Artificial evolution of controllers based on non-linear oscillators for bipedal locomotion

Julien Nicolas

December 16, 2005

Master Thesis, winter 2005-2006
Computer and Communication Sciences

Supervisor: Ludovic Righetti, BIRG EPFL
Prof. resp: Auke Jan Ijspeert, BIRG EPFL

Contents

Chapter 1

Acknowledgments

I would like to deeply thank my supervisor, Ludovic Righetti, for his precious advices as well as for his theoretical and technical support.
I would also like to thank Auke Ijspeert and the BIRG for giving me the opportunity to work in such an interesting field as bio-inspired robotics.

Chapter 2

Objectives

Millions of years of evolution made humans able to walk and run with an incredible precision. Although bipedal locomotion could seem to be a simple mechanical problem, researches done on artificial locomotion proved that it is far from obvious to generate a walking gait that is as fast and stable as humans do.
In this work, we will directly inspire us from biological considerations by using CPG based controllers and artificial evolution. First, several models of CPGs, implemented on a simulated robot, will be explored using genetic algorithms to find which system is best suited to locomotion. After that, we will try to improve our model by adding feedback pathways in the CPGs in order to be able to modify the speed of locomotion. In the same time, feedback pathways will be used to let our robot deal with external perturbations.

Chapter 3

Introduction

3.1 Biological bipedal locomotion

Amongst all skills of living species, locomotion is certainly one of the crucial ability that emerged from millions of years of evolution. Almost each species, going from unicellulars to humans, developed a form of locomotion perfectly suited to its morphology and environment to ensure its survival. In spite of the huge diversity of the animal reign, it has been shown that animals, and in particular vertebrates, share some common properties although they use completely different forms of locomotion [9].
To achieve locomotion the neural system generates rhythmic signals that are sent to the musculo-skeletal system in order to produce torques on the different joints of the animal [2]. Locomotion can be described by the interaction of three elements:

1. spinal central pattern generators (CPG).
2. sensory feedback.
3. descending supra-spinal control.

Bipedal locomotion seems to be more complicated than the process shown above since the balance is much more important with only two legs, making control crucial. It is not proved that humans use a simple CPG based locomotion like a lamprey or a fish would do, but we still have some evidences that the locomotion pattern are generated at the spinal level and are sent through the reticulospinal pathways. We can consider that humans use a system that is comparable to a CPG for their locomotion.
Studying bipedal locomotion in animals and in particular humans, gives us a good idea on how their gait could be modeled. Although bipedal locomotion has been studied by scientists of different domains, this problem has not been completely solved yet.

3.2 Bipedal locomotion in robotics

Researches made on bipedal locomotion in robotics have shown that we must mainly consider three different approaches to control walking:

Statically stable walking This corresponds to the case when the projection on the ground of the robot's center of gravity always lies within the footprint or the area between the two footprints of the robot. With this hypothesis the locomotion is quite easier to perform but is very slow.

Dynamic walking In this case, the robot's center of gravity can be elsewhere and corrections must be done at any time to maintain the robot on his feet. To achieve stability in those conditions it is mandatory to know everything from the robot's dynamics and in particular the speed and inertia of each of its parts. This method is without a doubt the more efficient in terms of velocity but is very tricky to implement. It also corresponds to the human locomotion.
The next section introduces different approaches based on dynamic walking.

Passive walking A third type of biped locomotion that has been studied by many researchers is called passive walking [6]. In this case no actuators are used and the robot uses its own weight and dynamics to walk down a slope without external input or control. The system has a stable limit cycle and the robots are even able to deal with small perturbations. We can also mention the concept of passive dynamic walkers. In this case small power sources on their ankles and/or hips are included to simulate gravity, which allows the robots to walk on a level ground without control servos.
This approach is very important since it allows to learn many features of robot's dynamics that are also useful for other kind of locomotion [5, 4].

3.2.1 Trajectory based methods

The basic idea of this method is to find walking trajectories and uses a criterion based on dynamic equations to test and prove that the locomotion is stable. These trajectories could be designed by trial-and-error or from predefined human recordings (motion capture). This approach does not provide any methodology to implement locomotion in a robot. It only allows to prove and measure the stability of the gait which is still an useful tool to analyze artificial motion.
In 1990, Vukobratovic [7] proposed a method called Zero Moment Point (ZMP) which seems to be the most promising trajectory based method. The Zero Moment Point is the point on the ground where the moment of inertial forces and gravitational forces has no component along the horizontal plane. To achieve the stability of the locomotion, the ZMP must always be

located within the robot's footprint. We can also notice that Vukobratovic distinguishes to different phases: single-support when only one foot touches the ground and double-support when both feet are on the ground. During this last phase, the locomotion is stable if the ZMP is located between the two footprints. This method also allows to deal with perturbations by introducing online stabilization.
This method has been successfully implemented on several robots such as Sony's QRIO and Honda robot.

Advantages

- Can be used to prove stability.
- Resistant against small perturbations.
- Well-suited for implementations on real robots.
- Can be adapted to dynamic walking.

Drawbacks

- A perfect knowledge of the robot's dynamics is mandatory.
- Not suited for unknown environment.
- Switching from online control to wanted trajectory is hard to achieve.
- Not suited to design locomotion controllers.

3.2.2 Heuristic control methods

As the ZMP method does not provide any methodology to design controllers, heuristics were developed to deal with that problem. In addition, heuristics decrease considerably the complexity of the dynamic equations involved in trajectory based methods and make implementation much easier.
One of the most efficient and interesting method using heuristics called Virtual Model Control (VMC) has been developed by J. Pratt [8] at the MIT's Leglab. This approach is based on virtual mechanical components like springs, dampers, dashpots,... These elements generate real torques or forces on the actuators. These torques create the same effect as that the virtual components would have produced if they were real. This method allows to achieve different kinds of movements just by choosing the appropriate virtual elements. The torque needed to produce the virtual force is typically computed using the Jacobian relating the reference frame of the virtual component to the robot.
In practice, movements are relatively easy to implement on controllers. For each different motion or behavior, an appropriate set of virtual components

has to be defined. For example, to maintain the robot in an upright and stable position, a kind of granny walker has to be introduced as virtual component to guarantee stability. To make the robot walk, a typical component is a virtual dogtrack bunny associated with a damper mechanism (cf fig 3.1). This model implies to consider different states to deal with the different actuation phases this means that, in a given phase, only certain motors are activated. This problem can be solved using a finite state machine.
The MIT's Leglab successfully implemented the method described above on their Flamingo and Turkey robots[1]

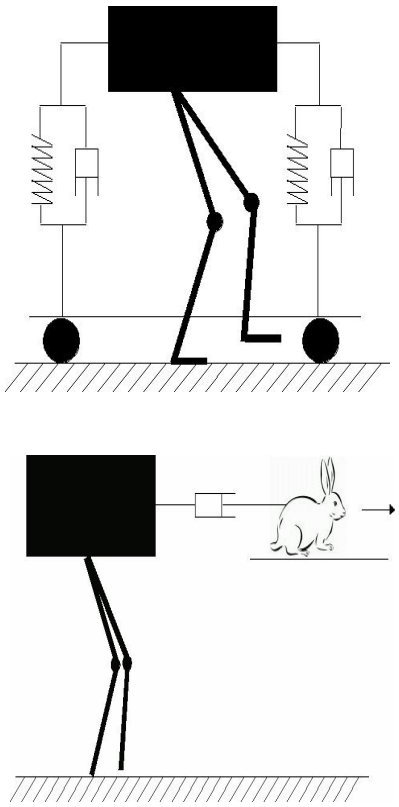

Figure 3.1: Example of virtual components

Advantages

- Detailed methodology to design controllers.
- Very intuitive approach.
- Complete knowledge of the environment is not required.
- Can deal with small perturbations.

[1]More details are given in [8]

- Efficient online control.

Drawbacks

- A perfect knowledge of the robot's dynamics is mandatory.
- The designer must ensure that the torques generated on the joints are supported by the robot to avoid damages.
- The mechanism based on finite state machines causes brutal transitions between the different phases.

3.2.3 CPG and reflex methods

The approach described in this section is based on Central Pattern Generator (CPG) and is directly inspired from biological considerations as mentioned in section 3.1. The main advantage of this method, compared to those described above, is that here, we do not need a perfect knowledge of the robot's dynamics. It is a more general and adaptive method to design controllers for bided robots. Reflexes are produced by the robot's feedback sensors and are used to manage perturbations and balance control.
The work done by Taga [11] proves that CPGs and reflexes can be used for locomotion on biped robots.
CPGs can be represented by different mathematical models such as oscillators, artificial neurons, vector fields,... Each CPG usually represent one degree of freedom (DOF) of a joint. Oscillator based CPGs are using the concept of limit cycles which are very convenient in the case of locomotion since they can return to their stable state after a small perturbation and are almost not influenced by a change in the initial conditions.

CPG and reflex loop Different models could be used to represent the interaction between the CPG and the reflex system. The choice of the system depends on the complexity of the feedback mechanism we want to represent. Note that a system with complicated feedback will be much more difficult to design as more parameters are involved in the system. Some models of feedback pathways are described below.

Open-loop Here, we present a simple model based on CPGs known as open-loop model (cf. fig 3.2). The CPG interacts directly with the actuators but receives no feedback from the sensors and the actuators (open-loop). Such a model has the advantage that it allows a relatively simple implementation since no sensors are used. However it is not very well suited to deal with perturbations in the environment.

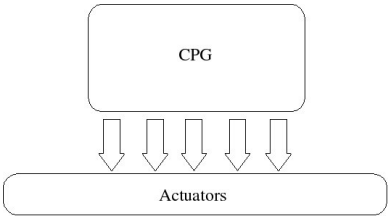

Figure 3.2: A simple open-loop model with no feedback. This model will be used in the first part of this project

Closed-loop In this case, we use sensor feedback to close the loop, this means that the CPG receives information from different kinds of sensors. The simplest method is to consider feedback pathways going only from the actuators to the CPGs (cf. fig 3.3). This approach allows us to maintain a system that is still relatively simple to represent and has the ability to be robust against small external perturbations.
A more complete and promising closed-loop model consists of three levels of feedback (cf. fig 3.4):

1. The fast reflex interferes directly on the joint's actuators and do not modify the CPG. It should correct the robot's position very quickly.

2. The CPG reflex influences the middle level of the system and corresponds to the feedback used on the simplified version of the closed-loop model described above. This kind of feedback has been introduced to deal with external perturbations such as wind, ground deformation,...

3. The global reflex interacts with the system on its higher level and will be used to modify the global behavior of the robot when encountering a new situation such as a slope.

In addition to the mechanism described above, two control system are added:

1. The visual system interacts with the CPG and directly with the actuators.

2. The vestibular system is also connected to the CPGs and the actuators. It is used to control the global balance of the robot.

This method seems to be very promising since it has already been successfully implemented on an artificial lamprey [14] and more recently on an artificial salamander [12] that even has the ability to switch between swimming and walking [13].

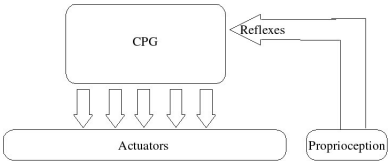

Figure 3.3: A minimal closed-loop model. Only the actuators interact with the CPGs

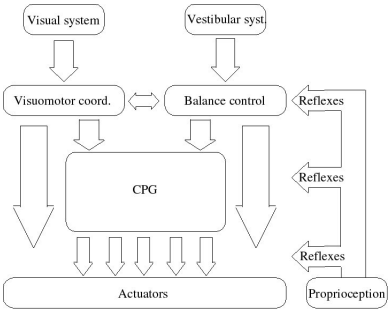

Figure 3.4: A closed loop model with three different reflexes type influencing different parts of the system. On the top left, the visual system interacts with the CPG and directly on the actuators. The vestibular system controls the balance of the robot and influences the CPG and the actuators.

Advantages

- A perfect knowledge of the robot's dynamics is not required.
- Robust against small perturbations.
- Almost not influenced by a change in the initial conditions.
- Oscillations produce smooth locomotion.
- Complex control input not required since the oscillation is generated by itself.

Drawbacks

- No clear methodology to design controllers.
- Parameters are hard to find by hand. So it is recommended to use learning algorithms.

Chapter 4

The robot

4.1 Real biped robots

In this last decade a lot of improvements have been done in the development of autonomous biped robots. Big companies like Honda, Fujitsu or Sony began developing expensive robots for research. Since a few year, robots like Qrio or Sony's Aibo are available at a reasonable price and are subject to several studies.
In this project, we will focus on Fujitsu's robot called HOAP 2 (Humanoid for Open Architecture Platform) [23]. HOAP 2 is 50 cm tall and is only 7 kg which makes it really easy to handle. Its 24 degree of freedom allows it to perform a lot of movements including hands and head control (fig 4.1). This robot also has servo angle sensors, gyroscope and pressure sensors in their feet, useful for locomotion with feedback.

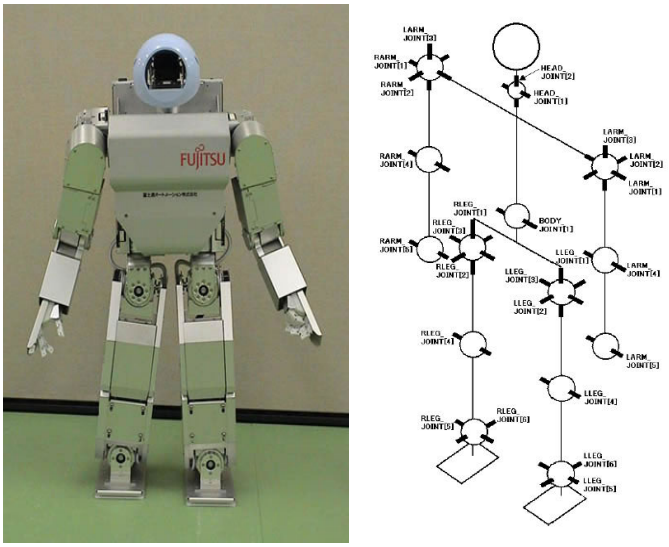

Figure 4.1: Left: The Hoap2 robot. Right: The different DOFs available on Hoap 2

4.2 The simulator

One of the drawbacks of these robots is that they are very fragile and expensive. As artificial evolution will be used to tune the controller parameters, we expect the robot to fall hundred times before finding a good trajectory. Experimenting controllers on the real robot would then certainly break it. Another advantage of simulated robots in the context of this project is that we can launch many simulations with different parameters.
The reasons given above clearly encouraged us to perform our research on a simulated robot. For this purpose, we used Webots [20], which is a complete toolkit to model, program and test robot's behaviors in their environment. Webots is developed by Cyberbotics and has become a reference software used by many universities and researchers. This software includes many useful devices such as cameras, light sensors, touch sensors, emitters/receivers, gps,... One of the most powerful feature of Webots is its fully customizable VRML based environment. For example, this will allow us to add external perturbations such as wind, slopes, steps,... This simulator also uses a powerful dynamic engine based on ODE (Open Dynamic Engine), which is an open-source library for simulating rigid body dynamics.
Webots also provides a collection of existing robots including Hoap 2 that was designed by Pascal Cominoli during his Master's thesis at the BIRG (EPFL) [21]. This virtual robot corresponds exactly to the real one in terms of weight, size and servo specifications.

Chapter 5

The CPG model

As described in 3.2.3 the model studied in this work uses Central Pattern Generators to control the robot's servos. To represent these CPGs and to generate signals, we use one or several non linear oscillators that are coupled together. A very interesting property of non linear oscillators is the concept of limit cycle [15][16]. A limit cycle can be described as an isolated closed trajectory of the system, which can be stable or unstable. The first case is very important since all trajectories close to the limit cycle are attracted by it. This principle will be useful to generate robust signals that can return to their stable position after a small perturbation. This section describes the different oscillators and coupling models used in this work.
For each experiment described in this chapter, we will use a simplified model of the robot this means that not all the servos will be used since some of them are not relevant for locomotion. We will then focus on 11 servos (1 for the body, 2 for each hip, 1 for each knee and 2 for each ankle). Both legs are connected to the body's servo and a phase difference of π is imposed between them since there is an obvious symmetry between the joints' angles of each leg. The servos responsible for the twist of the hip and the ankle are not used because they do not play an important role in walking.
As the servos' position is not always oscillating around 0, we add a bias to the output of each oscillators. Note that this bias will also be evolved and is used in every model described below.

5.1 Open-loop

5.1.1 Hopf oscillator

The first model used in this work is based on the Hopf's oscillator, a relatively simple system for preliminary experiments. Its main advantage is the possibility to easily control the amplitude and the frequency of the signal independently. The following equations describe this oscillator.

$$\dot{x} = (\mu - r^2)x - \omega y \quad (5.1)$$
$$\dot{y} = (\mu - r^2)y - \omega x \quad (5.2)$$

Where $r = \sqrt{x^2 + y^2}$, $\mu > 0$ determines the amplitude of the signal and ω controls the frequency of the oscillator. This oscillator has a stable limit cycle with radius $\sqrt{\mu}$ and angular velocity ω rad/s.

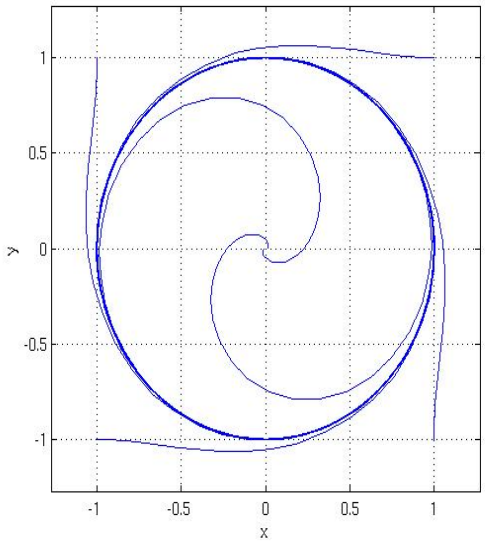

Figure 5.1: Phase plot of the Hopf's oscillator with different initial conditions. Here we set $\mu = 1.0$ and $\omega = 1.0$.

The coupling

1-way coupling With this first model, we will start with a one-way coupling, It acts only from one oscillator to the other in one direction (not in reverse) as shown on figure 5.2. This coupling is described by the following equations

$$\dot{x}_i = (\mu_i - r_i^2)x_i - \omega y_i + \epsilon_i x_{i-1} \quad (5.3)$$
$$\dot{y}_i = (\mu_i - r_i^2)y_i - \omega x_i \quad (5.4)$$

Where ϵ_i is the coupling strength ($\epsilon_1 = 0$ since the first servo is not coupled with any other joint) and ω has the same value for each oscillator. With this coupling model, the amplitude of the coupled oscillator's signal cannot be controlled as in the case of a single oscillator. This means that the amplitude depends on the μ_i parameter but is also influenced by the coupling

strength ϵ_i. This phenomenon is explained by the fact that the two involved oscillators are in resonance since their respective frequencies are the same. To deal with this problem, we will set a sufficiently large interval of possible values for the amplitude parameters, which will allow us to roughly set these values. The genetic algorithm will tune these values more precisely. An interesting property of this coupling model is that we can introduce a simple phase difference by changing the sign of ϵ_i. If $\epsilon_i < 0$, then the phase difference is equal to π and if $\epsilon_i > 0$ the phase difference is null (cf. fig 5.4). This allowed us to impose phase differences between the different servos as shown on figure 5.2. However, this simple model does not allow to produce other phase differences than those mentioned above and will probably not produce a very realistic gait. Completely arbitrary phase differences could have been introduced by using the two variables x_i and y_i in the coupling, but this would consequently enlarge the search space of the GA (five extra parameters). As the GA took approximately 12 hours to find a good solution for the model described above, we tried to simplify each model used as much as possible.

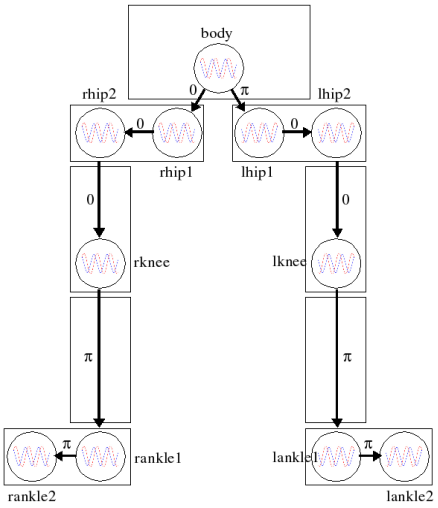

Figure 5.2: The different DOF used in this work. The arrows represent the unidirectional coupling of this first model. The values on the arrows correspond to the phase differences we imposed between the different oscillators.

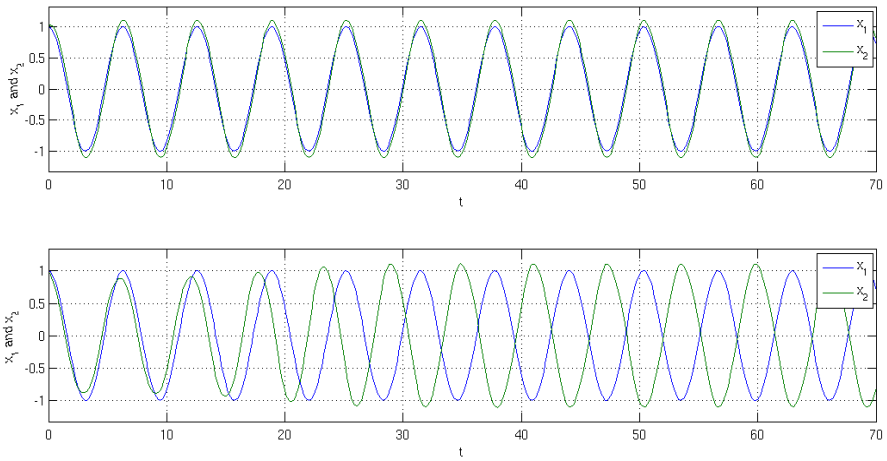

Figure 5.3: These figures show the first variables (x_1 and x_2) of two coupled oscillators. We can see here that for $\epsilon_i < 0$, the phase difference after a few steps is π (top plot). For $\epsilon_i > 0$ we have a phase difference equal to 0 (bottom plot). We can also see that the amplitude of x_2 is slightly changed due to the phenomenon of resonance. The parameters used here are the followings: $\mu_1 = \mu_2 = 1.0$, $\omega = 1.0$ and $\epsilon_2 = \pm 0.3$

2-way coupling The second coupling model we are using is the same as the previous one but with influence in both directions. Two adjacent oscillators are coupled in both directions. Note that the oscillators are also chained along the leg (cf. fig 5.5). With this coupling we are now able to introduce more complex phase differences like $\pi/2$ and $-\pi/2$ as described in table 5.1. Notice that, to reduce the search space, we set the phase differences between the different servos by hand (cf. fig 5.5). These phase differences were found empirically after a few tests and seemed to be the most realistic ones for this model. The following equations describe that model.

$$\dot{x}_i = (\mu_i - r_i^2)x_i - \omega y_i + \epsilon_{i,i-1}x_{i-1} + \epsilon_{i,i+1}x_{i+1} \quad (5.5)$$

$$\dot{y}_i = (\mu_i - r_i^2)y_i - \omega x_i \quad (5.6)$$

In this case, there are two parameters for the coupling strength ($\epsilon_{i,i-1}$ and $\epsilon_{i,i+1}$), one for each adjacent oscillator.

value of the coupling coefficients $\epsilon_{1,2}$ and $\epsilon_{2,1}$	phase difference
$\epsilon_{1,2} = \epsilon_{2,1} > 0$	0
$\epsilon_{1,2} > 0$ and $\epsilon_{2,1} = -\epsilon_{1,2}$	$\frac{\pi}{2}$
$\epsilon_{2,1} > 0$ and $\epsilon_{1,2} = -\epsilon_{2,1,}$	$-\frac{\pi}{2}$
$\epsilon_{1,2}, \epsilon_{2,1} < 0$ and $\epsilon_{1,2} > \epsilon_{2,1}$	π
$\epsilon_{1,2}, \epsilon_{2,1} < 0$ and $\epsilon_{1,2} < \epsilon_{2,1}$	$-\pi$

Table 5.1: Phase differences between two Hopf oscillators with 2-way coupling for different value of the coupling coefficients. The relation between these parameters will be used to produce a phase difference between the different servos of the robot.

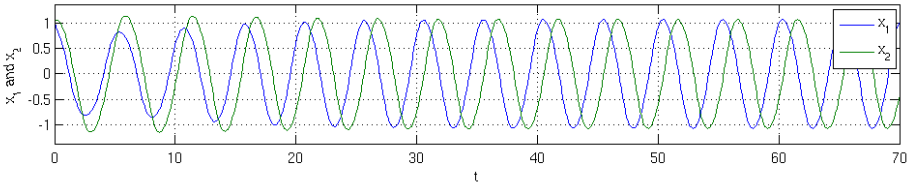

Figure 5.4: First variables (x_1 and x_2) of two oscillators coupled in both directions. This example shows a phase difference of $\pi/2$ between both signals. The parameters used here are the followings: $\mu_1 = \mu_2 = 1.0$, $\omega = 1.0$, $\epsilon_{1,2} = 0.5$ and $\epsilon_{2,1} = -0.5$

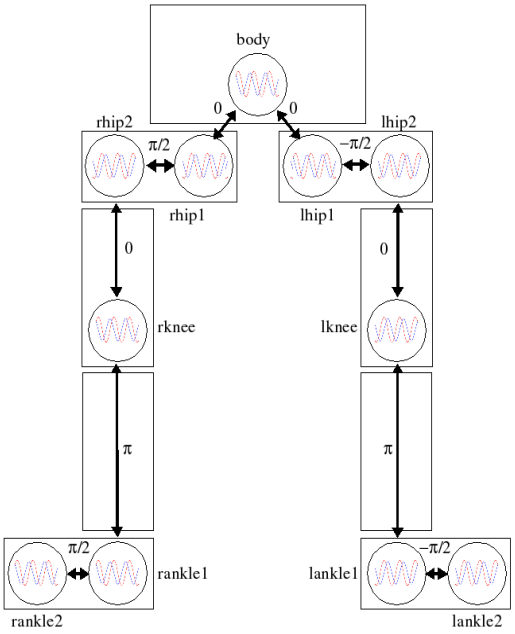

Figure 5.5: Bidirectional coupling between the servos. The phase differences we imposed are shown on the arrows.

Hopf with 2 oscillators per DOF In spite of its simplicity, Hopf oscillator has one major drawback for our use: it only generates simple sinusoidal signals. Even if such signals could be suited for locomotion, we know that real walking trajectories are more complex. In this section, we will introduce a new model in which two Hopf oscillators are used for each degree of freedom. More precisely we will use a weighted sum of the output of two coupled oscillators as global output of the system for each servo. As in the previous experiment, each pair of oscillator is coupled unidirectionally along the leg (cf. fig 5.7). The following equations describe this model.

$$\dot{x}_{1,i} = (\mu_i - r_{1,i}^2)x_{1,i} - \omega y_{1,i} + \epsilon_i x_{1,i-1} x_2 \quad (5.7)$$

$$\dot{y}_{1,i} = (\mu_i - r_{1,i}^2)y_{1,i} - \omega x_{1,i} \quad (5.8)$$

$$\dot{x}_{2,i} = (\mu_i - r_{2,i}^2)x_{2,i} - 2\omega y_{2,i} + d_i x_{1,i} \quad (5.9)$$

$$\dot{y}_{2,i} = (\mu_i - r_{2,i}^2)y_{2,i} - 2\omega x_{2,i} \quad (5.10)$$

$$s_{out,i} = h_{1,i}x_{1,i} + h_{2,i}x_{2,i} \quad (5.11)$$

with $\epsilon_1 = 0$ since the corresponding servo is not coupled to any other joint.

Here the parameters of each oscillators are the same as for the simple Hopf oscillator. The ϵ_i parameters correspond to the coupling strength be-

tween the two adjacent systems and the coefficient d_i allows us to introduce a phase difference between the two oscillators of each DOF.
The output sent to the servos corresponds to a weighted sum of the first variables of each oscillator. The parameters $h_{1,i}$ and $h_{2,i}$ represent the weights of this sum. We can also notice that the natural frequency of the second oscillator ($x_{2,i}$, $y_{2,i}$) is the double (and by consequence the first harmonic) of the one that rules the first oscillator ($x_{1,i}$, $y_{1,i}$). This model is then designed to produce a signal that corresponds to the first two terms of a Fourier series and should be closer to the real biped locomotion.

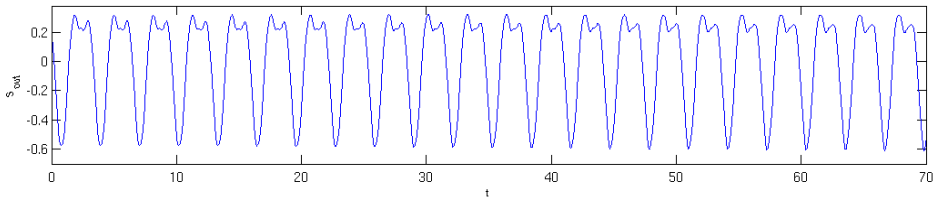

Figure 5.6: Example of evolution of the output $s_{out,i}$ generated by the pair of oscillators described above. We see that the shape of the signal is more complex and can be modified by changing the weights of the output sum. Here the parameters are $\mu_i = 1.0$, $\omega = 2.0$, $d_i = 0.3$, $h_{1,1} = 0.4$ and $h_{2,1} = 0.2$.

For this experiment, we kept some parameters found with the simple 1-way Hopf model, in particular we reused the values found for the amplitudes μ_i, the frequency ω and the bias. So the only parameters that are evolved here are the different coupling strengths and weights (ϵ_i, $h_{1,i}$, $h_{2,i}$ and d_i). The phase difference between two consecutive pairs of oscillators can be modified by changing the ϵ_i parameter as it was done with the simple 1-way Hopf model (cf. fig 5.2).

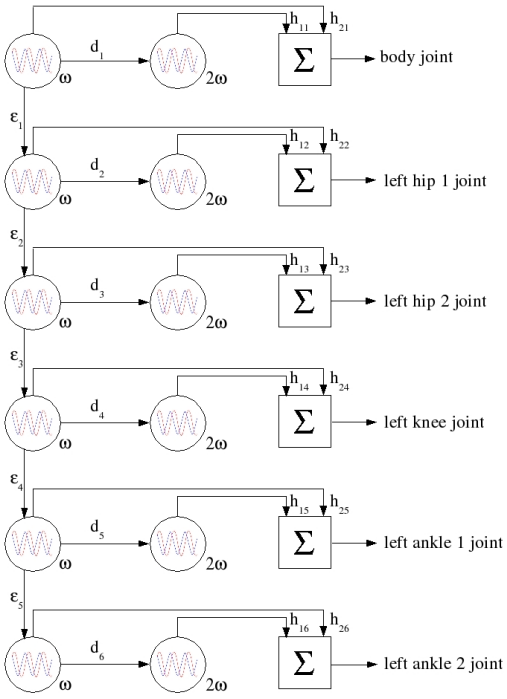

Figure 5.7: Model used with 2 oscillators for each DOF. The same coupling applies to the right leg. The weight of the different couplings involved is shown on the arrows.

5.1.2 Rayleigh's oscillator

The oscillator

To explore different signal shapes, we chose to use a relaxation oscillator as next model. More precisely the model used in this second experiment is based on the Rayleigh's oscillator. The following equations describe this model.

$$\dot{x} = y \tag{5.12}$$

$$\dot{y} = \delta(1 - qy^2)y - \omega^2 \tag{5.13}$$

In order to reduce the search space of the genetic algorithm, we set $\delta = 1$. The q coefficient allowed us to control the amplitude of the signal and the parameter ω was used to modify the frequency. Note that a variation of ω slightly changes the amplitude of the signal, but the genetic algorithm should deal with that.

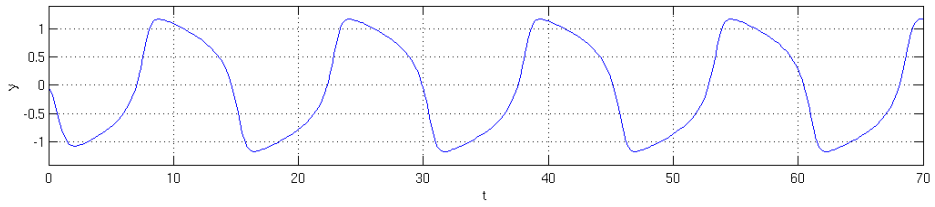

Figure 5.8: Evolution of the variable y for the Rayleigh oscillator. We see that the shape of the signal is very different from a simple sinus and is typical of relaxation oscillators. Here we set $\rho = 1.0$, $q = 1.0$ and $\omega = 0.5$.

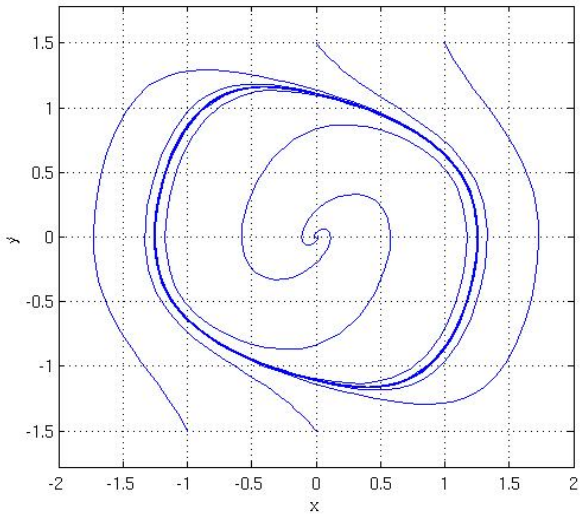

Figure 5.9: Phase plot for the Rayleigh oscillator with different initial conditions. The system converges to a stable limit cycle. Here we set $\rho = 1.0$, $q = 1.0$ and $\omega = 1.0$.

The coupling

$$\dot{x}_i = y_i \tag{5.14}$$
$$\dot{y}_i = (1 - q_i y_i^2) y_i - \omega^2 x_i - \epsilon_i y_{i-1} \tag{5.15}$$

As for the first coupling model used with the Hopf oscillator, we chose to use an unidirectional coupling along the leg from the hip to the ankle (cf. fig 5.2). Notice that ϵ_i can be used to impose a phase difference between two coupled oscillators. Here again, $\epsilon_i < 0$ produces a phase opposition while with $\epsilon_i > 0$, the oscillators are in phase. With this model, we also set the phase differences by hand (cf. fig 5.2).

5.1.3 Matsuoka's oscillator

The oscillator

The third oscillator model used in this work has been first studied by Matsuoka [17, 18] and is widely used in many researches on robotics and CPGs. It is based on the mutual inhibition of two artificial neurons that generate a periodic signal as output. The model consists of four state variables (cf. figure 5.11) and is governed by the following equations.

$$\tau_1 \dot{x}_1 = c - x_1 - \beta v_1 - \mu[x_2]^+ - \sum_j h_j[g_j]^+ \quad (5.16)$$

$$\tau_2 \dot{v}_1 = [x_1]^+ - v_1 \quad (5.17)$$

$$\tau_1 \dot{x}_2 = c - x_2 - \beta v_2 - \mu[x_1]^+ - \sum_j h_j[g_j]^- \quad (5.18)$$

$$\tau_2 \dot{v}_2 = [x_2]^+ - v_2 \quad (5.19)$$

$$y_k = [x_k]^+ = max(x_k, 0) \quad (5.20)$$

$$y_{out} = y_1 - y_2 \quad (5.21)$$

Each neuron is represented by two equations and has a state variable x that corresponds to the firing rate and a variable v that represents the self-inhibition. The two neurons inhibit and excite each other alternatively producing, as output, an oscillation that is given by 5.21. The parameter c corresponds to a tonic input and is directly proportional to the output signal and so allows to control the amplitude. The β and μ coefficients are constant values that are fixed arbitrarily in our experiment. The terms $-\sum_j h_j[g_j]^+$ and $-\sum_j h_j[g_j]^-$ define the external input of the system. They correspond to a feedback term and allow the oscillator to be entrained at the same frequency of the input. The time constants τ_1 and τ_2 determine the natural frequency of the oscillator when no external input is applied. The frequency of the output is roughly proportional to $1/\tau_1$ and we set τ_1 equals $2\tau_2$. Note that in our case, the external input will not be used here as we are dealing with an open-loop CPG model.

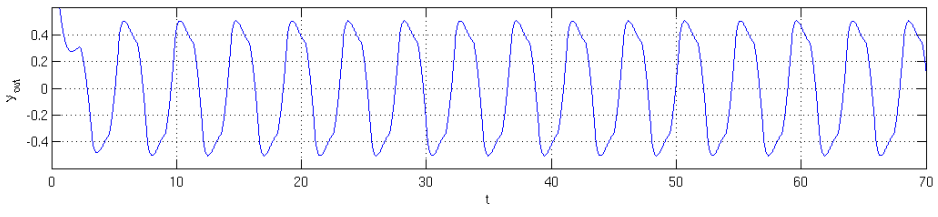

Figure 5.10: Typical output of the Matsuoka oscillator when no external input is applied. The parameters used here are: c = 1.0, $\beta = 2.0$, $\mu = 2.0$, $\tau_1 = 0.5$ and $\tau_2 = 1.0$.

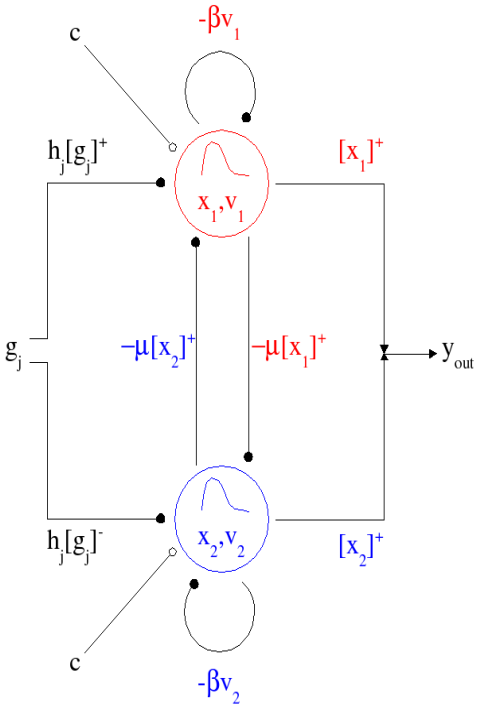

Figure 5.11: The two coupled neurons of the Matsuoka oscillator. The black circles correspond to inhibitory connections and white circles to excitatory connections. The self-inhibition is governed by the βv_i connections while mutual inhibition is done through the $\mu[x_i]^+$ connections.

The coupling

The coupling model used for that oscillator is once again an unidirectional coupling (cf. fig 5.13). This model has been studied by M. Williamson [19] and seems to be well-suited to our needs. The coupling is done by adding coupling terms to the equations of x_1' and x_2' as described in equations 5.23 to 5.27.
With this model it is also possible to introduce a phase difference between two coupled oscillators. The γ_i parameter can be used to achieve phase differences of 0, $\pi/2$ and π as described in table 5.2.
We also set the phase differences between the different servos by hand. These values are shown in figure 5.12.

value of γ_i	phase difference
$[0.0, 0.3]$	π
$[0.3, 0.5]$	$-\pi/2$
$[0.5, 0.7]$	$\pi/2$
$[0.7, 1.0]$	0

Table 5.2: The phase differences produced by changing the γ_i coefficient. Notice that for values of γ_1 in [0.3, 0.7] the phase difference is not precisely equal to $\pm\pi/2$ and varies for some intermediate values of γ_1 in this interval. However the GA should find the best values.

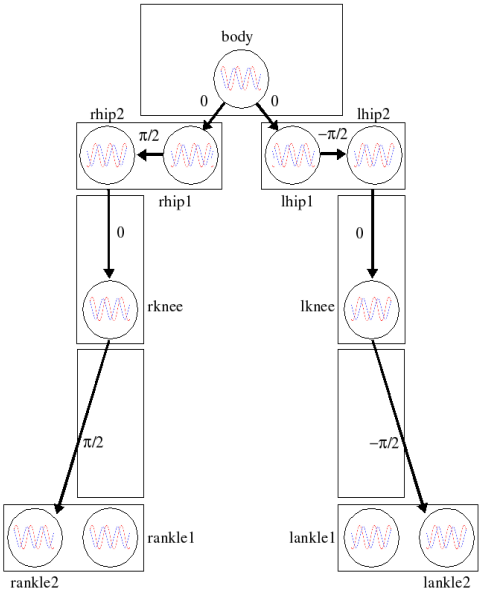

Figure 5.12: Couplings between the servos for the model based on Matsuoka oscillators. The phase differences we imposed between the different servos is shown on the arrows.

Notice that, with this model, the signals for the ankle1 servo are not evolved but are computed to be roughly parallel to the ground (cf. figure 5.12). The servo position then depends on the hip2 and knee joints' angle and the ankle2 servo is thus coupled directly to the knee's oscillator. We also fixed some parameter values (the central position of hip1 and ankle2 servos were set to 0).

$$
\begin{aligned}
\tau_1 \dot{x}_{1,i} &= c_i - x_{1,i} - \beta v_{1,i} - \mu[x_{2,i}]^+ - \\
&\quad - \alpha_{i,i+1}\gamma_i[x_{1,i+1}]^+ - \alpha_{i,i+1}(1-\gamma_i)[x_{2,i+1}]^+ && (5.22)\\
\tau_2 \dot{v}_{1,i} &= [x_1]^+ - v_{1,i} && (5.23)\\
\tau_1 \dot{x}_{2,i} &= c_i - x_{2,i} - \beta v_{2,i} - \mu[x_{1,i}]^+ - \\
&\quad - \alpha_{i,i+1}\gamma_i[x_{2,i+1}]^+ - \alpha_{i,i+1}(1-\gamma_i)[x_{1,i+1}]^+ && (5.24)\\
\tau_2 \dot{v}_{2,i} &= [x_{2,i}]^+ - v_{2,i} && (5.25)\\
y_{k,i} &= [x_{k,i}]^+ = max(x_{k,i}, 0) && (5.26)\\
y_{out,i} &= y_{1,i} - y_{2,i} && (5.27)
\end{aligned}
$$

Where $\alpha_{i,i+1}$ corresponds to the global coupling strength between oscillator i and i+1 and γ_i controls the relative coupling of each neuron.

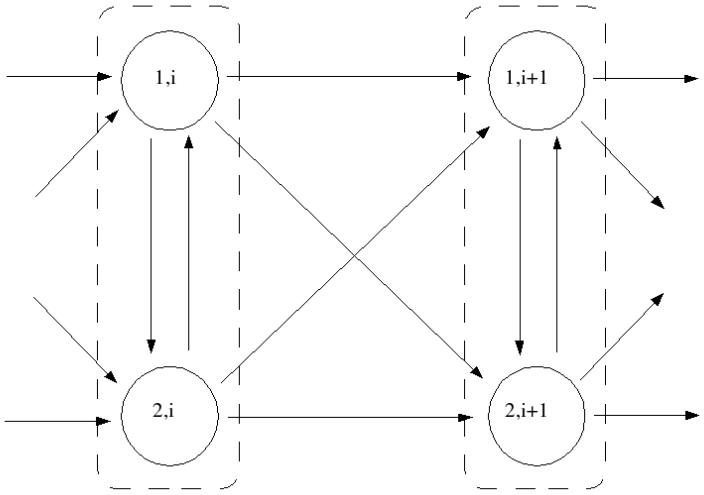

Figure 5.13: Coupling between two Matsuoka oscillators. They are only coupled in one direction (from 1 to 2). Each neuron of 1 (1,i and 2,1) is connected to both neurons of 2 (1,i+1 and 2,i+1). The relative strength of coupling 1,i → 1,i+1 against coupling 1,i → 2,i+1 can be modified through parameter γ_i.

5.2 Closed-loop

We will now introduce a more complex CPG model based on the closed-loop concept. As described above, this concept add the notion of feedback to the open-loop model. Feedback can be integrated at different levels of the robot depending of the task and the environment of the robot. (cf. section 3.2.3).

The results of certain experiments described above with the open-loop CPG model will be used here as starting point. We will try to integrate different feedback pathways to the best trajectories found in the first part of this project.

5.2.1 Speed adaptation and pendulum effect compensation

The feedback pathway we used is designed to guarantee the stability of the tilt of the robot in the sagital plane. This feedback pathway is very useful when one tries to modify the robots' speed because most of the time this results with the robot to fall in the direction of the walking. To achieve that, we will try to modify the amplitude of the signal of certain servos in order to compensate the variation of the angle of tilt ξ_{Tilt}.This angle is measured by the GPS function of the simulated robot which is located in its chest (the real robot uses gyroscopes to determine these angles). The feedback is added to the oscillators representing the hip2 and knee joints by introducing feedback terms to the appropriate equations as described below.
To introduce feedback in the Matsuoka oscillator based model, we simply add a feedback term to the two equations x'_1 and x'_2 of the oscillator responsible for the hip2 and knee servo as described in the following equations.

$$\begin{aligned}
\tau_1 \dot{x}_{1,i} &= c_i - x_{1,i} - \beta v_{1,i} - \mu[x_{2,i}]^+ - \\
&\quad - \alpha_{i,i+1}\gamma_i[x_{1,i+1}]^+ - \alpha_{i,i+1}(1-\gamma_i)[x_{2,i+1}]^+ + \\
&\quad + K_i\xi_{tilt} && (5.28)\\
\tau_2 \dot{v}_{1,i} &= [x_1]^+ - v_{1,i} && (5.29)\\
\tau_1 \dot{x}_{2,i} &= c_i - x_{2,i} - \beta v_{2,i} - \mu[x_{1,i}]^+ - \\
&\quad - \alpha_{i,i+1}\gamma_i[x_{2,i+1}]^+ - \alpha_{i,i+1}(1-\gamma_i)[x_{1,i+1}]^+ + \\
&\quad + K_i\xi_{tilt} && (5.30)\\
\tau_2 \dot{v}_{2,i} &= [x_{2,i}]^+ - v_{2,i} && (5.31)\\
y_{k,i} &= [x_{k,i}]^+ = max(x_{k,i}, 0) && (5.32)\\
y_{out,i} &= y_{1,i} - y_{2,i} && (5.33)
\end{aligned}$$

where K_i is the gain and is null for each oscillator except hip2 and knee.

Chapter 6

The genetic algorithm

Artificial evolution and in particular genetic algorithms (GA) are optimization methods directly inspired on the evolution of species. They are used in a lot of different domains for their ability to deal with high dimensional space problems. GA are based on the concept of exploration and exploitation. This means that the algorithm tries to explore the search space to find the most interesting parts and then tries to exploit those regions to find the extrema more precisely. Notice that these algorithms do not guarantee to find the global optimum of the problem and usually rather converge to local optima.
The first step a GA does is to create a randomly distributed initial population of individuals (in our case CPGs) that contain all the parameters we want to evolve. Then it evaluates these individuals to obtain a score for each of them, which is called fitness and corresponds to a specific phenotype of the individual. The fitness function describes how well an individual behaves (in our case we test its ability to walk). In the next step of the algorithm, the best individuals are selected and kept for the next generation. Before joining the new generation's population, operators like mutation and crossing-over are applied to these individuals to ensure genotypical diversity. All these steps are then iterated (except the initialization phase) until an ending criterion is satisfied.
To implement this algorithm we use Galib which is a free C/C++ library for designing genetic algorithms [22].

6.1 The genome

In this section, we will describe the different parameters that are evolved with the GA for each model described above. For each gene, according to the expected values of the parameters, we set by hand an interval of possible values to reduce the solution space.

6.1.1 Hopf model

1-way coupling

ω	μ_1	...	μ_6	x_{0_1}	...	x_{0_6}	ϵ_1	...	ϵ_5

Table 6.1: The genome of an individual with a simple unidirectional coupling. We have 11 DOF. For each of them we have 2 parameters corresponding to the amplitude μ and bias x_0. We also have 5 parameters for the coupling strength ϵ and one for the frequency ω, which is the same for every oscillator. This gives us a total of 18 parameters that correspond to the 18 genes (represented as float) of the genome.

2-way coupling

ω	μ_1	...	μ_6	x_{0_1}	...	x_{0_6}	ϵ_{12}	...	ϵ_{54}

Table 6.2: The genome of an individual with coupling in both directions (Hopf). The genome is the same as the one described above except that we added five genes corresponding to the coupling strength of connection going in reverse (from feet to hips)

2 oscillators per DOF

In this case the parameters that are evolved are different since some of them are taken from previous experiments. We will only try to optimize the weights of the sum and the coupling strength as described in section 5.1.1. The following table shows the genome used with that model.

d_1	...	d_6	$h_{1,1}$	...	$h_{2,6}$	ϵ_1	...	ϵ_5

Table 6.3: The 23 genes genome of an individual for the model with 2 oscillators for each DOF. Some parameters are kept from previous experiments and are not evolved. We have 6 genes for the phase relation between two oscillators within a same DOF, 12 genes for the weights of the output sum of each DOF and 5 genes governing the coupling strength between two consecutive pairs of oscillators.

6.1.2 Rayleigh model

The coding of the genome for the Rayleigh oscillator is rather similar to the one used with Hopf (1-way coupling). The following table describes the genome representation.

ω	q_1	...	q_6	x_{0_1}	...	x_{0_6}	ϵ_1	...	ϵ_5

Table 6.4: The genome encoding the parameters of coupled Rayleigh oscillators. It is similar to the one used with the Hopf based model with 1-way coupling, except the parameters q_i are responsible for the amplitude of the signals.

6.1.3 Matsuoka model

We present here the coding of the genome for the parameters used with the model based on Matsuoka oscillator. The role of the different parameters are described in section 5.1.3.

$\tau_{1,i}$	c_1	...	c_5	x_{0_1}	...	x_{0_3}	γ_1	...	γ_3	α_1	...	α_4

Table 6.5: The genome encoding the parameters of coupled Matsuoka oscillators. Notice that, as τ_2 is set to $2\tau_1$, we will not evolve this parameter.

6.2 The algorithm

In this work we use a steady-state algorithm, which means that the population of two consecutive generations is overlapping.

6.3 The fitness function

The fitness function is certainly the crucial part of the GA as it defines the goal our individuals will try to reach. The aim of that experiment is to make the robot walk and not fall. So the fitness function takes into account two abilities of the robot: the distance it reaches and the time before he falls. The fitness function is described as follow

$$f = D_z * (\frac{T}{TotalTime}) \tag{6.1}$$

where D_z corresponds to the distance (in the z direction) reached by the robot before falling and T is the time before it falls. With this fitness we will avoid the robot just to stand up without going forward or just fall as far as it can without trying to walk. If the robot does not fall, the simulation stops after a certain time. Also notice that the simulation starts later than the oscillators to be sure that all of them are stable.

6.4 The GA parameters

The intrinsic parameters of the GA are clearly not the most important part of this experiment. Therefore we used very common values indicated in the following table. As stopping criterion we set the maximum number of generations to 150, but for some experiments we ended the algorithm manually when the fitness no longer increased.

parameter	value
population size	100
maxnumber of generation	200
probability of mutation	0.25
probability of crossing over	0.9
proportion of replacement	0.5
selection scheme	Roulette Wheel

Table 6.6: Parameters of the genetic algorithm

Notice that, as we are dealing with real numbers genes, we use a Gaussian mutation operator. The crossing-over operator is only allowed to select a crossing-over point between two genes to make sure that the genome remains coherent. The selection scheme used here is the roulette wheel selector, which means that the probability of an individual to be kept for the next generation is proportional to its fitness.

Chapter 7

Analysis of the results

In this chapter, the different results found with each model will be presented and compared. As genetic algorithms does not guarantee to find global optima, the solutions proposed here might not be the best ones. Therefore several simulations were performed with different initial populations to try to explore the space search as widely as possible. For each model, we expose the solution that seemed to be the most efficient and realistic. Notice that, with some models, the GA did not manage to find any good solution in a reasonable time.

7.1 Open-loop

7.1.1 Hopf's oscillator

1-way coupling

After having tuned the intervals of possible values for the CPGs' parameters, quite fast convergence was achieved. A good solution was found after around 120 generations. Note that we had to run the algorithm several times to find a good solution as most of the simulations led to the robot to fall after a few steps. This shows us that the GA was far from exploring the whole solution space.

With that model, the robot managed to walk for about 30 steps before falling. The robot did not walk in straight line, it was significantly turning on its left. As the step amplitude was quite large, the robot's direction was slightly modified by each step. The contact of the feet and the ground was also not very realistic, because it was mostly done on the feet edges. The walking speed is approximately 0.25 m/s (cf. figure 7.2). A video of the robot walking is available on the web page of this project[1].

[1] birg.epfl.ch/page58711.html

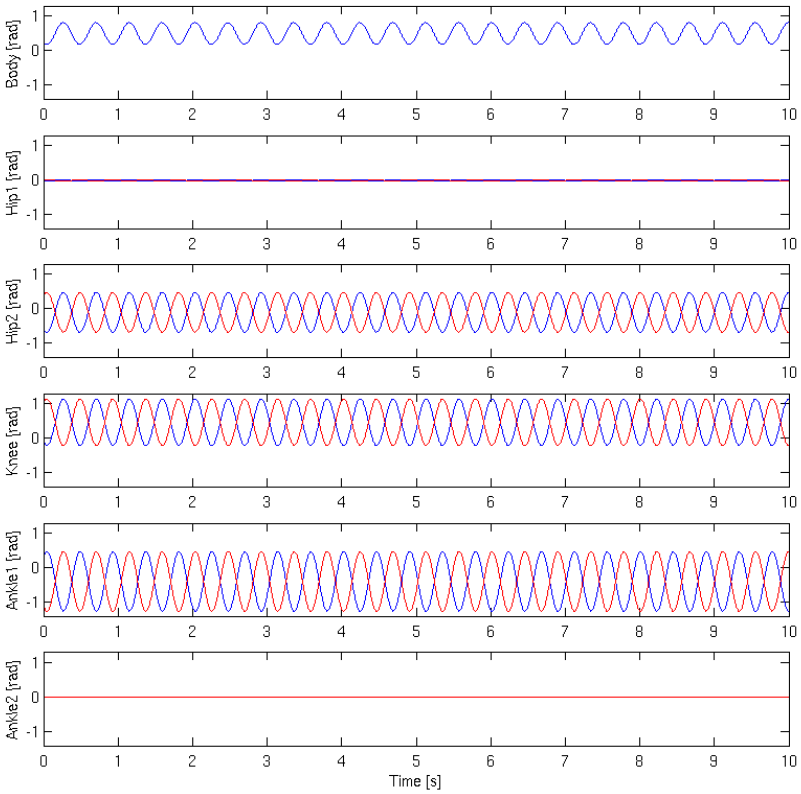

Figure 7.1: Output of the CPG of the best individual found by evolution for the model based on Hopf oscillators with unidirectional coupling. The blue curves correspond to the left leg and the red ones to the right leg. This plot only starts when the simulation begins (the oscillators are started earlier).

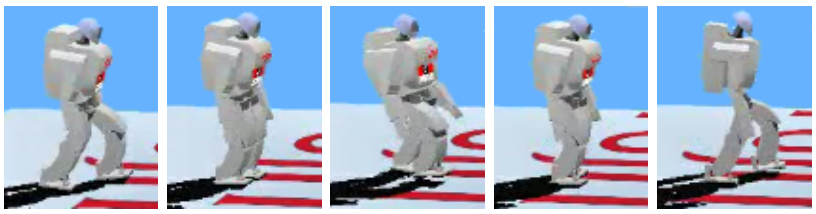

Figure 7.2: The robot is walking at 0.25 m/s.

The first thing we noticed is that the oscillators reached their limit cycle in approximately 6 periods from any initial conditions. As imposed the left and right leg were in phase opposition. Another important result, was that the resulting amplitude of the oscillators was pretty different from the natural amplitude because of the strong influence of the coupling. As we only introduced phase difference of 0 and π, the symmetry of the system (left/right leg) was not really correct since the body's servo was positive when the left leg was ahead and negative when the right leg was ahead. We also noticed that the servos that should control the lateral stability (hip1 and ankle2) played almost no role in locomotion (cf. figure 7.1).
With this model, we managed to make the robot walk for a few steps, but the solution found was not stable at all and the locomotion is clearly far from a realistic walking gait, because of the too simple phase differences used. To make a more realistic system, phase difference of $\pi/2$ and $-\pi/2$ should have been used.

2-way coupling

With this model we were able to impose phase differences of $\pi/2$ and $-\pi/2$, which should lead to a more realistic gait than with the previous model. As explained above, five extra parameters were added. The search space was then significantly enlarged and the impact on the evolution was negative: we did not observe any convergence within a reasonable time.
The best result allowed the robot to perform only two or three steps before falling. As the evolution for 200 generations took about 12 hours of simulation, we did not perform simulation any further with that model. The results obtained will not be analyzed as they are clearly not relevant enough.

2 oscillators per DOF

By adding a second oscillator on each joint, much more complex signals were generated and a large variety of signal shape could then be used. Unfortunately, the genetic algorithm was unable to find any good solution, even if there were less parameters to evolve. One possible explanation of this fact, is that the experiment was based on the best trajectory found with the 1-way Hopf model. As mentioned above, this trajectory does not reflect real bipedal locomotion in terms of phase differences between the different parts of the leg. This trajectory is certainly not robust enough to be modified as it was done by adding a second oscillator.

7.1.2 Rayleigh's oscillator

As explained above, the next system studied is based on relaxation oscillator and in particular the Rayleigh oscillator.

As expected, signals with a different shape from a simple sinus were produced but just like with the 2-way Hopf model. We ran several simulations of the GA and none of them seemed to converge significantly even after 200 generations. The result on the simulated robot was that it never managed to do more than one step.
The possible explanation of such bad performances of the evolution is that the possible intervals for the oscillators' parameters (mainly q_i that controls the amplitude) are much too large. To generate signal amplitude going from 0 to 1, we needed to set an interval for the parameter q roughly comprised between 0 and 10. This caused to enlarge the solution space consequently. In comparison with the Hopf oscillator, the interval for the parameter responsible for the amplitude is only approximately comprised between 0 and 1 for the same range of amplitude of the resulting signal.

7.1.3 Matsuoka's oscillator

Just like with the other models, we tuned the interval of possible values for the CPG's parameters by hand. We achieved a relatively fast convergence since the best individuals were found after approximately 60 generations as shown on figure 7.3. It was not useful to go on with more generations, as most of the time the fitness was the same, even after 200 generations.

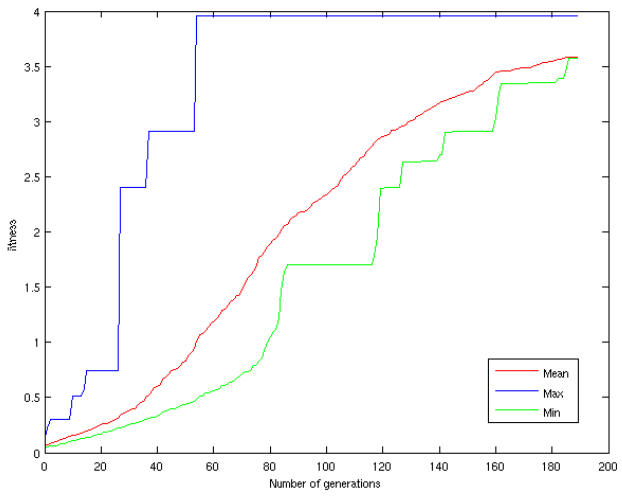

Figure 7.3: Evolution of the fitness function through generations with the genetic algorithm described above.

The best trajectories found with this model were significantly different from those found with Hopf oscillators. The steps were a bit smaller and

allowed the robot to walk in straight line. We managed to make the robot walk at approximately 0.28 m/s (cf. figure 7.4). We also noticed that the contact between the robot's feet and the ground was not perfect due to the tilt of the whole robot. Evolution showed that the Ankle2 servo was a critical part as the contact between feet and ground plays an important role. Computing the position of this servo to make the feet remaining parallel to the ground was then certainly an important modification in terms of stability. Videos of the robot walking are available on the web page of this project[2].

Concerning the CPG, we observed that it stabilized slowlier than with the Hopf model (10-15 periods). As shown on fig 7.5, the shape of the signals generated by the CPG are different from a simple sinus due to the mutual and self inhibition of the neurons. On this plot we can also see a small phase shift between the different oscillators along the leg.
According to the amplitude of the different servos, we can see that the Hip1 joint is more implied in lateral stability than the Ankle2 joint and is therefore an important DOF to produce a lateral movement that helps the legs going from behind to front without touching the ground.
An important result is that the oscillator responsible for the body servo plays no role in this solution as its amplitude is null. This might be a problem since the left an right hip1 servos are coupled to the body servo. It means that these two oscillators will not be synchronized with the body oscillator because it does not oscillate. However, the solution found by evolution works because the two hip1 oscillators have the same amplitude, initial conditions and are in phase. But as soon as there will be perturbations in the system, which could happen when introducing feedback, the two hip1 oscillators will not stay in phase any longer. To deal with that problem, some modifications were done on the model in order to keep it robust against perturbations when adding feedback pathways. These modifications are described in 7.2.

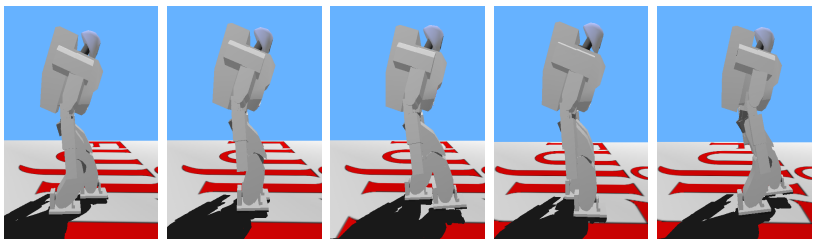

Figure 7.4: The robot is walking at 0.28 m/s.

[2]birg.epfl.ch/page58711.html

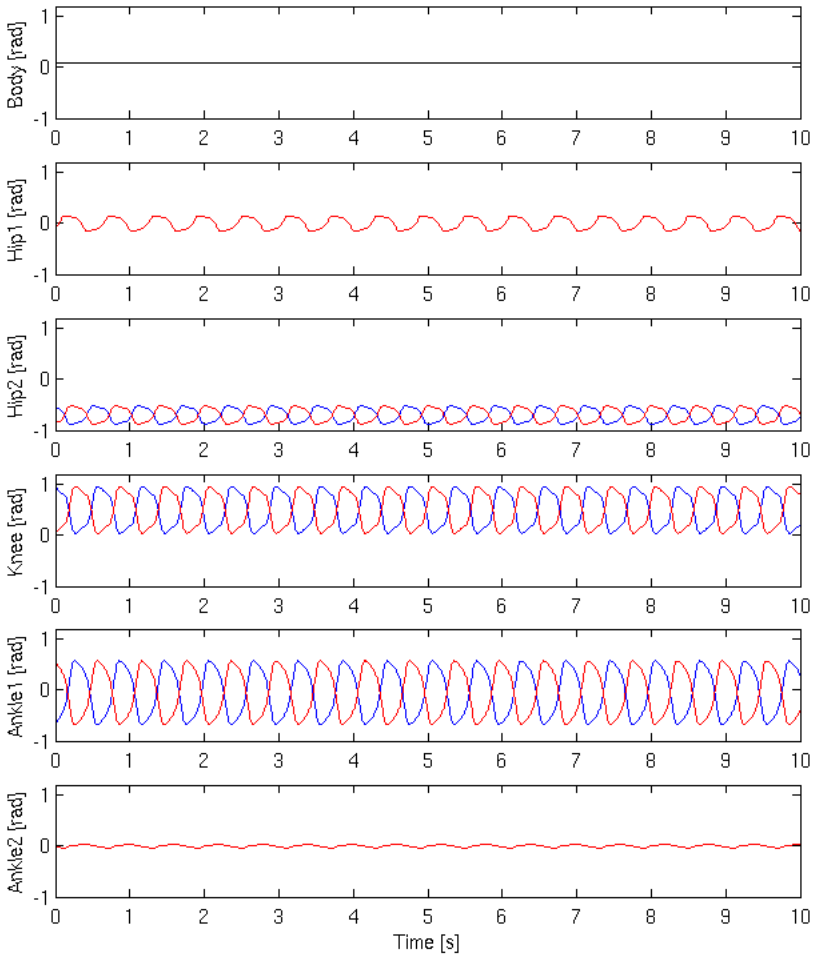

Figure 7.5: The 11 CPG's outputs of the best individual found by evolution. The blue curves correspond to the left leg and the red ones to the right leg. This plot only starts when the simulation begins (the oscillators are started earlier).

The considerations mentioned above only deal with the output of the CPG but not with the real movements produced by the servos during simulation. Figure 7.6 clearly shows a gap between the desired and measured values. Hence, the amplitude of the measured signals are much smaller than the desired ones and there is small phase delay between the two curves. These differences occur because the servos' gains implemented in Webots are too low and the signal frequency and amplitude we tried to impose

were too important. The shape of the measured signal is also different from the desire one, hence the trajectories look more like simple sinuses. We also noticed, on figure 7.6, there were small perturbations on the measured trajectory of the hip2 servos. These perturbations occurred when the feet touched the ground because the friction with the ground created a small resistance for the actuators. We will discuss about the servos' behavior under different frequencies more in details in the next section. Also notice that, at the beginning of the simulation, the positions of the desired and measured positions are very different since the initial conditions of the oscillators were not set to fit the initial position of the robot.

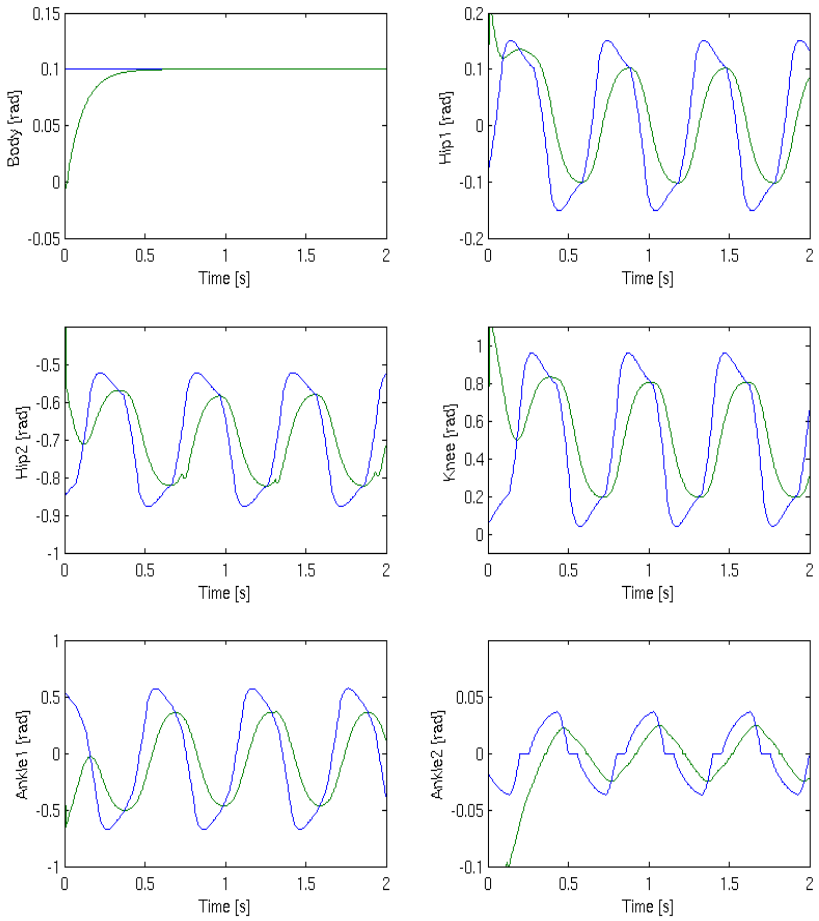

Figure 7.6: Deviation between the signals we try to impose to the servos (blue line) and the measured movements of the actuators (green line) during simulation.

7.2 Closed-loop

This section describes the results found when adding feedback pathways to the open-loop model that gave the best results, as described in section 5.2. We did not try to add feedback to models that use Hopf oscillators and to the model based on Rayleigh oscillators and focus on the model using Matsuoka oscillators since it gave by far the best results in an open-loop approach.
To find the optimal parameters for the gain of the feedback, we made an exhaustive search for the two parameters K_{knee} and K_{hip} (cf. figure 7.7) with different values of the time constant τ_1, which is inversely proportional to the frequency of the oscillators output. The results of this search are described in this section.
In addition, we tried to test the robustness of our closed-loop model against small perturbations.

7.2.1 Speed adaptation

To modify the walking velocity of the robot, the global frequency of the oscillators was increased. Starting from the best trajectories found with the open-loop approach, the frequency was modified after three steps of the robots. The frequency was not changed at the beginning of the simulation because it had too much impact on the initial conditions. With some frequencies the robot started to walk with the wrong leg. Changing the value of the frequency while the robot was already walking was then a good solution since the aim of this experiment is to modify the walking speed online. Notice that the best open-loop model found was slightly modified for the purpose of this experiment. First tests of that feedback model showed that the feedback pathways generated a small change of the oscillators' frequencies. The consequence of that phenomenon was that the oscillators directly or indirectly (through coupling) implied in feedback pathways (hip2, knee, ankle1 and ankle2) were no more synchronized with the oscillators that are not concerned by feedback pathways (body and hip1). In order to avoid this frequency modification, the coupling between the hip1 and hip2 servos and between the hip2 and knee servos was reinforced (by increasing the coupling strength) to keep the whole system synchronized. More precisely, every oscillator involved in the feedback pathways were synchronized with the hip1 oscillator. This could have been done by changing only the coupling strength between the hip1 and hip2 oscillators. But as the gains g_{hip} and g_{knee} are different, the modification of the hip2 and knee frequency was also different and thus prevented these two oscillators from synchronizing. By reinforcing the coupling between these servos, the knee oscillator was then synchronized with the hip2 oscillator. Also notice that, with the new model, a small phase shift could be seen in the oscillators involved in feedback path-

ways (cf. figure 7.9). This slightly changed the phase relation between the hip1 oscillator and the other oscillators. This phase shift is too small to have a real impact on the locomotion since the output of the hip1 oscillator was still close to its maximum when both legs are parallel. This is important since it allowed the swing leg to go from back to front without hitting the ground.
As explained above (cf. section 7.1.3), the global coupling model had to be modified because the two legs were not synchronized since the body oscillator's amplitude is null. To deal with that problem, the same oscillator was used for the left and right hip1 servos since, in our open-loop model, they are forced to have both the same amplitude, frequency and phase. To ensure that the both legs are synchronized and in phase opposition, phase differences of $\pi/2$ and $-\pi/2$ between the unique hip1 servo and the left and right hip2 servos were respectively imposed. The drawback of this modification was that the left and right hip1 joints, that mainly control the lateral stability of the robot, always generated the same movements. This could be a limitation since it could be useful to have two distinct signal, for example, if a feedback pathway controlling the lateral stability is added to the system.
As the systematic search for system's gains for one given frequency took approximately 8 hours, we decremented the frequency by steps of 5% of the initial value of $\tau_{initial}$ found with the open-loop model. Figure 7.7 shows the result of this search

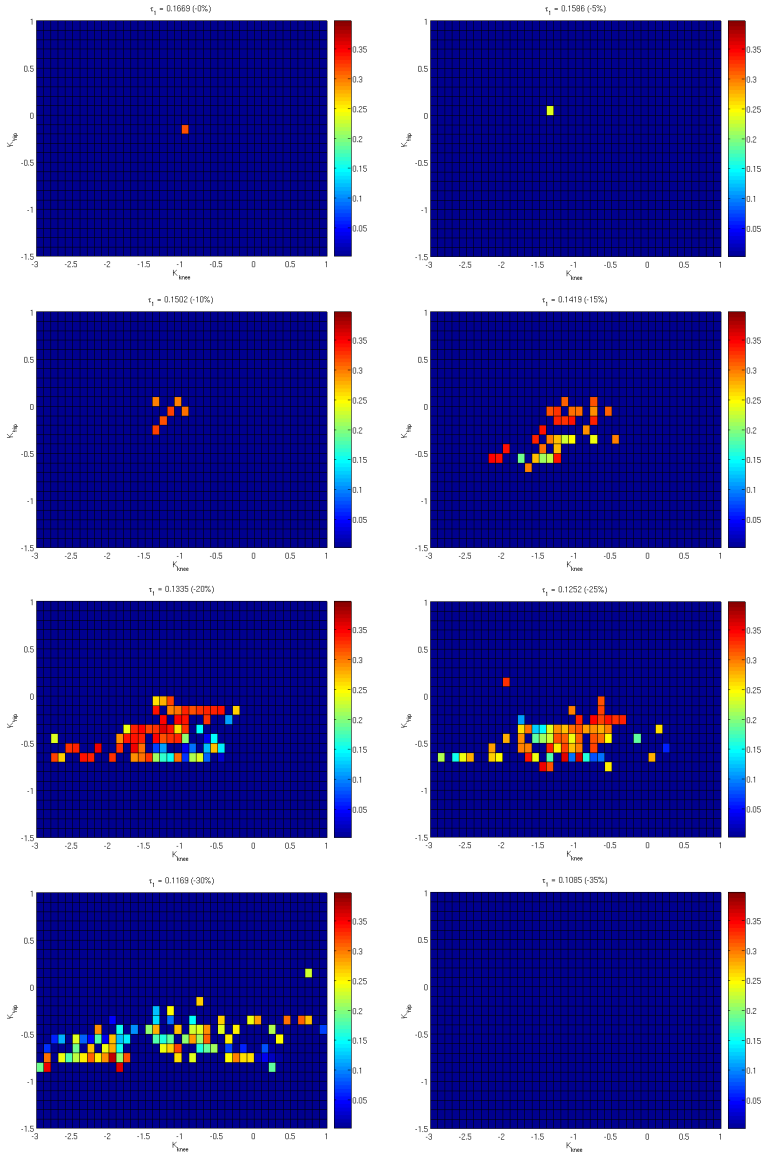

Figure 7.7: Velocity of the robot with different gains. Each plot corresponds to a different frequency. Only individuals that do not fall during the simulation are considered. The speed of the other ones is set to 0.

The gains giving the fastest locomotion for each frequencies are summarized in table 7.1.

τ_1	frequency [Hz]	K_{hip}	K_{knee}	velocity [m/s]
0.16698 (-0%)	1.66	-0.2	-1.0	0.31
0.15863 (-5%)	1.76	0.0	-1.4	0.23
0.15028 (-10%)	1.85	-0.3	-1.4	0.32
0.14193 (-15%)	1.96	-0.6	-2.2	0.35
0.13358 (-20%)	2.12	-0.4	-1.3	0.36
0.12523 (-25%)	2.23	-0.7	-1.0	0.36
0.11688 (-30%)	2.40	-0.8	-2.0	0.37

Table 7.1: Best gains found for different frequencies and the corresponding speeds. Notice that τ_1 is inversely proportional to the frequency of the oscillator's output. Typical signals produced by the oscillators with feedback can be seen on figure 7.8.

With the systematic search described above, we found optimal gains for different values of the global frequency of the system. As shown on figure 7.7, we noticed that, up to 10% of frequency diminution, there were only a few values for the gains that made the robot walk without falling. From 15% to 20% of diminution, we found much more gains that produced stable locomotion. These values were also roughly distributed within a limited range of the gain space we explored. A possible explanation is that, as we had to change our system when we added feedback pathways, the modified model could be better adapted to a range of frequencies going approximately from 2 Hz to 2.4 Hz. On the contrary, the model used with the open-loop approach seemed to work better under lower frequency. Also notice that over 2.4 Hz, we did not find any values for the gains that did not make the robot fall. As explained, we found values for the gains at given frequencies, but unfortunately we did not manage to find a monotone function that would link the values of K_{hip} and K_{knee} to the frequency, in order to be able to set the frequency of the system to any value in a continuous interval. This would have allowed us to perform online modifications of the robot's velocity. We also tried to find gains when the frequency is decreased but even with a diminution of 5%, the robot always fell.
As shown in table 7.1, we managed to make the robot walk at a speed of approximately 0.37 m/s for a frequency of 2.4 Hz, which represents an increase of 24% of the velocity found without using feedback pathways.

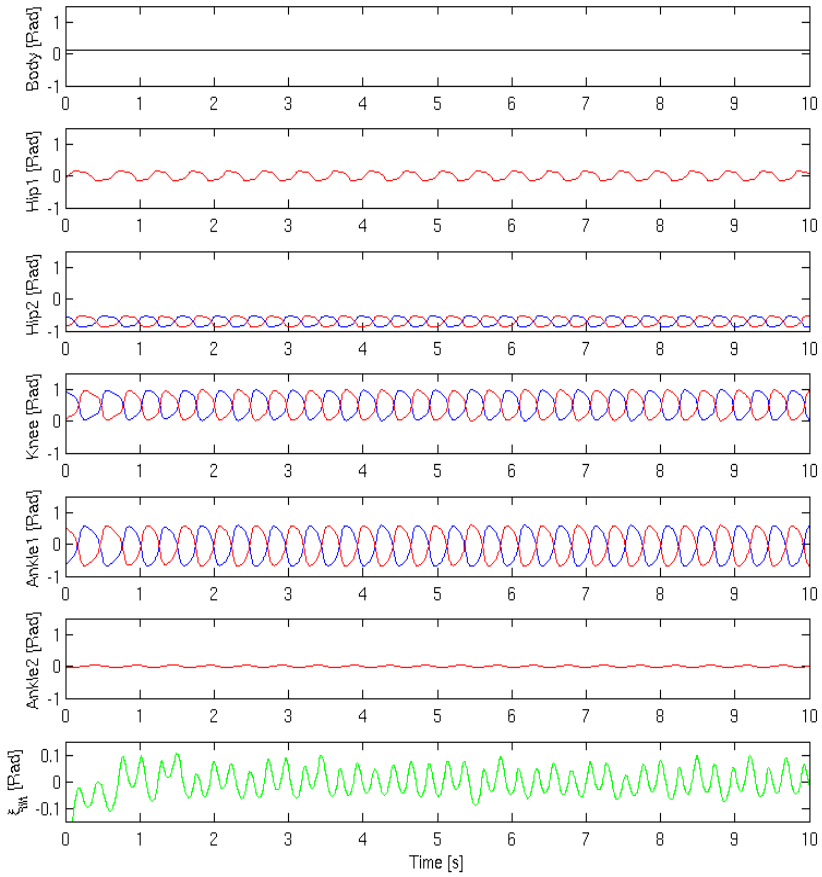

Figure 7.8: Signals sent to the different actuators when adding feedback to the model based on Matsuoka oscillators. The parameters are $\tau_1 = 0.11688$ (frequency = 2,4 Hz), $K_{hip} = -0.8$ and $K_{knee} = -2.0$. The bottom plot shows the tilt of the robot in the sagital plane (ξ_{tilt}).

To analyze the effect of the feedback, we plotted the signals generated by the oscillators during time (cf. figure 7.9). The first observation was that, as expected, the body behaved like an inverted pendulum, its frequency being half the natural frequency of the oscillators (cf. figure 7.9 (bottom right plot)), since the tilt angle is maximum each time the feet touched the ground. During the first 3-4 steps after the change of frequency, the tilt angle of the robot's body varied irregularly because perturbations were created by the sudden modification of the frequency, which often changed the robot's direction. After a few steps, the feedback pathways provided a global stabilization of the robot's tilt. The tilt angle variation became more regular,

which made the robot walk in straight line. The effect of the feedback on the signals produced by the oscillators emphasized a modification of the trajectories amplitude. This was an expected consequence since we simply added a proportional term to the oscillators. As all the best gains K_{hip} and K_{knee} found were negative, the amplitude was increased when the tilt angle was negative and decreased when the tilt angle was positive, which actually produced a modification of the step length. As explained above, the feedback we used also generated a small phase shift. Also notice that, as the ankle1 servo's position was computed from the position of the hip2 and the knee joints, its value was also modified by the feedback and, as expected, the feet actually stayed roughly parallel to the ground. The amplitude of the ankle2 oscillator is also modified since this joint is coupled to the rest of the system. Figure 7.10 show a typical example of the compensation done by the feedback pathways in order to prevent the robot from falling.

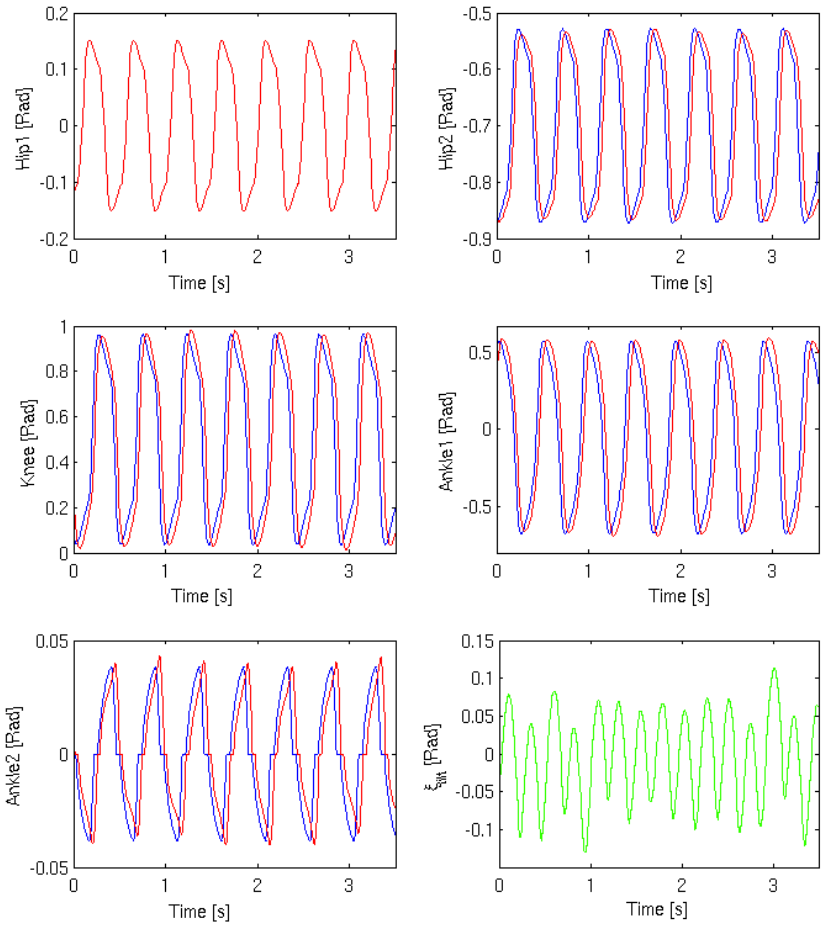

Figure 7.9: Signals produced by the oscillators with (red) and without (blue) feedback. The feedback pathways slightly modified the amplitude of the signals. A small phase difference is also produced by the feedback.

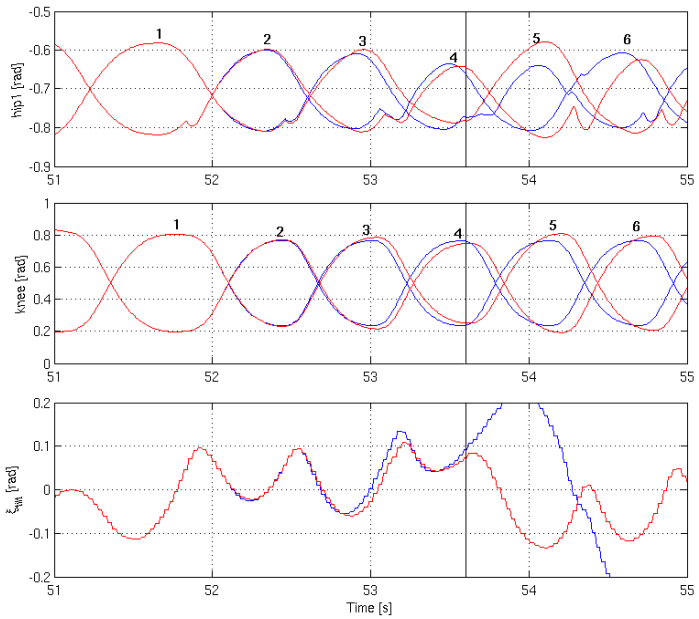

Figure 7.10: Measured values of the hip2 and knee servos (left and right legs) after the modification of frequency (-20%). The frequency is increased between step 1 and 2. The bottom plot corresponds to the value of tilt angle of the robot in the sagital plane. The red curves represent the trajectories with feedback ($K_{hip} = -0.4$ and $K_{knee} = -1.3$) and the blue ones correspond to model with both gains k_{hip} and k_{knee} set to 0. The vertical line corresponds to the fall of the robot.

Figure 7.10 showed us that, as soon as we modified the frequency, the robot started to lean forward, significantly increasing the mean value of the tilt angle. The consequence on locomotion was that the feet started to hit the ground earlier, causing bigger perturbations on the measured trajectories of the hip1 servos (steps 3 and 4) and the fall of the robot. With feedback, we observed that, as the robot's tilt angle increased (beginning of step 4), the length of step 4 decreased and thus prevented the foot from hitting the ground too abruptly. After that (step 5) the robot tilted backward and the step length increased. After 3 or 4 more steps the robot's tilt was stabilized and it continued walking normally. Also notice that the hip2 servo is much more involved in the stabilization process than the knee servo. The impact of the feedback applied on the knee servo is less obvious, but by removing it, the robot always fell. Servos that were indirectly influenced by the feedback (ankle1 and ankle2) also produced modified trajectories, but the impact on stabilization did not seem to be really obvious. On these plots we can also see that the feedback produced a small phase shift, which became constant after 6 or 7 steps.

As explained above, there was a gap between the desired positions of the servo and the measured ones. As the actuators do not manage to reach the desired amplitude, one could wonder if the small amplitude modifications created by adding feedback would produce a modification on the measured signals. Figure 7.11 shows small variations of the measured trajectories' amplitude, which means that there is an impact of the feedback pathways on the real positions of the servos. By consequence, the behavior of the robot with feedback as described above remains valid even if there is a deviation between desired and measured trajectories.

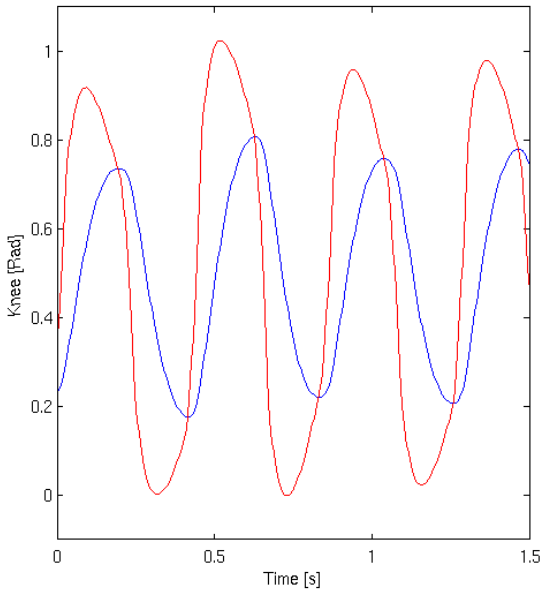

Figure 7.11: Deviation between the signals sent to the knee servo and the real trajectory measured during simulation. We observed that small amplitude variations, produced by the feedback pathway, on the desired trajectory also modified the amplitude of the measured signal. This result was also observed on the other joints involved in feedback pathways.

7.2.2 Resistance against perturbations

To test our system's robustness, external perturbations were applied on the robot. As the feedback pathway introduced was designed to control the balance in the sagital plane, external forces were generated in the direction of locomotion on the body joint of the robot during a few milliseconds. Unfortunately, we did not manage to apply forces greater than 1-2 Newtons without making the robot fall, which is clearly insufficient to consider that the robot is robust against perturbations. The main problem when we applied greater forces was that the impulsion done on the robot produced a

too fast tilt of the robot in the direction of walking and the robot was unable to react quickly enough to the sudden modification of the body's tilt. By consequence, we concluded that the gains of the feedback pathways were too low for that kind of perturbation and a different model should be used to deal with both perturbations and speed adaptation.

Chapter 8

Conclusion

8.1 The project

The study of bipedal locomotion is quite a new field that is only a few ten years old. However, researches done on this subject show the huge amount of different possibilities there are to try to imitate the incredibly complex human gait.
Inspired from biological considerations, we tried, in this work, to explore and analyze a few models using oscillator based CPGs as controllers for a simulated biped robot. Artificial evolution was also used to optimize the parameters of the oscillators, in order to generate an efficient locomotion.
In the first part of this project, several models of oscillators were artificially evolved. We firstly used Hopf oscillators with different types of coupling between them and achieved a walk of about 30 steps with a velocity of approximately 0.25 m/s. As this solution was found with the simplest coupling model we tested, this first experiment also showed that it was difficult to make the genetic algorithm converge in a reasonable time with more complex models. Most of the time, extra parameters added for more sophisticated coupling models led to a important enlargement of the search space that our algorithm did not manage to explore efficiently. The same problem occurred when we used a relaxation oscillator. In this case, the range of possible parameters was also too large to see any interesting solution emerging from artificial evolution.
We also studied Matsuoka oscillators, which are based on the mutual inhibition of two coupled neurons and are often used for bio-inspired neuronal models. This model gave by far the best results and the genetic algorithm converged easily. The best solution found allowed the robot to walk indefinitely at a speed of approximately 0.28 m/s and with a rather realistic gait. This last model was then improved by adding feedback pathways in order to be able to increase the speed of the locomotion by changing the global frequency of the system. This was done by using the tilt angle of the body

in the sagital plane to modify the trajectories of the hip2 and knee servos. This feedback pathway proved to be quite efficient since we managed to increase the velocity of the robot of about 25% (0.37 m/s) by increasing the frequency by 30%. The main impact of the feedback on locomotion was that it modified the trajectories of the hip1 servos, preventing the robot from falling forward when the frequency is increased.
We also tried to apply external perturbations on the robot in the direction of walking. Unfortunately, the robot was only able to remain stable after very small perturbations (1-2 Newtons). In addition, this model did not allow us to modify the speed of the robot online by changing frequencies comprised in a continuous range.
In this work, we have demonstrated that a rather simple feedback pathway model clearly improved the locomotion when the frequency is increased. Finally, this project has opened the way to many possible upgrades of the model since we only explored a small part of the possibilities offered by bio-inspired artificial locomotion.

8.2 Further work

This work explored different model of oscillators and couplings giving various results. A lot of other oscillators could obviously be used, particularly oscillators able to generate more complex signals closer to human locomotion could be of interest. The Matsuoka oscillator gave good results and should be studied more deeply.
In this study only one kind of feedback pathways was used. It would be useful to implement more complex models like a feedback pathway that controls the lateral stability of the robot or a feedback using the touch sensors of the robot's feet. One could also try to deal with more complex perturbations such as lateral wind, slopes or irregular ground. A more difficult but obviously important development would be to be able to control the robot's direction during simulation. The solutions found in this work should also be tested on the real Hoap2 robot.

Bibliography

[1] Christopher L. Vaughan. Theories of bipedal walking: an odyssey. *Journal of Biomechanics*, Volume 36, Issue 5, November 2000.

[2] A. J. Ijspeert. Vertebrate locomotion. *The handbook of brain theory and neural networks*. 649:654. MIT Press. 2003.

[3] Jonas Buchli, Auke Jan Ijspeert. Distributed central pattern generator model for robotics application based on phase sensitivity analysis.*Biologically Inspired Approaches to Advanced Information Technology: First International Workshop*, BioADIT 2004, Lecture Notes in Computer Science, Springer, 2004

[4] McGeer T. Passive dynamic walking. *International Journal of Robotics Research*. 9(2):62-82. 1990.

[5] Collins S. H., Ruina, A. A bipedal walking robot with efficient and human-like gait. *In Proc. IEEE International Conference on Robotics and Automation*. 2005.

[6] F. Asano, M. Yamakita, N. Kamamichi, Z-W Luo. A novel gait generation for biped walking robots based on mechanical energy constraint. *IEEE transaction on robotics and automation*. vol. 20, no 3. 2004.

[7] M. Vukobratovic, B. Borovac, D. Surla and D. Stokic. Biped locomotion: dynamics, Stability, Control and Applications. Springer. 1990.

[8] J. Pratt, P. Dilworth, G. Pratt. Virtual model control of a bipedal walking robot. *Proceeding of the IEEE International Conference on Robotics & Automation*. 193-198. 1997.

[9] S. Grillner. Control of motion in bipeds, tetrapods and fish. In V. Brooks (Ed.), *Handbook of Phisiology, The Nervous System, 2, Motor Control* . 1179-1236. American Physiology Society. Bethesda. 1981.

[10] G. Taga. Emergence of bipedal locomotion through entrainment among the neuro-musculo-skeletal system and the environment. PHYSIKA D, 75:190-208. 1994

[11] G. Taga. A model of the neuro- musculo-skeletal system for anticipatory adjustment of human locomotion during obstacle avoidance. *Biological Cybernetics*. 78(1):9-17. 1998

[12] A. J. Ijspeert. A connectionist central pattern generator for the aquatic and terrestrial gaits of a simulated salamander.*Biological Cybernetics*. 84(5):331-348. 2001

[13] A. J. Ijspeert, J-M Cabelguen. Gait transition from swimming to walking: investigation of salamander locomotion control using nonlinear oscillators. *Proceeding of adaptive motion in animals and machines*. 2003

[14] A. J. Ijspeert, J. Hallam, D. Willshaw. Evolving swimming controllers for a simulated lamprey with inspiration from neurobiology. *Adaptive behavior*. 7(2):151-172. 1999.

[15] S. Strogatz. Non-linear Dynamics and Chaos, *Addison Wesley Publishing Company*. 1994.

[16] A. Pikovsky, M. Rosenblum, J. Kurths. Synchronization, A universal concept in non-linear sciences. *Cambridge University Press*. 2001.

[17] K. Matsuoka. Sustained oscillations generated by mutually inhibiting neurons with adaptation. *Biol. Cybern.*. 52:367-376. 1985

[18] K. Matsuoka. Mechanisms of frequency and pattern control in the neural rhythm generators. *Biol. Cybern.*. 56:345-353. 1987.

[19] M. M. Williamson. Robot arm control exploiting natural dynamics. *thesis*. Massachusetts Institute of Technology. 1999.

[20] WEBOTS. www.cyberbotics.com. Commercial Mobile Robot Simulation Software.

[21] P. Cominoli, Development of a physical simulation of a real humanoid robot. *Master's thesis*. 2004-2005.

[22] Galib. lancet.mit.edu/ga/. A C++ Library of Genetic Algorithm Components.

[23] Fujitsu HOAP 2. www.automation.fujitsu.com/en/products/products09.html. Humanoid Robot HOAP-2.