INSTITUTO SUPERIOR TÉCNICO
Universidade Técnica de Lisboa

Using sensory feedback to improve locomotion performance of the salamander robot in different environments

João Pedro Lourenço Silvério

Master's Thesis in
Electrical and Computer Engineering

Professor:	Prof. Auke Jan Ijspeert
Supervisor:	Jérémie Knüsel
Co-supervisors*:	Prof. Alexandre Bernardino
	Prof. José Santos-Victor

October 2011

*Co-supervised from Instituto Superior Técnico (IST), Portugal

Acknowledgements

I would first like to express my deep gratitude to Professor Auke Ijspeert for having given me the opportunity of doing such an interesting project at Biorob. Also, many thanks to Jérémie Knüsel for supervising my project, for enlightening me with his precious insights and mostly for his patience. Without him this project would not have been possible.
I would also like to thank Professor Alexandre Bernardino, from IST, for his availability while accepting me as a Master student of his, and for his prompt receptiveness when I needed his help concerning the project.
To my friends from IST and from my year as an exchange student in EPFL, thank you for your friendship and support and for making these past five years the happiest and most productive years of my life. To my friends who have been with me all my life, thank you for your friendship and for having helped shaping me to be the person who I am today.
To finalize, I would like to take this opportunity to manifest how thankful I am to my special one, Ysamar Rodriguez, for her unconditional love and belief in me, to my family, my mother and father for their constant guidance, love and support through the last 24 years, and to my brother and sister for making my life better by being by my side in every moment. This project, as well as all the rest of my academic achievements are dedicated to these people.

Abstract

Salamanders, as the living tetrapods whose locomotor circuits most resemble the ones of the first terrestrial vertebrates, present two distinct types of motion: walking and swimming. In an attempt to explain gait transitions in vertebrates, several models have been used at EPFL's Biorobotics Laboratory to demonstrate how the transitions between gaits are performed by salamanders, eventually leading to the creation of the salamander robot, an amphibian robot that can robustly swim and walk. With the creation and further development of the salamander robot, several questions regarding its locomotion control came up, one of which is the role of sensory feedback in the robot's locomotion and the possibility of using it to improve performance.

It is now quite accepted that even though there is no need for sensory feedback to generate rhythmic activity in central pattern generators (CPGs), it is used to adjust locomotor responses depending on the environment. Based on the positive results of sensory feedback integration in the control loop of other quadruped robots, such as Tekken2, Aibo, iCub, or Ghostdog, the main purpose of this project is to implement a similar type of control in the salamander robot in order to make it better adapted to changing environments.

Results have shown that using sensory feedback from touch sensors on the robot's limbs is beneficial for the walking gait, especially in terrains with different frictions or inclinations.

Keywords: Biologically inspired robotics, salamander robot, central pattern generator (CPG), locomotion control, sensory feedback, Hopf oscillator.

Contents

List of Figures

List of Abbreviations

CPG	Central Pattern Generator
MLR	Mesencephalic Locomotor Region
LF	Left Forelimb
RF	Right Forelimb
LH	Left Hindlimb
RH	Right Hindlimb

Chapter 1

Introduction and scope

1.1 Overview

Animal locomotion has been extensively studied over the years by researchers from both robotics and biology fields. In fact, the interactions between these two fields regarding this topic have proven to be very profitable for both. Mobile robots, especially legged robots, are increasingly using models of biological locomotor systems that are successfully applied in nature, while biology uses robots as an embodiment for validation of new locomotion models. Studies of vertebrate locomotion in water, namely on the neural networks responsible for the locomotion of the lamprey, have revealed that the coordinated, rhythmic motion of the body of vertebrates is not directly controlled by the brain but rather by specific neural networks that are situaded along the spine. These networks, known as central pattern generators (CPGs), are formed by inter-connected oscillatory centers that generate bursts of rhythmic activity that control muscles and whose coupling is essential for coordination of movements. The term *central* comes from the fact that sensory feedback from the *peripheral* nervous system is not necessary for the generation of rhythmic activity. CPGs present several properties, like distributed control and robustness against perturbations, which make mathematical models of CPGs very suitable for robot locomotion control.

Salamanders, as tetrapods which most resemble the first terrestrial vertebrates, show two very distinct types of locomotion - walking and swimming - and the fact that they are much simpler than mammals, regarding their neural system, makes them much easier to be studied. In an attempt to understand gait transitions in vertebrates, researchers at EPFL's Biorobotics Laboratory (Biorob), developed several models to explain how the transitions between gaits

are performed by the salamander at a neurobiological level, eventually leading to the creation of the salamander robot, an amphibian robot which, besides the technological breakthrough it represents, has been used as test-bed for hypothesis regarding biological concepts behind locomotion.

Even though the generation of rhythmic patterns of activity does not depend on sensory feedback, several experiments have shown that sensory feedback is rather important in shaping the patterns of activity within a CPG - sensory feedback closes the control loop by providing the CPG with information from the environment. The creation and further development of the salamander robot at Biorob, allowed studying the role of sensory feedback and the possibility of using it to improve the performance of locomotion.

From the control point of view, designing a controller that is capable of performing adaptive walking should be a priority when developping a legged robot, as minor changes in the environment must not affect too much the locomotion performance. The inclusion of feedback from sensors in the control loop of other quadruped robots has shown interesting results in the recent past as the adaptability of these robots to changing environments increased (Righetti and Ijspeert, [3], Kimura et. al, [4]), with these improvements occurring especially in speed. Sensory feedback from the limbs allows the correct identification of swing and stance phases of locomotion (the phases when limbs are off and on the ground, respectively) and it has been shown that this capacity allows keeping consistent speed patterns even when the environment changes (e.g. inclination, friction).

This project aims at the study of the effect of sensory feedback in the salamander robot's walking gait. It addresses the need for improving its locomotor skills, since one of its possible future applications is to be used for inspection or exploration purposes in difficult environments, as well as questions from neurobiology such as the way how oscillators are coupled. Despite having a quite vast background, involving concepts from both biology and robotics, this project took considerable inspiration from the work of Righetti and Ijspeert, [3], mainly regarding the design of the feedback controller.

1.2 Project goals

As aforementioned, the idea behind this project is the work done by Righetti and Ijspeert (2008, [3]) in which a controller was developed that uses sensory feedback to modulate the transitions between swing and stance phases of locomotion in a four-legged robot. This

controller will be now adapted to the salamander robot which, besides the limbs, has other eight degrees of freedom - 8 active joints that control the bending of the body.
The purpose of this project is threefold:

- to design a controller for the salamander robot, that produces stable walking locomotion, using sensory feedback from the limbs to correctly identify the locomotion phases. This has never been done up to now in this robot.
- to compare the feedback controller with a traditional open-loop controller (that uses no sensory information), which shall be optimized for speed. This performance assessment will be done in different environments and the performance indicators are speed and gait stability.
- to provide some feedback to biology concerning the functioning of the neural system of the real salamander.

For the present case, stable walking is defined as the ability of the robot to walk along a straight line, when the controller instructs it to do so. This is measured by the tortuosity of the generated path, which is the ratio between the travelled distance and the distance between start and end points of the trajectory (see chapter 6).
The project relies on Webots 6.3.4, a simulation platform for mobile robots on which a model of the salamander robot is used that allows the optimization of the open-loop controller and the performance assessment. Testing the controller on the real robot was initially planned but the controller design and simulation setup exceeded the time they had been initially assigned leaving no time left for this task.
This dissertation's background has a very strong biological component which, at some point, may suggest that its purpose deviates too much from robotics. This assumption, however, shall be avoided as it should remain clear that the main problem that is being addressed is the study of a control strategy that allows a mobile robot to be better adapted to a diversified range of environments. Thus, although this control strategy involves sensory feedback, which stems from biology, the problem remains within the field of robotics.

Original contributions

1. Design of a feedback controller for the salamander robot's locomotion based on one previously designed for other quadruped robots.

2. Speed optimization of the traditional open-loop controller of the salamander robot in a standard flat terrain.

3. Analysis and comparison of the performance of open- and closed-loop controllers regarding speed and gait stability in different environments with several degrees of dificulty.

4. Results that show that sensory feedback from touch sensors in the limbs allow the closed-loop controller to perform better than the open-loop controller especially in terrains with distinct friction conditions, inclinations and in some cases, with rough texture.

1.3 State of the art

The salamander robot was the result of studies on both the lamprey's and salamander's spinal cord. In [5], Grillner works out the mechanisms behind the generation of propulsive waves of the lamprey's body and validates the resulting models through some of the first computer simulations regarding lamprey's locomotion, in which several patterns of activation are tested. These neural networks are described to the level of interneuron excitation and inhibition in a work by Grillner et. al, [6]. Kozlov et. al, in [7], propose a numerical model for the postural control system of the lamprey that allows it to maintain its dorsal side up when swimming. It shows that sensory input from the vestibular system triggers the generation of a torque in the direction opposed to the rolling of the body, providing postural correction. It is likely that this model could be applied to maintain the postural control of the salamander robot's swimming gait, if a proper way for simulating vestibular input could be devised (e.g. gyroscopes).
The vast work done in understanding the lamprey's locomotor circuits is somehow extendible to the salamander, as salamanders' spinal cord shares many similarities with the one of the lamprey. Chevallier et. al ([8]), reviews the neurobiological data about the organisation of the spinal networks responsible for swimming and stepping of salamanders, and introduces the salamander robot as a test-bed for locomotion models. The design and control of the robot is described in [9]. Ijspeert et. al, in [10], show that sensory feedback may have played an important role in the evolutionary transition from swimming to walking, as the inclusion of sensory signals from lateral stretch sensors resulted in the generation of S-shaped waves in a CPG designed for generating travelling waves. Nevertheless, this hypothesis suggests no improvement for the locomotion of the robot, reason why it was not covered by the present work.

In [4], sensory feedback is used to stabilize the walking motion of the mammal-like quadruped robot Tekken2, for which a few necessary conditions for stability are enunciated, which concern inter-limb coordination and body posture. Gyroscopes and inclinometers are used to measure the rolling and inclination of the body (simulate vestibular system), while a passive ankle joint allows the detection of swing and stance phases and stumbling on obstacles. Signals from these sensors are used as input to the CPG network that generates torques which act on the leg joints to provide the desired stability. This approach has proven to be successful for Tekken2 but is rather inadequate for the salamander robot: stability conditions differ from one robot to the other (e.g. the salamander robot's body is much closer to the ground due to its shorter limbs), the limbs are quite distinct (the ones of the salamander robot have a single degree of freedom against four in Tekken2) and the onboard technology currently available is rather limited for the salamander as it posesses a single accelerometer. Nevertheless, the concept of modulating CPG phase through the inclusion of sensory feedback from the limbs is shown to improve locomotion performance, which strongly supports the approach proposed in the present work for the salamander robot.

Righetti and Ijspeert proposed several improvements for the locomotion control of quadruped robots. In [11] a new CPG architecture that relies on Hopf oscillators is proposed that allows to independently control the duration of the ascending and descending phases of the oscillators (that correspond to swing and stance phases of the limbs). This approach was complemented in [3] in which sensory feedback was introduced to accerelate the transitions between the two phases when limbs touch and lift off the ground. The ability to control the duration of the locomotion phases together with the possibility to perform transitions allows a correct identification of swing and stance phases. This has been proven to increase performance concerning robot speed and controller robustness to incorrect control parameters choice. One of the most relevant disadvantages of this work is the fact that the limbs of the quadruped robots for which it was designed move back and forth, instead of rotating around an axis in a single direction as occurs in the salamander robot. This places clear limitations on applying the transition mechanism that is proposed in [3] because in the salamander robot the transitions occur in different positions of the oscillator's phase. Also, the proposed CPG network only involves quadruped locomotion thus not addressing control strategies that involve other degrees of freedom besides the ones of the limbs (as the salamander robot has other degrees of freedom for the body). Regardless of these limitations, the approach that

was followed in this dissertation was the tayloring of the proposed CPG network to adapt it to the salamander robot's mechanical and control specifications.

1.4 Report structure

This report is organized according to the following division of chapters:

- **Chapter 1: Introduction and scope** - The first chapter introduces the project, clarifies its goals and describes previous works.
- **Chapter 2: Background** - Biological background is covered to address the need for a better clarification of the project's scope. Neurobiological foundations of CPGs are discussed as well as their applications in robotics over time. The role of sensory feedback is also addressed by revealing biological evidence of its role in locomotion and a few concepts regarding stepping gaits are defined.
- **Chapter 3: Simulation environment** - Comprises the description of the robot model used in Webots, regarding its simulated physical properties and general parameters as integration, control and physics engine time steps.
- **Chapter 4: CPG network architecture and oscillator model** - The CPG network and oscillator model that were used are described in detail as well as the changes that were made to the original architecture in order to be valid for application in the salamander robot.
- **Chapter 5: Optimization of the open-loop controller** - Setup and results of the open-loop controller optimization.
- **Chapter 6: Controller performance** - In this chapter the simulation setup in different environments is described regarding their physical properties. The results of testing both controllers are depicted and discussed.
- **Chapter 7: Conclusions and future work** - Finally, the results are summarized and the advantages and disadvantages of the chosen approach are discussed. Recommendations on future work are also made.

Chapter 2

Background

2.1 Animal locomotion and robotics: an introduction

The ability to move is perhaps one of the most impressive skills that animals possess. Tailored by natural selection, millions of species have been shaped over time in order to adapt to different environments. Limbs became increasingly more complex and gaits more diversified: walking, swimming, jumping, flying or crawling gaits, for instance, are the consequence of the powerful process of evolution.

Animal gaits present dozens of degrees of freedom that are controlled with a surprising ability, and are interesting for their complexity, flexibility and energy efficiency that reach performance levels that have not yet been attained by robots. Designing a mobile robot, can be seen as an optimization problem since factors like cost, reliability and maneuverability have to be accounted for, so the development of more complex and morphologically diversified robots was accompanied by the need for appropriate locomotion strategies, leading many roboticists to turn to nature in search for control algorithms that could be applied in robot locomotion. One of the now most widespread methods for controlling locomotion in robots is the use of central pattern generator (CPG) models. CPGs are neural circuits found in animals that generate rhythmic patterns of neural activity without the need for rhythmic sensory input (section 2.5 discusses the role of sensory feedback). Spine CPGs are essential to vertebrate locomotion as they are responsible for generating the basic rhythmic patterns that control coordinated muscle activity, reaching high degrees of complexity of muscular synchronization. In fact, this kind of synchronization is absolutely essential to any animal gait and it is one in many other synchronization phenomena that are found in nature that have been mathe-

matically modeled ([12]). The majority of the CPG models implemented in robots consists of sets of coupled differential equations that are numerically integrated on microcontrollers and whose variables are used as control policies for robot joints.
The most modeled classes of animals are insects and lower vertebrates, being the lamprey the most expressive example of the latter ([5], [6]). CPG models have been used with high levels of success in hexapod and octopod robots, swimming and amphibian robots (lamprey and salamander robots) as well as quadruped and biped robots. Ijspeert, [13], reviews CPG models and their applications in robotics, topics that are adressed in 2.2.
A less common type of robot, a jumping robot, was developed at EPFL's Laboratory of Intelligent Systems (LIS) (Kovač et al., [14]). Unlike the previous examples, this was not the implementation of a CPG model but rather an example of how far the inspiration by biological concepts can go as many properties of the robot were modeled after jumping animals such as frogs, locusts and fleas.

While some researchers use biology to develop better robots, others use robots to better understand biology. CPG models are abstracted from biological activity that occurs at neuron level, so at some point all these models might need validation. Compared to computer simulations, real robots provide a series of advantages when used as tools for testing CPG models ([9]):

- Robots allow the model to interact with real environments using real sensors and actuators, thus eliminating biased results as consequence of sensor and actuator simulation models.
- In real environments the robot is acted by real forces. Simulation of complex models of forces (e.g. hydrodynamics, friction) and environments is difficult and very prone to errors which may as well influence results.

Despite these facts, computer simulations must not be disregarded as a way to successfully test locomotion models as they offer other types of advantages (section 3).

The present work stems from studies of the locomotion of an amphibian animal, the salamander, and its bio-inspired robot, the salamander robot (formally *Salamandra Robotica*, [9]). Salamanders are regarded as the tetrapods that most resemble the first terrestrial vertebrates, which came out of water [2]. As amphibians they present two ways of locomotion -

walking and swimming - that make them interesting from the point of view of understanding how transitions between gaits occur at a neurobiological level. Additionally, salamanders' neural system presents a number of neurons that is orders of magnitude smaller than that of mammals, so it is seen as easier to be studied. The purpose behind the creation of the salamander robot is twofold: the development of an amphibian robot capable of successfully move in distinct environments and the study and implementation of CPG models focused on gait transitions.

2.2 CPGs for robot locomotion

2.2.1 Neurobiological foundations

CPGs are neural networks that generate coordinated patterns of rhythmic activity without the need for rhythmic inputs neither sensory nor from higher control centers ([13]). Experiments in lampreys for instance, have revealed the existence of such neural circuits as the isolation of the lamprey's spinal cord followed by electrical/chemical stimulation produced clear patterns of activity (fictive locomotion).

Higher control centers in the brain, although not used as rythmic inputs, are responsible for the modulation of the moving patterns. The electrical stimulation of the Mesencephalic Locomotor Region (MLR) is known to induce locomotor behaviour, where low-level stimulation leads to slow movements and high-level stimulation to fast movements and gait transitions. This simple form of control releases bandwidth from the brain since there is no need for direct control pathways between brain and muscles.

2.2.2 CPG Models

Models of CPGs are quite diversified and are mostly distinguished by their level of abstraction. Some models, like biophysical or connectionist models, focus on the generation of rhythmic activity at a neuron level and use models of neurons (e.g. Hodgkin-Huxley, leaky-integrator) while other models assume the existence of these oscillatory centers and focus on the coupling between them rather than the way how rythms are generated. These are known as systems of coupled oscillators.

The dynamics of populations of oscillatory centers do not show considerable dependence on the models of the oscillators themselves, but rather on the type and topology of inter-

oscillator couplings. This means that equivalent results can be obtained among different types of oscillators. Popular types of oscillators are van der Pol, Stein, FitzHugh-Nagumo, phase oscillators ([2]) and Hopf oscillators. Hopf oscillators are known for being particularly suitable for phase space modulation allowing the adjustment of swing and stance phases ([11]) as well as for the inclusion of sensory feedback ([3]). This type of oscillator was applied for the control of the salamander robot in this project, and is discussed with higher detail further along the report.

Concerning robotics, there is not much to gain with the application of too complex models for locomotion control, reason by which the most used CPG models are models of coupled oscillators. These models are mostly governed by sets of differential equations, which describe the time evolution of the oscillators' variables, and that are numerically integrated by onboard microprocessors to generate the control policies for the robot's joints.

The oscillators in CPG models exhibit limit cycle behaviour, i.e., periodic behaviour of state variables in the time domain, together with resistance to trasient perturbations, after which the system rapidly returns to its normal rhythmic behaviour (figure 2.4). Also, most CPG models allow using fairly simple control signals as, similarly to biological CPGs, the speed of movements can be regulated by a simple variable that is an abstraction of the amount of simulation, reducing the dimensionality of the control problem (controllers do not need to generate motor control commands, for example). Under the same line of thought, abrupt changes in the control parameters do not affect the CPG output as the differential equations act as low-pass filters and the output signals are rather smooth (avoiding damage in motors and gearboxes). As aforementioned, CPGs also allow the inclusion of sensory feedback, without increasing too much the complexity of the system, as well as automatic transitions between swing and stance phases in walking gaits.

2.2.3 Hopf oscillators

Single-oscillator dynamics

Dynamical systems theory defines an oscillator as a dynamical system that exhibits a structurally stable limit cycle. A limit cycle is a closed curve in the phase space to which the oscillator's variables converge and it is said to be structurally stable when perturbations or small changes in the parameters do not destroy the qualitative behaviour of the system.

The Hopf oscillator is an example of a dynamical system that possesses such properties as its

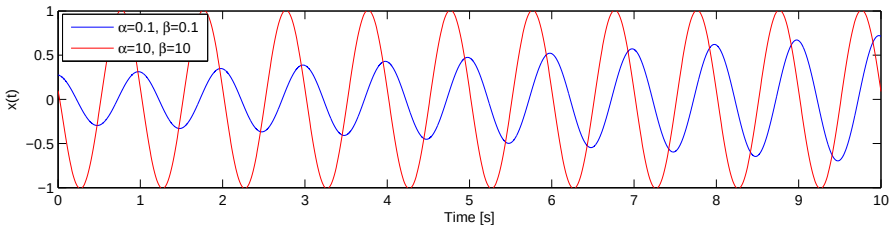


Figure 2.1: Time plot of the x variable of a Hopf oscillator for different values of α and β with randomly generated initial conditions and oscillator parameters $\omega = 2\pi$ and $\mu = 1$. High values of α and β result in a faster convergence to the limit cycle.

solutions converge to a closed curve centered in the origin of the phase space with constant radius. Its x and y variables exhibit, thus, a sine behaviour in steady state (Figure 2.2b).
The standard Hopf oscillator is governed by the following set of differential equations ([3]):

$$\dot{x} = \alpha(\mu - r^2)x - \omega y \tag{2.1}$$

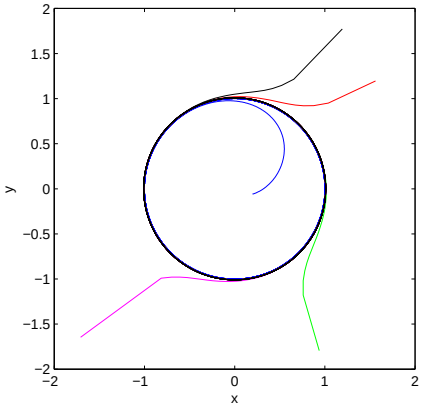
$$\dot{y} = \beta(\mu - r^2)y + \omega x \tag{2.2}$$

where $r = \sqrt{x^2 + y^2}$, ω is the frequency of the oscillations in $rad.s^{-1}$, x and y are the oscillator's variables, $\sqrt{\mu}$ is the amplitude of oscillations in limit cycle and α and β are parameters that control the speed of convergence of the system (Figure 2.1). The limit cycle of this system consists of stable periodic solutions of x and y of the form ([15]):

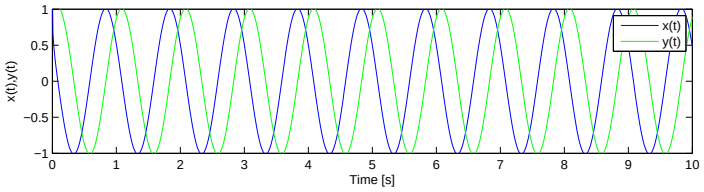
$$x(t) = \sqrt{\mu} sin(\omega t + \theta_0) \tag{2.3}$$

$$y(t) = \sqrt{\mu} cos(\omega t + \theta_0)$$

where θ_0 is determined by the initial conditions.
Figure 2.2a depicts the phase space of a Hopf oscillator (after numerical integration of equations 2.1 and 2.2) which shows the convergence of the system to the limit cycle of amplitude 1 for several different initial conditions. Figure 2.2b shows the time evolution of x and y for the case when the initial conditions are $x_0 = 1.03$ and $y_0 = 1.76$. Both figures are quite representative of the limit cycle property of the Hopf oscillator as they show how the system's variables converge to a stable periodic solution regardless of the initial conditions.



(a) Limit cycle in the phase space.



(b) Time evolution of the oscillator's variables.

Figure 2.2: Time evolution of the system both in phase space and time. There is a clear convergence to the harmonic limit cycle of amplitude 1. ($\omega = 2\pi$, $\mu = 1$, $\alpha = 10$, $\beta = 10$.)

In the interest of providing an analytical demonstration of the limit cycle for this oscillator, equations 2.1 and 2.2 may be re-written in polar coordinates ([16]):

$$\dot{r} = (\mu - r^2)r \tag{2.4}$$

$$\dot{\phi} = \omega \tag{2.5}$$

where α and β were made $\alpha = \beta = 1$, for simplicity. These two equations govern the behaviour of the system along the radial and angular directions, e_r and e_ϕ, respectively (figure 2.3).

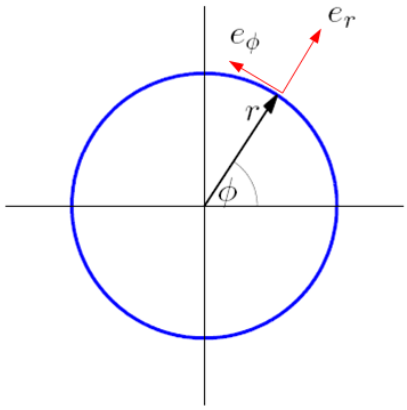


Figure 2.3: Representation of the polar coordinates r and ϕ in the phase space of the oscillator [1].

From the linear differential equation $\dot{\phi} = \omega$ results that the phase of the oscillator increases over time, at a constant rate defined by the value of ω. Consequently, the oscillator describes a constant-speed trajectory in the phase space, along the direction of e_ϕ. On the other hand, equation 2.4 has a fixed point at $r^* = \sqrt{\mu}$. A fixed point r^* is such that it verifies $\dot{r} = f(r^*) = 0$ and it can be either stable (behaving as an atraction point, towards which the system converges) or unstable (with the system diverging from it). r^* is a stable fixed point if ([17]):

$$\left.\frac{df}{dr}\right|_{r^*} < 0 \tag{2.6}$$

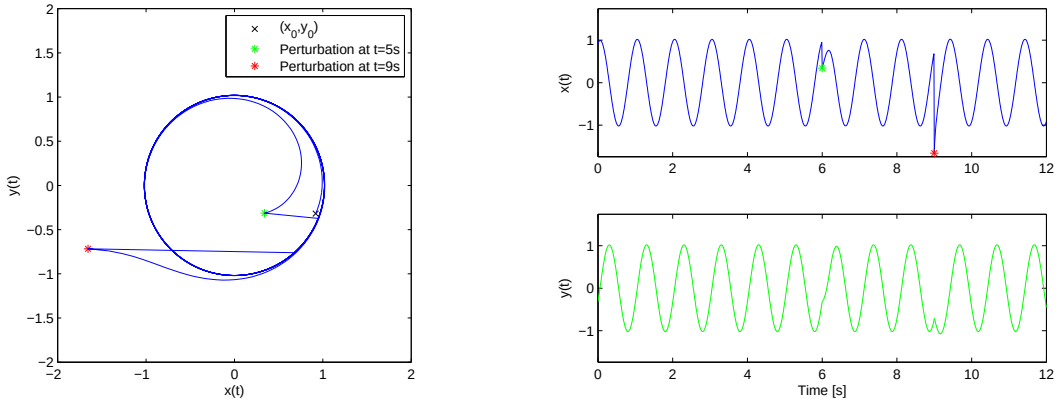
Applying 2.6 in 2.4:

$$\left.\frac{df}{dr}\right|_{r^*} = \mu - 3r^{*2} \tag{2.7}$$

$$= -2\mu < 0 \tag{2.8}$$

thus, $r^* = \sqrt{\mu}$ is a stable fixed point of the system, and it represents the radius of the trajectory in the x-y plane to which the oscillator converges. $r^* = 0$ is also a fixed point of $\dot{r}$ but unstable, since $\left.\frac{df}{dr}\right|_{r^*} = \mu > 0$ ([17]).

In figure 2.4 two perturbations were applied at $t = 6s$ and $t = 9s$ by randomly changing the value of x. The behaviour of the system shows its insensitivity to perturbations as x and y recovered their initial steady-state behaviour in a short period of time.



(a) Limit cycle in the phase space with added perturbations.

(b) Time evolution of the oscillator's variables.

Figure 2.4: Phase space potrait and time evolution of oscillator's variables. Perturbations were applied at t=6s and t=9s with the system recovering its harmonic behaviour. ($\omega = 2\pi$, $\mu = 1$, $\alpha = 5$, $\beta = 5$)

Inter-oscillator coupling

The concept behind the utilization of CPGs is having a network of interconnected oscillators that are synchronized. Coupling between Hopf oscillators can be obtained by modifying equations 2.1 and 2.2 to include an additional term (Amplitude Independent Diffusive Coupling - [18]):

$$\dot{x}_i = \alpha(\mu - {r_i}^2)x_i - \overline{\omega}y_i \tag{2.9}$$

$$\dot{y}_i = \beta(\mu - {r_i}^2)y_i + \overline{\omega}x_i \tag{2.10}$$

$$\overline{\omega} = \omega + \sum_{j=1}^{N} \frac{w_{ij}}{r_i}[(x_i y_j - x_j y_i)cos\Phi_{ij} - (x_i x_j + y_i y_j)sin\Phi_{ij}] \tag{2.11}$$

where w_{ij} and Φ_{ij} are, respectively, the strength of the coupling and the desired phase difference between oscillators i and j. The remaining variables are the same as in 2.1 and 2.2 except that for this case they are indexed by i and j.

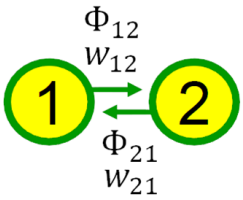
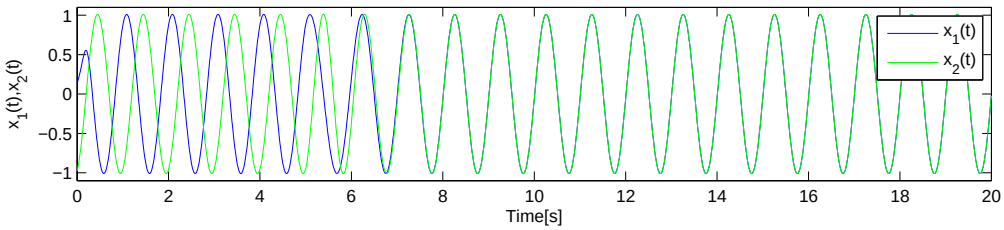
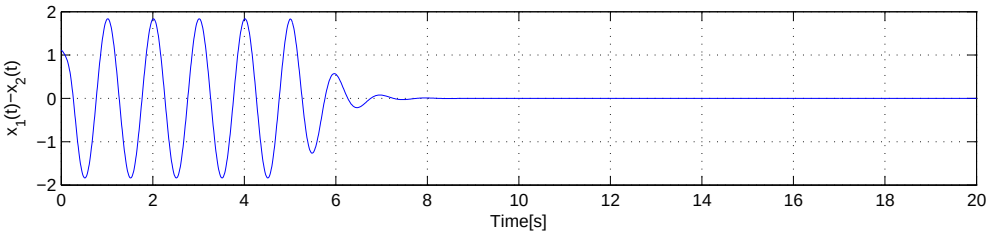


Figure 2.5: Schematic representation of a small network of two coupled oscillators and corresponding variables.

An example of this type of coupling is shown in Figure 2.6. For this case, a network of two oscillators (figure 2.5) was tested with a desired phase difference of $\Phi = 0$ (oscillators synchronized in phase). Both oscillators are initialized with random initial conditions and, at $t = 5s$, they are set to synchronize. Figure 2.6a shows how after $t = 5s$ the x variables of the oscillators, x_1 and x_2, synchronize as their phase difference goes to 0 (Figure 2.6b).
The resistance to perturbations of individual oscillators is also present in networks of coupled oscillators. In figures 2.7a and 2.7b, a perturbation is applied to the system after reaching synchronism, with the oscillators rapidly recovering the desired phase difference of, in this case, $\Phi = 0$. Also, this example serves to show the influence of the coupling weight on the speed as with a higher weight, the synchronism is reached in less time.
Another way of coupling Hopf oscillators is proposed by Righetti and Ijspeert in [3] and it also consists in modifying equations 2.2:



(a) Time plot of $x_1(t)$,$x_2(t)$



(b) Time plot of $x_1(t) - x_2(t)$

Figure 2.6: Time evolution of x_1 and x_2. After $t = 5s$ the oscillators synchronize to a phase difference of $\Phi = 0$. The coupling weights are $w_{12} = w_{21} = 1$.

$$\dot{x}_i = \alpha(\mu - {r_i}^2)x_i - \omega y_i \tag{2.12}$$

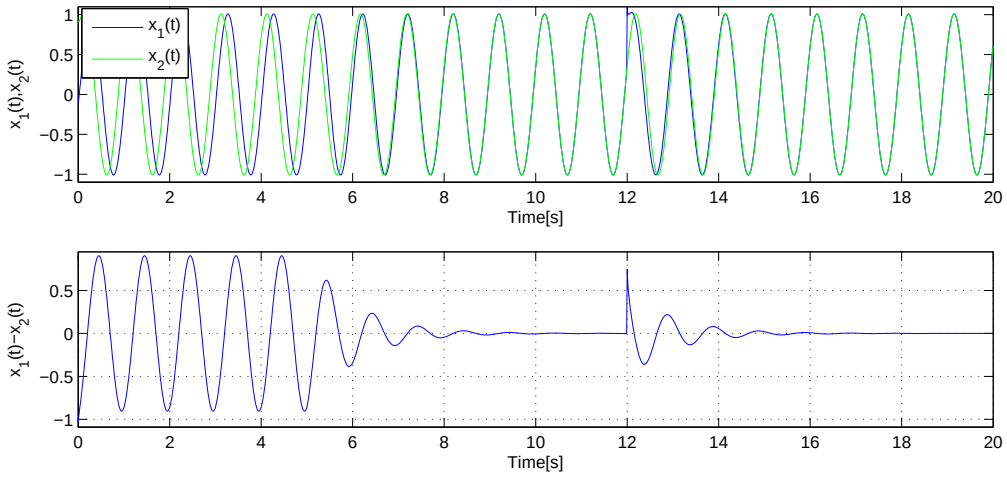
$$\dot{y}_i = \beta(\mu - {r_i}^2)y_i + \omega x_i + \sum_{j=1}^{N} k_{ij} y_j \tag{2.13}$$

$$\tag{2.14}$$

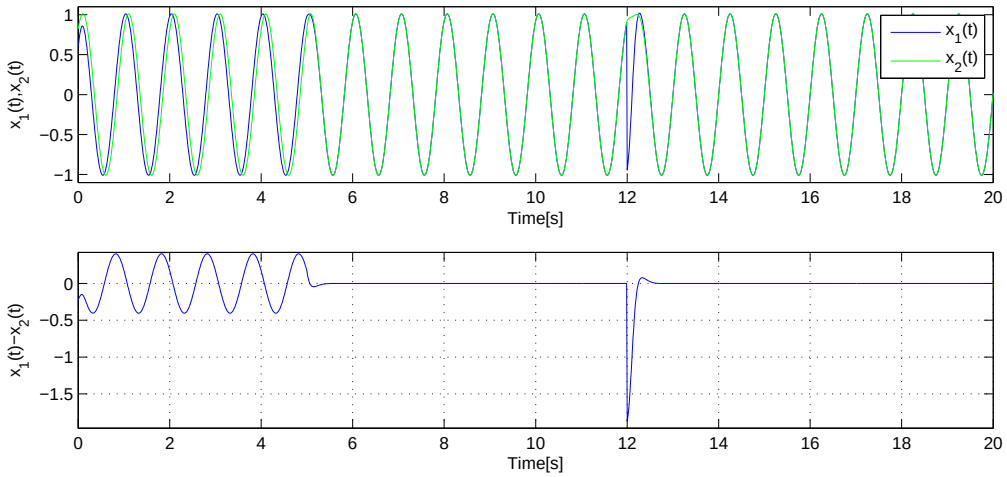
with N denoting the number of oscillators coupled to i and k_{ij} the coupling weight that are defined in a coupling matrix, K. This approach is tailored to obtain specific types of inter-oscillator synchronism that correspond to quadrupedal gait patterns (Figure 2.8). As a consequence, it does not allow choosing the phase difference as freely as in equations 2.11. Nevertheless, both approaches were used in the present work. More information on this subject is found in chapter 4.

2.3 The salamander robot

Salamanders are known as the living vertebrates that most closely resemble the first tetrapods that walked on Earth ([2]) therefore they are key animals in the study of the main evolutionary changes in the spinal locomotor circuits that occured during the transition from swimming to walking. These amphibians also have orders of magnitude fewer neurons than mammals,

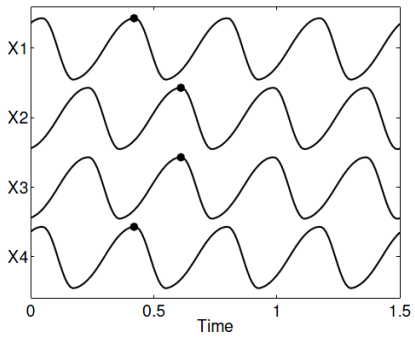


(a) $w_{12} = w_{21} = 0.5$



(b) $w_{12} = w_{21} = 5.0$

Figure 2.7: After $t = 5s$ the oscillators synchronize to a phase difference of $\Phi = 0$. At $t = 12s$ a perturbation is applied to x_1. Higher coupling weight, leads to faster synchronism



$$\begin{bmatrix} 0 & -1 & -1 & 1 \\ -1 & 0 & 1 & -1 \\ -1 & 1 & 0 & -1 \\ 1 & -1 & -1 & 0 \end{bmatrix}$$

Trot

Figure 2.8: Example of a coupling matrix for a trot gait and the generated x_i variables for a 4-oscillator network.

which make them quite more tractable for comprehension and modelling. Moreover, since their central nervous system shares many similarities with the one of the lamprey, the high number of existing studies on lamprey's locomotor system can be used to guide the understanding of salamader's locomotor circuitry as well ([8]).

Regarding the swimming gait of salamanders, it is identical to the one of lampreys: it is an anguiliform type of swimming, that is based on axial undulations that propagate from head to tail (**travelling waves**) with an approximately constant wavelength (figure 2.9). The limbs are folded backwards along the side of the body, and the wavelength is generally equal to the body length (the body makes a complete wave), and does not vary with the frequency of oscillations. The speed of locomotion is controlled by amplitude and frequency of the waves.

With respect to the walking gait, salamanders exhibit a quite common gait in reptiles, regarding its kinematics: the body makes an **S-shaped standing wave** with nodes at the girdles and diagonally oposed limbs are moved in phase (figure 2.10). The limbs are coordinated with the body in such a way that when the body is contracting to one side, the front limb on that side lifts from the ground and swings in the air to reach ground again. This increases the stride length, which is distance travelled by the body in one step cycle, and thus the general speed of locomotion ([8]).

As an amphibian, the salamander is an important subject in the studying of how transitions between gaits occur at a neurobiological level. Regarding the salamander's CPG, it has been shown that the spinal CPG, that is responsible for generating swimming gaits, is like that of the lamprey, formed by two chains of oscillatory centers, located on both sides of the

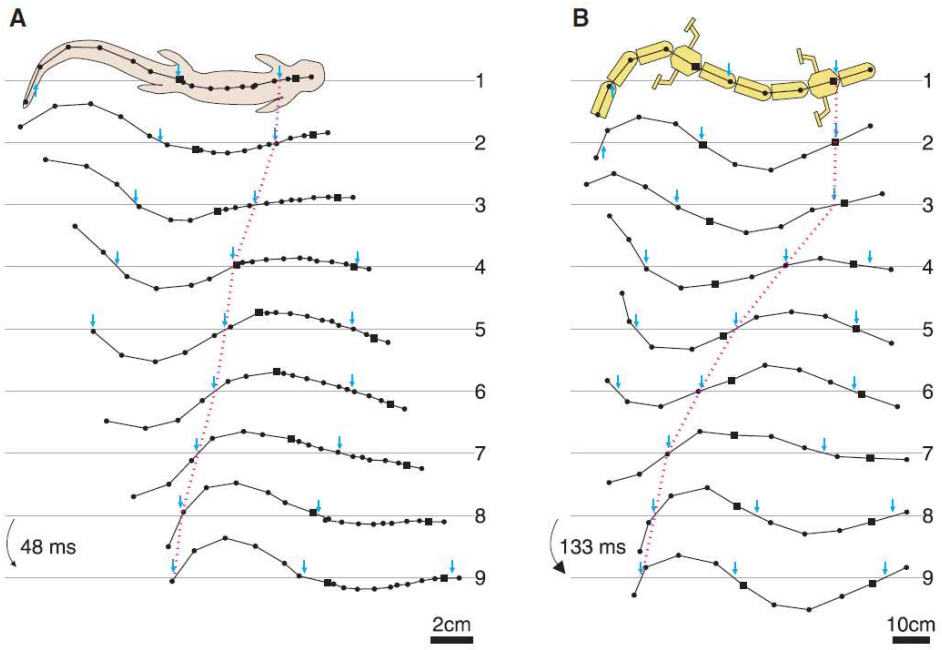


Figure 2.9: Sketch of swimming kinematics in the real and robotic salamander, on the left and right, respectively. Note the travelling wave in the body undulation, with the body making a complete wave (wavelength of one body length)([2]).

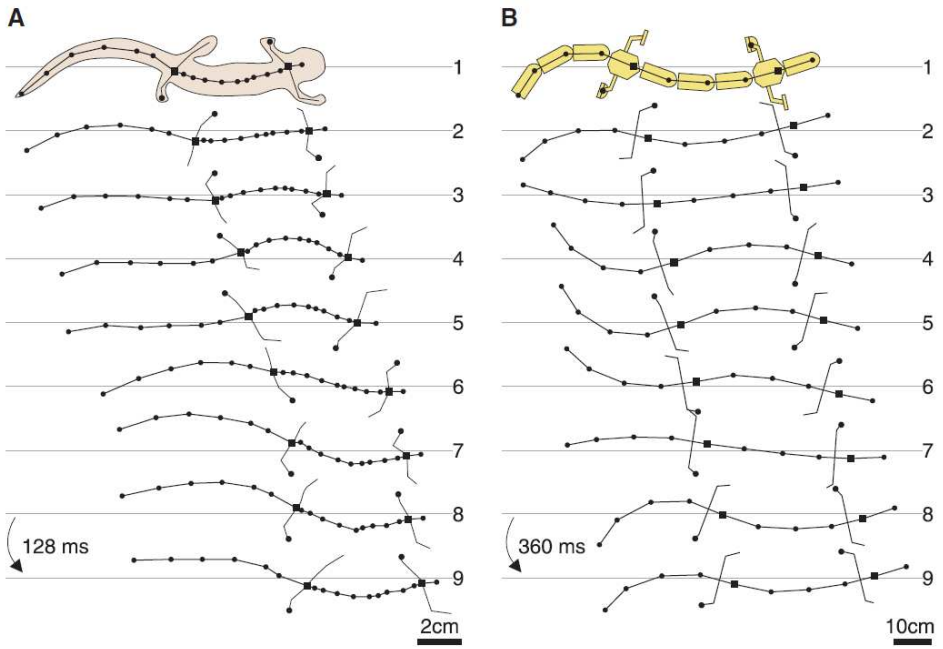


Figure 2.10: Sketch of walking kinematics of the real salamander (A) and the salamander robot (B). A dot at the extremity of a limb indicates ground contact, squares indicate girdles.([2])

spinal cord. The limb CPG consists of neural centers located in the cervical segments, for the forelimbs, and the thoracolumbar segments, for the hind limbs ([2]).

Another biological finding was that gaits are modulated by the mesencephalic locomotor region (MLR) located in the brain. As previously mentioned, the electrical stimulation of this region, induces locomotor behaviour in animals, being the speed and amplitude of movements directly proportional to the intensity of stimulation. Movements like turning, can be obtained by assymetrically stimulating the MLR. Increasing the simulation in the MLR also induces gait transitions in vertebrates.

The salamander robot is an amphibian robot, thus capable of both walking and swimming, that is inspired in real salamanders regarding both its kinematic structure and locomotion. The robot was developed at EPFL's Biorobotics Laboratory and its purpose is threefold: (i) to serve as embodyment for CPG models, that can be studied at neuron level, but whose efectiveness in locomotion needs a body to be tested; (ii) to compare the resulting gaits to the ones of the real animal; (iii) to show that CPGs can lead to robust locomotion control for robots with multiple articulated joints ([2]).

2.3.1 A test-bed for CPG models

Even though the structure of the locomotor circuit of the salamander is known, there is no solid background for how the oscillatory centers are coupled and how gait transitions occur at lower levels. Also, some aspects of salamander's locomotion still lack explanations, for instance, the reason why swimming frequencies are consistently higher than walking frequencies. Researchers at Biorob have proposed a CPG model that addresses these topics which is based on the following hypothesis:

1. The spinal CPG is similar to that of the lamprey and generates travelling waves when activated, while the limb CPG, when activated, forces the spinal CPG to the walking mode.

2. The limb to body coupling (coupling between limb and body oscillators) is stronger than the body to body coupling (coupling between body oscillators), thus the walking gait "overruns" the swimming gait when limbs are activated.

3. Walking frequencies saturate at a certain value, after which limbs stop and give place to the swimming gait.

4. For the same drive, walking frequencies are lower than swimming frequencies. This explains the rapid increase in the frequency of movements that is seen in the real salamander when switching from walking to swimming.

The CPG model is implemented by means of a system of coupled non-linear oscillators, where each oscillatory center is an amplitude controlled phase oscillator. Details on this implementation can be found in [13], [2] and [9], while the CPG model is described in detail in chapter 4.
The results of this experiment were quite impressive, showing that the proposed model could actually produce swimming and walking patterns consistent with the ones of the real salamander with abrupt gait transitions controlled by the drive.

2.4 The kinematics of locomotion

Quadrupedal gaits

Quadruped locomotion in vertebrates consists in the coordinated movement of the limbs in order to achieve different motion patterns. In [19], Alexander, R., defines a few necessary

concepts to correctly analyse gaits. For instance, a **stride** is a complete cycle of limb moving, e.g., from the instant when a limb touches the ground to the next setting down of the same foot. A stride comprises two distinct phases: the **swing** phase, during which the limb is off the ground, and the **stance** phase, when the limb is on the ground. The **stride length**, is defined as the distance travelled by the body during a stride, while the **stride frequency** is the number of strides per time unit and consequently the **stride duration**, t_{stride}, is the duration of a stride. The following equation verifies:

$$t_{stride} = t_{swing} + t_{stance} \tag{2.15}$$

where t_{swing} and t_{stance} are the duration of swing and stance phases respectively.
Another important parameter is the **duty factor**, here noted as DF, that is given by the fraction of the stride duration that a limb spends on the ground. It is defined as:

$$DF = \frac{t_{stance}}{t_{stride}} = \frac{t_{stance}}{t_{swing} + t_{stance}} \tag{2.16}$$

Walking and running gaits are commonly distinguished by their duty factors. Walking gaits usually present $DF > 0.5$ (both feet of a pair are on the ground for some time during the stride) while running gaits have $DF < 0.5$ (during a stride both feet must be simultaneously off the ground for some time). The variation of duty factor occurs in biology most commonly as a consequence of adjusting stance duration, since the speed of locomotion has a linear relation with the inverse of the stance duration, while the swing duration is almost constant for any gait ([3], [11]).

2.5 The role of sensory feedback

It has been mentioned that CPGs do not need sensory input to generate rhythmic patterns - rhythmic activity is generated by oscillatory circuits made of neurons with higher-level control from the brain. Nevertheless, the fact that sensory input is not necessary to generate rhythmic activity does not imply that it is of no use for locomotion at all. Several experiments have shown that sensory feedback is strongly coupled to CPG activity as it shapes the generated rhythmic patterns. In an experiment of such, Rossignol shows that a decerebrated cat placed on a treadmill can produce normal looking walking gaits and even gait transitions

when the treadmill is accelerated, suggesting that the perception of speed namely by the limbs modulates the frequency of the rhythmic patterns in CPGs. Also, according to Grillner ([5]), stretch receptors are essential to the generation of body undulations in lampreys as they feed the state of muscle contraction back to the CPG which acts according to it. The mechanical movement of the lamprey's tail also induces CPG activity whose frequency is the same as that of the mechanical movement ([13]).

Concerning sensory feedback in stepping gaits, it is known that the force sensing under the feet strongly influences the activity of biological CPGs. Essentially, this is the input that modulates the onset of swing and stance phases of walking vertebrates' CPGs, whose oscillators produce different responses depending on whether the limbs are on or off the ground. The correct onset of swing and stance phases has been shown to improve the speed of quadruped robot locomotion ([3], [4]) especially in non-flat terrains. The inclusion of sensory feedback in the control loop of a robot results in a CPG that is controlled by sensory information. It provides a strong coupling with the mechanical system, as the flow of the control occurs in both directions - the CPG generates the control signals that will act on the environment and the sensors retrieve information on which the CPG phase will depend, hence affecting the generation of further control signals.

Prior to this project, no studies were known concerning the inclusion of sensory feedback in the walking gait of the salamander robot. Ijspeert et al., for instance, in [10], use a mechanical model of the salamander to study the generation of the S-shaped waves on ground. This is done using sensory feedback from stretch sensitive sensors from the sides of the body on a CPG that naturally produces travelling waves. Nevertheless, no feedback from the limbs was included.

Concerning this project, the implemented CPG model, as well as the results of its application, are described in chapters 4 and 6. In chapter 4, the architecture of two different controllers is explained, one of which uses sensory feedback from touch sensors on the limbs to correctly identify swing and stance phases - the closed-loop controller - and the other uses no sensory information - the open-loop controller. The closed-loop controller, differs from the open-loop controller simply by the positions at which happens the onset of swing and stance phases. While in the case of the closed-loop controller a limb is in stance phase whenever it is in contact with the ground, in open-loop the joint angles to start each of the phases have to be previously set. Chapter 6 shows the advantages of using each of these two types of controllers.

Chapter 3

Simulation environment

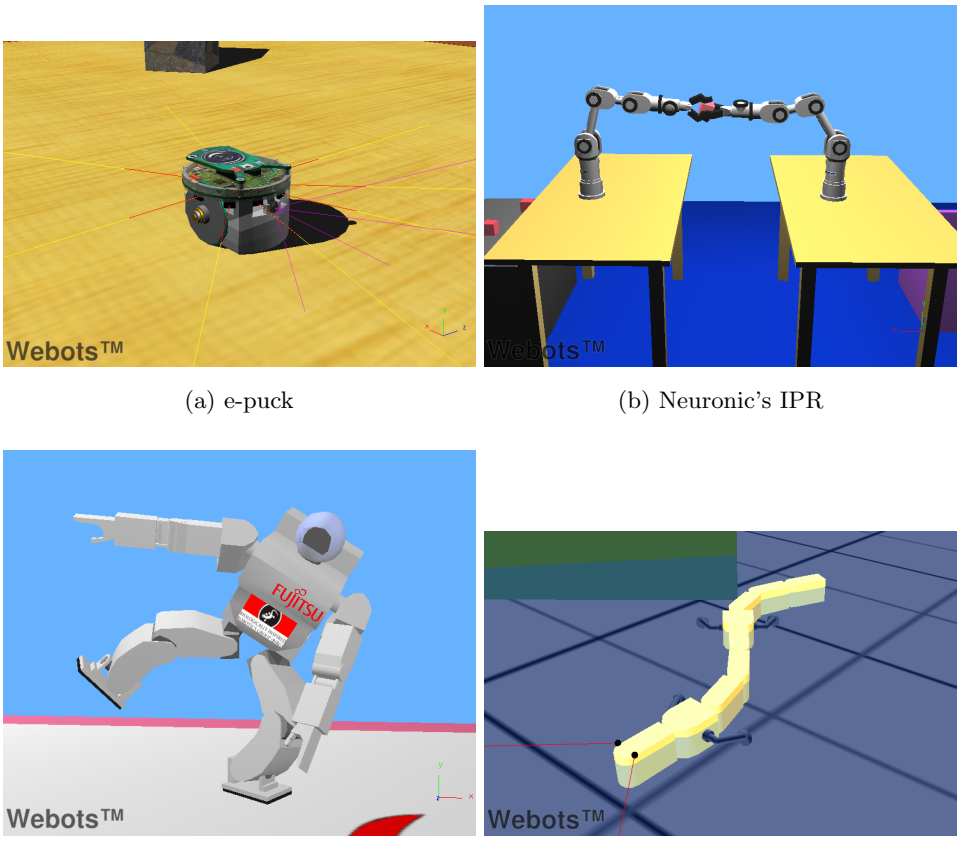
3.1 Webots - Mobile robotics simulation software

Webots is a software developed by Cyberbotics Ltd. ([20]) and the Swiss Federal Institute of Technology in Lausanne, which is commonly used to model, program and simulate mobile robots. Even though Webots comes with a few pre-defined commercial robot models (Aibo™, Khepera™, e-puck™(fig. 3.1a), Neuronics IPR™(fig. 3.1b), HOAP™(fig. 3.1c), to mention a few of the most well-known) it is possible for users to develop their own models. This opens a very wide range of opportunities for robotics researchers since it makes it possible to program the robots' controllers and use Webots environment as a test bed before using real robots. The controllers that are programmed in Webots can be uploaded to most of the comercial robots that have a corresponding model in simulation.

Webots allows the representation of passive objects, comprising a wide variety of shapes, and active objects, these ones known as mobile robots. Mobile robots are usually equipped with sensor and actuator devices, e.g. distance sensors, cameras, gyroscopes, force sensors, for the former and driving wheels and servomotors for the latter. A world in Webots is a three-dimensional representation of the robots and their environment being each object characterized by properties such as its shape, texture, physical properties, position, orientation, etc.

Webots relies on the Open Dynamics Engine (ODE, [21]) for physics simulations. Consequently, it is possible to simulate complex interaction forces such as friction, fluid dynamics or aerodynamics. Figure 3.1d, for example, shows the salamander robot in water, where the buoyancy of the body is simulated using Archimedes' principle. In this example, a physics plu-

gin is used to estimate the propulsive forces generated by body undulations and the reaction forces acting on the robot.



(a) e-puck (b) Neuronic's IPR

(c) HOAP (d) Salamander robot

Figure 3.1: Examples of robot models in Webots.

Robot controllers can be written in C/C++, Java, URBI, Python or MATLAB™. In this project every controller was written in C language.
The Webots User Guide, [22], provides a deeper explanatory introduction to Webots capabilities as well as several examples of applications.

3.2 The salamander robot model

Following the trend of most roboticists, researchers at Biorob have developed a model of the salamander robot for simulation and testing in Webots. This project makes use of the salamander robot's model as a test bed for the proposed control strategies as it offers many advantages when compared to using the real robot. Firstly, Webots allows the use of simulated touch sensors that can be added to any part of the robot, in simulation. These sensors

would be hard to integrate in the real robot as they would have to still be developed and placed on the limbs, consuming a high amount of time and going too much beyond the scope of the project. Secondly, in Webots it is possible to obtain real time coordinates of the robot's position at every simulation step. This allows very accurate estimations of speed and robot's trajectory, which were the main performance indicators that were used to compare the controllers. It would have been much harder to obtain accurate measurements of these two properties in the real robot. Finally, Webots has a fast simulation mode which was used in order to optimize the open-loop controller. Fast simulation mode allows accelerating simulations by neglecting the graphical representation of the simulated objects, thus reserving the computational resources available to the physics engine. By doing this it is possible to save enough time to sistematically test high ranges of control parameters, obtaining the combinations that lead to the best performances (optimization by systematic search, chapter 5).

There are a few different models of the salamander robot, with just minor differences between each of them, essentially regarding their shape. The model that was used in this project was the one represented in Figure 3.2. This model consists of 9 body segments with one active joint between each two segments, making a total of 8 joints, and 4 additional joints for position control of the limbs, which are attached to the second and sixth body segments. These joints correspond to a specific type of node in Webots hierarchy, the Servo nodes, that are placed between parent and children nodes allowing these two to move with respect to one another. Rotational joints are, in this case, abstractions of the electric motors used in the real robot, providing position, velocity and force control in simulation (see Webots Reference Manual, [23], for a more detailed explanation on Servo nodes). Body joints allow the bending of the salamander robot's body, which is essential for locomotion both in water and on ground, while limb joints make walking gaits possible.

3.2.1 Physical properties

Physical properties like geometry, weight and friction are important parameters of the robot model as they have direct influence in locomotion performance. Physics nodes are a specific type of node in Webots hierarchy that attributes physical properties such as mass distribution, density and friction to objects thus allowing the physics engine to compute more realistic forces. The friction between limbs and ground influences the performance of the walking gait

while body weight plays an important role on the swimming gait. The Webots model of the salamander robot was created with standard values for these properties.

Geometry

The geometry of the robot model was made quite similar to the real robot's. The length of each body segment is around 10cm, while width and height are 5cm and 6cm respectively. The second and sixth segments are slightly larger, with 6.5cm of width (3.2a). The total length of the robot is, hence, about 90cm. Limbs are also quite simple parts, as they are modeled by cylinders approximately 10cm long and bent by an angle of about 108°, with round tips to provide a smooth contact with the ground (3.2b).

Weight

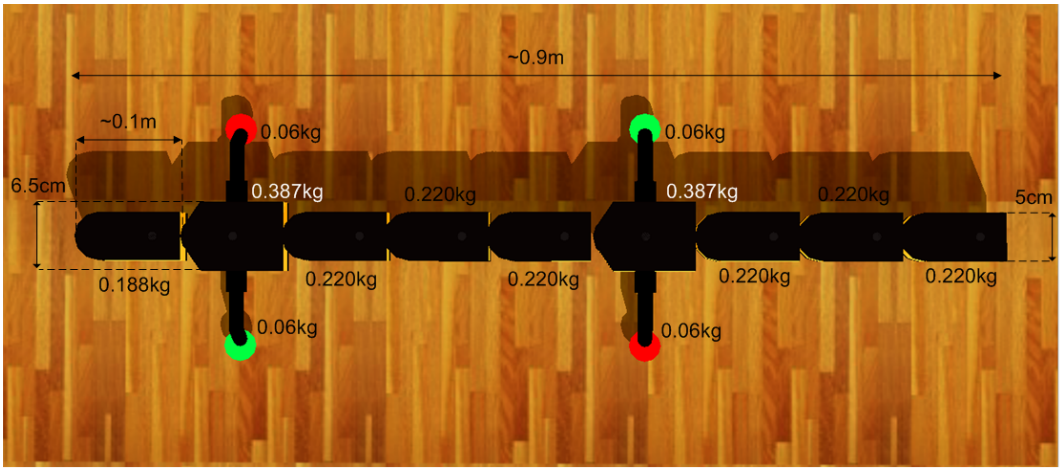
Regarding the weight of each segment of the robot's body, the weight distribution is depicted in figure 3.2a. The head segment is the lightest, weighing 0.188kg, while the remaining weigh 0.220kg, except for the second and sixth segments which weigh 0.387kg. Additionally, each limb weighs 60g, thus making the robot weigh a total of 2.522kg in simulation.

Friction

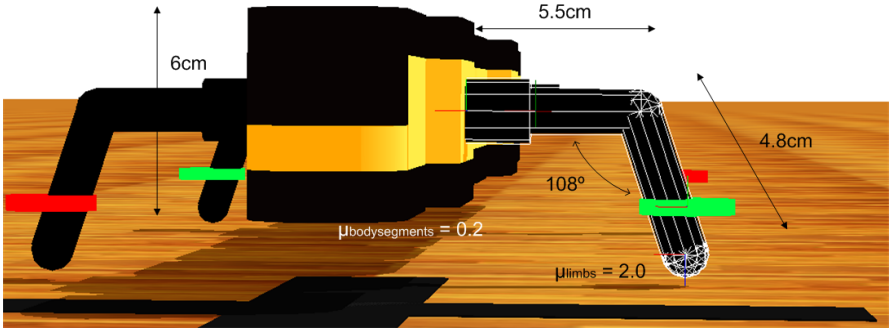
The friction model used by Webots is based on the Coulomb friction model, which establishes the following relation between the tangential and normal forces, F_T and F_N, acting on a surface:

$$F_T \leq \mu \cdot F_N \tag{3.1}$$

In this equation, the parameter $\mu > 0$, called friction coefficient, limits F_T, the tangential force due to friction, to a value between 0 and μF_N. The way how ODE applies this model is by making $F_T = \mu F_N$ and computing the resulting forces acting on every contact point with this fixed limit ([21]). Consequently, a body will only move along a surface if the applied force exceeds F_T along the direction of motion. The standard friction coefficient of body segments is $\mu_{bodysegments} = 0.2$, while the one of the limbs is $\mu_{limbs} = 2.0$ (figure 3.2b).



(a) Top view



(b) Front view

Figure 3.2: The robot's physical properties.

3.2.2 Joint position control

Position control is made by means of a P-controller, that is inherent to each Servo node, where the desired position results from the output of a Hopf oscillator from the CPG network (topic covered in chapter 4). The P-controller outputs the Servo's velocity to reach the desired position. Concerning the proportional gain, P, of each joint, limb joints were chosen to have $P = 800$, while body joints use $P = 10$, the standard value for P in Webots. These values were chosen so that limb position error was minimized (without compromising the stability of the controller), since ground locomotion is highly dependent on limb accuracy, while body joints were allowed to have a slight error margin to provide some smoothness to oscillations.

3.3 The world models

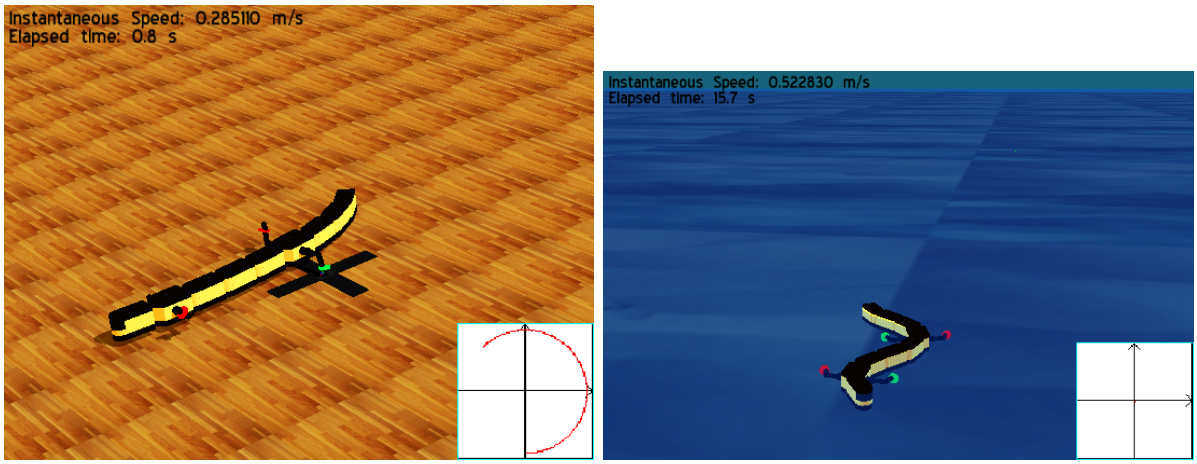
The real robot may be tested in an almost infinite diversity of environments either in land and in water, reason by which there are two standard models that are currently being used in Webots to simulate these two distinct environments.

3.3.1 Flat ground model

The standard flat ground model consists of a 100x100m plane with 0°inclination on which the walking gait of the salamander robot is tested (figure 3.3a). As the ground does not have a Physics node, the friction coefficients between the robot and ground at every contact point are the ones defined for the robot's body in 3.2.1 - Friction.

3.3.2 Water model

The water world model (Figure 3.3b) uses Webots' physics plugin in order to simulate hidrodynamics and consequently the interaction forces between the robot and the water.



(a) Standard flat terrain.

(b) Water world model.

Figure 3.3: Webots' flat ground and water models for testing walking and swimming gaits.

Initially, the inclusion of sensory feedback in the robot in the water world was studied but eventually it was disregarded. The idea was to use torque measurements from body joints to detect the intensity of the water flow and perform an automatic adjustment of swimming parameters (frequency and phase lag) to be better suited for locomotion in different flows. Ideally, since position control is used to control joints, higher torques would be needed for

higher intensity waterflows and vice-versa. Nevertheless, due to an apparent lack of accuracy of torque measurements in Webots, this approach was not followed.

3.3.3 Standard world modifications

For the accomplishment of the proposed goals, i.e. testing controllers on different terrains, the standard flat ground model was modified in order to represent different conditions: friction coefficients were changed as to simulate different degrees of slipperiness, slopes were added to simulate inclined planes, roughness was simulated by randomly distributing peaks of small height throughout the ground and also holes were added to the standard flat ground model. All these scenarios are described further in chapter 6 where the performance results are also depicted.

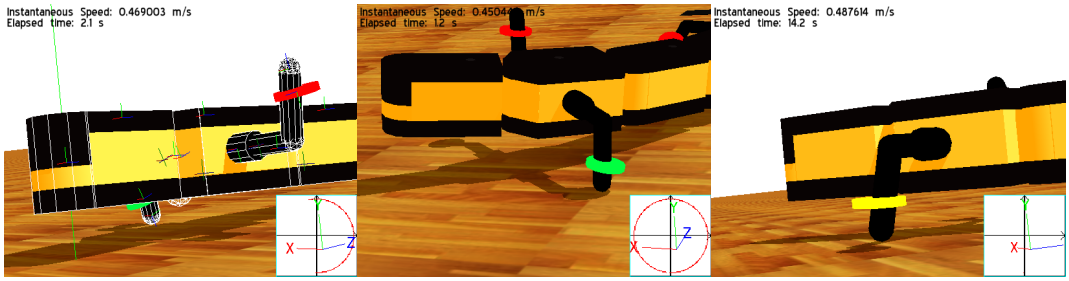
3.4 Simulation layout

All the Webots models that are used share a few common aspects regarding their layout. On the leftmost corner of Figure 3.3a it is represented the instantaneous speed of locomotion. This speed is estimated everytime the robot's head completes a full rotation and is based on two consecutive measurements of position. The display in the bottom-right corner of the screen shows the phase space of the oscillator of the left frontal limb. It is particularly useful to clearly see the positions at which swing and stance are being started as well as the general time evolution of the oscillator's phase. It is also possible to see a ring around every limb of the salamander robot. These rings, in figure 3.4, indicate the current frequency of the oscillator, ω, of the corresponding limb, thus allowing to understand in which phase of locomotion it is. The meaning of each color is:

- Red - $\omega = \omega_{swing}$. It means the robot is in swing phase.
- Green - $\omega = \omega_{stance}$. The robot is in stance phase.
- Yellow - $\omega = 0$. Limb is stopped, waiting for reaching the ground.

Limb stopping is an additional locomotion phase that was introduced in the controller in an attempt to provide some stability to the CPG network, when operating with sensory feedback, to avoid skipping the stance phase. This is a natural response in many locomotor systems found in nature since in most cases limbs support the weight of the body so skipping a stance

increases the chances of falling. Further detailed information on this choice can be found in chapter 4.



(a) Swing phase. (b) Stance phase (c) Limb stopping.

Figure 3.4: Visual indicators of locomotion phase

3.5 Time steps

Time steps play a very important role in the accuracy of Webots simulations. The simulation time step is the time interval between each two computations of the forces acting on the simulated objects ([22]), so this parameter should be as low as possible in order to better aproximate real world conditions where this value is infinitely small. The control step represents the time between each two control commands (e.g. sensor readings, joint actuation) ([22]).

3.5.1 Simulation step

The speed and degree of accuracy of each simulation are directly and inversely proportional to the value of the simulation step, t_{sim}, respectively: in order to get a physically accurate simulation, the total duration of each simulation increases as a consequence of the increasing number of computations. The value of t_{sim} was chosen to be:

$$t_{sim} = 2ms \tag{3.2}$$

This value results in simulations of acceptable duration without too much risk for physical accuracy.

3.5.2 Control step

The control step is the time interval at which sensor and actuator data is updated. Consequently, as the execution of a simulation step is atomic, i.e., it cannot be interrupted, the control step, $t_{control}$, must be a multiple of t_{sim} ([22]). For the present case, $t_{control}$ could be made $t_{control} = 2, 4, 6, 8, 10, ...$ In order to make each simulation step the result of exactly one actuation instruction, $t_{control}$ was made equal to t_{sim}:

$$t_{control} = 2ms \tag{3.3}$$

3.5.3 Integration step

In chapter 2, it was mentioned that the oscillators are governed by sets of differential equations that need to be numerically integrated in order to obtain the oscillator variables at every instant. For the present work, the numerical integration of the oscillator's equations (refer to chapter 4 for a full description of the CPG model) was done by using Euler method with the integration step $\Delta t = 2ms$ (in the real robot this integration is performed in a microcontroller in the head, using also Euler method, with $\Delta t = 10ms$, see [9]). For instance, being x a generic variable of the oscillator, its time evolution is given by:

$$x(t) = x(t - \Delta t) + \dot{x}(t - \Delta t) \cdot \Delta t \tag{3.4}$$

Chapter 4

CPG network architecture and oscillator model

4.1 CPG models for the salamander

In [8], Ijspeert et al. propose several CPG models for the salamander, under the assumption that the body CPG is similar to that of the lamprey and automatically generates travelling waves when activated. These models consist of separate limb and body CPGs whose oscillators may be coupled in several differents ways. However, the most widely used CPG model for the control of the salamander robot is the one implemented in [13] and [2] which presents two CPG networks, one for the body and one for limbs. The body CPG is composed of a double chain of 16 oscillators (8 pairs) with nearest-neighbour coupling while the limb CPG has 4 oscillators with interlimb coupling between forelimbs and hindlimbs (figure 4.1). The left and right chains that comprise the body CPG represent the chains of oscillatory centers in both sides of the salamander's spinal cord that control muscle activity on the left and right side of the body, by contracting and extending alternately in phase opposition. This double chain, however, is not strictly necessary for the control of the robot, as it can be performed with a single chain of 8 oscillators, one per body joint.

Coupling between oscillators is a consequence of the hypothesis behind the model (section 2.3.1):

1. The body CPG is programmed to generate travelling waves whenever activated. Thus, coupling between body oscillators is such that the left and right chains are in phase opposition (when one side of the body contracts, the other extends) and each two con-

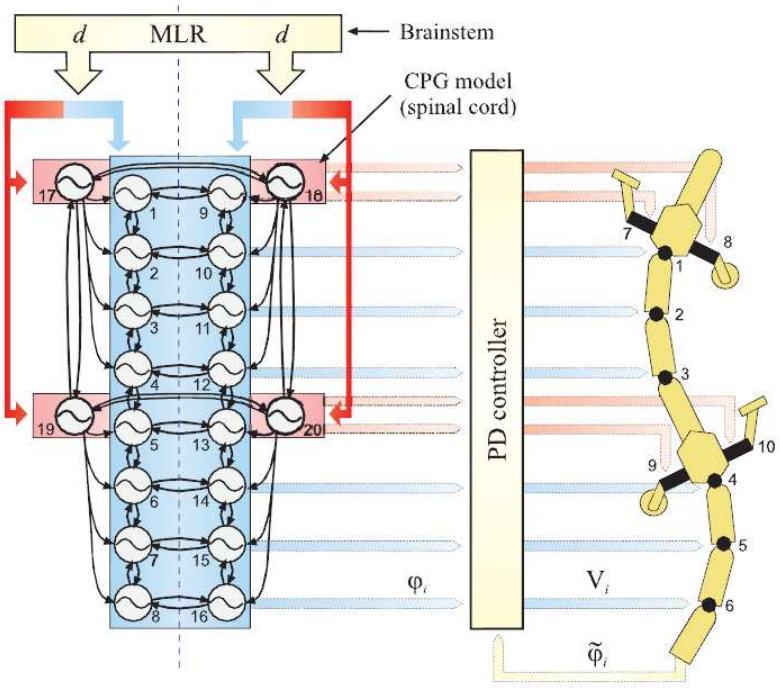


Figure 4.1: CPG network for control of locomotion of the salamander robot proposed by Ijspeert et al. in [2]. ϕ_i and $\widetilde{\phi}_i$ represent the desired and actual joint angle of servomotor i, while V_i is the corresponding control voltage.

secutive pair of oscillators have a constant phase lag.

2. The unidirectional limb-body coupling is used so that, in the walking gait, the limb CPG can force the body CPG to generate S-shaped standing waves instead of travelling waves. Besides projecting to each other, each oscillator of the limb CPG also projects to the closest 4 oscillators of the body CPG, as in figure 4.1. This is how synchronism between limb motion and body bending is attained.

3. Because in walking gait only diagonally oposed limbs are in phase, the two forelimbs are coupled and in anti-phase, just like the two hindlimbs and each two limbs on the same side.

The 8 pairs of body oscillators control 6 real elements, numbered 1-6 in figure 4.1, and 2 fictive joints placed in the limb elements. The setpoints, i.e., the desired joint positions of the

robot's servomotors (ϕ_i), are, for the body joints, obtained by taking the difference between the x_i signals from the left and right oscillators ([9]):

$$\phi_i = x_i - x_{i+8} \tag{4.1}$$

Moreover, limb setpoints are given by:

$$\phi_i = h(\theta_i) \tag{4.2}$$

where θ_i is the phase of oscillator i and $h()$ is a function that translates the oscillator's phase ($\theta_i \in (-\pi; \pi)$) to a monotonically increasing phase signal, that is necessary to control the limb joints.

Setpoints, ϕ_i, along with the actual joint angles ($\widetilde{\phi_i}$), serve as input to a proportional-derivative (PD) controller that outputs the voltage to control the motors.

In this model, the CPG activity is controlled by a simple drive signal, representing the stimulation coming from the brain stem, which is used to control the amplitude and frequency of the oscillators and consequently the gait. Nevertheless, this is a biological requirement for the control of the real animal that is not strictly necessary to control the robot, as frequency and amplitude parameters can be adjusted directly without intervention of a drive signal.

Regarding the present work, a simplified version of this network was used to generate the control policies for the robot (figure 4.2). The 16-oscillator body CPG was simplified to an 8-oscillator CPG from which the setpoint of body joints is taken, and a limb CPG that outputs the limb joint positions. Like the CPG network used in [2], the coupling between limb and body CPGs is unidiectional with only the limbs projecting to the body and not the opposite. The oscillators in the body CPG are coupled to their nearest neighbours, while the oscillators in the limb CPG are coupled with the two forelimbs in antiphase with each other, just like the two hindlimbs and each two limbs on the same side. This ensures that diagonally oposed limbs are in phase, while limbs on the same side of the body are in phase opposition, which is what happens in the real animal. Concerning the limb to body coupling, the forelimbs project to the first 5 body oscillators which correspond to the first five servomotors, while the hind-limbs project to the last three. Forelimbs must be coupled to trunk oscillators (the first five) in order to attain the synchronism that maximizes stride length as previously described in 2.3.

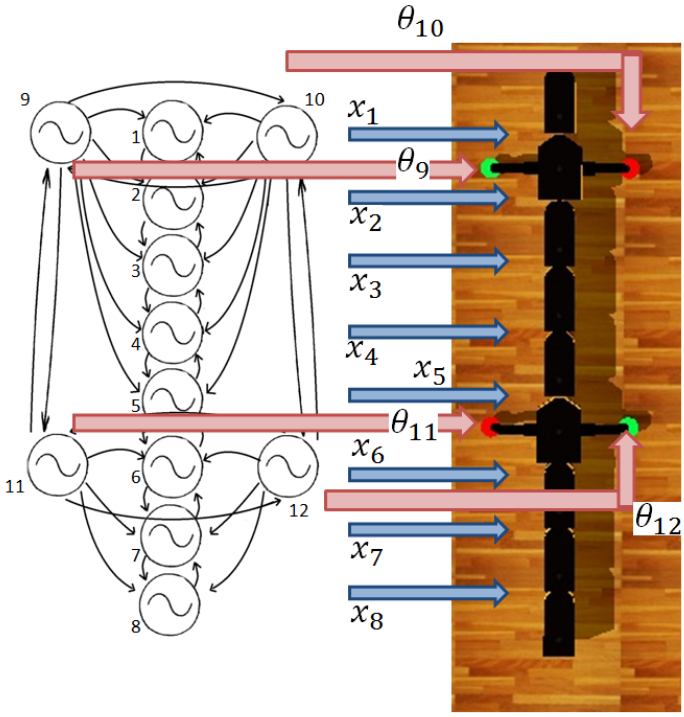


Figure 4.2: CPG network for control of locomotion of the salamander robot used in this project. The x_i variables from the oscillators are used to control the body joints, while the phase of the limb oscillators, θ_i, is used to set limb position.

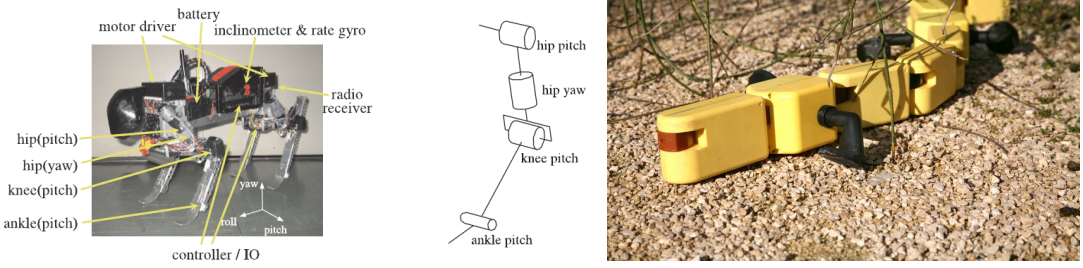
With the suggested coupling, any eventual change in the phase space of the limb oscillators as consequence of sensory feedback, will have an effect in all the remaining oscillators in the network.

4.2 Oscillator model

4.2.1 Hopf oscillators to control locomotion phase duration

After defining the architecture of the network, it is necessary to define the model of the oscillators that will be used. In [4], Kimura et al. propose a CPG model consisting of neural oscillators to control the quadruped robot *Tekken2*, whose limbs have 4 joints each, using sensory feedback. This model shows a complexity which is appropriate for a robot like *Tekken2* whose limbs have many degrees of freedom, but too complex for the salamander robot (figure 4.3) whose limb motion occurs around a single joint.

In [3], Righetti and Ijspeert propose the utilization of Hopf oscillators for the control of 3 different quadruped robots using sensory feedback. The proposed model of Hopf oscillator



(a) Tekken2 robot ([4])

(b) Salamander robot [24]

Figure 4.3: Complexity of the limbs of the Tekken2 robot and the salamander robot.

has the property of allowing to independently control the duration of swing and stance phases. Sensory feedback from touch sensors on the feet, is then used as an input from the environment that tells the CPG when the limb is on or off the ground, thus controlling the onset of the locomotion phases by explicitly changing the phase space of the oscillators according to their current state and sensor values. The control policy is the x variable and it represents joint angle of hip or shoulder joints.

The complete model of this oscillator, including the term for sensory feedback, is given by the following set of equations, for an oscillator i ([3]):

$$\dot{x}_i = \alpha(\mu - {r_i}^2)x_i - \omega_i y_i \tag{4.3}$$

$$\dot{y}_i = \beta(\mu - {r_i}^2)y_i + \omega_i x_i + \sum_{j=1}^{4} k_{ij} y_j + u_i \tag{4.4}$$

$$\omega = \frac{\omega_{stance}}{e^{-by_i}+1} + \frac{\omega_{swing}}{e^{by_i}+1} \tag{4.5}$$

where $r_i = \sqrt{{x_i}^2 + {y_i}^2}$, ω_i is the frequency of the oscillations in $rad.s^{-1}$, x_i and y_i are the i^{th} oscillator's variables, $\sqrt{\mu}$ is the amplitude of oscillations and α and β are parameters that control the speed of convergence to the limit cycle. The term $\sum_{j=1}^{4} k_{ij} y_j$ is the coupling term that couples oscillator i to oscillator j in the newtork, with k_{ij} being defined by the coupling matrix. Equation 4.5 works as a step function that selects either ω_{stance} and ω_{swing} according to the sign of the y variable, so that for $y_i > 0$ the limb will be in stance phase, while for $y_i < 0$ the limb will be in swing phase. b controls the speed of the switch. The feedback modulation is done by the term u_i which can take three different values depending on the sensory input:

$$u_i = \begin{cases} -sign(y_i) \cdot F, & \text{if} \quad \text{fast transitions}, \\ -\omega x_i - \sum_{j=1}^{4} k_{ij} y_j, & \text{if} \quad \text{stop transition}, \\ 0, & \text{otherwise} \end{cases} \tag{4.6}$$

where F is a parameter that controls the speed of transitions.
The inclusion of sensory feedback introduces two new phases in the step cycle, *stopping before transition* and *fast transitions*, that are initiated by the controller when the touch sensors detect specific conditions([3]):

1. *Stopping before transition* - The oscillator should stop in two cases, during swing to stance transition when the limb is not yet in contact with the ground and during stance to swing transitions, when the limb still supports significant body weight (i.e., to avoid premature lifting of the limbs). The dark grey areas in figure 4.4 indicate the activation zones for limb stopping. At these points of the oscillator cycle, if any of the mentioned cases verifies (touch sensors provide the necessary feedback), the term $u_i = -\omega x_i - \sum_{j=1}^{4} k_{ij} y_j$ is applied to equation 4.4 with the system converging to one of the fixed points $(x_i, y_i) = (\pm\sqrt{\mu}, 0)$, depending on which phase it is (righ-most image of figure 4.5). The limb stops at this position and u_i is set again to zero when the the stopping conditions no longer verify, with the oscillator returning to its normal phase plot (left-most image in figure 4.5).

2. *Fast transitions* - Fast transitions should occur in two cases: during stance when the weight under the foot becomes low (transition to swing) and during swing when the foot touches the ground (transition to stance). These transitions can be initiated when the oscillator phase is in the light grey areas as displayed in figure 4.4. The transition is performed in the phase space by making the y variable of the oscillator go to 0. The F parameter is chosen so that $\dot{y} \approx -sign(y)F$ (the sign of F depends on the direction of the transition), thus y goes to 0 at a speed that is a function of F (center image in figure 4.5). When y reaches the x axis of the phase space, u_i is re-set to 0 and the oscillator converges again to the limit cycle, with a new frequency that is different from the one it had before transition (either ω_{swing} or ω_{stance}).

Hopf oscillators are a special case of oscillators, in which the geometrical angle defined by its variables, x and y, directly give the phase of the oscillator. Thus, when addressing to the

phase of a Hopf oscillator the terms *phase* and *angle* will be used interchangeably.

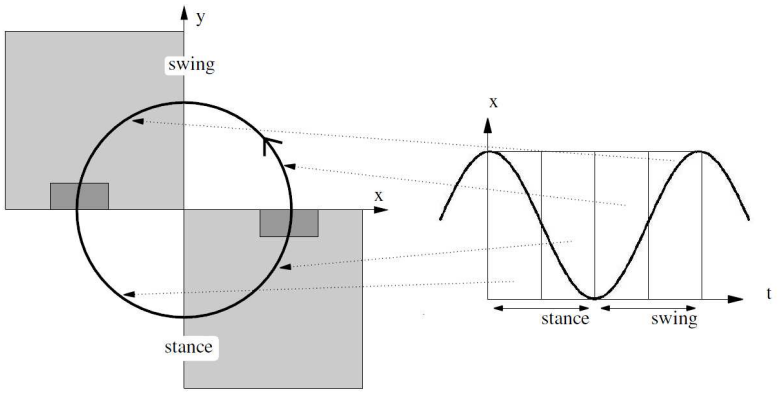


Figure 4.4: Feedback activation zones in the phase space. Light grey is the activation zone for fast transitions and dark grey for stop controls.

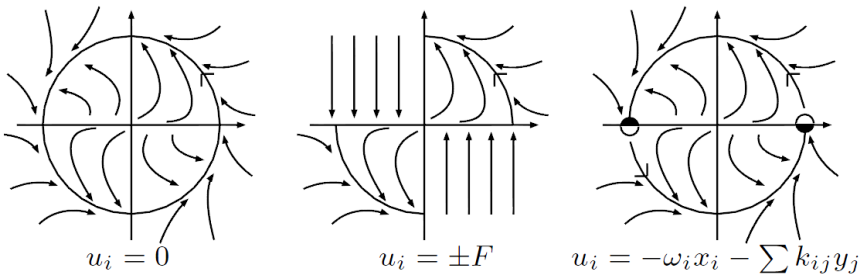


Figure 4.5: Effect of the feedback on the oscillator's phase space for the oscillator model used in [3].

4.2.2 Using the proposed model in the salamander robot

The previous oscillator model, was successfully implemented in the referred quadruped robots (iCub, Aibo, Ghostdog). For the case of these robots, a network of four oscillators was used, with an oscillator for each limb, with the x variable of each oscillator ($x_i \in [-\sqrt{\mu}; \sqrt{\mu}]$) corresponding to the desired joint angle in the sagittal plane (shoulder joints or hip joints). Because the limbs of the salamander robot need to make complete rotations, their setpoints must increase monotonically, instead of having rythmic movement like the ones of the iCub, Aibo or Ghostdog. For this reasion, the x variable cannot be used as control policy. Also,

the angle of the oscillators cannot be used to determine the salamander's limb position if the oscillator's equations remain exactly as defined. The most evident reason for this limitation is that, during a fast transition from swing to stance, when a limb should slow down, the angle of the oscillator is actually accelerating because y is rapidly going to zero. Therefore, fast transitions in the salamander robot cannot be implemented the way they have been described until now. The solution that was devised to provide the robot with a mechanism to rapidly switch limb velocity involves modifying the oscillator's equations from the work of Righetti and Ijspeert ([3]). The control policy that is used is, indeed, the angle of the oscillators, so the setpoint of joint i is given by:

$$\phi_i = \theta_i = atan2(x_i, y_i) \tag{4.7}$$

Figure 4.5 shows an example of the correspondance between oscillator phase and limb position. In order to use the sensory feedback to modulate the phase onset, the equation 4.5 is modified to directly incorporate sensory feedback:

$$\omega_i = \frac{\omega_{stance}}{e^{\gamma_i} + 1} + \frac{\omega_{swing}}{e^{-\gamma_i} + 1} \tag{4.8}$$

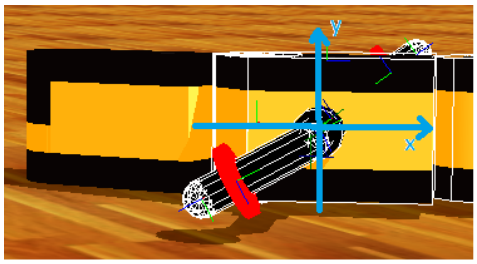
where:

$$\gamma_i = \begin{cases} -1000, & \text{if} \quad \text{limb } i \text{ is on the ground,} \\ 1000, & \text{if} \quad \text{limb } i \text{ is off the ground,} \end{cases} \tag{4.9}$$

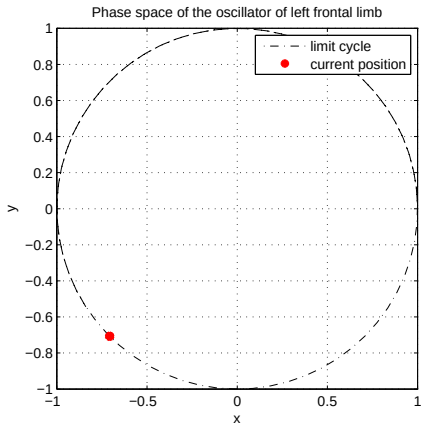
This expression can be further simplified, due to the fact that 4.8 is a step function, to:

$$\omega_i = \begin{cases} \omega_{stance}, & \text{if} \quad \text{limb } i \text{ is on the ground,} \\ \omega_{swing}, & \text{if} \quad \text{limb } i \text{ is off the ground,} \end{cases} \tag{4.10}$$

The detection of whether a limb is on or off the ground is made by simulated touch sensors in Webots. By instantaneously changing the frequency of the oscillators, the fast transitions are now performed in a way that is consistent with the control limitations of the salamander robot. Since $\phi_i = \theta_i$, the angular speed of each limb will now be the same as the frequency of the corresponding oscillator.



(a) Limb position



(b) Oscillator phase

Figure 4.6: On the right the phase of the oscillator, and on the left, the position of the limb.

The term u_i is removed from equation 4.4 since fast transitions are no longer performed that way and limb stopping is done as follows.

4.2.2.1 Limb stopping

In order to prevent one limb from completing a full rotation without touching the ground, limb stopping is used as well by making the limb stop every time it passes by the position of -90° without touching the ground. This is equivalent to setting ω_i, the frequency of the oscillator i, to zero when the corresponding limb does not touch the ground:

$$\omega_i = \begin{cases} 0, & \text{if} \quad \theta_i = -90° \text{ and limb is not on the ground,} \\ \omega_{stance}, & \text{if} \quad \text{limb } i \text{ is on the ground,} \\ \omega_{swing}, & \text{if} \quad \text{limb } i \text{ is off the ground,} \end{cases} \tag{4.11}$$

4.2.2.2 Body oscillators

Regarding the oscillators for the body CPG, the model that will be used is the same as for the limbs but with a constant frequency, ω_{spine} (defined in 4.4). The amplitude of the oscillators is set to $\mu = 1.0$ so the setpoint of the body joint i, ϕ_i, is obtained by multiplying x_i by the desired amplitude:

$$\phi_i = A \cdot x_i \tag{4.12}$$

Figure 4.2 depicts this relationship between oscillator variables and robot joints.

4.3 Open- and closed-loop controllers

Since a CPG network capable of integrating sensory feedback from touch sensors has now been defined, it is necessary to explain the controllers that will be compared. The closed-loop controller, whose architecture was extensively described in sections 4.1 and 4.2 differs from the open-loop controller simply by the angles at which happens the onset of swing and stance phases. While in the case of the closed-loop controller a limb is in stance phase whenever it is in contact with the ground, in open-loop the joint angles to start each of the phases have to be previously set. These angles, $\phi_{stanceonset} \in [0; 3 \cdot \frac{\pi}{2}]$ and $\phi_{swingonset} \in [-\frac{\pi}{2}; 0]$ (figure 4.7), will influence the global frequency of motion, f_{global}, given by:

$$f_{global} = \frac{1}{t_{swing} + t_{stance}} \tag{4.13}$$

where,

$$t_{swing} = \frac{\phi_{stanceonset} - \phi_{swingonset}}{2\pi f_{swing}} \tag{4.14}$$

$$t_{stance} = \frac{2\pi + \phi_{swingonset} - \phi_{stanceonset}}{2\pi f_{stance}} \tag{4.15}$$

Therefore, the frequency of a limb oscillator i in open-loop is given by (recalling that θ_i is the phase of oscillator i, according to eq.4.7):

$$\omega_i = \begin{cases} \omega_{swing}, & \text{if} \quad \phi_{stanceonset} \leq \theta_i \leq \phi_{swingonset}, \\ \omega_{stance}, & \text{otherwise,} \end{cases} \tag{4.16}$$

For every pair of frequencies $\omega_{swing} = 2\pi f_{swing}, \omega_{stance} = 2\pi f_{stance}$ the speed in open-loop highly depends on the angles to start each of the locomotion phases. For this reason, taking into account that the closed-loop controller will be compared to the open-loop controller, it is necessary to optimize the open-loop controller for speed, on these two variables, $\phi_{swingonset}$ and $\phi_{stanceonset}$. The results of this optimization are described in the next chapter.

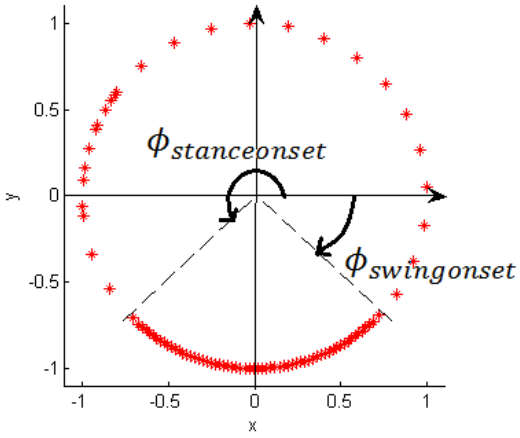


Figure 4.7: Representation of the phase space of a limb oscillator with $\phi_{stanceonset}$ and $\phi_{swingonset}$. Red dots are samples of the oscillator's time evolution (more dense in stance phase due to lower angular speed).

4.4 Inter-oscillator coupling

To provide limb and body coordination, the phase relations between each oscillator in the network has to be defined. For the present case, as in [2] and [8], diagonally oposed limbs are in phase, while the two frontal limbs as well as the two hind limbs and the limbs on the same side are in anti-phase.

Regarding the phase lag between body oscillators, since it is desired that the body generates standing S-shaped waves, the phase lag between the body oscillators is zero, except for the phase lag between oscillators 5 and 6, from figure 4.2, which has to be π since these represent the node of the standing wave.

The coupling between limbs is done by adding a coupling term, $\sum_{j=1}^{4} k_{ij} y_j$, to the original equations of the Hopf oscilator (2.2) in the $\dot{y}$ variable, having as in [3]:

$$\dot{y}_i = \beta(\mu - {r_i}^2) y_i + \omega_i x_i + \sum_{j=1}^{4} k_{ij} y_j \tag{4.17}$$

with the sign k_{ij} parameters defined by the coupling matrix:

$$K = \begin{bmatrix} 0 & -1 & -1 & 1 \\ -1 & 0 & 1 & -1 \\ -1 & 1 & 0 & -1 \\ 1 & -1 & -1 & 0 \end{bmatrix} \tag{4.18}$$

The values in this matrix are such that the phase relations that were previously chosen for the 4 limbs apply ([3]). Recalling equation 2.11, the coupling between limb and body oscillators as well as between neighbour body oscillators is done by using the following equation [18]:

$$\overline{\omega} = \omega_{spine} + \sum_{j=1}^{N} \frac{w_{ij}}{r_i}[(x_i y_j - x_j y_i)cos\Phi_{ij} - (x_i x_j + y_i y_j)sin\Phi_{ij}] \tag{4.19}$$

where w_{ij} is the coupling weight between oscillators i and j. In order to generate S-shaped waves, Φ_{ij} is zero except for the 5^{th} and 6^{th} oscillators, where $\Phi_{ij} = \pi$. In order to verify the coordination of the S-shaped wave with limb movements that increase the stride length, body oscillators must be in phase with left limbs, so $\Phi_{ij} = 0$ when i is the index of one of the left limbs, and in phase opposition with right limbs thus $\Phi_{ij} = \pi$ when i is the index of a limb on the right side of the body. To summarize the values of the phase differences (according to the indexes defined in figure 4.2):

- $\Phi_{ij} = -\Phi_{ji} = \pi$, when i and j are the indexes of limb oscillators
- $\Phi_{5,6} = -\Phi_{6,5} = \pi$, the node of the standing wave
- $\Phi_{i,j} = 0$, when $i = 9; 11$ (left limbs) and $j = 1, ..., 8$
- $\Phi_{i,j} = \pi$, when $i = 10; 12$ (right limbs) and $j = 1, ..., 8$
- $\Phi_{i,j} = 0$, for the remaining oscillators (body-body coupling)

The Hopf oscillators of the body CPG are then defined by:

$$\dot{x}_i = \alpha(\mu - {r_i}^2)x_i - \overline{\omega_i} y_i \tag{4.20}$$

$$\dot{y}_i = \beta(\mu - {r_i}^2)y_i + \overline{\omega_i} x_i + \sum_{j=1}^{4} k_{ij} y_j \tag{4.21}$$

The frequency ω_{spine} is the intrinsec frequency of body oscillators. This is the rate at which each oscillator completes a full rotation. In open-loop this is a given parameter, so it is made $\omega_{spine} = 2\pi f_{global}$. In the case of the closed-loop controller, the global frequency of motion will eventually be a consequence of the environment, so for this reason, for the body oscillators of the closed-loop controller it is set to:

$$\omega_{spine} = 2\frac{\omega_{swing}\omega_{stance}}{\omega_{swing} + \omega_{stance}}; \tag{4.22}$$

4.5 Network parameters

In order to fully describe the used network it is necessary to define the remaining parameters which have been left unknown until now.

α and β parameters

These two parameters control the rate of convergence of the oscillator to the limit cycle ([3]). In this implementation of the CPG network these parameters are defined as $\alpha = 5$ and $\beta = 10$. These values were the values used in the first tests of the controller when fast transitions were still attempted to be implemented in the salamander robot and so they remained. It should be clarified, though, that for the current model of the oscillator the values of these two parameters have very little relevance since the oscillator amplitude is always constant and equal to $\sqrt{\mu}$. Since the parameter that modulates swing and stance phases is ω, the oscillator is always in its limit cycle. Despite this fact, when the simulations start, all the x_i and y_i are initialized to random values between zero and one, so there is at some point a period of convergence to the limit cycle and α and β play a role on how this convergence happens. However, since the oscillators are given a few thousands of iterations to converge before starting issuing control commands, in the end α and β have no effect on the control.

μ parameter

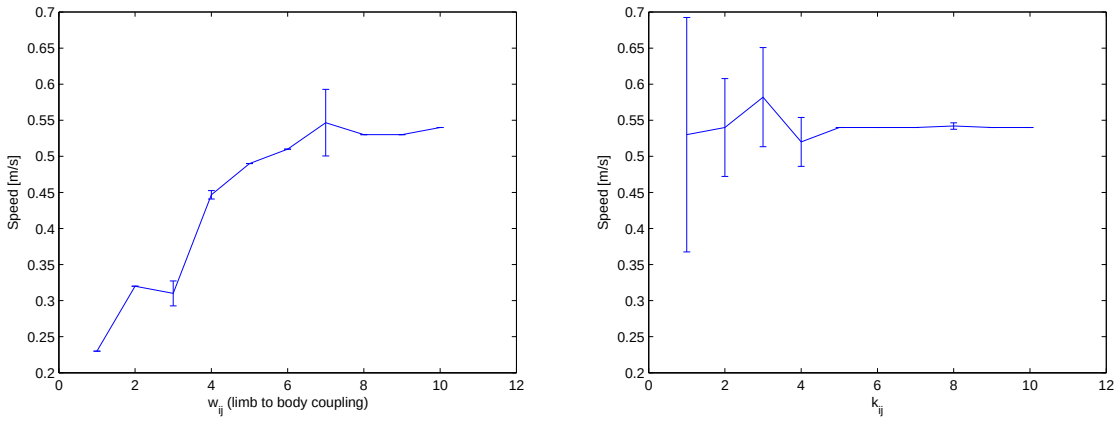
The parameter μ is set to $\mu = 1.0$ since the control policy of each limb is the phase of the corresponding oscillator and, so, independent of its amplitude, and also because the amplitude of body oscillations is controlled externally by the parameter A (eq. 4.12).

k_{ij}, w_{ij} parameters

k_{ij} represents the coupling weights between limbs while w_{ij} may be either the coupling weights between limb and body oscillators or simply between body oscillators (i is always the index of a body oscillator, while j may be an index of a body or a limb oscillator, since limbs project to body). The weights were chosen so that the gaits generated by the CPG network would have high speeds and repeatibility. This way, they were set to:

- $k_{ij} = 10$ - limb to limb coupling
- $w_{ij} = 1$ - when j is the index of a body oscillator
- $w_{ij} = 10$ - when j is the index of a limb oscillator (limb to body coupling)

In order to get a better perception on how these weights affect the speed, tests were made in which only one of the coupling weights was decreased, while the others remained constant. In figures 4.8 to 4.9 it is represented the mean and standard deviation of 5 measurements of speed at each value of the weights for a gait with $f_{swing} = 1.3$Hz $f_{stance} = 1.2$Hz and $\phi_{swingonset} = -\frac{\pi}{4}$rad, $\phi_{stanceonset} = 5\frac{\pi}{4}$rad.



(a) Effect of changing limb-to-body coupling (b) Effect of changing interlimb coupling weights(k_{ij})

Figure 4.8: Effect on the speed of changing the coupling weights between limbs and limbs and body.

Figure 4.8b shows that decreasing k_{ij} highly increases the uncertainty of the speed resulting from the gait. Since this is the interlimb coupling and swing and stance phases have different frequencies, if the coupling between limbs is too low, they may not phase-lock with the appropriate phase differences.

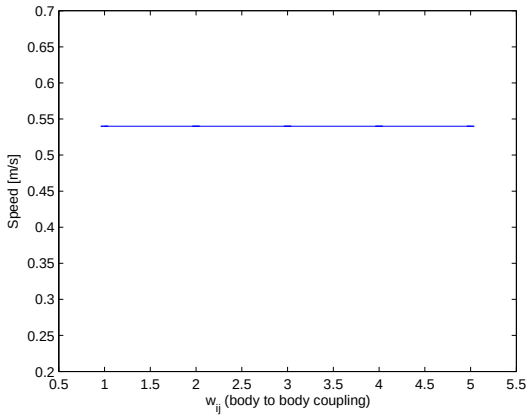


Figure 4.9: Effect on speed of changing body oscillator intercoupling weights.

Decreasing w_{ij} (when j is the index of a limb oscillator) reduces the speed of the gait as is shown in figure 4.8a. This is the effect of the limb-to-body coupling that, apparently, when too low, results in low speeds possibly because, again, the limbs and body do not phase lock at a phase difference that maximizes stride length.

In figure 4.9 it can be seen that changing the weights of the coupling between body oscillators does not affect the speed.

The main reason why so strict values were imposed to these weights were the facts that, first of all, the speed of the open-loop controller needed to be the most deterministic possible for the same fixed parameters since it was going to be optimized (so the choice $k_{ij} = 10$), and second, also because of the optimization, it was intended to get the controller to work at the highest speeds possible (then the choice for the body-to-limb weights as $w_{ij} = 10$).

4.6 Validation

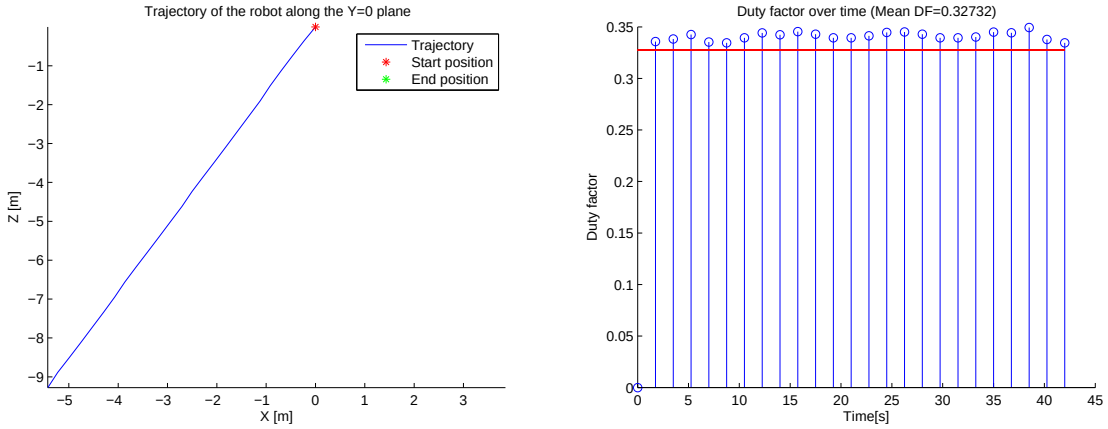
Open-loop controller

A 30s simulation of the salamander walking gait in flat terrain was run with the open-loop controller using the CPG architecture and oscillator described in the previous sections. The used parameters were:

- $f_{swing} = 1.0Hz$
- $f_{stance} = 0.5Hz$
- $\phi_{stanceonset} = 5\frac{\pi}{4}$

- $\phi_{swingonset} = -\frac{\pi}{4}$
- $A = 0.5rad$ (amplitude of body oscillations)

The script *matlab_logplotter.m* in the folder *Webots_logs* of the documentation does the gait analysis after a simulation is run in Webots. The gait generated by the open-loop controller moved the robot along the trajectory in figure 4.10 achieving an average speed of 0.36 m/s and a mean duty factor of 0.33 (figure 4.10). The phase relations between body and limbs also respect the desired properties (figures 4.11 and 4.12)



(a) Robot trajectory

(b) Robot's duty factor over time

Figure 4.10: Trajectory and duty factor of the robot in open-loop.

Closed-loop controller

With limb stopping

To show the capacity of this network model to implement the closed-loop controller, a simulation was run with this type of controller with limb stopping. The parameters of this gait were:

- $f_{swing} = 1.0Hz$
- $f_{stance} = 0.5Hz$
- $A = 0.5rad$ (amplitude of body oscillations)

The controller generated a gait which allowed the robot to move along the trajectory shown in figure 4.13 at an average speed of 0.39m/s, with a mean duty factor of 0.34 (figure 4.13). The

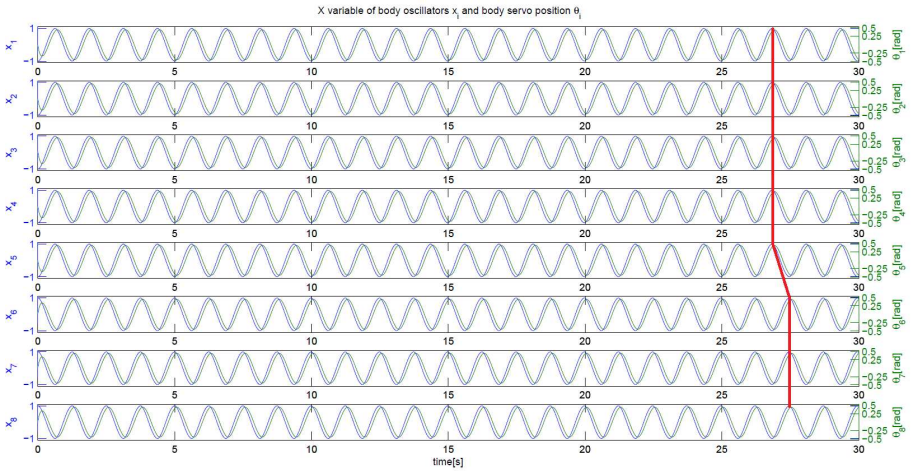


Figure 4.11: Body oscillators' phase during locomotion with open-loop controller. On the left side (blue) the x_i variables of the body oscillators. On the right (green) the actual joint position. The red line shows the first 5 oscillators in phase between each others and in phase opposition with the last 3.

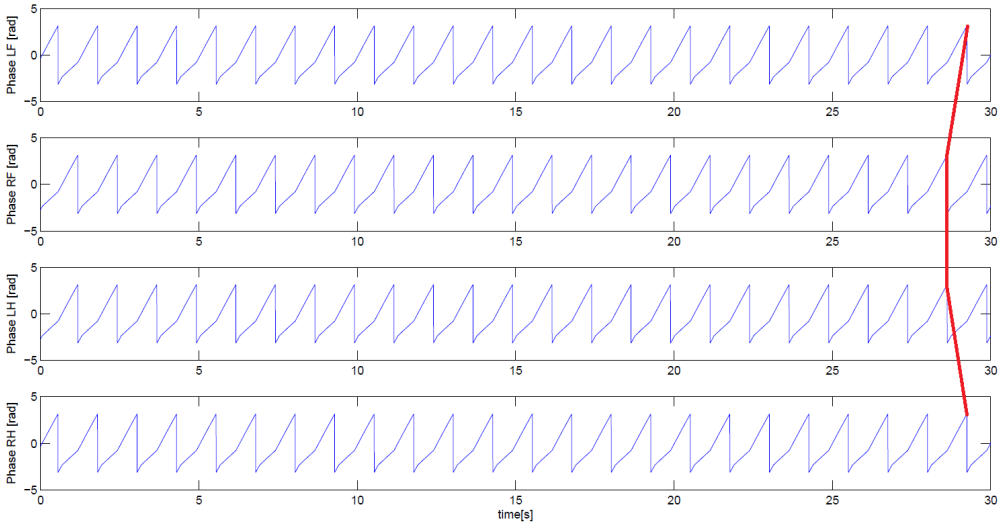
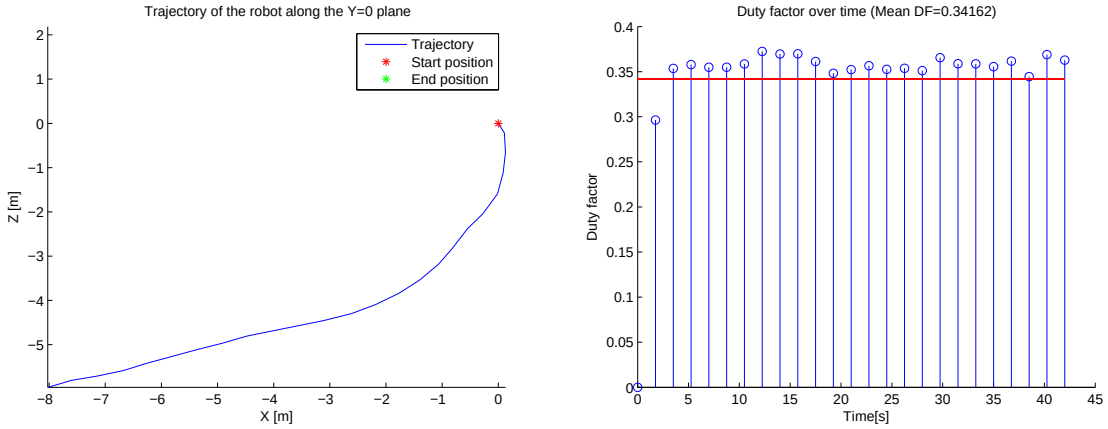


Figure 4.12: Limb oscillators' phase during locomotion with open-loop controller.

x variable of body oscillators is represented in figure 4.14 while the phase of limb oscillators is plotted in 4.15. It is possible to verify that the controller maintains the desired phase relations between limb and body oscillators.



(a) Robot trajectory

(b) Robot's duty factor over time

Figure 4.13: Trajectory and duty factor of the robot in closed-loop.

Without limb stopping

Deactivating the limb stopping, led to a slower gait (0.31m/s), to undesired phase relations between the limb oscillators (figure 4.16) and a very tortuos trajectory (figure 4.17). These facts validate the choice for a controller that performs limb stopping since it allows faster, more stable gaits.

4.7 Summary

To sum up, the Hopf oscillators that compose the limb CPG are governed by the following set of equations:

$$\dot{x}_i = \alpha(\mu - {r_i}^2)x_i - \omega_i y_i \tag{4.23}$$

$$\dot{y}_i = \beta(\mu - {r_i}^2)y_i + \omega_i x_i + \sum_{j=1}^{4} k_{ij} y_j \tag{4.24}$$

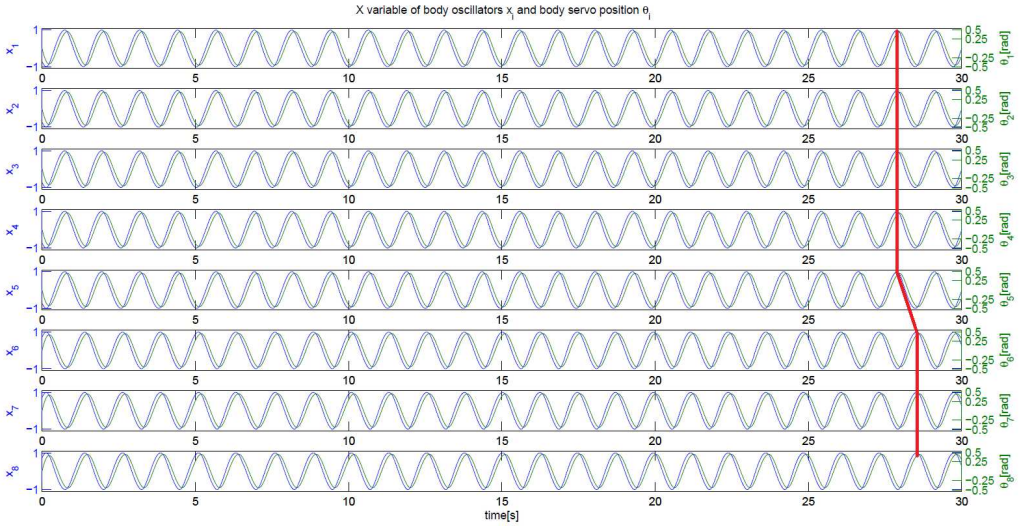


Figure 4.14: Body oscillators' phase during locomotion with closed-loop controller performing limb stopping.

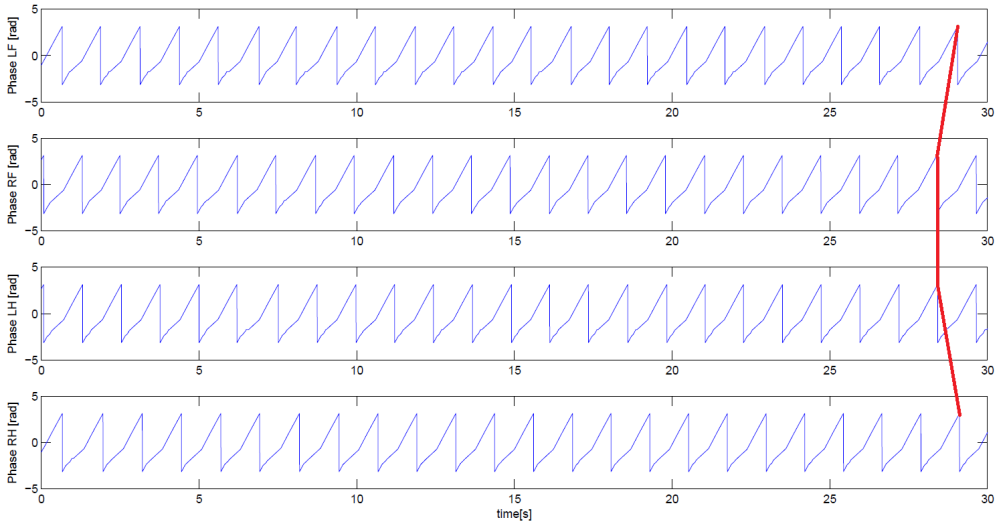


Figure 4.15: Limb oscillators' phase during locomotion with closed-loop controller performing limb stopping.

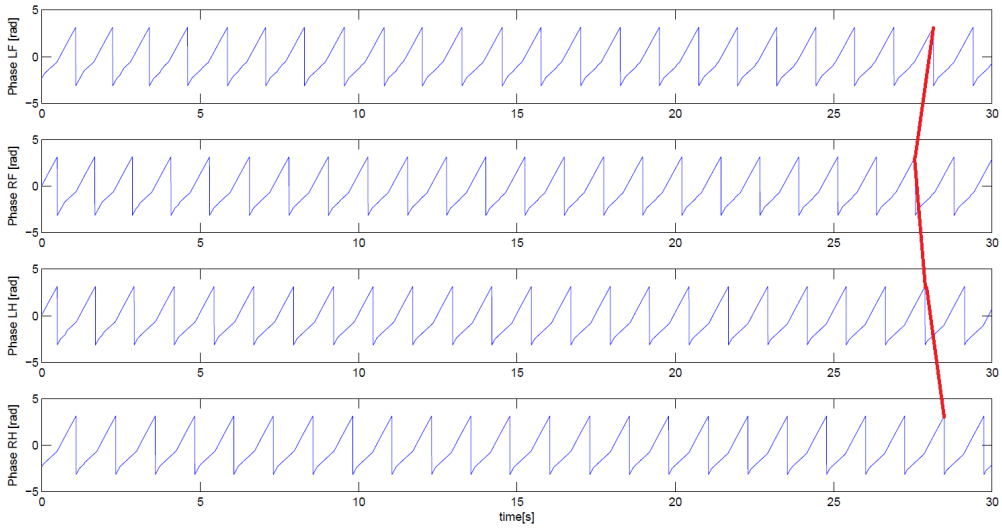


Figure 4.16: Limb oscillators' phase during locomotion with closed-loop controller without limb stopping. The red line shows undesired phase relations.

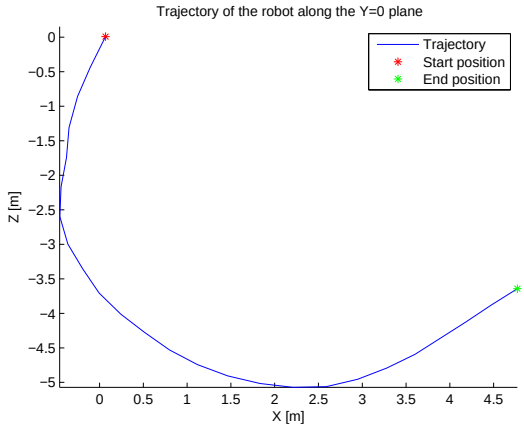


Figure 4.17: Trajectory during locomotion with closed-loop controller without limb stopping.

$$\omega_i = \begin{cases} 0, & \text{if} \quad \theta_i = -90^\circ \text{ and limb is not on the ground,} \\ \omega_{stance}, & \text{if} \quad \text{limb } i \text{ is on the ground,} \\ \omega_{swing}, & \text{if} \quad \text{limb } i \text{ is off the ground,} \end{cases} \tag{4.25}$$

while the oscillators from the body CPG are governed by the following:

$$\dot{x}_i = \alpha(\mu - {r_i}^2)x_i - \overline{\omega_i}y_i \tag{4.26}$$

$$\dot{y}_i = \beta(\mu - {r_i}^2)y_i + \overline{\omega_i}x_i + \sum_{j=1}^{4} k_{ij}y_j \tag{4.27}$$

$$\overline{\omega_i} = \omega_{spine} + \sum_{j=1}^{N} \frac{w_{ij}}{r_i}[(x_i y_j - x_j y_i)cos\Phi_{ij} - (x_i x_j + y_i y_j)sin\Phi_{ij}] \tag{4.28}$$

Chapter 5

Optimization of the open-loop controller

5.1 Optimization setup

As mentioned before, the speed of locomotion in open-loop is greatly influenced by the predefined angles to start the swing and stance phases. Contrary to the closed-loop controller, that uses sensory information to know the current phase of locomotion, the open-loop controller relies on the choice of $\phi_{swingonset}$ and $\phi_{stanceonset}$, which greatly influence the resulting speed of the gait.
In the performance assessment between open- and closed-loop, it is essential to compare the closed-loop against the best possible open-loop controller. For this reason, the open-loop controller is optimized so that for each pair of frequencies (f_{swing}, f_{stance}), it knows which is the pair of values ($\phi_{swingonset}$,$\phi_{stanceonset}$) that maximizes speed. The optimization is done for a single, standard, 'canonic' case - the flat terrain - as optimizing in all the different cases is way too time consuming (1 optimization takes around one week) and goes against the purpose of having a closed-loop controller - a controller that can easily adapt to new environments without requiring extensive tayloring. The parameters that result from this optimization are the ones that will be used in further analysis on different terrains.

According to literature on the gaits of bipedal and quadrupedal animals ([19]), the distinction that is generally made is that walking gaits have duty factors greater than 0.5 while running gaits have duty factors smaller than 0.5. Recalling the definition of duty factor (2.16), it is

the percentage of the stride duration that a limb spends on the ground, thus it is given by:

$$DF = \frac{t_{stance}}{t_{swing} + t_{stance}} \tag{5.1}$$

So it can be infered that faster gaits are usually the ones whose limbs spend shorter periods of time on the ground.

Considering the architecture of the open-loop controller as it was defined in chapter 4, there are four parameters that need to be defined in order to generate a walking gait:

- f_{swing} - linear frequency of the swing phase [Hz]
- f_{stance} - linear frequency of the stance phase [Hz]
- $\phi_{stanceonset}$ - angle to onset the stance phase [rad]
- $\phi_{swingonset}$ - angle to onset the swing phase [rad]

The last two parameters are important in the sense that they control the time spent by the limbs on the ground by regulating the duration of the stance phase.

The optimization of the open-loop controller consisted in running systematic tests, for every pair of frequencies (f_{swing}, f_{stance}), trying all the possible combinations of $\phi_{stanceonset} \in [0; 3 \cdot \frac{\pi}{2}]$ and $\phi_{swingonset} \in [-\frac{\pi}{2}; 0]$ in order to determine which lead to highest speeds (figure 4.7 shows these angles in the oscillator's phase space). The ranges and granularities used in the systematic tests were:

- $f_{swing} \in [0.1; 1.5], \qquad \Delta f_{swing} = 0.1;$
- $f_{stance} \in [0.1; 1.5], \qquad \Delta f_{stance} = 0.1;$
- $\phi_{stanceonset} \in [\pi; \frac{3\pi}{2}], \qquad \Delta \phi_{stanceonset} = \frac{\pi}{20};$
- $\phi_{swingonset} \in [-\frac{\pi}{2}; 0], \qquad \Delta \phi_{swingonset} = \frac{\pi}{20};$

The optimization was performed in Webots environment with the robot and flat terrain models that are defined in chapter 3.

Fixed A parameter

The maximum amplitude of body oscillations, A, was fixed and set to 0.5 radians. Since sensory feedback will be used to modulate the duration of the locomotion phases, and this parameter (unlike f_{swing}, f_{stance}, $\phi_{stanceonset}$ and $\phi_{swingonset}$) is not directly related to it, it was fixed in order to reduce the number of parameters to be tested and thus the complexity of the optimization. Conclusive qualitative results regarding the utilization of this type of sensory feedback are less likely to depend on the maximum amplitude of the body movements than on any of the other four mentioned parameters, since it is not directly affected by whether limbs are on or off the ground. The value of 0.5 radians was chosen because it is high enough to show why body bending is used in the real salamander (increases stride length and consequently speed) and low enough to not bend the body by too much.

5.2 Results of optimization

The optimization resulted in the speed profile that is represented in figure 5.1 where speed is plotted as function of both frequencies of swing and stance. This is the speed obtained for each pair of frequencies when the onset angles are the ones in Figures 5.2a and 5.2b. Literature, such as [3], suggests that the speed of locomotion in quadruped animals is controlled by the duration of the stance phase. The speed profile obtained for this controller seems to go against this widely accepted property of gaits as speed seems to be more dependent on swing frequency than on stance frequency.

The region in the top-right corner of the graph, shows a decrease in speed, that goes against the general tendency of the plot, which is the increase with the frequencies of swing and stance. This happens due to the limitations of the PD controllers of the robot's body joints - since the setpoint ϕ oscillates between $-A$ and A, when frequencies are too high, the controller does not respond fast enough so the body joints do not reach the desired amplitude. As a consequence of the lower amplitude of body oscillations, the stride length is shorter and the speed decreases.

Regarding the ideal angles for each frequency, figure 5.2 shows how the ideal angles vary with the frequencies. The conclusions from these figures are quite intuitive, since the ideal angles are such that the locomotion phase with the highest frequency has the longest duration. In the case of $f_{swing} > f_{stance}$ the duration of the stance is minimal, with swing ending at

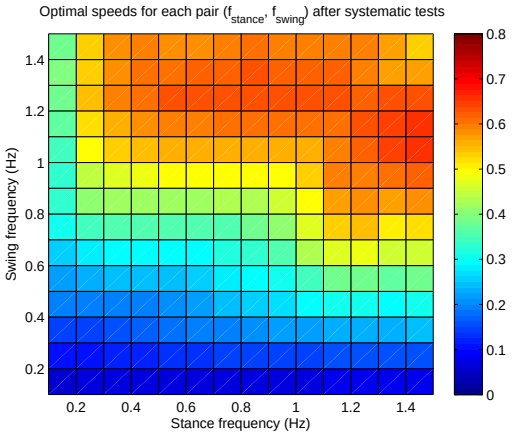
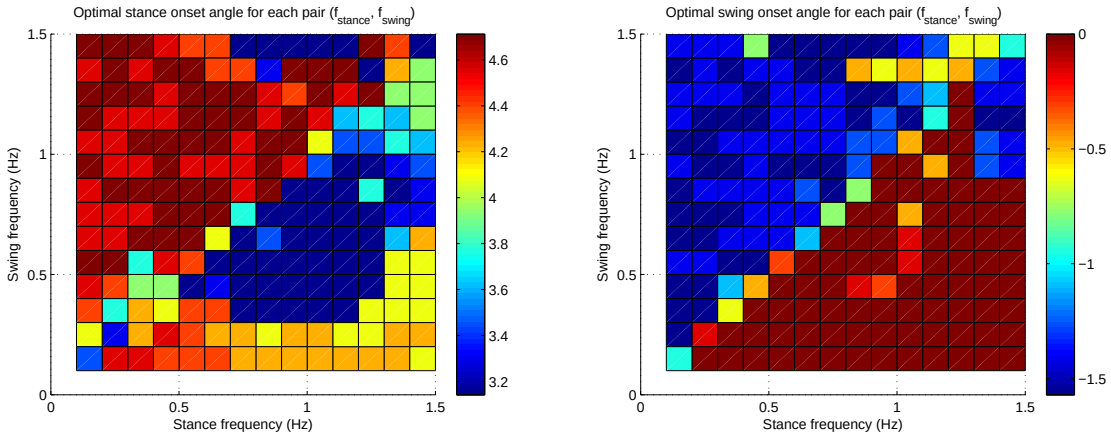


Figure 5.1: Speed (m/s) as function of f_{swing} and f_{stance}.

$\phi_{stanceonset} \approx 3\frac{\pi}{2}$ and restarting again a little after $\phi_{swingonset} \approx -\frac{\pi}{2}$. The same property is verified when $f_{stance} > f_{swing}$ since stance starts very early around 180° while swing starts very late, around 0°. These facts are also supported by figure 5.3a that represent the swing cycle as a percentage of the whole stride duration showing that swing phase occupies a very high percentage of the stride when $f_{swing} > f_{stance}$ while this value is reduced to 50% when $f_{stance} > f_{swing}$.

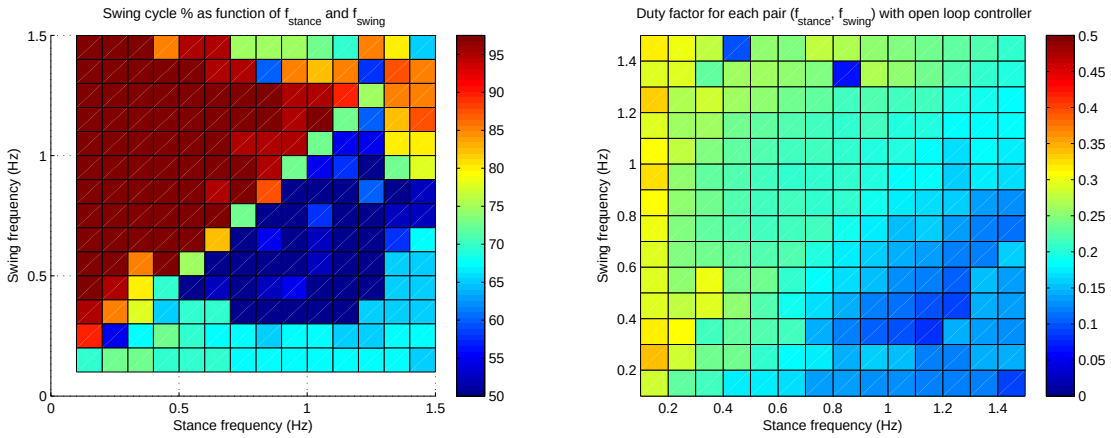


(a) Ideal stance onset angles (rad).

(b) Ideal swing onset angles (rad).

Figure 5.2: Ideal phase onset angles for each pair f_{swing}, f_{stance}.

Regarding the duty factors (figure 5.3b), it is important to notice that the measurements obtained show generally low duty factors, indicating that this is in fact a fast gait. Nevertheless, speed and duty factor graphs are contradictory in respect to the regions where their maxima and minima are located. Since high speeds are related to low duty factors, the lowest duty



(a) Swing cycle percentage.

(b) Duty factors as functions of frequency.

Figure 5.3: Other locomotion parameters.

factors should be located in the top-right corner of the figure.

5.3 Aspects about the optimization

This optimization of the open-loop controller was done so that the tests in different terrains could be done using the best controller that was possible to obtain. It is then important to remind that the fact that the closed-loop controller might outperform the optimized open-loop controller in some terrains does not mean that the open-loop controller could not be improved or optimized again for that same situation. However, one of the advantages of using a closed-loop controller is precisely avoiding constant optimization in open-loop.

Chapter 6

Controller performance

This chapter will focus on the performance analysis of the open-loop controller and the closed-loop controller that uses touch sensors on the limbs to provide sensory information to is CPG on the current phase of locomotion. The analysis will focus on average speed of locomotion and gait stability.

The gait stability is measured by the tortuosity of the trajectory, i.e., by how much it deviates from a straight line. The tortuosity, τ, is given by:

$$\tau = \frac{L}{C} \tag{6.1}$$

Where L is the distance travelled by the robot and C is the distance between initial and final points of the trajectory.

While operating in open-loop, every controller parameter is previously set. The main difference between the open-loop and the closed-loop controller is that the latter does not need to be programmed with the positions to start swing and stance since that information is obtained through sensory feedback. Due to this fact, it also happens that since limb oscillators are coupled to body oscillators the sensory feedback will affect both limb and body oscillators' phase space meaning that different configurations for the body, other than the standing S-shaped wave, might appear which may or not be better adapted to new environments.

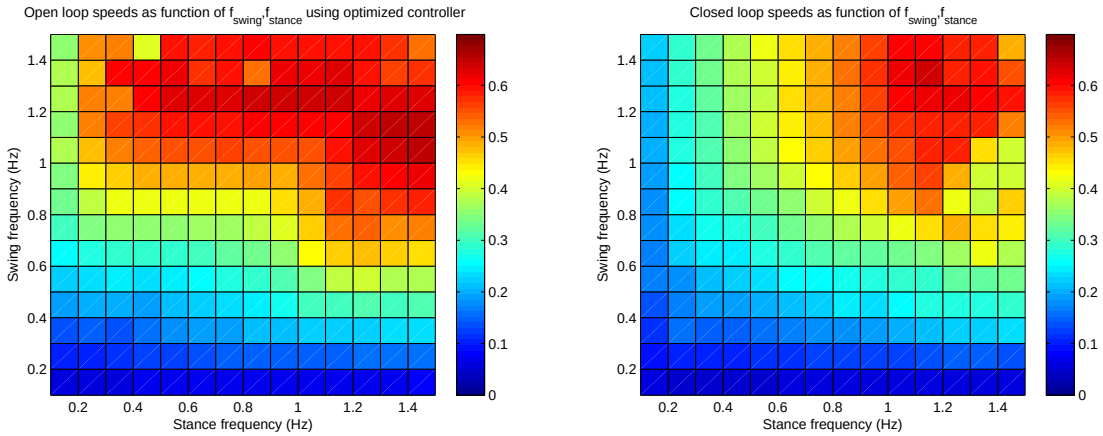
The performance analysis of the controllers is done by applying them in different environments, which include: standard flat terrain, slopes with different inclinations, an uneven, or rough, terrain, terrains with square steps and terrains with different friction coefficients.

6.1 Flat terrain

This was the terrain in which the open-loop controller was optimized and it is the standard flat terrain of the salamander model (*salamander_floor.wbt*), figure 3.3a.

Speed

As the open-loop controller was optimized for speed in this terrain it should not be expected that the closed-loop controller outperforms it. Figures 6.1a and 6.1b show the speed as function of swing and stance frequencies for both controllers.



(a) Open-loop controller

(b) Closed-loop controller

Figure 6.1: Speed [m/s] as function of f_{stance} and f_{swing} for open- and closed-loop in flat terrain.

The optimization that was performed to the open-loop controller led to swing and stance onset positions that favor the phase with highest frequency (chapter 5). For this reason the open loop controller has such high speeds for $f_{swing} \neq f_{stance}$ when compared to the closed-loop controller which is forced to respect the swing and stance phases that are onset by the sensory feedback.

Figure 6.2a shows how the speed relates to global frequency. Recalling the definition of global frequency for the open-loop controller, from chapter 4:

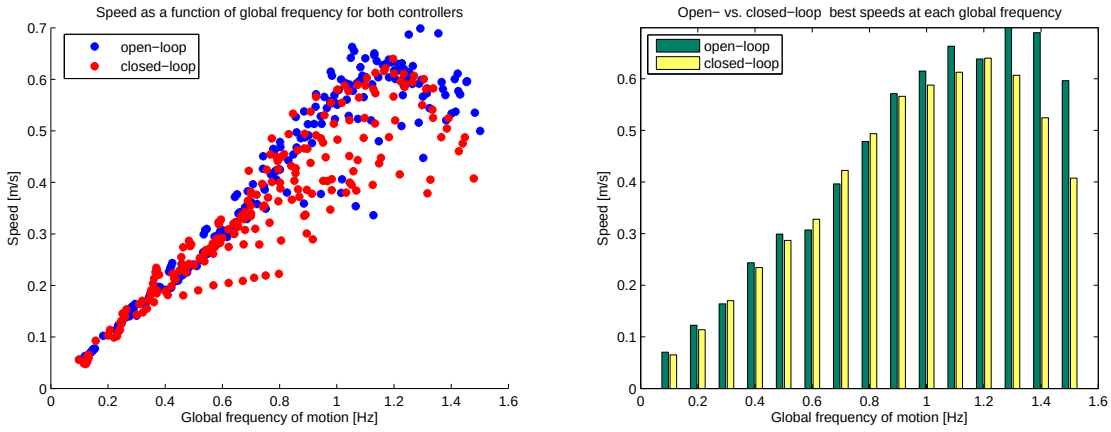
$$f_{global} = \frac{1}{t_{swing} + t_{stance}}$$

while for the closed-loop controller it has to be measured, because there is no closed-form expression since feedback affects the intrinsic frequency of the oscillators. It is thus given by:

$$f_{global} = \frac{1}{T_{average}} = \frac{N_{cycles}}{T_{simulation}} \tag{6.2}$$

with $T_{average} = \frac{T_{simulation}}{N_{cycles}}$, the average period of locomotion, and where $T_{simulation}$ is the total duration of the simulation in Webots and N_{cycles} the number of completed rotations performed by the CPG (measured by the head oscillator).

In order to illustrate the best performance of each controller at each global frequency, the data was discretized in bins of 0.1Hz around specific values of f_{global} and the maximum speed in this interval was taken. The result is shown in figure 6.2b. It is clear from both figures that the trend is maintained, with the optimized open-loop controller outperforming the closed-loop controller for most of the frequencies.



(a) Speed as a function of global frequency for both controllers

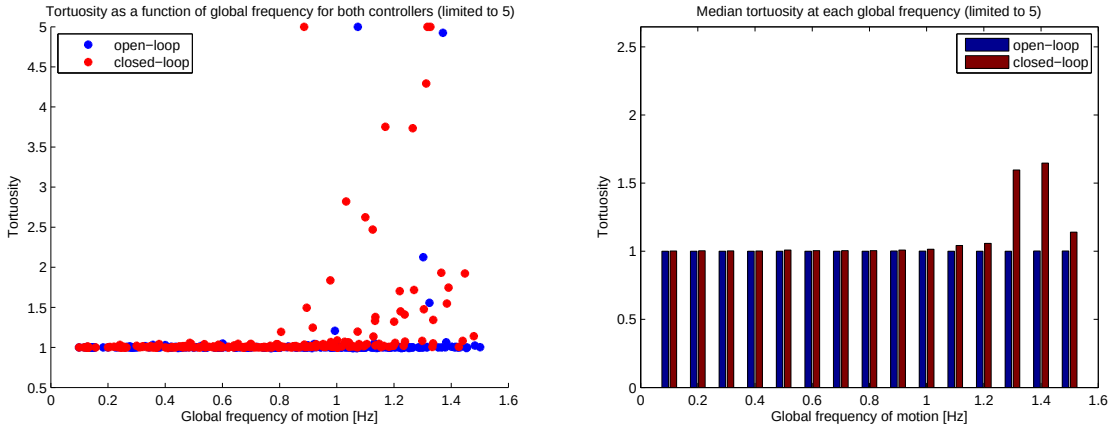
(b) Best speeds in intervals of 0.1 Hz around each value of f_{global}

Figure 6.2: Speed of both controllers against global frequency of motion.

Tortuosity

Concerning the tortuosity, no particular trend was identified, besides the natural tendency of an increase of tortuosity with the frequency of motion. Figures 6.3a and 6.3b show this trend.

Each point in figure 6.3a is the median of the tortuosity of several measurements at that frequency. Similarly to what was done previously to speed, this time, the frequency range was discretized in bins of 0.1Hz and the median tortuosity in every bin for each controller was taken, being the result shown in figure 6.3b.



(a) Tortuosity as a function of global frequency of motion for both controllers.

(b) Median of the tortuosity around each frequency for both controllers.

Figure 6.3: Tortuosity as function of global frequency for open- and closed-loop, with the controllers tested in flat terrain.

6.2 Slope with varying inclination

In order to keep comparing both of the controllers, other environments had to be tested. This time the robot had to go up to the top of a slope (figure 6.4) with two different inclinations, 10° and 20°. This experiment has been done before in other quadruped robots, such as the iCub, Ghostdog or Aibo in [3], but not in the salamander robot.

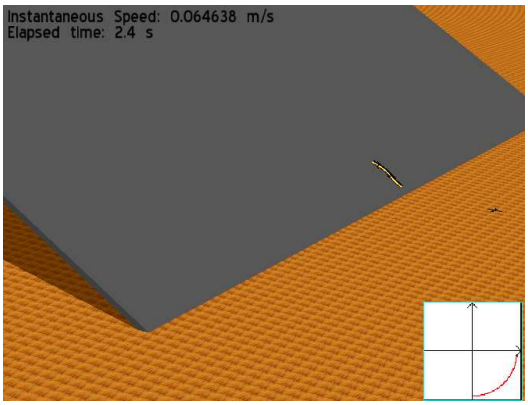


Figure 6.4: World model with slope of 20°.

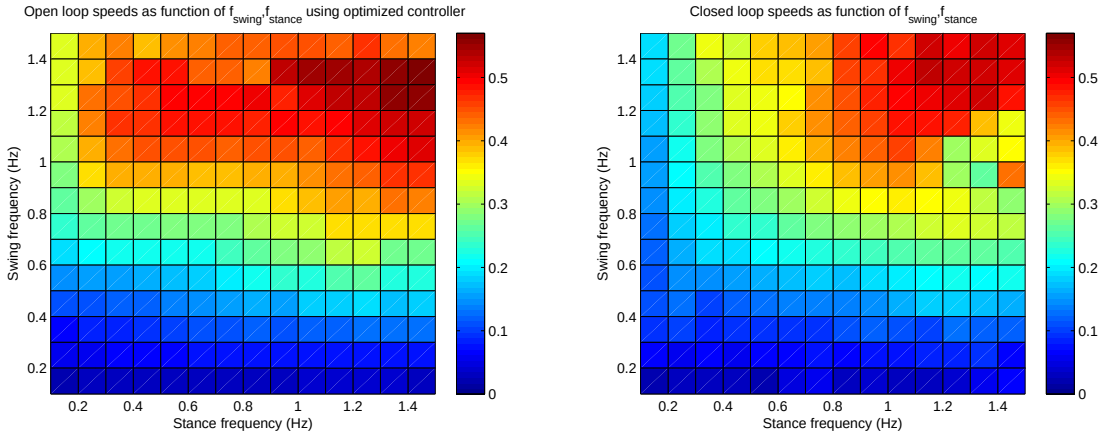
6.2.1 10° slope

Speed

Using a slope with a 10° inclination it was seen that the closed-loop controller performs better than the open-loop controller for low values of the global frequency, while the oppsite happens at high values of f_{global} (Figures 6.6a and 6.6b). The speed profile as function of f_{swing} and f_{stance} is represented on Figures 6.5a and 6.5b. The explanation to this outperformance at low frequencies is addressed in the following section, with a steeper slope.

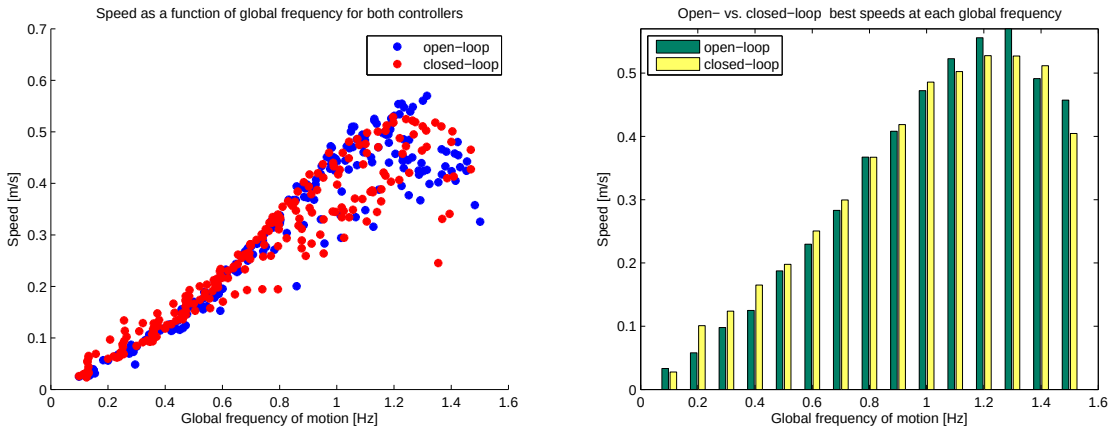
Tortuosity

Tortuosity suffers no major changes between the two controllers, showing only a small tendency for increasing while operating in closed-loop (figures 6.7a and 6.7b).



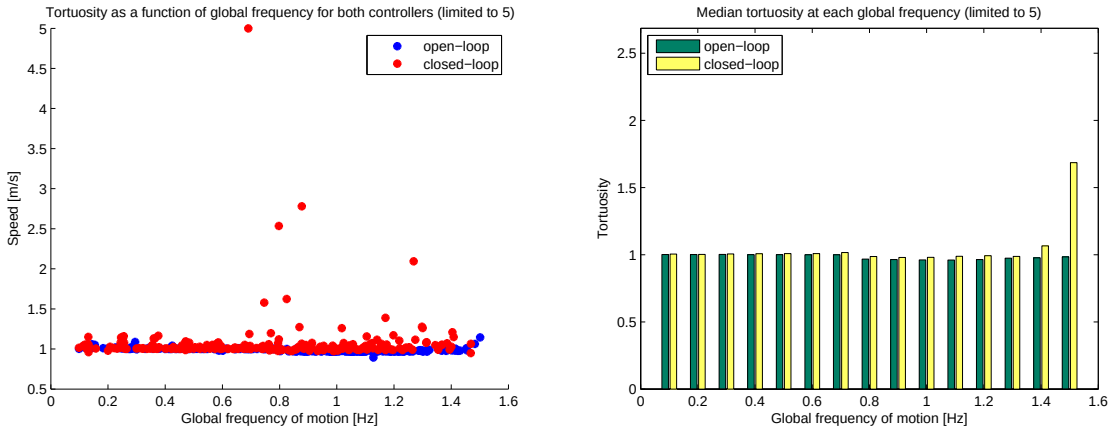
(a) Speed vs. f_{swing}, f_{stance} for open-loop contoller (b) Speed vs. f_{swing}, f_{stance} for closed-loop contoller

Figure 6.5: Speed [m/s] as a function of f_{swing} and f_{stance} when the robot goes up a 10° slope.



(a) Speed as function of the global frequency of motion for both controllers (b) Best speeds around specific values of f_{global}

Figure 6.6: Speed as a function of the global frequency of motion when the robot goes up a 10° slope.

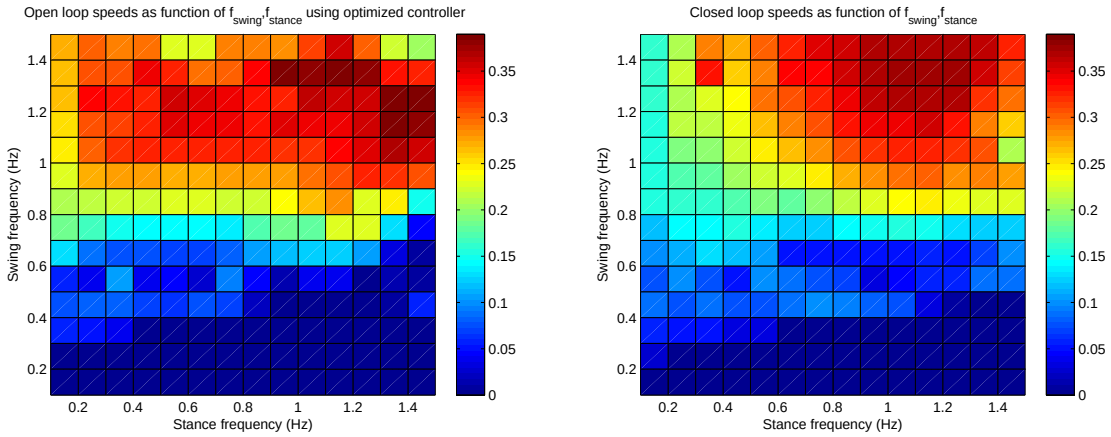


(a) Tortuosity as a function of global frequency of motion (b) Median of the tortuosity around specific frequencies

Figure 6.7: Tortuosity as function of the global frequency of motion when the robot goes up a 10° slope.

6.2.2 20° slope

As this slope is steeper than the previous, at some frequencies, the salamander slides down along the surface as a consequence of the inclination. Figures 6.8a and 6.8b show this fact as dark blue regions at low swing frequency as the Webots supervisor sets the speed to zero when the robot gets off the slope.

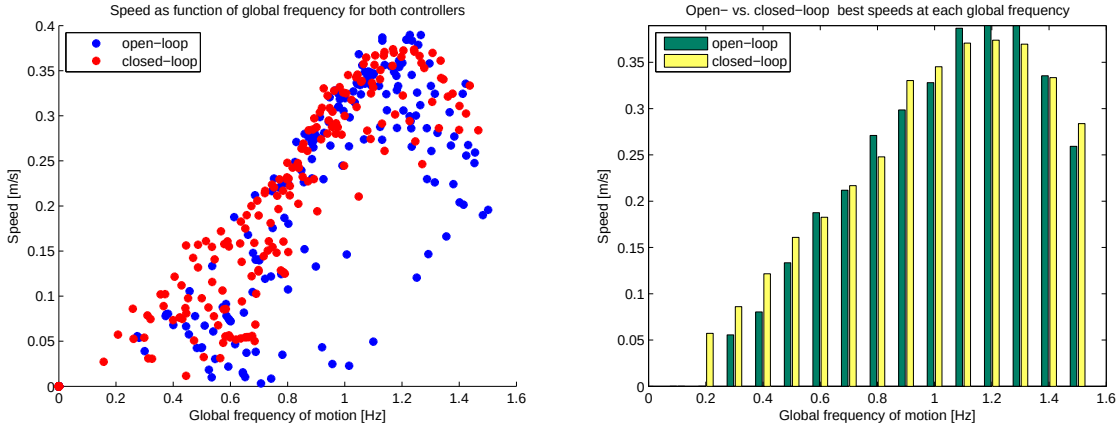


(a) Speed vs. f_{swing},f_{stance} for open-loop controller (b) Speed vs. f_{swing},f_{stance} for closed-loop controller

Figure 6.8: Speed [m/s] as function of f_{swing} and f_{stance} when the robot goes up a 20° slope.

Speed

The sensory feedback controller seems to present some advantages in this case. In figure 6.8b the dark blue region is slightly smaller than that of the open-loop controller, indicating that, at low frequencies, the robot does not slide down the slope as often as in open-loop, and in figure 6.9 the closed-loop controller outperforms the open-loop for many values of the global frequency.



(a) Speed vs. global frequency of motion for both controllers

(b) Best speed in bins of 0.1Hz around specific values of frequency.

Figure 6.9: Speed as function of global frequency of motion when the robot goes up a 20° slope.

In order to understand the reason why closed-loop controller performs better than the open-loop for low frequencies, two movies of Webots simulations were made, for both of the controllers at the same global frequency[1]. The closed-loop controller has a measured frequency of $f_{global} = 0.2$Hz for $f_{swing} = 0.3$Hz and $f_{stance} = 0.1$Hz. On the other hand, $f_{global} = 0.2$Hz with the open-loop controller corresponds to $f_{swing} = 0.3$Hz, $f_{stance} = 0.1$Hz with $\phi_{swing} = -\frac{\pi}{2}$ and $\phi_{stance} = 4.4rad$ from the solutions of the optimization that was performed. The movie *CL_slope_20degrees.avi* shows the gait generated by the closed-loop controller while *OL_slope_20degrees.avi* shows the result of controlling in open-loop. By visual inspection it is obvious that the reason why the open-loop controller leads to poor gaits

[1]These movies, as well as a few others for the next terrains, can be consulted in the DVD that comes with this dissertation

is that the limbs spend too long time in the air, leaving the body as the only contacting point with the ground. Due to the facts that the body's friction coefficent is smaller than the limbs' and the body segments exert no force in the direction of motion, the robot ends up sliding down the slope whenever the limbs are not mechanically supporting it.

The two movies suggest, then, that the advantage of using closed-loop control in this case is the fact that this controller allows long duration stance phases. In order to estimate the duty factor at each global frequency, the data was discretized in bins of 0.1Hz and it was taken the highest value of duty factor within each bin, resulting in the plot in figure 6.10. This figure confirms that gaits generated by the closed-loop controller have higher duty factors than the gaits with open-loop control. Recalling the definition of duty factor as the time percentage of stride during which the limbs are on the ground, this is an indicator of a longer stance phase in the closed-loop.

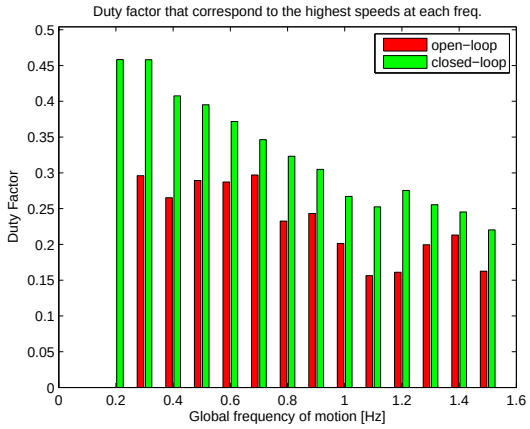
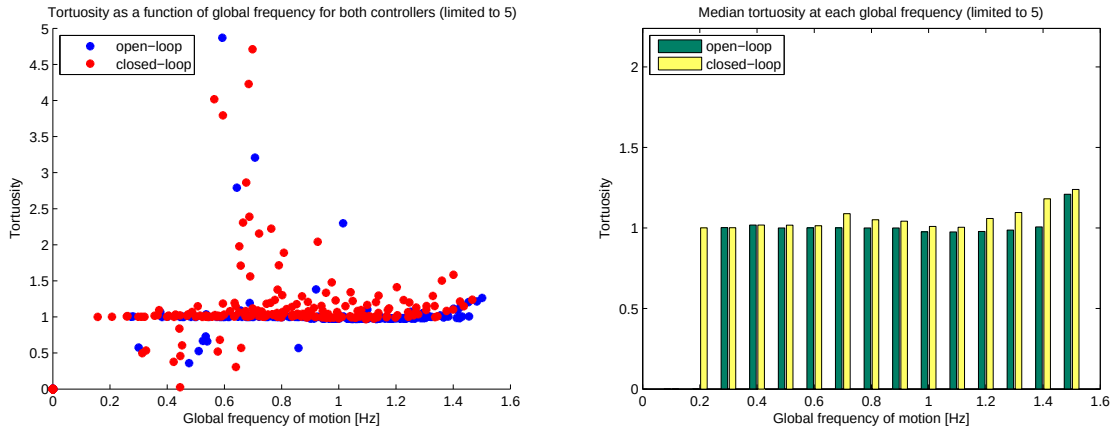


Figure 6.10: Duty factor as function of global frequency at specific frequencies.

The previous data supports that the closed-loop controller is the best controller for this terrain at low frequencies since it correctly identifies the stance phase, allowing longer stance duration and consequently reducing the sliding. The open-loop controller could certainly be optimized for this terrain, similarly to what was done for the flat surface. The optimization would lead to a set of parameters, namely $\phi_{swingonset}$ and $\phi_{stanceonset}$, that maximize the duty factor but, as a drawback, whenever the steepness of the terrain was changed, these parameters would no longer be ideal. Due to its adaptive nature, the closed-loop controller would be capable of automatically identifying the stance phase and thus eventually outperform the open-loop.



(a) Tortuosity as a function of global frequency of motion.

(b) Median of the tortuosity around specific frequencies.

Figure 6.11: Tortuosity as a function of the global frequency of motion when the robot goes up a 20° slope.

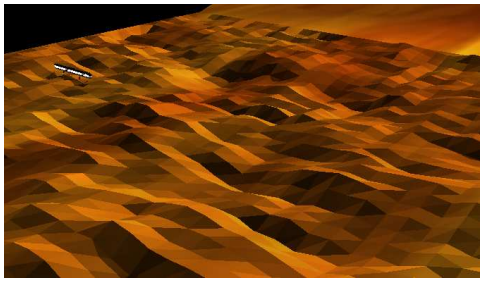
Tortuosity

Regarding tortuosity, since the terrain is, again, not particularly uneven, the tortuosity is generally low, showing higher values for the closed-loop controller (figures 6.11a and 6.11b). This happens mostly because of the adaptive nature of the controller so the gait is not always symmetric leading to some unexpected bending which generates trajectories with more turns.

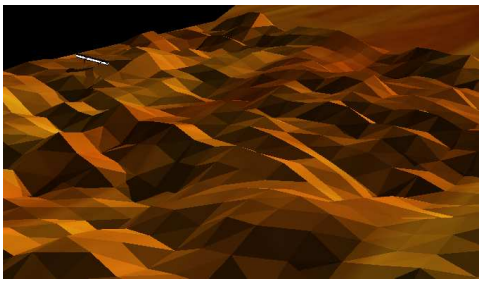
6.3 Rough terrains

Uneven terrains are particularly interesting because they correspond to cases where the changes relatively to the standard terrain are not changes of static parameters like friction or inclination. In this context, rough terrains are interesting tests-beds for controller robustness since they simulate situations in which the robot faces changes in the environment that are unexpected and hard to model, such as sudden changes in inclination or 'valleys'. Kimura et al. [4] as well as Righetti and Ijspeert [3], simulated this type of unpredictability in other quadruped robots.

In figures 6.12a and 6.12b the two Webots world models that were used are represented. Both terrains were generated by the application *terrain_maker.exe* which generates the terrains based on the desired characteristics such as peak resolution (as number of peaks per meter), terrain width and elevation.



(a) Resolution = 5 peaks/m, elevation = 2



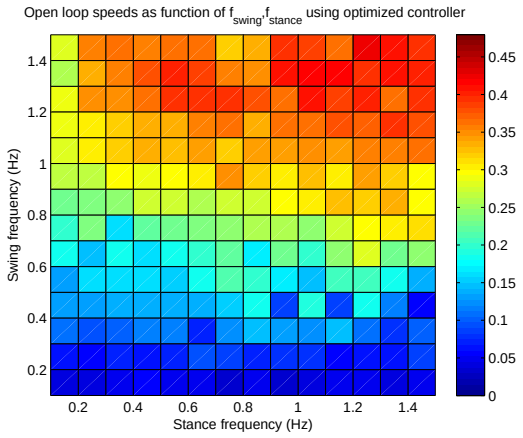
(b) Resolution = 3 peaks/m, elevation = 5

Figure 6.12: Models of worlds with bumps used in Webots to simulate rough terrains.

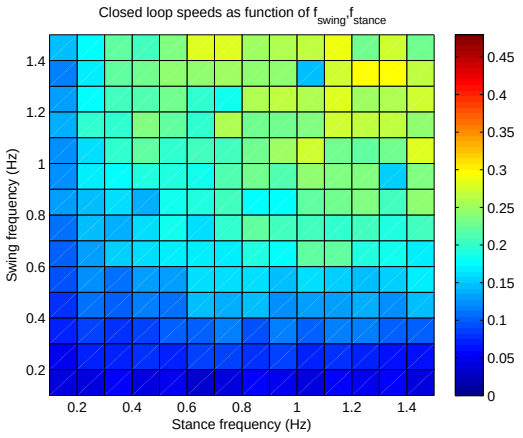
6.3.1 Resolution=5 peaks/m, elevation=2

Speed

The first terrain, in figure 6.12a, has been created so that it does not represent a particularly difficult terrain for locomotion so the elevation was to set to 2. In this terrain, the open-loop controller performs better than the closed-loop in terms of speed, as suggested by figures 6.13 and 6.14.



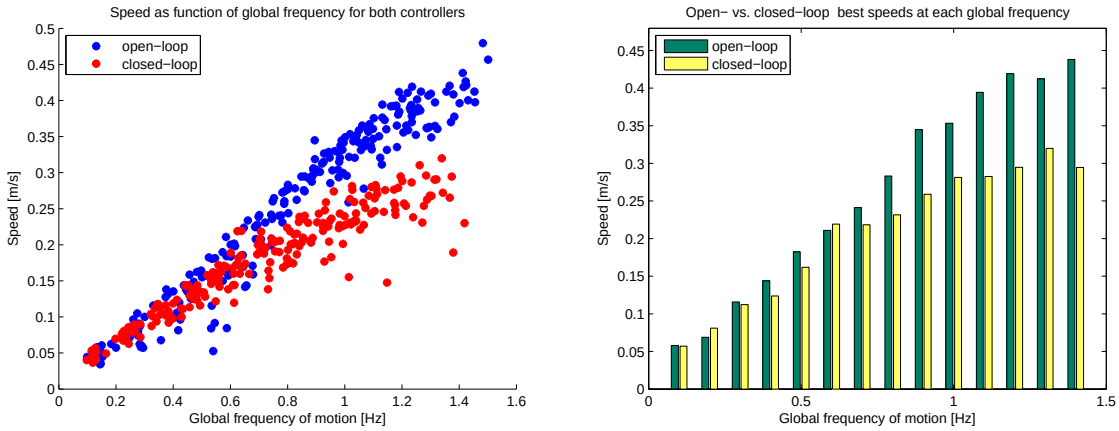
(a) Speed for open-loop controller



(b) Speed for closed-loop controller

Figure 6.13: Speed [m/s] as function of f_{stance} and f_{swing} for open- and closed-loop in a rough terrain with a resolution of 5 peaks/m and an elevation of 2.

In this case, the closed-loop controller, as a controller that allows adaptive walking, clearly



(a) Speed vs. global frequency

(b) Best speed at each global frequency

Figure 6.14: Speed as function of f_{global} for open- and closed-loop for a rough terrain with 5 peaks per meter and elevation = 2.

shows one of its most interesting properties when applied to the salamander robot: the shaping of body oscillators phase space. Figure 6.15 shows the x variable of each body oscillator and corresponding body servomotor position when the robot is controlled in closed-loop at frequencies $f_{stance} = 0.8$ and $f_{swing} = 0.9$. The phase relations between oscillators are kept, while the phase space of each oscillator is modified as a consequence of the sensory input coming from the limbs.

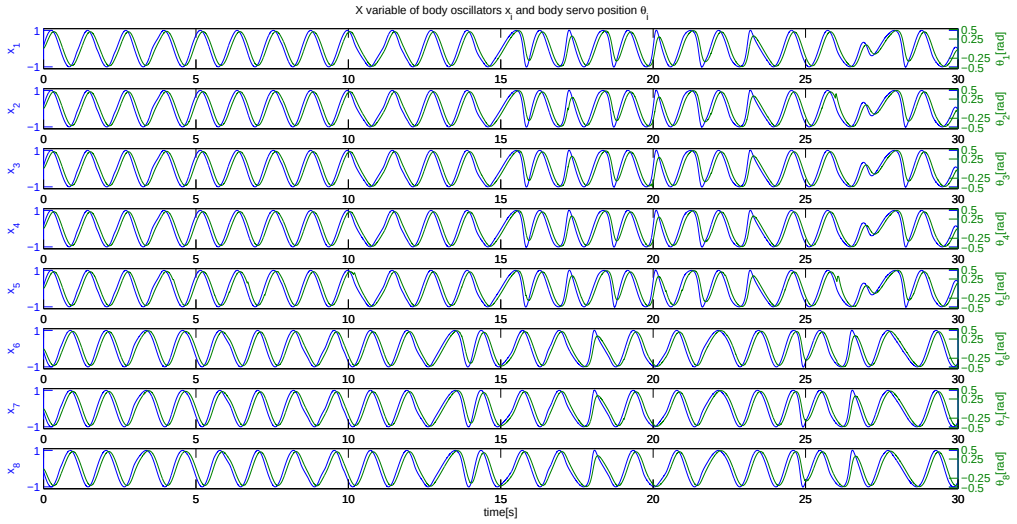
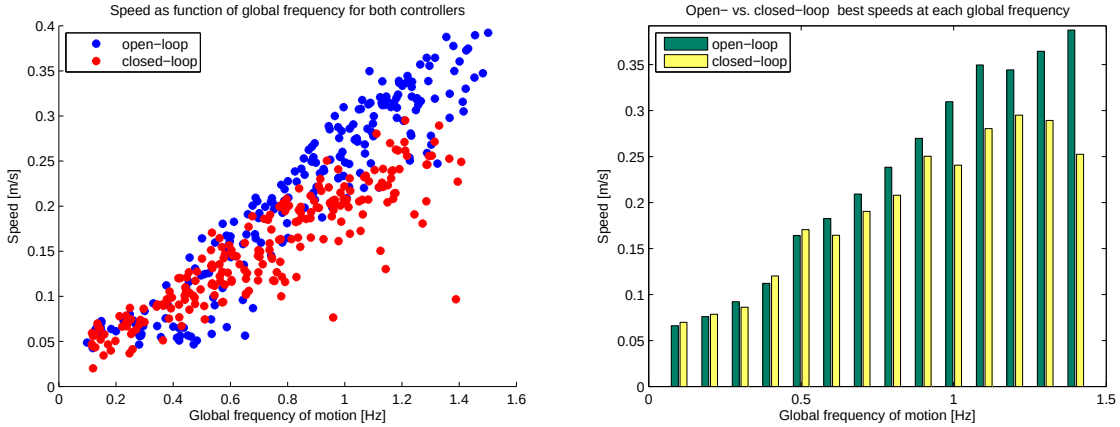


Figure 6.15: x coordinate of each body oscillator and correponding body servo position.

For this specific terrain there seems to exist no advantage in using sensory feedback since the optimized open-loop controller is able to overcome the difficulties of this terrain, while the sensory feedback generates gaits that perform poorly.

6.3.2 Resolution=3 peaks/m, elevation=5

In this terrain (figure 6.12b) peaks have higher elevations making it more prone to the existence of either more valleys and of higher inclinations. For the present case, two different situations were tested, with body servomotor amplitudes, A, equal to 0.5rad and 0.25rad.
This variation of A was done because, with $A = 0.5$ the sides of the body seemed to be coliding with the peaks, so decreasing A, could perhaps improve performace.



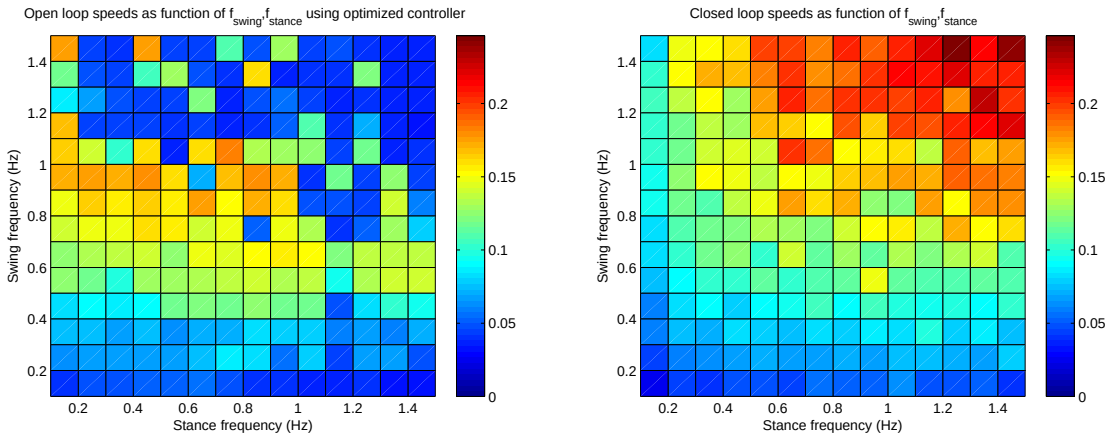
(a) Speed vs. global frequency

(b) Best speed at each global frequency

Figure 6.16: Speed as function of f_{global} for open- and closed-loop in an uneven terrain of higher difficulty (3 peaks/m, elevation=5, A=0.5rad.)

While for the first case, $A = 0.5rad$, the performance seems to be analagous to the previous case, with the optimized open-loop controller outperforming the closed-loop (figure 6.16), for $A = 0.25rad$ the situation is quite the opposite. The exhaustive tests represented in figures 6.17 and 6.18 show that open-loop controller is outperformed, especially at high frequencies, by the controller that uses sensory feedback.

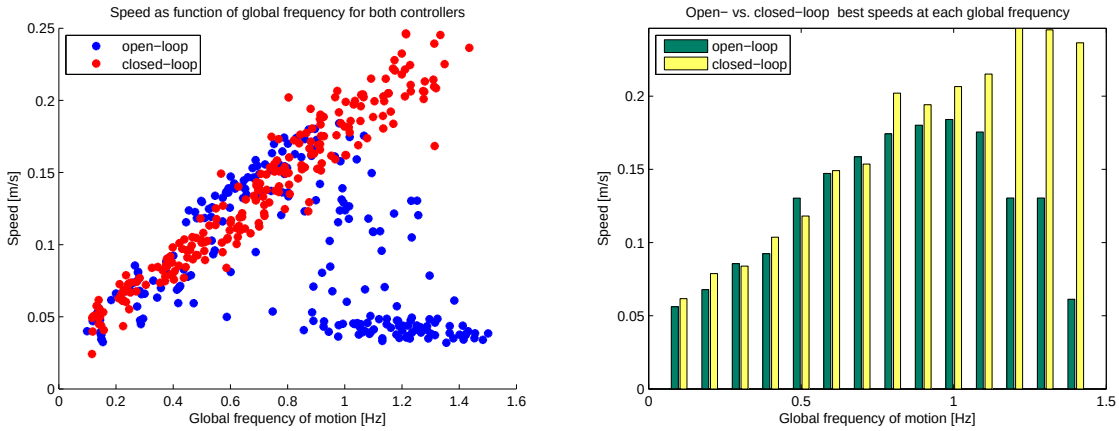
The movies *openloop_bumpy_bodyamp025.avi* and *closedloop_bumpy_bodyamp025.avi* are very



(a) Open-loop controller

(b) Closed-loop controller

Figure 6.17: Speed [m/s] as function of f_{stance} and f_{swing} for open- and closed-loop in hard uneven terrain (3 peaks/m, elevation = 5) with body servo amplitude $A = 0.25$.



(a) Speed vs. global frequency

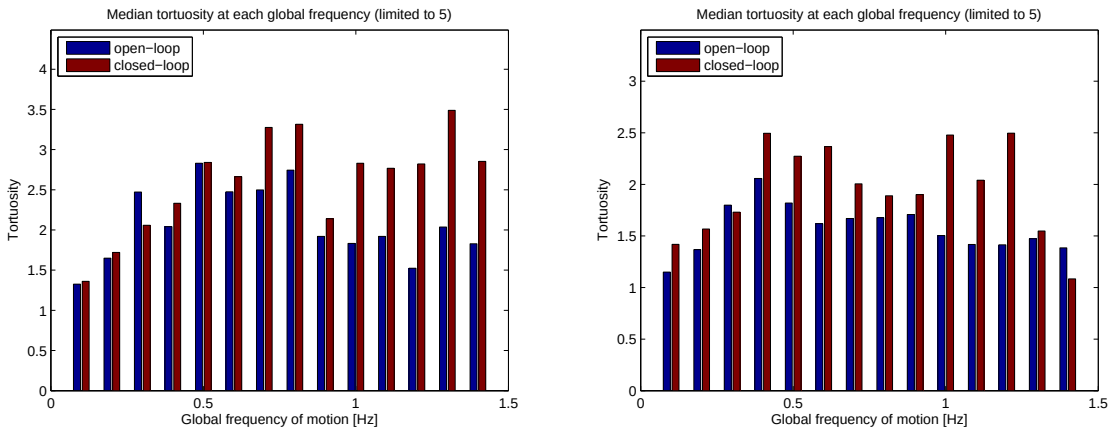
(b) Best speed at each global frequency

Figure 6.18: Speed as function of global frequency in an uneven terrain with 3 peaks/m, elevation=5. Body oscillations amplitude is now A=0.25rad.

clear in explaining the reasons of such differences between the controllers: the optimized open-loop controller often makes the robot get stuck in valleys, while the closed-loop controller manages to take the robot out of these. Two properties of the closed-loop controller are good hypothesis, either alone or together, to explain this: first, as seen in section 6.2, sensory feedback allows better performance while in presence of slopes by making the robot not slide down along them, second, as was seen in 6.3.1, the presence of sensory feedback alters the phase space of body oscillators leading to unexpected body bending which in this case helps the body bouncing and finding alternative paths. For the case in which $A = 0.5rad$, the open-loop controller may not have suffered from this issue since the high body oscillations allowed it to easily bounce outside these 'valleys' and consequently not getting stuck.
The geometry of these terrains has properties that are difficult to find in real environments which have generally smoother surfaces unlike the Webots world in which hills are created with polygons. Nevertheless, the qualitative value of these results obtained in simulation validates the utilization of sensory feedback as a way to improve locomotion performance.

Tortuosity

Figure 6.19 shows that both controllers are particularly unstable in this type of terrain. The closed-loop controller seems to be the less stable which means that the sensory feedback is affecting the body oscillators' phase in such a way that the generated gaits are less efficient.



(a) A=0.5 rad

(b) A=0.25 rad

Figure 6.19: Tortuosity as function of global frequency in an uneven terrain with 3 peaks/m, elevation=5.

6.4 Steps

Another terrain that was used to test the controllers was a terrain with square steps of varying height (figure 6.20). These steps model any ground discontinuity in natural environments, such as holes for example, that may lead to a limb skiping stance phase, thus not touching the ground during the stride. At this point two terrains were tested, varying the maximum step height, which was set to 2.5cm and 5cm.

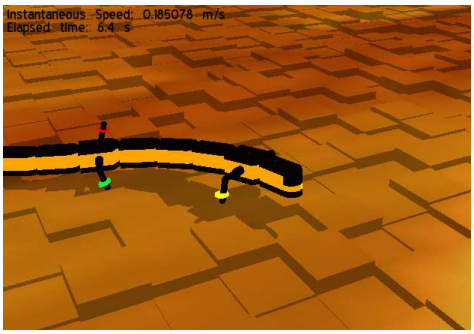


Figure 6.20: Webots terrain with steps.

6.4.1 Maximum step height of 2.5cm

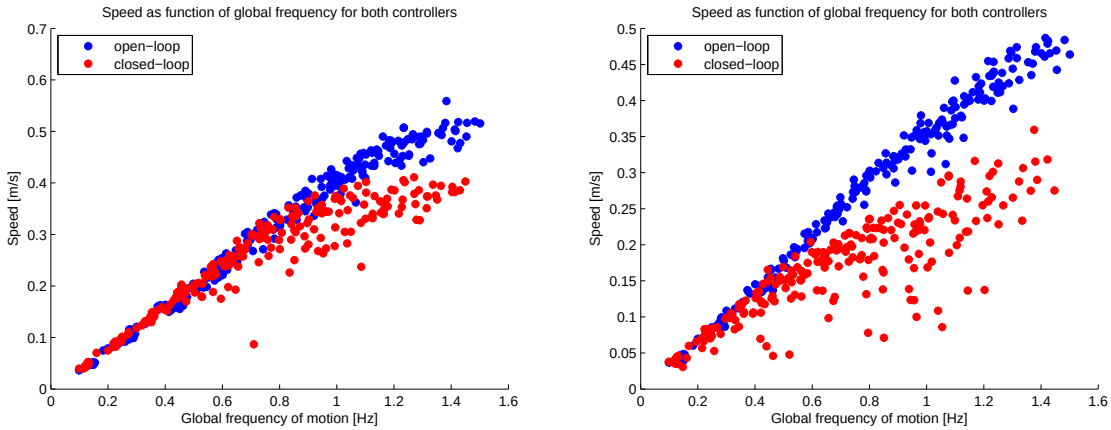
Speed

Figure 6.21a shows that the closed-loop controller represents no advantage in terms of speed for this type of terrains. The video *CL_steps.avi* shows that although the sensory input actually manages to avoid skiping the stance phase by stopping the limbs perpendicularly to the ground, this affects also the body bending which, unlike in section 6.3.2, this time has a negative effect on the speed of the gait.

6.4.2 Maximum step size of 5cm

Speed

Increasing the step height only decreases the performance of the closed-loop controller, as is clear from figure 6.21b where the open-loop controller always reaches equal or higher speeds than the sensory feedback controller. Similarly to what was done before there was as an attempt to somehow make the gait more stable by decreasing the amplitude of body oscil-



(a) Speed vs. global frequency for maximum step height of 2.5cm

(b) Speed vs. global frequency for maximum step height of 5cm

Figure 6.21: Speed as function of frequency of motion in terrains with different step heights.

lations, so making $A = 0.25$rad. Figure 6.22 shows that for this time no positive outcome resulted from the decrease so the optimized open-loop controller keeps generating gaits that reach higher speeds in these conditions.

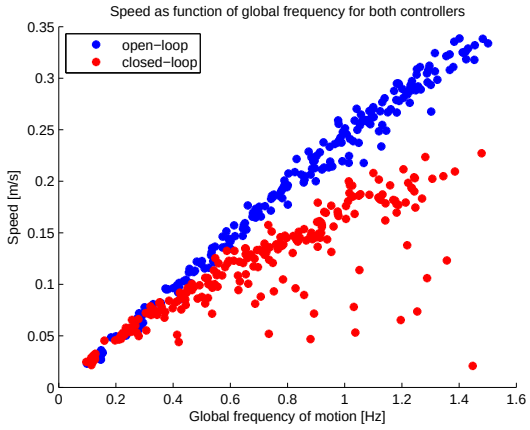


Figure 6.22: Speed in the world with steps at A=0.25rad.

Tortuosity

Figure 6.23 shows that once again there is no particular advantage in using the closed-loop controller to generate more stable gaits as the tortuosity is higher than for the open-loop controller.

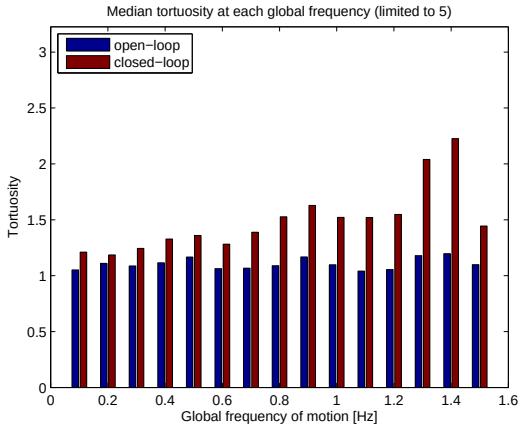


Figure 6.23: Tortuosity in the world with steps at A=0.25rad.

6.5 Friction

Being ground friction one of the physical properties that most influence quadruped locomotion on ground, a few different test terrains were set for this section. The method to model ground friction variations in Webots was by changing the Coulomb Friction coefficient of the robot's parts (see Webots manual in [23]). There are essentially 3 types of nodes that contribute to the friction model of the salamander robot in Webots: the limbs, the limb touch sensors and the body segments. The friction coefficient, μ, acts as parameter that limits the maximum friction force of the ground according to (ODE Manual [21]):

$$F_T \leq \mu F_N \tag{6.3}$$

Eq. 6.3 indicates that, the lower the value of μ, the more slippery the ground is. The default parameters for these friction coefficients are:

- $\mu_{limbs} = 2.0$
- $\mu_{touchsensors} = 1.0$
- $\mu_{bodysegments} = 0.2$

In some cases, the ground friction is not uniformly distributed so limbs may 'see' different frictions than body segments. Also, since the limbs that are used in the robot may be done with different materials, it was found to be interesting to see how the controllers behave when only the friction coefficient of the limbs changes. Following this idea, four different experimental setups were made:

- **Low limb friction** - only μ_{limbs} and $\mu_{touchsensors}$ are changed while $\mu_{bodysegments}$ is kept constant. For the sake of consistency this change is done by the same proportion of 1/6 so $\mu_{limbs} = 0.33$ and $\mu_{touchsensors} = 0.167$.

- **High limb friction** - Analogously to the previous point but now using a factor of 6, resulting in $\mu_{limbs} = 12.0$ and $\mu_{touchsensors} = 6.0$.

- **Low ground friction** - This is equivalent to the situations when both limbs and body segments are subject to the same friction. Following the same pattern of consistency, every parameter is decreased by the same proportion: $\mu_{limbs} = 0.33$, $\mu_{touchsensors} = 0.167$ and $\mu_{bodysegments} = 0.033$

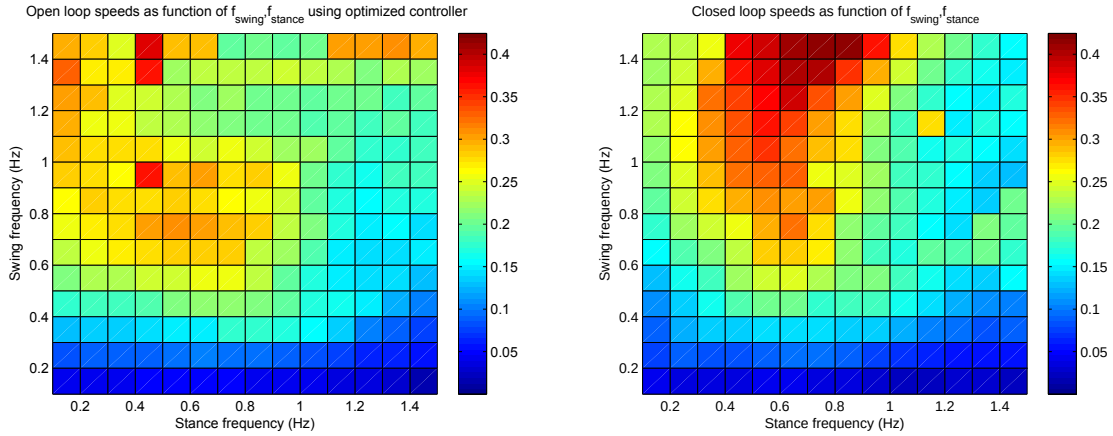
- **High ground friction** - Every coefficient is made 6 times greater: $\mu_{limbs} = 12.0$, $\mu_{touchsensors} = 6.0$ and $\mu_{bodysegments} = 1.2$

All the experiments were made in the standard flat terrain from section 6.1.

6.5.1 Low limb friction

Limb slippery is perhaps one of the most common difficulties experienced by quadruped robots during walking gaits. In the case of the salamander robot, as a robot that swims and walks, this is seen particularly when the robot comes out of the water and starts the walking gait. Also, unlike most of the other quadruped robots (Tekken2 ([4]), iCub, Ghostdog or Aibo ([3]), for example) the salamander robot has its body totally or partially in contact with the ground, being acted by friction forces and so being subject to a new constraint that makes this robot's case even more specific.

Once again, the positions to start swing and stance play a very important role in the generation of a gait that allows forward motion since it is during stance phase, while limbs touch the ground, that the friction dependent forces that push the robot forward are exerted. Figures 6.24 and 6.25 show that both controllers have their maximum speeds at low stance frequencies and also that the closed-loop controller outperforms the open-loop once again because of its ability to detect when to start stance phase. The fact that the stance phase has a longer duration in closed-loop is supported by figure 6.25b where the duty factor range is larger for the closed-loop controller. Also the videos *open_low_limb_friction.avi* and

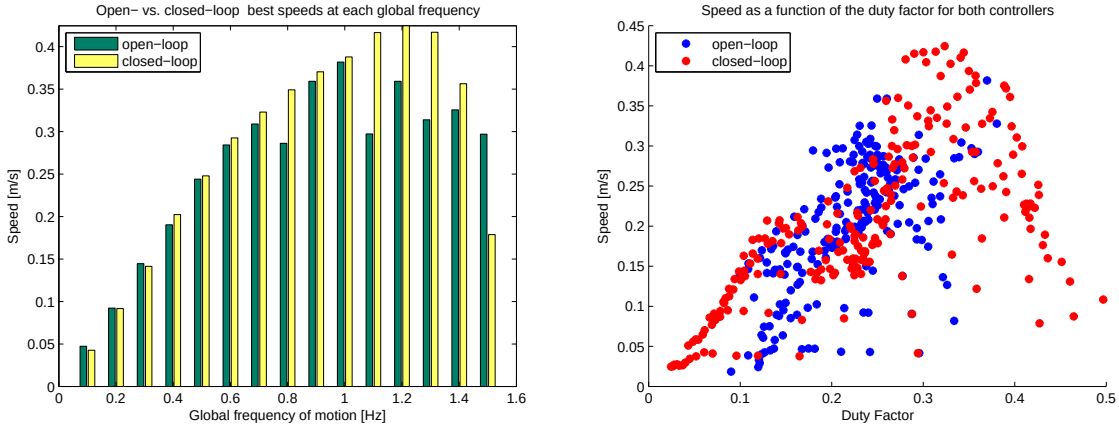


(a) Open-loop controller speed

(b) Closed-loop controller speed

Figure 6.24: Speed [m/s] as function of f_{stance} and f_{swing} for open- and closed-loop in terrains with low limb friction.

closed_low_limb_friction.avi show how the early onset of the phase allows faster locomotion.



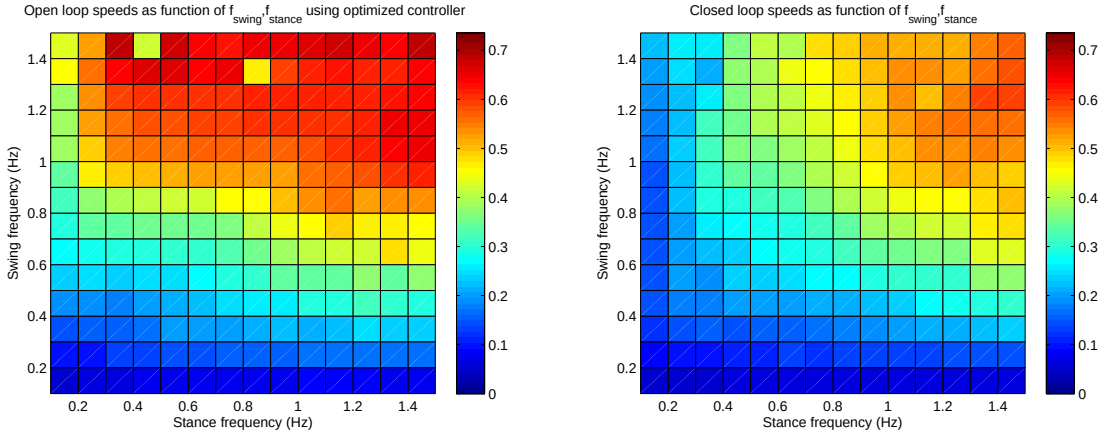
(a) Speed as function of frequency

(b) Speed as function of duty factor

Figure 6.25: Relation between speed and frequency and speed and duty factor for the case where the limb friction is low.

6.5.2 High limb friction

For high limb friction the open-loop controller shows better performance than the closed-loop (figures 6.26 and 6.27).



(a) Open-loop controller speed

(b) Closed-loop controller speed

Figure 6.26: Speed [m/s] as function of f_{stance} and f_{swing} for open- and closed-loop in terrains with high limb friction.

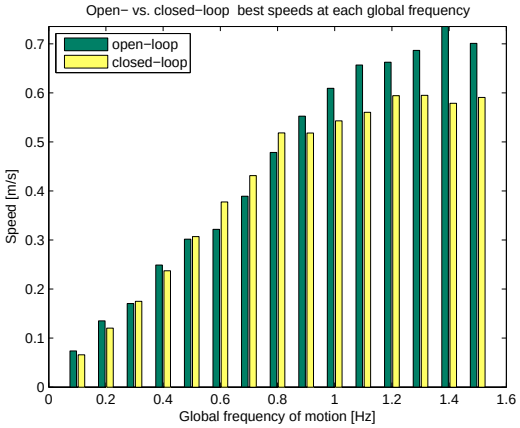
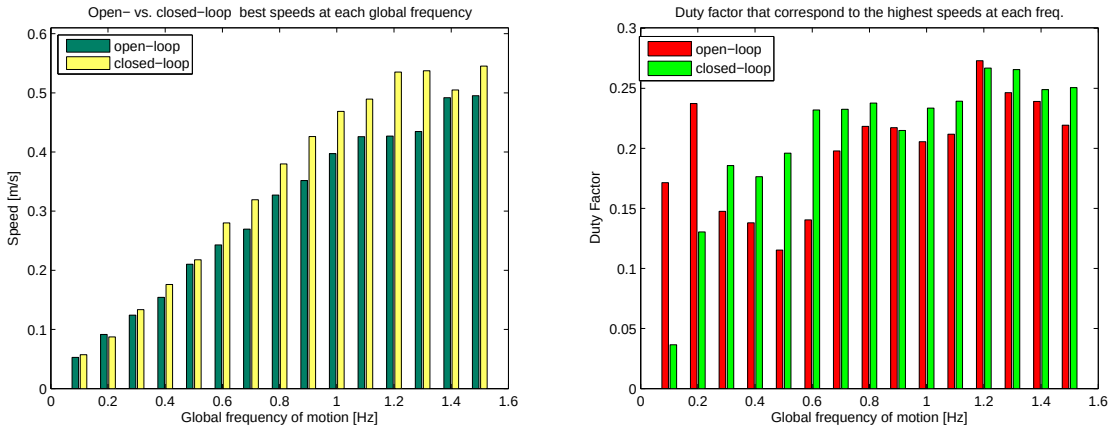


Figure 6.27: Speed as function of global frequency of motion when limbs have high friction coefficients in contact with the ground.

6.5.3 Low global friction

The case of low global friction corresponds to when the whole body is subject to the same friction coefficient. This leads to situations where the whole body slides (as if it was on an surface made of ice for example). For this case the closed-loop controller seems to be able to once again successfully control the onset of stance phase, walking at higher duty factors and consequently reducing slipperiness, which leads to faster gaits especially at high frequencies (figures 6.28a and 6.28b).



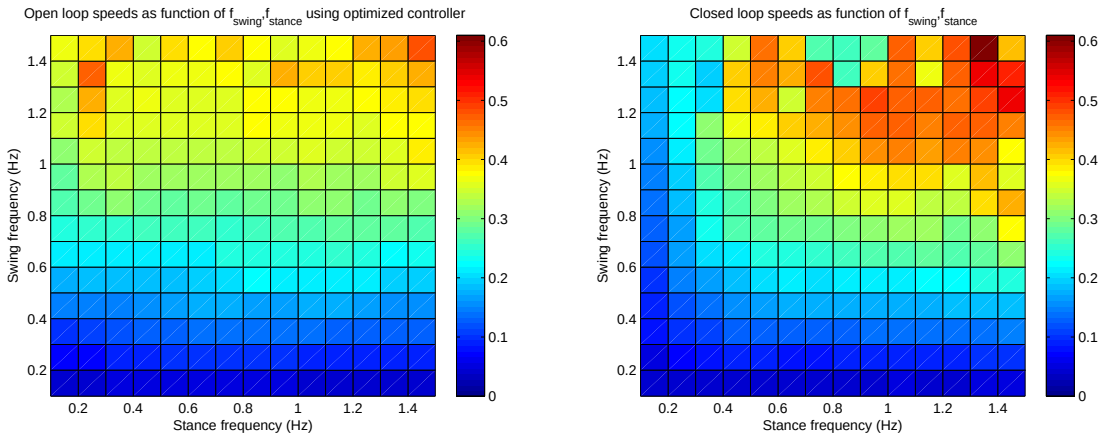
(a) Speed as function of frequency

(b) Speed as function of duty factors

Figure 6.28: Speed and duty factors for both controllers when the ground friction coefficient is uniform and very low.

6.5.4 High global friction

At high global friction all the body experiences an increase in friction forces by the same proportion. This is equivalent for example to a 'sticky' ground that offers high resistance to forward motion. It appears, from figure 6.29 that in open-loop the speed is independent from the stance frequency while in the case of closed-loop the speed increases when both frequencies increase. This indicates that again the closed-loop controller is correctly identifying the stance phase by making it last longer, but unlike for low friction, this time it favours high stance frequencies since the ground friction is higher which leads to higher forces of forward motion.



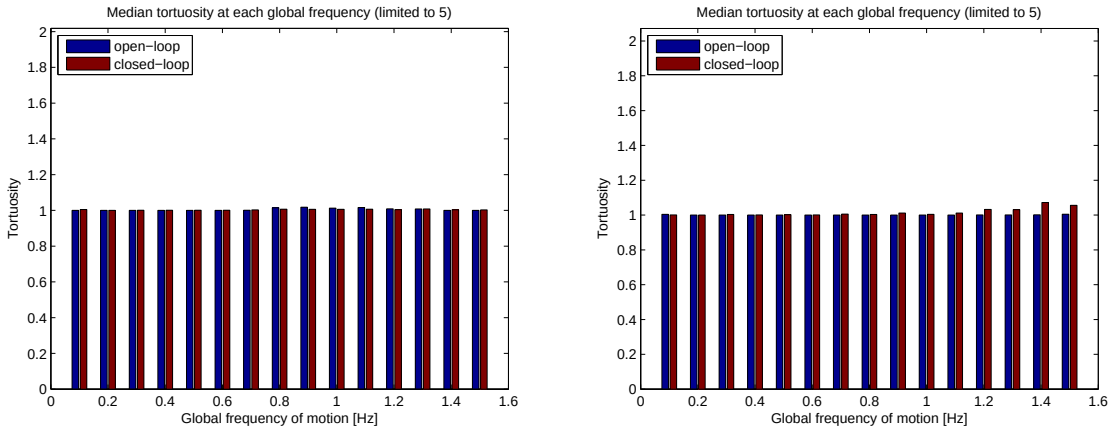
(a) Open-loop speed vs. frequencies

(b) Closed-loop speed vs. frequencies

Figure 6.29: Speed [m/s] as function of swing and stance frequencies when the ground friction is uniform and high.

6.5.5 Tortuosity for friction worlds

Figure 6.30 shows that it is a tendency for these worlds that the tortuosity does not increase. Both open- and closed-loop controller seem to show good stability.



(a) Low limb friction

(b) Low friction, all segments.

Figure 6.30: Tortuosity for worlds with low friction.

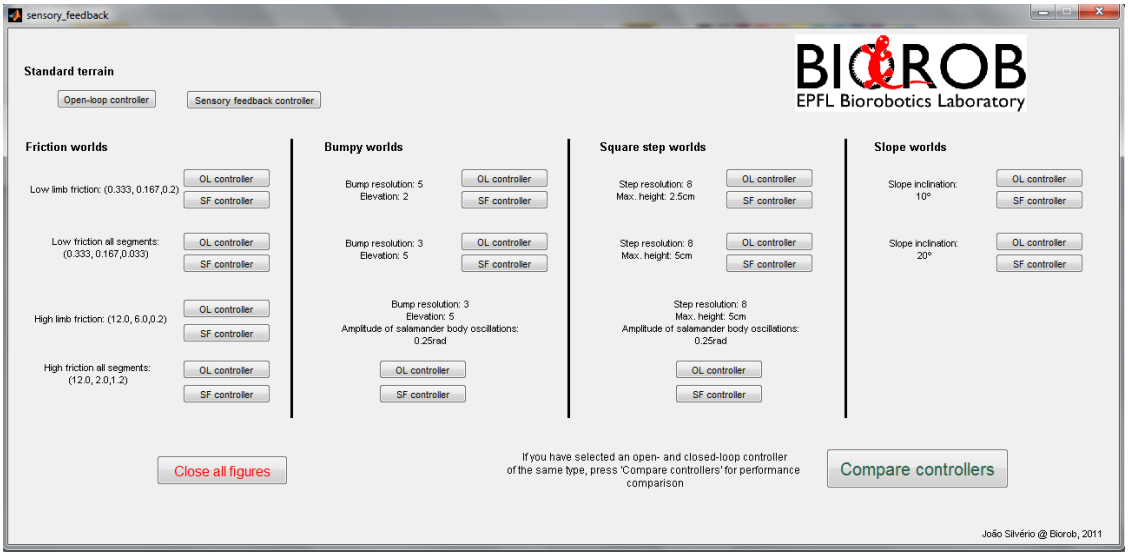


Figure 6.31: Graphical user interface for result analysis in MATLAB.

6.6 MATLAB Graphical User Interface for analysis of the results

It is available in the folder 'Optimization' of the attached DVD, a graphical user interface to simplify further analysis of the results. The interface is divided in the several Webots worlds that were used and it is possible to see the individual performance indicators of both controllers (figure 6.31). After selecting one controller at each time over the same world, it is possible to see the comparison between the two by pressing 'Compare controllers'.

6.7 Summary

Table 6.1 summarizes how the controllers performed in the different scenarios, regarding speed and gait stability. Each cell has the name of the controller that performed the best in each terrain at the corresponding performance indicator.

Terrain		**Performance indicator**	
		Speed	Tortuosity
Slope	10° inclination	Open-/Closed-loop(for high/low f_{global})	Open-loop
	20° inclination	Open-/Closed-loop(for high/low f_{global})	Even
Rough	res=5, elev.=2; A=0.5	Open-loop	Even
	res=3, elev.=5; A=0.5	Open-loop	Even
	res=3, elev.=5; A=0.25	Closed-loop	Even
Steps	Step height = 2.5cm, A=0.5	Open-loop	Open-loop
	Step height = 5cm, A=0.5	Open-loop	Open-loop
	Step height = 5cm, A=0.25	Open-loop	Open-loop
Friction	Low limb friction	Closed-loop	Even
	High limb friction	Open-loop	Even
	Low limb and body friction	Closed-loop	Even
	High limb and body friction	Closed-loop	Even

Table 6.1: Summary of controller performance.

Chapter 7

Conclusions and future work

The implementation of the sensory feedback in the salamander robot revealed that this can be an effective control method to maximize the speed of the robot under certain environments. It has been seen that optimizing an open-loop controller is time consuming and very terrain-specific, since the ideal parameters change if the environment changes as well. Besides, many terrains, such as the rough ones, or even terrains with holes, do not allow fixed, ideal, control parameters due to their unpredictability, demanding instead, a controller that can react to uncertainty.

Nevertheless, the closed-loop controller that was used, showed its best results in the terrains where there was a change of static parameters like friction or inclination. In the cases where the closed-loop controller outperformed the open-loop one, it happened because the closed-loop controller was capable of correctly identifying the stance phase. These were mainly terrains in which low duty factors represented an advantage, because of high sliperiness or inclination. In less predictable terrains, the robot dealt with the problem of unefficient body bending that was a consequence of the sensory input from the limbs which resulted, in most of the cases, in a decrease of the performance.

Regarding tortuosity, the gait stability measurement that was chosen, there seems to be really no advantage in using closed-loop control. Due to the limb-body coupling, the sensory feedback has proven to force the body CPG into bending assimetrically thus generating quite tortuos trajectories.

Some of the work that could be done in the future concerns the topic of limb-body relation, so performing an exhaustive analysis on how coupling affects locomotion in closed-loop could reveal interesting results. Taking into account that this model uses 2 CPGs making a total

of 12 oscillators, a lot could be done in trying new methods for coupling the oscillators in closed-loop, mainly regarding the coupling weights that are used and how limbs project to the body, since the sensory input is at the limbs.

Another interesting idea would be developing a model with two degrees of freedom for the limbs, in order to allow the implementation of more sofisticated methods of sensory feedback. Also, it should be interesting if the controller that was used in this project could be applied to the real robot, so discovering efficient ways of detecting ground touching in the real salamander robot would bring the problem to real life , where no simulation artifacts can affect the results.

Bibliography

[1] A. J. Ijspeert, "Introduction to nonlinear dynamical systems," *Models of Biologically Inspired Sensory-Motor Systems lecture notes*, 2009.

[2] A. J. Ijspeert, A. Crespi, D. Ryczko, and J.-M. Cabelguen, "From swimming to walking with a salamander robot driven by a spinal cord model," *Science*, vol. 315, pp. 1416–1419, 2007.

[3] L. Righetti and A. J. Ijspeert, "Pattern generators with sensory feedback for the control of quadruped locomotion," *Proceedings of the 2008 IEEE International Conference on Robotics and Automation (ICRA 2008)*, vol. 26, pp. 819–824, May 19-23, 2008.

[4] H. Kimura, Y. Fukuoka, and A. H. Cohen, "Adaptive dynamic walking of a quadruped robot on natural ground based on biological concepts," *The International Journal of Robotics Research*, vol. 26, p. 475, 2007.

[5] S. Grillner, "Neural networks for vertebrate locomotion," *Scientific American*, pp. 64–69, January, 1996.

[6] S. Grillner, T. Deliagina, O. Ekeberg, A. E. Manira, R. H. Hill, A. Lansner, G. N. Orlovsky, and P. Walln, "Neural networks that co-ordinate locomotion and body orientation in lamprey," *Elsevier Science Ltd.*, vol. 18, No.6, pp. 270–279, 1995.

[7] A. K. Kozlov, E. Aurell, G. N. Orlovsky, T. G. Deliagina, P. V. Zelenin, J. Hellgren-Kotaleski, and S. Grillner, "Modeling postural control in the lamprey," *Biological Cybernetics*, vol. 84, pp. 323–330, 2001.

[8] S. Chevallier, A. J. Ijspeert, D. Ryczko, F. Nagy, and J.-M. Cabelguen, "Organisation of the spinal central pattern generators for locomotion in the salamander: Biology and modelling," *Brain Research Reviews*, vol. 57, no. 1, pp. 147 – 161, 2008.

[9] A. Crespi and A. J. Ijspeert, *Artificial Life Models in Hardware*, chapter *Salamandra Robotica*: A Biologically Inspired Amphibious Robot that Swims and Walks, pp. 35–64. London: Springer, 2009.

[10] A. J. Ijspeert, A. Crespi, and J. M. Cabelguen, "Simulation and robotics studies of salamander locomotion. Applying neurobiological principles to the control of locomotion in robots," *Neuroinformatics*, vol. 3, no. 3, pp. 171–196, 2005.

[11] L. Righetti and A. J. Ijspeert, "Design methodologies for central pattern generators: an application to crawling humanoids," in *Proceedings of Robotics: Science and Systems*, pp. 191–198, 2006.

[12] A. Pikovsky, M. Rosenblum, and J. Kurths, *Synchronization: A Universal Concept in Nonlinear Sciences (Cambridge Nonlinear Science Series)*. Cambridge University Press, 1 ed., May 2003.

[13] A. J. Ijspeert, "Central pattern generators for locomotion control in animals and robots: A review," *Elsevier: Neural Networks*, vol. 21, pp. 642–653, 2008.

[14] M. Kovac, M. Fuchs, A. Guignard, J.-C. Zufferey, and D. Floreano, "A miniature 7g jumping robot," in *Proceedings of the IEEE International Conference on Robotics and Automation (ICRA'2008)* (S. Hutchinson, ed.), pp. 373 – 378, 2008.

[15] A. Ahmadi, E. Mangieri, K. Maharatna, and M. Zwolinski, "Physical realizable circuit structure for adaptive frequency Hopf oscillator," in *NEWCAS-TAISA'09*, IEEE, June 2009.

[16] L. Righetti, J. Buchli, and A. Ijspeert, "Dynamic hebbian learning in adaptive frequency oscillators," *Physica D*, vol. 216, no. 2, pp. 269–281, 2006.

[17] M. Vidyasagar, *Nonlinear systems analysis*. Upper Saddle River, NJ, USA: Prentice-Hall, Inc., 1992.

[18] J. van den Kieboom, "Biped locomotion and stability - a practical approach," Master's thesis, University of Groningen (Netherlands), Dept. of Artificial Intelligence, 2009.

[19] R. M. Alexander, "The gaits of bipedal and quadrupedal animals.," *The International Journal of Robotics and Research*, vol. 3, pp. 49–59, 1984.

[20] Cyberbotics. `http://www.cyberbotics.com`.

[21] Open Dynamics Engine Manual. `http://opende.sourceforge.net/wiki/index.php/Manual`.

[22] © 2011 Cyberbotics Ltd., "Webots user guide - release 6.3.4." `http://www.cyberbotics.com/guide.pdf`, February 2011.

[23] © 2011 Cyberbotics Ltd., "Webots reference manual - release 6.3.4." `http://www.cyberbotics.com/reference.pdf`, February 2011.

[24] EPFL Biorobotics Laboratory (BioRob). `http://birg2.epfl.ch/images/salam_robot/lowres_a_herzog_outside3.jpg`.