. Therefore the bending angle can be limited to a range of ±3.7° since this is the minimum rate of change needed to bend from one extreme to the other in 50 steps. That is, the range from the maximum negative value to the maximum positive divided by the number of 2mm steps to advance a full section length.

Direction is limited to ±180° to prevent the optimisation from searching multiple rotations around the axis. Greater constraint for direction is not used as the algorithm will need to compensate for the direction angles of previous sections and when bending angles are close to 0, direction can flip to find a suitable solution.

5.5 Objective Function

For tip following the objective function needs to take the suggested changes from the optimisation, adjust the current pose to include these changes, compare the new arrangement to the desired path, and return a performance value that represents the quality of the solution. Figure 5-5 remains representative of the scenarios this is considering. The performance value used is a weighted sum of the error at the tip and the maximum error in the path.

5.5.1 Possible Methods for Comparing the Path

There are many possible ways to compare two lines using different methods of representation and metrics employed. However for the optimisation, this process needs to be fast since the optimisation uses from 500 to 2000 comparisons per step.

One aim of the algorithm is to be usable in real-time by the end-user, as such the results need to be fast. Speed of the operation is a combination of the computational complexity and the number of calculations required.

5.5.1.1 Direct Comparison

Since the graphical representation of the snake and the desired path are already arrays of Cartesian co-ordinates, it makes logical sense to compare these data sets. Therefore, assuming the resolution is the same, aligning the indexes at the tip gives each point on the snake its comparative double within the target path, represented by the yellow lines in Figure 5-9. Taking the Euclidean distance for each pair, and finding the maximum distance gives the deviation of the path.

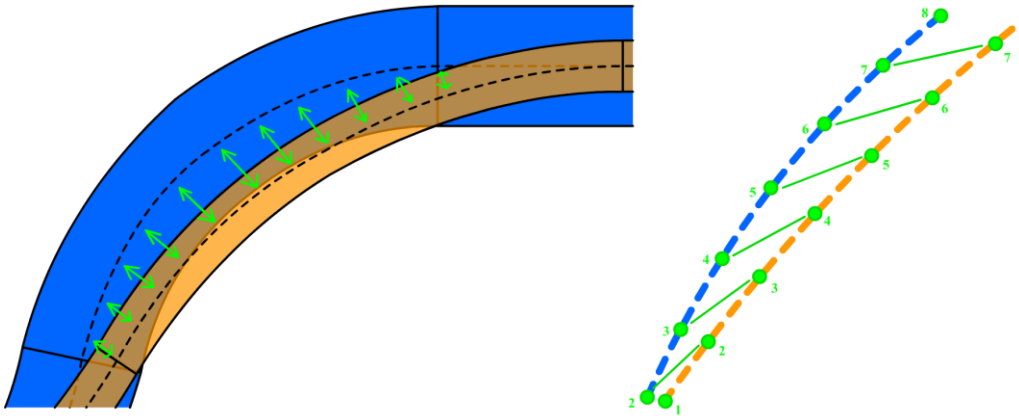


Figure 5-9: Illustration of Direct Comparison method and misalignment causing incorrect performance analysis. Blue is the current pose, orange the new pose.

However this process is inefficient. Consider a 10-section snake with 50 points per section, by using this method of comparison would require 1500 operations per function evaluation. Furthermore this process does not consider the misalignment of the two paths as the snake curves along its length. That is, a comparison will be made with a point further away than the closest point, as shown in the figure.

5.5.1.2 B-Spline

Alternately, because the snake is already parameterised into a series of curves which the MATLAB code in Figure 5-2 showed can be expressed formulaically in polar co-

ordinates. If the target path can also be conveyed in this manner it would potentially be feasible to compare the two equations rather than hundreds of points.

Various methods were assessed, but even the most promising method of representing the sections using B-Splines becomes too intricate as more sections are included. This is due to the way that any line or curve approximation process would not be able to represent the whole path in one equation, therefore needing several delimiters to separate it from other parts of the outline. As the number of sections increase, the number of delimiters increases exponentially. Thus, it still remains simpler to represent the profile of the path and the Snake Arm in Cartesian co-ordinates.

5.5.1.3 Many-to-One

To tackle the inefficiencies of the direct comparison method it became apparent it was not necessary to compare every point. Instead, compare a low resolution set with a more densely populated set. Since the target path is already made from Cartesian points spaced 2mm or less apart and the suggested snake arm is the variable each run it makes the most sense to use the pose for the sparser array. Through experimentation it was found that a snake pose with 5 points per section was acceptable.

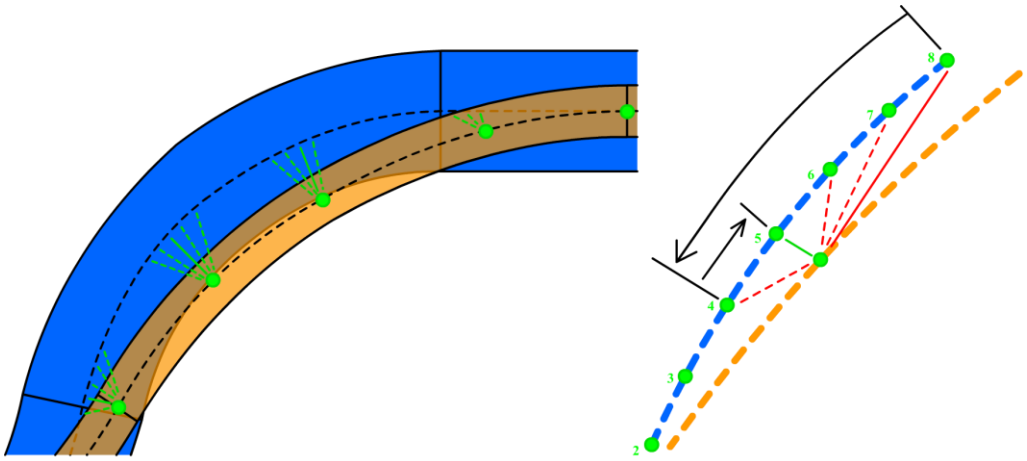


Figure 5-10: Diagram to help explain the Many-to-One approach. Blue is the current pose, orange the new pose.

To align the two sets, two techniques are used. Firstly, the tip is indexed against the latest value in the target path and the path index that corresponds to the remaining snake points is calculated by the axial length from the tip. Next, to alleviate the staggered errors found previously, each comparison uses a gradient search to find its best match. On average this technique uses 250 operations per function evaluation, which is 83% more efficient than the direct comparison method.

5.5.2 Implementing Many-to-One

The first step in a real scenario is to determine the number of sections currently involved and the total length. For this example 10 sections are used, each 100mm, for simplicity. As with Figure 5-5, the snake is part-way through a tip following operation as this makes it easier to explain the overall process. In this instance it is also assumed the path consists of 2mm spaced points, as such the snake will fit in 500 points. Taking the final point of the path and working backwards, each point on the snake profile can be assigned a reference to index on the Path. Therefore the tip will reference the final point, or programmatically $index_{tip} = length(Path)$. Following section ends can be referenced a section length away, that is $index_{tip-1} = length(Path) - 1*(100/2)$ and $index_{tip-2} = length(Path) -2*(100/2)$, etc. In a similar manner the intermediate points can also be referenced as described in the pseudo code.

Once the references are found, each point is taken in turn to perform a gradient-search to find its best partner. This eliminates the staggering problem the previous direct comparison technique suffered from. The gradient-search checks the indexes either side of the reference for a better result, if one is found the process is repeated with this as the reference. If both indexes either side of the reference are worse, then it is assumed the best pair is found.

The result of this part of the process is an array of the minimum distances between the two point arrays at multiple locations along the length. To represent this as a single scalar performance value all that is required is the largest distance at any given point; hence the maximum of the array of minimums.

5.6 Streamlining the Algorithm

As with any development process, a number of redundant or inefficient steps may have been taken or assumptions that are only valid for set case. The following improvements show how the algorithm can be polished to save computation time, decrease deviations and iron out potential issues.

5.6.1 Feed-In Approximation

Initially the process was carried out with the full snake arm represented at all times however most these sections are outside the environment, not required and completely superfluous to the task. This means the optimisation is performing excessive operations trying to adjust parameters which can be fixed at zero until needed. By excluding these degrees of freedom it is possible to approximate the feed-in mechanism and speed up the algorithm exponentially for the early calculations.

By taking this approach it also removes the assumption that the feed-in system can only work linearly and in-line with the entry point. This is advantageous, as Chapter 6 will show, since the final design for the feed-in mechanism uses a rotating drum to coil the Snake Arm.

5.6.2 Dealing with variable feed-rates

Originally the process dealt with fixed steps at the tip of 2mm, which are duplicated at the base. However in a real scenario it is extremely unlikely that these 2mm

movements can be guaranteed from the path planning, and there are times were smaller movements would be preferable.

To achieve this, the Euclidean spacing of each step is taken and recorded as the Arm is progressed. This spacing is used to set the advancement of the base for its respective step and to provide the alignment information for the Many-to-One comparison.

5.6.3 Performance Weighting

Given that the objective function is judged on two factors, the path deviation and the tip error, it is important that the values are scaled to reflect their significance. For starters, the ideal tip error is less than 2.5mm while path deviations up to 20-30mm are still acceptable. Through trial and error, it was found that a weighting of 10 on the tip error allows for a high accuracy in the tip, without overly affecting the quality of the path.

Another way the performance value is weighted is by leaving the Euclidean distance squared. This creates a rapidly increasing error the further from the path the pose is. As such, it means the chances of solutions being found within a useable radius are greatly increased.

5.7 Simulation Tests and Results

The performance criteria of the tip following are the error in the tip position, the deviation from the desired path and the computational time. Each criterion is split into the maximum value and the average value. Since the criteria are very subjective to the number of sections involved and the complexity of the path, graphs are provided to give a clearer picture of how the values vary over the test.

Tests are performed in LabView and analysed afterwards in MATLAB. This prevents any of the post-processes from skewing the results, namely the computation times,

and gives better tools for viewing and studying the results. The computer used has an Intel® Core™ i7 CPU 960 running at 3.20Ghz and 12Gb of DDR3 Tri-Channel RAM.

Initial tests use a path generated by the forward kinematics, in each case the path is given in the snake parameter form. All tests show the target path superimposed from the camera prospective, since the actual path may be obscured by the results. All the simulations use a section length of 100mm.

5.7.1 Generic Profiles Tests of Tip Following

The first simulations feature few sections to test the algorithm works and represent the shorter tasks that do not require the complete length of the snake arm. Initially the tip following is simulated on clear paths, 90° bends or S-Bend profiles, before more complex trajectories. These scenarios would be for more generic usage, mainly for inspection and would be more commonly of interest to an on-line user; hence the computation time is more important than the accuracy.

Table 5-1: Results Summary of Short Tests.

	Profile	Total Time	Max Time	Avg Time	Max Dev	Avg Dev	Max Tip	Avg Tip
S-Bend Profiles	[0, π/2, -π/2, 0]	10.29	0.260	0.051	19.50	7.170	1.29	0.475
	[0, π/4, -π/4, 0]	11.09	0.324	0.055	5.577	1.935	0.66	0.162
	[0, π/6, -π/6, 0]	10.75	0.352	0.053	3.674	1.225	0.54	0.129
90° Pr	[0, 0, 0, π/2, 0]	23.52	0.438	0.093	16.59	2.298	1.10	0.151

	[0, 0, π/4, π/4, 0]	22.27	0.429	0.088	1.952	0.651	0.26	0.044
	[0, π/6, π/6, π/6, 0]	18.92	0.554	0.075	1.517	0.699	0.17	0.037

Time measurements in seconds, Deviations and Tip errors are in millimetres

5.7.1.1 90° Bends

With the design of the arm it is possible to make a 90° bend in a single section, but the snake need not bend this sharply to achieve the shape. However, with this in mind, it is important to look at how the curvature of the bend affects the results of the tip following and prove whether the extreme case is usable if required.

The three tests that follow each use a 5-section path to keep the number of steps and number of sections involved the same. Note that the number of sections is not limited to five, this is the number of section used in the path assignment. Table 5-1 summarises the results and Figure 5-11 shows the consequences graphically.

Not unexpectedly the sharper bend creates more deviation, but not outside a useable range, meanwhile the other bends give tight and consistent movements in relation to their respective profiles. There is a noticeable difference in the computational time due to the differences in complexity, however it is surprising that the profile with the longest initial straight still has the longest completion time since these straight sections tend to have a negligible calculation time. As such, this indicates how much the algorithm finds shallow curves easier to evaluate.

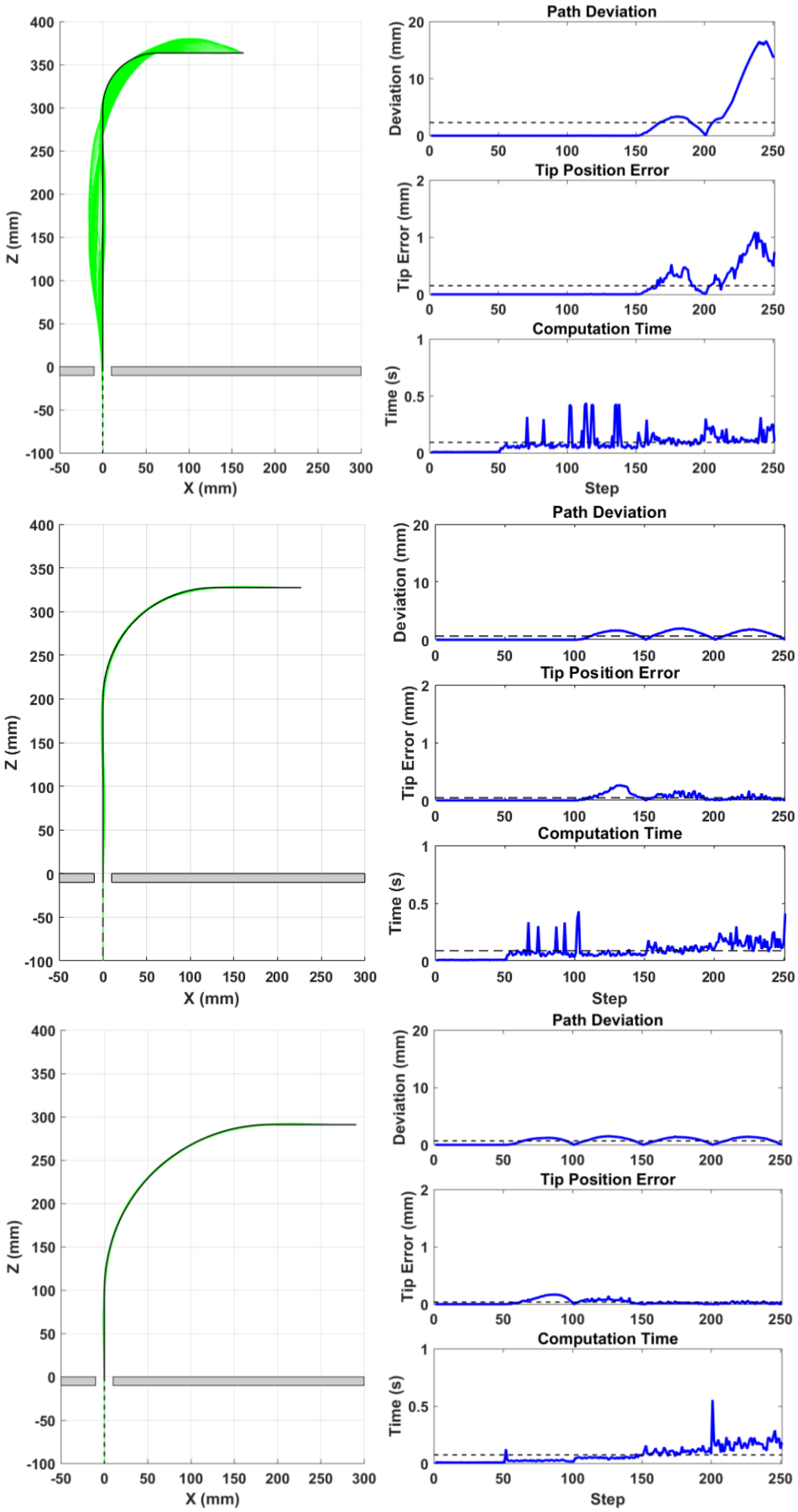


Figure 5-11: 90 degree bend profiles, in black, of different curvatures. Overlaid is every pose used in the Tip Following process to display the deviations.

5.7.1.2 S-Bends

Another common profile is an S-Bend. This is when the Snake performs a bend in one direction followed by its mirror. In turn this creates a path where the trajectory is still in the same direction after the manoeuvre but shifted in the horizontal plane. For human comprehension these are easy to understand and create, which explains their frequent use in user supplied paths.

As with the 90° bends, the quality of the Tip Following is heavily dependent on the severity of the action. The tests exhibited in Figure 5-12 use different bending angles to substantiate the abilities of the algorithm again and prove the most extreme case is still within an operative range of 20mm.

Naturally, the deviations of the tighter S-Bend are significantly greater than the others, again this is because the algorithm is working at the level of the constraints to achieve this result. However, the action required in the profiles is very similar. The lengths before, during, and after are equivalent which is why the computational times are almost identical.

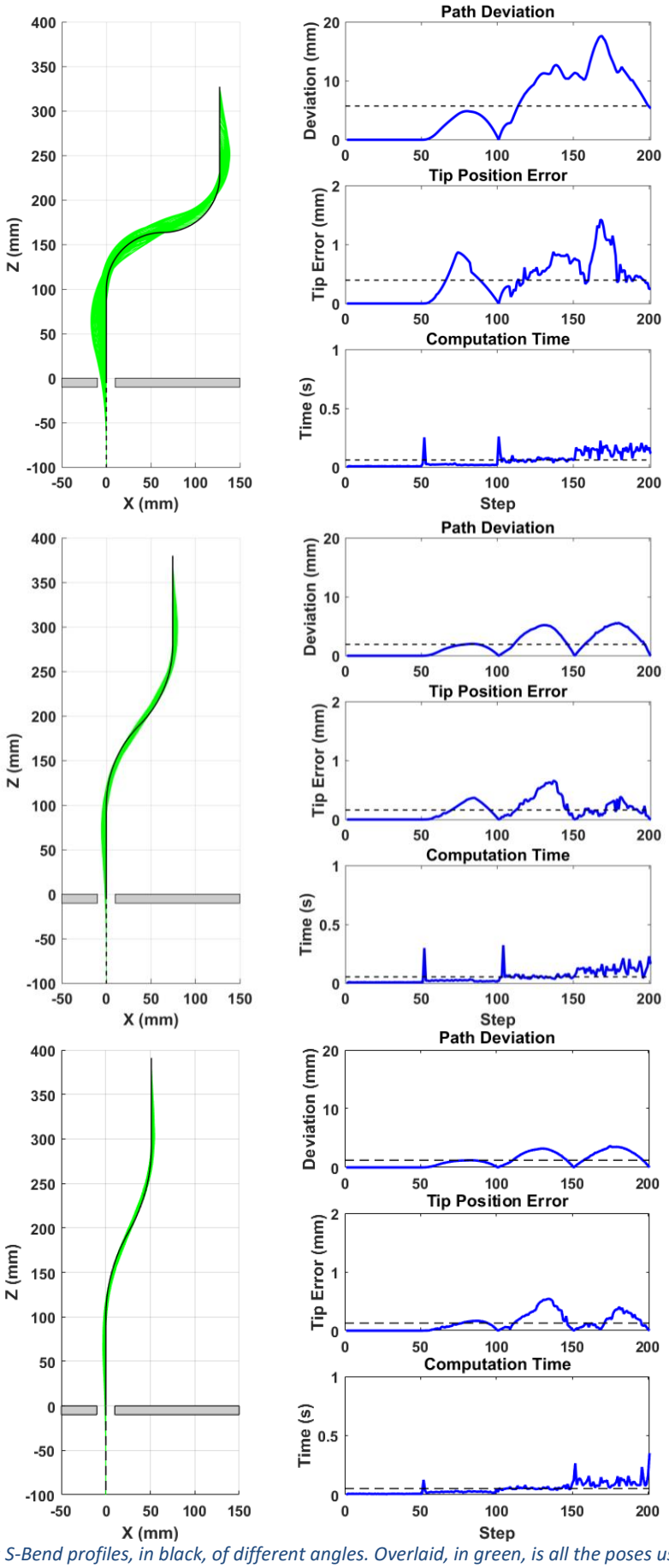


Figure 5-12: S-Bend profiles, in black, of different angles. Overlaid, in green, is all the poses used in the Tip Following process.

5.7.2 Full-Length Trials

These tests use the majority of the length of the Snake Arm and they would most commonly represent scenarios where the snake has to operate in a restrictive environment and has to weave around obstacles. As such, the deviation is of great interest with these assessments and how different paths can alleviate the errors generated.

5.7.2.1 Generic Scenario Path Performance

While other simulation tests just express the success of the algorithm to follow a path, this experiment aims to prove that this work can be used in a realistic scenario. Figure 5-13 presents a path, with set 45° curves for all the routing actions, through a generic environment. A smoother path could be found that would improve the results, but it is more likely that an operator would chooses a profile like the one utilised as it is intuitive and immediately comprehensible.

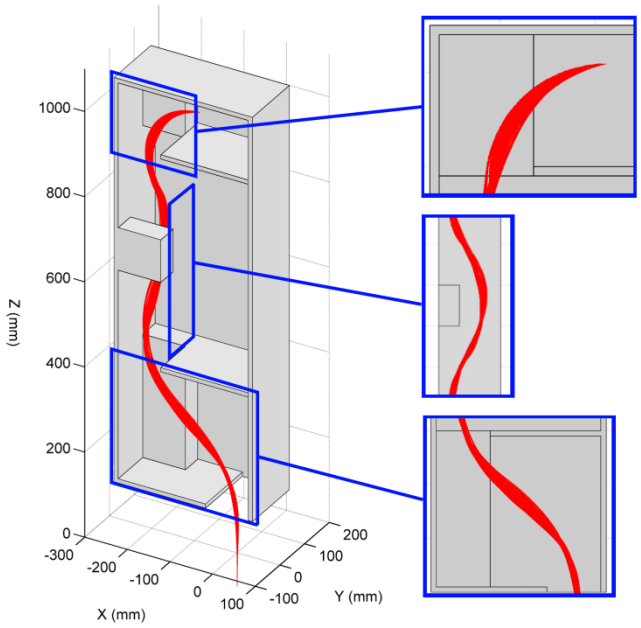


Figure 5-13: Overlay of all poses used for Tip Following within a generic scenario.

The navigation task was completed in 3mins 56 seconds, with an average computational time of 0.391s and maximum of 2.301s. This means for an on-the-fly run the equivalent robot could be run at 5 mm/s to accomplish the task without any interruptions. The tip position was maintained at an average of 0.177 mm and a maximum of 0.90 mm. As for the path deviations, the average is 6.340 mm and the maximum 20.15 mm. The peak of the drift is caused by the last few bends, as it can be interpreted from the graph in Figure 5-13. This is because the profile combines a 45° S-Bend followed immediately by a 90° bend of curvature equal to Figure 5-11B. In a more pragmatic path this combination could have been rectified by merging the two actions into a naturalistic motion.

5.7.2.2 10-Section Random Paths

A series of simulations were performed using randomly generated path profiles to demonstrate the flexibility of the method as a general purpose solution for multiple scenarios. In this way it proves that the system is not limited to pre-selected paths and gives a wider view of the range of results. Each path profile is calculated as a 10-section Snake Arm starting at the origin.

Figure 5-14 shows the results from a sample set of 25 runs, which is also summarised in Table 5-2. The three performance criteria for each step are displayed with the global average, average maximum and average minimum. The red lines represent runs where the respective criteria exceeded a threshold of 30mm in path deviation or 3.5mm in tip error. No threshold is set for the computational time as the results are notably uniform.

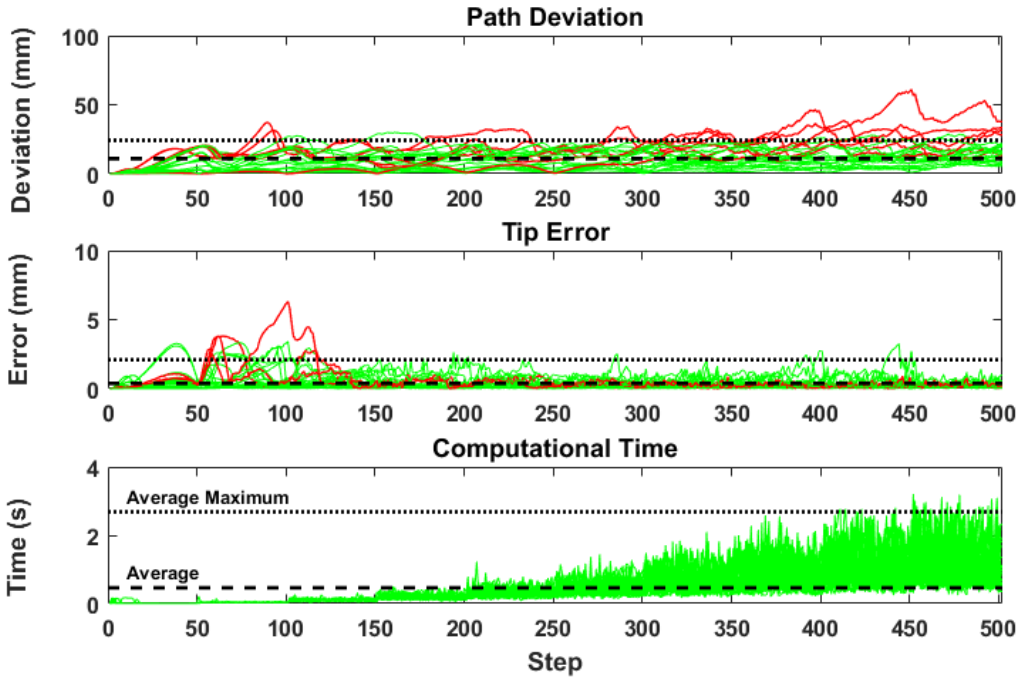


Figure 5-14: Performance data for 25 random runs of the proposed method

By analysing these outliers the profile features that cause the significant errors can be determined and provided as guidance for further operations. This is useful as at times as the features may be required in the path being planned. By knowing the error generation it is possible to say that they are either in an acceptable location or the profile could be adjusted in a way to move the resulting path in to an acceptable location.

Table 5-2: Result Data from Random 10-Section Paths

Run	Path Deviation		Tip Error		Computation Time	
	Max	Average	Max	Average	Max	Average
1	19.17	5.41	0.87	0.15	2.48	0.46
2	24.47	12.48	3.44	0.67	2.50	0.43
3	9.63	4.40	0.43	0.11	2.62	0.51
4	13.53	2.52	1.40	0.11	2.72	0.50
5	28.79	16.25	3.80	0.47	2.45	0.35
6	13.25	8.90	1.34	0.41	2.27	0.41
7	32.43	15.04	3.33	0.47	2.55	0.32
8	30.58	15.32	2.59	0.53	2.38	0.35
9	9.34	3.40	0.63	0.15	2.56	0.48
10	22.26	7.76	1.18	0.25	2.39	0.40
11	60.95	26.61	3.27	1.00	2.37	0.37
12	7.82	3.17	0.61	0.14	2.31	0.47
13	13.73	7.52	1.36	0.24	2.79	0.48
14	27.88	11.19	1.78	0.47	2.40	0.41
15	11.13	3.44	0.37	0.10	2.77	0.53
16	36.41	20.72	3.08	0.62	2.74	0.45
17	29.88	10.20	2.28	0.49	3.19	0.53
18	26.23	11.58	0.91	0.26	2.70	0.50
19	31.42	14.26	2.42	0.41	2.74	0.47
20	21.09	11.29	2.50	0.28	2.97	0.49
21	27.40	13.59	6.30	0.67	3.07	0.56
22	38.46	10.99	1.06	0.19	2.97	0.52
23	21.87	9.31	1.35	0.29	3.21	0.51
24	25.82	16.64	3.79	0.50	2.87	0.53
25	22.62	12.17	2.30	0.62	3.09	0.49
Average	**24.25**	**10.97**	**2.10**	**0.38**	**2.68**	**0.46**
Max	60.95	26.61	6.30	1.00	3.21	0.56
Min	7.82	2.52	0.37	0.10	2.27	0.32

5.7.2.2.1 Path Deviation Outliers

The outliers for the Path Deviation are all marred by similar problems. They have been caused generally by sharp changes in direction or a series of such occurrences. This is quite prominent in cases when the switchback changes the plane dramatically.

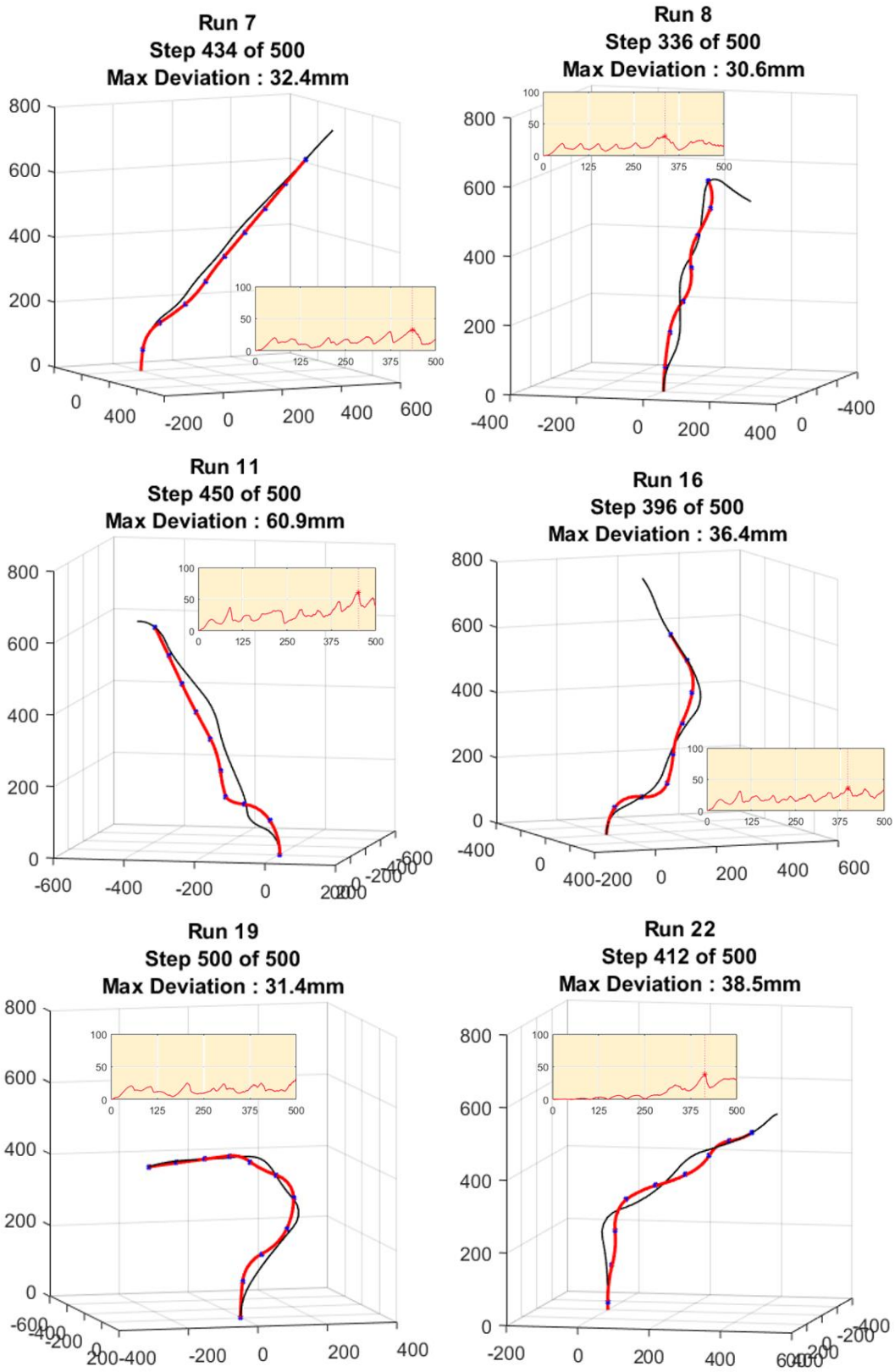


Figure 5-15: Path Deviation Outliers at their most deviated step. Subplot shows path deviation vs. step with the current step indicated

Runs 7 and 11, as shown in Figure 5-15, demonstrate the precarious problem with this type of error. Due to the way the objective function is calculated, having a feature that causes substantial drift from the desired path creates a large region of “acceptable error” as the smooth path needs not be any better than the path through the bends. This is why a large percentage of the snake body is distant from the path it should be following. In the case of Run 11, the issue is exasperated by the profile feature being located immediately at the start of the path.

5.7.2.2.2 Tip Error Outliers

In each of the cases, Figure 5-16, it is clear that the error is caused by a considerable bend at the beginning of each run. At this point there is only one or two sections available to the algorithm making it harder for the system to find a better solution. This hypothesis is further backed up since all the tip errors in Figure 5-14 settle below 2mm after 150 steps when 4 or more sections are involved.

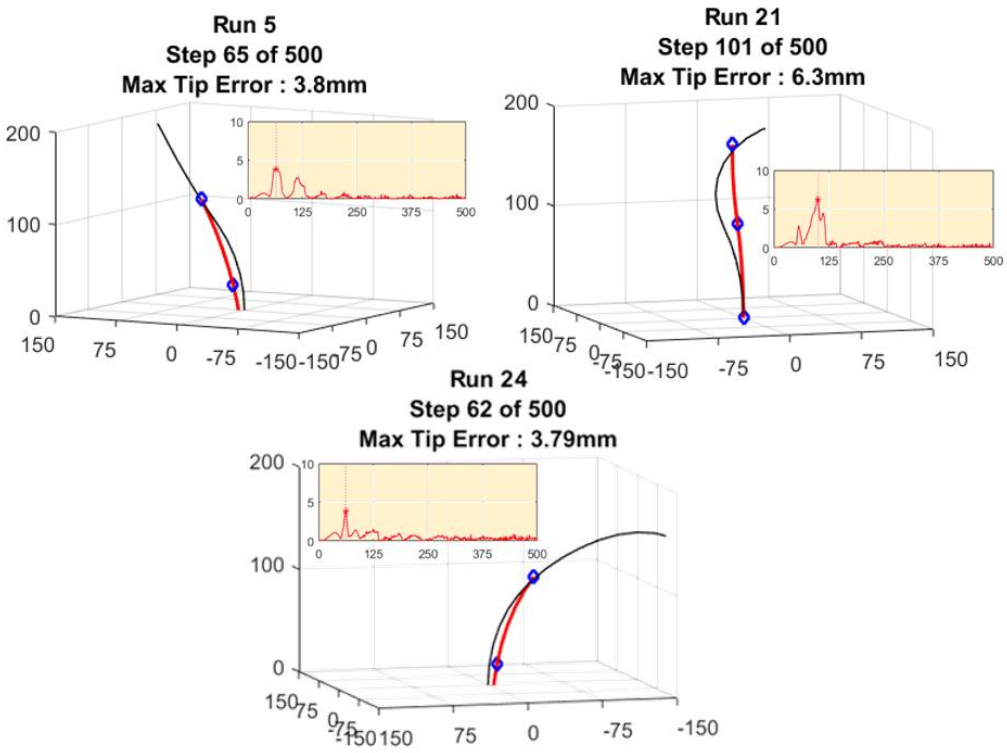


Figure 5-16: Each tip error outlier at the step of greatest error. Subplot shows tip error vs. step with the current step indicated

5.8 Physical Trials with Second Prototype

While this algorithm can be proved by simulation alone, it is preferable to see these actions demonstrated on a physical prototype. For this goal, the tip following was processed on the second prototype with pre-calculated data. It is also possible run the system on-the-fly by buffering the calculated steps as they are computed, and passed to the Snake Arm at a fixed rate.

The demonstration has robot follow an extended bump profile, limited to one plane for ease of observation. Figure 5-17 shows a time series of this run with each step depicted with the graph of the commanded pose. Due to the nature of the design, the snake is orientated inverted to reduce the effects of gravity on the configuration. The maximum deviation of the simulation was 14.71mm and occurred on step 405. Figure 5-18 shows this step and provides a comparison between the simulated and physical performance.

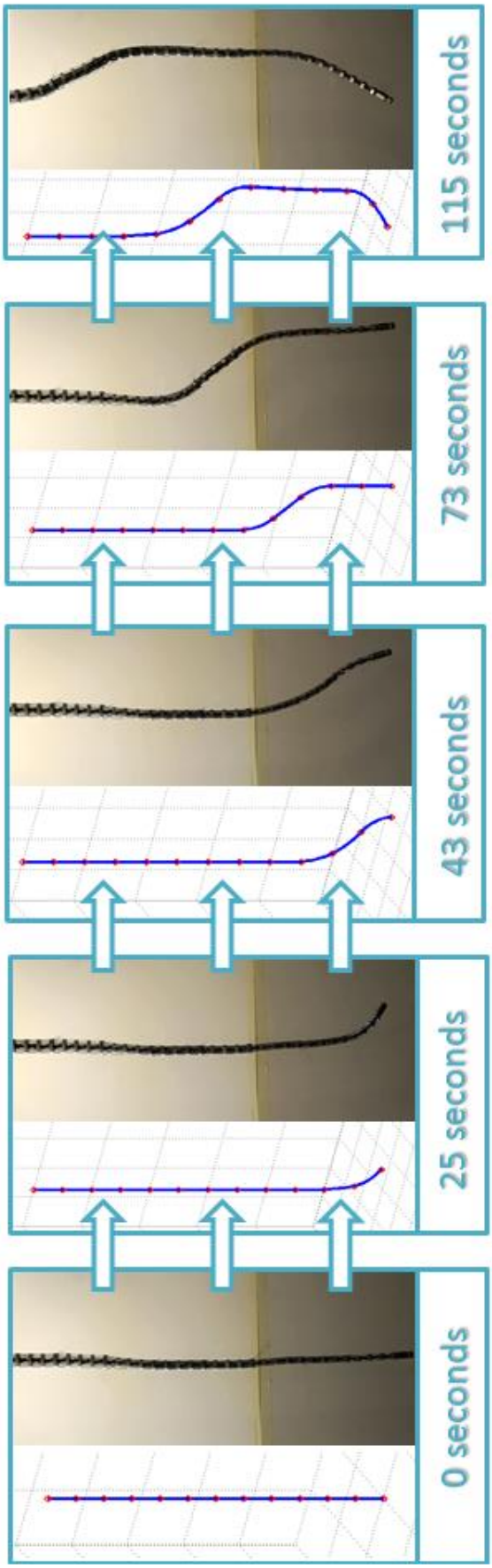


Figure 5-17: Time Series of Poses during Tip Following Trial

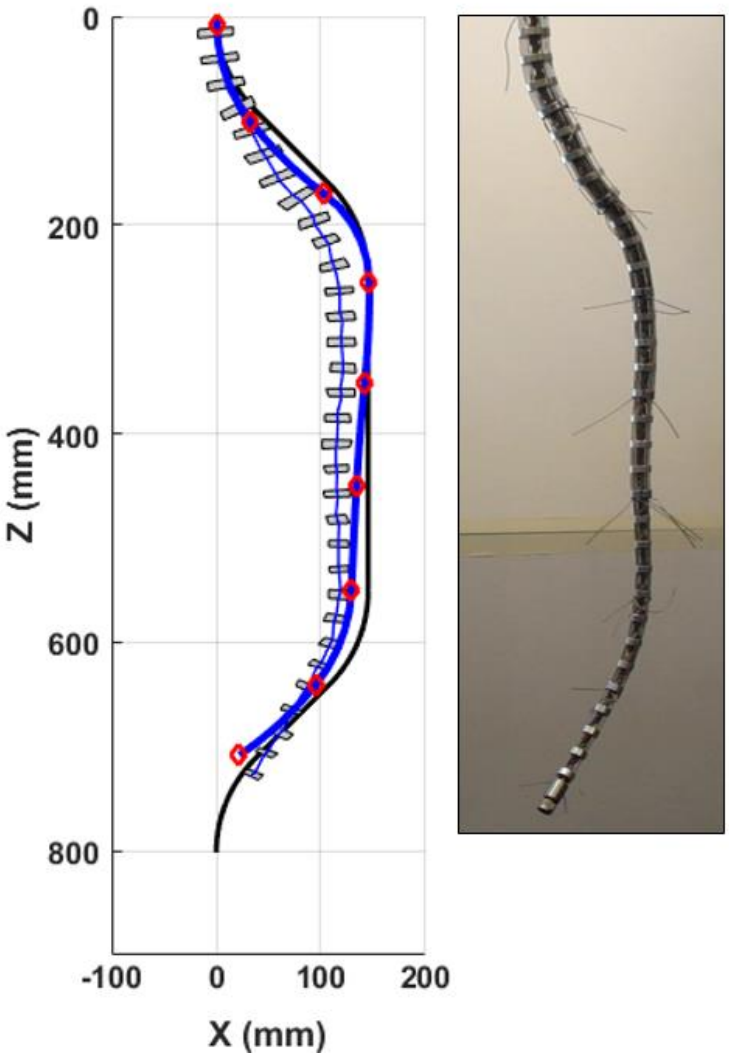


Figure 5-18: Extended bump path test at the step with the maximum deviation. Graphical comparison of tip following path (blue) and the physical robot against the trajectory in black.

It is apparent in the graph that the prototype does not achieve the desired bending angles, this early system suffered with a lower than required stiffness and therefore struggles against gravity. However it is clear that the shape of the Snake does clearly correlate with the intended positioning. Chapter 7 will include better examples on the final hardware.

5.9 Chapter Summary

This chapter has presented how the Snake Arm can be represented both visually and in a manner that can be interpreted by the computer. By using a set of geometric kinematics, the end point of the section can be found. From this, a recursive algorithm was created to determine the points of the complete structure. The body of the continuum joint is then simulated as an arc using the assumption the system creates smooth, uniform curves.

Using this representation, or a similarly created one for a different Snake Arm design and behaviour, an algorithm was proposed to solve the longitudinal navigation. Several methods were discussed that could be implemented to create the algorithm before settling on an optimisation routine. For this an objective function was created using a Many-to-one method for judging the performance of the configuration after other techniques either shown not to be accurate or too computationally intensive. This approach was then streamlined by approximating the feed-in movement and to deal with variable step sizes.

The final algorithm has then be tested and validated with a series of simulations ranging from simple profiles to random complex shapes. Not only did this determine the features which cause the most difficulty for the process. But also showed the performance of the code to keep the deviations and tip errors below 20mm and 1mm at a rate of 0.50s per step on average.

6 Feed-In

6.1 Introduction

As suggested in the previous chapter the Tip Following requires a method to advance the base position of the Snake Arm towards the environment. This movement has been designed to be operated linearly through an arbitrary handover point. This means the feed-in must provide the requested advancement to a high-level of accuracy, while maintaining the trajectory of the movement for the continuum robot.

Other hyper-redundant manipulators surveyed use linear tracks or industrial robot arms to perform these advancement trajectories. With these methods there needs to be enough space outside the environment, at least equal to the length of the path to be explored.

The less conventional solution proposed in this thesis is to coil the Snake Arm around the actuation pack. Providing the feed-in action by rotating the drum and uncoiling. In turn this makes the stored footprint of the robot smaller and enables operation while mounted on the walking hexapod.

6.2 Specification

To support the Snake Arm during transport and storage the actuation pack is designed with a helical channel which the body can reside within, as illustrated by Figure 6-1. This recess has a diameter of 275mm with a pitch of 42mm, which is based on the shape of the Snake Arm and the space required for the hardware. Consequentially, this means there is 1.452 revolutions to coil the complete length of the robot at 1256mm. This can be represented using

$$\begin{bmatrix} x_n \\ y_n \\ z_n \end{bmatrix} = \begin{matrix} r_{helix}\cos(2\pi n) \\ r_{helix}\sin(2\pi n) \\ n \times \lambda_{helix} \end{matrix} \tag{6.1}$$

where *n* is the rotation from the start position in revolutions, r is the radius and λ is the pitch. By calculating (6.1) from 0 … 1.452 an array of Cartesian points can be determined which are used to represent the helix graphically.

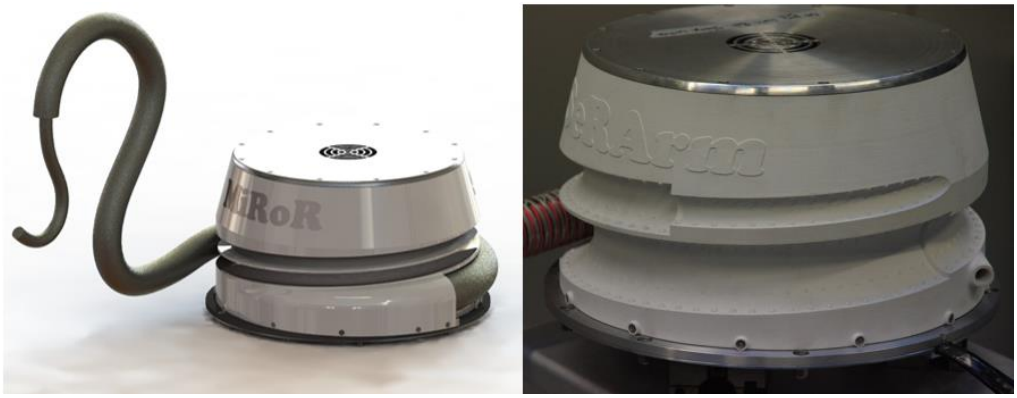


Figure 6-1: The actuation pack for the Final Prototype, Concept (Left) Actual (Right)

6.2.1 Coiled Profile

Once provided with the shape of the helix, it is possible to find a configuration for the Snake Arm that matches the channel. In the design, there is a maximum of 2mm clearance for this recess, so any path deviation must be below this value. Initial evaluation used an approach based on the forward kinematics to find a suitable pose.

For a homogenous Snake Arm, that is one with same size section lengths, a helical pose can be created by having the same bending angle for each section while the direction angle follows the rule

$$\phi = \left[a + a_0 \ , \ 2a + a_0 \ , \cdots , \ na + a_0\right] \tag{6.2}$$

where a_0 is the starting direction and a is the pitch angle. To align with the current helix a_0 = 180° and a = 1.92°, for a Snake Arm with twelve sections of S = 100mm and θ is 41.67°.

However, for the final demonstrator as described in Chapter 3.5.4, the Snake Arm does not have the same size section lengths. They decrease from 144mm at the base to 50mm at the tip. In turn this means the bending angles vary based on

$$\theta = S/r_{helix} \tag{6.3}$$

where S is the array of lengths to return the array of θ in the same order. Using (6.2) for the direction angles as before. Figure 6-2 shows the specified helix, from (6.1), and the two attempts at profiling the path using the forward kinematics.

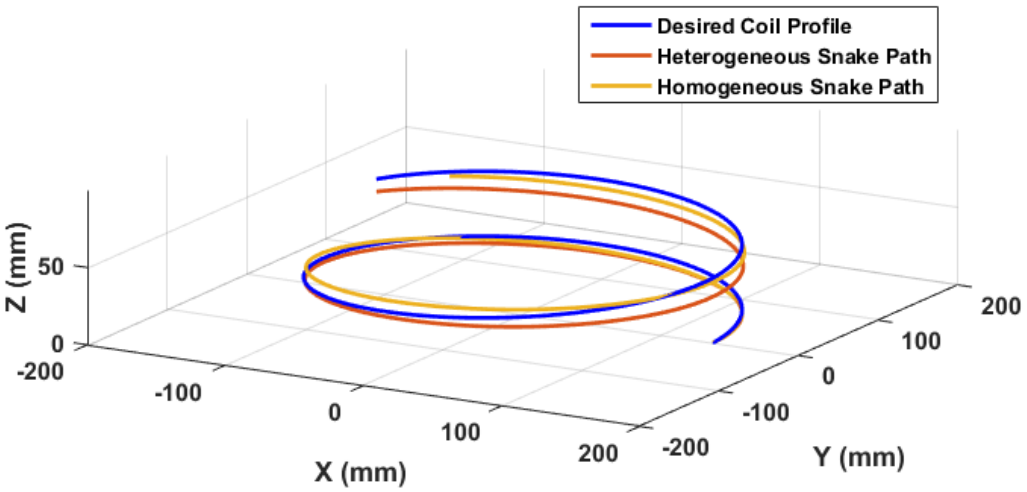


Figure 6-2: Comparison of Forward Kinematic attempts to match coil profile

While these configurations are representative of a helical shape, they do not equal the proposed helix and are definitely not within the 2mm clearance. The problem is that the designed helix assumes instantaneous curvature at the base. That is, the path is straight from the base to the final height, like the hypotenuse of a triangle, skewing the cross-sectional profile. Whereas the continuum joints have to perform a uniform curve along each section length.

6.2.2 Optimising the Profile

Rather than attempt to solve this problem with a convoluted implementation of the inverse kinematics, it was decided to attempt to find the configuration by re-using the optimisation from the Tip Following, see Chapter 5.

Using (6.1) the helical path is known and can be set as the target. Given base position is set at [137.5, 0, 0]', such that the origin is the central axis of the helix, it is possible to specify the tip position to be [135.76, 21.80, 60.98]'. The only changes required to the already developed methodology was to remove the expected feed-in from the algorithm. To initialise, the heterogeneous Snake Arm was applied. The resulting coiled profile has a maximum path deviation of 1.72mm with a tip error of 0.05mm, this is represented in Figure 6-3.

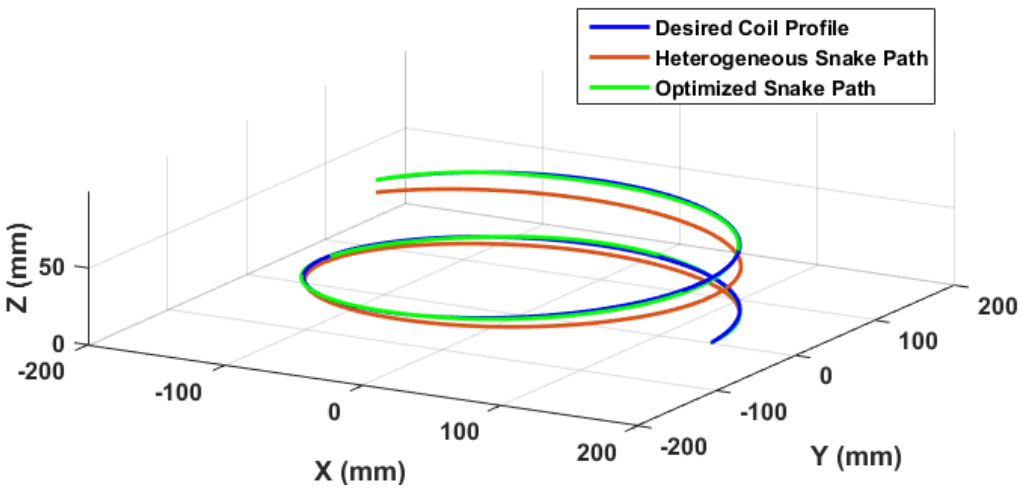


Figure 6-3: Optimised Pose to match the profile of the helical channel

6.3 Proposed Uncoiling Methods

From the coiled configuration there are several potential mechanisms and trajectories to uncoil and feed the robot. For the purpose of this project there was enough time to implement one fully, however the techniques used in Section 6.4 are generic and can be applied to develop the other methods below and further derivations.

6.3.1 Fixed Drum

Before deciding on the coiled position one suggestion from the concept art, Figure 1-1, was to use the excess body of the Snake Arm to control the feed-in and in turn manage the placement of unused sections. This method, as depicted in Figure 6-4, operates in an analogous to the way a cobra attacks when threatened. However, this concept does not provide anywhere to position the robot when the power has been turned off. Nevertheless, similar techniques could be implemented to uncoil the body from the drum.

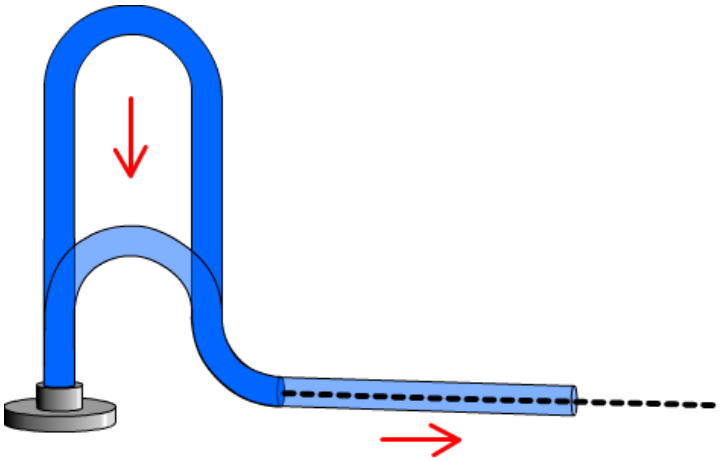


Figure 6-4: Non-Coiled Method for Feed-In using Spare Length of Arm

There are two options when considering a vertical trajectory, either co-linear with the central axis of the drum or offset by the radius. For a horizontal path, the Snake Arm needs to unhook each loop and straighten the body as it passes over the drum. Other possibilities can be found, but tend to be variations or extensions of these three with superfluous actions.

As Figure 6-5 details, with all the routes, the robot is required to untwist as it travels. This leads to a rotation about the central axis of the tip and tighter bending than ideal. While both the Tip Following and the mechanical system could handle this, it would be ill-advised to pursue this approach while alternatives are feasible.

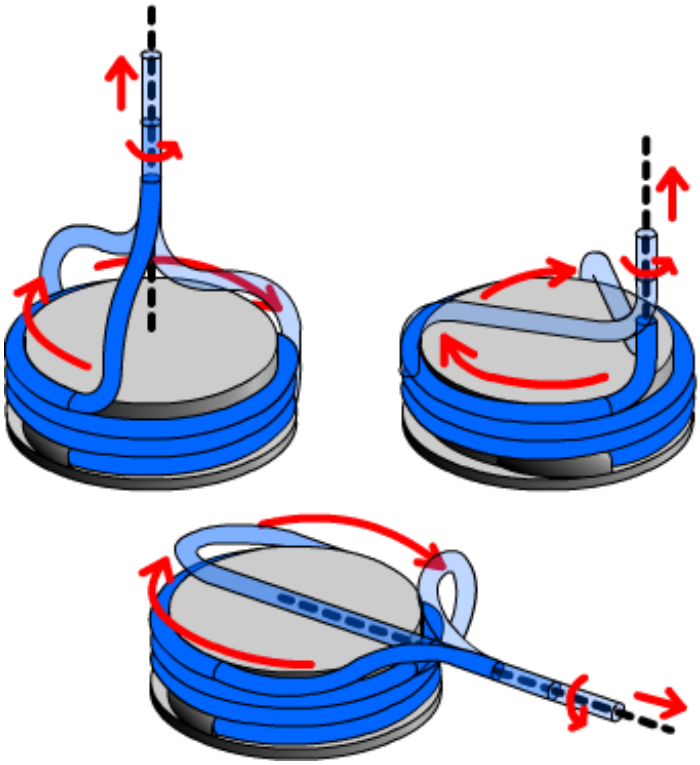


Figure 6-5: Passive Uncoiling Methods

6.3.2 Active Drum

By actuating the drum to rotate, the twisting motion can be removed while the Snake Arm is uncoiled. This leads to more potential uncoiling strategies. By changing the starting rotation, it is also possible to feed in any direction without altering the algorithm.

Vertically, the system can uncoil on to a trajectory that intersects with the virtual square corner of drum as seen in Figure 6-6a. This leads to a bend that descends as the body is fed off. Similarly, this can be repeated for any angle above the pitch with the same effect and with careful planning the trajectory can be maintained. When the angle is equal to the pitch, Figure 6-6c, this method is no longer feasible as the trajectory does not intersect at the virtual corner.

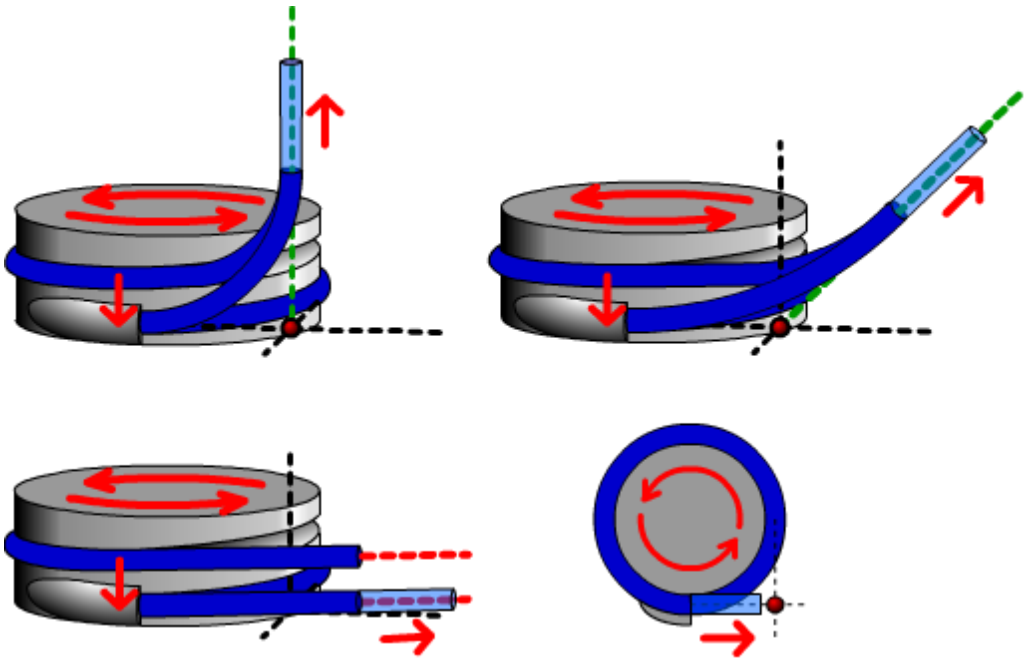


Figure 6-6: Uncoiling a) Vertically B) 45degrees C) At Pitch angle

However the focal scenario, a gas turbine engine, intends the Snake Arm to be mounted at the front fan and precede inwards. Which means ideally the trajectory should be horizontal. Since this angle is less than the pitch, it cannot be solved with a single bend since the trajectory descends faster than the helix. Instead an S-Bend is required. Equally, this would be the requirement for any angle less than the pitch.

A horizontal movement can be produced at any height of the helix. Logically, however, it should occur where a clear reference plane can be made. This suggests the action should be at the top or the bottom, but the mid-point can also be considered. When starting at the top the Snake Arm begins straight and an upwards S-Bend is formed, getting tighter, as the process runs, Figure 6-7a. While starting at the base creates the opposite effect, a tight downwards S-Bend leads into a straight pose, Figure 6-7b. The mid-point has both these actions reflected about the plane at its height, Figure 6-7c.

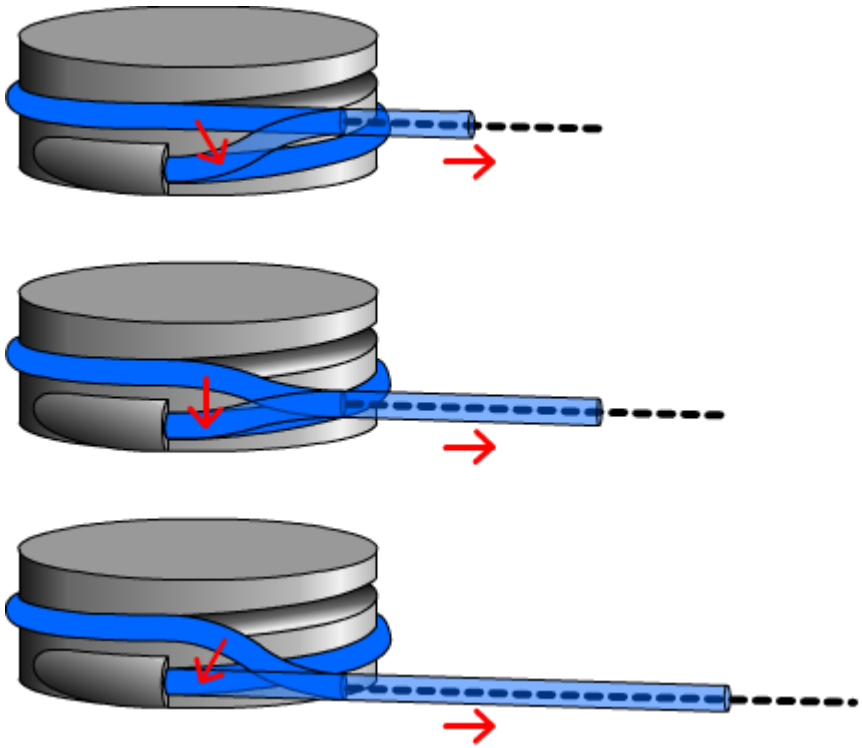


Figure 6-7: Horizontal uncoiling at three different heights

Starting at the top gives an immediate point to Tip Follow from, but limits the final range. While starting at the bottom needs more initial space, however, it provides a greater reach. Operating from the mid-point allows the smoothness and length of the operation to be varied, although only offers an average stretch. For the Gas Turbine the reach is of most importance, while hexapod operations prefer immediate Tip Following actions.

In the end, the decision came down to the force and stiffness simulations performed by Xin Dong. In summary, it was found that the closer the initial bend is to the base section the weaker the resulting pose would be. This means that for the rigidity of the system, the top has the worst affect while the bottom has the least. As such the coiling / uncoiling method to be implemented is a horizontal trajectory on the base plane of the robot.

6.4 Planning the movement

To start determining how to process this coiling motion, the Snake Arm was considered in the fully uncoiled pose since it is easier to comprehend the coiling rather than the uncoiling as each section can be taken in turn. The trajectory considered is horizontal, intersecting directly at the mounting point, the Tip Following handover point is considered to be a sliding point on this line.

6.4.1 Initial bend

Each section, as the drum rotates, needs to bend to conform to the shell of the actuation pack. From the coil profile the final position is known, so the base section can be actuated from straight to this state with the start of the coiling action, as Figure 6-8 demonstrates. However the fixed length of the section leads to a lateral error, as previously explained in Section 5.3.1 and present in the middle stage of the figure.

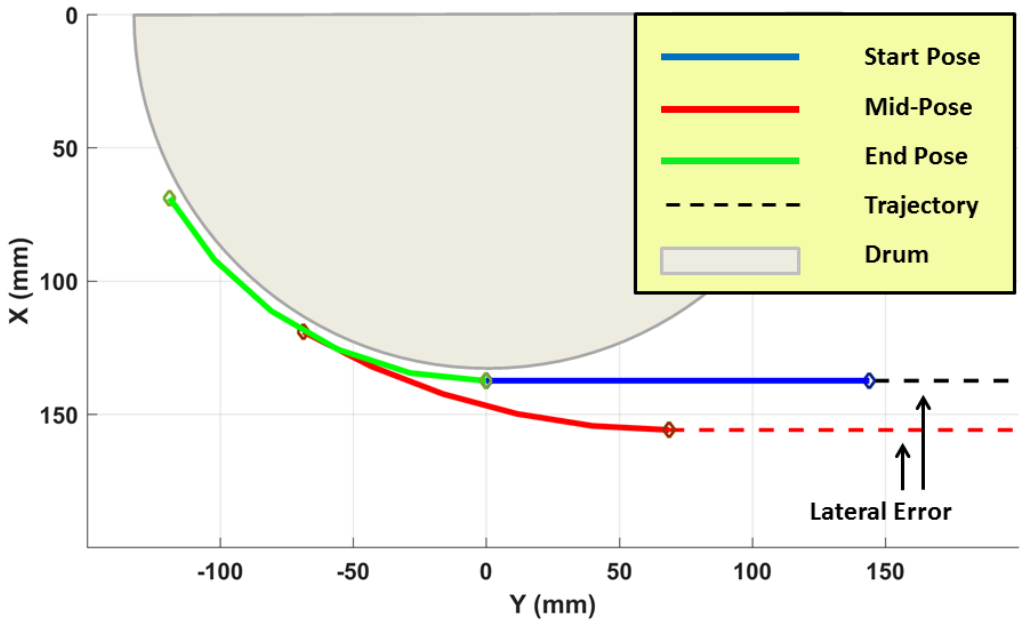


Figure 6-8: Base section coiling on drum and the Lateral Error caused in the process

The amount of bending per step can be found by

$$\theta_i = \theta_{final,i} \frac{\Delta L}{S_i} \tag{6.4}$$

while

$$\phi_i = \phi_{final\ ,\ i}$$

Where θ_{final} and ϕ_{final} are the configuration values from the coiling profile. The corresponding rotation of the drum is defined by

$$\Delta\alpha = \frac{\Delta L}{r\sqrt{1+\left(\lambda_{helix}/2\pi r\right)^2}} \tag{6.5}$$

After combining these actions for the length of the base section, the Snake Arm is in a configuration similar to Figure 6-9. Where the remaining sections would now be straight along the dashed green line at the pitch angle.

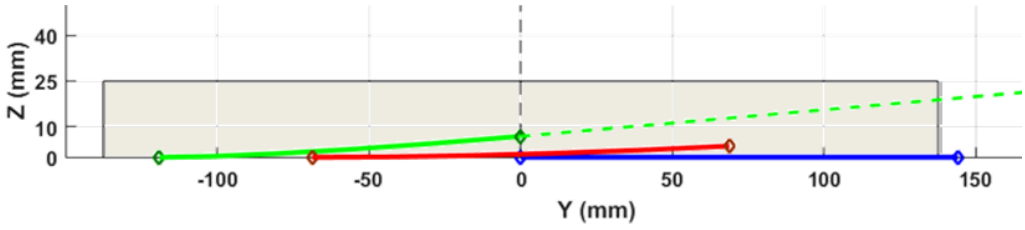


Figure 6-9: Side View of Initial Bend

6.4.2 Vertical movement

Considering the state of the Snake after this initial bend, the base section is now slanted up by the pitch angle creating a vertical shift in the position. As suggested previously, the simplest solution is to create an S-Bend with the next two sections as illustrated before in Figure 6-7. Given the kinematics of the system, it is possible to use this S-Bend to also correct the errors caused during the initial bend therefore ensuring the handover point occurs on the proposed trajectory.

After the initial bend is finished, the first section of the S-Bend needs to be coiled on the drum and the correction moved further down the body. However if this was attempted from the current position the section would intersect with the body of the actuation pack, therefore it needs to be straightened before being coiled. The point this occurs is set at the middle of the initial bend movement, such that the rate of change required is not too fast and the S-Bend does not become overly tight. At this stage, the section is returned to straight by

$$\theta = \theta_{split} - \frac{2\theta_{split}}{S_{split}} \times i \tag{6.6}$$

where the index, i, is millimetres from the split point and continues until S/2mm; after which the section will be straight.

Two sections are still needed to guarantee the corrections, and at some point the straightening section will hinder the S-Bend formation. Therefore, another section along the length is included from the split point. At first it extends the complimentary curve, until the S-Bend is re-formed over the new pair. This means the vertical movement goes through three stages, as Figure 6-10 depicts, while the Snake Arm is coiled.

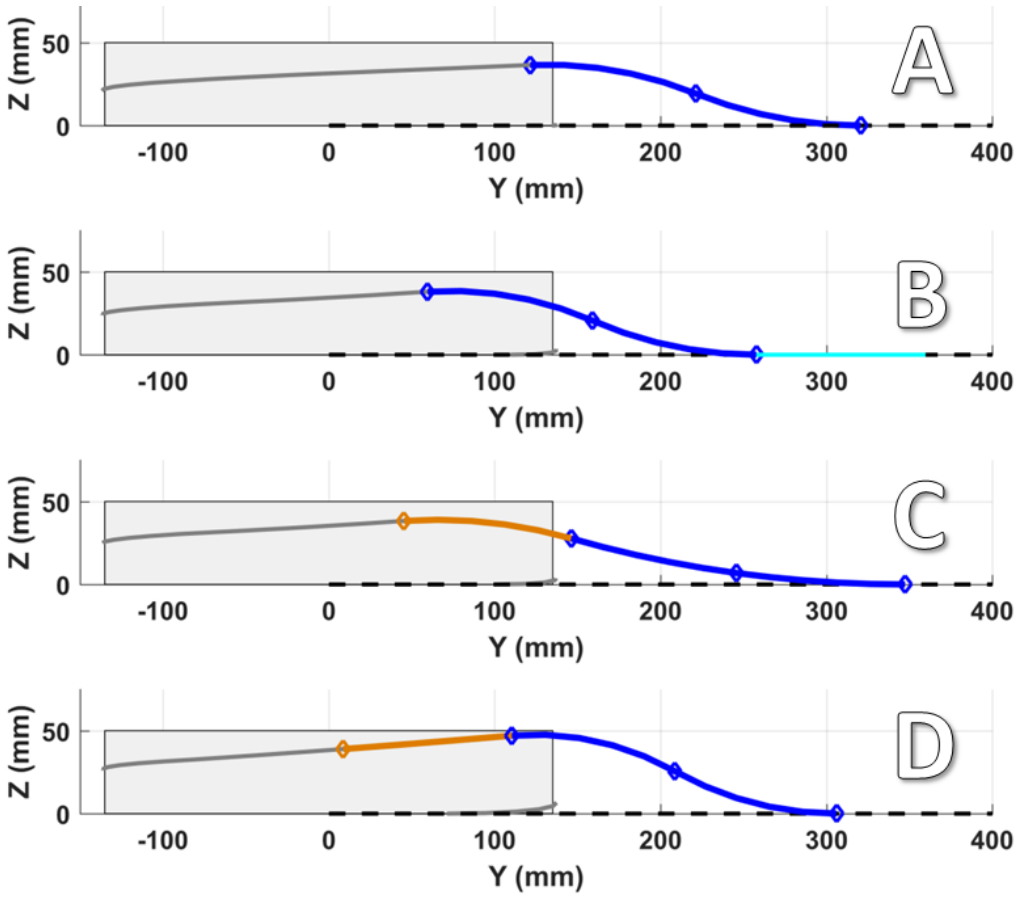


Figure 6-10: Stages of the Vertical Bend A) Two-Section involved in making an S-Bend B) The current pair reached the split point, the next section (shown in cyan) is considered for future use C) First section begins straightening (orange), the other two sections make an extended curve. D) First Section is now straight, with the remaining sections in an equivalent state to A)

6.4.3 Actions required

In summary, three or four sections are involved for the coiling at any given time. One is moving into its final position, as specified by the profile and (6.6), another is potentially straightening, while the other two are performing the S-Bend. The factors to be considered in this correction are the lateral and vertical drifts from the Tip Following trajectory as well as the vertical and horizontal rotation. Given that the coiling and straightening are fixed actions, only the S-Bend can be varied. There exists only one configuration where all these errors are zero, therefore it would over constrain the problem if the desired Y-position along the axis was to be included.

6.5 Optimising the Movement

The easiest method to find the corrective action is to use another optimisation routine to position the sections and minimise the errors. Alternative techniques are possible, such as the Jacobian method discussed in Section 5.3.2 or using inverse kinematic techniques. But the aim is to make this method non-specific to the model, as with the Tip Following, therefore allowing it to be re-addressed to different designs simply.

6.5.1 Objective Function

For this process the objective function is a combination of the four rectifications required. The drifts are represented by their planar distance from the trajectory, Figure 6-11a. While the rotations are represented, in radians, by

$$\begin{aligned} \mathbf{q}' &= R_{Sol} \times \mathbf{q} \\ \omega_x &= \left|\arctan\left(\mathbf{q}'_x / \mathbf{q}'_y\right)\right| \\ \omega_y &= \left|\arctan\left(\mathbf{q}'_z / \mathbf{q}'_y\right)\right| \end{aligned} \tag{6.7}$$

when

$$\mathbf{q} = \begin{bmatrix} 0 \\ 0 \\ 1 \end{bmatrix} \quad and \quad \begin{bmatrix} 0 \\ 1 \\ 0 \end{bmatrix} = R_{Traj} \times \mathbf{q} \tag{6.8}$$

where R_{Sol} and R_{Traj} are the rotational matrices of the section end and the trajectory respectively, as conveyed in Figure 6-11b.

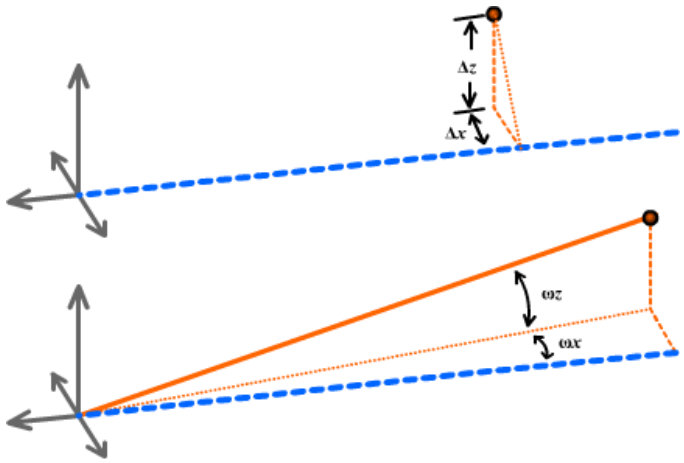


Figure 6-11: Formulation of the four criteria to evaluate the optimisation solution

Since the angles use radians while the drift is measured in millimetres, it was found that a weighting of 100 applied to the angles ωx and ωy led to a balanced result. With this weighting there is a slight favour towards the angles over the drift, which is ideal for the Tip Following. This is because angular error will cause divergence over the length of the Arm, whereas a translational error is consist.

6.5.2 Setup

The optimisation itself is setup to use the previous solution as the initial for its process, since any new solution should be relatively close to the old. With this in mind the constraints are set such that

$$-0.710° < \theta < 0.710 \qquad -180° < \phi < 180° \tag{6.9}$$

similar to the values used for the Tip Following. This method will however limit the current process to contiguous sampling, but in the next section a method is employed to deal with this.

6.5.3 Fixed Step Validation

Initial validation of this approach was performed using fixed 2mm steps covering the length. For each step the errors after the optimisation are analysed by the same criteria and displayed in Figure 6-12.

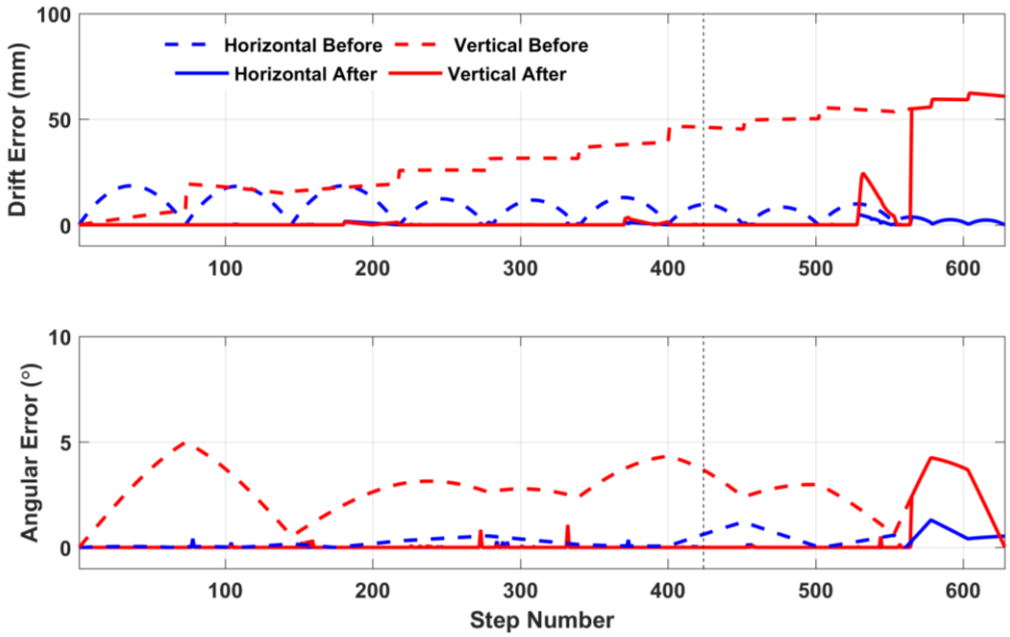


Figure 6-12: Errors before and after the optimised S-Bend

It is clear from the figure that the errors have been reduced to a negligible level. To the point that is difficult to show the variations in the results in the same scale of the original errors. There are noticeable spikes at 273, 332 and 373. These are caused when the length of the sections involved changes, and likely a programmatic typo which is still being investigated, but they are within the target tolerance.

Through this validation it was found that once reaching the pose shown in Figure 6-13 at step 424 (as indicated on the previous figure by the dashed line) it is impractical to try and achieve good results bending the last three sections. The main reason being that these sections are only 50mm long each and there is insufficient length remaining. As such the position at step 424 is considered as the starting position for Tip Following, it is referred to as “TF_READY”. From this point the robot will fold / unfold the

remaining sections by the process of two steps in Figure 6-13a and Figure 6-13b to store the Snake Arm for storage and operations on the walking hexapod.

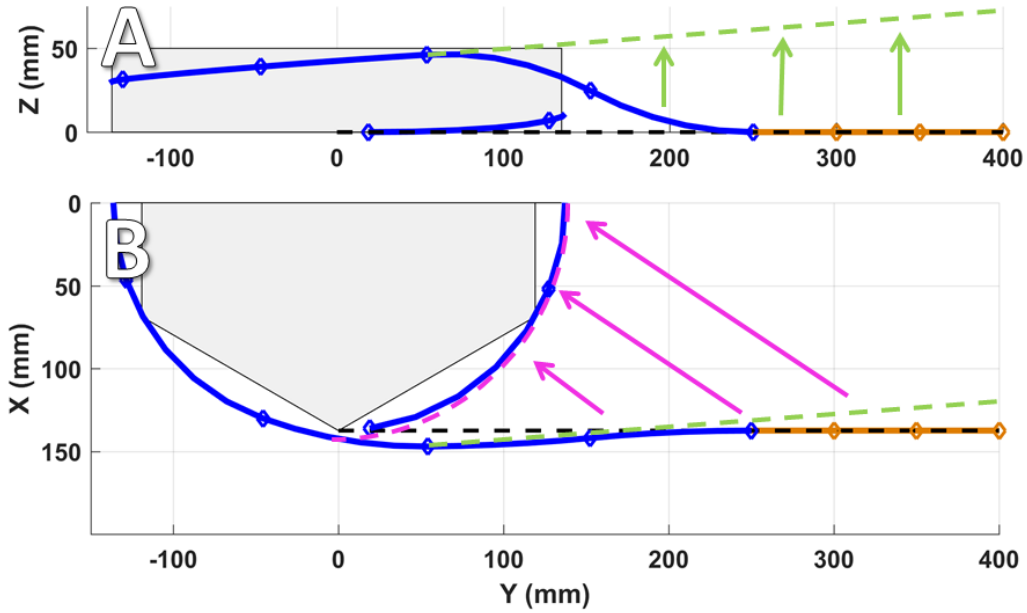


Figure 6-13: TF_READY pose and the steps to pack the remaining sections A) Straighten all sections that have not been coiled B) Wrap remain sections into their coiled pose.

6.6 Non-Continuous Step Function

The data gained from the validation can be used as reference for a randomly-accessed function. Generally this works by indexing the array of poses generated, and using the configuration as the initial input to the optimisation. This section describes the method used step by step and is summarised in the flowchart in Figure 6-14. The index in this code is considered to start at 0, to avoid any confusion.

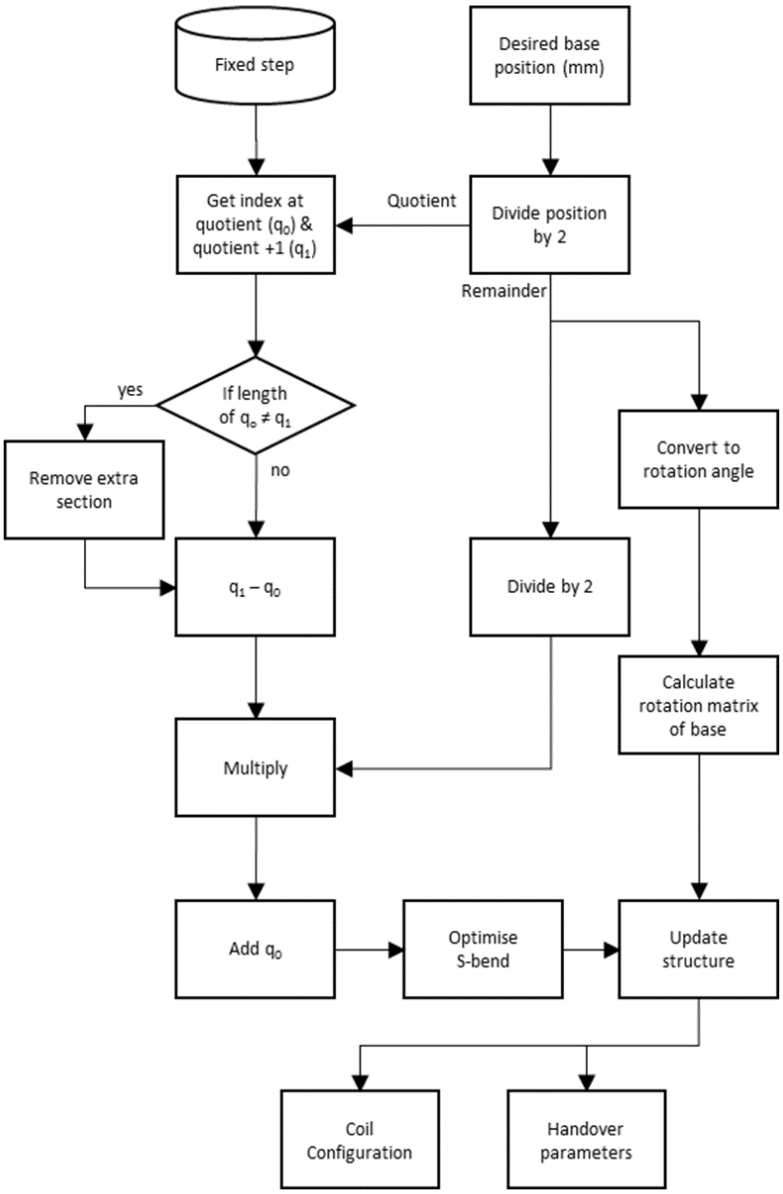


Figure 6-14: Flowchart for the Non-Continuous Step function

The input for the function is the desired position in millimetres. Based on the total extension length of the tip following, where 0mm is the TF_READY position and 848mm is the full actuated length. Due to the validation being performed as a coiling process, the array of poses needs to be inverted.

The first action taken is to divide the input value by 2. Since the validation was taken every 2mm, the quotient provides the index of the proceeding pose, and adding one

gives the subsequent. Meanwhile the remainder can be used to calculate a rotation matrix to adjust the base position and rotation of the new configuration.

When sections are removed from the S-bend this leads to a discrepancy in the length of the arrays, making the comparison that follows impossible. Therefore the extra section on the proceeding pose is removed, and considered as part of the tip following. By the mechanics of the previous function, this section is straight or will be immediately after.

Taking the difference between the θ angles of the subsequent pose and the proceeding one provides the amount of change that occurs in the bending. Such that

$$\begin{bmatrix} \Delta q_1 \\ \Delta q_2 \\ \vdots \\ \Delta q_n \end{bmatrix} = \begin{bmatrix} q_{1,1} \\ q_{1,2} \\ \vdots \\ q_{1,n} \end{bmatrix} - \begin{bmatrix} q_{2,1} \\ q_{2,2} \\ \vdots \\ q_{2,n} \end{bmatrix} \tag{6.10}$$

Multiplying these values by half the remainder gives the difference to the new pose. Adding to the proceeding pose θ, gives the intervening configuration at the desired point. As for ϕ, the direction angles only change when the sections switch between the states defined in Section 6.4.2. With this mind the new configuration can directly copy the φ of the subsequent.

For the final stage in the process, the last two sections from the intervening pose are separated. These are then processed by the optimisation as before, and then reattached to the formation. The rotation variable is updated in the structure and the result parameters are identified for use by the tip following in the next chapter.

6.7 Results

The quality of this feed-in system is dependent on two key aspects: the computational performance and the accuracy. If the code is slow it will affect the duration of the Tip

Following when combined, if the accuracy is poor it will affect the drift of the Tip Following when combined. To verify the computational, the raw outputs of the function will be assessed, whereas the accuracy will be checked using simulation of the configuration. The same computer is used as in Section 5.7

6.7.1 Non-Continuous Function Performance

The performance of the function needs to be investigated. Firstly the computational time, which can vary based on what advancement is requested, and secondly the actual advancement that is produced by the coiling since so far it has been assumed equivalent.

Figure 6-15 below shows the comparison between the requested movement and the computational time. This was sampled every 0.3mm from 0mm to 848mm. From this data the average time is 0.031s with a maximum of 0.081s. It was expected that the points when no re-calculation is needed would have significantly less computation time since the indexed pose can be used. The magnitude of the measured periods are almost negligible to the Tip Following

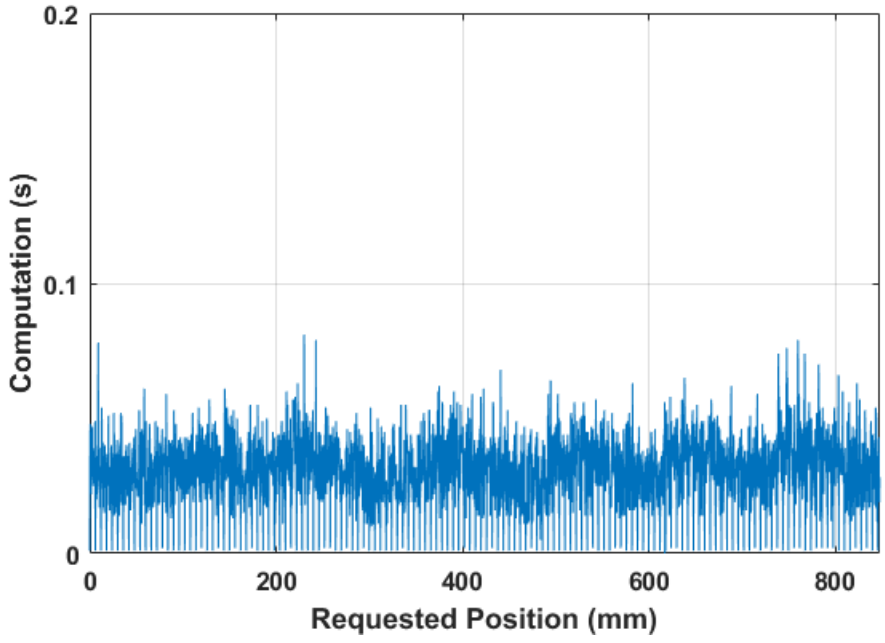


Figure 6-15: Computation time versus the requested position

Using the same sampling, the graph in Figure 6-16 was generated to compare the requested position against the solution out. There is small variance in the line due to the values Δx and Δz not being exactly zero, so the point is not always on the trajectory. Overall there is a 1:1 relationship demonstrating this method is suitable for use with the Tip Following algorithm presented in Chapter 5

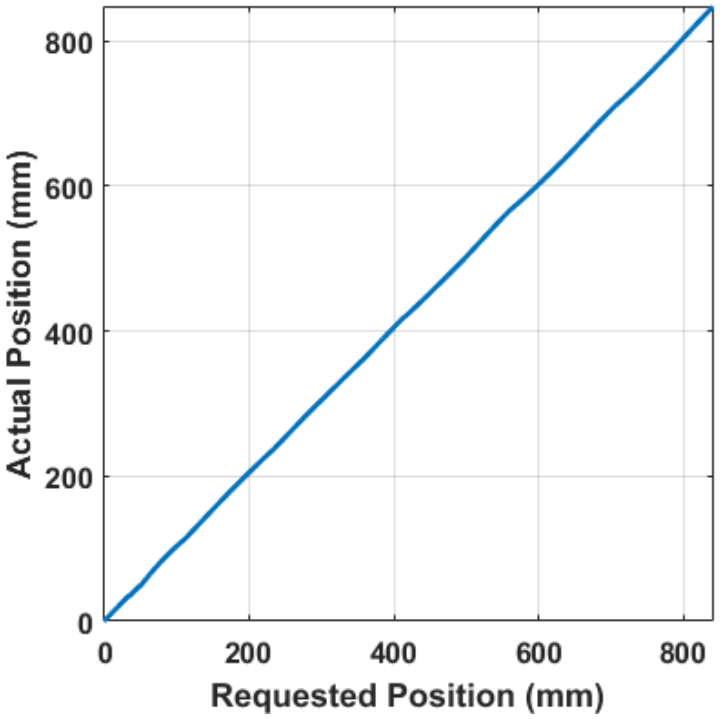


Figure 6-16: Requested Position versus the Actual Position of handover point from the feed-in function

6.7.2 Simulation

The timeseries in Figure 6-17 represents a simulation of the process in action. Note that as sections are uncoiled and no longer considered part of the feed-in they are coloured orange. From this figure it is clear that all the sections involved can coil on and off the drum and that the trajectory is maintained. The next step is to verify the accuracy quantitatively.

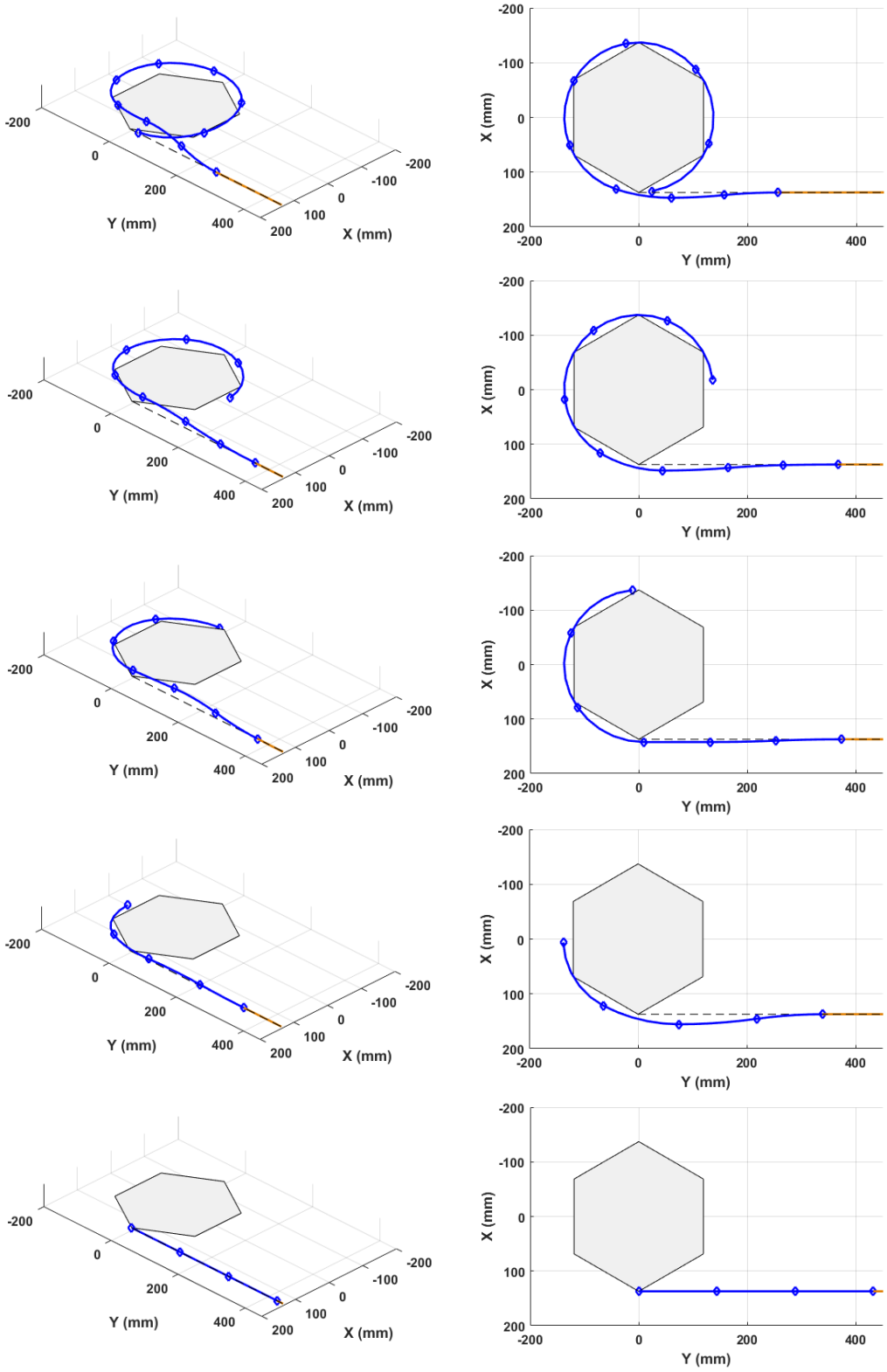


Figure 6-17: Timeseries of an uncoiling process.

Using the same measurements as the objective function, the graphs in Figure 6-18 where produced. The first shows the drift, while the second denotes the rotation. In both graphs the cloud of data points represent all the samples in the simulation. The process used the same sampling as before. The dotted circles on both graphs represent the desired specification. These are 3mm and 1° respectively. It is plan to see the majority of these points are within this limit. The outliers that can be seen have been determined to have a direct link to the previous mentioned programmatic error.

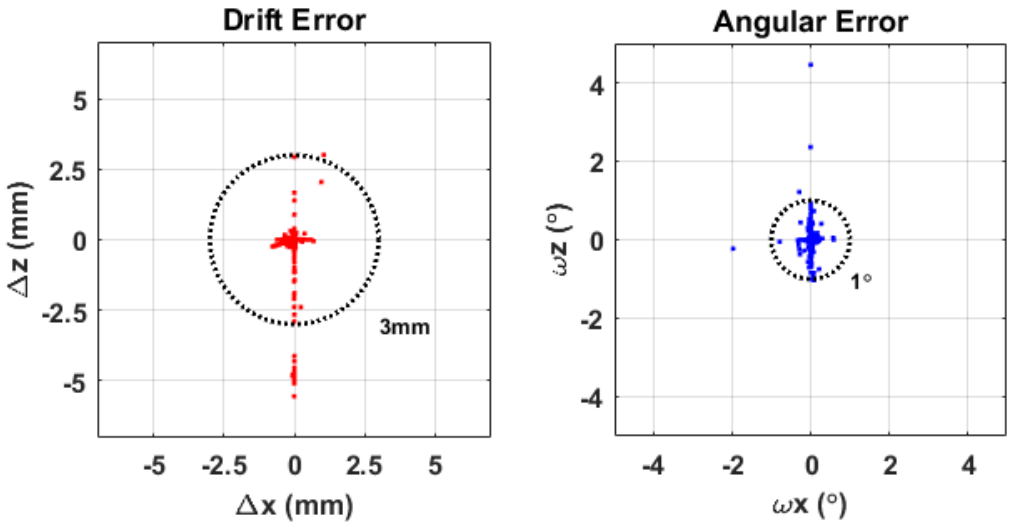


Figure 6-18: Point clouds of the drift and angular errors after the optimisation process.

6.7.3 Physical Trials

The poses generated can be tested on the fourth prototype. By running through the fixed step validation list backwards the Snake Arm can be coiled from its rested straight state to the TF_Ready state, as pictured in Figure 6-19. Then by providing the simulated function forwards the system can be uncoiled as intended to any point.



Figure 6-19: Final Prototype coiled to the pose TF_READY

A timeseries of video stills of a trial can be seen in Figure 6-20. The coiling is successful, with the trade over point within the expected performance. However the mechanical problems within the demonstrator cause the tip to dip and rise as it is uncoiled.

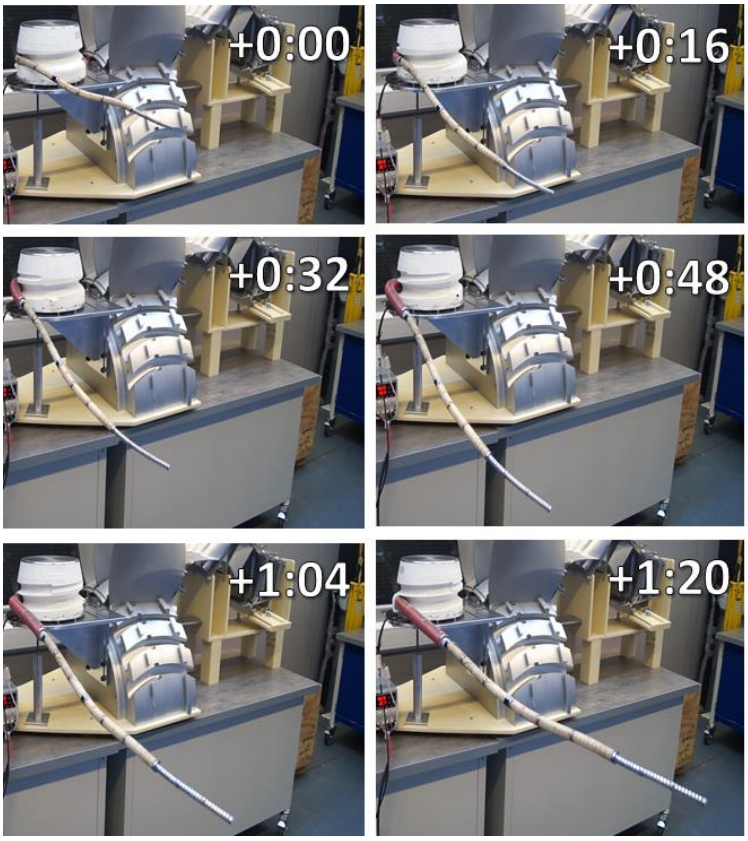


Figure 6-20: Timeseries of uncoiling action performed on Final Demonstrator

6.8 Chapter Summary

This chapter has developed an active method for coiling and feeding the Snake Arm. From a specified helical path, a coiled profile was found that would store the body of the robot within the recess. This was performed by an optimisation process after forward kinematic techniques would not find a sufficient solution because of the geometric principles.

Different uncoiling strategies were analysed. The one chosen to best solve the scenarios required within the project. However, all the methods in 6.3.2 are immediately feasible using the techniques developed. Since the lateral errors in 6.4.1, the single-bend approaches would still require an S-Bend. This means that for any trajectory that intersects with the pitch trajectory at the virtual corner can be solved in the same way.

The stages to coil the Snake Arm were investigated, leading to three clear actions required. The bending to coil a section into the drum and the point the subsequent section needs to prepare were determined to be fixed and the equations governing their actions proposed. From this point an S-Bend was shown to solve the remaining movement, and compensated for the errors caused in the process.

To define the configuration of the S-Bend, an optimisation routine was set up. This was able to minimise the four errors suggested to ensure the handover point exists on the desired trajectory. This was then validated using a fixed-step, before being implemented into a function that could be polled at any value of the range.

The performance and accuracy of the resulting algorithm has been tested and verified such that it is suitable to be used with the Tip Following. Computational times are on average only 0.031s and therefore negligible to the combined processing time, while the linear advancement is equivalent to the requested input. For accuracy, the drift was kept mostly within 3mm with only 1° of angular error. At these levels, there should be no problems for the Tip Following.

7 Combined System

7.1 Introduction

As briefly detailed in Chapter 3, the system can be broken down into two distinct categories: the hardware architecture and the software interaction. In this Chapter the combination to create the complete system is discussed. The main challenge remaining is to conglomerate the Tip Following from Chapter 5 with the Feed-In from Chapter 6 and operate it from the High-Level.

7.1.1 Hardware

The hardware is setup across three main parts, as depicted in Figure 7-1, there is the Linux PC which runs the High-Level Control, the Windows PC running the LabView interface and the Tip Following, and the robot's on-board controllers. These are connected via a local Ethernet network. Cameras mounted within the end-effector communicate via wireless to the router and can be accessed by the Windows PC.

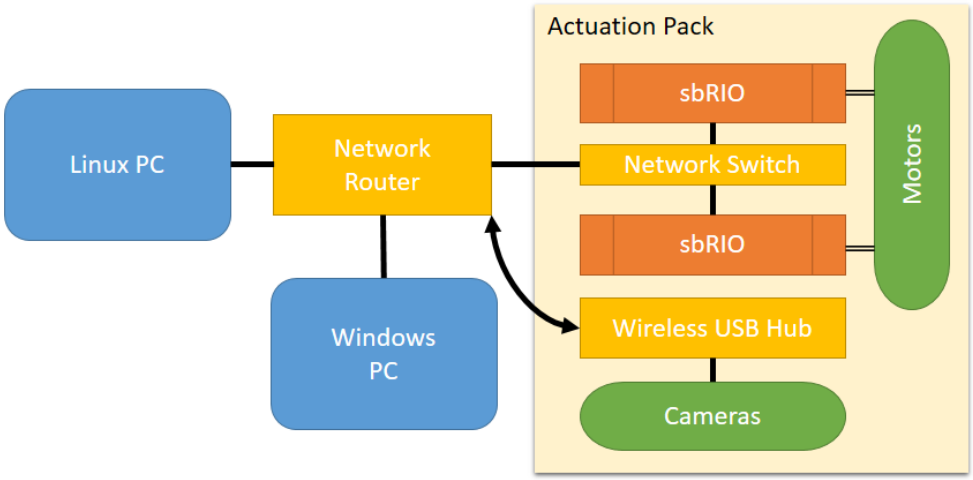


Figure 7-1: Hardware connection topography for the final prototype

7.1.2 Software

On the software side, the Snake Arm itself is controlled by the LabView GUI running on the Windows PC utilizing the in-built communication with the sbRIOs. This GUI can either take direct user interaction, import and export pre-calculated pose arrays, or be commanded directly by the High-Level. The final Tip Following uses the Feed-In as a sub-VI within itself. It is either commanded directly by a user, or from the High-Level controller; the output can either be returned to the High-Level or saved to be imported. All communication with the High-Level happens using the ROSbridge interface over ethernet.

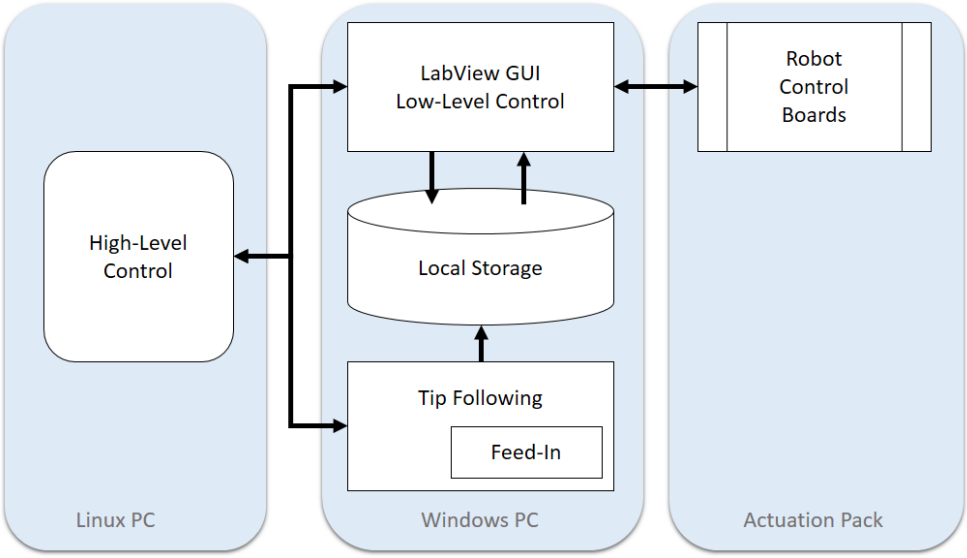


Figure 7-2: Software connections

7.2 Combining the Feed-In with Tip Following

The Tip Following needs to combine the Feed-In, without causing more errors in the results and compensating for any errors that remain after the Feed-In. Looking at the flowchart of a single step, Figure 7-3, there are two outputs the Feed-in needs to provide.

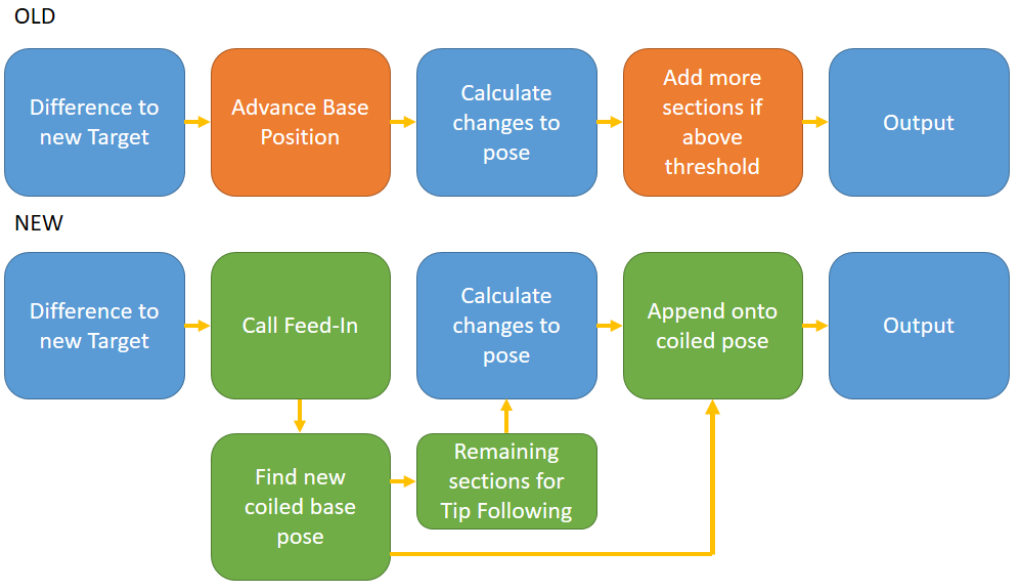


Figure 7-3: Flowchart of Tip Following algorithm, and the steps added to make the combined procedure

From the difference in the tip position and the new goal, as before, the advancement of the base is determined. This is then given to the Feed-In which produces the coiled position and calculates the arbitrary base position for the Tip Following as well as the number of sections available. The Tip Following then assigns the base position to the previous result and adds extra sections if needed. Once the optimisation completes, the two halves are joined to make the full Snake Arm representation.

7.2.1 Linear Stage

During the testing and demonstration of the prototype it was found that the mechanical errors in the hardware made the coiling particularly hard. Since the behaviour of the robot was not as intended, this often lead the snapping of a steel cable within the arm. It was decided to avoid additional delays to the project and to mount the actuation pack on a linear track, Figure 7-4, instead for the Gas Turbine experiments. This was implemented in the same way as the Feed-In. Further work after the project will explore improving the mechanical system to make this operation viable, but has yet to be done at time of writing.

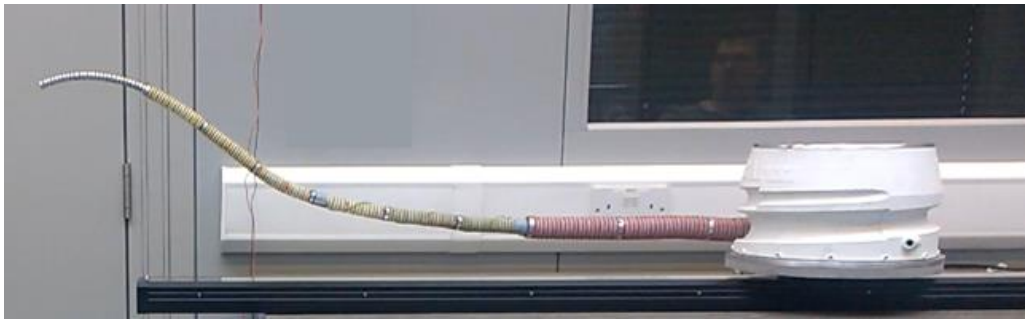


Figure 7-4: Final Prototype mounted on Linear Stage

With focus shifted to a linear implementation, less time was dedicated to the uncoiling methodology. As such the final corrections to the implementation have not been emplaced. A rough experiment of the combined algorithm is displayed in Figure 7-5, where the arm is navigated along a single planar sweeping arc within an enclosed environment. The slight "bunching" effect at the tip is caused by a small misalignment of requested base pose and the actual, it is believed that updating the Tip Following with the actual position of the Feed-In will combat this issue as the Tip Following is assuming the advancement formed is exactly the same as requested

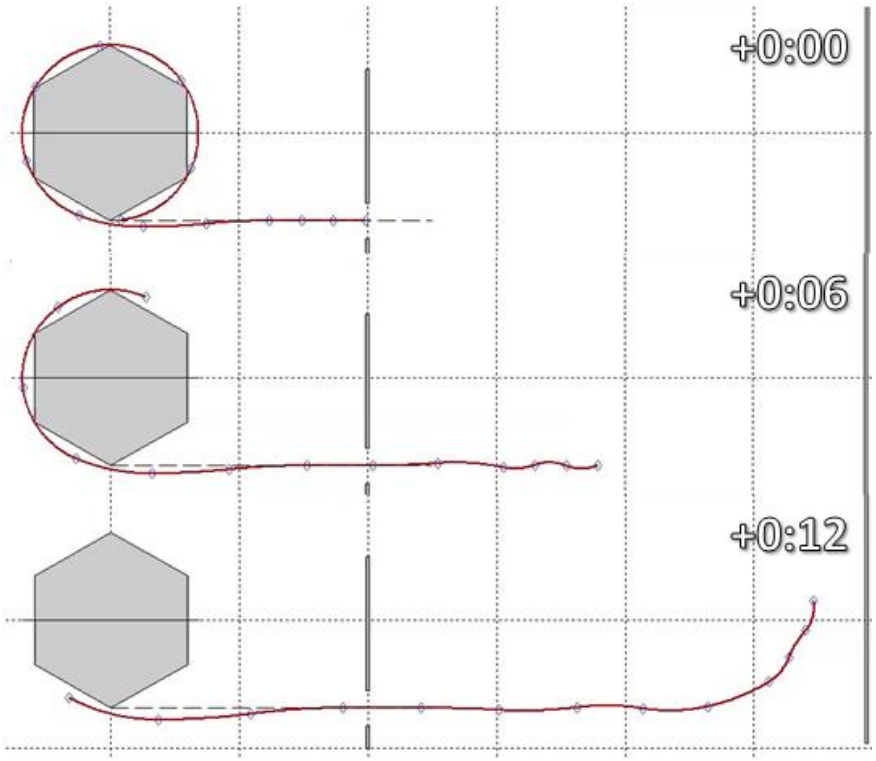


Figure 7-5: Early experiment of the combined uncoiling and Tip Following algorithm

7.3 Communicating with High-Level Control

The high-level control of the robot has been written in Python using the Robotic Operating System (ROS), it is the work of Felix Messmer and the rest of the team at Fraunhofer IPA. The LINUX computer it is implemented on communicates to the rest of the system using Ethernet. Messages are managed by the ROSBridge V2 module, they appear as strings using the standard JavaScript Object Notation (JSON) format.

Communication with the ROSBridge can work in two ways. These are either “Topic and Subscriber” or “Service and Request”. In the first format, one of the agents on the network can create a “Topic” which can be “Subscribed” to. Typically this is for regularly polled message threads, where updates are sent regularly regardless of them being read or required. Whereas the “Service” allows agents to send a “Request” which the “Service” provider will respond to when it is complete. In this way it prevents flooding the network with large messages and the request handle allows the answer to be correctly correlated with the question. The code implemented in the final prototype makes use of both methods.

7.3.1 Commanding the Low-Level and Tip Following

The HLC communicates with two aspects of the system, as shown previously in Figure 7-2, the low level control and the Tip Following. With the LLC, the High-Level can command actions and receive the current status. Both these interactions use the configuration notation for the Snake Arm, and are sent using the “Topic and Subscriber” approach. Actions are sent frequently enough when running a series of poses that the request method would slow the process. The status updates are non-critical if messages are missed, but regular publishing is required.

Commanding the Tip Following is more elaborate. A large string is required to send all data required, and a larger string is used to send the resulting array of poses. The data the algorithm needs is: the trajectory to follow, the current configuration of the arm and the previous path history. Only the trajectory is needed if the pose is known to be the initial position, whether that is TF_READY for coiled operation or fully straight for linear.

To ensure successful transmission and receipt not only is the “Service and Request” method used, but also the messages are fragmented to prevent loss caused by buffer overflows. These fragments can then be reassembled on the receiving end.

7.3.2 Features of High-Level Controller

The purpose of the HLC is to provide more advanced features to the system, more abstract from the robot itself. Core functionality comes from the ability to simulate the Snake Arm, the hexapod and the environment. This allows the high-level to co-ordinate their relative positions to each other within the global frame, and align the Snake Arm.

7.3.2.1 Planning Paths using CAD Data

The logical extension of this is to use the CAD data of a scenario, align the entry point in simulation and physically. Then using path planning protocols a route can be found to a desired end position. Traditional techniques generate paths that tend to have sharp, instantaneous bends around edges and bottlenecks, as depicted in Figure 7-6. To prevent this a torus shape constraint is added, where the cross-section radius is equal, if not greater, than the minimum bending radius of the robot. The path is then broken down into Cartesian points along its length, with no more than 2mm separation. In this way a path can be found that is achievable by the Tip Following.

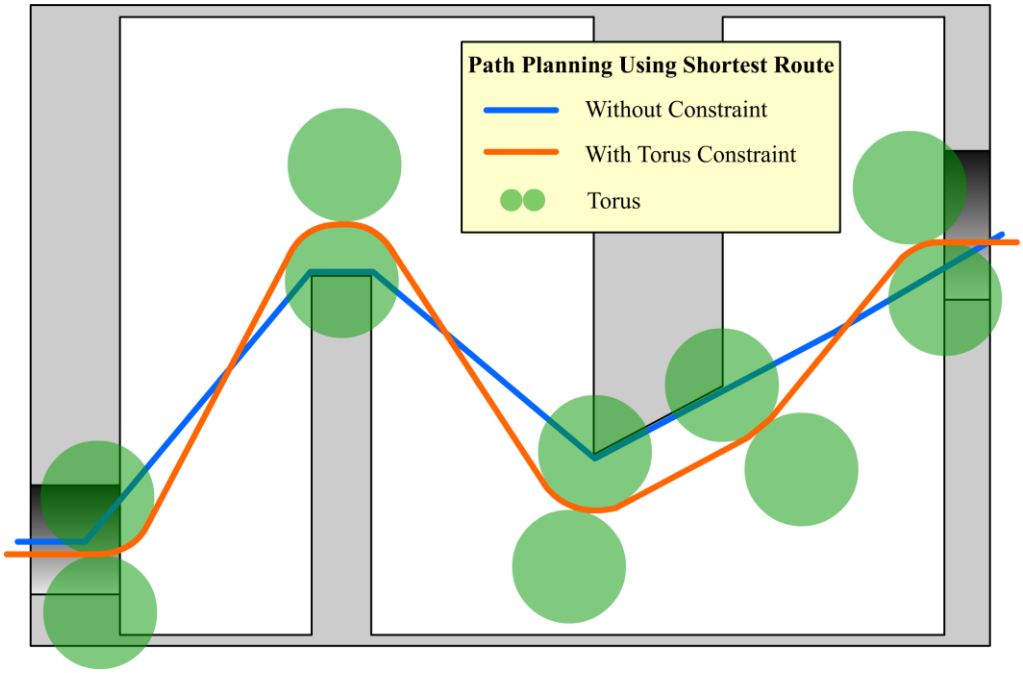


Figure 7-6: Diagram of the effect the torus constraint has on the path planning

The screenshot in Figure 7-7 displays the simulated environment, after a path has been planned. To operate the user moves the cursor ball to the desired position and orientation, where the blue arrow is the axial trajectory aimed for in the final pose. Once calculated the path is shown as a series of white arrows, these attempt to simulate the rotation by their direction but since the Tip Following does not consider this it is not assumed to be correct.

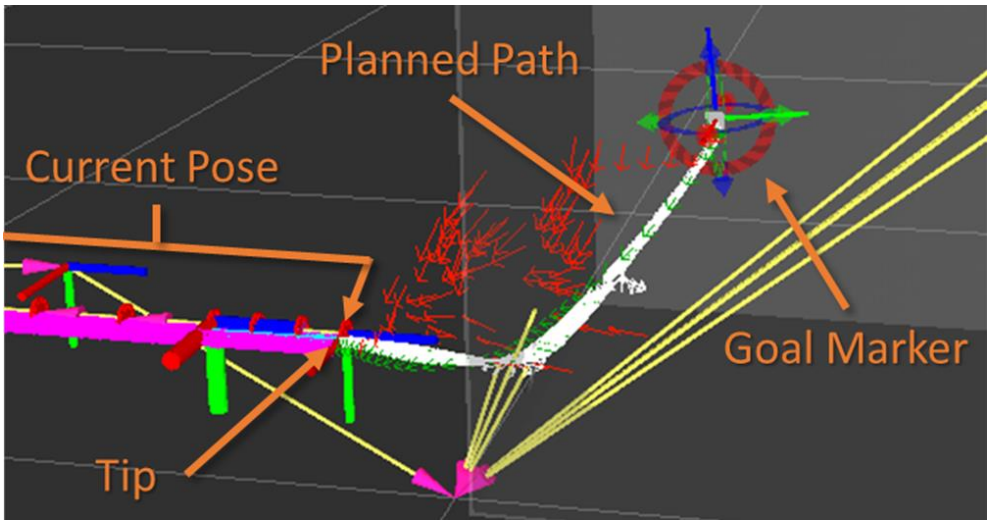


Figure 7-7: CAD visualisation used for generating paths based on the goal position and environment data

7.4 User Interaction with Tip Following

Previously, the user generated a path for the Snake Arm to follow by entering the desired final shape. While this method works fine it has some limitations. First and foremost, the user may not know what the environment is or where they want the robot to go ahead of time. The other issue, as stressed previously, the configuration parameters become incomprehensible to a user as more sections are added making this method unintuitive.

Through the setup of the high level, a set of commands have been implemented that allow the user or the intelligent controller to operate the robot interactively in a turn-base fly-by-wire process. This means that for each “turn” the user can decide the heading and the action, when the turn completes they can command the next turn. These commands can be implemented in the low level as well, as not dependent on the advanced functions.

7.4.1 Endoscope Mode

Endoscope Mode is the main method for user operation. As the name suggests, it tries to simulate the behaviour of a normal Endoscope. The aim being this would be a familiar system for end-users and therefore reduce the learning curve.

By controlling a two-axis cursor, Figure 7-8a, the tip section can be directly controlled. With the aid of the camera view, the user can then determine which route is best to head next. Then provide a length for the Snake Arm to be advanced straight from the new position. By extending the endoscopic action over more sections a smoother wider curve can be made which would reduce drift in the Tip Following.

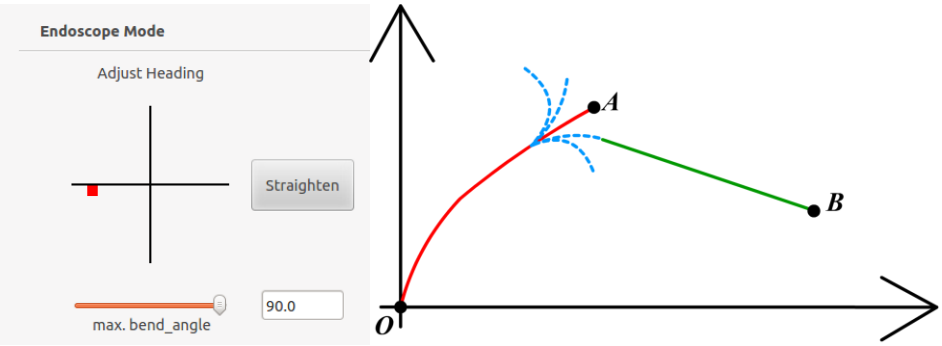


Figure 7-8: Interface for Endoscopic navigation and illustration of the process

To implement this, the process in Figure 7-8b is used. The new arrangement of the Snake Arm becomes the history of poses. This can cause a slight drift in future navigation from previous, as the Tip Following it not exactly equal to the desired path. Therefore it allows the errors to be greater than they would be given the original path. The trajectory itself is just a set of Cartesian points transformed to start from the tip in the correct orientation.

The interface can be made more intuitive by including the work on implementing an Xbox 360 Controller in Chapter 4. As described, this would allow the user to have a more direct feel for the actions, and resulting outcome. Due to time constraints, this has not been instigated at time of writing.

7.4.2 "GoHome" Mode

By using the Endoscope mode, or just several consecutive trajectories, the returning function is no longer a simple case of reversing the array of poses used. Instead it is now required to reverse each step in turn, including the Endoscopic changes to the tip sections. Therefore a single-action return method was implemented, referred to as "GoHome".

The easiest approach is to take the current pose as the trajectory for a new Tip Following request from the initial position. The result can then be used in reverse to retreat the Snake Arm. However there is the possibility, as before, that the current pose is not equal to the starting pose. This would lead to a small jump in position but it should be compensated for by interpolation protection of the low-level control.

7.4.3 Branching Mode

With the addition of the "GoHome", a new mode is possible. Instead of retreating to starting position, the reversal can be halted part way allowing a different trajectory to be calculated. In this way the arm, can "branch" from the original path. So as Figure 7-9 demonstrates, the Arm can be run to point A then returned to B to navigate to C without having to reverse to O.

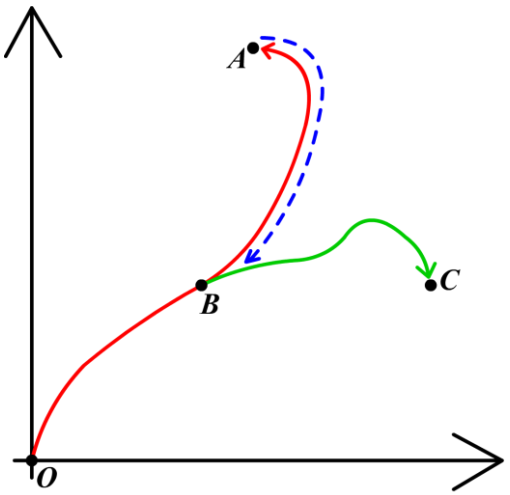


Figure 7-9: Branching Mode, rather than retreating to O the GoHome can be stopped at B and a new path can be planned to C

Programmatically this works by interrupting the "GoHome" action while the robot is moving. Taking the path used to reach the current pose as the history, a new task can be computed as normal. Unlike the previous modes this does not add any drift to the system but does build off the drift present in the "GoHome".

7.5 Including the Machining Capabilities

The machining algorithm is used to generate an array of poses such that the tip sections of the robot can perform a machining operation as required within the engine scenario. To create the machining path, the technique used takes what was learnt by the method in Chapter 5.3.2 and applied with a more sophisticated Jacobian for the problem. This allows the algorithm to produce a smooth trajectory while maintain the angle of intercept between the tool and the blade.

Before the part can be machined, the tip sections need to be moved into position relative to the work piece. This is determined by another inverse kinematic solution that has been developed. Once the desired position is calculated, the configuration between the two poses is interpolated.

Both these actions are imported into the main low-level GUI. As the flowchart in Figure 7-10 shows, after the Snake Arm is navigated into the target area, the arm can be moved into place by the positioning array. Then the machining array can be repeated as long as required until the process is considered completed. Finally the arm can be reset by reversing the positioning array and Tip Following can be continued as normal.

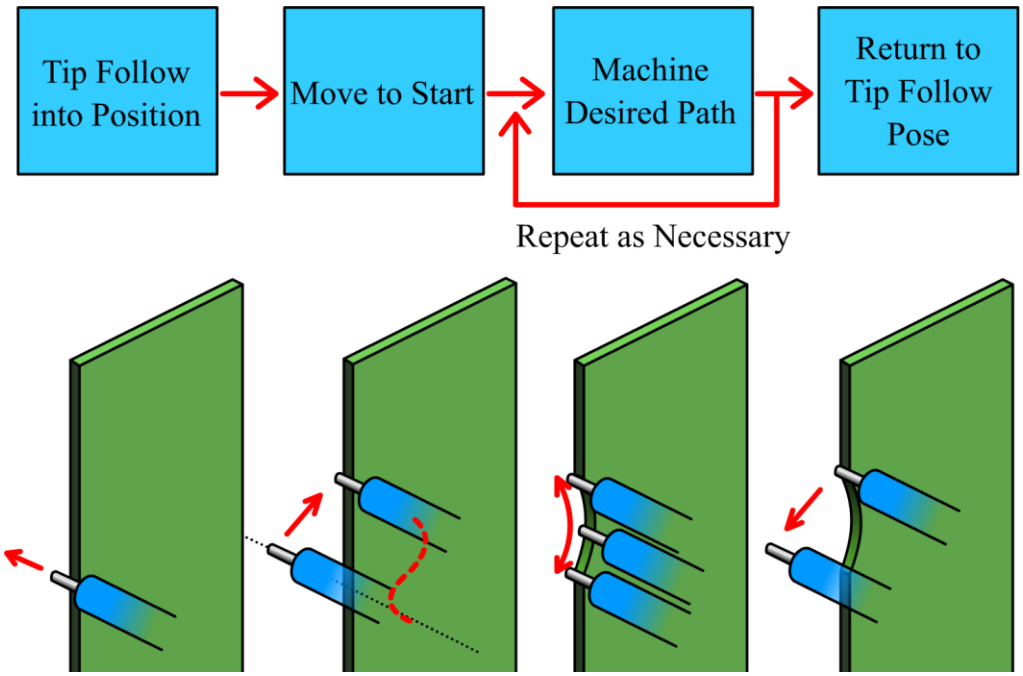


Figure 7-10: Machining Process

7.6 Results

Through the combined system the full functionality of the system can be realised. Firstly the combined Tip Following, with representation of the physical feed-in, is simulated to prove the new modes of action and the communication with the high-level controller is functioning without detrimental effects. Then the planned deliverables are performed.

7.6.1 Simulation of Tip Following with HLC

Several simulations where performed to test the complete Tip Following algorithm working, and the different modes it can be operated with. The simulation in this section is one continuous process, consisting of five actions. Each action is presented graphically, with the performance data summarised at the end.

7.6.1.1 TF_Ready to CAD location

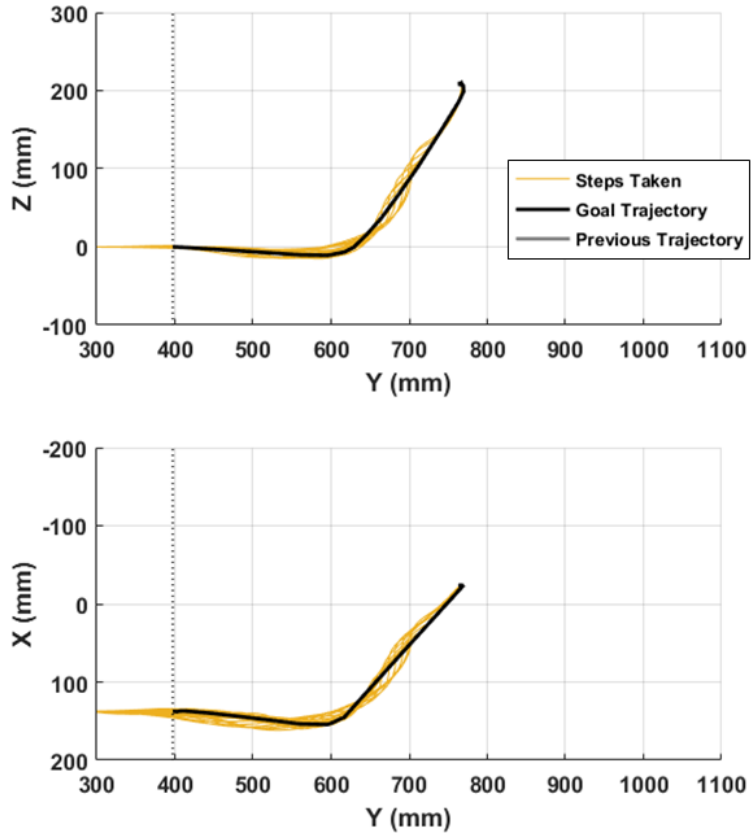


Figure 7-11: Side view and Top view of first action

Figure 7-11 shows a selection of the paths used to navigate from the initial position, marked by the dotted line, to a new point set by the goal of the CAD method. Figure 7-12 displays the planned path before and the resulting pose of the arm after execution. For these trials a simple box environment is considered, such that the arm has to navigate through a small hole but is free to explore once inside.

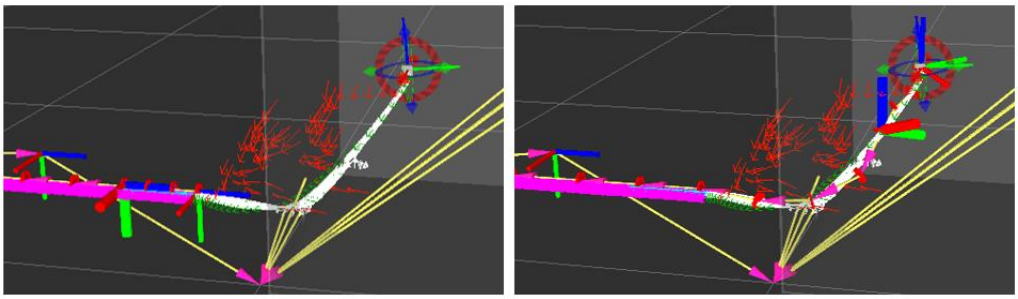


Figure 7-12: Commanded CAD movement in the visualization, planned motion (left) and resulting action (right)

7.6.1.2 *Extend from CAD pose by Endoscope*

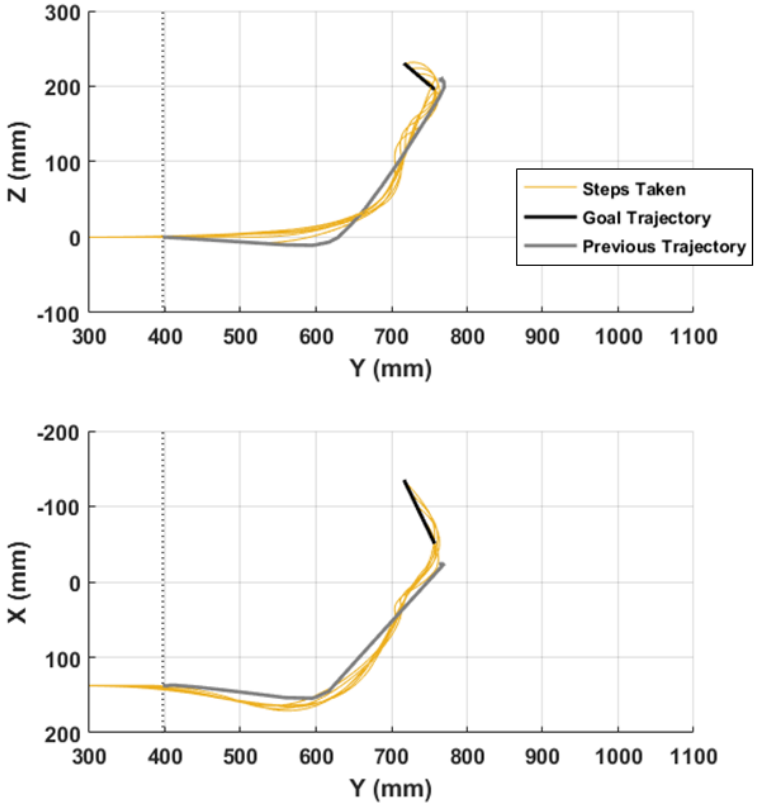


Figure 7-13: Result series after Endoscopic navigation

The grey line in Figure 7-13 represents the previous pose at the end of the previous action. It is obvious that this line does not connect with the black trajectory. This is because the tip was modified using Endoscope mode to be facing a different direction. Path deviations in this run are larger than normal because an extreme angle was taken for the Endoscope to make it noticeable. This leads to the perceptible overshoot evident along the length of the trajectory. In normal use, smaller angles would be used and over shorter distances. From Figure 7-14 the heading can be seen and the distance to be advanced is set at 100mm.

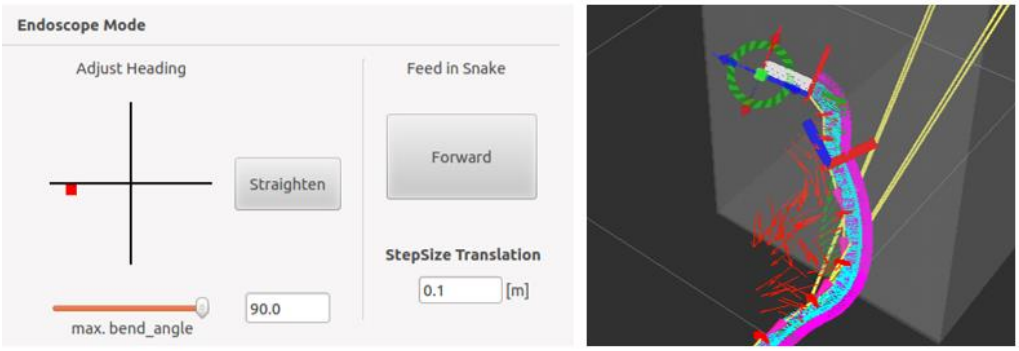


Figure 7-14: Commanded Action and the resulting trajectory in the visualization

7.6.1.3 "GoHome" for Branching followed by a CAD goal

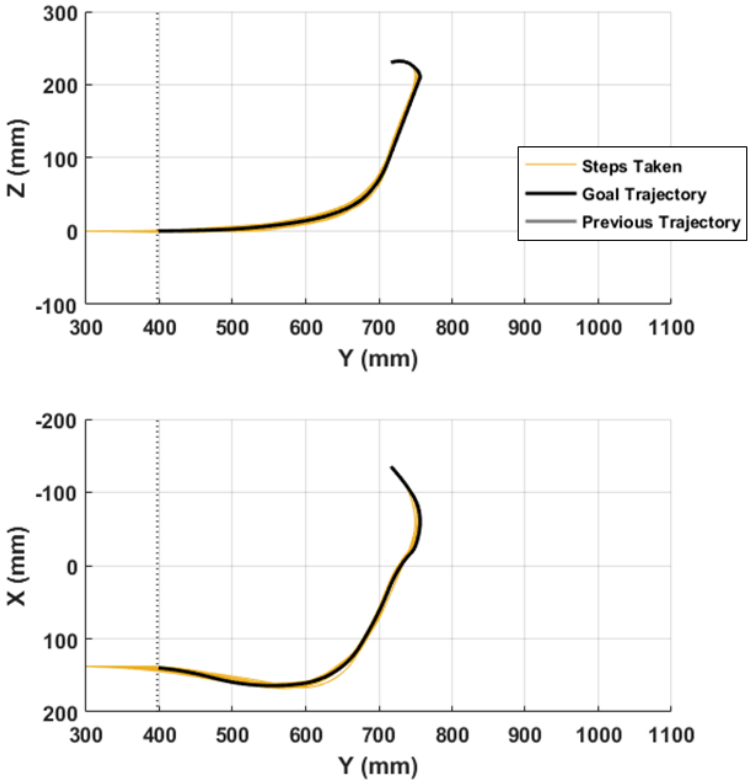


Figure 7-15: Results after planning "GoHome" action

In the aim of branching of in a different direction, the "GoHome" action is called. As mentioned in Section 7.4.2 the results, in Figure 7-15, are planned from the base then reversed to perform the desired movement. The execution is then interrupted approximately half way, and the path in Figure 7-16 is then proposed.

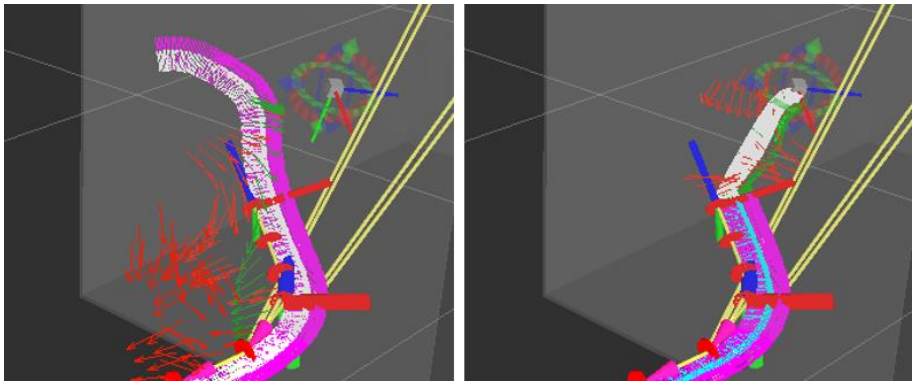


Figure 7-16: Visualization of the execution interrupted (left) and the new path generated (right)

Figure 7-17 has a grey line to show the previous pose before the "GoHome" and the results from the Branching action. Similarly to the Endoscope action, there is overshoot along the trajectory. This is cause by the sharp angle used by the path planner. The torus constraint only limits the path to the mechanical limit. A better path would use longer curves and minimise the number of direction changes needed.

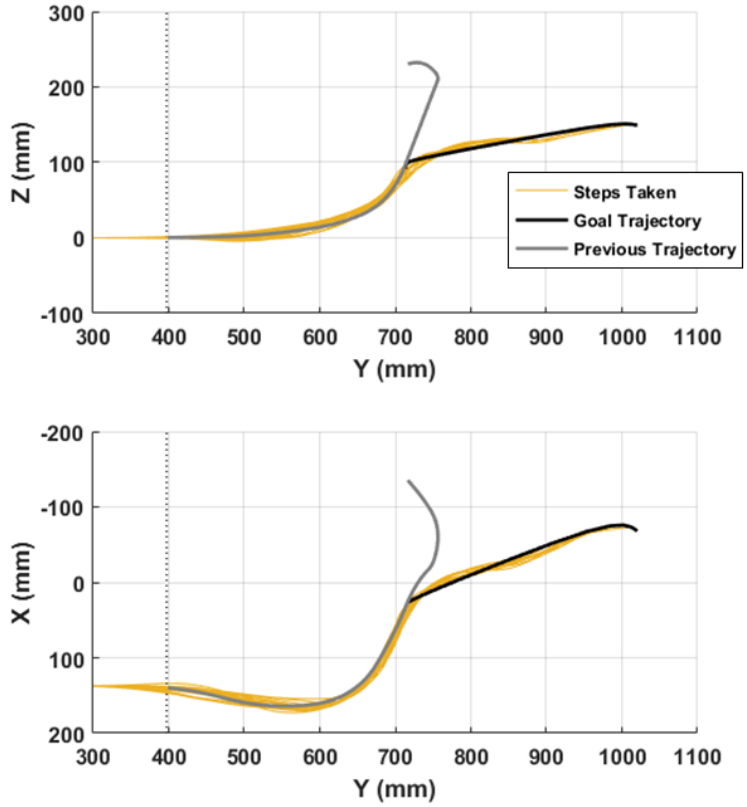


Figure 7-17: Poses used to navigate the Branching action

7.6.1.4 *"GoHome" back to initial pose*

The final action is to return the simulated robot to the initial position. Calling the "GoHome" mode again, a new retraction path is calculated that takes into account the branching. The resulting deviations are displayed in Figure 7-18.

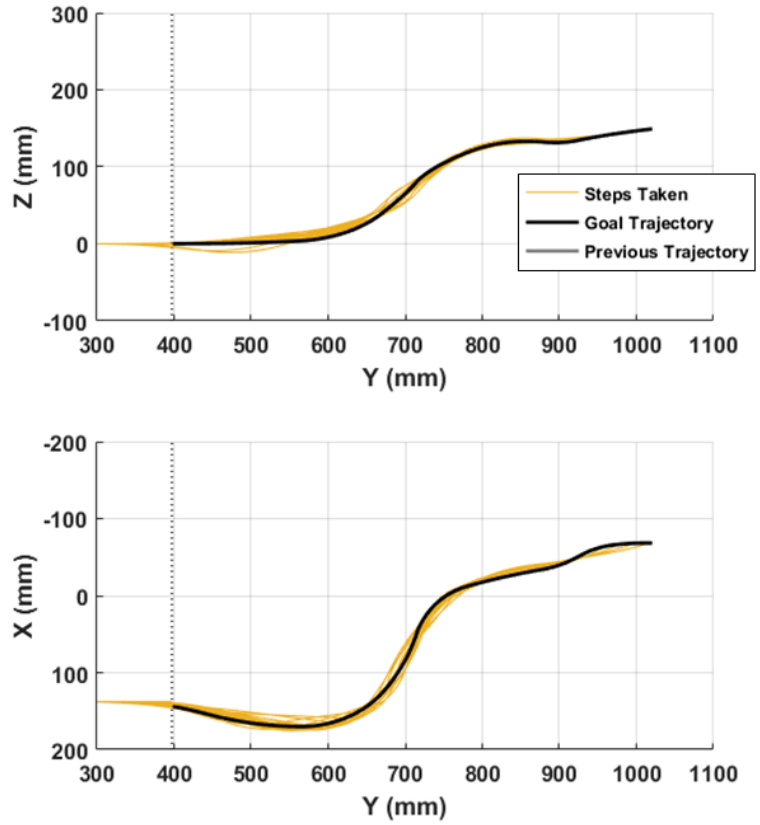


Figure 7-18: Combined overlay of poses for the final action of the simulation

7.6.1.5 *Performance results*

Using the same criteria as the Tip Following presented in Chapter 5, the performance is summarised by the graph in Figure 7-19. Each line represents an action taken in this simulation, and the dashed line of the same colour represents its average value. Note that different runs have different lengths, hence some lines stop earlier than others.

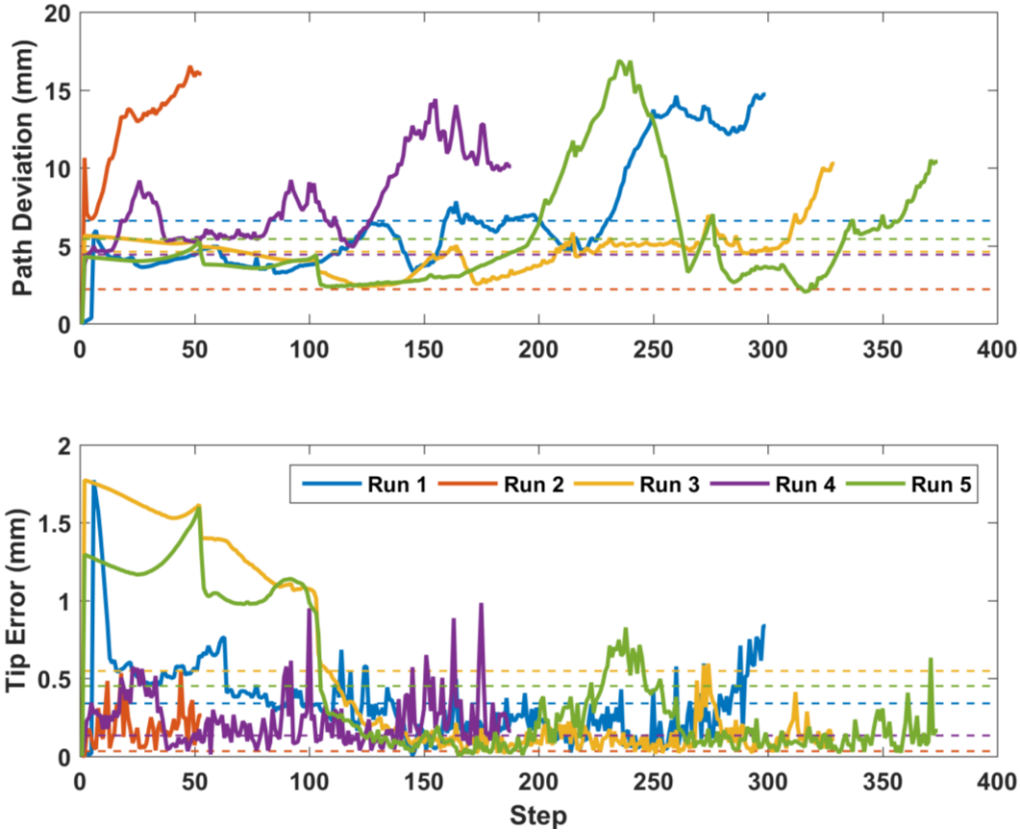


Figure 7-19: Summary of the Path Deviation and Tip Errors generated through the simulation

Firstly all results are below 20mm and 2mm for the Path Deviation and Tip Error which is within the acceptable margins. If it was not for a few errors at the tip, the performance would be akin to the basic Tip Following without the extra modes. The two main errors by Run 3 and Run 5 are due to these being "GoHome" actions. Since this is calculated from the base out, the desired path does not align perfectly with the initial point. Similar spikes are expected when combined with the coiling action.

The spike in the start of Run 1 is caused by the immediate bending in the path, when all the sections involved in the Tip Following are directly on the initial pose. Thus there is little interest for the optimisation to move from this starting condition. This is collaborated by the miniscule path deviation for a few steps shown on the top graph.

7.6.2 Tip Following through Engine Model for Defect Inspection

As specified in the Introduction, the main scenario being considered for this prototype is the inspection and repair of a gas turbine engine, without removing it from the plane. For this the arm has to reach from the front fan to IP1 and IP2, as indicated on Figure 7-20, which are the first two stages of the intermediary pressure compressor. Three defects have been added to the blades to be inspected, as numbered in the figure. Each trial to reach a defect was performed independently using the Endoscope mode to allow the best chance of reaching the target.



Figure 7-20: Snake Arm in-situ of 1:1 scale model of a gas turbine engine.

Inspection is performed using an off-the-shelf Machida videoscope. This can be inserted into the robot by means of the 8mm working channel along the entire length of the arm. The images in Figure 7-21 show the camera view of each defect, and the relative distance for the picture of defect 1. Pictures of the robot in-situ for defect 2 and 3 are presented in the appendix, but are difficult to interpret since the geometry of the environment makes photography difficult; the same reasons this project exists.

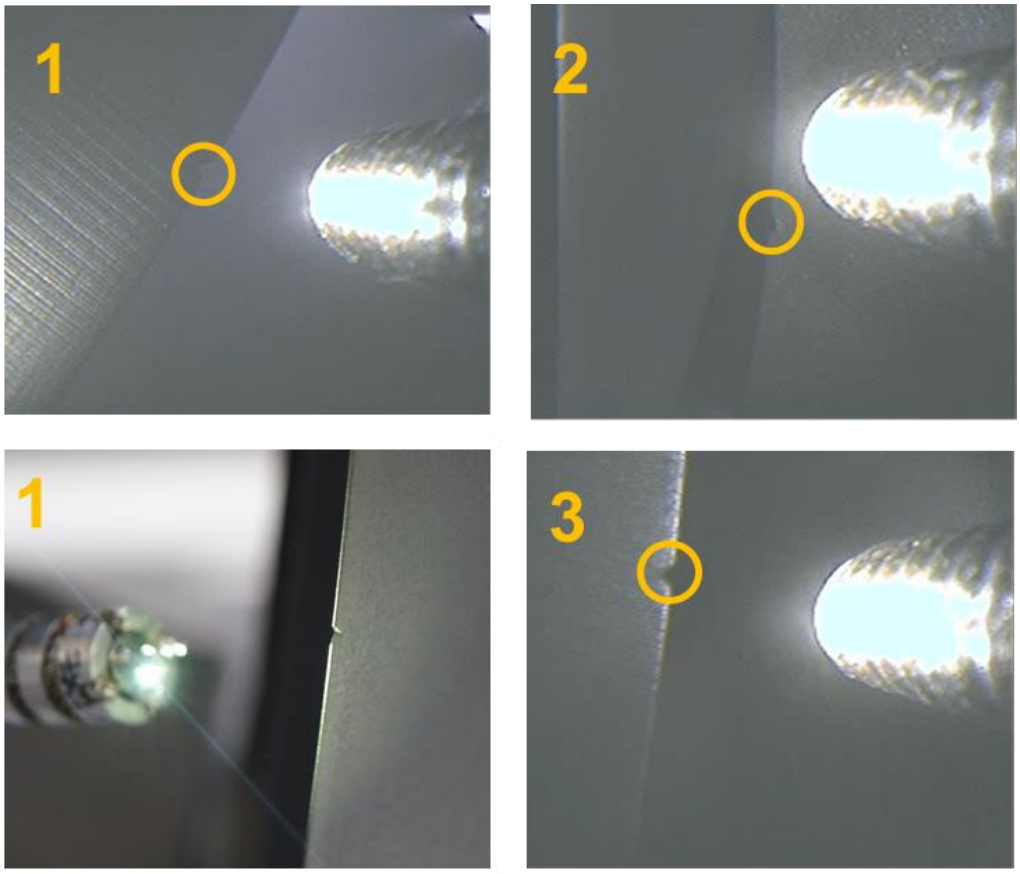


Figure 7-21: End-effector point-of-view of the three defects.

Once operations in the engine where completed, the Snake Arm was retreated using "GoHome". The timeseries in Figure 7-22 shows the body being withdrawn from Defect 1 smoothly.

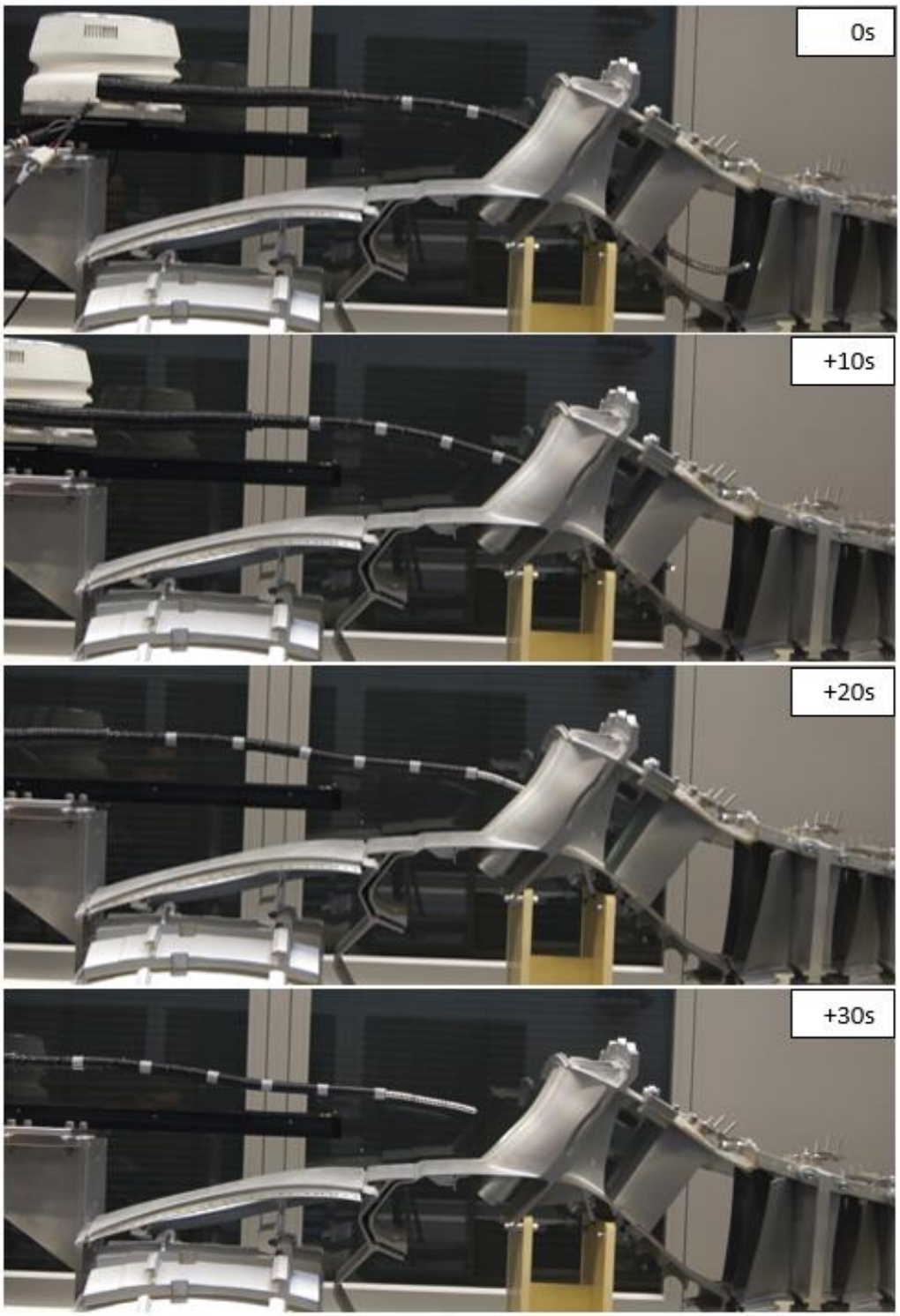


Figure 7-22: Timeseries of the GoHome action from Defect 1

7.6.3 Inspection of RPV head co-operating with walking hexapod

The other inspection scenario proposed for the Final Prototype is a combined function with the Walking Hexapod. Mounting the robot on top, the Hexapod can carry it into situations where it is ideal to avoid human interaction to setup the hardware. In this case it is the Reactor Pressure Vessel (RPV) head of a civil nuclear power plant. The part can be removed and placed in another room on an inspection stand, but the room still has an increased level of radiation requiring protection and red tape. Alternately, the Hexapod can walk under the head, and the Snake Arm is used to inspect the weld seams and the oxidisation on the ports.

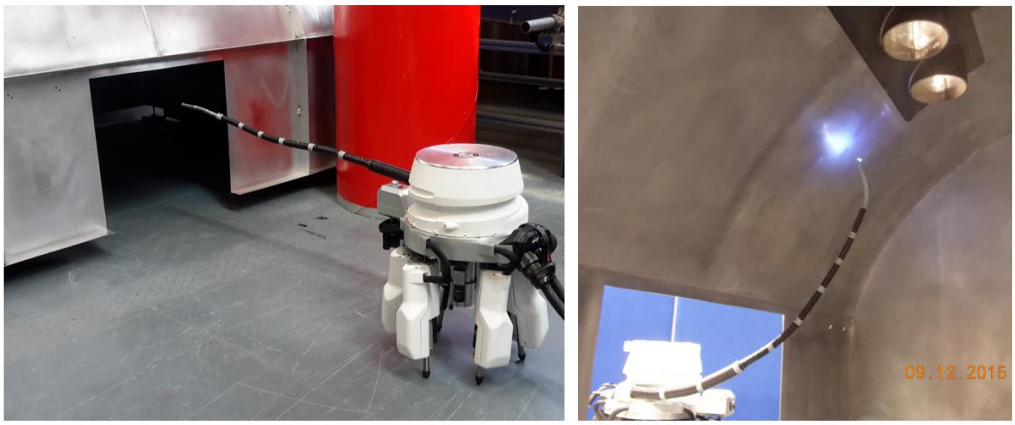


Figure 7-23: Snake Arm mounted on top of Walking Hex as it enters the mock-up environment

For the Snake Arm, there are no issues in the process and the poses required have been tested thoroughly with the Hexapod in the loop. In terms of the network diagram, Figure 7-2, the Hexapod acts as a network switch between the sbRIOs and the rest of the network, and therefore has no effect. The Walking Hexapod has been navigated into the environment and Snake Arm used to look at the top, as shown in Figure 7-23. Unfortunately the camera within the end-effector broke and no images could be retrieved.

7.6.4 Machining Trials

The final goal for the engine demo is to be able to repair the foreign object damage found within the compressor. The two factors that are required for this are the machining of the shape within the engine and machining the materials found within a jet engine. For the first task the robot was navigated to into the engine a set normal to the blade, slightly off-set from the edge. From here the machining mode was engaged. The results are shown in the images of Figure 7-24.

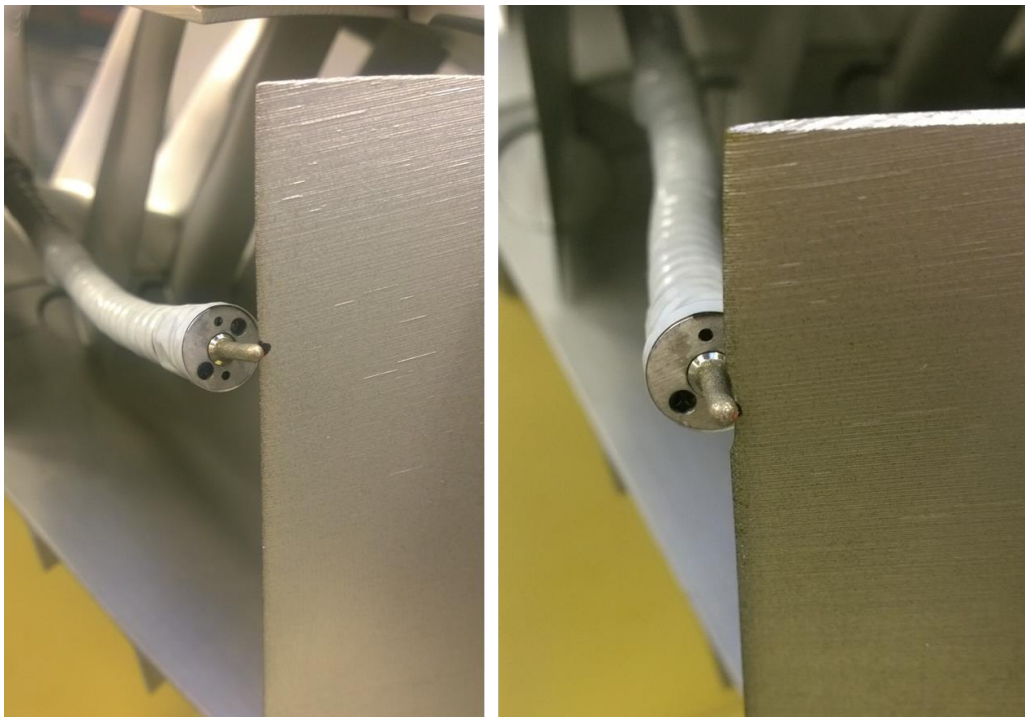


Figure 7-24: Before and after of a machining operation on the leading edge of IP1

As mentioned, modern engines are rarely made from 3D printed plastic parts. Instead they use titanium. Therefore the machining needs to be demonstrated on a realistic part to proof the operation can be completed on a live engine. A rough experiment was performed on an old blade using the three section prototype. The image in Figure 7-25 shows the process in action. A trial with the final prototype is planned on a serviceable engine in January 2016, as it was delayed by problems with the end-effector.



Figure 7-25: Machining on titanium blade, mid-process

7.7 Chapter Summary

This chapter has covered the integration of all the programs, both entailed in this thesis and of third-parties, to create the complete final prototype. The connections between the hardware and the communication map presents the inter-dependencies and how the system is laid out on a broader scale.

Through the careful planning of both Tip Following and Feed-In algorithms, a simple method has be developed to combine the two processes. In the same scheme as the individual techniques, this can be re-implemented for alternative feed-in approaches. Unfortunately this feature had to be used to integrate a linear stage, as mechanical errors made the system unreliable and led to internal damage.

Integration of other parts of the system has been completed. For the High Level Control this involved parsing JSON strings, while the Machining mode could use the import features already built-in to the GUI of the final prototype. The HLC allows the Tip Following to be commanded based on a path planned through a simulated representation of the environment. Through the addition of these features, an extra set of orders was created to extend the abilities of the Tip Following.

These new orders were tested, alongside the High Level Control, in simulation to show if their inclusion has had any effect on the performance. From the results it is clear that the combined algorithm is working, however it has demonstrated that the path planning and endoscope mode are both able to create undesirable routes. Nevertheless, the deviations and the tip errors were below 20mm and 2mm respectively, well within the requirements of the project.

Physical trials have been performed on both the gas turbine engine model and the RPV head mock-up. The Snake Arm has been able to reach all the defects specified on IP1 and IP2 within the engine for inspection to the requirements of the end-user. Afterwards, with a bespoke end-effector, the robot entered the engine again to perform a machining operation on the leading edge of IP1. For the RPV head, the Snake Arm has been demonstrated to work while mounted on-board the WalkingHex. The hexapod has then entered the scenario and the Snake Arm proved it can inspect the ports. With these experiments, the system has delivered the goals of the project to an acceptable level.

8 Conclusions

This project set out to create a generic robot capable of in-situ repair in restricted environments. Through the completion of the planned demonstrations from the initial brief the system has been proven to work within these specific scenarios and validated its use in a broad range of situations. This was achieved by the successful completion of the major aims and academic challenges: control of a high number of degrees of freedom, pose navigation through complex environments, active feed-in from a coiled position, and combining the elements into a complete package.

The successes and achievements from this research has led to a multitude of new projects starting that are using the knowledge gained and lessons learned.

8.1 Low-Level Control

The first main objective was to create the initial low-level system. From the literature survey it was decided that cable actuation would be the most suitable method for this structure given the scale and stiffness required. This was implemented with a segregated arm design and helical spools. A position control loop was developed which could operate every motor simultaneously within the limits of the hardware while maintaining a fast response.

The resulting system became the groundwork for all other work to be implemented. The accuracy and repeatability are on par with the anticipated performance, while the speed of movements exceeded expectations.

8.2 Navigation

The next objective and first academic challenge for the project was to develop an algorithm that could manage the configurations of the Snake Arm such that is could follow a path for navigation. This was set up such that the same techniques could be

used by similar robots of different designs and kinematics. By using an optimisation routine, solutions are found that can minimise the path deviations and maintain the tip advancement.

Through simulations and physical trials the algorithm has been demonstrated to work within the demands of the planned scenario. At times when the results are outside the normal operations, undesirable path features have been seen to be the cause.

8.3 Coiling Process

This leads to the next academic challenge, the ability to feed the Snake Arm from a rotating drum. Firstly the coiled profile was found and by analysing the different uncoiling scenarios, the best suited approach was chosen to implement. This was broken down into the required stages. Another optimisation routine was created to solve the final arrangement, and again so different hardware designs could use similar techniques. From this a function was created to take a desired base advancement and produce the coiled state with the parameters at the handover point for the Tip Following.

Testing this function showed that the computational time would have a negligible effect on the Tip Following and that the positional accuracy was within the tolerance set for the handover point.

8.4 Final Prototype

By the combination of the work presented within and that of third-parties, a final prototype has been realised that can perform the deliverables required for the project. Tip Following tasks can be created from the desired end pose, target in CAD environment, or on-the-fly using an endoscopic mode. Once at the objective, machining operations can be performed in-situ. This system works on its own, or

mounted on the walking hexapod. Unfortunately, mechanical errors led to the coiling not being implemented for the closing demonstrations.

8.5 Future Work

During the development of the prototypes for this research several decisions were made that cut potential ideas from expansion, improvements to the current infrastructure were conceptualised, or new avenues for exploration were discovered. In this section a brief description for the activities that are being considered for further progress and study.

8.5.1 Mechanical & Control

- **Fix or categorise mechanical errors** – It was discovered that there is a mechanical backlash in the cable spool system. This is caused by the spooling not behaving as intended. It is planned to fix the actuation design or at least categorise it. This would make improvements to the kinematic behaviour allowing for greater functionality and more specifically the coiling.
- **Headless operation** – The next step for the product is to work without an attached Windows PC. This will be done by relocating the majority of the low-level code on-board the sbRIOs and setting the system up to run headless. Once achieved the product will be a single self-contained unit.

8.5.2 Tip Following

- **Alternate design models -** As stated in section 5.2 the current Tip Following algorithm has been planned based on one set of geometric kinematics. It is theorised that the algorithm will continue to work for alternative models of

continuum arms or for entirely different behaviours. An extreme example would be navigating an Active Cannula; widely different behaviour but same goal.

- **Dealing with fixed or rigid sections –** During the development there was at one stage going to be a motor mounted on the end-effector, rather than driving the spindle with a torque cable. This led to a proof-of-concept that the current Tip Following algorithm is robust to a rigid section of fixed length being added to the tip. Extending this would be interesting to find what limitations rigid sections can add, or how this knowledge can be adapted for retrieval in error states.

8.5.3 Feed-In

- **Implementation of alternate uncoiling methods** – Due to time constraints, only the most critical uncoiling method was developed. To demonstrate the full potential of the completed prototype it would be beneficial to develop at least the vertical and 45° active-drum concepts
- **Adaptive Feed-In** – From the development it was clear that the lateral errors would cause the one bend solutions to become S-bends as well. With this being the case, a unified uncoiling strategy could be proposed; therefore allowing any tangential trajectory to be proposed.

8.5.4 Combined

- **Coiling and Tip Following –** With the current aim to improve the mechanical behaviour, the next step is to combine the Coiling and Tip Following algorithm. It would be a tremendous feat to be able to demonstrate this on a physical system, working in-situ

- **Enhanced path planning** – The current path planning provided by the high-level only considers the bare necessities to allow the Tip Following to operate. In an ideal solution the planner would optimise the path after a route is found to have the most clearance, large smooth curves and minimise the number of direction changes. This would greatly improve the performance of the Tip Following.

9 Bibliography

[1] *The Jet Engine*, 5th ed.: Rolls-Royce plc, 1996.

[2] K. Pathak, J. Franch, and S. K. Agrawal, "Velocity and position control of a wheeled inverted pendulum by partial feedback linearization," *Robotics, IEEE Transactions on,* vol. 21, pp. 505-513, 2005.

[3] A. B. Katake, A. D. Mahindrakar, and R. N. Banavar, "Study of underactuated mechanisms in the presence of holonomic constraints: the constrained Acrobot," in *Industrial Technology 2000. Proceedings of IEEE International Conference on*, 2000, pp. 703-706 vol.2.

[4] *KUKA Website*. Available: http://www.kuka-robotics.com/en/

[5] *Segway Webpage*. Available: http://www.segway.com/

[6] M. Szczys. (31/12/15). *Mantis9 PCB Mill*. Available: http://hackaday.com/2012/06/13/mantis9-pcb-mill/

[7] "ST Robotics Website."

[8] G. S. Chirikjian and J. W. Burdick, "Hyper-redundant robot mechanisms and their applications," in *Intelligent Robots and Systems '91. 'Intelligence for Mechanical Systems, Proceedings IROS '91. IEEE/RSJ International Workshop on*, 1991, pp. 185-190 vol.1.

[9] W. McMahan, V. Chitrakaran, M. Csencsits, D. Dawson, I. D. Walker, B. A. Jones, M. Pritts, D. Dienno, M. Grissom, and C. D. Rahn, "Field trials and testing of the OctArm continuum manipulator," in *Robotics and Automation, 2006. ICRA 2006. Proceedings 2006 IEEE International Conference on*, 2006, pp. 2336-2341.

[10] *OC Robotics Website*. Available: http://www.ocrobotics.com/

[11] M. W. Hannan and I. D. Walker, "The `elephant trunk' manipulator, design and implementation," in *Advanced Intelligent Mechatronics, 2001. Proceedings. 2001 IEEE/ASME International Conference on*, 2001, pp. 14-19 vol.1.

[12] G. Immega and K. Antonelli, "The KSI tentacle manipulator," in *Robotics and Automation, 1995. Proceedings., 1995 IEEE International Conference on*, 1995, pp. 3149-3154 vol.3.

[13] R. Buckingham, "Snake arm robots," *Industrial Robot,* vol. 29, pp. 242-245, 2002.

[14] R. Colbaugh, K. Glass, and H. Seraji, "Control Of A Serpentine Robot For Inspection Tasks," 1994.

[15] H. Ohno and S. Hirose, "Design of slim slime robot and its gait of locomotion," in *Intelligent Robots and Systems, 2001. Proceedings. 2001 IEEE/RSJ International Conference on*, 2001, pp. 707-715 vol.2.

[16] (31/12/2015). *Modular Snake - Biorobotics Laboratory at Carnegie Mellon*. Available: http://biorobotics.ri.cmu.edu/projects/modsnake/index.html

[17] HiBot. (31/12/2015). *ACM-R5H* Available: http://www.hibot.co.jp/en/products/robots_1/acm-r5h_33

[18] E. Paljug, T. Ohm, and S. Hayati, "The JPL Serpentine Robot: a 12-DOF system for inspection," in *Robotics and Automation, 1995. Proceedings., 1995 IEEE International Conference on*, 1995, pp. 3143-3148 vol.3.

[19] A. A. Maciejewski and C. A. Klein, "Obstacle avoidance for kinematically redundant manipulators in dynamically varying environments," *The international journal of robotics research,* vol. 4, pp. 109-117, 1985.

[20] Y. Hyun-Soo and Y. Byung-Ju, "A 4-DOF flexible continuum robot using a spring backbone," in *Mechatronics and Automation, 2009. ICMA 2009. International Conference on*, 2009, pp. 1249-1254.

[21] S. Neppalli and B. A. Jones, "Design, construction, and analysis of a continuum robot," in *Intelligent Robots and Systems, 2007. IROS 2007. IEEE/RSJ International Conference on*, 2007, pp. 1503-1507.

[22] B. A. Jones, R. L. Gray, and K. Turlapati, "Three dimensional statics for continuum robotics," in *Intelligent Robots and Systems, 2009. IROS 2009. IEEE/RSJ International Conference on*, 2009, pp. 2659-2664.

[23] A. Bajo and N. Simaan, "Finding lost wrenches: Using continuum robots for contact detection and estimation of contact location," in *Robotics and Automation (ICRA), 2010 IEEE International Conference on*, 2010, pp. 3666-3673.

[24] H. Haiyan, W. Pengfei, Z. Bo, L. Mantian, and S. Lining, "Design of a novel snake-like robotic colonoscope," in *Robotics and Biomimetics (ROBIO), 2009 IEEE International Conference on*, 2009, pp. 1957-1961.

[25] C. Q. Li and C. D. Rahn, "Design of continuous backbone, cable-driven robots," *Journal of Mechanical Design,* vol. 124, pp. 265-271, Jun 2002.

[26] G. Z. Lum, S. K. Mustafa, H. R. Lim, W. B. Lim, G. Yang, and S. H. Yeo, "Design and motion control of a cable-driven dexterous robotic arm," in *Sustainable Utilization and Development in Engineering and Technology (STUDENT), 2010 IEEE Conference on*, 2010, pp. 106-111.

[27] V. C. Anderson, "Tensor Arm Manipulator," US Patent 3,497,083, 1970.

[28] W. Birchard, "Digital Linear Actuator," US Patent 4,751,821, 1988.

[29] M. Cianchetti, A. Arienti, M. Follador, B. Mazzolai, P. Dario, and C. Laschi, "Design concept and validation of a robotic arm inspired by the octopus," *Materials Science and Engineering: C,* vol. 31, pp. 1230-1239, 2011.

[30] W. McMahan and I. D. Walker, "Octopus-inspired grasp-synergies for continuum manipulators," in *Robotics and Biomimetics, 2008. ROBIO 2008. IEEE International Conference on*, 2009, pp. 945-950.

[31] K. Suzumori, S. Iikura, and H. Tanaka, "Development of flexible microactuator and its applications to robotic mechanisms," in *Robotics and Automation, 1991. Proceedings., 1991 IEEE International Conference on*, 1991, pp. 1622-1627 vol.2.

[32] *FESTO Website*. Available: www.festo.com

[33] D. Trivedi, D. Dienno, and C. D. Rahn, "Optimal, model-based design of soft robotic manipulators," *Journal of Mechanical Design,* vol. 130, Sep 2008.

[34] M. Ivanescu, N. Bizdoaca, and D. Pana, "Dynamic control for a tentacle manipulator with SMA actuators," in *Robotics and Automation, 2003. Proceedings. ICRA '03. IEEE International Conference on*, 2003, pp. 2079-2084 vol.2.

[35] P. Senthilkumar, G. N. Dayananda, M. Umapathy, and V. Shankar, "Experimental evaluation of a shape memory alloy wire actuator with a modulated adaptive controller for position control," *Smart Materials & Structures,* vol. 21, Jan 2012.

[36] R. J. Webster, A. M. Okamura, and N. J. Cowan, "Toward Active Cannulas: Miniature Snake-Like Surgical Robots," in *Intelligent Robots and Systems, 2006 IEEE/RSJ International Conference on*, 2006, pp. 2857-2863.

[37] Y. Haga, M. Mizushima, T. Matsunaga, and M. Esashi, "Medical and welfare applications of shape memory alloy microcoil actuators," *Smart Materials & Structures,* vol. 14, pp. S266-S272, Oct 2005.

[38] L. A. Lyons, R. J. Webster, and R. Alterovitz, "Motion planning for active cannulas," in *Intelligent Robots and Systems, 2009. IROS 2009. IEEE/RSJ International Conference on*, 2009, pp. 801-806.

[39] G. S. Chirikjian and J. W. Burdick, "Design and experiments with a 30 DOF robot," in *Robotics and Automation, 1993. Proceedings., 1993 IEEE International Conference on*, 1993, pp. 113-119 vol.3.

[40] G. S. Chirikjian and J. W. Burdick, "Kinematically optimal hyper-redundant manipulator configurations," in *Robotics and Automation, 1992. Proceedings., 1992 IEEE International Conference on*, 1992, pp. 415-420 vol.1.

[41] G. S. Chirikjian and J. W. Burdick, "A modal approach to hyper-redundant manipulator kinematics," *IEEE Transactions on Robotics and Automation,* vol. 10, pp. 343-354, 1994.

[42] I. Ebert-Uphoff and G. S. Chirikjian, "Inverse kinematics of discretely actuated hyper-redundant manipulators using workspace densities," in *Robotics and Automation, 1996. Proceedings., 1996 IEEE International Conference on*, 1996, pp. 139-145 vol.1.

[43] G. S. Chirikjian, "A general numerical method for hyper-redundant manipulator inverse kinematics," in *Robotics and Automation, 1993. Proceedings., 1993 IEEE International Conference on*, 1993, pp. 107-112 vol.3.

[44] G. S. Chirikjian, "A continuum approach to hyper-redundant manipulator dynamics," in *Intelligent Robots and Systems '93, IROS '93. Proceedings of the 1993 IEEE/RSJ International Conference on*, 1993, pp. 1059-1066 vol.2.

[45] G. S. Chirikjian and J. W. Burdick, "An obstacle avoidance algorithm for hyper-redundant manipulators," in *Robotics and Automation, 1990. Proceedings., 1990 IEEE International Conference on*, 1990, pp. 625-631 vol.1.

[46] R. J. Webster, J. S. Kim, N. J. Cowan, G. S. Chirikjian, and A. M. Okamura, "Nonholonomic modeling of needle steering," *International Journal of Robotics Research,* vol. 25, pp. 509-525, May-Jun 2006.

[47] G. S. Chirikjian and J. W. Burdick, "Kinematics of hyper-redundant robot locomotion with applications to grasping," in *Robotics and Automation, 1991. Proceedings., 1991 IEEE International Conference on*, 1991, pp. 720-725 vol.1.

[48] G. S. Chirikjian and J. W. Burdick, "Parallel formulation of the inverse kinematics of modular hyper-redundant manipulators," in *Robotics and Automation, 1991. Proceedings., 1991 IEEE International Conference on*, 1991, pp. 708-713 vol.1.

[49] G. S. Chirikjian, "General methods for computing hyper-redundant manipulator inverse kinematics," in *Intelligent Robots and Systems '93, IROS '93. Proceedings of the 1993 IEEE/RSJ International Conference on*, 1993, pp. 1067-1073 vol.2.

[50] I. D. Walker and M. W. Hannan, "A novel `elephant's trunk' robot," in *Advanced Intelligent Mechatronics, 1999. Proceedings. 1999 IEEE/ASME International Conference on*, 1999, pp. 410-415.

[51] M. W. Hannan and I. D. Walker, "Analysis and initial experiments for a novel elephant's trunk robot," in *Intelligent Robots and Systems, 2000. (IROS 2000). Proceedings. 2000 IEEE/RSJ International Conference on*, 2000, pp. 330-337 vol.1.

[52] W. McMahan, B. A. Jones, and I. D. Walker, "Design and implementation of a multi-section continuum robot: Air-Octor," in *Intelligent Robots and Systems, 2005. (IROS 2005). 2005 IEEE/RSJ International Conference on*, 2005, pp. 2578-2585.

[53] D. C. Rucker, B. A. Jones, and R. J. Webster, "A Geometrically Exact Model for Externally Loaded Concentric-Tube Continuum Robots," *Robotics, IEEE Transactions on,* vol. 26, pp. 769-780, 2010.

[54] R. J. Webster and B. A. Jones, "Design and Kinematic Modeling of Constant Curvature Continuum Robots: A Review," *International Journal of Robotics Research,* vol. 29, pp. 1661-1683, Nov 2010.

[55] B. A. Jones, W. McMahan, and I. D. Walker, "Practical kinematics for real-time implementation of continuum robots," in *Robotics and Automation, 2006. ICRA 2006. Proceedings 2006 IEEE International Conference on*, 2006, pp. 1840-1847.

[56] B. A. Jones and I. D. Walker, "Kinematics for multisection continuum robots," *Robotics, IEEE Transactions on,* vol. 22, pp. 43-55, 2006.

[57] I. A. Gravagne and I. D. Walker, "On the kinematics of remotely-actuated continuum robots," in *Robotics and Automation, 2000. Proceedings. ICRA '00. IEEE International Conference on*, 2000, pp. 2544-2550 vol.3.

[58] M. W. Hannan and I. D. Walker, "Kinematics and the implementation of an elephant's trunk manipulator and other continuum style robots," *Journal of Robotic Systems,* vol. 20, pp. 45-63, Feb 2003.

[59] E. Tatlicioglu, I. D. Walker, and D. M. Dawson, "Dynamic Modelling for Planar Extensible Continuum Robot Manipulators," in *Robotics and Automation, 2007 IEEE International Conference on*, 2007, pp. 1357-1362.

[60] S. Neppalli, M. A. Csencsits, B. A. Jones, and I. D. Walker, "Closed-Form Inverse Kinematics for Continuum Manipulators," *Advanced Robotics,* vol. 23, pp. 2077-2091, 2009.

[61] I. A. Gravagne and I. D. Walker, "Uniform regulation of a multi-section continuum manipulator," in *Robotics and Automation, 2002. Proceedings. ICRA '02. IEEE International Conference on*, 2002, pp. 1519-1524 vol.2.

[62] I. A. Gravagne, C. D. Rahn, and I. D. Walker, "Large deflection dynamics and control for planar continuum robots," *Mechatronics, IEEE/ASME Transactions on,* vol. 8, pp. 299-307, 2003.

[63] M. Hannan and I. Walker, "Vision based shape estimation for continuum robots," in *Robotics and Automation, 2003. Proceedings. ICRA '03. IEEE International Conference on*, 2003, pp. 3449-3454 vol.3.

[64] M. W. Hannan and I. D. Walker, "Real-time shape estimation for continuum robots using vision," *Robotica,* vol. 23, pp. 645-651, Sep-Oct 2005.

[65] M. Csencsits, B. A. Jones, W. McMahan, V. Iyengar, and I. D. Walker, "User interfaces for continuum robot arms," in *Intelligent Robots and Systems, 2005. (IROS 2005). 2005 IEEE/RSJ International Conference on*, 2005, pp. 3123-3130.

[66] S. Neppalli, M. A. Csencsits, B. A. Jones, and I. Walker, "A geometrical approach to inverse kinematics for continuum manipulators," in *Intelligent Robots and Systems, 2008. IROS 2008. IEEE/RSJ International Conference on*, 2008, pp. 3565-3570.

[67] R. Buckingham and A. Graham, "Snaking around in a nuclear jungle," *Industrial Robot-an International Journal,* vol. 32, pp. 120-127, 2005.

[68] R. Buckingham, "Link Assembly for a Snake Like Robot Arm," US Patent US 7543518 B2, 2009.

[69] R. Buckingham, "Articulating Arm for Positioning a Tool at a Location," US Patent US 7,171,279 B2, 2007.

[70] R. Buckingham, "System and Method for Controlling a Robotic Arm," US Patent, 2009.

[71] S. Battilotti and L. Lanari, "GLOBAL SET-POINT CONTROL VIA LINK POSITION MEASUREMENT FOR FLEXIBLE-JOINT ROBOTS," *Systems & Control Letters,* vol. 25, pp. 21-29, May 1995.

[72] M. Schlemmer, "On-line trajectory optimization for kinematically redundant robot-manipulators and avoidance of moving obstacles," in *Robotics and Automation, 1996. Proceedings., 1996 IEEE International Conference on*, 1996, pp. 474-479 vol.1.

[73] I. S. Godage, D. T. Branson, E. Guglielmino, and D. G. Caldwell, "Path planning for multisection continuum arms," in *Mechatronics and Automation (ICMA), 2012 International Conference on*, 2012, pp. 1208-1213.

[74] H. Choset and W. Henning, "A Follow-the-Leader Approach to Serpentine Robot Motion Planning," *Journal of Aerospace Engineering,* vol. 12, pp. 65-73, 1999.

[75] C. Yi, L. Jiahui, and I. W. Hunter, "Modular continuum robotic endoscope design and path planning," in *Robotics and Automation (ICRA), 2014 IEEE International Conference on*, 2014, pp. 5393-5400.

[76] A. C. Graham and R. O. Buckingham, "Robot with a deployment apparatus for an arm comprising a plurality of links," US8635928 B2, 2014.

[77] J. S. Mehling, M. A. Diftler, M. Chu, and M. Valvo, "A Minimally Invasive Tendril Robot for In-Space Inspection," in *Biomedical Robotics and Biomechatronics, 2006. BioRob 2006. The First IEEE/RAS-EMBS International Conference on*, 2006, pp. 690-695.

[78] X. Dong, M. Raffles, S. C. Guzman, D. Axinte, and J. Kell, "Design and analysis of a family of snake arm robots connected by compliant joints," *Mechanism and Machine Theory,* vol. 77, pp. 73-91, 2014.

[79] S. C. Jacobsen, H. Ko, E. K. Iversen, and C. C. Davis, "Antagonistic control of a tendon driven manipulator," in *Robotics and Automation, 1989. Proceedings., 1989 IEEE International Conference on*, 1989, pp. 1334-1339 vol.3.

[80] N. S. Nise, "Improving Transient Response via Cascade Compensation," in *Control Systems Engineering, 5Th Ed, Isv*, ed: Wiley India Pvt. Limited, 2009, pp. 449-458.