Grenoble INP – ENSIMAG
École Nationale Supérieure d'Informatique et de Mathématiques Appliquées

Rapport de projet de fin d'études

EPFL Biorobotics Lab, Switzerland

Rehabilitation robotics using Central Pattern Generators

Sarah MOUSSOUNI
3e année ENSIMAG – Option SLE
M2R MOSIG – Option DEMIPS

08 Février 2010 – 18 Juillet 2010

EPFL Biorobotics Laboratory
EPFL, Swiss Federal Institute of Technology
EPFL STI IBI BIOROB INN 237, Station 14
CH-1015 Lausanne, Switzerland

Professeur : Auke Jan IJSPEERT
Superviseur : Renaud Ronsse BioRob Lab
Mohamed Bouri LSRO Lab
Tuteur de l'école: Florence Maraninchi

Abstract

Rehabilitation robotics is an application of engineering to design and develop technological solutions for people suffering from movement disorders. Nowadays, the existing rehabilitation programs require many training sessions and sometimes up to three physiotherapists to move one patient only. Therefore, rehabilitation robotics is a promising research avenue to take over some of this time- and energy-consuming workload.

Since 2009, the BioRobotics Lab at EPFL (Switzerland) has started to investigate this field, targeting the development of novel rehabilitation methods using advanced control techniques As first goal, the lab focuses on the rehabilitation of locomotion (lower limb).

One of the control technique developed by the lab is based on adaptive oscillators, i.e. oscillators equipped with an adaptive mechanism which enables to adapt their intrinsic frequency to one of the frequency components of an input signal. This approach is thought to be of particular relevance in the field of locomotion rehabilitation, the artificial oscillator augmenting the biological Central Pattern Generator (CPG). CPGs are neuronal oscillators located in the spinal cord and producing coordinated multidimentional rhythmic signals, under the control of simple inputs.

This thesis aims at investigating the way to integrate these adaptive oscillators into innovative rehabilitation protocols. As initial validation, this method is implemented on a knee orthosis, developed at the Laboratoire des Systèmes Robotiques (EPFL, Switzerland). It is targeting the development of autonomous rehabilitation robots that can , for example, be used at home as an extension of standard therapy. Based on results obtained for upper limb assistance, our study pioneers the use of adaptive oscillators in lower limb rehabilitation.

In contrast to most standard protocols in rehabilitation robotics, where the reference movement is pre-specified by the therapist, this method permits the patient to choose the frequency and amplitude of the intended movement, out of any constraint. The frequency spectrum of the knee kinematics is learned by the coupled oscillators and makes the orthosis able to synchronize with the movement, providing augmentation.

To validate the proposed method, a first study is conducted on a model created with Simulink. This simulates the implementation in order to control and adjust the experimental settings. The second stage is an actual validation using the knee orthosis. The test is made with a target movement of increasing complexity: first, a single harmonic (sinusoidal), then more complex signals with more then one frequency component.

Key words: Rehabilitation robotics, lower limbs paresis, Adaptive frequency oscillators, Central Pattern Generator, knee orthosis

Acknowledgments

I want to express my deep gratitude to my supervisors: Mohamed BOURI and Renaud RONSSE. I learnt a lot through them, not only scientifically, but also in the way to be a researcher. This is something very precious that I will keep.

I would like to thank Prof. Auke J. Ijspeert for providing the opportunity to do my Master thesis. It was a real pleasure to spend these few months in his Lab.

Special thanks to my ENSIMAG supervisor Florence Maraninchi, for her advice, guidance and support.

Lastly, I thank my family, for their unconditional support and for providing a cheerful atmosphere with constant encouragement.

Contents

List of Figures

Chapter 1

Introduction

Rehabilitation robotics is a special branch of robotics which focuses on machines that are used to help people to recover from severe physical trauma.[1] There are many areas of physical therapy where robots can provide support to the patients. One of these areas is the rehabilitation of locomotion.

People suffering from walking deficiencies have better recovery expectancies using intensive rehabilitation program. However, in standard rehabilitation process, many physiotherapists often work with one patient to support limbs and help him/her to move at early stages. It is time and energy consuming with many limitations. Then, rehabilitation robotics tools have come out to overcome some of the physiotherapist's workload. Ref [4] [6]

These robots fall into two main classes: robots designed to compensate for lost skills, including manipulation, self-feeding, or mobility; and those developed to cure or retrain lost motor function after a disabling event such as stroke. Research on the effectiveness of robotic therapy has shown that rehabilitation technologies provide new options for repetitive movement training that can complement efforts to improve the therapy performance.

In that respect, our project goal is to investigate an innovative rehabilitation protocol for locomotion using existing rehabilitation robots. This protocol is based on the theory of Central Pattern Generators (CPG) and adaptive oscillators, which are non-linear oscillators capable to synchronize with any periodic input. They are currently developed at the Biorob Lab (EPFL). These oscillators can adapt their frequency without any signal processing or the need to specify a time window or similar free parameters. They have been used to model various biological processes and they are also used in engineering fields, such as autonomous robotics.

The major contribution of this thesis is to show that the method using adaptive oscillators reduces the metabolic cost of the human performer and this for a simple movement viewed as a sinusoidal signal to a multi-frequencies one.

[1] www.wisegeek.com

The idea is the following: each moving joint can be viewed as an oscillator that is synchronized with a second artificial oscillator, which in turn feeds some energy back to the user through the robotics interface. This method provides some augmentation to the user as a physiotherapist adds during the standard rehabilitation process. However, the robot leaves all high-level parameters up to the patients, such as the movement amplitude and frequency.

The rehabilitation robots used were developed at the LSRO Lab (EPFL). The first exploration was done on a simple knee active orthosis offering one degree of freedom (1 DOF). Later, this thesis will be the basis of a second exploration which will be done on the Lambda robot offering 3 degrees of freedom per leg. Ref[7]

In this document, we will discuss the following points: firstly, a short introduction of the project context and environment is provided. The following chapter presents the adaptive oscillator, its architecture and properties; and explains the innovation that can be made using this method. The next part of the report describes the theoretical studies about the kinematic and dynamical models of the lower limbs and the simulation made with Simulink platform to validate the method. The last chapter describes the preliminary work done on the knee-orthosis to test the rehabilitation method. Finally, the experimental results are provided with a discussion about the future work that will be done in the next months, as an extension of this thesis.

1.1 Context of the project

The following project is driven by a collaboration between the Biorobotics Laboratory (Biorob Lab) providing methods and daily guidance, and the Laboratory of Robotics Systems (LSRO) which provides robotics platforms and software support, both at EPFL, Switzerland.

My supervisors are Dr Renaud Ronse [2], post-doctoral fellow at the Biorobotics Laboratory (Directed by Professor Auke Ijspeert [3]) and Dr Mohamed Bouri[4], the head of the parallel robotics design group at the Laboratory of Robotics Systems (Directed by the Professor R. Clavel).

The Biorobotics Laboratory (BioRob) is part of Institute of Bioengineering in the School of Engineering at the EPFL. Its research interests are at the intersection between robotics, computational neuroscience, nonlinear dynamical systems, and machine learning. One of their recent fields of investigation is rehabilitation robotics. They target to embed their research in advanced control techniques (such as Central Pattern Generators -CPG-) and optimization into robotics platforms providing rehabilitation therapy to disabled persons. Therefore, the investigation of the rehabilitation of locomotion (lower limb), is one of the new objectives of

[2] renaud.ronsse@epfl.ch
[3] auke.ijspeert@epfl.ch
[4] mohamed.bouri@epfl.ch

the Lab and motivated this research project.

The LSRO Lab is a multidisciplinary research unit working mostly in the fields of robotics, micro-robotics and high precision instrumentation. One of their fields of interest is medical robotics and instrumentation. They work on robotics assistance to offer the possibility of performing therapeutic mobilization treatments on paralyzed people and on a challenging locomotor re-education project. The active Knee-orthosis is one of the rehabilitation prototypes developed in the lab. It is used during this thesis as test platforms for the novel rehabilitation method using adaptive oscillators.

1.2 Environment : Presentation of the rehabilitation orthosis

The work described in this thesis uses the the Knee-orthosis. The rehabilitation orthosis (figure 3.1) is used for simulation, validation of the method and for the proof of the concept. It is provided by the LSRO Lab. It is a one-axis robotic arm allowing the mobilisaition of the lower extremities and rehabilitation. It is able to control the flexion and extension of the knee. It is a 1 DOF (Degree Of Freedom) robot.

The Electrical DC motor, an angular position sensor and a force sensor allow the total instrumentation of the device. The used DC motor and the harmonic drive gear provide a resistant torque as well as an assistive torque to implement different interaction strategies between the leg and the orthosis arm. A joint torque up to 50Nm and a speed up to 110°/s may be obtained in the exercises.

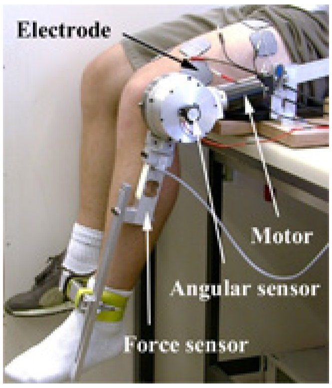


Figure 1.1: Knee orthosis

The figure 1.2 shows the branching schema of the knee orthosis. Where : the reducing coefficient $n = 120$, the torque is computed as following : $\Gamma_{axle} = \Gamma_{motor} * n$ and the motor is commanded by an electric current $\pm 10A$ provided by the amplifier.

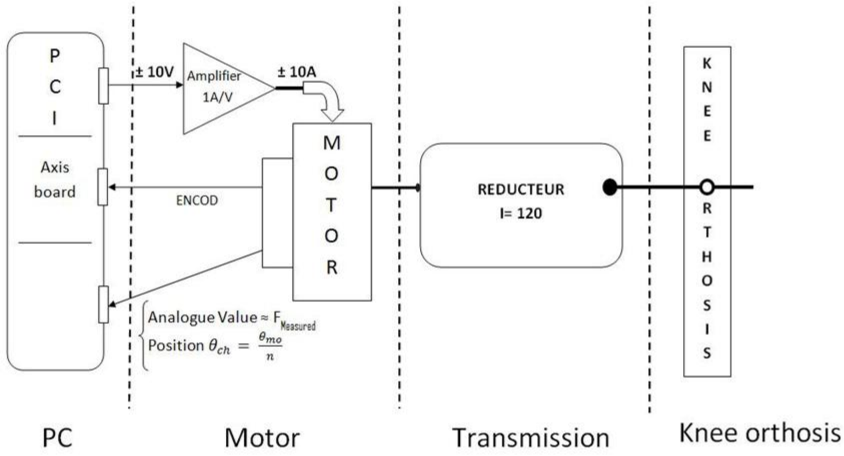


Figure 1.2: Branching schema of the Knee orthosis system

1.3 Project outline

1.3.1 Project stages

The project is decomposed in five stages. In each stages, precise objectives are fixed and a set of tasks have to be done to reach the final aim of the project. It is decomposed as following:

1. This stage aims at understanding the mathematical background of the adaptive CPGs theory and reproduce simulations of the different articles in order to handle with these tools. Matlab, SimuLink and C++ are used to implement and simulate the CPGs.

2. This part consists in deriving the kinematic and dynamical models of a representative human lower body, and adapting the rehabilitation method based on adaptive CPGs on this new configuration with Knee orthosis. Simulation and first experiments will be done on simulink and matlab.

3. This stage focuses on the derivation of a model of the geometrical configuration of the one degree of freedom knee active orthosis and on the programming of the robot interface (C++).

4. A various experimentations will be done to simulate different human movement: sinusoidal and multi-frequency signals

5. To finish, the whole implementation on robots will be tested on intact people.

1.3.2 Project schedule

The figure 1.3 shows the detailed project schedule and the work done during the five months.

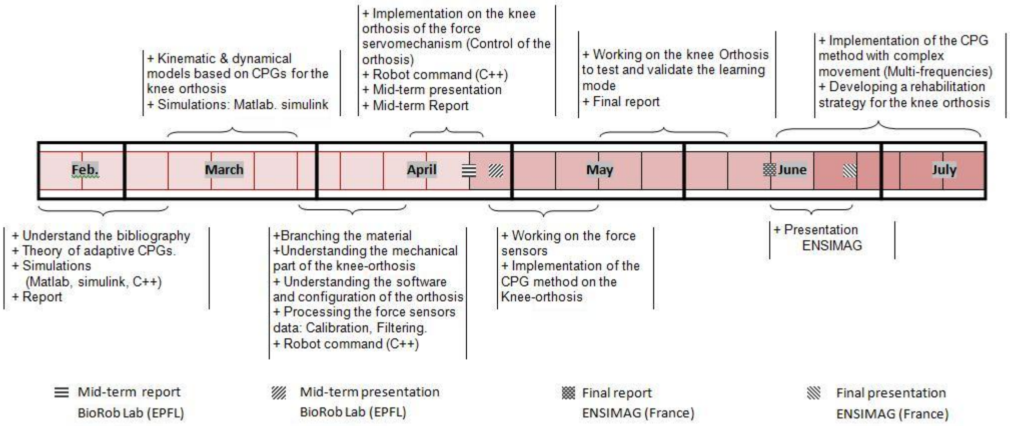


Figure 1.3: Project schedule

Chapter 2

Adaptive oscillators

Nonlinear oscillators are widely used to model various physical and biological processes. Specially in the modeling and control in the field of robotics. Oscillator models are interesting because their ability to synchronize with other oscillators or with any input signal. In the case of classical oscillators, it exists a phase-locking region around the intrinsic frequency delimiting the basin of attraction of synchronization, because of, these oscillators cannot dynamically adapt their parameters outside this region. It is difficult to choose the right parameters of the oscillator to ensure that it will synchronize as desired. ref [2][3] Some recent studies targeted the development of oscillator able to adapt to a wider range of frequencies. The Biorob Lab has designed adaptive oscillators whose frequency can be adapted to the frequency of periodic input signals. It add a large contribution in the field of adaptive oscillators. In this chapter, we will review this fame work of adaptive oscillators.

2.1 Central Pattern Generators

Central pattern generators are biological neural networks that produce coordinated multidimensional rhythmic signals, under the control of simple input signals.

Oscillators are ables to imitate CPGs. The new mechanism of adaptive oscillator using coupled adaptive non-linear oscillators is able to learn arbitrary rhythmic signals. The intrinsic frequencies (ω) is automatically adjusted to replicate the teaching signal and one the signal is removed the trajectory remains embedded into the dynamical system without an external algorithm and can be played back.
the process is dynamic and can be applied to many kinds of teaching signals.

2.2 Architecture of the adaptive oscillators

2.2.1 Basic unit: Hopf Oscillator

These oscillators are able to learn the frequency of complex periodic input signal without any external optimization process.

To illustrate the frequency learning method, we describe the augmented Hopf oscillator proposed in reference [2,3]. The frequency of the oscillator is used as a new state variable. This variable will converge to one of the frequencies of the input signal. The frequency component adapted will depend on the initial conditions for ω. The learning rule equations of these oscillator are as follows:

$$\dot{x} = \gamma(\mu - r^2)x - \omega y + \epsilon F(t) \tag{2.1}$$

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

$$\dot{\omega} = -\epsilon F(t)(\frac{y}{r}) \tag{2.3}$$

where $r = \sqrt{x^2 + y^2}$, μ controls the amplitude of the oscillator, γ controls the speed of recovery after perturbation, ϵ is a coupling constant , $F(t)$ is a periodic input to to which the oscillator will adapt its frequency , ω is the frequency of the oscillator, which will be adapted to the frequency of the input signal. In the case of multi-frequency signal, the oscillator will converge to one of the frequencies. The adaptation has an infinite basin of attraction. ref [2][3]

2.3 Numerical simulations

2.3.1 Learning a simple periodic signal

As first example, we use the adaptive Hopf oscillator to learn a simple periodic signal. Periodic functions are used to describe oscillations and waves. For example, the sine function is periodic with period 2π. The period is the duration of one cycle in a repeating event, so the period is the reciprocal of the frequency. In our example the frequency is equal to 20: The teaching signal $F(t) = sin(20t)$.
For the first experiment, the adaptive oscillator is tested with varying the initial intrinsic frequency $\omega(0)$ = 5, 10, 20, 30, 35. This example will illustrate the ability of the oscillator to adapt its frequency to the frequency of the input signal and will show the duration of this adaptation.

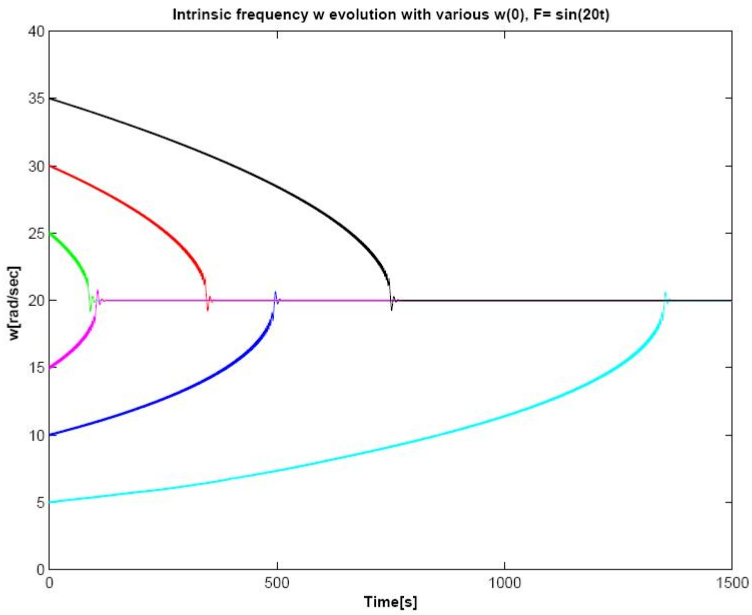


Figure 2.1: Convergence of the frequency of an adaptive frequency oscillator driven by a signal F(t) = sin(20t) for different initial intrinsic frequencies

The figure 2.1 shows that in all cases the oscillator is able to adapt it own frequency to the frequency of the input signal and for the different $\omega(0)$. However, the synchronization duration is different and depends on the initial frequency of the oscillator.

As a second simulation, we plots the evolution of ω for different values of ϵ =[0.2 , 0.5 , 0.8, 1] using the teaching signal$F(t) = sin(30t)$. $\omega(0)$=35.

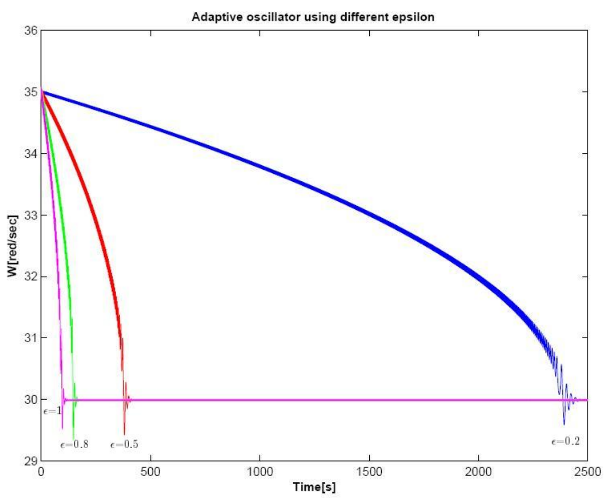


Figure 2.2: Convergence of the adaptive oscillator frequency using different epsilon

The figure 2.2 clearly shows that the coupling constant ϵ determines the convergence speed of the oscillator around the teaching frequency. ϵ will be fixed at 1 as a default value.

2.3.2 Learning multi-frequency signals

[1] To simulate the learning of multi-frequency signal, we used the following input :

$$F(t) = sin(10 * t) + cos(25 * t) + sin(32 * t)$$

We will show through this example how the adaptive oscillator will converge and give a formula to compute the frequency to which the oscillator will adapt its own ω. The following figure plots the evolution of the intrinsic frequency of the adaptive oscillator when we have different initial values of the state variables and illustrate the frequency component that attract ω.

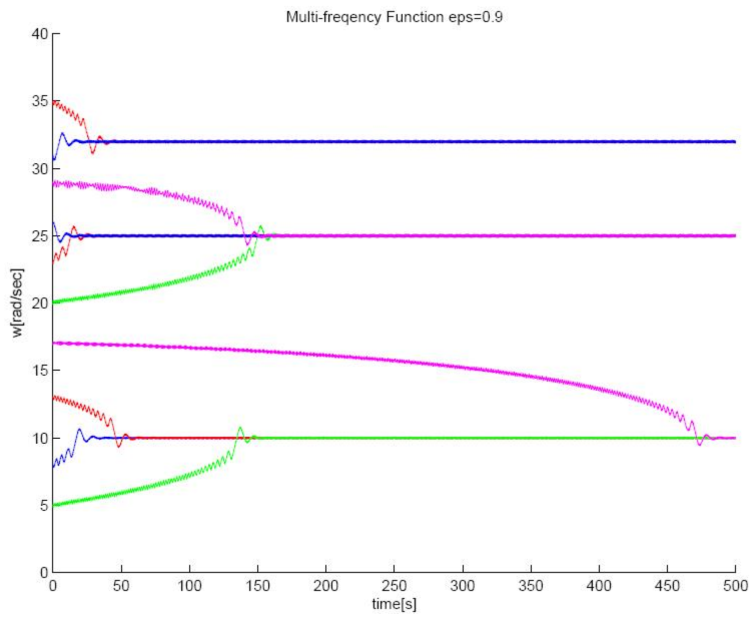


Figure 2.3: Graph showing the convergence of the adaptive oscillator frequency to one of the frequencies components of the teaching signal

Remark: The adaptive oscillators are able to learn also the pseudo-periodic signals. see ref [3].

2.4 Coupled Oscillators for learning multi-frequency signal

The basic idea is to use the adaptive property of the oscillators to learn the different frequencies of a periodic teaching signal. It will be similar to dynamic Fourier series represen-

[1]This section will be used in the future work on the multi-frequency movements. ref [1]

tation, each oscillator will encode one frequency component of the learning signal.
The previous section showed that an adaptive oscillator is able to learn one frequency component of a multi-frequency signal. Now, feedback is used to couple a set of adaptive oscillator and to make each oscillator learning one frequency component of the teaching signal.

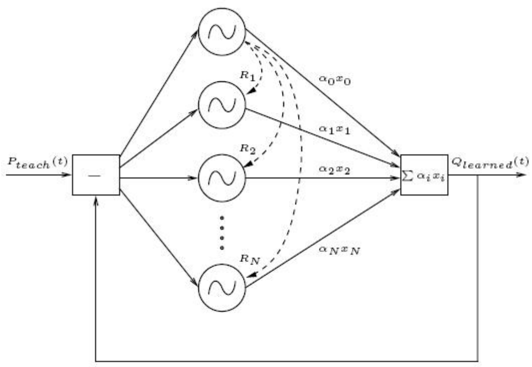


Figure 2.4: illustration of the multi-frequecies learning method

According to figure 2.5, the equation 2.1, 2.2 and 2.3 will be adapted to the new learning schema. The frequencies learned by the other oscillators will be considered and deleted from the main teach signal. The equation are the following:

$$
\begin{aligned}
\dot{x}_i &= \gamma(\mu - r_i^2)x_i - \omega_i y_i + \epsilon F(t) + \tau \sin(\theta_i - \phi_i) \\
\dot{y}_i &= \gamma(\mu - r_i^2)y_i + \omega_i x_i \\
\dot{\omega}_i &= -\epsilon F(t)\frac{y_i}{r_i} \\
\dot{\alpha}_i &= \eta x_i F(t) \\
\dot{\phi}_i &= \sin\left(\frac{\omega_i}{\omega_0}\theta_0 - \theta_i - \phi_i\right)
\end{aligned}
$$

with

$$
\begin{aligned}
\theta_i &= \mathrm{sgn}(x_i)\cos^{-1}\left(-\frac{y_i}{r_i}\right) \\
F(t) &= P_{\text{teach}}(t) - Q_{\text{learned}}(t) \\
Q_{\text{learned}}(t) &= \sum_{i=0}^{N} \alpha_i x_i
\end{aligned}
$$

Figure 2.5: Equation of the coupled oscillator to learn a multi-frequency signal

More details are given in ref[1]

Chapter 3

Model of the Knee active Orthosis

To implement and validate a reliable rehabilitation protocol, a computer simulation is used first to explore and gain new insights into adaptive oscillators. It estimates the performances of the system and provides preliminary theoretical results.The whole system is modeled using a Simulink Model for the control, parameter adjustment of the orthosis.

The study would like to prove that the adaptive oscillators provide augmentation to the user and leave all high-level parameters of the movement up to him.

The system is decomposed into two main blocks. The first block is the mechanical model of the orthosis and the human movement simulator. The second block is the learning and augmentation block. It targets to integrate the adaptive oscillator to the global system and study the learning method of the movement.

3.1 Dynamical model of a representative human lower limb

A first step consist in finding an equivalent dynamical system to represent the human lower limb and its interaction with the orthosis. This section provides a description of the lower limb and orthosis modeling. The model provide the evolution of the orthosis position.

3.1.1 Dynamical model

The movement of the human using the knee orthosis has one degree of freedom. It moves along a circular path. It is similar to the movement of a pendulum. Figure 3.1 shows the similarities between the two systems.

The primary forces acting on the leg are the gravitational forces. In addition, there may be a damping force from friction at the pivot. The model constructed describes how the angular position θ of the orthosis varies as a function of time t. The gravitational force is directed downward and has magnitude mg where g is the gravitational acceleration and m is the mass of the leg + the orthosis axle.The force acting in the tangential direction is $-mgsin(\theta)$. We add damping to the model. We make the simplest possible assumption about the damping force, that it is proportional to angular velocity, $-b\frac{d\theta}{dt}$. In addition, the human provides a torque u during its movement as an input.

The simple equation is written as follows:

$$I\ddot{\theta} = -mglsin\theta - b\dot{\theta} + u$$

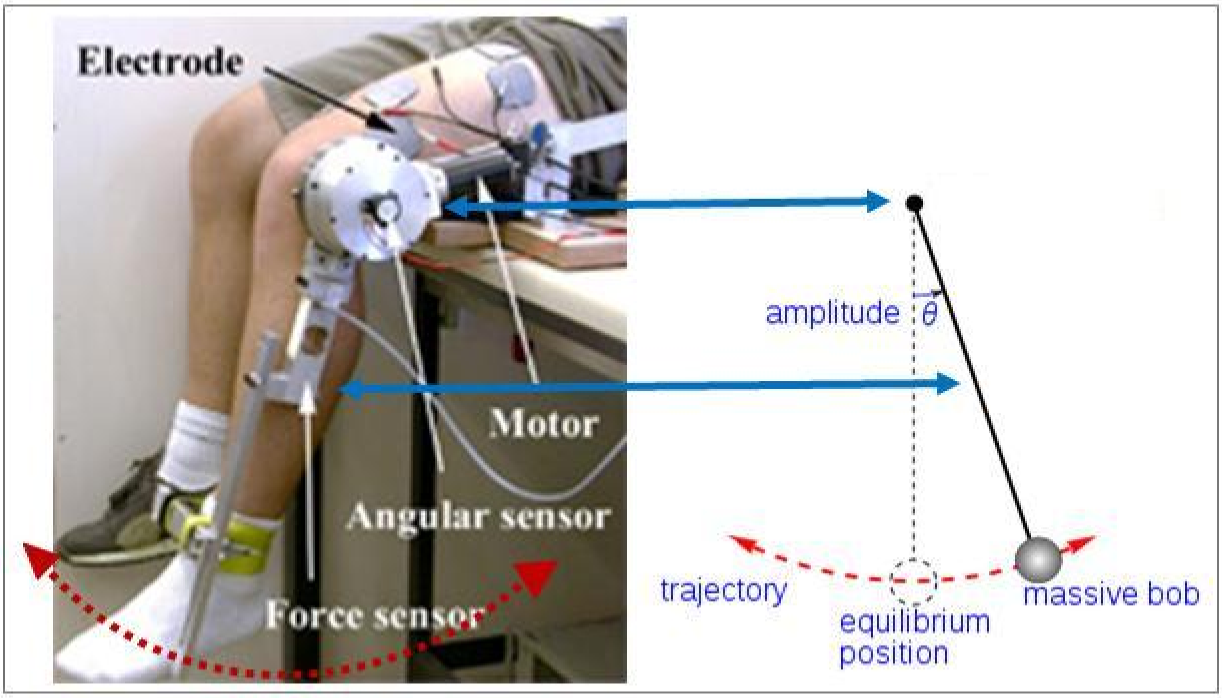


Figure 3.1: Figure showing the similarities between the pendulum and the knee-orthosis robot

3.1.2 Position controller of the knee orthosis

The model of the human controller is a PID controller used to simulate the torque $U_h(t)$ provided by the human. The reference movement is a periodic function that can be changed to simulate different desired movement. Using the dynamical model of human lower limb described the previous section, the real-time position is provided and simulates the orthosis sensors measurement.

A closed-loop feedback mechanism controlled by a PID minimizes the position error. It adjusts the actual position with providing a torque according to this formula:

$$U_h(t) = K_p e(t) + K_i \int_0^x e(t)d\tau + K_d \frac{d}{dt}e(t) \tag{3.1}$$

where :

Proportional gain, Kp

Larger values typically mean faster response since the larger error is linked the proportional term compensation. An excessively large proportional gain will lead to process instability and oscillation.

Integral gain, Ki

Larger values imply steady state errors are eliminated more quickly. The trade-off is larger overshoot: any negative error integrated during transient response must be integrated away by positive error before reaching steady state.

Derivative gain, Kd

Larger values decrease overshoot, but slow down transient response and may lead to instability due to signal noise amplification in the differentiation of the error.

Algorithm:

```
previousError = 0
integral = 0
start:
    error = setpoint - actualPosition
    integral = integral + (error*dt)
    derivative = (error - previousError)/dt
    output = (Kp*error) + (Ki*integral) + (Kd*derivative)
    previousError = error
    wait(dt)
goto start
```

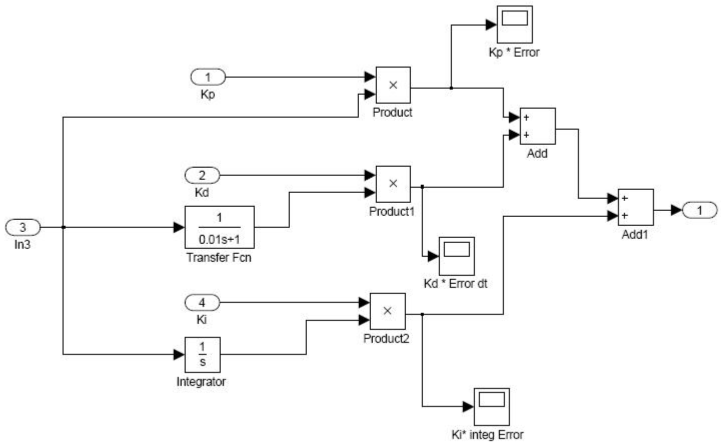


Figure 3.2: The PID controller implemented with the Simulink library

3.2 Adaptive oscillator and augmentation estimator block

The learning method using the adaptive oscillator and the computation of the orthosis augmentation are provided in the following section. To finish, some simulation are described to show the validation of the method and the discussion made about the proposed solution.

The method developed in this work uses the adaptive oscillator presented in the previous chapter and consists in learning the frequency of the human movement. It means that the oscillator estimates and provides on each sampling period the corresponding x, y formula 2.1, 2.2 and ω formula 2.3 of the input signal. The input signal $\theta(t)$ is the orthosis-human position.

The next step consists in using the data provided by the oscillator to build an estimated evolution of the human position, velocity and acceleration. This estimation makes able to compute an estimated human torque.

The details of the implementation are given in the following sections and the rehabilitation method is described in the last section provided by example of simulation results.

3.2.1 The adaptive oscillator and signal estimator

This part of the system is decomposed into two main blocks [see figure 3.3]:

(1) The modified Hopf oscillator which is the adaptive oscillator and which computes the frequency of the input signal by adapting its own intrinsic frequency to this one.

(2) The position and movement characteristics estimator used to estimate the position, velocity and acceleration of the orthosis and human leg.

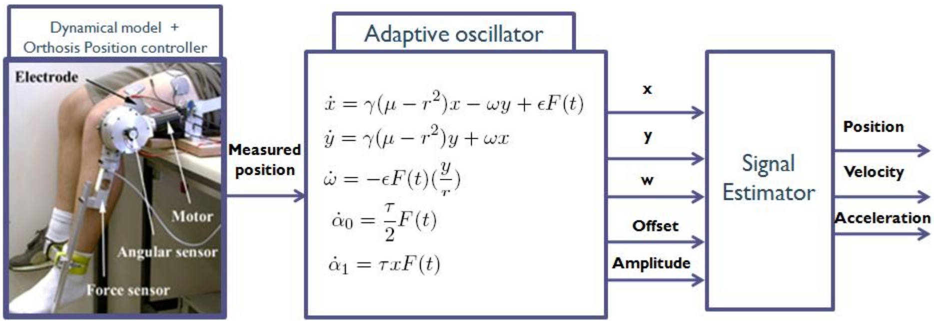


Figure 3.3: Figure showing the adaptive oscillator and estimator blocks implemented in the Simulink model

(1) The adaptive oscillator

Using the Hopf adaptive oscillator system formula 2.1,2.2,2.3, an adaptive block was implemented using the Simulink S-function standards [Appendix]. Additional variable $\alpha 0$ and $\alpha 1$ corresponding respectively to the offset and amplitude of the signal are computed using the evolution of the usual input variables of the adaptive oscillator.

$$\dot{\alpha}_0 = \frac{\tau}{2}F(t) \tag{3.2}$$

$$\dot{\alpha}_1 = \tau x F(t) \tag{3.3}$$

Where $F(t)$ depends on the position the human leg (Orthosis) and the estimated position computed by the next block $F(t) = \theta(t) - \hat{\theta}(t)$

(2) The position and movement characteristics estimator

To estimate the position, velocity and acceleration of the movement, using the previous results, the equations are as follows:
For the first study, we consider that the human movement is sinusoidal and can be estimated thanks to the offset, amplitude and x position data as follows :

$$\hat{\theta} = \alpha_0 + \alpha_1 x \tag{3.4}$$

The velocity is simply computed using the amplitude , frequency and y position data. It is the derivative of the estimated position.

$$\dot{\hat{\theta}} = \alpha_1 \omega y \tag{3.5}$$

The acceleration is the estimated velocity derivative. It is computed as following:

$$\hat{\ddot{\theta}} = \alpha_1 \omega^2 x \tag{3.6}$$

3.2.2 The augmentation estimator block

Using the previous results, an estimated user torque can be computed using the following dynamical equation of the system:

$$\hat{u} = mglsin(\hat{\theta}) + b\hat{\dot{\theta}} + I\hat{\ddot{\theta}} \tag{3.7}$$

Where $\hat{u}$ is the estimated total torque of the system. $\hat{u} = \hat{u}_{Human} + \hat{u}_{OrthosisAugmentation}$

To provide an assistance, the torque provided by the human have to be estimated and augmented to decrease his effort. The estimated torque $\hat{u}$ can be easily computed using the result of the signal estimator for the $\hat{\theta}, \hat{\dot{\theta}}$ and $\hat{\ddot{\theta}}$ [See formula 3.4, 3.5, 3.6]. The augmented torque will be a simple gain torque computed as follows:

$$u_{Augmentation} = K\hat{u}_{Human} = K(\hat{u} - \hat{u}_{OrthosisAugmentation}) \tag{3.8}$$

Where $K \in [0, 1[$ specifies the augmentation added to help the human to do the movement with less effort.

3.3 Simulations and Results

3.3.1 Simulation 1 : Simple stable sinusoidal movement

As a first simulation, the user provides a stationary effort doing periodic sinusoidal movement. The referent movement is a function defined as follows:

$$\theta(t) = \alpha * sin(\omega * t)$$

Where α the amplitude is equal to $\frac{\pi}{2}$, ω the frequency of the movement is equal to 2π

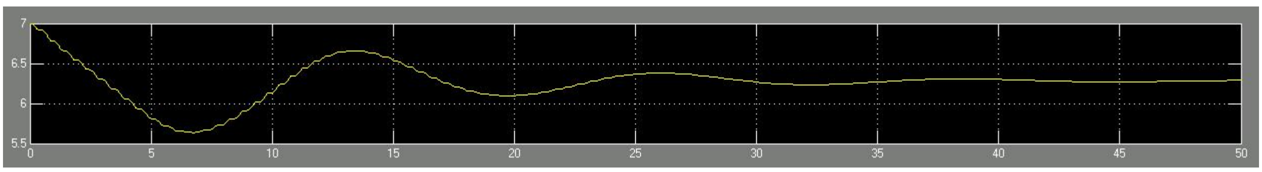


Figure 3.4: Evolution of the adaptive oscillator frequency

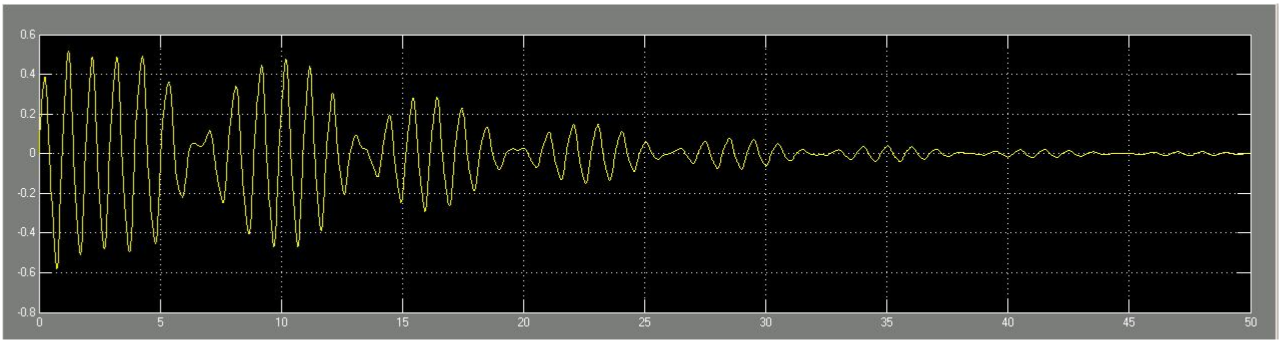


Figure 3.5: Evolution of the error of estimation: Current Position - Estimated position [deg]

Discussion

The figure 3.4 shows that using an adaptive oscillator with an initial $\omega = 7$ can, in less than 25 sec, adapt its frequency to the frequency of the input signal $\omega = 2\pi$. In addition, the figure 3.5 shows clearly the relation between the frequency adaptation and the position estimation. The error of the estimated signal θ decreases during the oscillator adaptation stage and converges to 0 when the oscillator learnt the signal frequency. In this case the estimator torque block can easily compute the human torque and provide the desired augmentation according to the formula 3.8. [Appendix [1][2] Simulink model]

3.3.2 Simulation 2 : Sinusoidal movement with variable frequency

In this simulation, the input signal is a sinusoidal one which the initial frequency $\omega = 2\pi$ and after 40 sec decreases to $\omega = \pi$. This means that the adaptive oscillator has to adapt its intrinsic frequency respectively from 2π to π. The initial frequency of the oscillator is equal to 7.

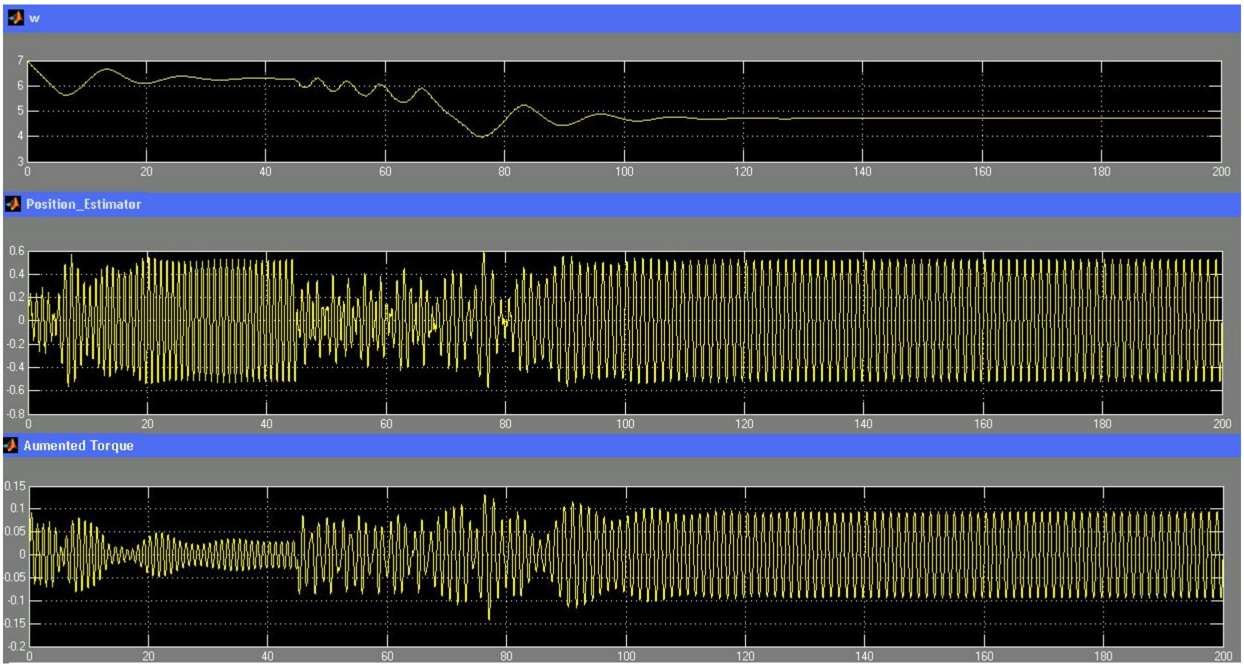


Figure 3.6: figure showing the evolution of the adaptive oscillator frequency to the two different frequencies of the input signal and the estimation of position which depend on the frequency adaptation

Discussion The figure 3.6 shows that the adaptive oscillator adapts its frequency to the frequency component of the signal in a very short time. The signal estimator using the data providing by the oscillators estimates in parallel the correct characteristics of the user movement and provides the estimated torque for each sampling time.

The results of the previous simulations show the ability of the learning block (adaptive frequency + signal estimator) to learn the human movement in a short time and with an error of estimation which converge to 0. This results validate our expectation of the system performance. The next stage is the implementation and validation of the method on the knee orthosis.

Chapter 4

Implementation of the method on the knee orthosis

In the previous chapters, the general method for designing the lower limb assistance using adaptive oscillators was simulated and validated. Furthermore, it was shown that the method provides a the expected results (signal estimation) for the sinusoidal movement.

This last chapter will discuss the different stages to implement and test the augmentation method on the knee orthosis.

The work on the orthosis is decomposed into three mains stages: (1) Implementation of the transparent mode stage (2) The learning stage (3) The torque augmentation stage. The figure 4.1 shows the decomposition schema that will be detailed in the following chapter.

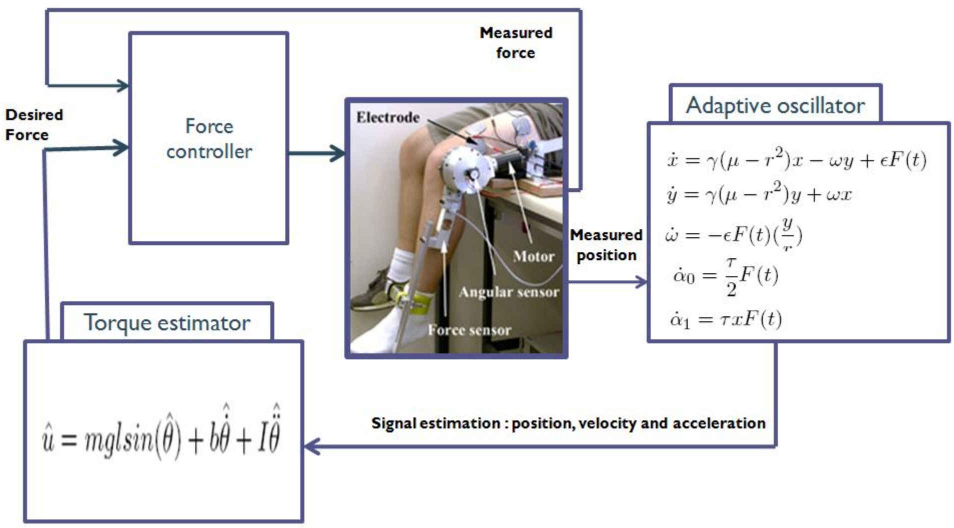


Figure 4.1: Global schema of the rehabilitation system implementaed on the knee-orthosis

4.1 Step 1 : Implementation of the transparent mode

The objective of this preliminary work is to offer the transparent mode to the user, such that the user will move the orthosis with a low perception of the mechanical impedance during user-driven motions. [Appendix [4,5]]

4.1.1 Force calibration and curve interpolation

The preliminary work done on the orthosis consisted in calibrating the data transmitted by the force sensor.

The force sensor provides an analogical value corresponding to the force voltage (Volt). This data must be processed and calibrated to have an equivalent force value in Newton. Therefore, a force calibration curve was constructed using a set of measurements.

The method of measurement is as follows: using a dynamometer, a device for measuring force, the orthosis is initialized on the vertical position (0°) . A set of force value in a range of [-5, 10] KgN is applied on the orthosis. The voltage measured by the force sensors makes able to draw the calibration curve. This curve provides the correspondence between any voltage (Volt) measured and the force value in (Newton). See figure 4.2.

To construct the final Force-voltage curve, the Spline quadratic interpolation method was implemented in C++ and used on the range of the discrete measurements done using the dynamometer. The Spline interpolation uses low-degree polynomials in each of the intervals, and chooses the polynomial pieces such that they fit smoothly together [code in the Appendix [3]].

Force	-5	-4	-3	-2	-1	0
Tension	-0.85	-0.75	-0.67	-0.55	-0.41	-0.23

Force	1	2	3	4	5	6	7	8	10
Tension	-0.1	0	0.1	0.2	0.37	0.58	0.65	0.77	0.96

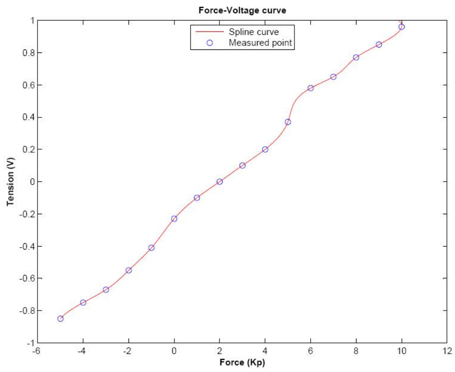


Figure 4.2: The graph represents the measured Force-voltage values and the spline interpolation used to represent the curve

Discussion

The curve constructed from the set of measurements has some interfering signal noise. The next step is signal filtering. It targets removing some frequencies and suppressing interfering noise.

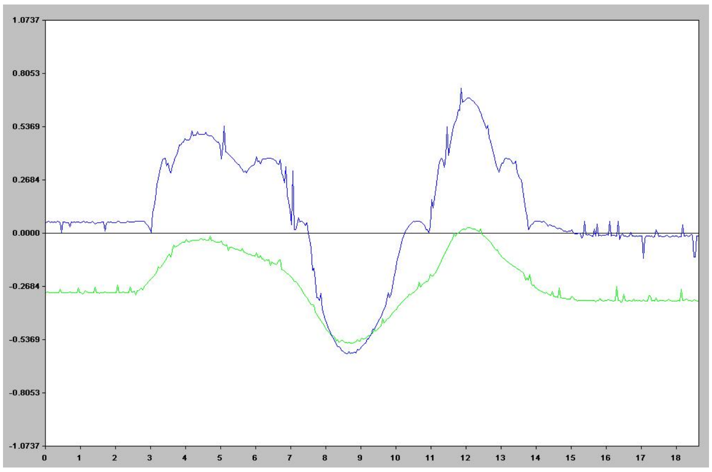


Figure 4.3: The calibration of the measured force signal. The green signal corresponds to the measured voltage (Volt), The blue signal represents the calibrated and interpolated signal

4.1.2 Force filtering

The next stage in processing the measured force is the filtering process. It is used to eliminate unwanted frequencies from the received signal. While the correct filter settings can significantly improve the visibility of a defect signal, incorrect settings can distort the signal presentation and even eliminate the defect signal completely. Therefore, it is important to understand the concept of signal filtering. The filter used is a low Pass Filter. It is adjusted in Hertz (Hz). It was calculated as following:

$$\frac{y_f}{y} = \frac{1}{1+\tau s}$$

$$=> y_f + \frac{\tau y_f}{T_e} - \frac{\tau y_f^-}{T_e}$$

$$=> (1 + \frac{\tau}{T_e})y_f = \frac{\tau}{T_e} y_f^- + y$$

$$=> (\frac{T_e + \tau}{T_e})y_f = \frac{\tau}{T_e} y_f^- + \frac{T_e}{T_e} y$$

$$=> y_f = \frac{\tau}{T_e + \tau} y_f^- + \frac{T_e}{T_e + \tau} y$$

Where y_f is the filtered signal, y is the measured signal, y_f^- is the previous value of the signal, T_e is the sampling time, $\tau = \frac{1}{B_p}$, where B_p is the bandwidth, B_p= 20 Hz

Results : The following figure illustrate the filtering result.

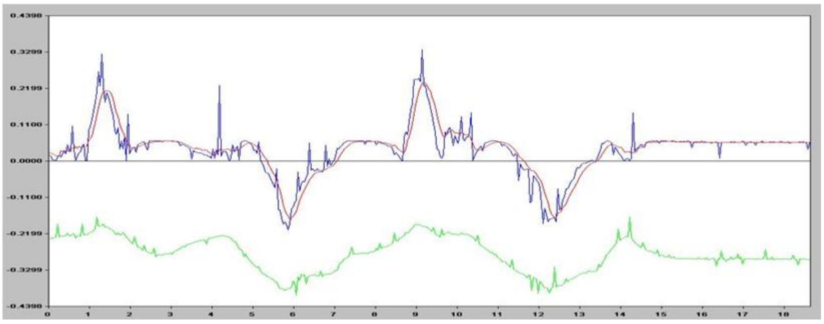


Figure 4.4: Curve of filtred signal, the green signal represent the tension measured by the force sensors. The blue signal is the corresponding force curve before applying the filter, the red curve is the filtered signal

4.1.3 Characterization of the knee orthosis

This section targets to study the characterization of the orthosis. It is used in the estimation of the gravitational and dry friction torque of the orthosis.

Force measured for a specific tension curve

This study targets to measure the force applied on the orthosis arm when a defined tension is sent. The method is the following: The orthosis arm is fixed to the position (0 °)[see fig 4.5] and a tension increasing from -4 Volt to +5 Volt is sent. Fixed, the orthosis arm cannot move and the force sensors will be able to measure the force applied by the motor on the orthosis arm. The following table details the measurements done on the orthosis (See fig 4.6).

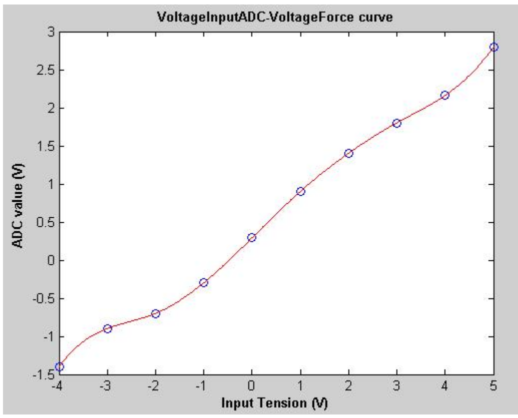


Figure 4.6: Curve representing the measured tension (Volt) according to the input tension sent to the motor

Applied Tension	-4	-3	-2	-1	0	1	2	3	4	5
ADC measured	-1.40	-0.90	-0.70	-0.30	+0.29	+0.90	+1.80	+1.40	+2.16	+2.80

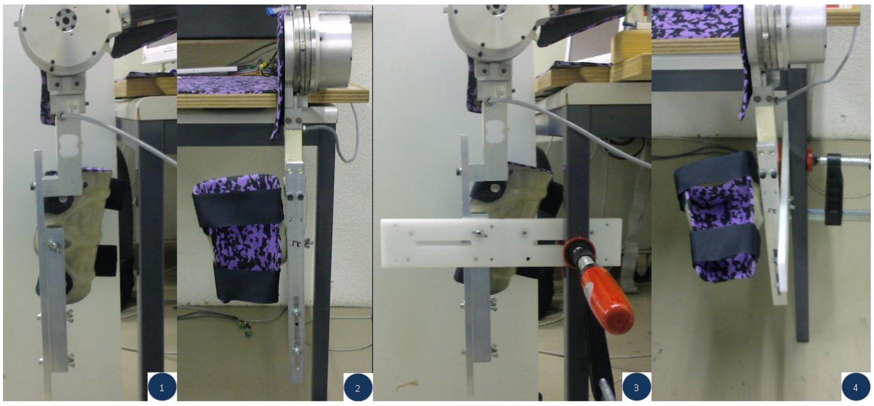


Figure 4.5: Figure showing the method used to fix the orthosis arm. (1)(2) Pictures before fixing the orthosis. (3)(4) Shows the fixing method

.

Characterization of the static torque of the knee orthosis

In a flexion or extension movement, the subject using the orthosis will be confronted to external forces which will disturb the user movement and increase its effort to move the orthosis arm. Therefore, here, we compute the required input voltage to reach a position which is not the vertical one.

For this experiment a set of eleven position are chosen in the [-60 $^\circ$, +20 $^\circ$] range. Each 10 $^\circ$, the corresponding sent tension is registered in both direction (flexing(High-bottom)-

extension (Bottom-high)). See figure 4.7. The results are providing in the following table:

θ	-60	-50	-40	-30	-20	-10	0	10	20
Flexing	-0.27	-0.26	-0.24	-0.2300	-0.1800	-0.16	-0.120	-0.10	-0.08
Extension	-0.05	-0.06	-0.04	-0.0106	-0.0156	0.02	0.052	0.10	0.16

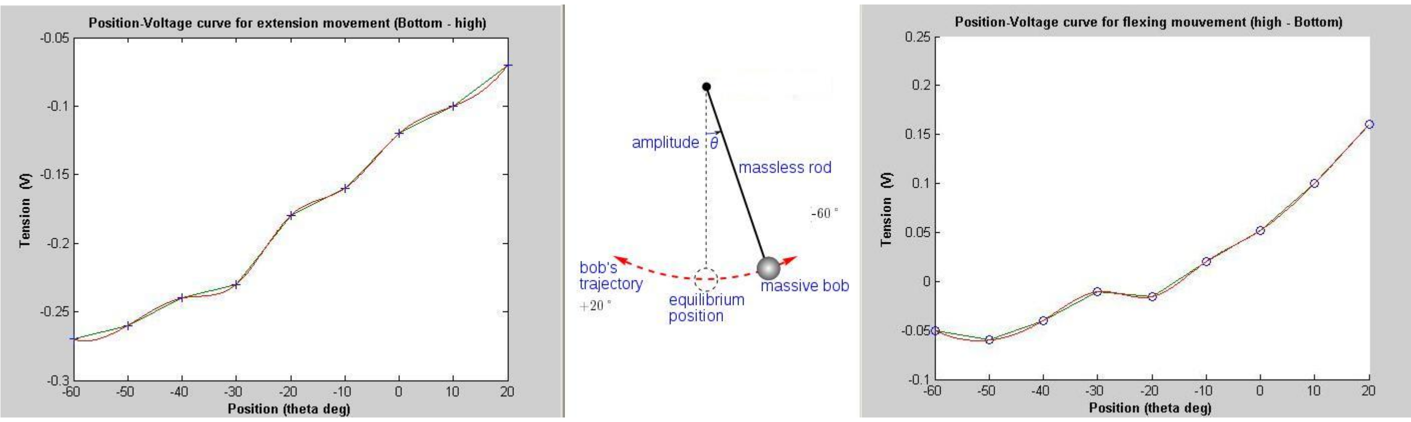


Figure 4.7: shows the curve displaying the tension sent according to the movement direction and the targeted position

Discussion

The obtained curves show that the required torque when the human is doing in flexion or extension movement is different. The gravitational force in a position is the same when we do a flexion or extension movement. It is linked to the position and not to the velocity or acceleration. So, the difference between the two curves is due to the dry friction which influences with different degree the orthosis arm depending on the movement direction. It means when we are in flexion movement. The orthosis arm is reaching the vertical position (0°). The required torque is smaller than the torque when the orthosis is going up doing a extension movement.

To eliminate the gravitational and dry friction forces when the user is using the orthosis, a torque Γ is added to overcome the influence of this forces. It is computed as follows:

Each measurement is done according to a sampling period $SamplingPeriod = 0.0002$. Where the incrementation of the velocity during one sampling period is $Res_v = \frac{Res_\theta}{SamplingPeriod}$, we consider that the maximum velocity $\omega^+ = 5 * Res_v$, the minimum velocity $\omega^- = -\omega^+$ and the current velocity $\omega_{current} = \frac{\theta_{current} - \theta_{old}}{SamplingPeriod}$.

Using this data, the torque is estimated thanks to the previous measurements and according to the following algorithm:

Algorithm :

if ($\omega_{current} \leq \omega^-$) **then**
$\Gamma = \Gamma^-$;
else if ($\omega_{current} \geq \omega^+$) **then**
$\Gamma = \Gamma^+$;
else
$\Gamma = \frac{\Gamma^+ - \Gamma^-}{\omega^+ - \omega^-}$;

Where Γ^+ is the torque measured when it is a flexing movement, Γ^- is the torque measured when it is a extension movement. See figure 4.8

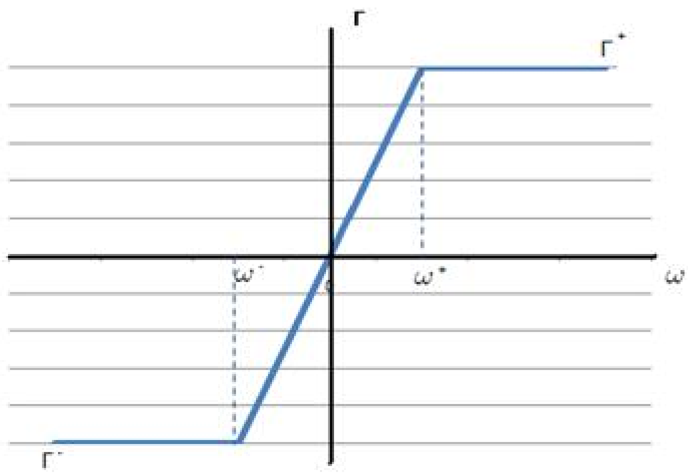


Figure 4.8: The function chosen to compute the required torque to send vs the direction and velocity of the movement

4.1.4 Closed loop

To make the user able to move the orthosis with a low perception of the mechanical impedance during user-driven motions, a closed loop is implemented using the previous results and providing an additional torque which will drive the interaction force between the human and the orthosis to zero : $F_{interaction} \rightarrow F_{Desired} = 0$. To reach this objective, the orthosis is provided by a PID controller.

The figure 4.9 shows the global schema of the transparent mode which will be detailed the following sections.

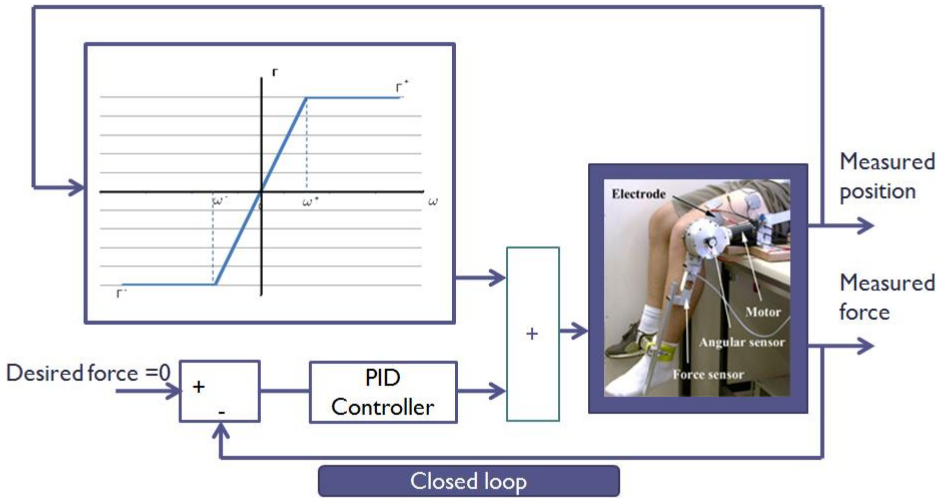


Figure 4.9: Closed loop of the transparent mode

Force control with proportional gain

As a first experiment. the closed loop use only a proportional gain K_p. The providing torque U_{app} is computed using the effects of a closed feedback loop. It means that the measured force provided by the force sensors is used to reach the desired one $F_{Desired} = 0$ as following:

$$U_{app} = K_p[F_{desired} - F_{measured}]$$

The following figure illustrate the torque sent to decrease the human effort to move the orthosis. The proportional gain K_p= 1. It means that the orthosis must provide the same force provided by the user to have $F_{interaction} \rightarrow F_{Desired} = 0$.

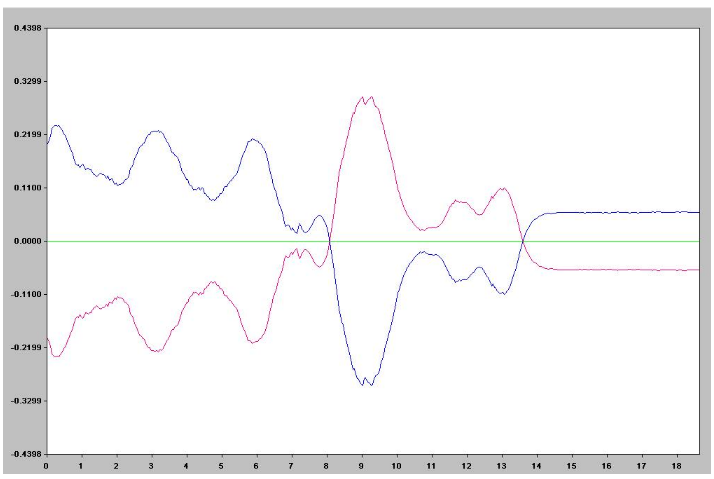


Figure 4.10: In blue, the measured toque that the human provide to move the orthosis, in red, the computed torque using the previous formula with Kp = 1

Force control with a proportional and integral gain

In this closed loop the provided torque U_{app} used the effects of a closed feedback loop with the measured force and the desired one. The measurement error ϵ_F is defined by : $\epsilon_F = T_{desired} - T_{measured}$. So, the provided torque is computed by the following equation:

$$U_{app} = K_p \epsilon_F - K_i \int_0^t \epsilon_F(t) dt$$

Where K_i is the integral gain

4.1.5 Discussion

Using the closed loop in addition to the estimated torque, the orthosis moves freely and the human effort decreased considerably. Figure 4.11 shows that the human effort when the orthosis is without the transparent mode is larger than using the closed loop. The third curve illustrate the effect of the additional gravitational torque to decrease the human force.
The transparent mode makes us able to implement the learning block and to start the study of the rehabilitation method.

Figure 4.11: The three figures shows the human effort in three diffrent cases :(1)Without transparent mode. force: [-4,4] Newton (2) With orthosis controller Kp=2. force: [-3,2] Newton (3) With orthosis controller and compensation torque Kp=2, force: [-2,1]Newton. [images from demonstration video]

4.2 Adaptive oscillator and augmentation torque

The validation of the transparent mode makes the user able to do its movement without any additional efforts. The learning mode can be at this stage implemented and integrated to the whole rehabilitation system.

The goal of this section is to describe the different stages to implement, test and validate the new rehabilitation protocol using the adaptive oscillator.

The learning block was implemented in C++ [Appendix [6]]. A first part details the preliminary test to validate the learning-block using a reference movement with a known amplitude and frequency. Afterwards, a second step provides the results of the learning protocol using the movement of a subject using the orthosis. To finish, the whole torque estimation and rehabilitation protocol validation are described with different experimentations done with a subject.

4.2.1 First stage: Validation of the learning block using a reference movement

To validate the learning block, as a first experimentation. A first routine was write to send a reference torque. This provided torque is a simple sinusoidal signal. This torque makes the movement of the orthosis leg similar to the movement of a pendulum fig 4.12. The following function describes the sent torque:

$$Torque(t) = \alpha sin(\omega t)$$

Where: α is the amplitude of the signal, ω frequency of the signal

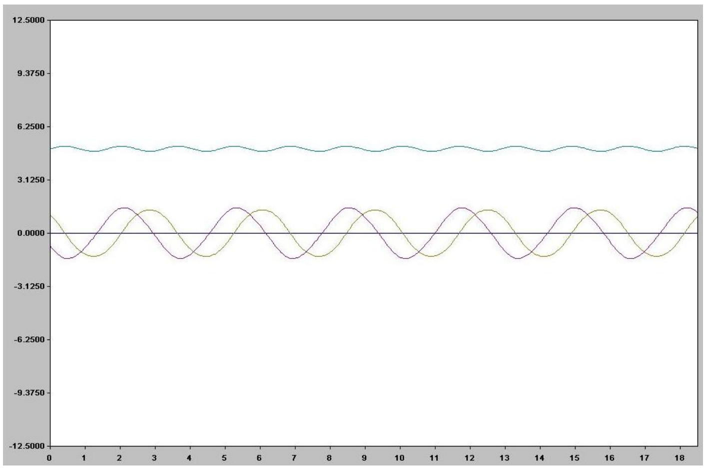


Figure 4.12: Simulation using an in blue signal in the top represents the frequency of the oscillator, The two other signals represent state variables x, y . The input signal frequency = 4 Hz

The simulation is using an input signal $F(t) = \alpha sin(\omega t)$ with $\alpha = 3$ and $\omega = 4$, the blue signal in the top represents the frequency of the oscillator, The two other signals represent the xy state variables.

The experimentations consists on sending sequentially a various frequency and prove the ability to adapt the intrinsic frequency of the oscillator with the input one. The Hopf adaptive oscillator is tested with different signal frequency.

4.2.2 Results

The adaptive oscillator is able to adapt its frequency to the input signal. The position of the orthosis and the movement velocity and acceleration are correctly estimated and as shown in the figure 4.13

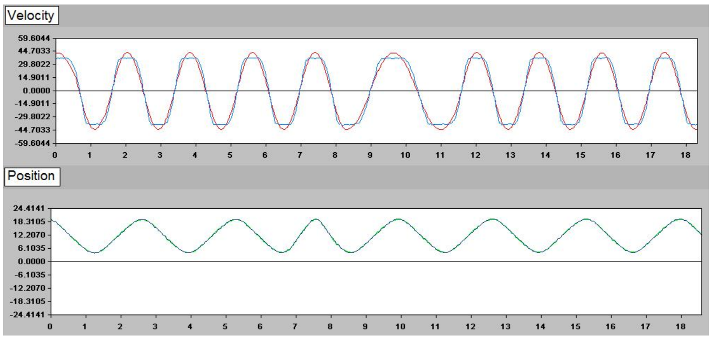


Figure 4.13: The figure shows the estimated position and velocity of the sent signal : in red (velocity) and blue(position) of the input signal , in bleu (velocity) and green(position) of the estimated signal

4.2.3 Second stage: Human movement

In this section stage the test is made on the orthosis in interaction with a human. The subject is asked to provide a movement. It is a periodic flexion/extension movement.

To ensure that the amplitude is same ($\pm\epsilon_{error}$), the user must repeat the same movement with the same velocity in this limited space [- 20 $^\circ$, + 20 $^\circ$].

The augmentation block is used to reduce the human effort uses an augmentation $K = 0.5$.

$$u_{Augmentation} = K\hat{u}_{Human} = K(\hat{u} - \hat{u}_{OrthosisAugmentation})$$

4.2.4 Results and discussion

The adaptive oscillator is able to follow the human movement and to reproduce the signal characteristics (position, velocity and acceleration). During the experiment, the user changed the frequency of its movement and the test results showed that the assistance system takes a short time to learn the input signal and to provide the augmentation that the human need. The system has enough time to follow the movement and the user move the orthosis with a low perception of the augmentation system process.

Chapter 5

Conclusion

The main goal of this study was to investigate a rehabilitation protocol based on the theory of adaptive oscillators. The major contribution of this thesis was to propose a solution to make the orthosis transparent for the user by creating a transparent mode and to study the augmentation strategy to reduce the human effort.

The work was decomposed into two main stages. To implement and validate a reliable rehabilitation system, a computer simulation was firstly used to explore and gain new insights into the use of this new technology of adaptive oscillator. It made us able to estimate the performance of the system and to have preliminary theoretical results.The whole system was modeled using Simulink for the control and adjustment of the orthosis. The second stage targeted the implementation of the rehabilitation method on the Knee orthosis and the validation of the on-line learning of the movement and the device augmentation. A set of experiments was done to validate the rehabilitation method. The first one consisted to provide a reference movement to the orthosis. This movement was previously tested in the Simulink model which made us able to compare the simulation results with the orthosis on-line learning and augmentation. The second experiment was done on a user which provide a movement . The subject tested firstly the orthosis without any assistance and compared it the augmentation mode.

This study made us able to investigate and validate the rehabilitation method on the knee orthosis and to demonstrate the feasibility of using adaptive oscillators for human augmentation. Similar experiments will be realized using EMG to have the precise evolution of the participant effort during the assistance stage. Also, the next step in this thesis will be to generalize the study to complex signals such as multi-frequencies movement. The experimentations will be done during the next month and the results will make a major contribution to the investigation of the rehabilitation method using adaptive oscillators. As future work, this study will be extended to a more complex robot as the Lambda robot offering 3 degrees of freedom per leg.

Chapter 6

References

[1] L. Righetti and A. J. Ijspeert. Programmable central pattern generators: an application to biped locomotion control. In Proc. IEEE International Conference on Robotics and Automation ICRA 2006, pages 1585 1590, 2006. doi:10.1109/ROBOT.2006.1641933.

[2] J. Buchli, L. Righetti and A. J. Ijspeert . Frequency analysis with coupled nonlinear oscillators. Physica D, 237: 1705..1718, 2008.

[3] L. Righetti, J. Buchli and A. J. Ijspeert. Dynamic hebbian learning in adaptive frequency oscillators. Physica D, 216: 269..281, 2006.

[4] A. Dollar and H. Herr. Lower extremity exoskeletons and active orthoses: Challenges and state-of-the-art. IEEE Transactions on Robotics, 24(1): 144 ..158, 2008. ISSN 1552..3098. doi: 10.1109/TRO.2008.915453.

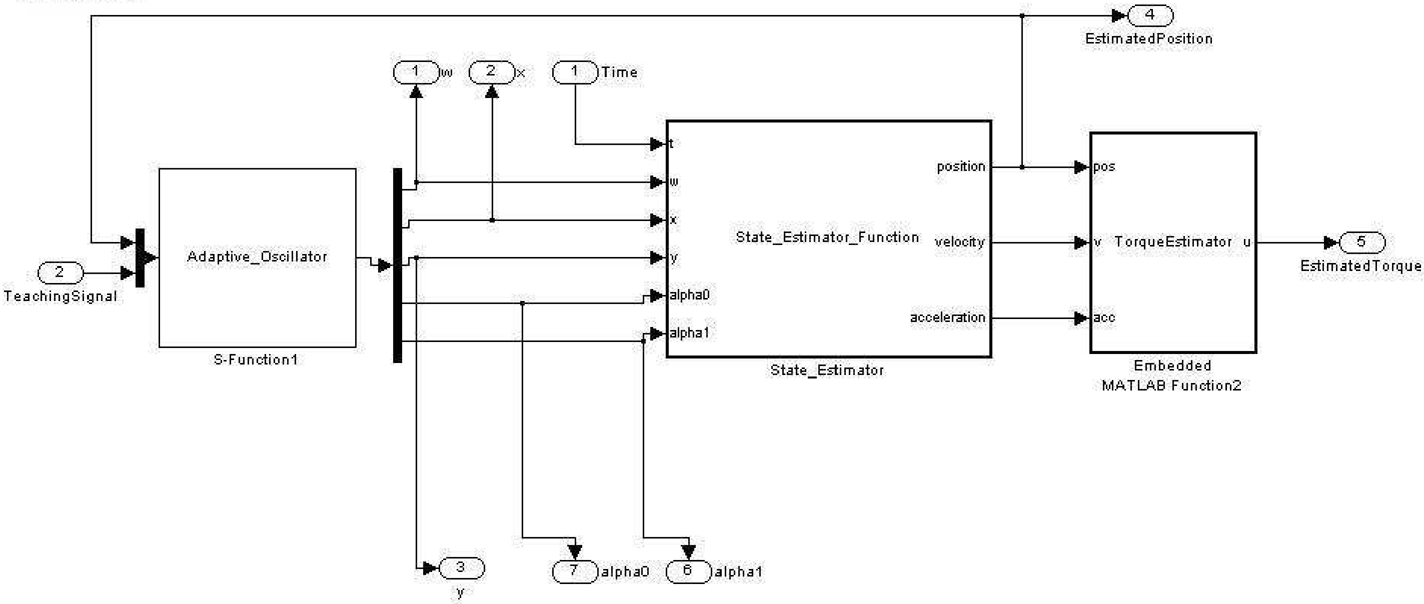
[5] A. Gams, A. J. Ijspeert, S. Schaal and J. Lenarcic. On-line learning and modulation of periodic movements with nonlinear dynamical systems. Auton Robot, 27: 3..23, 2009.

[6] S. Hesse, H. Schmidt, C. Werner and A. Bardeleben. Upper and lower extremity robotic devices for rehabilitation and for studying motor control. Curr Opin Neurol, 16(6): 705..710, 2003. doi:10.1097/01.wco.0000102630.16692.38.

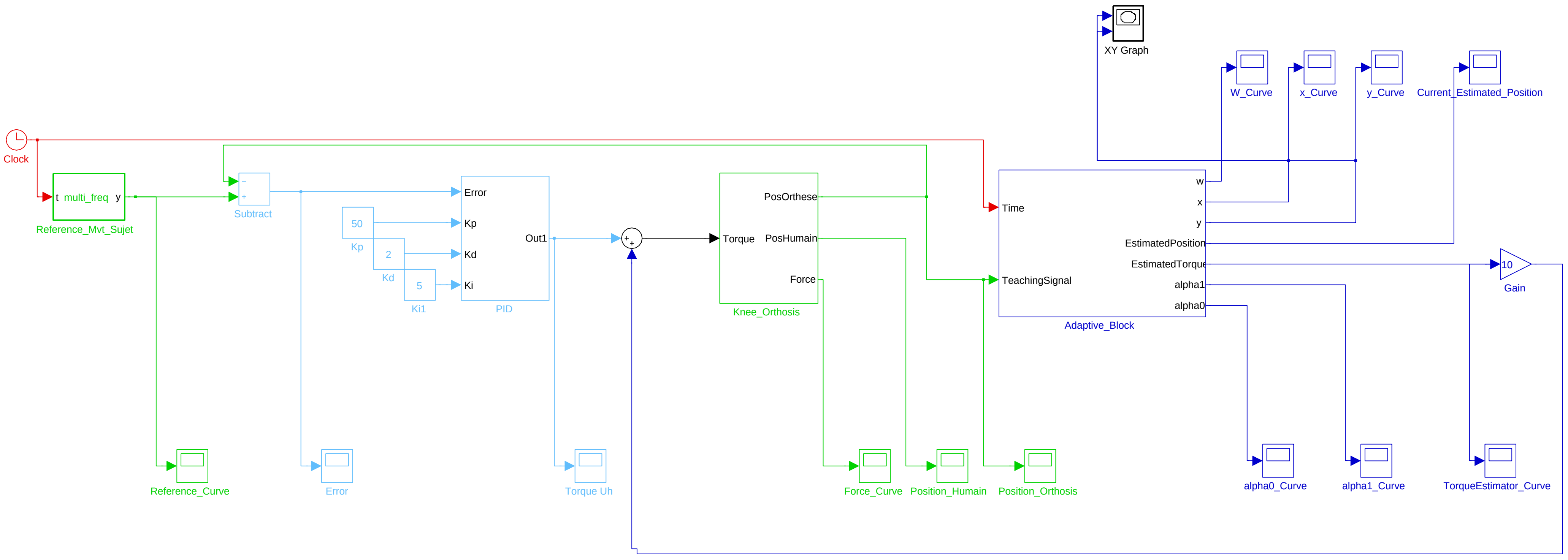
[7] M. Bouri, Member, IEEE, B. Le Gall, R. Clavele. A new concept of parallel robot for rehabilitation and fitness: The Lambda

Appendix

ANNEX 1



4
EstimatedPosition
1 w
2 x
1 Time
2
TeachingSignal
Adaptive_Oscillator
S-Function1
t
w
x
y
alpha0
alpha1
State_Estimator_Function
position
velocity
acceleration
State_Estimator
pos
v
TorqueEstimator
u
acc
Embedded
MATLAB Function2
5
EstimatedTorque
3
y
7 alpha0
6 alpha1


XY Graph
W_Curve
x_Curve
y_Curve
Current_Estimated_Position
Clock
t
multi_freq
y
Reference_Mvt_Sujet
Subtract
Error
Kp
Kd
Ki
Out1
50
Kp
2
Kd
5
Ki1
PID
Torque
PosOrthese
PosHumain
Force
Knee_Orthosis
Time
TeachingSignal
w
x
y
EstimatedPosition
EstimatedTorque
alpha1
alpha0
Adaptive_Block
10
Gain
Reference_Curve
Error
Torque Uh
Force_Curve
Position_Humain
Position_Orthosis
alpha0_Curve
alpha1_Curve
TorqueEstimator_Curve

```
ANNEX  : [3]
/*-----------------------------------------------------------------*
 * PROJET : Rehabilitation robotics using CPG
 *-----------------------------------------------------------------*
   Author       : Sarah MOUSSOUNI
   Supervisor   : Mohamed BOURI (LSRO Lab)
                  Renaud RONSSE (BioRob Lab)

   Date         : Feb - July 2010

   EPFL         : Ecole Polytechnique Federale de Lausanne
 *---------------------------------------------------------------*/


#include "Interpolation.h"
#include "math.h"

/*==========================*/
/*      Interpolation       */
/*==========================*/

//****************************************************************************80

void interval ( int n, double x[], double xval, int *left, int *right )

//****************************************************************************80
//
//  Purpose:
//    interval searches a sorted array for successive brackets of a value.
//  Discussion:
//    If the values in the vector are thought of as defining intervals
//    on the real line, then this routine searches for the interval
//    nearest to or containing the given value.
//
//    It is always true that RIGHT = LEFT+1.
//
//    If XVAL < X[0], then LEFT = 1, RIGHT = 2, and
//      XVAL   < X[0] < X[1];
//    If X(1) <= XVAL < X[N-1], then
//      X[LEFT-1] <= XVAL < X[RIGHT-1];
//    If X[N-1] <= XVAL, then LEFT = N-1, RIGHT = N, and
//      X[LEFT-1] <= X[RIGHT-1] <= XVAL.
//  Parameters:
//    Input, int N, length of input array.
//    Input, double X[N], an array that has been sorted into ascending order.
//    Input, double XVAL, a value to be bracketed.
//    Output, int *LEFT, *RIGHT, the results of the search.
//
{
  int i;

  for ( i = 2; i <= n - 1; i++ )
  {
    if ( xval < x[i-1] )
    {
      *left = i - 1;
      *right = i;
      return;
    }

   }

  *left = n - 1;
  *right = n;

  return;
}
//****************************************************************************80
/*  SPLINE_QUADRATIC evaluates a quadratic spline at a point */
void spline_quadratic ( int ndata, double tdata[], double ydata[], double tval, double *
    yval, double *ypval )

//****************************************************************************80
//  Parameters:
```

```
//
//    Input:
//      + int NDATA, the number of data points defining the spline. NDATA should be odd
    and at least 3.
//      + double TDATA[NDATA], YDATA[NDATA], the values of the independent
//      and dependent variables at the data points.  The values of TDATA should
//      be distinct and increasing.
//      + double TVAL, the point at which the spline is to be evaluated.
//
//    Output:
//      + double *YVAL the value of the spline
//      + double *YPVAL, its first derivative dYdT at TVAL.
//      + YPVAL is not reliable if TVAL is exactly equal to TDATA(I) for some I.


{
  double dif1;
  double dif2;
  int left;
  int right;
  double t1;
  double t2;
  double t3;
  double y1;
  double y2;
  double y3;

//
//  Find the interval [ TDATA(LEFT), TDATA(RIGHT) ] that contains, or is
//  nearest to, TVAL.
//
  interval( ndata, tdata, tval, &left, &right );
//
//  Force LEFT to be odd.
//
  if ( left % 2 == 0 )
  {
    left = left - 1;
  }
//
//  Copy out the three abscissas.
//
  t1 = tdata[left-1];
  t2 = tdata[left  ];
  t3 = tdata[left+1];

/*  if ( t2 <= t1 || t3 <= t2 )
  {
    cout << "\n";
    cout << "SPLINE_QUADRATIC_VAL - Fatal error!\n";
    cout << "  T2 <= T1 or T3 <= T2.\n";
    exit ( 1 );
  }*/


//
//  Construct and evaluate a parabolic interpolant for the data
//  in each dimension.
//
  y1 = ydata[left-1];
  y2 = ydata[left  ];
  y3 = ydata[left+1];

  dif1 = ( y2 - y1 ) / ( t2 - t1 );

  dif2 = ( ( y3 - y1 ) / ( t3 - t1 )
       - (   y2 - y1 ) / ( t2 - t1 ) ) / ( t3 - t2 );

  *yval = y1 + ( tval - t1 ) * ( dif1 + ( tval - t2 ) * dif2 );
  *ypval = dif1 + dif2 * ( 2.0 * tval - t1 - t2 );

  return;
}
```

```
ANNEX [4]
/*-----------------------------------------------------------------*
 * PROJET : Rehabilitation robotics using CPG
 *-----------------------------------------------------------------*
   Author       : Sarah MOUSSOUNI
   Supervisor   : Mohamed BOURI (LSRO Lab)
                  Renaud RONSSE (BioRob Lab)

   Date         : Feb - July 2010

   EPFL         : Ecole Polytechnique Federale de Lausanne
 *----------------------------------------------------------------*/


#include "VariableG.h"
/*------------------------------------------------------------------------ */
/*                      Orthosis characteristic                           */
/*------------------------------------------------------------------------ */

double masse             = 0.3;      // Masse of the orthosis leg
double g                 = 9.81;     // Gravity
double length_leg        = 0.3;      // Longueur pied de levier


/*------------------------------------------------------------------------ */

double sampling_period;


// Variable :Transparent mode
//***************************

double analogForce_V    =0;     // Force sensor value
double TMeasured        =0;     // Measured torque
double TMeasuredFiltred =0;     // Filtred measured torque


// Filtering parameter
double Bp=20;                   // Bande passante

// For the gravitational torque
double Torque_ip=0;
double torque_BH=0;             // Torque measured with a move from bottom to high
double torque_HB=0;             // Torque measured with a move from high to bottom
double Torque_grav  =0;         // Gravitational Torque
double Torque_grav_2    =0;     // Gravitational Torque


// Closed loop
double Kp=0;                    // Proportional gain
double Ki=0;                    // Integral gain
double u_i=0;                   // Integral gain (limited)
double u_i_bounded=0;
double previous_error = 0;



double Force_desiree=0;
double Error_Torque =0;         // Desired Force  - Filtred Force
double Torque_App   =0;         // Applied Torque for the transparent mode
double Torque_Send  =0;         // Total torque to send




// Variable :Adaptive freqiency block
//**************************************



// HOPF oscillator variables
//****************************
```

```
long double w_prec=10       ,w_;                // Old value of w and current value
long double x_prec=1        ,x_;                // Old value of x and current value
long double y_prec=0        ,y_;                // Old value of y and current value
long double Offset_prec=0   ,Offset;        // Old value of Offset and current value
long double Amplitude_prec=0,Amplitude;     // Old value of Amplitude and current value


// Estimated variables
//*********************

extern long double estimated_Signal=0;
extern long double estimated_Velocity=0;
extern long double estimated_Acceleration=0;

extern double Estimated_Torque=0;
extern long double Estimated_Human_Torque=0;
extern double Torque_Augmentation=0;
```

ANNEX [5]

```
/*-----------------------------------------------------------------*
 * PROJET : Rehabilitation robotics using CPG
 *-----------------------------------------------------------------*
   Author       : Sarah MOUSSOUNI
   Supervisor   : Mohamed BOURI (LSRO Lab)
                  Renaud RONSSE (BioRob Lab)

   Date         : Feb - July 2010

   EPFL         : Ecole Polytechnique Federale de Lausanne
 *---------------------------------------------------------------*/

#include "ForceProcessing.h"
#include "math.h"


/*===================================*/
/*      Orthosis characteristic      */
/*===================================*/
/*
 Caracteristique de l'orthese:
 Calcule du couple de force correspondant au voltage mesuré par
 les capteur de force
 */


double VoltToForce (double voltage){

    double Force_Measured=0;
    int Nb_Mesure=17;
    double Force[17]  ={   -5,    -4,    -3,    -2,    -1,-0.5,     0,   0.5, 1, 2,   3,   4,    5,
       6,    7,    8,   10};
    double Tension[17]={-0.85,-0.75,-0.67,-0.55,-0.41,-0.38,-0.23,-0.21,-0.1, 0,0.1,0.2,0.
    37,0.58,0.65,0.77,0.96};

     //****** interpolation
    double Fval;
    double Fpval;

    //void spline_quadratic_val ( int ndata, double tdata[], double ydata[], double tval,
    double *yval, double *ypval )
    spline_quadratic( Nb_Mesure,Tension, Force, voltage, &Fval,&Fpval );

    Force_Measured =Fval;

    return Force_Measured;
}


/* Caracteristique voltage - position */


/*
    PositionToVolt  :   function compute according to a set of measurement
                        the voltage corresponding to the input position
                        This voltage (Torque) make the orthosis stable
                        in the position
    @Input          :
        Desired_position : the deseared position
        Direction : precise the direction of the orthosis  movement
                    '1' : velocity <0 from bottom to high
                    '2' : velocity >0 from high   to bottom
    @Output
        Torque (voltage) : which make the orthosis able to stay stable on the input
    position

*/
double PositionToVolt (double Desired_position, int Direction){


    int Nb_Mesure=11;
```

```
    double Theta[11]        ={-80  ,-70  ,-60  ,-50  ,-40  ,-30    ,-20    ,-10  ,     0, ↙
    10   ,20  };
    // Decendre vel >0
    double Tension_HB[11]   ={-0.31,-0.40,-0.27,-0.26,-0.24,-0.2300,-0.1800,-0.16,-0.120,-↙
    0.10,-0.08};
    // Monter   vel <0
    double Tension_BH[11]   ={-0.09,+0.08,-0.05,-0.06,-0.04,-0.0106,-0.0156,+0.02,+0.052,+↙
    0.10,+0.16};

     //****** interpolation
    double Tval;
    double Tpval;
    double Voltage_Measured=0;

   if (Direction == 1) // Vel <0
        spline_quadratic( Nb_Mesure,Theta,Tension_BH,  Desired_position, &Tval,&Tpval );

    else if (Direction == 2) // val >0
        spline_quadratic( Nb_Mesure,Theta,Tension_HB,  Desired_position, &Tval,&Tpval );

    else
        printf(" Invalid direction %d : '1' for inc , '2' for dec ... \n ", Direction);

    Voltage_Measured =Tval;

    return Voltage_Measured;

}


/*
    ForceAppToTension:  function compute the torque to send to have the deseared force

    @Input          :
        ForceDesiree : the force that we want to apply
    @Output
        Torque to apply to have the deseared force

*/

double ForceAppToTension (double ForceDesiree){

    int Nb_Mesure=10;
    double TensionEntree[10]={  -4,    -3,    -2,    -1,     0,     1,     2,     3,     4,     5}↙
    ;
    double ADC[10]          ={-1.40,-0.90,-0.70,-0.30,+0.29,+0.90,+1.80,+1.40,+2.16,+2.80}↙
    ;
    double ForceADC[10];

    double Tval;
    double Tpval;

    for (int i=0; i<Nb_Mesure; i++)
    {
        ForceADC[i]=VoltToForce(ADC[i]);
    }


    spline_quadratic( Nb_Mesure,ForceADC,TensionEntree, ForceDesiree, &Tval,&Tpval );


    return Tval;

}



/*===================================*/
/*         Force processing          */
/*===================================*/

//---------------------------------------------------------------------------------------↙
    --------------------------------
//Fonction Calculant la torque humain mesuré
```

```
/*===============================================================================
    ==============================*/

/* Fonction de mesure du torque appliqué par l'utilisateur*/
double MeasureTorqueValue(double ADC)
{
    double  to_f, a1_f, b1_f;

    to_f = 1.0/20.0;

    // Etalonage
    analogForce_V = ADC;  //ReadAnalogValue  (0)
    TMeasured= VoltToForce(analogForce_V) * length_leg -0.052;

    // FILTRAGE
    a1_f = to_f/(sampling_period + to_f);
    b1_f = sampling_period/(sampling_period + to_f);
    TMeasuredFiltred = a1_f * TMeasuredFiltred + b1_f * TMeasured ;

    return TMeasuredFiltred;
}
```

ANNEX [6]

```
/*-----------------------------------------------------------------*
 * PROJET : Rehabilitation robotics using CPG
 *-----------------------------------------------------------------*
   Author       : Sarah MOUSSOUNI
   Supervisor   : Mohamed BOURI (LSRO Lab)
                  Renaud RONSSE (BioRob Lab)

   Date         : Feb - July 2010

   EPFL         : Ecole Polytechnique Federale de Lausanne
 *----------------------------------------------------------------*/

#include "Adapatateur_fraquence.h"
#include "math.h"

/*****************************************************************************************
    *****************************/
// Variables Globales

    long double epsilon =   1;
    long double mu      =   1;
    long double n       =   5;

/*=========================================================================================
    ==============================*/
/* Partie HOPF oscillator */
/*=========================================================================================
    ==============================*/

/* Function of x state variable*/
long double fx (long double w, long double x, long double y, long double position_)
{
    long double r=sqrt(pow(x,2)+pow(y,2));
    return (1-pow(r,2))* x - w*y+ epsilon* position_;
}

/* Function of y state variable*/
long double fy (long double w, long double x, long double y )
{
    long double r=sqrt(pow(x,2)+pow(y,2));
    return (1-pow(r,2))* y +  w * x;
}

/* Function of w state variable*/
long double fw (long double w, long double x, long double y, long double position_)
{
    long double r=sqrt(pow(x,2)+pow(y,2));
    return - epsilon * position_ *(y/r);
}

long double fOffset (long double w, long double x, long double y, long double
    estimated_position)
{
    return n * estimated_position;
}

long double fAmplitude (long double w, long double x, long double y , long double
    estimated_position)
{
    return n * x * estimated_position;
}

double Hopf_simple(double position, double estimated_position ,double sampling_time)
{



        x_= x_prec + fx(w_prec,x_prec,y_prec,position - estimated_position) *
    sampling_time;
        y_= y_prec + fy(w_prec,x_prec,y_prec) * sampling_time;
        w_= w_prec + fw(w_prec,x_prec,y_prec,position - estimated_position) *
    sampling_time;
```

```
        if (w_ <= 0.1)    w_=10; // 1.5 * 3.14;


        Offset  = Offset_prec       + fOffset(w_prec,x_prec,y_prec,position - 
    estimated_position) * sampling_time;
        Amplitude   = Amplitude_prec    + fAmplitude(w_prec,x_prec,y_prec,position - 
    estimated_position) * sampling_time;

        w_prec=w_;
        x_prec=x_;
        y_prec=y_;
        Offset_prec = Offset;
        Amplitude_prec  = Amplitude;

    return w_;
}


/*=================================================================================
    ==============================*/
/* Partie Estimator */
/*=================================================================================
    ==============================*/



int Learned_Signal (long double w , long double x , long double y, long double  Offset, 
    long double Amplitude)
{

    //position=alpha0+alpha1*x;
    estimated_Signal        = Offset  + Amplitude * x;

    //velocity=alpha1*w*y;
    estimated_Velocity      = - Amplitude * w * y;

    //acceleration=-alpha1*(w^2)*x;
    estimated_Acceleration  = - Amplitude * x * w * w;

    return 0;
}



/*=================================================================================
    ==============================*/
/* Partie Torque Estimator */
/*=================================================================================
    ==============================*/


double EstimatedTorqueFct ()
{


    double grav ;
    double visco;
    double inertie_Acc;
    double inertie_totale;

    double RapRed           = 120;       // Rapport de reduction
    double inertie_charge   = (masse * length_leg * length_leg)/3;
    double inertie_moteur   = 0.134 * 0.0001;
    double inertie_HD       = 0.360 * 0.0001;

    grav        = masse * g * length_leg * sin(estimated_Signal);

    if (estimated_Velocity >= 0 )
        visco       = -0.35 * length_leg;
    else
        visco       = +0.30 * length_leg;
```

```
    visco =0;

    inertie_Acc = masse * length_leg * length_leg * estimated_Acceleration;

        Estimated_Torque = ((grav + visco + inertie_Acc)/RapRed);
        Estimated_Human_Torque = Estimated_Torque - Torque_Augmentation -Torque_Send;

    return Estimated_Torque;
}
/*************************************************************************/

// USING A REFERENCE MOVEMENT

/*************************************************************************/
double InputSignal (double freq, double Amplitude, double time, double tmp)
{

    double v=0;

    v= Amplitude * sin (freq * (time/tmp));

    return v;
}


double error_estimation=0;
double ControllerAdaptiveSystem (int x_idx_, double* Param__, double
    Position_measured_art)
{

    static st_=0;

    double freq         = Param__[1];
    double Amplitude    = Param__[2];
    double tmp          = Param__[3];

    if (st_==0)
    {
        time_ = 0;
        st_=-1;
    }


     /* position sin */

    v_tension = InputSignal ( freq, Amplitude, time_, tmp);

    time_ = time_ +sampling_period ;

    /* Hopf oscillator */

    // Position signal is the learned signal
    error_estimation =  Position_measured_art - estimated_Signal;
    Hopf_simple(Position_measured_art,estimated_Signal,sampling_period);

    /* reconstruction du signal*/
    Learned_Signal (w_ , x_ , y_ , Offset , Amplitude);

    /* Estimated torque */
    EstimatedTorqueFct ();

    return v_tension;
}

/*=====================================================================*/
```