ALMA MATER STUDIORUM · UNIVERSITÀ DI BOLOGNA

SCUOLA DI INGEGNERIA E ARCHITETTURA

DIPARTIMENTO di
INGEGNERIA DELL'ENERGIA ELETTRICA E DELL'INFORMAZIONE
"Guglielmo Marconi"
DEI

CORSO DI LAUREA MAGISTRALE
IN AUTOMATION ENGINEERING

Tesi di Laurea in Industrial Robotics M

SHAPE-BASED COMPLIANCE CONTROL FOR SNAKE ROBOTS

Relatore:
Chiar.mo Prof.
Claudio Melchiorri

Correlatori:
Chiar.mo Prof.
Howie Choset

Chiar.mo Prof.
Matthew J. Travers

Candidato:
Francesco Ruscelli

II Sessione
Anno Accademico 2015/2016

Abstract

I serpenti robot sono una classe di meccanismi iper-ridondanti che appartiene alla robotica modulare. Grazie alla loro forma snella ed allungata e all'alto grado di ridondanza possono muoversi in ambienti complessi con elevata agilità. L'abilità di spostarsi, manipolare e adattarsi efficientemente ad una grande varietà di terreni li rende ideali per diverse applicazioni, come ad esempio attività di ricerca e soccorso, ispezione o ricognizione. I robot serpenti si muovono nello spazio modificando la propria forma, senza necessità di ulteriori dispositivi quali ruote od arti. Tali deformazioni, che consistono in movimenti ondulatori ciclici che generano uno spostamento dell'intero meccanismo, vengono definiti andature. La maggior parte di esse sono ispirate al mondo naturale, come lo strisciamento, il movimento laterale o il movimento a concertina, mentre altre sono create per applicazioni specifiche, come il rotolamento o l'arrampicamento.

Un serpente robot con molti gradi di libertà deve essere capace di coordinare i propri giunti e reagire ad ostacoli in tempo reale per riuscire a muoversi efficacemente in ambienti complessi o non strutturati. Inoltre, aumentare la semplicità e ridurre il numero di controllori necessari alla locomozione alleggerisce una struttura di controllo che potrebbe richiedere complessità per ulteriori attività specifiche. L'obiettivo di questa tesi è ottenere un comportamento autonomo cedevole che si adatti alla conformazione dell'ambiente in cui il robot si sta spostando, accrescendone le capacità di locomozione attraverso terreni irregolari. Sfruttando la cedevolezza intrinseca del serpente robot utilizzato in questo lavoro, il *SEA Snake*, e utilizzando un controllo che combina cedevolezza attiva ad una struttura di coordinazione che ammette una decentralizzazione variabile del robot, si dimostra come tre andature possano essere modificate per ottenere una locomozione efficiente in ambienti complessi non noti a priori o non modellabili.

Abstract

Snake robots are a class of hyper-redundant mechanisms that belong to the branch of modular robotics. Their long and slender shape, along with their high redundant kinematics, allows them to navigate complex environments with a high degree of dexterity. The ability to locomote, manipulate and adapt efficiently to a wide range of terrains make them ideal for a diverse set of tasks such as urban search and rescue, inspection and reconnaissance. Snake robots generate net displacement by internal mechanism deformation, hence wheels, legs or tracks are not necessary. These cyclic shape changes are called gaits. Most of them are bio-inspired, like slithering, sidewinding or concertina, while others are crafted for specific tasks, like rolling or climbing. Biological snakes own unique shape-changing capabilities together with many different exteroceptive sensors, a superior flexibility and a refined proprioception. For a snake-like robot with many degrees-of-freedom to locomote through complex unstructured spaces it must be able to sense the surroundings, coordinate its joints, and react to unknown features in real-time. Devising a compliant motion is part of the general problem of achieving autonomous compliant behaviours that allow a snake robot to adapt efficiently to changes in its environment. Transitional gaits should be able to overcome unknown environmental features without being overly complicated to do so. In fact, increasing the simplicity and reducing the quantity of controllers lighten a structure that may require complexity for precise ulterior tasks. Aim of this work is furthering the locomotion capabilities of snake robots through uneven terrains, shortening the gap between biological and robotic snakes. Using the inherent mechanical compliance of the snake robot used in this work, the *SEA snake*, it is proposed a controller which combines virtual compliance together with an underlying coordination structure and a variable decentralization to augment three different gaits for hyper redundant mechanisms. It is shown how using this control technique the snake robot is able to locomote through environments without any a priori knowledge of them.

Acknowledgements

First of all, I would like to thank my parents, who supported me in every decision I made, always putting me before them. They've been an unshakable help all these months, a precious resource and an irreplaceable encouragement. Thank you for guiding me through life. I would like to express my gratitude to Claudio Melchiorri and Howie Choset, who gave me the opportunity to work on this thesis and to go through such an important experience, which I believe has been of fundamental importance for me, both on a personal and a professional level. A huge thank you goes to Matthew J. Travers, an outstanding researcher and a brilliant human being who assisted me through these months with his cleverness and resourcefulness. Without his guidance this work would not have been possible. I would like to thank Julian W. and Guillaume S., as lab mate I worked with the most during these months. Their exceptional intelligence, their availability and talent inspired and motivated me. It was a pleasure and a privilege working with you. I take the opportunity to thank the Biorobotics lab, a wonderful environment full of skillful and enthusiastic people, and all my lab mates, in particular Ky, Brad, Angela, Glenn, Chaohui, Puru, and Alex.

A heartfelt thank you goes to all the friends I found in Pittsburgh, Alessandro, Dragan, Lucas, Arvind, Haewon, Tejas, Lair, Andrew, Sylwia, Lavinia, Marianna and many others, with whom I shared great memories and all my friends who bolstered me up every day even if they were so far on the other side of the ocean, in particular Federico G., Enrico C., Michele D., Davide C., Francesco C., Federico R., Matteo S. and Enrico D. .

Last but not least, I thank Elena as a friend, a lab, house and travel mate, a colleague, an outstanding human being who supported me in everything. She helped me in the darkest moments and was there in the happiest ones, pushing me in incredible experiences I'll never forget. I would never be where I am if you hadn't been there.

Contents

Acknowledgements II

1 Introduction 1

1.1 Snake robots 1

1.2 Compliance in mobile robots 4

2 Hardware and Software 7

2.1 SEA Snake 7

2.2 Software 12

3 Background and Related Work 17

3.1 Serpenoid curve and Parametrized Gaits 17

3.2 Shape functions 18

3.3 Admittance control 21

3.4 Shape-based compliance 22

3.5 Decentralization 24

4 Compliant Gaits 25

4.1 Slithering Gait Compliance 25

4.1.1 Introduction 25

4.1.2 Background 26

4.1.3 Wavelength compliance and variable centralization 26

4.1.4 Temporal phase compliance 31

4.1.5 Full controller 32

4.1.6 Experimental results 33

4.1.7 Discussion 35

4.2 Rolling Helix Gait Compliance 36

4.2.1 Introduction 36

4.2.2 Background 37

4.2.3 Radius Compliance 37

4.2.4 Variable Decentralization 39

4.2.5 Adaptive controller 40
4.2.6 Full controller 41
4.2.7 Experimental results and discussion 42
4.3 Sidewinding Gait Compliance 43
4.3.1 Introduction 43
4.3.2 Complementary filter and Virtual Chassis 45
4.3.3 Inertial turning 46
4.3.4 Shape-based anti-compliance 50
4.3.5 Decentralization 51
4.3.6 Full controller 52
4.3.7 Experimental comparison and results 53
4.3.8 Discussion 54

5 Conclusions 57

Bibliography 59

List of Figures

1.1 A photo of the Series Elastic Actuated Snake robot. (Source: [1]). . . . 2
2.1 A photo of a SEA module. (Source: [1]). 7
2.2 Scheme of the Series Elastic Actuation. (Source: [42]). 8
2.3 Cascaded Inner Torque PID 10
2.4 Cascaded Outer Torque PID 10
2.5 Cascaded Intermediate Torque PID 10
2.6 Block Diagram of the SEA Module electronics 11
3.1 A representation of a path in the Shape space 19
4.1 Snake robot experiencing external forces and corresponding plot of the shape force 27
4.2 An illustration of activation windows for the slithering gait 29
4.3 The snake robot compliantly adjusts its spatial frequency as it moves through an unknown environment 30
4.4 A demonstration of multiple independent activation windows 30
4.5 The response of the temporal phase Ω_t to an external forcing τ'_{ext} in a fully centralized shape-based controller 32
4.6 Comparison of shape-based and distributed-feedback CPG control methods for a snake robot traversing a irregularly spaced peg array . 34
4.7 Shape of the rolling gait for the *SEA Snake*. 38
4.8 Shape of the helix gait for the *SEA Snake*. 39
4.9 An illustration of activation windows for the rolling helix gait 40
4.10 Sequence of the *SEA snake* passing through a set of pegs. 45
4.11 Virtual Chassis for different gait of the snake robot. (Source: [53]). . 46
4.12 Potential landscape of the bistable dynamical system 47
4.13 Evolution of the heading and the phase in time 48
4.14 A sequence of the heading and the phase of the *SEA Snake* in the Potential Field. 49
4.15 Comparison of different controllers for a snake robot passing through a set of pegs. 54
4.16 Mean time before reversal for different controllers 55

Chapter 1

Introduction

1.1 Snake robots

Among many different kinds of robots, such as wheeled, legged or flying machines, the class of snake robots have gained particular attention thanks to their unique characteristics. Not only they represent a challenge for the bio-inspired robotics world, which is interested in the principles underlying the many different movements of biological snakes, but they proved to be extremely useful for many practical applications. Their long and slender body, combined with their hyper-redundant architecture, allow them to thread through tightly packed volumes or narrow spaces, while their shape-changing capabilities make them ideal for a wide variety of terrains. The snake's simple shape fully exploits its modular design, which consists of a series of identical actuated links chained together, plus one tail to connect the tether (that provides signals and power) and, if needed, one head, a different module built for specific tasks, e.g. manipulation or vision.

Such a shape allows the distribution of actuators and sensors along the snake body that, together with an effective coordination of its hyper-redundant links, enhances the interaction between mechanism and the environment and grants the ability to perform motions that go beyond the reach of conventional robots (e.g. legged or wheeled devices). Beside a versatile locomotion and a high maneuverability, these properties enable a direct manipulation of the environment around it. Snake robots are used for many practical applications, such as search and rescue in post-disaster scenarios, inside collapsed buildings or rubbles, searching for survivors or assessing the situation in narrow locations or crumbled areas unreachable by other devices. Another application these robot are suited for is inspection and repair in dangerous or inaccessible areas like industrial or nuclear power plants, which they can extensively navigate; locomoting on floor, climbing on poles, crawling though pipes and swimming in cooling system's pools. Snake robots have been used in archeology to

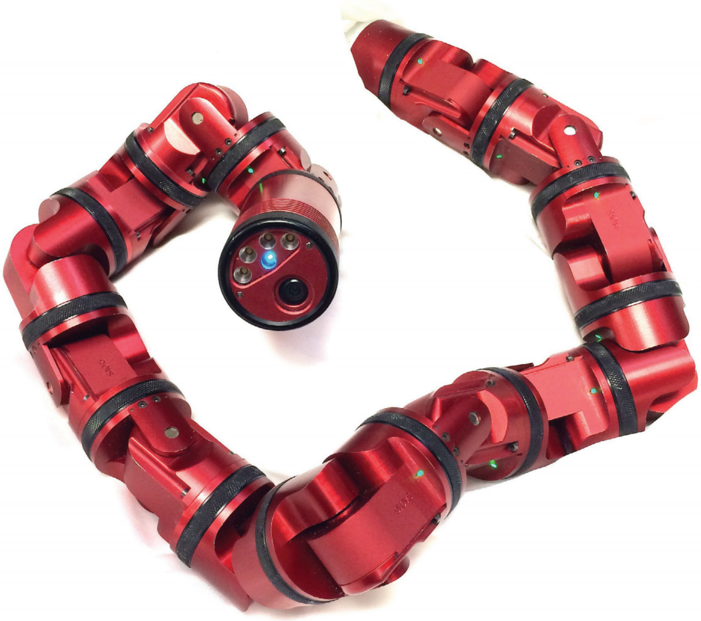

Figure 1.1: A photo of the Series Elastic Actuated Snake robot. (Source: [1]).

explore pyramids, since they can sidewind on sand (and many granular surfaces) and slither through narrow passageways. Scout and reconnaissance is yet another application of the snake robot. It can covertly access hidden positions, climb to vantage points and move through either urban or natural environments.

There is a significant amount of prior work in snake robot literature. Different versions of snake-like mechanisms were developed for various tasks, from manipulation to locomotion. Many of these snake-like robots use passive wheels to move, like in [2, 3], while others rely solely on their internal shape changes. These robots have been called in many ways, such as "tentacles" [4, 5], "tensor-arms" [6], "spines" [7], "elephant trunk" [8]. The widespread term "hyper-redundant robot", a long-established equivalent to "snake robot", was formerly introduced by Chirikjian and Burdick in [9]. The same authors defined a novel approach based on the backbone curve, effectively laying the foundation of kinematic theory for snake robots [10, 11, 12, 13]. Beside the backbone approach, many other strategies have been explored to control snake robots, such as follow-the-leader [14, 15] or CPG controllers [16, 17].

Hirose greatly contributed to the topic presenting a novel tool to describe the serpentine locomotion of snakes; he formulated the well-known serpenoid curve [18], that defines the control of a snake robot in terms of its backbone curve. Several researches were done to explore the many capabilities of these versatile mechanisms, such as obstacle avoidance [9], adaptive and obstacle aided locomotion [19, 20, 21], model-based approaches for locomotion and path planning [22, 23, 24, 25, 26], model-free controls [18, 27, 28, 15], manipulation and grasping [29, 30, 31, 32, 5, 33, 34], etc. An extensive survey on different techniques for snake robot locomotion and modeling is presented in [35]. The Biorobotics Lab at Carnegie Mellon University built many snake-like devices (e.g. Unified Snake, *SEA Snake*, etc.) and developed many techniques for estimation, hybrid control, motion and path planning for snake robots. In particular, the Biorobotics Lab team devised a way to control their snake robots based on a backbone approach. By modifying the Serpenoid Curve, which accounted only for a planar motion, and extending it to two orthogonal planes, lateral and dorsal, they achieved a powerful framework that allowed to define many 3D snake's motions. While many gaits are directly inspired by biological snakes, other are manufactured exploiting the versatility of this high redundant mechanism. Bio-inspired gaits:

- Linear Progression. A worm-like movement, it consists of a sine wave sent through the backbone of the snake on the dorsal plane. Though the gait it's not very efficient, it can be used for fitting into tight areas.

- Slithering. A common gait among the biological snakes, it consists of an undulatory motion used to propel the animal forward. Different modes of this gait can be found in nature, mainly depending on the snake species, the environment and the aspect of the movement that the snake is optimizing (such as speed, energy, compliance, etc.). The most common one is lateral undulation, in which lateral waves propagate along the snake body, from head to tail. The slithering gait used by snake robots mimics this motion, relying on a sinusoidal wave that travels along the lateral plane of the snake backbone to propel forward.

- Sidewinding. One of the most efficient transitional gait, it is used by biological snakes to move through various terrains. Differently from many others, this gait is fully independent from environmental features, relying only on internal shape changes to locomote, while incurring in minimal environmental friction. As a consequence, it is used mainly where ground does not provide the necessary reaction forces for other locomotion gaits, such as rocky or slippery terrains. Due to its stability, robustness and friction minimization, this gait is ideal for the displacement of snake robots through a wide variety of environments.

- Rolling. An efficient, reliable and simple way to move the snake, the mechanism is bent into a "C" while rolling sideways.

- Swimming. A sine wave is sent from head to tail of the snake robot to propel it forward in water.

- Pole climbing. A crafted gait that exploits the high number of degrees of freedom characteristic of the mechanism to climb on poles or similar structures. The snake robot coils up around a pole and rolls its body, effectively climbing up or down the obstacle.

- Pipe Crawling. Similar to pole climbing, the snake rolls inside the obstacle (or pipe, tube, conduit, etc.) while coiled up into a helix.

Snake robots, with their many internal degrees of freedom, are able to navigate through a wide range of environments with dexterity. They can coordinate their hyper-redundant links to perform motions that go beyond the ability of conventional robots, like wheeled or legged robots. However, having many degrees of freedom is advantageous only if the system is able to coordinate them efficiently to achieve the desired goals. A modular, hyper-redundant design poses non-trivial research problems both in the hardware design and control strategies. To this end, two claims are made for this work. First, the control architecture must be capable of coordinating all the DOFs in real time. Second, the used framework shouldn't be overly complex, lightening the structure, limiting the computational burden and reducing the number of exteroceptive sensors needed.

Beyond the specific tasks a snake robot is used for, the proposed work focus on the general ability to transit and reach desired locations. While in open spaces and even terrains an open-loop controller is enough for the snake to move efficiently, in unstructured, cluttered or unknown environments a feed-back control strategy is required.

Specifically, three gaits have been considered in this work and augmented: slithering, sidewinding and pole climbing.

1.2 Compliance in mobile robots

In mobile robot locomotion, two general categories exist to define a motion strategy for the mechanism: model-based and model-free controls. Model-based approaches are used when a model of the environment is available or straightforwardly obtainable. This method allows conventional planning and control techniques to be applied, but it strongly relies on the accuracy of the environmental model. This assumption is excessively limiting for unknown environments or too complex terrains to model. Model-free control techniques usually define feed forward joint-space kinematics that produce a defined locomotion gait. Knowledge of the environment is not necessary, but blind robots driven by kinematic-based gaits through complex terrain usually result in failures. A viable way to solve this problem in a model-free context is allowing a certain degree of flexibility in the system, such that a mobile robot that navigates though an unknown environment won't break or get stuck. Animals show this ability in any situation, complying their motion to terrains and obstacles.

The term compliance in the context of robot control was first introduced by Mason, in [36]. It was formerly described as a motion that "occurs when the manipulator position is constrained by the task geometry". Effectively, compliance represents one way to endow the system with the ability to interact with the environment when it constrains or impedes movement.

In short, compliance in locomotion is greatly advantageous when robots deal with any terrains other than flat and open ones, or when a model of the external features is not provided. Many compliance strategy have been devised for legged robot, wheeled robots or limbless mechanisms, mainly relying on compliance actuators.

Differently from stiff actuators, which won't allow any deviation from the commanded position, compliant actuators are not rigid, and applied external forces can move them from the equilibrium position reached, granting flexibility.

Compliance actuators are divided in two different families. Passive compliance actuators are built with elastic elements, such as springs or rubbery materials that can absorb shocks and store energy in the system. Active compliance actuators are achieved with control strategies such as admittance control, that virtually mimics the behaviour of a spring.

Biological snakes possess compliant actuation on the whole body thanks to the muscle-tendon based system [37, 38], which allows their body to smoothly adapt or react to the surroundings. Still, the majority of bio-inspired snake robots uses stiff actuators and only a few include mechanical compliance. The OmniTread robot, created by Borenstein et al. [39] and the Slim Slime robot, created by Ohno and Hirose [40] rely on bellows-like pneumatic actuators which enable compliance but, because of the use of pneumatic actuation, are difficult to control in terms of force exerted and kinematic shape. The ACM R4.1, from Takaoka et al. [41], use as elastic element a rubber ring, which enable compliance, though limited, and force sensing.

The *SEA Snake* is another mechanism that allows compliance. Each module contains a Series Elastic Actuator (SEA), which consists of a stiff actuator in series with a spring, to achieve passive compliance.

While passive compliance is an advantageous property that makes the mechanism flexible, more resilient and immune to shocks and strains, it is not enough for an effective displacement though uneven terrains and obstacles. Hence, strategies for active compliance are devised in this work to enhance the snake robot ability to move.

Chapter 2

Hardware and Software

2.1 SEA Snake

The mechanism used in this work is the *SEA Snake* (Series Elastic Actuated Snake), the last generation robot created at the Biorobotics Lab (The robotics Institute, Carnegie Mellon University) [1]. It consists of a series chain of identical 1-DOF actuated link, the S-Series Actuator built by HEBI Robotics. Each module is rotated by ninety degrees with respect to the previous module's axis, to achieve an alternate orientation in the lateral and dorsal plane of the robot. A block diagram of the SEA Module electronics is shown in Figure 2.6.

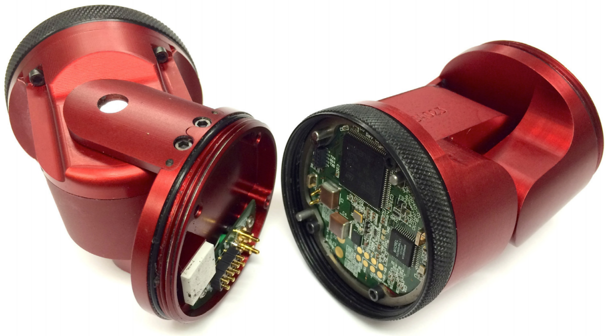

Figure 2.1: A photo of a SEA module. (Source: [1]).

Unlike the previous generations, the *SEA Snake* contains a Series Elastic Actuator in each module, which enables mechanical compliance and the ability to sense torques.

Series Elastic Actuators were first proposed by Pratt and Williamson in [42]. This non-stiff actuator links together a passive mechanical spring and a motor as shown in Figure 2.2.

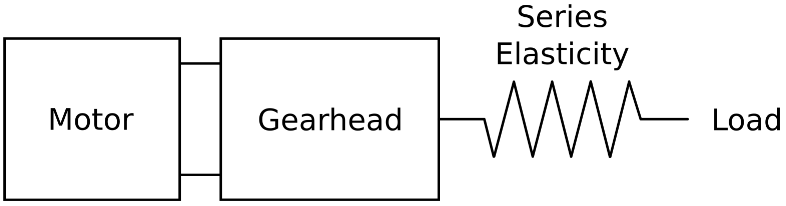

Figure 2.2: Scheme of the Series Elastic Actuation. (Source: [42]).

While stiffness is fundamental for tasks requiring precision, repeatability and stability (e.g. industrial robots for manipulation), many robots benefit from a semi-flexible structure, in particular mobile robots inspired by nature, where elasticity is a fundamental characteristics of animals. Series Elastic Actuators provide shock tolerance, accurate force control, mechanical compliance and energy storage. The *SEA Snake* modules uses a similar version of SEA developed in [43]. A typical snake robot consists of 16-18 modules linked together, plus a tail for the tether and a task-dependent head. The *SEA Snake* robot is the first generation of robots developed by the Biorobotics Lab that uses a brushless motor, a modified Maxon EC 20 flat motor with a nominal speed of 9300 RPM, powered by a 48V system voltage. The maximum torque output for each module is 7Nm. Two rotation encoders are included in the device in order to measure the spring deflection. The rugged design of the robot is compact and components are densely assembled to minimize the volume. Additionally, a sealed housing allows the snake to be splash-proof (IP66 Standards). Each modules contains a 32-bit processor and a 100 Mbps Ethernet data bus for communication, components that outperform the previous U-Snake (Unified Snake) [44]. The sensors in each module are a 3-axis Accelerometer, a 3-axis Gyro and Temperature, Voltage, Current and Torque sensors. Table 2.1 presents an overview on the specifications of the *SEA Snake* robot.

A cascaded PID controller enables angular position, velocity and torque control for the SEA modules. Each PID controller runs at 1kHz. In particular, four strategies are implemented for the motor control for each module.

1. Direct PWM. The torque commands directly the motor Pulse Width Modulation (PWM) from -1 to 1.

2. Cascaded Inner Torque. Position and velocity outer loops generates torque commands, which are combined to a desired feed-forward torque to define a set point for the inner torque controller, which compares desired and actual

SEA Module specifications	
Dimensions	Diameter: 5.1 cm Length (module): 6.4 cm Length (full 16 module robot): 1.174 m
Mass	Module: 205 g Full 16 module robot: 3.657 kg
Actuation	Max Torque: 7 N-m Max Speed: 33 RPM
Power	48 V Current (resting): 40 mA Current (max): 600 mA
Communincation	100 Mbps Ethernet
Sensing	Angular Position and Velocity Output Torque 3-axis Accelerometer 3-axis Gyro Temperature Voltage Current

Table 2.1: Table of the SEA Module specifications. (Source: [1]).

output torque thanks to the spring deflection (calculated as the difference between the two encoder positions). The scheme of this strategy is shown in Figure 2.3.

3. Cascaded Outer Torque. Position, Velocity, and Torque PID controllers all directly sum to generate motor PWM command, as shown in Figure 2.4.

4. Cascaded Intermediate Torque. Position outer loop generates a torque signal which is combined with a feed-forward torque signal. The intermediate torque signal is then passed through an inner Torque PID, which generates a PWM signal that ultimately get summed with a velocity PID output to generate motor PWM commands. The relative scheme is shown in Figure 2.5.

Each controller can be tuned by the user, who can define PID values K_p, K_i and K_d (proportional, integral and derivative terms), and additional parameters.

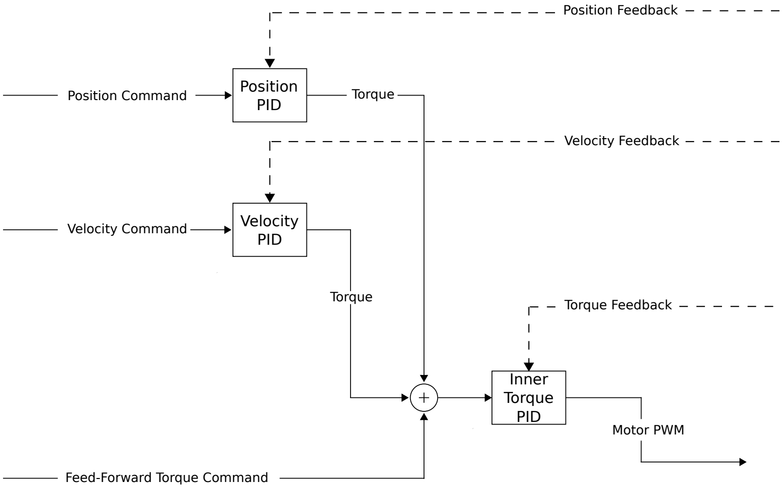

Figure 2.3: Cascaded Inner Torque PID controller for *SEA Snake* joint actuators.

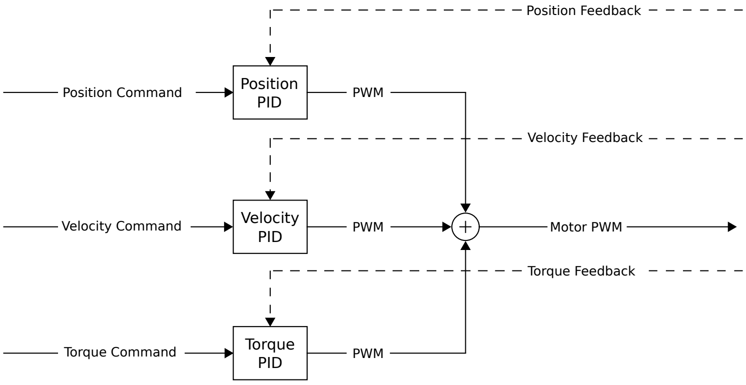

Figure 2.4: Cascaded Outer Torque PID controller for *SEA Snake* joint actuators.

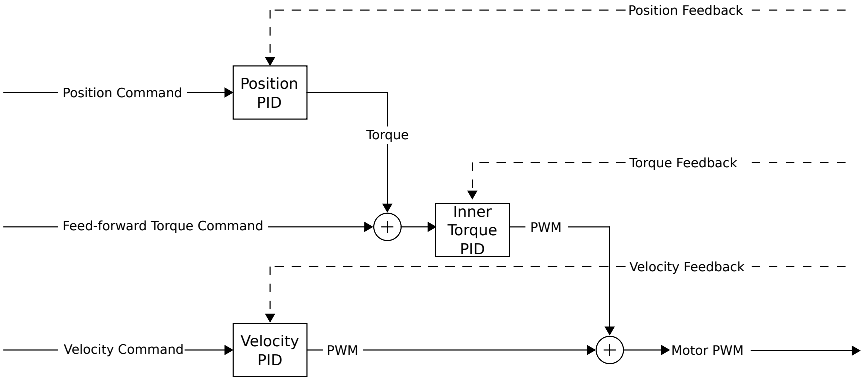

Figure 2.5: Cascaded Intermediate Torque PID controller for *SEA Snake* joint actuators.

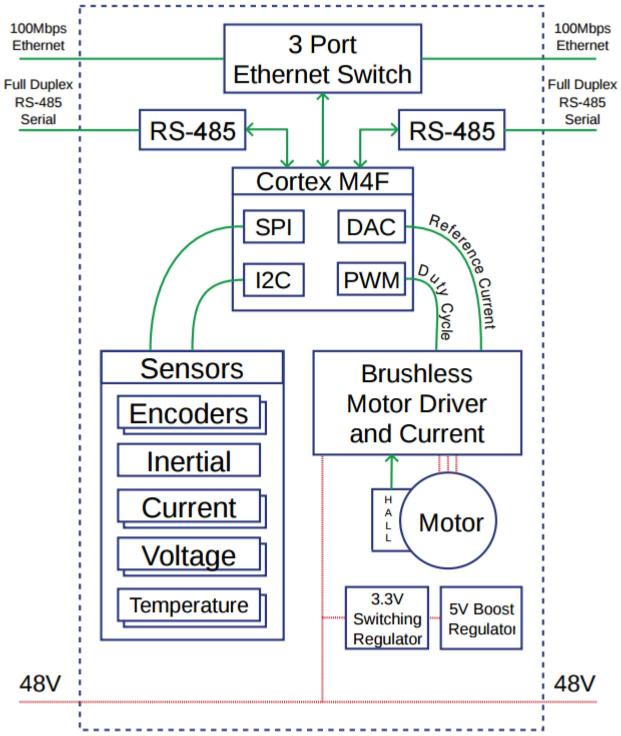

Figure 2.6: Block Diagram of the SEA Module electronics. (Source: [1]).

2.2 Software

The *SEA Snake* modules are built by HEBI robotics, a company founded in 2014 by a team from the Biorobotics Lab at Carnegie Mellon University. HEBI hardware comes with specific software tools to make using and controlling modules straightforward.

The module GUI is a simple platform to monitor and control modules connected to the network. Any feedback from the modules, i.e. sensed torque, current, voltage, temperature, position is displayed on the GUI. Additionally, it's possible to send commands and edit settings for individual modules. The platform requires Java 8u40 or newer to run. This simple graphic user interface is extremely useful to test and diagnose the hardware.

HEBI robotics provides a simple way to connect a module (or a group of modules) and MATLAB, allowing the user to control the hardware directly on it.

MATLAB is a powerful numerical computing environment developed by MathWorks which allows matrix manipulation, data analysis and plotting, algorithm implementation, etc. It can be used to interface with other languages, including C++, Java, Python and ROS. These premises makes this environment a powerful tool for programming robots.

Since MATLAB is a versatile and powerful tool, but it's not suited for real-time control, HEBI created libraries that are able to work around this limitation to enable research directly on it; prototypes of control schemes and algorithms can be tested in real time on hardware without the need for other programming languages. Code can be eventually translated into C/C++ and used on the same hardware.

The library is called SBK build, and needs to be incorporated in each project that connects the HEBI modules. An easy script can be created to add all the necessary files:

```
function [] = startup()
% startup sets up libraries and should be started once on startup.
currentDir = fileparts(mfilename('fullpath'));
addpath(fullfile(currentDir , 'hebi'));
end
```

SBK build includes HebiLookup, which periodically searches on the network for connected modules. Displaying *HebiLookup* provides an overview of all HEBI modules currently turned on and connected to the network.

```
>> HebiLookup

ans =
HebiLookup with properties:

                  lookupAddresses:{'10.10.10.255'}  IPv4
                  lookupFrequency: 5 Hz
    initialGroupFeedbackFrequency: 100 Hz

    Module  Family                Name              Mac Address
    ------  ----------------      ---------------   ---------------
    1       SEA Snake             SA002              00:1E:C0:F4:3F:78
    2       SEA Snake             SA003              00:1E:C0:F4:3F:78
    3       SEA Snake             SA007              00:1E:C0:F4:3F:78
    4       SEA Snake             SA011              00:1E:C0:F4:3F:78
    5       SEA Snake             SA013              00:1E:C0:F4:3F:78
    6       SEA Snake             SA017              00:1E:C0:F4:3F:78
    7       SEA Snake             SA021              00:1E:C0:F4:3F:78
    8       SEA Snake             SA024              00:1E:C0:F4:3F:78
    9       SEA Snake             SA025              00:1E:C0:F4:3F:78
    10      SEA Snake             SA026              00:1E:C0:F4:3F:78
    11      SEA Snake             SA027              00:1E:C0:F4:3F:78
    12      SEA Snake             SA032              00:1E:C0:F4:3F:78
    13      SEA Snake             SA036              00:1E:C0:F4:3F:78
    14      SEA Snake             SA040              00:1E:C0:F4:3F:78
    15      SEA Snake             SA042              00:1E:C0:F4:3F:78
    16      SEA Snake             SA054              00:1E:C0:F4:3F:78
    17      spare                 X5_test            0D:1E:42:C0:FF:EE
    18      Snake Monster         SA001              8B:05:84:48:BE:15
    19      Snake Monster         SA055              8B:05:84:48:BE:D5
    20      Snake Monster         SA056              8B:05:84:48:BE:F4
    21      Snake Monster         SA057              8B:05:84:48:BE:0A
```

Since many modules are to be commanded as a whole in a modular robot, the SBK build allows to create groups; this way, the user can send commands to and retrieve feedback from an individual mechanism made of several modules. Groups are identified by settable parameters, such as name, family or Mac Address, and they can be created specifying each individual modules:

```
if ~exist('snake', 'var')
    fprintf(['Creating new group\n']);
    names = {'SA099', 'SA044', 'SA07'},
    snake = HebiLookup.newGroupFromNames('*', names);
```

```
end
fprintf('Success!\n\n');
```

or by selecting one of the connected modules. In this case, all the modules linked to it become a family:

```
if ~exist('snake', 'var')
    fprintf(['Creating new group\n']);
    snake = HebiLookup.newConnectedGroupFromName('*', 'SA042');
end
fprintf('Success!\n\n');
```

In particular for the SBK libraries, HebiLookup commands are:

HebiLookup.clearGroups clears the current groups of modules;
HebiLookup.clearModuleList clears the list of connected modules;
HebiLookup.newConnectedGroupFromMac creates a new group by specifying the hardware Mac Address of one module. All the modules connected to it are included in the created group;
HebiLookup.newConnectedGroupFromName creates a new group by specifying the user defined name of one module. All the modules connected to it are included in the created group;
HebiLookup.newGroupFromFamily creates a new group by specifying a previously defined family of modules;
HebiLookup.newGroupFromMacs creates a new group by specifying the hardware Mac Address of the modules to be included;
HebiLookup.newGroupFromNames creates a new group by specifying the user defined name of the modules to be included;

HebiLookup.setInitialGroupCommandLifetime sets the current group command lifetime;
HebiLookup.setInitialGroupFeedbackFrequency sets the current group feedback rate;
HebiLookup.setLookupAddresses sets the lookup target address;
HebiLookup.setLookupFrequency sets the lookup and info request rate.

When a group is created it is possible to send commands, retrieve feedback, change properties and log data by using the HebiGroup Methods:

getNumModules - returns the number of grouped modules;
getFeedbackFrequency - returns the current feedback frequency;
setFeedbackFrequency - sets the feedback frequency;

getCommandLifetime - returns the current command lifetime;
setCommandLifetime - sets the command lifetime;

getGains - returns the current gains;
getInfo - returns meta information on the grouped modules;
getNextFeedback - returns the next new synchronized feedback;
set - sends commands and settings;
startLog - starts background logging;
stopLog - stops logging and returns a readable format.

Chapter 3

Background and Related Work

3.1 Serpenoid curve and Parametrized Gaits

The control of the robot is based on the serpenoid curve, first introduced by Hirose [18], and reformulated in [45], which sinusoidally varies the curvature of the snake body. In particular for the an N-joints snake robot, two parametrized sine waves propagates through the two planes (lateral and dorsal) of the snake, defining the joint angles θ_i, $1 \leq i \leq N$ at time t. This control equation is referred as the "compound serpenoid curve".

$$\begin{cases} \theta_{i\,lat} = \phi_{lat} + A_{lat}\sin(\omega_{S\,lat} s_{i\,lat} - \omega_{T\,lat} t) & lateral \\ \theta_{i\,dor} = \phi_{dor} + A_{dor}\sin(\omega_{S\,dor} s_{i\,dor} - \omega_{T\,dor} t + \delta) & dorsal \end{cases} \tag{3.1}$$

where $\theta_{i\,lat}$ and $\theta_{i\,dor}$ are the commanded link angles on the lateral and the dorsal plane, ϕ_{lat} and ϕ_{dor} are the angular offset, A_{lat} and A_{dor} the amplitude, δ the phase shift. $\omega_{T\,lat}$ and $\omega_{T\,dor}$ are the temporal phase, while $\omega_{S\,lat}$ and $\omega_{S\,dor}$ describe the spatial frequency of the wave, which determine the wavelength of waves, hence the number of waves on the snake robot's body on both the lateral and the dorsal plane. $s_{i\,lat} = \{0{,}2 \cdot l_s, \ldots, N \cdot l_s\}$ and $s_{i\,dor} = \{l_s{,}3 \cdot l_s, \ldots, (N-1) \cdot l_s\}$ are the distances from the head of the module i, where l_s is the length of one module. The parameters are used to define different open-loop joint trajectories for locomotion, called gaits, such as sidewinding, rolling, pole climbing, slithering. The advantage of using the serpenoid curve equation is the coordination of a large number of DOF with a lower number of parameters, often intuitively meaningful. However, this framework has a major drawback. The control takes place in the shape space of the robot, making it difficult to visualize the actual pose of the mechanism. In order to obtain the kinematic configuration of the robot, a forward kinematic map is needed, built starting from a known frame oriented with respect to the world. Because of this reason, a relationship between gait parameters and kinematic pose is not straightforward, and an analytical expression between the two is difficult to extract.

3.2 Shape functions

A mobile robot that locomotes through complicated terrains is not an easy problem in robotics. A high degree-of-freedom system moving through an environment which is impossible to map in a reasonable way is an even harder problem that needs simplification to be solved in real time.

Hyper redundant mechanisms need a precise coordination of the internal degrees of freedom, an underlying structure that governs simultaneously the motion of each link.

Shape functions allow high degrees of freedom tractability, providing a low-dimensional and compact framework that coordinates each DOF. This approach allows to describe the geometry of a mechanism as a defined shape: it couples degree of freedom by means of shapes, allowing dimensionality reduction, i.e. projecting from an N joint space to a M low-dimensional subspace, hence considerably simplifying the control of highly-redundant robots.

First of all, a general introduction to shape functions must be provided.

For the type of locomoting system that are used in this work, the configuration space C can be separated into two components, the position space and the shape space:

$$C = G \times B \tag{3.2}$$

where the configuration space C of the robot is the space of all the possible configurations of the mechanism. The position space G corresponds to the position and orientation of a frame attached to a body's center of mass and the shape space B is the space of internally controlled joint variables for the system.

A vector field in the shape space represents infinitesimal motion in the same space, which translates in the corresponding motion in the position space. A path drown in the shape field effectively defines a particular motion of the robot. A path describing a gait is a closed path, because the movement of the internal variables is cyclical. The tangent vector along each point of this gait can be computed and compared to the underlying vector field, in order to see how well aligned the two vectors are. The more the two are aligned, the more the infinitesimal motion is going to contribute to the overall net locomotion of the system. It is then advantageous to work inside regions of the shape space producing a good locomotion pattern, i.e. an efficient gait. The region can be parametrized, defining a low dimensional structure that can scale up and down the same motion in the vector field. i.e. a relationship between shape space and work space depending on some well-defined parameters:

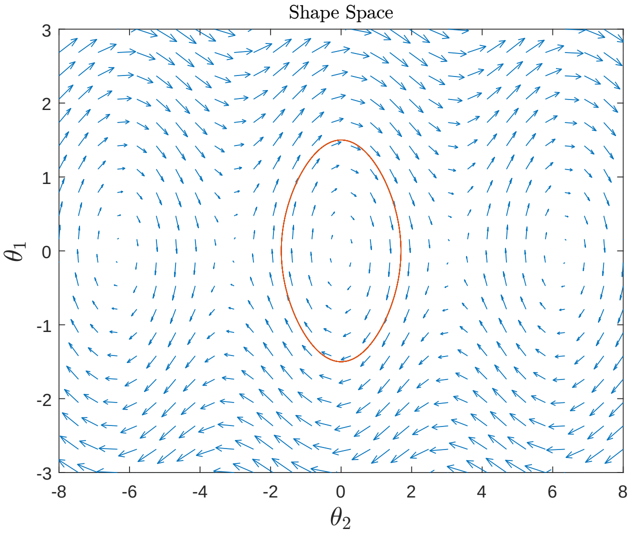

Figure 3.1: A representation of a path in the Shape space, corresponding to a specific gait for the robot.

$$\theta = h(\sigma) \tag{3.3}$$

where $h(\sigma)$ is the function of a specific region, a shape parametrization which defines the fundamental structure of the motion, and σ represent the shape parameters, i.e. variables that characterize the shape function h.

In the context of motion planning, where motion primitives, simple behaviours or discrete locomotion elements, can be concatenated together to form motion plans, the presented framework defines a low-dimensional way to represent families of motion that produce similar robot locomotion.

This particular structure can be extended from motion planning to control system design of very complex systems. This approach stresses how simultaneous coordination of the internal degree of freedom matters for any mobile robots, from legged mechanism to snake-like robots. For a more complex and general N-dimensional system (e.g. a robot with a N dimensional shape space), the same structure can be applied if a dimensionality reduction via any basis function is found, projecting the high N dimensional space to a lower M dimensional space. State variables $\theta \in \mathbb{R}^N$ are described in terms of bases $\beta_j \in \mathbb{R}^N$ and parameters $\alpha_j \in \mathbb{R}$, $i \in N, M < N$:

$$\begin{aligned}\theta &= \sigma_1(t)\beta_1(i) + \sigma_1(t)\beta_1(i) + ... + \sigma_M(t)\beta_M(i) \\ &= \sum_{j=1}^{M} \sigma_j(t)\beta_j(i) = h(\sigma, \beta)\end{aligned} \tag{3.4}$$

The basis function consists of the same parameters, or the same concept behind the original parameters, to define in the subspace a locomotion pattern belonging to a good region in the shape subspace.

Formally, a shape function $h : \Sigma \rightarrow \mathbb{R}^N$ maps a point σ in the shape parameter space $\Sigma \in \mathbb{R}^M$, into the joint space of an N-joint mechanism $\theta \in \mathbb{R}^N$,

$$\theta_\gamma = h(\sigma_\gamma, \beta) \tag{3.5}$$

Shape functions are composed of:

- shape bases β, which define the spatial coupling between different degree-of-freedom, encoding static shapes for a particular mechanism;
- shape parameters σ_γ where $\gamma = \{0, d\}$ correspond respectively to the nominal and desired shape parameters.

The term σ represents the defined shape parameters and β is a proper basis function (or a set of basis functions). Ultimately, basis functions and shapes for bio-inspired robot locomotion originate from the actual movement of animals; in other words, parametrized good locomotion patterns stem from basis functions corresponding to certain stances or configurations that are found in nature. In the particular case of the snake robot used in this work, shape functions are sinusoidal looking fixed configuration extracted from the serpenoid curve defined by Hirose [18].

Starting from the general formulation of the serpenoid curve, it is possible to rewrite it in terms of shape functions, like described in [46, 47]. In particular, a simple formulation of the serpenoid function (3.1) can be rewritten in terms of a set of basis functions $\beta = (1, sin(\eta s(i)), cos(\eta s(i)))$ and the inherent shape parameters $\sigma = (\kappa, Acos(\omega t), Asin(\omega t))$, as described in (3.6).

$$\begin{aligned} \theta &= \kappa + sin(\eta s(i) - \omega t) \\ &= \kappa + Acos(\omega t)sin(\eta s(i)) - Asin(\omega t)cos(\eta s(i)) \\ &= \sigma_1(t) + \sigma_2(t)sin(\eta s(i)) + \sigma_3(t)cos(\eta s(i)) \\ &= \sigma_1(t)\beta_1(i) + \sigma_2(t)\beta_2(i) + \sigma_3(t)\beta_3(i) \end{aligned} \tag{3.6}$$

where θ are the joint angles, $s(i)$ the module distance from the head, κ is the offset, A is the amplitude, η is the spatial frequency and ω is the temporal frequency.

Since coordination between elements of a hyper-redundant mechanisms is fundamental for an efficient locomotion, shape functions, that connect together and simplify the system, are a core tool in this work. From libraries of independent motions, a proper way to actually control the robots is to be found. A first and easy approach consist in exploiting the low dimensional parametrization to generate joint angles that end up being references for PID controller loops. In other words, shape-based control connects high level planning to low level control. This middle layer allows the robot to be controlled in terms of shapes rather than individual joints, which provides an intuitive way to specify behaviors and potentially incorporate feedback into trajectory generation. As presented in this work, it is possible to use the shape function formulation (3.6) to produce compliant motion in snake robot's gaits.

3.3 Admittance control

Model free control methods allow to define an easy structure in which high level planners define a specific feed-forward trajectory and low level controls regulate the system to track the desired open-loop sequence. While this approach is useful in simple scenarios, it becomes detrimental in environments where the nominal operation is ineffective or even impossible, resulting in the robot thrashing in place or getting stuck.

Instead on relying only on a library of predefined open loop motions selected for specific conditions, a solution can be found in connecting this "blind" controls to the environment, in such a way that the system is able to respond or react to environmental features.

Admittance control provides an intuitive way to extend nominal feed forward system trajectories with respect to external forcing, linking with a dynamic relationship nominal joint signals θ_0 to desired signals θ_d. Therefore, the open-loop signals θ_0 are allowed to adapt and warp, producing desired signals θ_d in presence of torques exerted by the environment on the system. Hence, admittance control provides a way to effectively modify nominal signals based on external inputs, defining a dynamic relationship between external forces and joint position, velocity and acceleration.

For a system with N joints, joint angles $\theta \in \mathbb{R}^N$, an admittance controller is specified by:

$$M(\ddot{\theta}_d) + B(\dot{\theta}_d) + K(\theta_d - \theta_0) = F_{ext} \tag{3.7}$$

The dynamical system defined is monostable, where the desired angles θ_d vary around the single attractor θ_0 in response to a forcing term F_{ext}. The matrices M, B, and K are respectively effective mass, damping and spring constant matrices. The desired joint signals θ_d follow with linear dynamics of a spring-mass-damper the nominal trajectory defined by θ_0.

3.4 Shape-based compliance

Linking together a shape-based framework with admittance control allows to define a middle-layer in the control architecture between high level planning and low level control that naturally includes compliance for articulated locomotion.

While pure force control strategies for hyper-redundant mechanisms provide high compliance but weaken the underlying coordination structure of the robot, pure position control strategies confer way too much structure for the robot to move in confined spaces, not allowing any compliance.

This work supports the belief that a middle way between compliance and rigidity is needed for the robot to navigate through an unstructured environment without losing its underlying shape, deforming it to adapt to the terrain and the obstacles.

Shape-based control allows the system to be described by shapes, that coordinate the many degree of freedom of the robot. Hence, augmenting the architecture with a modified admittance control grants an intuitive way to make the system compliant, allowing the nominal shapes to warp and stretch in the presence of external forces, [48].

In other words, in a shape-based framework that defines a desired motion, i.e. a desired joint angles θ_d given fixed nominal parameters σ_d and a good basis β,

$$\theta_d = h(\sigma_d, \beta) \tag{3.8}$$

a middle layer which introduces feedback from the environment is needed,

$$\sigma_d = g(\sigma_0, \lambda) \tag{3.9}$$

where σ_0 is the nominal shape parameter (that describe the nominal motion) and λ is a term that takes into account feedback.

The chosen strategy to create the middle layer, as stated before, is the admittance control. But instead of using the conventional admittance framework described by (3.10), which creates a relationship between join position and external forces,

$$M(\ddot{\theta}_d) + B(\dot{\theta}_d) + K(\theta_d - \theta_0) = F_{ext} \tag{3.10}$$

it is used a modified version which connects shape parameters and external forces. This is done by using an analytical function that defines the relationship between the high-dimensional joint space and the low-dimensional shape space, effectively performing a change of coordinates from one to the other. Following this approach, the angles $\theta \in \mathbb{R}^N$ can be described as:

$$\begin{aligned} \theta &= h(\sigma, \beta) \\ \dot{\theta} &= \frac{\partial h}{\partial \sigma}\dot{\sigma} = J_\sigma \dot{\sigma} \\ \ddot{\theta} &= \dot{J}_\sigma \dot{\sigma} + J_\sigma \ddot{\sigma} \end{aligned} \tag{3.11}$$

where σ are the shape parameters and $J_\sigma \in \mathbb{R}^{M\times N}$ is the Jacobian of the shape function $h \in \mathbb{R}^M$ with respect to $\sigma \in \mathbb{R}$:

$$J_\sigma = \begin{pmatrix} \frac{\partial h_1(\sigma)}{\partial \sigma_1} & \cdots & \frac{\partial h_N(\sigma)}{\partial \sigma_1} \\ \vdots & \ddots & \vdots \\ \frac{\partial h_1(\sigma)}{\partial \sigma_M} & \cdots & \frac{\partial h_N(\sigma)}{\partial \sigma_M} \end{pmatrix} \tag{3.12}$$

Hence, shape-based control is based on an admittance control framework, where the control takes place directly in the shape-parameter space Σ. The desired shape parameters σ_d vary compliantly around a nominal shape parametrization σ_0 and the function g ends up being:

$$M'_\sigma(\ddot{\sigma}_d) + B'_\sigma(\dot{\sigma}_d) + K'_\sigma(\sigma_d - \sigma_0) = F_\sigma \tag{3.13}$$

where $M' = J_\sigma^T M J_\sigma$, $B' = J_\sigma^T B J_\sigma$, and $K' = J_\sigma^T K J_\sigma$. More generally, M', B' and K' are the matrices that control the dynamic response of the shape parameters in response to a forcing term F_σ and can be directly defined in the shape space.

The forcing function is selected to be a function of external joint torques τ_{ext},

$$F_\sigma = J_\sigma \tau_{ext} = \tau'_{ext} \tag{3.14}$$

since the Jacobian J_σ, as described in (3.11), maps joint torques to shape forces τ_{ext}.

While the fundamental structure of the shape defined by the overall coordination of the links of a mechanism is constrained by a particular basis specified by β_i, the shape itself can freely change within the same basis, stretching and warping in response to external forces without losing the underlying coordination.

The shape-based compliance control is devised supporting the belief that a snake robot shouldn't rely only on force control to be compliant, but it should also have a defined set of kinematics to interact with the world. By using shape functions a pure position control for the snake could be performed, projecting the control into a lower dimensional space and giving some sort of articulation in doing so (i.e. a defined coordination of the robot's joints). This would result in a rigid motion unable to deal with the environment. On the other hand, by using a conventional admittance controller a pure force control could be realized, resulting in a very decentralized control that would make the snake lose its gait's shape, disrupting a motion that relies solely on it.

A middle way between these two extremes is found in this controller, where a good locomotion pattern (i.e. gait) is taken and modified to adapt to certain environment configurations.

3.5 Decentralization

In robotics there are two extremes of centralization in joint coordination: fully centralized, where a single controller coordinates all of the degrees-of-freedom in the system, and fully decentralized, where each degree-of-freedom is controlled independently. Between these extremes lies a spectrum in which the greater the level of centralization, the more each joint is influenced by the motion of the other joints in the system. The degree to which a controller is centralized is typically a fixed property in the controller specification. Fully centralized control may be effective in a uniform environment, but they lose effectiveness in terrains where different portions of the robot should independently adapt to local environmental features. On the other hand, a decentralized control weakens the coordination of the robot as a whole, making it less structured.

Chapter 4

Compliant Gaits

Snake robots, with their many degrees-of-freedom, have the potential to take on different configurations in order to maneuver through unstructured terrains, confined spaces and cluttered environments. Among the gaits devised, three in particular are considered and augmented to allow compliance:

- Slithering;
- Pole climbing;
- Sidewinding.

The developed control techniques are implemented on the *SEA Snake* and different experimental setups are used to test them.

4.1 Slithering Gait Compliance

4.1.1 Introduction

Slithering gait is a two dimensional gait (i.e. takes place only in the lateral plane of the snake backbone) that propels the snake forward thanks to lateral undulation. Unlike other snake movements, as [21] specifies, this gait requires environmental features to move forward, since it is not self-transporting. While irregular terrains often limit the movement of conventional robots, they are essential for this slithering-like gait. Supporting the belief that a behaviour in between rigidity and compliance is beneficial to locomotion in unstructured environments, the proposed controller presents shape based compliance applied to a slithering gait. The work is based on [47], and extend it to include full compliance in amplitude, wavelength and temporal phase of the serpenoid shape function.

It is shown how enabling compliance in these parameters improves locomotive performance in irregular terrain and that decentralizing control for spatial parameters like amplitude and wavelength improves locomotion performance.

4.1.2 Background

The locomotion gait is a planar slithering, hence only half of the modules are actually being commanded. The corresponding open-loop serpenoid curve is:

$$\theta_i = \phi + A \sin(\omega_S s_i - \omega_T t) \tag{4.1}$$

where ϕ is the offset, which for this particular gait is set to 0, A is the amplitude, ω_S is the spatial frequency and ω_T is the temporal frequency.

4.1.3 Wavelength compliance and variable centralization

In Section 3.1 is described how the spatial frequency ω_S in (4.1) defines the number of waves on a snake-like robot's body. Making this parameter compliant enables the number of waves to oscillate around a nominal value in response to the obstacles sensed by the snake. Exploiting the shape function (3.6), it is selected as shape parameter the spatial frequency $\sigma = \omega_S(t)$, which consists of a nominal and a desired parameter as described in Section 3.2:

$$M^{\sigma}_{\omega_S}(\ddot{\omega}_S(t)) + B^{\sigma}_{\omega_S}(\dot{\omega}_S(t)) + K^{\sigma}_{\omega_S}(\omega_S(t) - \omega_{S,0}) = F_{\sigma}(t) \tag{4.2}$$

where ω_S is the desired spatial frequency shape parameter, $\omega_{S,0}$ is the nominal spatial frequency, $M^{\sigma}_{\omega_S}$, $B^{\sigma}_{\omega_S}$ and $K^{\sigma}_{\omega_S}$ are the virtual mass, damping and spring constants that regulate the dynamical behaviour of the shape parameter. F_{σ} represent the forcing term that depends on external torques sensed by the snake.

Hence, allowing the spatial frequency parameter to adapt gives the robot the freedom to conform to environmental features, which, as demonstrated, results in more effective locomotion. In Figure 4.1 it is shown how the snake reacts to external forces; when a forcing is applied, the torques measured at all joints are mapped into a shape force, resulting in a change in the wavelength of the serpenoid curve. When forcing is removed, the wavelength returns to its nominal value. In particular, ω_s has a nominal set point producing 1.5 body waves, and complies to one or two body waves.

The approach presented in this work implements a controller that is nominally in between the fully centralized and fully decentralized examples. This design is motivated by the desire to allow robot to traverse a variety of non-uniform environments without manual adjustment of the control strategy or heavy computation. Figure 4.1 shows how in a centralized compliant controller, locally applied forces

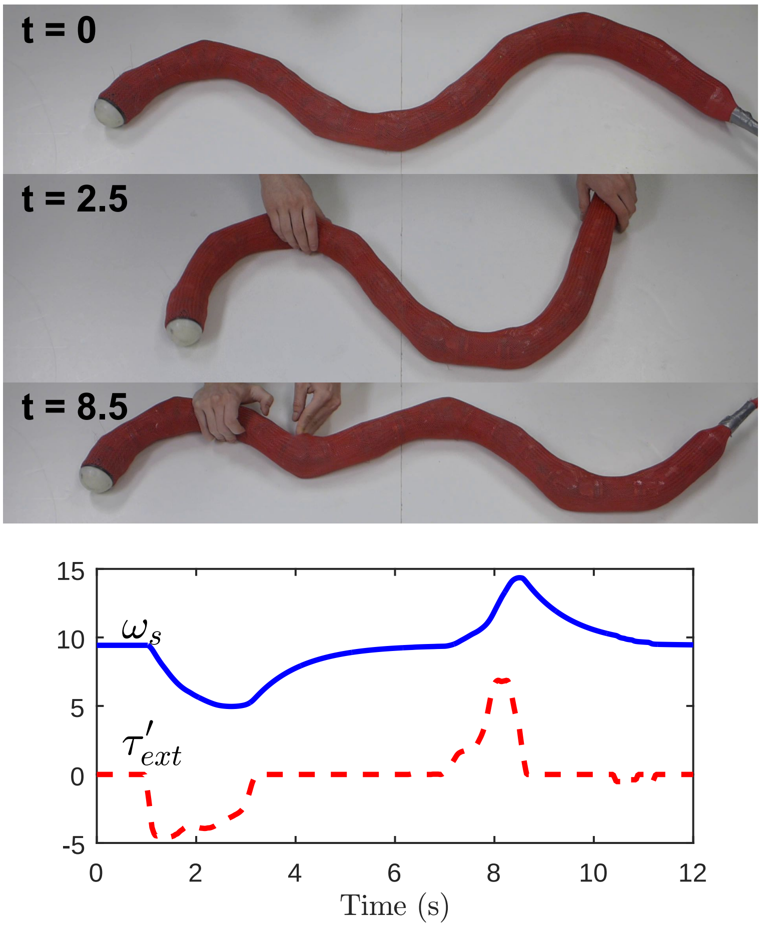

Figure 4.1: Snake robot experiencing external forces, above, and corresponding plot of the shape force τ'_{ext} and serpenoid spatial frequency ω_s, below. (Source: [49]).

change the wavelength over the entire body. The shape is allowed to deform locally to environmental features with a less centralized controller. This is accomplished with a modified version of the activation windows presented in [47]. Activation windows should be independent to allow for the shape in each window to independently comply in response to local external forcing. The full shape function is a summation of activation windows. In regions where windows overlap, their shape parameters are summed. Direct summation is acceptable when the shape function h is linear in the shape parameters, but parameters like frequency and phase are nonlinear in the shape function. In Figure 4.2 is shown how Gaussian windows overlap, making them impractical for nonlinear parameters like spatial frequency, while the sigmoid-rectangular windows overlap minimally. Hence, the prior activation function is modified from a Gaussian, which produces interference between windows, to a smooth-rectangular step function from a sum of sigmoids,

$$\omega_s(s) = \sum_{j=1}^{W} \omega_{s,j} \left(\frac{1}{1 + e^{-m(s - s_{j,start})}} + \frac{1}{1 + e^{m(s - s_{j,end})}} \right) \tag{4.3}$$

where m controls the length of the transition period between shape parameters in neighboring windows, W is the number of windows, window j spans the backbone over arc length $(s_{j,start}, s_{j,end})$, and the spatial frequency in window j is $\omega_{s,j}$. For each window to contain a fixed number of waves, their start and end positions are anchored to specific locations on the base serpenoid curve, and move with the waveform, as depicted in Figure 4.2. The anchor points of the windows on the backbone curve are controller design variables, and could be placed, for example, at regions of largest or zero curvature. Section 4.1.3 shows the snake with three activation windows, each of which covers half of a wave length. The first two windows have a high spatial frequency, and coordinate only a few joints each, while the third has a low spatial frequency and coordinates many joints.

The wavelength complies in response to torques, as specified by (4.2). The distance between anchor points changes with the wavelength, which changes the size and number of windows covering the body length. A window manager subroutine calculates these changes in windows by maintaining an array of window lengths and positions. If the wavelength at the front of the body increases, the size of the front window grows, and the subsequent windows move back. The subroutine also generates new windows at the head as the waveform travels, and pulls windows back onto the backbone from the tail if the front windows shrink.

In particular, Figure 4.2 shows how the window edges are attached to the points of zero curvature on the backbone curve, where the black circles represent joints which lie at fixed locations along it. At $t = 1$, the wave has moved along the backbone, and the wavelength in the red window has grown. Where it previously coupled three joints, it now couples five, and the corresponding curvature along the backbone at $t = 1$ consequently changes. Hence, the number of joints covered by a window varies with window size. This means that the degree of centralization in coordination can vary widely within the same controller in response to external forces. In spatial parameters like wavelength, such a variation led to improved performance in experiments. However, as it's shown next, not all parameters benefit from decentralization.

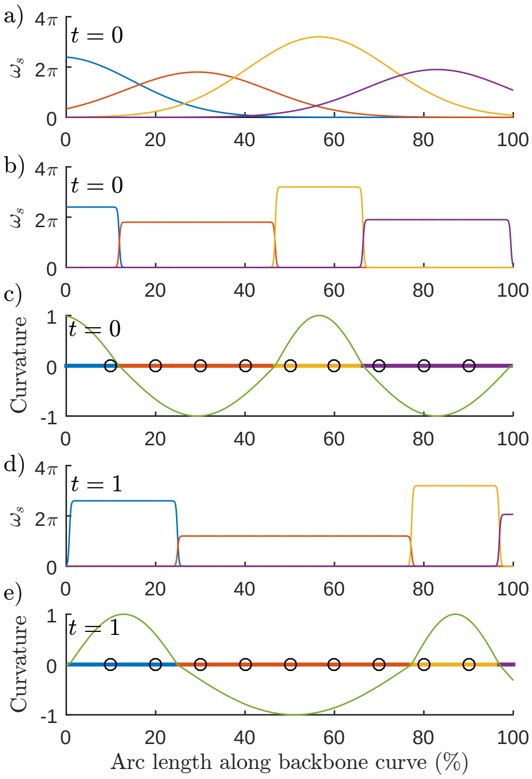

Figure 4.2: An illustration of activation windows, which locally modulate the spatial frequency ω_s of the shape. a) Gaussian spatial activation windows at time $t = 0$. b) Smooth-rectangular activation windows at $t = 0$. c) Curvature along the backbone at $t = 0$. d) Smooth-rectangular activation windows at $t = 1$. e) Curvature along the backbone at $t = 1$. (Source: [49]).

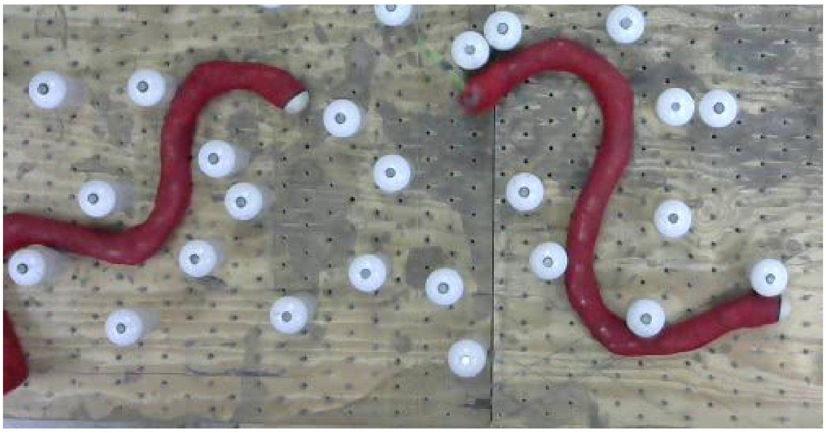

Figure 4.3: The snake robot compliantly adjusts its spatial frequency as it moves through an unknown environment. A larger spatial frequency corresponds to more waves present on the robot.

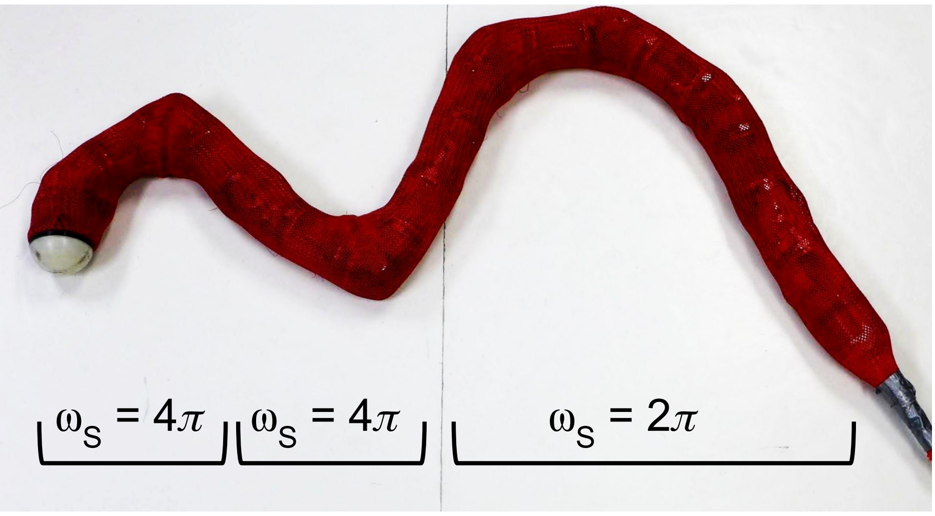

Figure 4.4: A demonstration of multiple independent activation windows. (Source: [49]).

4.1.4 Temporal phase compliance

The temporal frequency ω_T of the serpenoid curve (3.1) determines the speed at which the curvature wave travels along the backbone. In this work the temporal phase $\omega_T t$ is allowed to be compliant, which allows the serpenoid wave to slow down in response to forces. It is defined a temporal phase compliant parameter $\sigma = \Omega_t$, with a nominal value that changes in time, $\Omega_{t,0} = \omega_T t$:

$$M^{\sigma}_{\Omega_t}(\ddot{\Omega}_t(t)) + B^{\sigma}_{\Omega_t}(\dot{\Omega}_t(t)) + K^{\sigma}_{\Omega_t}(\Omega_t(t) - \Omega_{t,0}) = F_{\sigma}(t) \tag{4.4}$$

where $M^{\sigma}_{\Omega_t}$, $B^{\sigma}_{\Omega_t}$ and $K^{\sigma}_{\Omega_t}$ are the virtual mass, damping and spring constant that regulates the dynamical behaviour of the shape parameter. F_σ represent the forcing term that depends on external torques sensed by the snake.

When forces prevent the serpenoid from freely progressing, and the snake robot can no longer glide forward freely, the phase will be delayed. As the phase falls farther behind its nominal value, the force applied by the joints will grow while other compliant parameters adjust. When the shape has conformed to obstacles that had prevented motion, the snake will be able to progress, and the phase will return to its nominal value. An example of compliant temporal phase is shown in Figure 4.5, where a force is manually applied to the snake robot at $t = 2.75$ to slow the phase, and released at $t = 3.75$, after which the phase increases to return to its nominal value, $\Omega_T t$. When the temporal phase parameter is allowed to comply locally, the overall waveform, which provides structure between independently acting activation windows, degrades. As such, this parameter is centrally controlled. In the experiments, control decentralization of temporal phase results in degraded performance while full centralization leads to measurable improvements, as the timing of the shape evolution is uniform across the robot but complies in response to external forces.

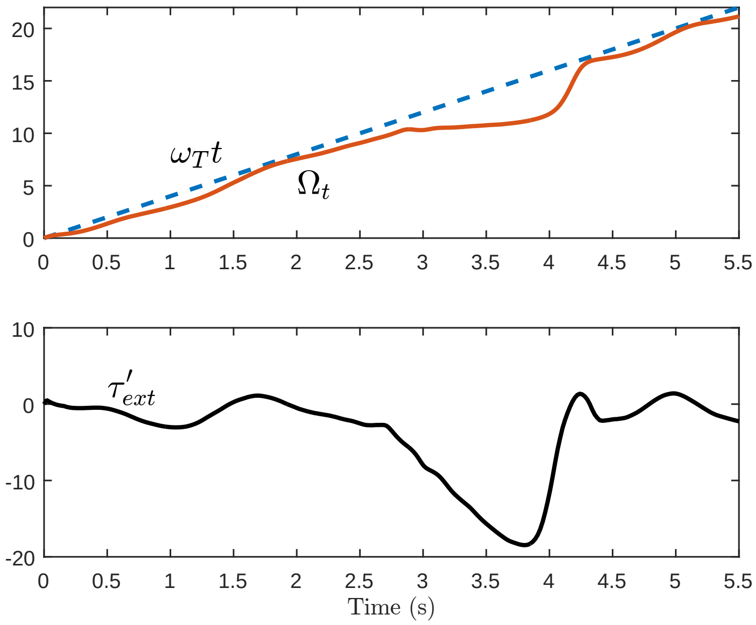

Figure 4.5: The response of the temporal phase Ω_t to an external forcing τ'_{ext} in a fully centralized shape-based controller. (Source: [49]).

4.1.5 Full controller

The work in [47] introduces shape based compliance in locomotion, implementing amplitude compliance on a snake robot. The controller is here extended to make compliant all the parameters of the shape parametrization of the serpenoid curve (3.6). Hence, for a snake robot with N links, joint angles $\theta_i \in \mathbb{R}^N$, and W decentralized windows, the shape parameters $\sigma_f \in \mathbb{R}^{1\times n_\sigma}$ are amplitude A, spatial frequency ω_S and temporal frequency Ω_T:

$$\sigma_f = \{A(s,t), \omega_S(s,t), \Omega_T(t)\}^T \tag{4.5}$$

where the amplitude and the spatial frequency are decentralized parameters as shown in (4.3), i.e. $W \times 1$ vectors that consist of one parameter for each independent section on the snake (hence time and space dependent):

- $A(s,t) = \{A_1(s,t), A_2(s,t), \dots, A_W(s,t)\}$
- $\omega_S(s,t) = \{\omega_{S,1}(s,t), \omega_{S,2}(s,t), \dots, \omega_{S,W}(s,t)\}$

The fully centralized temporal frequency parameter Ω_T is a time dependent scalar, one for the whole snake's body. The resulting shape function is:

$$h_i(A(s,t), \omega_S(s,t), \Omega_T(t)) = A(s,t)\sin(\omega_S(s,t)s_i - \Omega_T(t)t), \tag{4.6}$$

and the full shape-based compliant controller reads:

$$M_f^\sigma(\ddot{\sigma}_{f,d}(s,t)) + B_f^\sigma(\dot{\sigma}_{f,d}(s,t)) + K_f^\sigma(\sigma_{f,d}(s,t) - \sigma_{f,0}) = F_\sigma(s,t) \tag{4.7}$$

where $M_f^\sigma, B_f^\sigma, K_f^\sigma \in \mathbb{R}^{n_\sigma \times n_\sigma}$ are respectively the effective mass, damping and spring constant matrices of the system. The term $\sigma_{f,0}$ represents the nominal shape parameters $\sigma_{f,0} = \{A_0, \omega_{S,0}, \Omega_{T,0}\}^T$. $F_\sigma(s,t) \in \mathbb{R}^{n_\sigma \times 1}$ is a mapping of the external forces $\tau_{ext}(s,t) \in \mathbb{R}^N$ from joint space $Q \in \mathbb{R}^N$ to shape space $\Sigma \in \mathbb{R}^{n_\sigma}$ as seen in (3.14), where the mapping function is the Jacobian matrix $J(h) \in \mathbb{R}^{N \times n_\sigma}$ defined in (3.12). In particular,

$$J_\sigma^T(h) = \begin{pmatrix} \frac{\partial h_1(A,\omega_S,\Omega_T)}{\partial A_1} & \cdots & \frac{\partial h_N(A,\omega_S,\Omega_T)}{\partial A_1} \\ \vdots & \ddots & \vdots \\ \frac{\partial h_1(A,\omega_S,\Omega_T)}{\partial A_W} & \cdots & \frac{\partial h_N(A,\omega_S,\Omega_T)}{\partial A_W} \\ \frac{\partial h_1(A,\omega_S,\Omega_T)}{\partial \omega_{S,1}} & \cdots & \frac{\partial h_N(A,\omega_S,\Omega_T)}{\partial \omega_{S,1}} \\ \vdots & \ddots & \vdots \\ \frac{\partial h_1(A,\omega_S,\Omega_T)}{\partial \omega_{S,W}} & \cdots & \frac{\partial h_N(A,\omega_S,\Omega_T)}{\partial \omega_{S,W}} \\ \frac{\partial h_1(A,\omega_S,\Omega_T)}{\partial \Omega_T} & \cdots & \frac{\partial h_N(A,\omega_S,\Omega_T)}{\partial \Omega_T} \end{pmatrix} \tag{4.8}$$

Finally, the joint angles θ_i are computed at each time step t using (4.1).

4.1.6 Experimental results

An experimental comparison of controller performance was conducted with the snake robot and peg board shown in Figure 4.3. A modified CPG controller was designed and implemented on the same robot to compare it to the novel approach presented in this work. A description of the CPG controller is provided in [49]. The robot used throughout this work was composed of eighteen identical series-elastic actuated modules [1], arranged such that the axes of rotation of neighboring modules were torsionally rotated ninety degrees relative to each other; in this work, only planar gaits were tested, so only nine modules were active. Deflection between the input and output of a rubber torsional elastic element was measured using two absolute angular encoders and used to compute the torque experienced by each module. The pegs were placed in an irregular pattern. The robot was covered with a braided polyester sleeve to reduce friction, with reflective markers attached along the backbone over the joint axes. The isotropic friction of the polyester sleeve prevented the robot from moving without using the pegs. Data was collected with a four-camera OptiTrack motion capture system (NaturalPoint Inc., 2011).

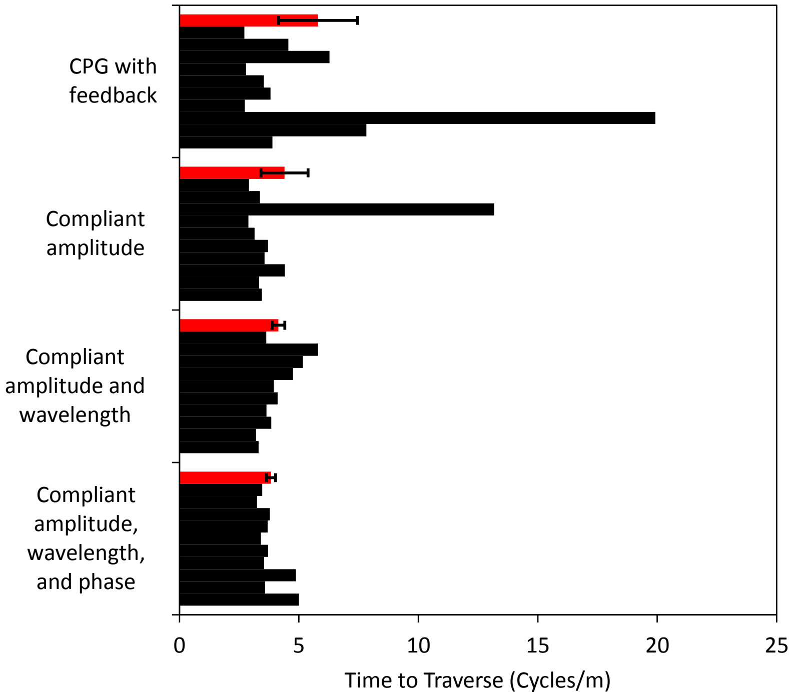

Figure 4.6: Comparison of shape-based and distributed-feedback CPG control methods for a snake robot traversing a irregularly spaced peg array. Black bars represent 10 individual trials and the red bars controller mean, with standard error bars. (Source: [49]).

For each trial, the robot was started at a different location in the pegs, and its position was tracked until either two minutes passed or the front half of the robot left the outer boundary of the pegs. Ten trials were conducted for each controller, with the same nominal wave parameters ($\omega_{s,0} = 3\pi$, $A_0 = \frac{\pi}{4}$). The distance traversed was measured in terms of the smoothed arc length travelled by the averaged marker position. Performance was compared with a "time to traverse" metric, which measures the number of body wave cycles required to move one meter. Locomotive speed could be increased by increasing the nominal temporal frequency; normalizing by number of cycles removes this effect. Controllers that result in poor locomotion performance either through becoming trapped frequently or thrashing in place will have a high value, and those that progress smoothly will have a lower value.

Prior experiments identified the best anchor points for activation windows [47], and found that shape-based control performed best when the nominal window corresponded to one half wavelength, as shown in Figure 4.2. Further experiments compare the distributed-feedback CPG against the shape-based controller with either one, two, or three compliant parameters. Figure 4.6 presents the results of experiments comparing implementations of shape-based compliant and CPG-with-feedback control. The average performance for each controller is represented by the red bars with standard error. A highly successful controller will minimize both the time to traverse and the variance in the time to traverse, as large variance indicates that the robot was stuck in place rather than travelling smoothly.

Shape-based compliance outperformed the CPG controller, taking fewer cycles (less normalized time) on average to traverse the pegs. Shape-based compliance lowered both the average and variance of the time to traverse, and qualitatively made the path travelled smoother and more natural. This makes the robot's behavior more predictable and potentially easier to steer, either via a high level path planner or a remote human operator. The addition of spatial frequency compliance results in the snake "gliding" more naturally through the pegs, as its shape conforms more easily to the peg spacings. The addition of temporal phase compliance results in smoother gaits, because momentary jams cause a delay in phase while other parameters comply to the tight space. The data tend to be dominated by events in which the robot can not progress, which are heavily penalized via the time to traverse metric. In fully centralized and fully decentralized controllers, these events happen more frequently than for the best controllers tested in our full experiment.

Both the CPG and the shape-based control are locally compliant in modulating serpenoid amplitude and wavelength with damped-spring dynamics. However, shape-based control offers a number of advantages. A shape-based controller is able to propagate shape information along the shape curve, which would be difficult to implement with CPGs. The incorporation of joint torque feedback is more intuitive in shape-based control, as it provides a mapping function between joint and shape space. Both methods coordinate joints and can range in centralization, but shape-based control allows the level of centralization to vary, so the robot can comply to the environment while still maintaining shape structure. Shape-based control offers predictable, intuitive responses; joints will be coordinated within in a well defined shape even when significantly perturbed from its steady state trajectory. It is found that an intermediate level of centralization performed best for spatial parameters, but that a fully centralized level performed best for the temporal parameter. In the CPG approach, which is more decentralized in both spatial and temporal parameters, it is found that the waveform degenerated in highly constrained positions.

4.1.7 Discussion

This part of the work extended shape-based control to spatial frequency and temporal phase serpenoid shape parameters. With this controller, a snake robot navigated its way autonomously through unknown, irregular environment with only joint-level torque feedback. The shape-based framework provides intuitive ways to design behaviors and synthesize feedback. This approach works equivalently with any number of joints, as the joint angles are taken from a continuous shape curve, and is fully extensible to three-dimensional motion. The approach is purely a reactive control strategy, which does not require a physical model of the robot or the environment. Because our snake robot operates in friction dominated domains, a full model including contact between the robot and terrain has not been necessary, and may be computationally prohibitive. These results, without a full analytical model, are experimental in nature. A theoretical analysis on how shape parameter variations interact with robot and environment geometry within our method is a direction of future work.

Adding spatial frequency and temporal phase compliance resulted in a more natural, smooth motion, while decreasing the variability in locomotor performance. In the future it will be tested whether this is the case for a wider range of peg spacing, both irregular and regular. Simply making more parameters locally compliant within a shape does not always improve performance. Similarly, varying the centralization was beneficial for spatial parameters, but unproductive in temporal parameters. For a given terrain, there does appear to be a level of centralization, shape structure, and compliance that maximizes performance. In future work, it will be investigated how to select these factors from the environment and robot design. Additional sensing modes, such as contact sensors, IMUs, vision, and directional steering, will be incorporated to the controller.

4.2 Rolling Helix Gait Compliance

4.2.1 Introduction

The rolling helix gait is an useful form of motion used by snake-like robots to climb on poles or crawl through pipes. Since it is a particular case of the rolling gait, the snake robot is able to smoothly transition between the two gaits. This capability is extremely useful for the snake locomotion, since the robot is able to transport itself close to an obstacle using the rolling gait and then climbing it switching to the rolling helix shape. Unlike other gaits, both the rolling and the rolling helix are not inspired by biological snake; they are specifically crafted exploiting the hyper-redundant architecture of the robot to achieve specific tasks, such as moving to locations above the ground otherwise unreachable, or crawl inside pipes inaccessible by humans or other devices. The gait can be intuitively controlled by an operator, who steers and tightens the snake using few parameters. In this section it is proposed a controller that autonomously governs the climbing behaviour of the snake, tightening it around an obstacle enough for the snake to climb it without falling. Additionally, the controller continuously regulates the helix radius to adapt to a possibly variable diameter of the climbed obstacle without slipping. The conceived controller shares similarities with the method proposed in [50].

It is shown how a relatively simple control structure allows the snake robot to autonomously conform to the feature that is being climbed in response to external forces sensed. Adjusting the radius of the rolling helix results in a change in the tightening force the snake exerts on the obstacle, making the gait suitable for many circumstances, e.g. fragile or robust objects or in presence of a slippery, rough or mixed texture.

4.2.2 Background

The rolling helix is a gait developed for the snake robot to climb on pipes or poles, as described in [51]. It is derived from the parametric equations of the helix:

$$\begin{aligned} x &= r \cdot cos(\theta) \\ y &= r \cdot sin(\theta) \\ z &= p \cdot \theta \end{aligned} \tag{4.9}$$

where r is the radius and $2\pi p$ is the pitch of the helix (i.e. the vertical separation of the helix's loops).
The curvature κ and the torsion τ depend both on the radius and the pitch of the helix:

$$\begin{cases} \kappa(s) = \dfrac{r}{r^2 + p^2} \\ \tau(s) = \dfrac{p}{r^2 + p^2} \end{cases} \tag{4.10}$$

The radius of the helix r can be adjusted by the user with a joystick while the snake robot is climbing pipes of different sizes.

4.2.3 Radius Compliance

In the previous project (Section 4.1) shape-based structure was used to make the slithering gait compliant in an unstructured environment. The same controller is extended and implemented for the rolling helix gait, to achieve an autonomous compliant behaviour while climbing.

Shape-based architecture is highly adaptable to different configuration of the robot (i.e. gaits) since it depends on specified shapes. A good shape basis is found such that the many degree of freedom are well coordinated. Referring to the equation (3.1), both the amplitudes A_{lat}, A_{dor} and both the spatial frequencies $\omega_{S,lat}$, $\omega_{S,dor}$

in the lateral and dorsal plane of the serpenoid are chosen as shape parameters σ and allowed to be compliant:

$$\sigma = \{A_{lat}(t), A_{dor}(t), \omega_{S,lat}(t), \omega_{S,dor}(t)\} \tag{4.11}$$

The two parameters are linearly dependent, so that radius and pitch change simultaneously to keep a ratio that define a effective helix shape for climbing. Specifically, $\omega_S(t) = k \cdot A(t) = \omega_{S,lat}(t) = \omega_{S,dor}(t) = k \cdot A_{lat}(t) = k \cdot A_{dor}(t)$ where k is a defined fixed constant and the radius and the pitch (both for the lateral and the dorsal plane) are defined as:

$$\begin{cases} p & = \frac{1}{\left(\left(\frac{A}{(2sin(\omega_S m)}\right)^2 + 1\right) \cdot \omega_S} \\ r & = \frac{A}{2sin(\omega_S m)} \cdot p \end{cases} \tag{4.12}$$

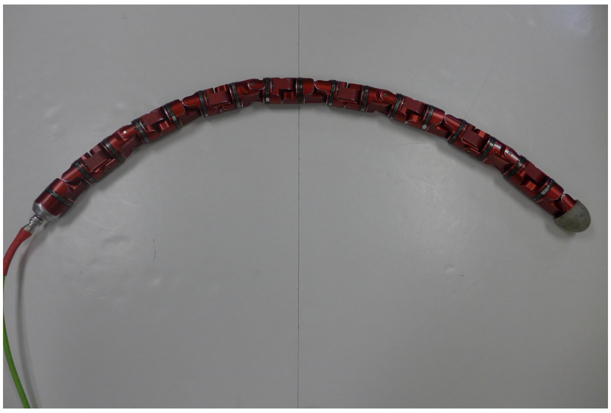

Figure 4.7: Shape of the rolling gait for the *SEA Snake*.

The initial configuration of the snake is the rolling gait: when the rolling helix gait is issued the radius is attracted with second-order dynamics to the nominal radius defined by $\sigma_{r,0}$, which results in the snake starting to wind up in a helical shape. If an external force is applied by the pole (i.e. the radius of the pole is increasing), the compliant parameter $\sigma_{r,d}$ responds varying around the nominal set point $\sigma_{r,0}$, changing the radius (and consequently the pitch) of the helix.
In the particular case of the rolling helix gait is added the term τ_D, a desired virtual force, to the admittance control:

$$M_r^\sigma(\ddot{\sigma}_{r,d}(t)) + B_r^\sigma(\dot{\sigma}_{r,d}(t)) + K_r^\sigma(\sigma_{r,d}(t) - \sigma_{r,0}) = \tau_\sigma(t) + \tau_D \tag{4.13}$$

where τ_r is the projection of the external forces τ_{ext} in the shape space by the Jacobian $J_r(h)$, the mapping term defined in (3.12). The virtual force τ_D is user

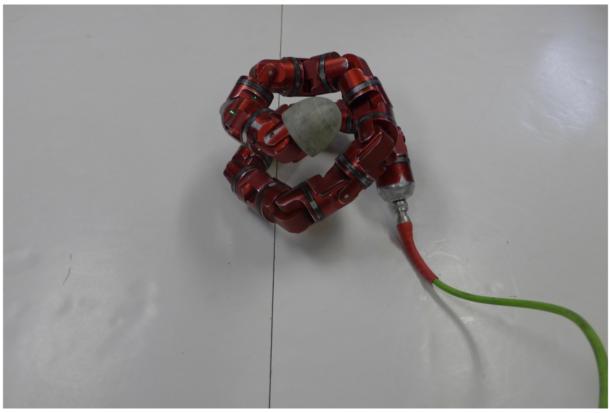

Figure 4.8: Shape of the helix gait for the *SEA Snake*.

defined, and it "squeeze" the snake until the force exerted by the coils is enough to get the necessary friction on the pole to climb it. It can be adjusted depending on the material of the object climbed and the friction desired.

4.2.4 Variable Decentralization

To better comply with changes in the diameter of the pole, it is desirable to allow different portions of the snake's body to adapt independently to local features. To achieve decentralization the radius parameter is changed separately along the snake. Similarly to the last project, this is accomplished by positioning activation windows along the coils of the snake's robot, so that only the joints covered by one window are coupled. The independent windows are defined for the shape parameters A and ω_S by using sigmoid functions:

$$\begin{aligned} A(s,t) &= \sum_{j=1}^{W} A_j \left(\frac{1}{1+e^{-m(s-s_{j,start})}} + \frac{1}{1+e^{m(s-s_{j,end})}} \right) \\ \omega_S(s,t) &= \sum_{j=1}^{W} \omega_{S,j} \left(\frac{1}{1+e^{-m(s-s_{j,start})}} + \frac{1}{1+e^{m(s-s_{j,end})}} \right) \end{aligned} \tag{4.14}$$

where m controls the length of the transition period between shape parameters in neighboring windows, W is the number of windows, window j spans the backbone over arc length $(s_{j,start}, s_{j,end})$.

Windows are made so that the start and end position of each one are anchored to specific locations on the base of the serpenoid curve. The anchor points are controller design variables, and they can be defined, for instance, so to cover half length of one coil of the helix. Since the radius and therefore the length of each coil of the helix is compliant, windows change size according to it. Consequently the number of windows on the snake is allowed to change, and a window manager subroutine calculates these changes by maintaining an array of window lengths and positions. As shown in Figure 4.9, at time $t = 0$ the snake robot is in the starting position and only one window is covering the whole snake. At time $t = 1$ the snake robot is coiled up in a helical shape with a smaller radius r and more windows appear on the snake, one for each half length of a helix's coil.

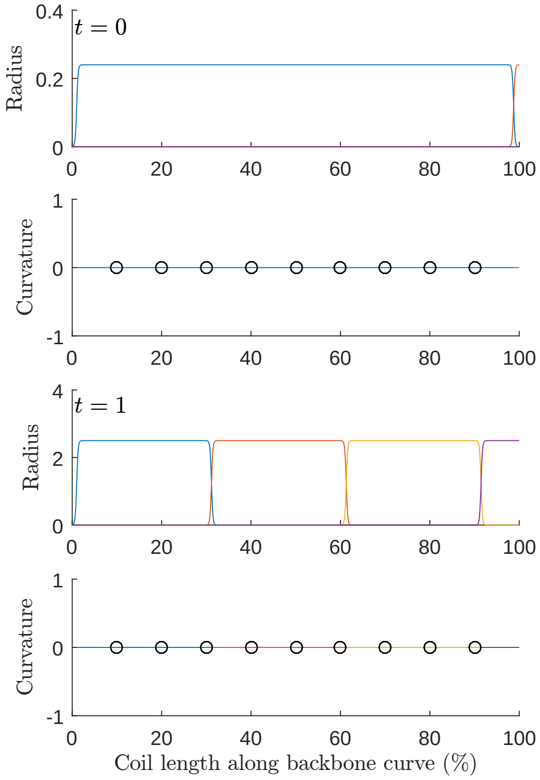

Figure 4.9: An illustration of activation windows, which locally modulate the radius r along the backbone of the snake. The window edges are linked to half the length of one helix's coil. Black circles represent joints, which lie at fixed locations along the backbone arc.

4.2.5 Adaptive controller

An adaptive control is implemented on top of the admittance control. The controller governs the set point $\sigma_{r,0}$ of each window in response to the forces sensed by the modules. If the sum of the forces in one window τ_r drops under a certain threshold, the set point increases. Similarly, if the sum of the forces rises to a specified threshold, the set point decreases. Hence, the nominal radius, i.e. the attractor in the dynamic system, changes in time slowly conforming to the environment. While the compliant controller is responding to sudden variations, the nominal shape parameter r_0 slowly adapts to the diameter of the pole based on the force sensed by each module.

4.2.6 Full controller

For a snake robot with N links, joint angles $\theta_i \in \mathbb{R}^N$, and W decentralized windows, the compliant controller for the helix rolling gait consists of a basis function based on the serpenoid curve (3.6) adapted for the specific helix rolling gait with shape parameters $\sigma_{r,d}$, a $W \times 1$ vector which consists of one parameter for each independent section:

$$\sigma_r = \{R(s,t)\}^T \tag{4.15}$$

where $R(s,t) = k \cdot A(s,t) = \omega_S(s,t)$. The controlled variable is decentralized as shown in (4.14):

$$R(s,t) = \{R_1(s,t), R_2(s,t), \dots, R_W(s,t)\} \tag{4.16}$$

which correspond to the shape parameters:

- $A(s,t) = \{A_1(s,t), A_2(s,t), \dots, A_W(s,t)\}$
- $\omega_S(s,t) = \{\omega_{S,1}(s,t), \omega_{S,2}(s,t), \dots, \omega_{S,W}(s,t)\}$

Hence, the full shape functions on the lateral and the dorsal plane are:

$$\begin{aligned} h_i^{lat}(A(s,t), \omega_S(s,t)) &= A(s,t)\cos(\omega_S(s,t)s_i^{lat} - \omega_T t) \\ h_i^{dor}(A(s,t), \omega_S(s,t)) &= A(s,t)\sin(\omega_S(s,t)s_i^{dor} - \omega_T t), \end{aligned} \tag{4.17}$$

where $A(s,t)$ and $\omega_S(s,t)$ are equal on both planes. The shape-based compliant controller is:

$$M_r^\sigma(\ddot{\sigma}_{r,d}(t)) + B_r^\sigma(\dot{\sigma}_{r,d}(t)) + K_r^\sigma(\sigma_{r,d}(t) - \sigma_{r,0}) = F_\sigma(s,t) + \tau_D \tag{4.18}$$

where $M_r^\sigma, B_r^\sigma, K_r^\sigma \in \mathbb{R}^{W \times W}$ are respectively the effective mass, damping and spring constant matrices of the system. The term $\sigma_{r,0}$ represents the nominal shape parameters $\sigma_{r,0} = \{R_0\}^T$. $F_\sigma(s,t) \in \mathbb{R}^{W \times 1}$ is a mapping of the external forces $\tau_{ext}(s,t) \in \mathbb{R}^N$ from joint space $Q \in \mathbb{R}^W$ to shape space $\Sigma \in \mathbb{R}^N$ as seen in (3.14), where the mapping function is the Jacobian matrix $J(h) \in \mathbb{R}^{N \times W}$ defined in (3.12). In particular,

$$J_{\sigma}^{T}(h) = \begin{pmatrix} \frac{\partial h_1(R)}{\partial R_1} & \dots & \frac{\partial h_N(R)}{\partial R_1} \\ \vdots & \ddots & \vdots \\ \frac{\partial h_1(R)}{\partial R_W} & \dots & \frac{\partial h_N(R)}{\partial R_W} \end{pmatrix} \tag{4.19}$$

Finally, the joint angles θ_i are computed at each time step t using (4.1).

4.2.7 Experimental results and discussion

Shape-based control was developed on the *SEA Snake* for the rolling helix gait, having as shape parameters the amplitude and the spatial frequency in both planes (from which the radius and the pitch of the helix depend). Decentralization was implemented using windows for each half of the helix's coil. The robot was covered with a braided polyester sleeve or padded with rubber to increase friction. Experiments were conducted making the *SEA Snake* climbs different poles with variable radius, starting from the initial configuration of the rolling gait.

Thanks to the shape-based compliance together with the rudimental adaptive control, the snake can adapt and comply locally to changes in the diameter of the pole without slipping or falling. Decentralization is necessary to overcome local variation in the diameter, since different sections of the snake need to comply differently to the environment. The squeezing virtual force τ_D is user defined: a controller could be implemented to adjust this term depending on the particular situation (such as local friction between the snake and the object, weight of the snake, segment of snake exerting force on the environment if part of it can detach from the pole for different purposes).

The Helix Rolling gait was formerly implemented for joystick-controlled snake robots. The user could roll the snake for locomotion and tune the helix radius while climbing on obstacles. Exploiting the force sensors in the *SEA Snake* and a shape based architecture, this work proposed a new way to make the snake autonomous. Sensing the forces exerted by the environment, the mechanism can change its own shape while climbing, effectively tightening or loosening its grip on the obstacle. Furthermore, for a joystick-controlled snake, the ability to autonomously comply to variation in diameter of the object make this gait even more intuitive and simple for the user, reducing the number of parameters to control.

4.3 Sidewinding Gait Compliance

4.3.1 Introduction

Sidewinding is one of the most efficient transitional gait the snake robot uses to locomote through different terrains. However, when used to autonomously maneuver in a cluttered environment, classic sidewinding turns out to be less effective. To address this problem, two architecturally simple schemes are presented that control both the heading and the shape of the snake robot while sidewinding, in order to adapt its body to the environment in real-time without losing a given direction of travel. The first control strategy simulates biological snake's proprioception, i.e. the perception of its own body locomotion. The core element of this architecture is a bio-inspired strategy, based on a simple dynamical system approach, that mimics the behaviour of biological snakes when faced with a set of obstacles. Parallel to it a controller continuously adjust the snake heading, steering it back to the nominal direction. The second control structure, based on a modified admittance controller, adapts the snake robot's body to the environment exploiting the *shape-based architecture*, which provides an intuitive way to coordinate the many joints of the hyper-redundant robot, limiting the complexity of the control structure and the number of exteroceptive sensors required. Specifically, previous works is extended on shape-based compliance defining an anti-compliant behaviour for the robot to react against the obstacles, emphasizing how a biological inspired locomotion is found in between the concept of complying and reacting to the external environment.

This work propose different versions of the controllers and experimentally compare them against the open-loop sidewinding gait. It is found that simple schemes for anti-compliance and simulated proprioception are enough for the snake-like robot to easily locomote and overcome obstacles.

Depending on the environment and the terrain's features they are deployed in, these hyper-redundant mechanisms can assume different locomoting gaits, in order to move efficiently [45]. Sidewinding is one of the many gaits, formerly inspired by the sidewinding biological snakes [37, 11]. Differently from many others, this gait is fully independent from environmental features, relying only on internal shape changes to locomote. As a consequence, it is used by snakes mainly where ground does not provide the necessary reaction forces for other locomotion gaits, such as rocky or slippery terrains, proving to be one of the fastest gait. However, while it is robust and efficient in open spaces, when the snake robot is faced with obstacles this gait loses its effectiveness.

Hence, inspired by the proprioception showed by biological snakes and the ability to deform their sidewinding shape to adapt to and overcome cluttered terrains, it is proposed a way to deal with obstacles while retaining the direction of motion.

The work emphasize how a straightforward, limited set of controllers is important to grant reliability, robustness and the ability to react efficiently in real-time to the environment while moving. To these ends, a high-level heading control is combined with mid-level shape adaptivity. The high level controller defines a motion that closely resemble one of the many techniques used by sidewinding biological snakes to pass through obstacles. By pushing against one while moving in a straightforward sidewinding, the snake gets diverted and ends up oriented in the opposite direction. Thereupon it fluidly inverts its motion and continues sidewinding on the same initial direction.

The proposed controller is based on a simple dynamical system approach to switch between behaviours that govern the snake direction without resorting to a finite state machine and discrete changes, keeping the level of complexity low and the motion fluid. Effectively, it is designed a bistable system that uses the readings from the inertial measurement units (IMU) on each snake's modules, i.e. the orientation of the reference frame fixed to the snake, to regulate a single parameter that smoothly inverts the snake motion.

Parallel to the switching direction control, an heading adjustment is implemented that uses the same readings and exploits conical sidewinding [52] to correct small changes in the direction, in order to keep a robust straight sidewinding movement.

The underlying structure of the project, shape-based architecture, provides a mean to straightforwardly link the orientation of the snake with a single parameter of the snake's shape. Mid level shape adaptivity relies on the same architecture. Prior work showed how shape-based control can be exploited as a powerful tool to control the snake intuitively, linking high level planning to low level control [47]. The main advantages are a few directly controlled parameters compared to the high number of degrees of freedom (DOF), a straightforward method to use the feedback from the snake's sensors, and an architecture to decentralize the control and propagate information along the snake's body. The shape-based compliant scheme is extended introducing anti-compliance. While adapting and conforming to the environment can be useful for some locomotion gaits, in this particular case it is shown how reacting against the environment is more effective, especially in the case of a self-locomoting gait that does not need environment to move forward (differently to slithering gait): instead of adapting and conforming to it, the snake robot deform its body in order to push back and reacting against sensed forces. Furthermore, as presented in [49], it is found that a layered decentralization is beneficial for the sidewinding snake locomotion in a cluttered environment. A blind snake moving through an unstructured terrain requires different portions of the snake to respond differently to external forces. Since many parameters are controlled, different approaches can be taken. In this work, it is exploited a variable coordination centralization and a fully centralized controller depending on the specific parameter, such that one varies independently on different portions of the body while the other changes on the body as a whole.

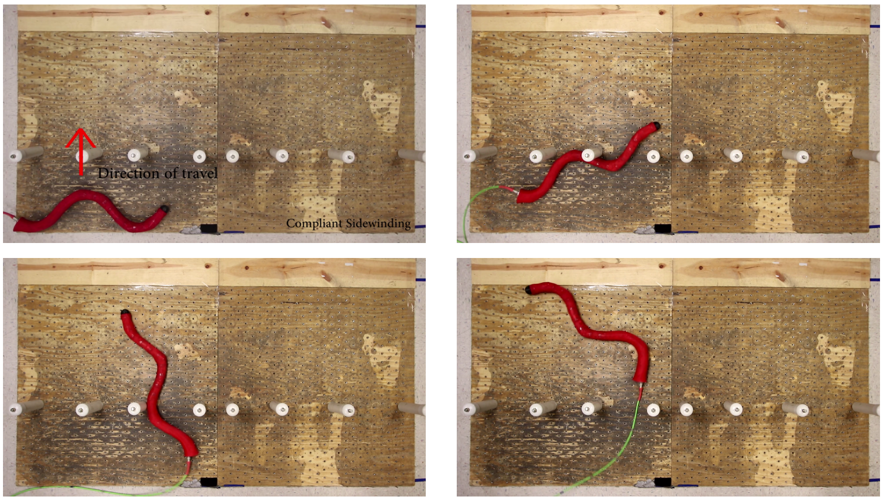

Figure 4.10: Sequence of the *SEA snake* passing through a set of pegs.

4.3.2 Complementary filter and Virtual Chassis

The framework used for this project is based on the previous elements in the robot control literature presented in Chapter 3. An additional preexisting method, fundamental for this part of the work, is briefly reviewed in this section.

The presented method uses inertial measurement units (IMUs) to estimate the snake's pose $T \in SO(3)$. In particular for modular snake's robot (e.g. *SEA Snake*, Unified Snake) that possess an IMU for each module, the high number of measurement can be used to get a refined estimation. However, a filtering technique is necessary when using IMUs. As a matter of fact, the accelerometer data is reliable on the long term, but it is very susceptible to external forces, causing the readings to be very noisy. Similarly, the gyroscope data is reliable on the short term, but it starts to drift on the long term.

The Complementary Filter is a simple way to address these problems in order to estimate correctly the orientation of a moving body. It merges the readings from accelerometers and gyroscopes to create a single and accurate angles estimation.

The Virtual Chassis is a method that implicitly uses the Complementary Filter to define the snake's body coordinate frame presented by Rollinson and Choset [53]. Instead of representing the system in a body frame static to a certain fixed position along the links (i.e snake's modules), it defines a body frame that is intuitive when considering the position and the orientation of the system as a whole.

The Virtual Chassis reference frame is defined at each instant of time aligning it with the principal moment of inertia taken at the kinematic center of mass of the robot. In particular, the kinematic center of mass is calculated by averaging the position of all the modules in an arbitrary reference frame, while the principal

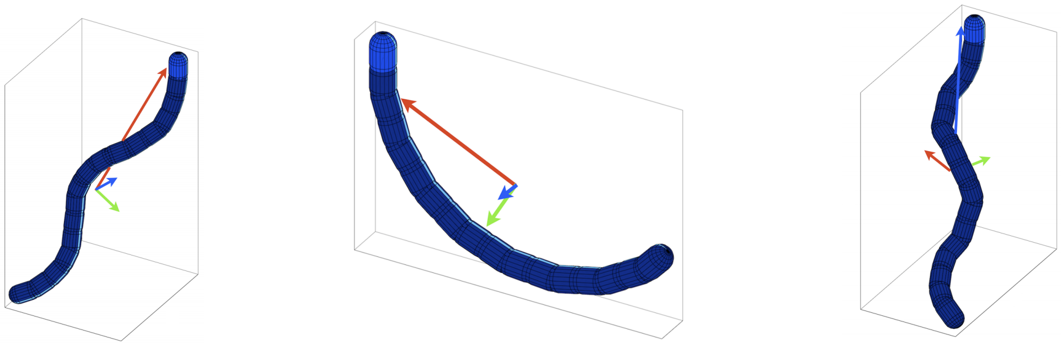

Figure 4.11: Virtual Chassis for different gait of the snake robot. (Source: [53]).

moment of inertia is determined through the Singular Value Decomposition (SVD) of the zero-mean position of all the modules (i.e. the position in a reference frame fixed at the kinematic center of mass).

The Virtual Chassis is particularly effective when the shape of the robot is symmetrical and cyclical in its motion, that is, the robot's main axis is well defined at each instant of time. If the shape's geometry get somehow disrupted the VC method cannot distinguish anymore a main axis and fails to estimate the pose.

4.3.3 Inertial turning

Inspired by the movement of a sidewinder snake through a set of pegs, a method to mimic its proprioception is presented. The biological snake deforms its body and turns around the obstacle in order to pass through an opening, and smoothly resumes sidewinding in the initial direction of movement. Starting from this observation, it is fixed a nominal direction which the snake has to follow while sidewinding.

The nominal direction of locomotion, i.e. the desired orientation of the snake's body coordinate frame fixed at the kinematic center of mass, is provided by the Virtual Chassis. The VC takes as a input (filtered by the Complementary Filter) the signals of the IMUs on each module. Since the snake robot is an hyper redundant mechanism, the high number of IMUs allows for a precise estimation.

The snake has to retain such direction adjusting it if it's diverted by environmental features. Two controllers are developed and implemented on the robot:

Direction reversal: The first controller invert the motion of the snake (from forward sidewinding to backward sidewinding) when its heading is reversed;

Heading Adjustment: The second controller adjust the heading if the snake diverts from the nominal direction.

Both controllers requires the estimate of the robot orientation $T \in SO(3)$ at each instant of time.

Direction reversal

The reversal is the core controller of the snake robot: the snake eventually enters the peg array, being pushed by the obstacle while travelling forward and ending up oriented on the opposite direction. A dynamical switch smoothly reverses the snake motion to resume moving on the nominal direction.

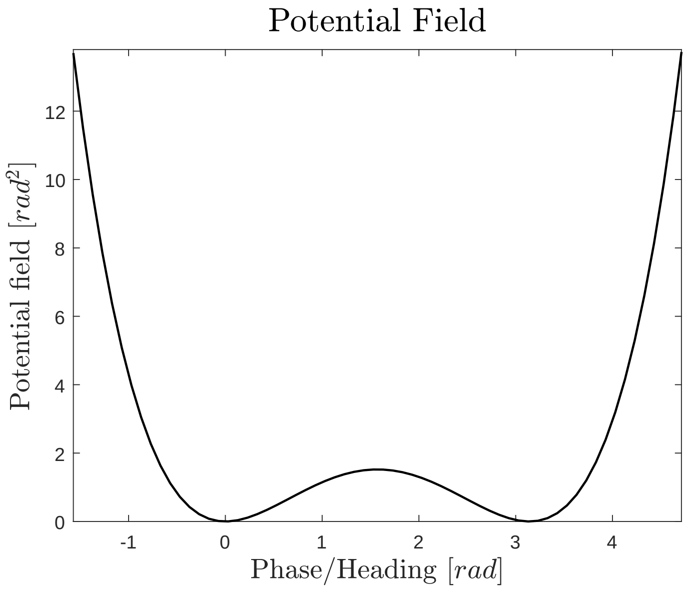

Figure 4.12: Potential landscape of the bistable dynamical system with two stability points at $x = 0$ and $x = \pi$.

The movement and the direction of the sidewinding gait are given by the sections of the snake in contact with the ground. By varying the offset β between the two serpenoid curves on the lateral and dorsal plane, it is possible to change the raised portions and contact points with the ground, effectively changing direction.

$$\begin{cases} \theta_{i\,lat} = A_{lat} \sin(\omega_{S\,lat} s_{i\,lat} - \omega_{T\,lat} t) & lateral \\ \theta_{i\,dor} = A_{dor} \sin(\omega_{S\,dor} s_{i\,dor} - \omega_{T\,dor} t + \delta + \beta) & dorsal \end{cases} \tag{4.20}$$

It is built a dynamic system in which the offset β is a function of the heading h. The term δ is a constant offset characteristic of sidewinding ($\frac{3}{4}pi$), creating the necessary offset between the to waves that allows a straight locomotion.

$$\dot{\beta} = K_b(\beta - \beta_0)(\beta - \beta_1)(\beta - \beta_2) + K_h(h - \beta) \tag{4.21}$$

A bistable potential landscape of the system is built, (i.e. a potential function consisting of two wells as depicted in Figure 4.12), where β_0 and β_2 are the stable

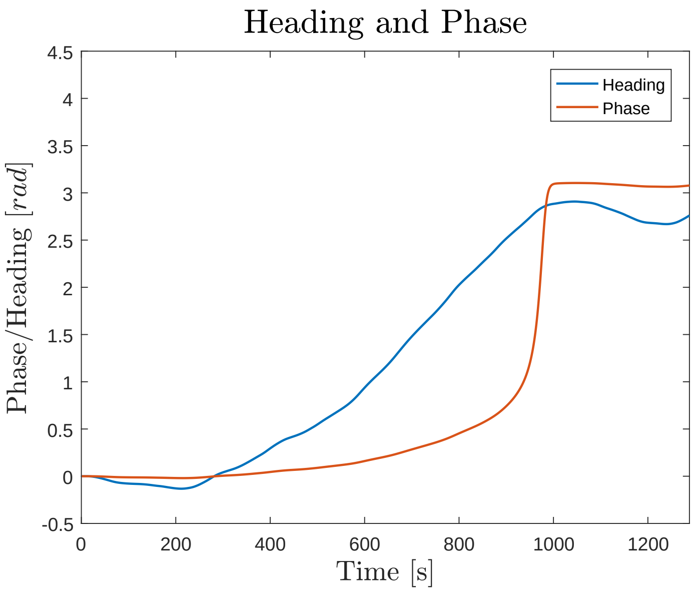

Figure 4.13: Evolution of the heading and the phase in time.

equilibrium points and β_1 is the unstable equilibrium point between the two. The K_b is a degree of freedom that shapes the bistable landscape, while K_h is a parameter that define how strongly h and β are related. The system initial state of the variable β is on the first attractor 0, which corresponds to a forward sidewinding. Minor perturbations of the heading h of the snake do not affect the parameter β, which is attracted by the first stable point. Since the two variables are linked by $K_h(h-\beta)$, if the heading h considerably changes, it drags the offset β along as if a spring was between the two, eventually pulling it over the unstable point. As soon as the offset β climbs over, it starts being attracted by the second stable point π, inverting the snake robot motion to a backward sidewinding. Since a slow change in the phase of the serpenoid curve would produce macroscopic changes in the snake's shape, potentially hindering the sidewinding motion, the relationship between heading h and offset β is chosen to be not linear, so that the phase would rapidly switch from one potential well to the other. A sequence of the heading and the phase in the potential well while the direction is being reversed is shown in Figure 4.14. At time $t = 1$ the heading and phase are in the first well ($x = 0$). At time $t = 2$ the snake is being pushed by the pegs, and start changing direction, which result in a change in the heading. At time $t = 3$ the heading is in the second well, and the phase β is being dragged over the potential hill by the heading. At time $t = 4$, β is in the second well ($x = \pi$) and the sidewind motion is reversed. In Figure 4.13 is shown the same snake's reversal in terms of phase and heading evolving in time.

Since the dynamic system is bistable, after the first inertial turning, i.e. the

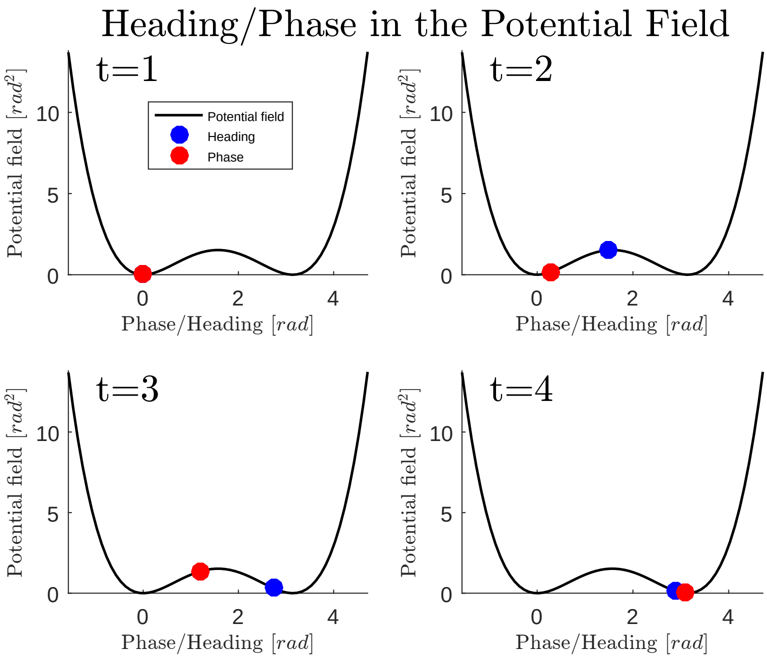

Figure 4.14: A sequence of the heading and the phase of the *SEA Snake* in the Potential Field.

offset β is in the second potential well, the snake wouldn't be able to turn around again. Hence β is made to be modular to wrap it around the bistable region: when it is pushed over the second well, it starts over from the first one, making the inertial turning repeatable in time.

Heading adjustment

A different controller autonomously adjusts small variations in the snake robot direction. To this end, it exploits the conical sidewinding [52].

The reduced model of sidewinding is geometrically intuitive: a helical tread (representing the discrete snake's backbone) wrapped around a virtual elliptical cylinder. As the cylinder moves forward and the tread rolls with it, the gait advances. Similarly the reduced model of conical sidewinding is a helical tread wrapped around the surface of a cone. As the cone rolls, portions along the cone's body are faster than others, and its axis rotates in the plane drawing a circle centered at the apex. Hence the snake follows a curved trajectory with a radius of curvature determined by the taper of the cone. Since the snake has to follow a constant direction while locomoting through a cluttered environment, it continuously adjusts its heading steering back to the nominal direction. Whenever the orientation (provided by the Virtual Chassis) is different from the nominal one, the snake switch from the sidewinding gait to the conical sidewinding gait, tapering the amplitude of the serpenoid on one half of the robot's body and resuming sidewinding when the direction is correct.

4.3.4 Shape-based anti-compliance

Together with the other gaits presented, a compliant controller is devised for the sidewinding gait to achieve autonomous compliant behaviors that allows a snake robot to adapt in real-time to changes in its environment.

In the literature of autonomous robot locomotion many approaches and solutions are found to maneuver through a cluttered environment. Some strategies are based on avoiding completely the obstacles, while others find their way through adapting to them. Many motions even benefit from the presence of obstacles [21], up to the extent where some robots can only move thanks to clutter, pushing against or pulling in order to move [49]. In the case of the sidewinding gait, the snake transports itself without the assistance of the environment. This work supports the idea that a simple control scheme is enough for an efficient movement, limiting complexity, computational burden and the number of exteroceptive sensors. To this end, the shape-based framework is exploited, extending the work in [47], for the sidewinding gait.

The serpenoid curve equation is rewritten in term of shape function. In particular, the wave-like function on the lateral plane of the snake robot is defined as a shape basis:

$$g = A_{lat} \sin(\omega_{S\,lat} s_i - \omega_{T\,lat} t) \tag{4.22}$$

Spatial frequency $\omega_{S\,lat}$ and amplitude A_{lat} on the lateral plane are selected as shape functions, such that amplitude of curvature and number of windows along the backbone are allowed to change, effectively making the static shape fully compliant.

The control scheme makes use only of force sensors located on each module while locomoting. Hence, when an obstacle is detected, i.e. when the snake robot comes in contact with it, it pushes back and propels forward. It is shown how anti-compliance is beneficial to this task: when an external force is sensed, the snake robot reacts to it exerting an opposite force, hence pushing back. Specifically, the controlled spatial frequency parameter squeezes the snake, adapting the wavelength of the serpenoid to the openings, while the controlled amplitude parameter pushes the obstacles propelling forward the snake. Starting from the shape-based compliance presented in Section 3.4 formulated as (3.13), the equation for anti-compliance is modified:

$$M_\sigma(\ddot{\sigma}_d) + B_\sigma(\dot{\sigma}_d) + K_\sigma(\sigma_d - \sigma_0) = -F_\sigma \tag{4.23}$$

When a force is applied to the snake's body, the torques sensed by the modules are mapped into a shape force, resulting in a change in the amplitude and wavelength of the serpenoid curve. When forcing is removed, the wavelength and amplitude return to their nominal values. While a compliant controller acts in accordance with external forces, an anti-compliant controller pushes back if a force is applied.

4.3.5 Decentralization

It is hypothesized that a decentralization is advantageous for a blind snake that has to move through an unknown terrain, so that different portions of the snake can react independently. It is shown how a different degree of centralization between the shape parameters result in a more effective and reliable motion. The desired behaviour lies in between a two extremes, a rigid motion that does not allow any interaction with external features, and a complete loss of the underlying shape, which would result in the inability to move forward. That is, a nominal cyclic movement is stretched and modified by the environment without losing a fundamental coordination between joints and a mid level coordination between different portions of the snake.

Specifically spatial frequency is fully centralized, such that any force felt by the snake robot changes the whole body's spatial frequency parameter. Amplitude is variably decentralized, depending on the spatial frequency of the serpenoid curve (i.e. how many waves on the snake). Each half wavelength is amplitude independent. Moreover, the windows that define the independent segments are not fixed on the snake's backbone. Instead, they are moved along the snake at the same velocity of the serpenoid wave, in order to pass back the information down the body (i.e. the deformation of the snake body's shape), following the wave that propagates from head to tail.

Decentralization is achieved, as presented in [49], using activation windows positioned along the backbone curve of the snake. Only the joints covered by the same window are coupled, isolating them from other joints. This approach allows to vary the degree of control decentralization depending on the number of windows present on the snake's body.

Independent windows are defined by using sigmoid functions:

$$A(s,t) = \sum_{j=1}^{W} A_{s,j}\left(\frac{1}{1+e^{-m(s-s_{j,start})}} + \frac{1}{1+e^{m(s-s_{j,end})}}\right) \tag{4.24}$$

where m controls the length of the transition period between shape parameters in neighboring windows, W is the number of windows, window j spans the backbone over arc length $(s_{j,start}, s_{j,end})$.

4.3.6 Full controller

The full controller for the sidewinding gait consists of a shape-based anti-compliant part and a dynamical system part based on inertial measurements.

For a snake robot with N links with joint angles $\theta_i \in \mathbb{R}^N$, and W decentralized windows, the shape parameters σ_g are the amplitude A and the spatial frequency ω_S:

$$\sigma_g = \{A(s,t), \omega_S(t)\}^T \tag{4.25}$$

where σ_g is a $W + 1 \times 1$ vector, the amplitude parameter is decentralized as shown in (4.24) being a $W \times 1$ vector that consists of one parameter for each independent section on the snake $A(s,t) = \{A_1(s,t), A_2(s,t), \dots, A_W(s,t)\}$ and the spatial frequency is a fully centralized parameter, i.e. a time dependent scalar $\omega_S(t)$ along the whole snake's body. The resulting shape function (of the lateral plane) is:

$$h_i(A(s,t), \omega_S(t)) = A(s,t)\sin(\omega_S(t)s_i - \omega_T t), \tag{4.26}$$

and the full shape-based compliant controller reads:

$$M_g^\sigma(\ddot{\sigma}_{g,d}(s,t)) + B_g^\sigma(\dot{\sigma}_{g,d}(s,t)) + K_f^\sigma(\sigma_{g,d}(s,t) - \sigma_{g,0}) = F_\sigma(g,t) \tag{4.27}$$

where $M_g^\sigma, B_g^\sigma, K_g^\sigma \in \mathbb{R}^{(W+1)\times(W+1)}$ are respectively the effective mass, damping and spring constant matrices of the system. The term $\sigma_{f,0}$ represents the nominal shape parameters $\sigma_{f,0} = \{A_0, \omega_{S,0}\}^T$. $F_\sigma(s,t) \in \mathbb{R}^{(W+1)\times 1}$ is a mapping of the external forces $\tau_{ext}(s,t) \in \mathbb{R}^N$ from joint space $Q \in \mathbb{R}^N$ to shape space $\Sigma \in \mathbb{R}^{W+1}$ as seen in (3.14), where the mapping function is the Jacobian matrix $J(h) \in \mathbb{R}^{N\times(W+1)}$ defined in (3.12). In particular,

$$J_\sigma^T(h) = \begin{pmatrix} \frac{\partial h_1(A,\omega_S)}{\partial A_1} & \dots & \frac{\partial h_N(A,\omega_S)}{\partial A_1} \\ \vdots & \ddots & \vdots \\ \frac{\partial h_1(A,\omega_S)}{\partial A_W} & \dots & \frac{\partial h_N(A,\omega_S)}{\partial A_W} \\ \frac{\partial h_1(A,\omega_S)}{\partial \omega_S} & \dots & \frac{\partial h_N(A,\omega_S)}{\partial \omega_S} \end{pmatrix} \tag{4.28}$$

Finally, the joint angles θ_i are computed at each time step t using (4.1).

As for the inertial controller, a initialization for the pose estimation is needed; before the snake is started, the readings from the IMUs are exploited to estimate the initial pose of the robot. Once a reference frame is found for one the joints (usually the head module) by merging the IMUs reading and exploiting the knowledge of the robot kinematics, the Virtual Chassis, via SVD, finds the main inertial axis of the robot, centers the found reference frame on the center of mass and aligns it to the main inertial axis, effectively returning a pose of the snake $T = \{T_x, T_y, T_z\} \in SO(3)$ in the world frame. The inertial controller requires only T_z, which, for the sidewinding gait, corresponds to the snake's direction of motion.

While the snake robot is moving the current direction of the robot h is estimated in the same way, by averaging the readings from the IMUs for each cycle of the periodic gait to make the estimate more robust and using the VC to define the robot's pose w.r.t. the world frame $T(t) = \{T_x(t), T_y(t), T_z(t)\}$. By using it in the dynamical system described in (4.21), h governs the dynamics of the parameter β, which represent the phase offset used to reverse the robot's locomotion (by changing the section of the sidewinding gait in contact with the ground).

4.3.7 Experimental comparison and results

A set of experiments was conducted using different versions of the controller to pass though pegs while sidewinding. The robot used in this work is the *SEA Snake*, composed of eighteen identical series-elastic actuated modules, each one rotated by ninety degrees with respect to the previous module's axis, to achieve an alternate orientation in the lateral and dorsal plane of the robot. IMUs on each module were used in order to estimate the orientation, while torque sensors were exploited by the modified admittance controller. The gait speed, defined by the temporal frequency, was the same for all the trials. The robot was covered with a braided polyester sleeve in order to smooth the snake body, removing sharp edges that could hinder the movement. Starting with the full controller (i.e. spatial frequency and amplitude), one controller parameter is removed at a time, up to the open loop version:

- Spatial Frequency and Amplitude Compliant (AC-SFC)
- Amplitude Compliant (AC)
- Spatial Frequency Compliant (SFC)
- Open Loop (OL)

The chosen environment for the experiments was a peg board with an array of irregularly placed pegs. Twenty one experiments were conducted for each controller. Specifically, three experiments each for seven different starting positions with respect to the same peg array. Performance was compared considering the first window the snake went through and the time it took to go past the peg array. The number of windows was counted from the first possible window inside which the snake could enter. Controllers that resulted in poor locomotion performance either passed through one of the latest windows or failed to pass through.

The graphs in Figure 4.15 and Figure 4.16 present the experimental results. The first shows the mean of the number of opening passed over by the snake before entering the peg array and reversing, and the number of failed attempts to pass through, i.e. number of times the snake leaves the peg board without entering in any opening.

The second figure shows the mean of the required time for the snake up to the moment where it reverses its motion beyond the peg array.

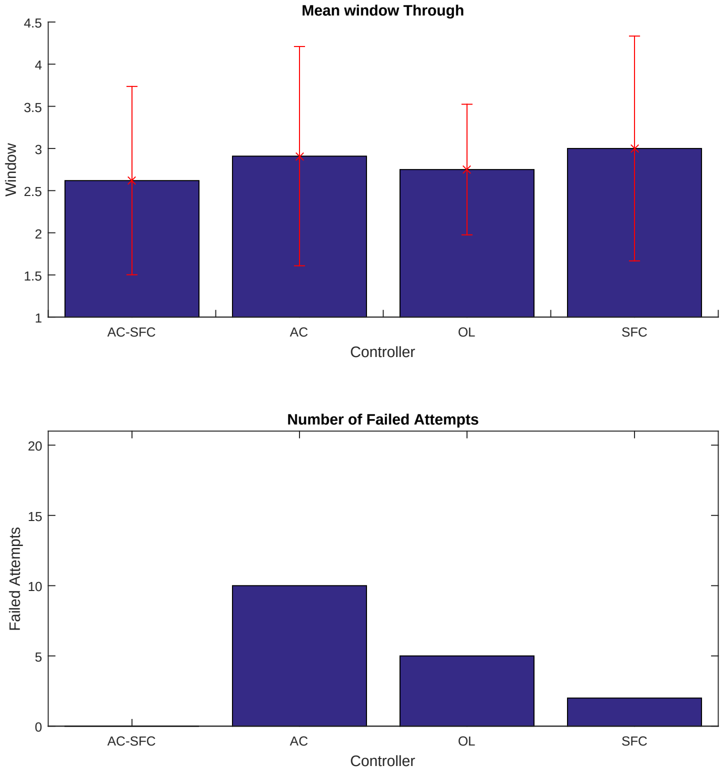

Figure 4.15: Comparison of different controllers for a snake robot passing through a set of pegs. The first graph shows the mean of number of window the snake pass by before passing through. The second graph shows the number of failed attempts for each controller. The red bars indicate the standard deviations.

4.3.8 Discussion

The core movement of the snake, the inertial turning, proved to be reliable in each experiment. Every time the snake robot got inside an opening it turned around to

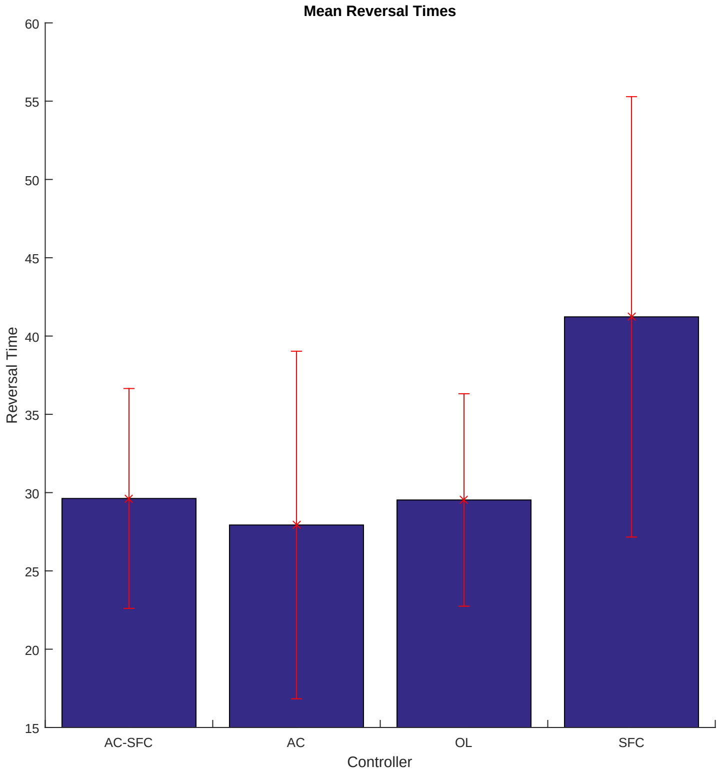

Figure 4.16: Mean of required times for the snake to reverse its motion for each controller. The red bars indicate the standard deviations.

resume sidewinding in the correct direction. Similarly, the heading adjustment was constantly correcting the heading of the snake, making the direction of locomotion robust. Experiments showed how a certain degree of flexibility is beneficial for the snake to adapt to opening between obstacles. Specifically, how shape anti-compliance improves the ability to get past obstacles. Open loop sidewinding, which is a fixed gait unable to interact with the environment, endorses our hypothesis: while succeeding in some cases, it is unreliable and manages to pass through opening of only a fixed length.

The spatial frequency controller is very reliable at the expense of speed; while it fails only a few time, it is the slowest controller. On the contrary, the amplitude compliant controller is the fastest controller, but has the highest amount of failed attempts, hence being very unreliable. The full controller (spatial frequency and amplitude compliant) combines together reliability and speed, never failing to pass through and without sacrificing speed. While the open loop blindly moves against the obstacles, the compliant controller allows the shape to conform to the pegs while propelling forward. Spatial frequency anti-compliance guarantees reliability while amplitude anti-compliance guarantees speed. Overall, the full compliant controller is slightly faster than the open loop while being more reliable.

This work proposes a control structure for a sidewinding snake robot that navigates through environment without any previous knowledge of it. It supports the idea that transitional gaits such as sidewinding should be able to overcome unknown environmental features without being overly complicated to do so. In fact, increasing the simplicity and reducing the quantity of controllers lighten a structure that may require complexity for precise ulterior tasks. The core concept of the devised controller is a bio-inspired technique for the sidewinding snake to pass through obstacle exploiting the environment. Mimicking biological snake behaviour, a rudimentary proprioception allows the snake to keep a defined direction through its path.

Furthermore, the work presented extends the previous achievements on shape-based compliance on the sidewinding gait, exposing the necessity for this particular locomotion gait, and hypothetically for similar self-transporting motions, of anti-compliance when locomoting through a cluttered environment.

In future work will be implement a self-steering control to follow any given curved path, exploiting additional exteroceptive features, such as contact and vision sensors. Further research can be directed towards a mixed compliance/anti-compliance architecture on the same mechanism.

Chapter 5

Conclusions

A mechanism with a high number of degrees of freedom has both advantages and disadvantages: redundancy potentially allows the robot to better comply with the environment and adapt to obstacles, maneuvering through unstructured terrain and confined spaces. Yet, having many degree of freedom makes the system harder to control, and is effective and beneficial only if it is possible to coordinate the motion of the joints in real time to respond to the unknown surroundings.

This work proved shape-based compliance to be a powerful control strategy for a hyper-redundant mechanism such as the *SEA Snake*. The shape-based compliance is an intuitive, compact and light architecture which allows compliance without losing the advantages of a well-defined gait shape. The hyper-redundant mechanisms is described as a geometrical shape that owns certain properties, which are allowed to vary in response to external information. This relationship between external world and internal coordination takes a step toward a biologically inspired approach in which a certain gait (or compound movement for net locomotion in general) is dependent both on the internal kinematics of the robot and the external environment's features. Obstacles are not plainly avoided, but exploited to propel the snake robot forward or used in an active way to support and enhance locomotion. Not only structural properties that define the geometry of the mechanism, but also connecting properties such as centralization are linked to the external world. In fact, supporting the idea that a biological system can change its shape locally or globally depending on the obstacles faced (i.e. the same aspect of a biological movement responds to environmental stimuli in different ways given specific situations), the robot's degree of centralization is made variable and dependent on the external world; each feature owns a specific degree of centralization that varies along the shape in time, depending on its structural level or the macroscopic characteristic it represents.

The first project presented a strategy to modify an open-loop slithering gait to make it effective in unknown and unstructured environments. From the thrashing movements of the open-loop control inside an unstructured set of pegs, the snake became able to successfully exploit the obstacles to propel itself forward without getting stuck. In the second project a previously joystick-controlled climbing gait was enhanced, making it autonomous. The snake robot senses the forces exerted by the object being climbed and is able to adjust its radius to adapt to it without slipping or falling. The third project proposed a novel method to grant proprioception and flexibility to a sidewinding snake robot, enabling an effective locomotion through a set of pegs without any a priori knowledge of it.

These augmentations are made by using a simple framework that does not overly increase complexity. This is of fundamental importance for hyper-redundant systems where locomotion is only part of the control of a mechanism that has to interact with the environment to complete specific and potentially complex tasks. Finally, the shape based compliance control was successfully implemented on different gaits, demonstrating how such a simple structure can be easily applied to different configurations or shapes. Depending solely on the geometry of the mechanism, not only it can be used for distinct motions of the same mechanism, but it can be extended to many robot topologies; a serial topology, typical of snake robots or robotics arms, a network topology of conventional legged robots, a continuous topology of a multiple spatial dimensions robot or even a distributed control of multi-agent robot system such as drone swarms.

Bibliography

[1] D. Rollinson, Y. Bilgen, B. Brown, F. Enner, S. Ford, C. Layton, J. Rembisz et al. “Design and architecture of a series elastic snake robot.” In IROS, pp. 4630-4636. 2014.

[2] M. Hara, S. Satomura, H. Fukushima, T. Kamegawa, H. Igarashi, and F. Matsuno. “Control of a Snake-like Robot Using the Screw Drive Mechanism.” In Proceedings 2007 IEEE International Conference on Robotics and Automation, 28(3):541-554, Apr. 2007.

[3] T. Kamegawa, T. Harada, and A. Gofuku. “Realization of cylinder climbing locomotion with helical form by a snake robot with passive wheels.” In IEEE International Conference on Robotics and Automation (ICRA), pages 3067-3072, Kobe, Japan, 2009.

[4] J. S. Pettinato and H. E. Stephanou. “Manipulability and stability of a tentacle based robot manipulator.” In Proc. IEEE Int. Conf. Robotics Automat. Scottsdale, AZ, May 1989.

[5] M. Ivanescu and I. Badea. “Dynamic control for a tentacle manipulator.” In Pror. Int. Conf. on Robotics and Factories of the Future, Dec. 4-7, 1984, Charlotte, NC, USA, pp. 317-328.

[6] V. V. Anderson and R. C. Hom. “Tensor-arm Manipulator design.” In ASME Trans., vol. 67-DE-57, pp. 1-12, 1967.

[7] T. S. Drozda. “Spine robot ... The verdict’s yet to come.” Manufacturing Engineering, vol. 93, no. 3, pp. 110-112, 1984.

[8] A. Morecki, K. Jaworek, W. Pogorzelski, T. Zielinska, J. Fraczek, G. Malczyk. “Robotic system-Elephant trunk type elastic manipulator combined with a quadruped walking machine.” In Proc. of Oxnard, CA: JPL Publication, 8945, Dec. 15, 1989. Second Int. Conf. on Robotics and Factories of the Future, San Diego, July 1987, pp. 649456.

[9] G. Chirikjian, J. Burdick. “An Obstacle Avoidance Algorithm for Hyper-Redundant Manipulators.” In 1990 IEEE International Conference on Robotics and Automation, Cincinnati, OH, May 13-18, 1990.

[10] G. Chirikjian and J. Burdick. “A modal approach to hyper-redundant manipulator kinematics.” In IEEE Transactions on Robotics and Automation, vol. 10,

pp. 343-354, June 1994.

[11] J. Burdick, J. Radford, and G. Chirikjian, “A Sidewinding Locomotion Gait for Hyper-Redundant Robots.” In Robotics and Automation, vol. 3, pp. 101-106, 1993.

[12] G.S. Chirikjian, J. W. Burdick. “Kinematically Optimal Hyper-Redundant Manipulator Configurations.” In IEEE Transaction on Robotics and Automation, Vol. 11, No. 6, December 1995.

[13] G.S. Chirikjian. “A General Numerical Method for Hyper-Redundant Manipulator Inverse Kinematics".” In Proc. IEEE Int. Conf. on Robotics and Automation, Atlanta, May 1993, pp. 107 - 112.

[14] H. Yamada and S. Hirose. “Approximations to continuous curves of Active Cord Mechanism made of arc-shaped joints or double joints.” In IEEE International Conference on Robotics and Automation (ICRA), pages 703-708. IEEE, 2010.

[15] H. Yamada and S. Hirose. “Steering of pedal wave of a snake-like robot by superposition of curvatures.” In IEEE International Conference on Intelligent Robots and Systems, 2010.

[16] J. Gonzalez-Gomez, H. Zhang, E. Boemo, and J. Zhang. “Locomotion capabilities of a modular robot with eight pitch-yaw-connecting modules.” In 9th international conference on climbing and walking robots. Citeseer, 2006.

[17] A. J. Ijspeert, and A. Crespi. “Online trajectory generation in an amphibious snake robot using a lamprey-like central pattern generator model.” In Robotics and Automation, pp. 262-268. IEEE, 2007.

[18] S.Hirose. “Biologically Inspired Robots.” In Oxford University Press, 1993.

[19] P. Liljeback , K. Pettersen, O. Stavdahl, and J. Gravdahl. “Experimental Investigation of Obstacle-Aided Locomotion With a Snake Robot.” In Robotics, IEEE Transactions on Robotics, vol. 27, no. 4, pp. 792-800, 2011.

[20] A. A. Transeth, S. A. Fjerdingen, and P. Liljeback. “Snake robot obstacle-aided locomotion on inclined and vertical planes: Modeling, control strategies and simulation.” In Mechatronics (ICM), pp. 321-328. IEEE, 2013.

[21] T.Kano and A. Ishiguro. “Obstacles Are Beneficial to Me! Scaffold-based Locomotion of a Snake-like Robot Using Decentralized Control.” In IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS) November 3-7, 2013. Tokyo, Japan.

[22] J. Ostrowski and J. Burdick. “Gait kinematics for a serpentine robot.” In IEEE International Conference on Robotics and Automation, 1996.

[23] M. Tanaka and F. Matsuno. “Control of 3-dimensional snake robots by using redundancy.” In IEEE International Conference on Robotics and Automation, 2008.

[24] F. Matsuno and K. Mogi. “Redundancy controllable system and control of snake robots based on kinematic model.” In IEEE Conference on Decision and Control, 2000.

[25] F. Matsuno and K. Suenaga. “Control of redundant 3d snake robot based on kinematic model.” In IEEE International Conference on Robotics and Automation, 2003.

[26] P. Prautsch and T. Mita. “Control and analysis of the gait of snake robots.” In IEEE International Conference on Control Applications, 1999.

[27] M. Tesch, K. Lipkin, I. Brown, R. Hatton, A. Peck, J. Rembisz, and H. Choset. “Parameterized and scripted gaits for modular snake robots.” In Advanced Robotics, 23:1131-1158, 2009.

[28] D.P. Tsakiris, M. Sfakiotakis, A. Menciassi, G. La Spina, and P. Dario. “Polychaete-like undulatory robotic locomotion.” In IEEE International Conference on Robotics and Automation, 2005.

[29] G. Chirikjian and J. Burdick. “Kinematics of hyper-redundant locomotion with applications to grasping,” In International Conference on Robotics and Automation, 1991.

[30] F. Naccarato and P. C. Hughes. “An inverse kinematics algorithm for a highly redundant variable-geometry-truss manipulator.” In JPL, Proceedings of the 3rd Annual Conference on Aerospace Computational Control, Volume 1; pp. 407-420.

[31] H. Mochiyama, H. Kobayashi. “The shape Jacobian of a manipulator with hyper degrees of freedom.” In Proc. 1999 IEEE Int. Conf on Robotics and Automation, Detroit, May 1999, pp. 2831-2842.

[32] H. Mochiyama, E. Shimeura, H. Kobayashi. “Direct Kinematics of Manipulators with Hyper Degrees of Freedom and Serret-Frenet Formula.” In Proc. 1998 IEEE Inf. Conf on Robotics and Automation, Leuven, Belgium, May 1998, pp. 1653-1658.

[33] M. Shahinpoor, H.Kalhor, M.Jamshidi. “On Magnetically Activated Robotic Tensor Arms.” In Proceedings of the International Symposium on Robot Manipulators: Modeling, Control, and Education, Nov. 12-14, 1986, Albuquerque, New Mexico.

[34] S. Hirose, Y. Umetani. “Kinematic Control of Active Cord Mechanism with Tactile Sensors.” In Proceedings of Second International CISM-IFT Symposium on Theory and Practice of Robots and Manipulators, pp. 241-252, 1976.

[35] A. A. Transeth, K. Y. Pettersen, and P. Liljeback. “A survey on snake robot modeling and locomotion.” In Robotica, vol. 27, p. 999, Mar. 2009.

[36] M. T. Mason. “Compliance and Force Control for Computer Controlled Manipulators.” PhD thesis, Massachusetts Institute of Technology, 1979.

[37] W. Mosauer. “On the Locomotion of Snakes.” In Science, 76(1982):583-585, 1932.

[38] J. Gray. “The Mechanism of Locomotion in Snakes.” In Journal of Experimental Biology, 23(2):101-123, Dec. 1946.

[39] J. Borenstein, M. Hansen, and A. Borrell. “The OmniTread OT-4 Serpentine Robot- Design and Performance.” In Journal of Field Robotics, 24(7):601-621, 2007.

[40] H. Ohno and S. Hirose. “Design of slim slime robot and its gait of locomotion.” In Proceedings 2001 IEEE/RSJ International Conference on Intelligent Robots and Systems., pages 707-715, 2001.

[41] S. Takaoka, H. Yamada, and S. Hirose. “Snake-like active wheel robot ACM-R4.1 with joint torque sensor and limiter.” In IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS), pages 1081-1086, San Francisco, USA, Sept. 2011. Ieee.

[42] G. A. Pratt and M. M. Williamson. “Series elastic actuators.” In Proceedings 1995 IEEE/RSJ International Conference on Intelligent Robots and Systems, pp. 399-406, 1995.

[43] D. Rollinson, S. Ford, B. Brown, and H. Choset. “Design and Modeling of a Series Elastic Element for Snake Robots.” In ASME Dynamic Systems and Control Conference (DSCC), (Palo Alto, USA), 2013.

[44]] C. Wright, A. Buchan, B. Brown, J. Geist, M. Schwerin, D. Rollinson, M. Tesch, and H. Choset. “Design and Architecture of the Unified Modular Snake Robot.” In IEEE International Conference on Robotics and Automation (ICRA), (St. Paul, USA), pp. 4347-4354, 2012.

[45] M. Tesch, K. Lipkin, I. Brown, R. Hatton, A. Peck, J. Rembisz, and H. Choset. “Parameterized and scripted gaits for modular snake robots.” In Advanced Robotics 23, no. 9 (2009): 1131-1158.

[46] C. Gong, D. I. Goldman, and H. Choset. “Simplifying Gait Design via Shape Basis Optimization.” In Proceedings of Robotics: Science and Systems, Ann Arbor, Michigan, 2016.

[47] M. Travers, J. Whitman, P. Schiebel, D. Goldman and H. Choset. “Shape-Based Compliance in Locomotion.” In Proceedings of Robotics: Science and Systems, Ann Arbor, Michigan, 2016.

[48] M. Travers, and H. Choset. “Shape-Constrained Whole-Body Adaptivity.” In International Symposium on Safety, Security, and Rescue Robotics, West Lafayette, IN (2015).

[49] J. Whitman, F. Ruscelli, M. Travers and H. Choset. “Shape-based compliant control with variable coordination centralization on a snake robot.” In 55th IEEE Conference on Decision and Control, December 12-14, 2016, Las Vegas, NV, USA.

[50] D. Rollinson and H. Choset. “Gait-based compliant control for snake robots.” In Robotics and Automation (ICRA), pp. 5138-5143. IEEE, 2013.

[51] Weikun Zhen, Chaohui Gong and Howie Choset. “Modeling Rolling Gaits of A Snake Robot.” In 2015 IEEE International Conference on Robotics and Automation (ICRA).

[52] Chaohui Gong, Ross L. Hatton and Howie Choset. “Conical Sidewinding.” In 2012 IEEE International Conference on Robotics and Automation RiverCentre, Saint Paul, Minnesota, USA May 14-18, 2012.

[53] D. Rollinson and H. Choset. “Virtual chassis for snake robots” In IEEE International Conference on Intelligent Robots and Systems, pp. 221-226, 2011.