Design and control of a quadruped cheetah robot with compliant spine

N. (Nicolò) Botteghi

MSc Report

Committee:
Prof.dr.ir. S. Stramigioli
Dr.ir. G.A. Folkertsma
Prof. L. Marconi

February 2017

013RAM2017

Robotics and Mechatronics
EE-Math-CS
University of Twente
P.O. Box 217
7500 AE Enschede
The Netherlands

ALMA MATER STUDIORUM - UNIVERSITÀ DI BOLOGNA

SCUOLA DI INGEGNERIA E ARCHITETTURA

DIPARTIMENTO di

INGEGNERIA DELL'ENERGIA ELETTRICA E DELL'INFORMAZIONE

"Guglielmo Marconi"

DEI

CORSO DI LAUREA MAGISTRALE IN INGEGNERIA DELL'AUTOMAZIONE

TESI DI LAUREA MAGISTRALE

in

Automatic Control And System Theory M

Design and control of a quadruped cheetah robot with compliant spine

CANDIDATO
Nicolò Botteghi

RELATORE
Chiar.mo Prof. Lorenzo Marconi

CORRELATORI
Chiar.mo Prof. Stefano Stramigioli
G.A. Folkertsma

Anno Accademico 2015/16

Sessione III

...there are no problems,
there are just solutions
waiting to be found...

Acknowledgments

There are a lot of people that I would like to thank because without them I wouldn't be able to conclude and success in my project.

Firstly, I would like to thank Professor Lorenzo Marconi and Professor Stefano Stramigioli because they gave my the possibility to spend the last six months at University of Twente and without them I couldn't even start my research.

Secondly, but not for importance, my direct supervisor Geert Folkertsma for the assistance provided me and above all for his availabilty and his regard.

Then I need to thank Gerben te Riet for all the help regarding the electronics, the laser-cutted and 3D-printed mechanical parts of the robot, Marcel Schwirtz that taught me some basical and practical electronics skills and all the staff and students of RAM group for the beautiful time that I was able to spend.

Finally, I would like to thank my family for providing me support and continuous encouragement throughout these six months of thesis.

This accomplishment would not have been possible without them.

Thank you!!!

Contents

Chapter 1

Introduction

The aim of the project is to build a quadruped robot than can run fast and in an energy efficient way. Because we would like to achieve energy efficiency it is natural to take a look at the animal world and because the other goal is to achieve fast running we have chosen the cheetah as model for our robot. The idea is to build a robot in which the locomotion is provided by body, the mechanical design and brain, the control system, in order to mimic what happens in nature. Unlike most of the renowned quadruped robots [10, 11, 13], in which the main body is rigid and the legs are multi-actuated, this robot generates locomotion using oscillators, the legs, coupled to a compliant spine to allow relative motion between the limbs. The work of the brain is to provide synchronization of the legs and to control the energy injected through springs into the system. With a relatively simple mechanical structure and a small amount of energy injected every period, thanks to the presence of numerous passive elements, we are able to generate locomotion. The first step was change and improve the simulation model in order to strongly speed-up the simulations, then, starting from the mechanical structure of a two-legged cheetah robot [7], we redesigned and enhanced the controller to make it able to control the energy of the system then we coupled two identical two-legged robots in order to obtain a four-legged robot and eventually a controller, with the same principles of the previous one, was implemented to allow running. A very suitable approach to model these kind of systems [5, 6], in which the main focus is on the energy and the power that bodies exchange, is using the Port-Hamiltonian systems theory and the Bond-Graph notation [4]. The whole model was built adopting these tools. Firstly, we are going to present the theoretical instruments necessary and used for the realization of the simulation models (Chapter 2), secondly, an analysis of the two-legged model and prototype will be shown including the improved model and the controllers implemented (Chapter 3), thirdly, we are going to exhibit the work on the four-legged simulation model and prototype (Chapter 4). Last, but not least we are going to discuss the results obtained, with advantages and disadvantages of the proposed solutions and possible improvements (Chapter 5). We propose few appendices in order to highlight some interesting aspects of the project as the Dirac structures, the electronic and electric structure, the ground-contact model implemented and the gait analysis (Appendices A, B, C, D).

Chapter 2

Theoretical background

The studies has been done by using an energetic approach of modelling dynamical systems, that is described pretty well by the so-called Bond-Graph notation, the homogeneous matrices and the concepts of twist and wrench for a rigid body [4].

2.1 Twist and wrench for a rigid body

In order to talk about twist and wrench for a rigid body, we need first to introduce the notion of homogeneous matrix.

2.1.1 Homogeneous coordinates

Let's start by considering a generic point P and its coordinates in the Euclidean space.

$$P = \begin{pmatrix} p_x \\ p_y \\ p_z \end{pmatrix} \tag{2.1}$$

To describe the generic motion, the rotational matrices are not enough because we need to include also translations. Thus, we need to move first to the so-called homogeneous coordinates and we can equivalently write the coordinates of P by just adding a fourth coordinate equal to 1.

$$P = \begin{pmatrix} p_x \\ p_y \\ p_z \\ 1 \end{pmatrix} \tag{2.2}$$

It is worth to highlight that with this representation, we can characterize even points at infinity by simply setting the fourth coordinate to 0.

2.1.2 Homogeneous matrices

Let's suppose now to know the coordinates of point P with respect to the reference system Ψ_i, the relative rotation, described by the rotational matrix R_i^j and the relative translation, described by the vector O_i^j, with respect to another reference system Ψ_j. How can we express P in Ψ_j? We can easily answer if we introduce the homogeneous matrices.

$$P^j = H_i^j P^i \tag{2.3}$$

where

$$H_i^j = \begin{pmatrix} R_i^j & O_i^j \\ 0_3 & 1 \end{pmatrix} \tag{2.4}$$

The matrix H_i^j is 4 by 4 and it is composed by a rotational matrix 3 by 3, R_i^j, a translational vector 3 by 1, O_i^j, a 0-vector 1 by 3, 0_3 and a scalar term, 1. Thanks to the structure , the homogeneous matrices are invertible and we can also apply the so-called chain rule, that is very useful in robotic modelling. We need to set just a starting point and then build the whole transformations of the model by only multiply matrices.

$$(H_i^j)^{-1} = H_j^i \tag{2.5}$$

$$H_a^n = H_m^n H_l^m ... H_a^b \tag{2.6}$$

From a geometrical point of view, R belongs to the special orthonormal group, $SO(3)$, that for definition is the group of matrices with determinant equal to 1 and in which $R^{-1} = R^T$, while H belongs to the special euclidean group, $SE(3)$, where:

$$SE(3) = \{\begin{pmatrix} R & O \\ 0_3 & 1 \end{pmatrix} \textit{ such that } R \in SO(3),\ O \in \mathbb{R}^3\} \tag{2.7}$$

2.1.3 Twist

The velocity of point mass P in the Euclidean space with respect to an observer not moving with respect to Ψ can be easily computed by the time derivative of its coordnates.

$$(x, y, z) \in \mathbb{R}^3 \longmapsto (\dot{x}, \dot{y}, \dot{z}) \in \mathbb{R} \tag{2.8}$$

For rigid bodies everything is much more complicated because in the most general case we can have 3 rotations and 3 translations. However we can associate a configuration of the body to an homogeneous matrix. If we consider Ψ_i fixed to body I and Ψ_j fixed to body J, it is possible to describe their relative configuration by the relative configuration of the 2 frames which is represented by H_i^j using their change of coordinates. We can also see that H_i^j represents the change of coordinates from frame Ψ_i to Ψ_j, but also the physical motion which brings Ψ_j to Ψ_i for points expressed in either of the 2 frames. As we said before, for point mass, the velocity is easily computed by the time derivative of its coordinates, so one could think that this holds also for H-matrices, but actually is not true because:

- $\dot{H}$ has many more elements than necessary to express 6-dimensional infinitesimal motion
- from the information of $\dot{H}$ would not possible to have an idea of the relative motion without knowing H
- if we consider interconnected bodies, the velocity $\dot{H}$ could not be used since each of the bodies would have a different configuration H and different $\dot{H}$

So in order to describe the motion of a rigid body we will introduce the notion of twist. A twist or generalized velocity is a real geometrical object, belonging to $se(3)$ which describes the relative instantaneous motion of the body rigidly connected to frame Ψ_i with respect to the body rigidly connected to frame Ψ_j expressed numerically as a vector in the frame Ψ_k:

$$ {}^kT_i^j = \begin{pmatrix} \omega \\ v \end{pmatrix} \tag{2.9} $$

or in the matrix form as:

$$ {}^k\tilde{T}_i^j = \begin{pmatrix} \tilde{\omega} & v \\ 0_3 & 0 \end{pmatrix} \tag{2.10} $$

The twist vector T is composed by the vector that represents the angular velocities, ω and the vector that represents the linear velocities v. It is also possible to change coordinates of a twinst using the so-called Adjoint matrix:

$$ {}^kT_i^j = A_{dH_l^k} {}^lT_i^j \tag{2.11} $$

where

$$ A_{dH_l^k} = \begin{pmatrix} R_l^k & 0 \\ \tilde{p}_l^k R_l^k & R_l^k \end{pmatrix} \tag{2.12} $$

An important feature of twists is that they are independent of the configuration of the body and can therefore be used to describe the relative motions of any body and are the key to define power ports in rigid multi-body mechanical systems.

2.1.4 Wrench

The dual element of a twist, T, is a wrench of generalized force, W. Its dimension is the same as a twist, it belongs to $se^*(3)$ and its product with a twist correspond to a scalar quantity that represents the power exchanged by a wrench with relative motion represented by the twist. A wrench in a vector form can be represented by:

$$
{}^{k}W_i^j = \begin{pmatrix} \tau & f \end{pmatrix} \tag{2.13}
$$

while the power:

$$
P = WT = \begin{pmatrix} \tau & f \end{pmatrix} \begin{pmatrix} \omega \\ v \end{pmatrix} = \tau\omega + vf \tag{2.14}
$$

where τ represents the torque and f the linear force transmitted.

2.2 Energetic approach of modelling physical systen: the Bond-Graph notation

First of all we need to introduce the notion of port. A port is simply an interaction between submodels. In the physical domain such interaction is coupled to an exchange of energy i.e. power. In a domain-free terminology we can call it power bond. The power in every domain can be expressed as the product of the variables named effort e and flow f:

$$
P = e \times f \tag{2.15}
$$

In the physical domain, for example, the effort is a force or in a more general framework a wrench while the flow is a velocity or again in the most general case a twist. In this context we can now introduce the Bond-Graph notation that fits very well with the problem of modelling dynamical system by using an energetic approch.
Let's start by defining the basic elements that we can use to build a model.

2.2.1 Edges

The edges of the graph are the bond and they represents the signals of effort and flow that the nodes of the graph exchange. Their grafical representation is an half-arrow:

$$
\rightharpoondown \tag{2.16}
$$

Positive orientation of the half-arrow

The half-arrow just represents positive orientation of the power that the nodes of the graph exchange.

2.2.2 Nodes

Then we can define the types of nodes of the graph and we can classify them in five groups of basic physical behaviour.

Storage type

Storage elements are the one that can store energy reversibly and they are of two type:

- C-type elements
- I-type elements

C-type storage elements are the ones in which the flow is integrated into a generalized dispacement and related to the conjugate effort while I-type are the ones in which the effort is integrated in a generalized momentum and related to the conjugate flow. These two elements are dual in sense that they can be transformed in each other by interchanging the roles of the conjugate variables effort and flow. C-types are:

- ideal spring
- ideal capacitor
- ideal reservoir
- ideal heat capacitor

While I-types are:

- ideal mass
- ideal inductor
- ideal fluid inertia

Irreversible transformations

Irreversible transformantions are 1-port elements that dissipates energy. The term "irreversible" comes from the entropy principle of thermodynamics and it means that they are elements that are going to increase the entropy. Some examples are:

- ideal electric resistor
- ideal friction
- ideal fluid resistor
- ideal heat resistance

But also more complex ones as:

- diodes
- non linear friction in a mechanical contact

Reverisble transformations

Reversible transformations can be divided in 2 categories: the so-called tranformers, TF, if they relate the effort of both port or the flow of both port and the so-called gyrators, GY, if they relate the flow of one port with the effort of the other or viceversa. Some examples of transformer are:

- ideal electric transformer
- ideal lever
- ideal gearbox
- ideal piston-cylinder combination
- ideal positive displacement pump

And some examples of gyrators are:

- ideal centrifugal pump
- ideal turbine
- ideal electric motor

Supply and demand

Supply and demand are elements that can generate or drain energy. They can also be considered storage elements that are infinitely large with respect to the storage process of interest. They can be divided in source of effort, Se and sorce of flow, Sf. Some examples of sources of effort are:

- ideal voltage source
- ideal pressure source

While sources of flow are:

- ideal current source
- ideal velocity source

Distribution

The 0-junction and the 1-junction are the most powerful features of the Bond-Graph representation. In a 0-junction all efforts have to be identical and the flows sum equal to 0 while, dually, in a 1-junction all flows have to be identical and the efforts sum equal to 0.

2.2.3 Causalities

Ports and bonds show that two bilateral signals are involved in a relation, but there is no need to make a priori choice of the direction of the corresponding signals. A particular choice of the direction or causality is needed before a set of computable relations can be found or some particular analysis can be performed. The symbol is called "causal stroke" and it indicates where the effort signal comes out i.e. where it enters the connection port. In figure 2.1 we can see examples of causal strokes.

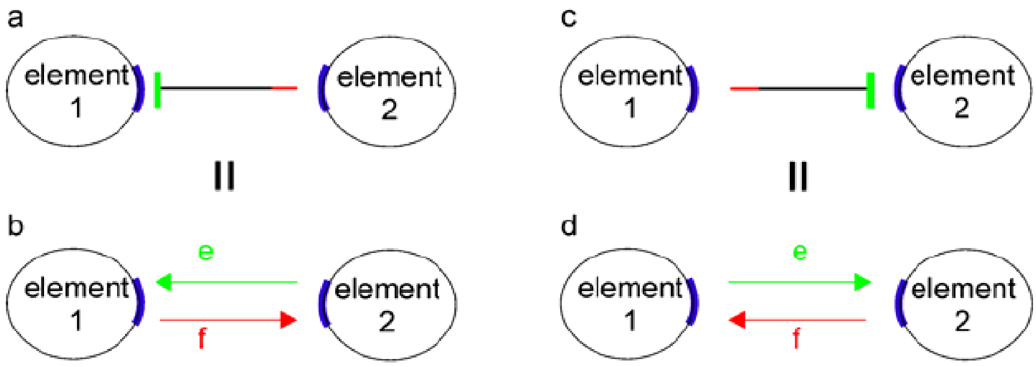

Figure 2.1: Examples of causal strokes. The effort direction (green arrow) determines the position of the stroke (green), while the flow (red arrow) will get the opposite direction. Figure reproduced from [4].

Fixed causalities

Some nodes has fixed causalities. For example sources of effort always has an effort as output signal i.e. the causal stroke is attached to the end of the bond that is connected to rest of the system. For sources of flow the opposite holds.

Preferred causalities

For some component the integral form is preferred so the causality is preferred too. We prefer the integral much more than the derivative formulation because the latter amplifies noises and also we can't set an initial condition as we can do with the integral. An example is the C-element, a capacitor in particular, that you can see in figure 2.2.

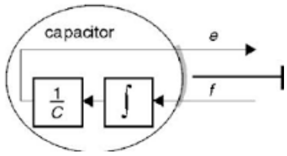

Figure 2.2: Example of preferred causality in which we can see that we prefer and chose the flow as input variable and the effort as output one in order to have the constitutive equation in the integral form. Figure reproduced from [4].

Arbitrary causalities

We deal with arbitrary causalities when the causality in neither fixed or preferred as in the case of irreversible transformations.

$$u = Ri \Leftrightarrow i = \frac{u}{R} \tag{2.17}$$

Chapter 3

Two-legged cheetah robot

The work started on a two-legged cheetah robot [7] that could run on a circumference around a pole, sustained by a boom because it couldn't stand on its own. Despite the running phase was decent, the controller could be far more developed in order to control the energy injected and the stability of the run. Before the design of the new controllers, the software model was simplified in order to speed up heavily the simulations.

3.1 Simulation model

The starting model were developed on 20-sim [1], as you can see in figure 3.1 and it was structured in five principal submodels. These five submodels were chosen to emphatize the legs, the "hoppers", the spine and the support of the robot, the "boomguide" and the connection between them, the "backbones". In more details we will show:

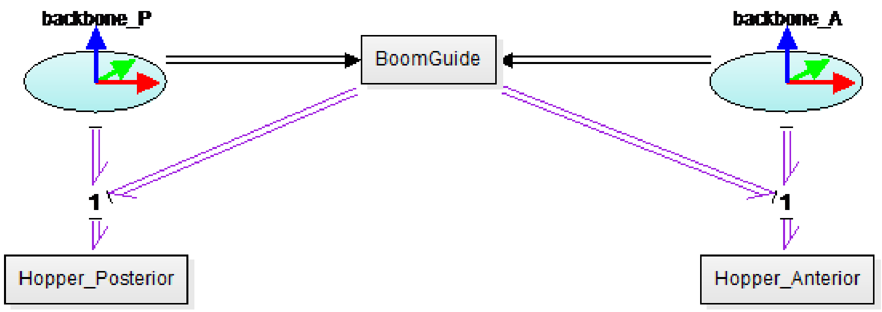

Figure 3.1: 20-sim initial model in which we can highlight the two legs, called HopperPosterior and HopperAnterior, the backbones, i.e. the bodies that connect spine and legs and the BoomGuide that combines the spine joint, the boom and the carriage necessary to substain the robot.

1. BoomGuide: the "boom-guide" block includes the parts relative to the "carriage" and to the "boom" that support the cheetah and to the "spine" joint that connects them. The spine plays a central role in the locomotion so it deserves few words. The elastic spine of the cheetah [14] can be modelled as a rotational joint with springs connected in a way to generate chosen level of stiffness in its free degree of freedom. It is important also to highlight that the asymmetric position of the spine with respect to the legs is fundamental to induce a preferred-direction locomotion. It is noticed that symmetrical configurations of the spine do not generate locomotion, while asymmetric, but lower in terms of height still generate locomotion, but with a smaller speed.

2. backbone P: it is the rigid body that connects posterior leg and spine. It's shape and dimensions are different from the backbone A because we need to introduce asymmetry of the spine to generate locomotion. It's worth to notice that every rigid body of the cheetah is modelled as the scheme in figure 3.2 in which we can highlight six principal elements:

    - I-type storage: it is the block that describes the mass and the inertial properties of that mass.
    - Se or source of effort: it is a force generator and in that specific case it generates the gravity force.

- RTF or rotational transformer: it is a transformation that allow us to set the gravity force always applied toward the vertical axis even if the body rotates.
- MTF or modulated transformer: it is a transformation that allows us to study the body in the baricentral frame where the inertia matrix is diagonal.
- MGY or modulated gyrator: it is a transformation that allows us to represents the Coriolis forces applied on the body.
- 1-junction: it constraints the velocities to be the same and $\sum effort = 0$. It is necessary in order to connect this body with others.

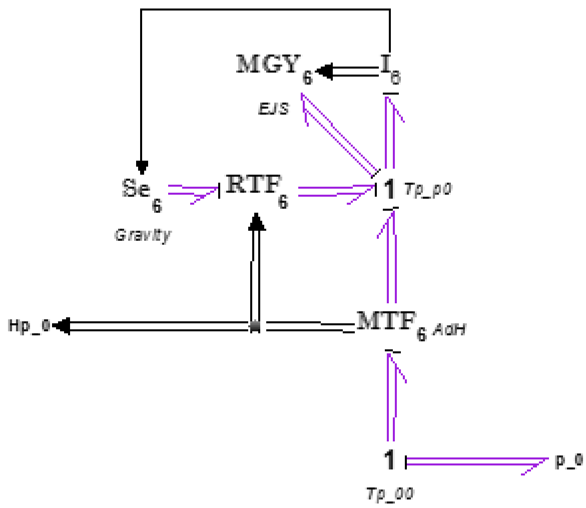

Figure 3.2: 20-sim rigid body model. It includes all the matrices transformations necessary to apply the gravity force in the correct way and to study the body in the reference frame fixed to its baricenter, in which the the inertia matrix is diagonal and more handy.

3. backbone A: identical to backbone P in terms of model, but the parameters of the body, mass and inertia, are different, again to induce the asymmetry of the spine.
4. Hopper Posterior: the "hopper" is the model of the leg it includes five rigid bodies, modelled exacly as the "backbones" and five joints, as

shown in figure 3.3. A model that describes well the behaviour of a leg is called SLIP [15]. Basically, the leg is modelled as a mass-spring system. The idea in our model is to constrain the foot so that it can jump up and down on a straight line while a spring, in series with the foot, accumulates energy, by compressing itself, every impact with the ground. That amount of energy will be given back to the system during the liftoff of the foot. The spring reduces the energy injected by the motor and it is a good way to mimic what muscles do. The bodies that constitute the leg are:

- Pelvis: rigid body fixed to backbone and femur.
- Femur: rigid body in which the servomotor is installed. It is connected rigidly to pelvis and fibula, while it is connected through a rotational joint to the tibia.
- Fibula: together with tibia, they constrain the motion of the foot to a straight line. It is fixed to femur and connected to foot through a revolute joint.
- Tibia: together with fibula, they constrain the motion of the foot to a straight line. Is is connected to foot and femur through two rotational joints.
- Foot: It is the body that it is in contact with the external environment. It is connected to tibia and fibula through two rotational joints.

While the joints are:

- Hip: fixed joint that connects pelvis and femur.
- Fibh: fixed joint that connects fibula and femur.
- Knee: rotational joint that connects femur and tibia.
- Heel: rotational joint that connects foot and fibula.
- Ankle: rotational joint that connects foot and tibia.

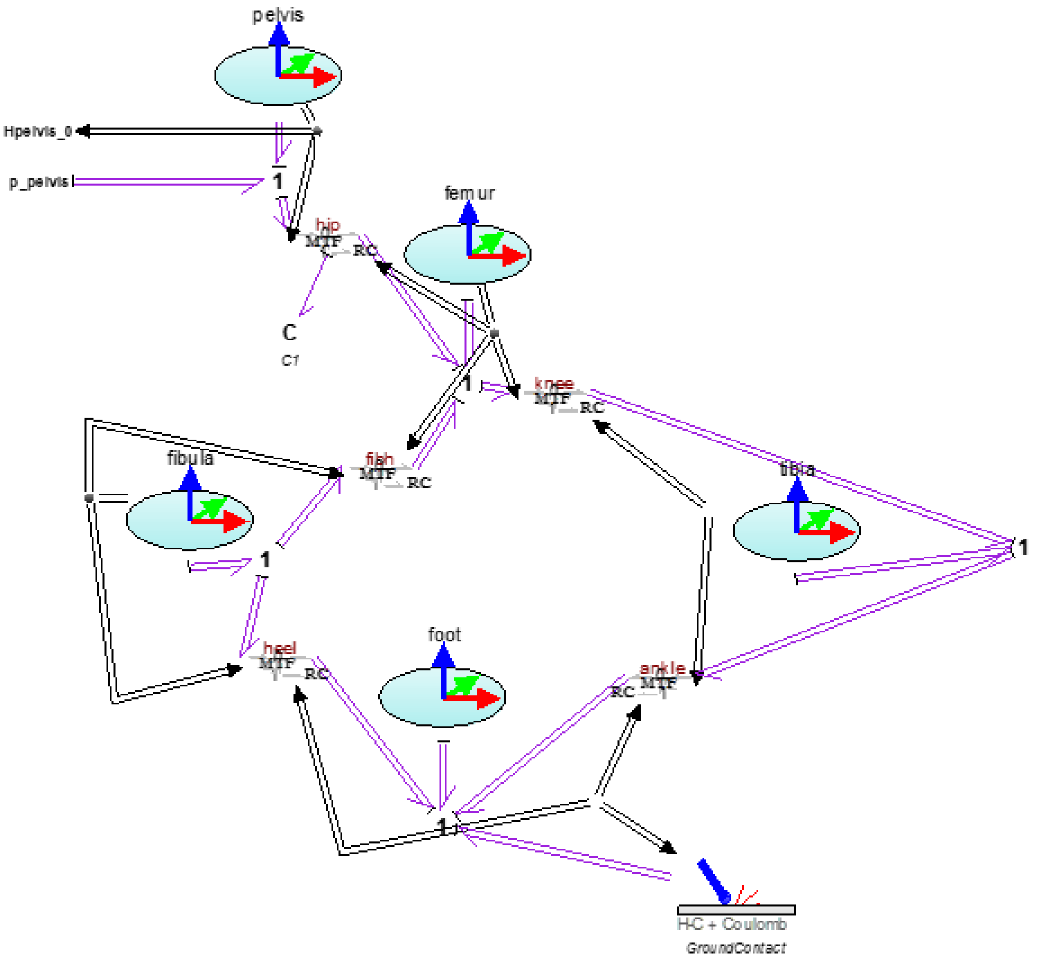

Figure 3.3: 20-sim leg model. We can highlight that the five rigid bodies are connected each other through the joints mentioned above. This model reflects exactly the prototype design.

The joints, for the exception of the "knee" one, were modelled as in figure 3.4. If we overlook at the transformation between reference frames, used to set the center of compliance of the joints and their positions in the space with respect to the other rigid bodies, we can see that they are modelled as resistive-dissipative, R-type, elements connected to C-type storage, i.e. springs.

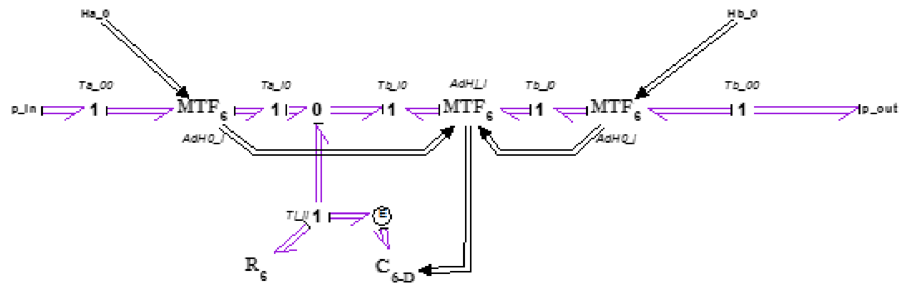

Figure 3.4: 20-sim non-actuated joints model. The joint is modelled as a 6-dimensional elastic-dissipative system with low stiffness in the free motion direction and high stiffness in the constrained one. The MTFs are used to choose the position of the center of compliance of the joints.

The "knee" joint is particular because it is actuated. The motor can inject energy into the system through this joint. Its model almost the same of the non-actuated one, with the exception of an extra transformation, STF, to connect the output power of the motor and in particular its rotational speed, 1-dimensional, with the free degree of freedom of twist the joint, as shown in figure 3.5.

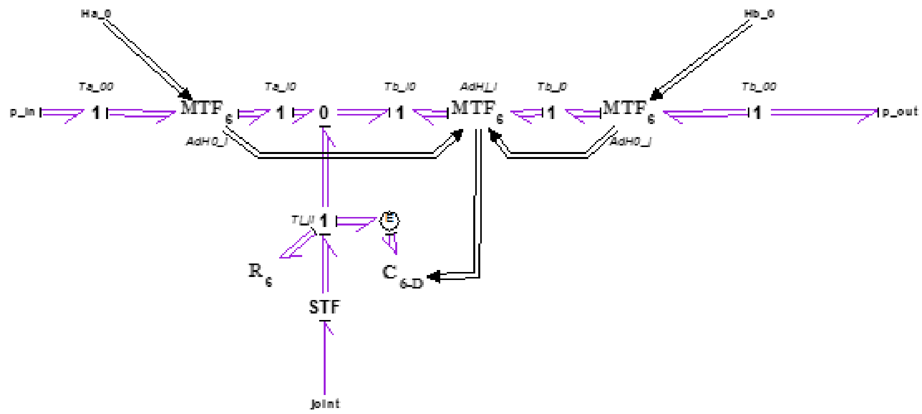

Figure 3.5: 20-sim actuated joints model. The trasformation, STF block, injects in the joint the power coming from the motor. Because the power of the motor is a scalar value, it is also select the suitable degree of freedom in which inject this amount, i.e. the free rotation allowed by the joint.

5. Hopper Anterior: identical to Hopper Posterior in model and physical dimensions.

3.2 Prototype design

The legs of the cheetah were mechanically designed keeping in mind the Evans mechanism, shown in figure 3.6. As result, the leg allows the foot to translate along a line thanks to three revolute joints. The only actuated joint is the knee one, but it is not directly actuated by the motor. The energy is injected into the system through a spring, that is in series with the motor, from here the acronym SEA or series elastic actuation. There is also another spring, in parallel with the previous one, necessary to prevent over-extension of leg. The stiffness of two parallel springs can be expressed by equation 3.1:

$$K_{equivalent} = K_{spring} + K_{support} = 2570 + 175 = 2745\ N/m \tag{3.1}$$

Where:

- $K_{equivalent}$: overall stiffness of the equivalent spring.
- K_{spring}: stiffness of SEA spring.
- $K_{support}$: stiffness of the support spring.

It is easy to see that the final stiffness is mostly due to the SEA spring. That's the reason why we can neglect the presence of the other one and why we can claim that the energy is injected int the system just through the SEA spring.

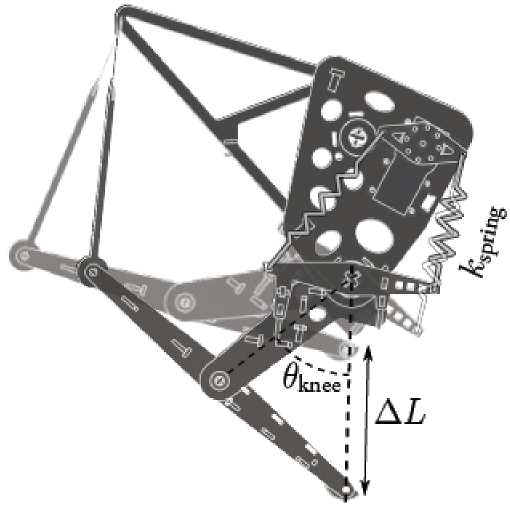

Figure 3.6: Cheetah leg. Starting from the Evans mechanism, the leg of the cheetah were designed. The tip of the foot can only move along a straight line (dashed line). The SEA spring is connected through suitable supports to the knee joint and motor. Figure reproduced from [7].

The two legs are connected though the so called "pelvis", anterior and posterior, to the "backbone", anterior and posterior respectively. The "backbones" are fixed to the "spine" joint as you can see in figure 3.7.

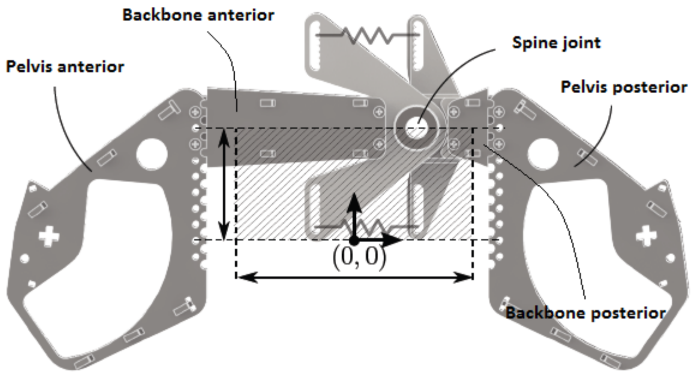

Figure 3.7: Cheetah main body. It comprehends "pelvises", "backbones" and "spine". Figure reproduced from [7].

The "spine" joint is simply a rotational joint, made by four bearings, with a certain degree of stiffness due to the presence of four springs with stiffness of 500 N/m each, connected to their supports. The whole mechanical structure is shown in figure 3.8.

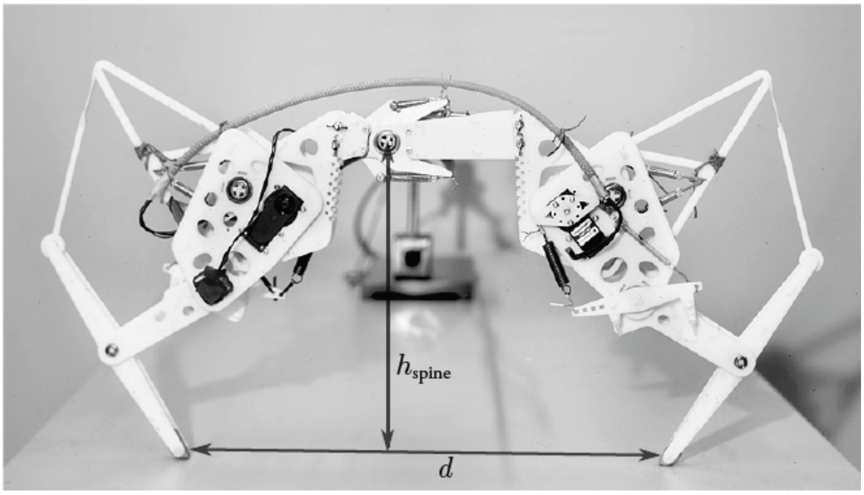

Figure 3.8: Cheetah prototype. Figure reproduced from [7].

The feet were redesigned in order to increment the stability of the robot by increasing the area of contact with the terrain, to increase the push-forward force trasmitted to the ground by changing their shape, to reduce the impact force by build them with an elastic material and to slightly improve the energy efficiency because the feet now can store and release a small amount of energy due to their elasticity. The new design is shown in figure 3.9.

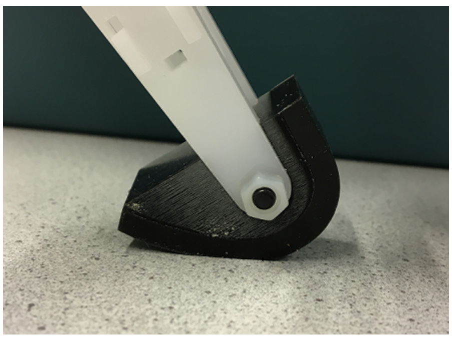

Figure 3.9: Foot design. The feet were 3D-printed with different materials. An elastic layer (black) and a rigid core (transparent).

The final setup used in the tests is shown in figure 3.10. It includes the robot, the electronic and electric parts (in appendix B you can find more details about), the boom, the carriage and the power supply.

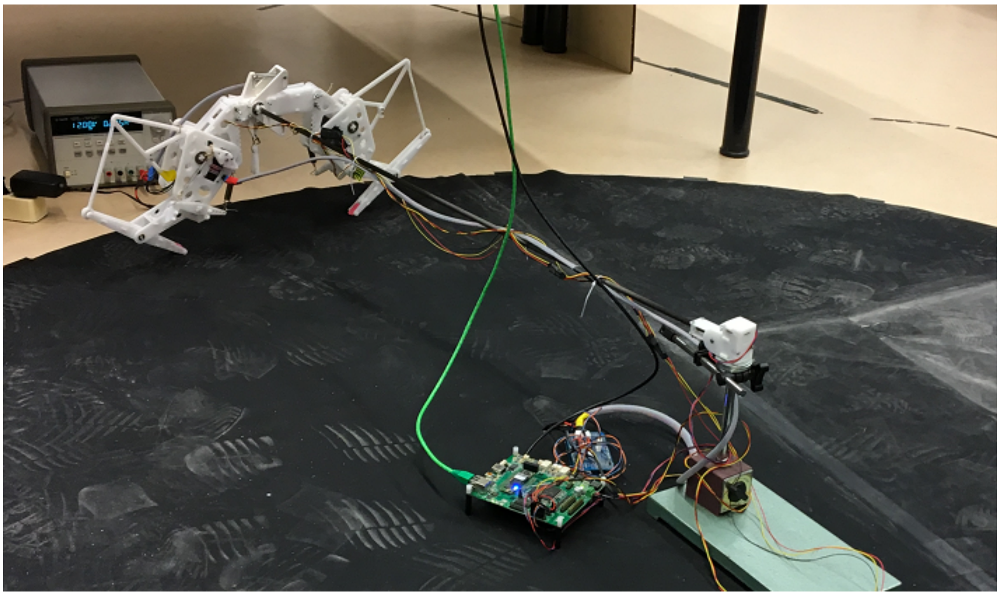

Figure 3.10: Cheetah setup. The cheetah runs on a black rubber carper to increase the friction between feet and ground in order to help the locomotion.

3.3 Simulation-model simplification

The initial model, developed on the software 20-sim, of the two-legged cheetah was terribly slow, around fifteen minutes of computations for fifteen seconds of simulation, because of the high number of bodies and joints implemented. Because the goal of the research is to design and control a four-legs cheetah and for efficiency reasons, a strong speed up was necessary. The model simplification lies in the fact that the cheetah can be modeled schematically as two bodies, anterior and posterior, asymmetrically connected by a rotational joint, the spine. Futhermore each of the body is connected through a spring to a foot, as shown in figure 3.11.

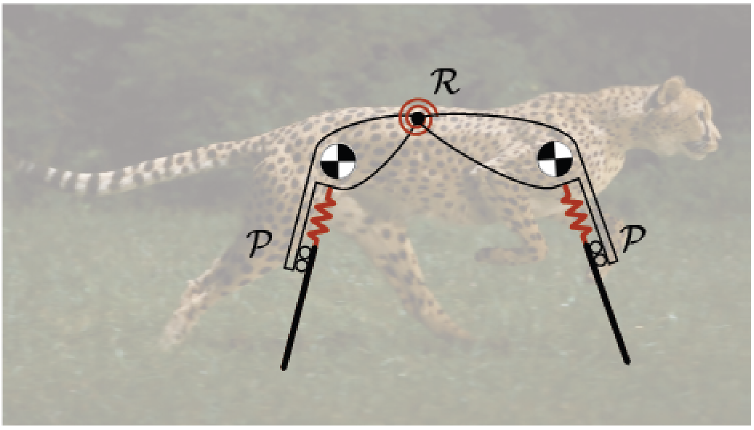

Figure 3.11: Cheetah physical model. The complexity of the spine of the animal can be represented well through an elastic rotational joint. The anterior and posterior halves are instead modeled as systems mass-translation spring-mass. Where the two masses are the one of the feet and the mass of the upper bodies. Figure reproduced from [7].

The focus was mainly on the "Hopper" block. The bodies called "femur" and "pelvis" are fixed, because the "hip" joint is fixed, so first of all, in order to reduce the complexity of the model, we could substitute these two bodies with the equivalent one, called simply "body", i.e. the body with mass equal to the sum of the masses and inertial properties defined thanks to SolidWorks. Then we replaced "foot", "tibia" and "fibula" with a new body, with very small mass and parallelepipedal shape, called simply "foot". In this configuration the "knee" joint, that connect "foot" and "body", is a linear joint with passive elements, rather than a rotational one. So how could we adapt the controller, without changing it, to inject the equivalent amount of energy as before? The "not changing the controller" part deserves few words: the goal now is to find an equivalent model of the previous one. To do that we kept the control parts as they were and we just changed the mechanical parts, then the "model validation" was done by comparing the behavior of the two models. If the final behaviors would be enough similar it means that the new model is good. In the original model, the motor injected energy in the knee joint through the SEA and because the knees were revolute joints we needed to impose an angular speed, $\dot{\theta}$. Now we need a transformation from the angular velocities to the equivalent linear one. That kind of mapping can be find using equation 3.2.

$$v = 2 * l_{leg} * \omega * \cos\theta \tag{3.2}$$

where:

- v is the linear velocity
- l_{leg} is the physical leg lenght
- θ is the angular position

In figure 3.12 is shown the equivalent model of the legs designed.

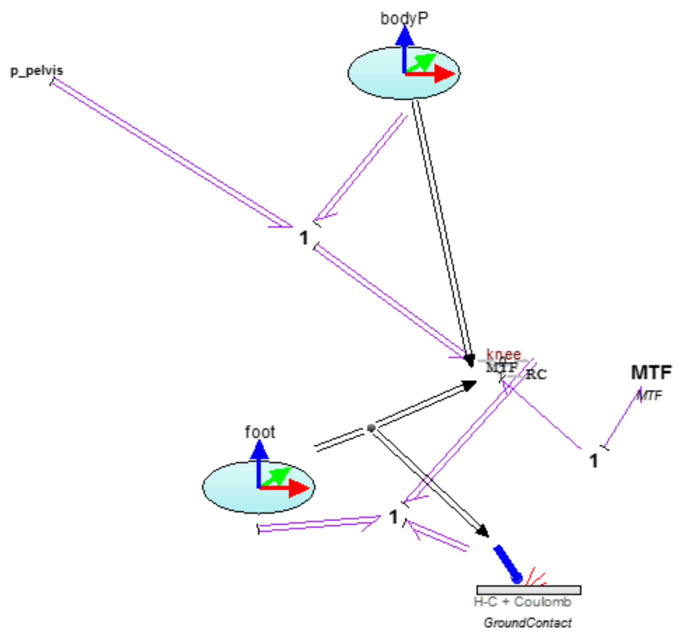

Figure 3.12: Equivalent leg model. The complexity and the number of bodies and joints is hightly decreased. Now we have just two bodies, representing upper-body and foot, connected by an elastic translational joint. The knee joint is still actuated and the amount of energy injected by the motor is mapped from rotation to translation by a suitable transformation.

3.4 Equivalent-model validation and final considerations

The validation of the new model were done by comparing the new behaviour with the older one, remembering that the controller was kept untouched. In particular in figure 3.13 and 3.14 the feet positions $(x(t), y(t), z(t))$ comparison is shown.

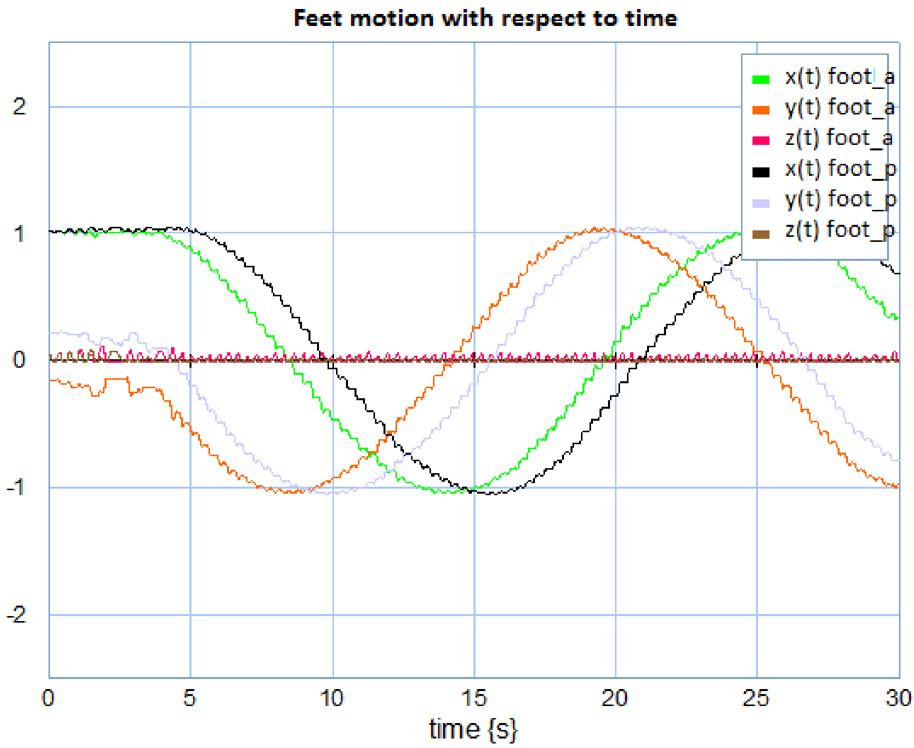

Figure 3.13: New model feet motion. You can see the position with respect to time of the anterior foot (respectively green, orange and red) and the position with respect to time of the posterior foot (respectively black, grey and brown).

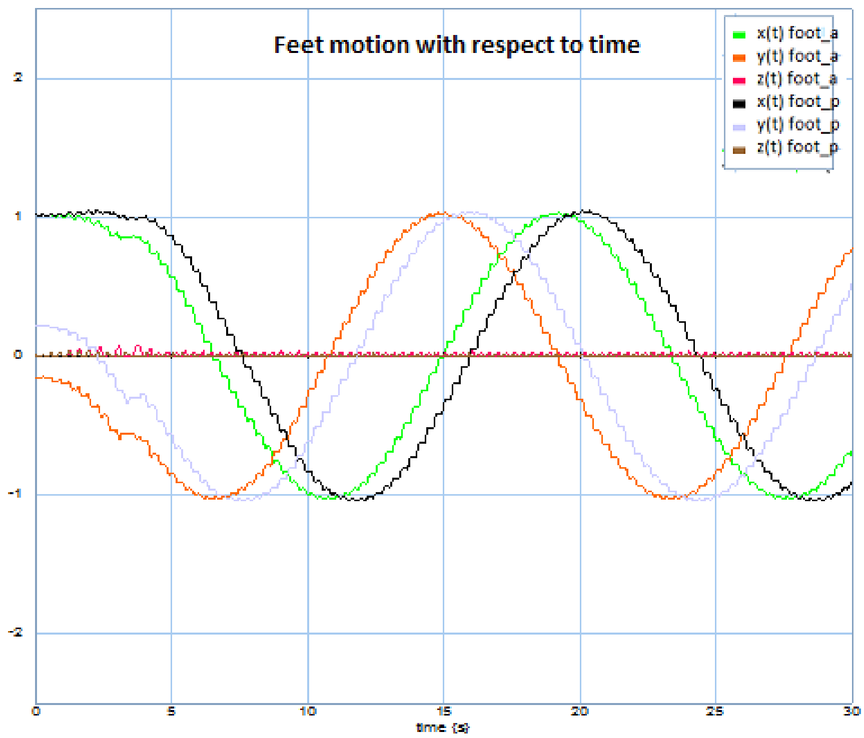

Figure 3.14: Old model feet motion. You can see the position with respect to time of the anterior foot (respectively green, orange and red) and the position with respect to time of the posterior foot (respectively black, grey and brown.)

The positions where extracted from the homogeneous matrices of each body and in particular considering the elements:

- $H[1, 4]$ correspondent to $x(t)$.
- $H[2, 4]$ correspondent to $y(t)$.
- $H[3, 4]$ correspondent to $z(t)$.

While in figure 3.15 and 3.16 the upper bodies positions $(x(t), y(t), z(t))$ comparison is shown.

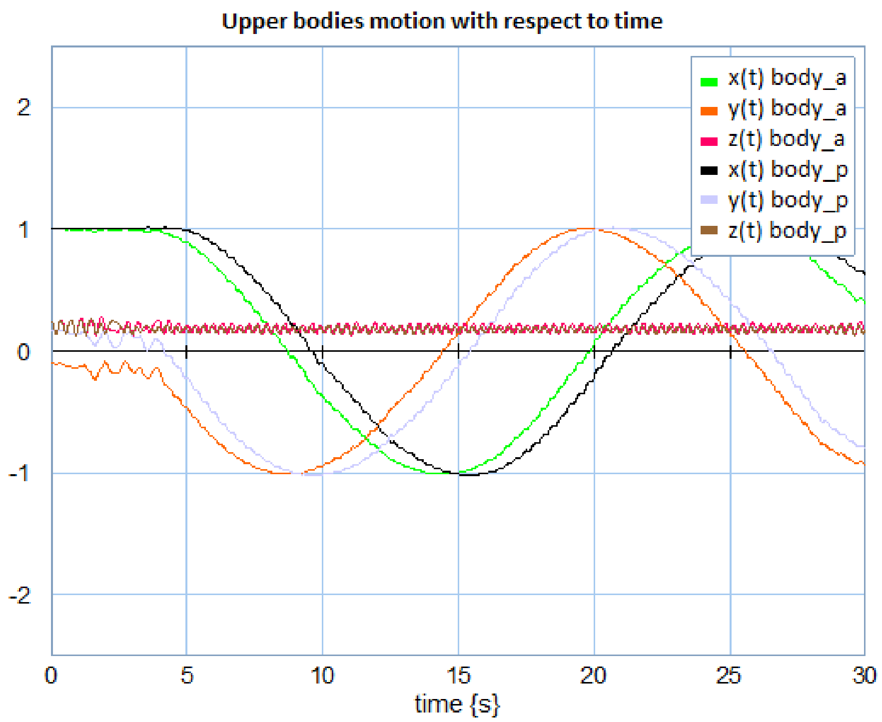

Figure 3.15: New model upper bodies motion. You can see the position with respect to time of the anterior upper-body (respectively green, orange and red) and the position with respect to time of the posterior upper-body (respectively black, grey and brown).

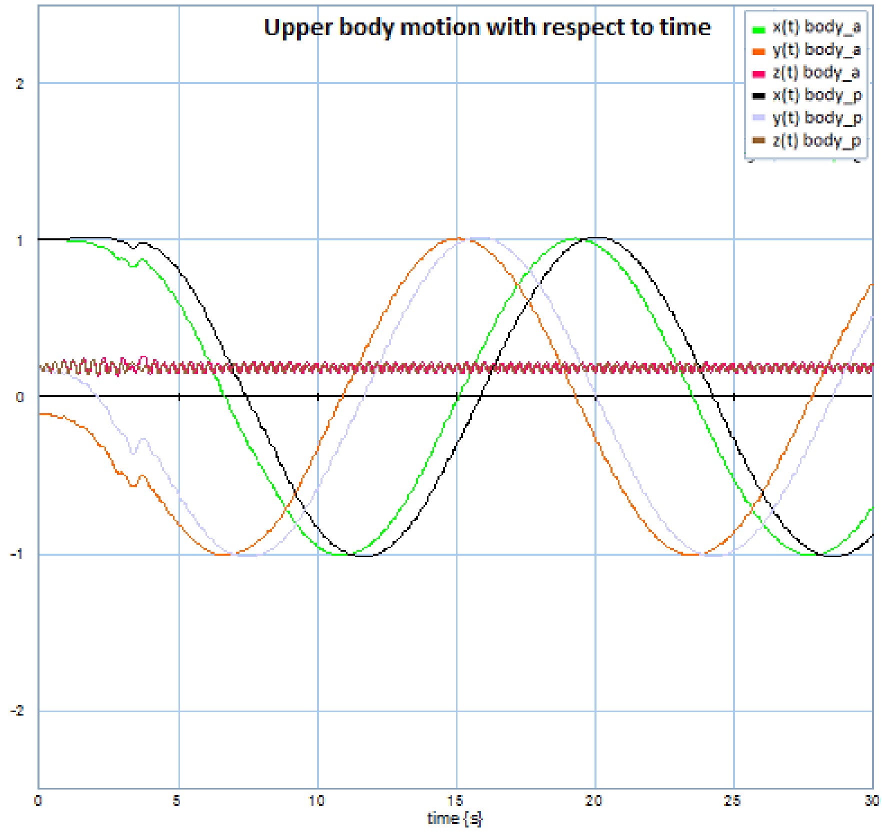

Figure 3.16: Old model upper body motion. You can see the position with respect to time of the anterior upper-body (respectively green, orange and red) and the position with respect to time of the posterior upper-body (respectively black, grey and brown).

It is easy to see that the behaviour is pretty much the same. There is only a light speed difference between them, the older model is a little big faster in terms of speed of the cheetah. However we can claim that the new model is a good equivalent one for the original model. Thanks to the new model we achieved a speed up in the simulation of a factor bigger than five and that was exactly what we wanted.

3.5 Old controller structure and introduced improvements

From previous sperimental trials it was discovered a possible running frequency i.e. a phase difference between feet touchdowns that allows locomotion. The period of the gait is equivalent to 0.36 s with an average estimated touchdown time of 0.16 s, a flight phase of 0.2 s and a delay between the touchdown time of the legs of 0.05 s, for an overall running frequency around 3 Hz. The idea of the developed controller, already implemented on the cheetah before the start of the project, was to keep the above-mentioned phase difference by injecting a controlled and suitable amount of energy into the system through an hysteresis controller. The first step was to measure the current phase difference between the feet through the estimation of the contact event between feet and ground. In order to do that we looked at the "knee" sensors: overcoming a certain threshold means that a compression of the spring occurred and that means that the foot touched the ground because there can't be compression in any other situations. After computing the touchdown time of each foot we computed the measured phase difference as:

$$\psi_m = \frac{TouchdownAnteriorFoot - TouchdownPosteriorFoot}{GaitPeriod} \tag{3.3}$$

It's a reactive controller because every time we detects the ground we reacts with a certain energy injection. Because we have servomotors we can change the position of the motors in order to compress more or less the springs with an hysteresis control. We simply check at every period if the measured phase difference is smaller or bigger that the ideal one, used as setpoint, then we apply two different control inputs if we are in a case or in the other. Although well designed, this controller shows its main drawback in the stability of the run. The run is pretty instable and the cheetah often stumbles in the ground. This is due to two big problems of the reactive controller:

- non-smoothness of the control action

- rough estimation of the ground contact events

In the first case it was implemented a PI controller to improve the control action, while in the second case it was used a low-pass filter to filter the sensors outputs and to be able to choose more precise thresholds that reduced the ground contact event estimation error. Even with the improvements introduced the "reactive" controller was not stable and reliable enough to be implemented on the four-legged robot, but it was really useful because it allowed to discover the running frequency of the robot and the correlation between the good running behaviour of the cheetah and the constancy of the peaks of the "knee" sensors outputs. These notions were used to implement the energy control.

3.6 Energy control

Working on the "reactive" controller, we figured out the running frequency or natural frequency of the robot and we noticed that if the compression of the SEA springs was held constant the cheetah could run in a very stable way without falling or stumbling. With these ideas in mind, we developed the "energy" controller which focus is to excite the system with prefixed frequency inputs, to control the energy injected and to keep the energy level of the system constant. The cheetah can move thanks its mechanical dynamics, but because of the energy losses it is necessary to injected energy through the SEA springs by the motors. So two main questions arise:

- exciting the system natural frequency is enough to provide the wanted phase difference?
- can we control the energy injected i.e. can we control the potential energy of the springs?

To excite the system, we generate clock-based setpoints for the servomotors i.e. square waves with period equal to the gait period, duty cycle equal to the ratio between touchdown time and period of the gait and delay between each other equal to the phase difference between the legs. Unfortunately only exciting the system with this fixed-frequency setpoints with constant amplitude is not enough for steady run, but if we combine it with the energy control, that regulates the amplitude of the square wave i.e. the energy injected, we can induce stable locomotion. The proposed controllers were tested through software simulations and on the "half-cheetah" prototype.

3.6.1 "Energy"

To introduce the energy control we need first to take into account the springs and their potential energy. By previous studies, it was proved that the energy stored in the SEA springs is proportional to the energy of the overall system [6]. If we consider a linear spring, its potential energy can be expressed in function of the dispacement.

$$E(x) = \frac{1}{2}kx^2 \tag{3.4}$$

For a non-linear spring, instead, the potential energy is expressed as the integral of its force.

$$E(x) = \int_0^x f(u)du \tag{3.5}$$

where:

- k: spring elastic constant.
- x: spring displacement.
- u: integration variable for the displacement.
- $f(u)$: non linear function in function of the displacement representing the force of the non-linear spring.

In the cheetah design the springs are hightly non-linear, the idea is to control an amount of energy that is proportional to the potential energy. The choosen term is pretty similar to the potential energy for linear spring equation, but without considering the spring stiffness k or if you prefer by keeping the stiffness of the spring equal to the constant 1.

$$E(x) = \frac{1}{2}x^2 \tag{3.6}$$

This reasoning holds because a strong proportionality exists between the spring, linear or non linear, compression and its potential energy, as shown in equation 3.4 and 3.5. To measure the displacement of the spring we use the sensors (see Appendix B for more details) placed on the knees that generate an output that is proportional to the spring compression.

3.6.2 Controller

The final goal is to design an energy-efficient running cheetah, so the controller job should be injecting, into the system through the springs, an amount of energy equal to the one lost because of the frictions, i.e. the mechanical dissiaptions that are present. Unfortunately measuring the losses in the prototype is not doable, but if the controller would keep the energy level or at least an estimated energy level of the system constant, the half-cheetah, thanks also to its own dynamics, would run. In order to generate the correct phase difference between the front and rear legs, two square waves generators generate fixed-frequency setpoints for the motors. The servomotors used have a position controller implemented on them, so we can control easily the position of the rotor. The choice was to let the motors work only to aid the compression of the springs, though a suitable variation of angular position. The posterior setpoint is delayed by 0.05 s with respect to the anterior one in order to generate "forward" locomotion. The period and the width of the pulses are fixed respectively to 0.36 s and to 0.16 s while the amplitude is variable and it is correct every period by the "energy" controller. In particular the controller generates a correction of a "compression" offset. In figure 3.17 and 3.18, it is possible to see the controller design.

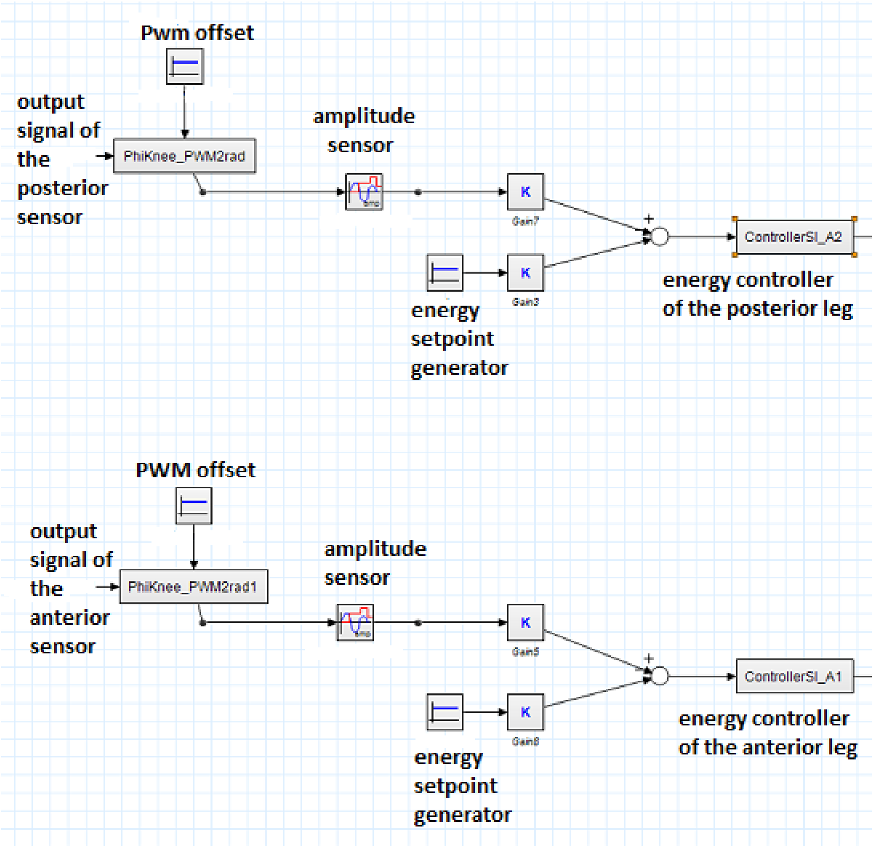

Figure 3.17: Control scheme (part 1). Starting from the reading of the "knee" sensor, we measure through the block "amplitude sensor" the maximum values of the signal coming from the encoders. Then we generate using a gain, suitably defined, the "measured energy" term. We compare this signal with a setpoint to generate and error and to feed the controller. It is worth to notice that the signal form the sensors is a PWM wave so we need to convert it in radians before elaborating it.

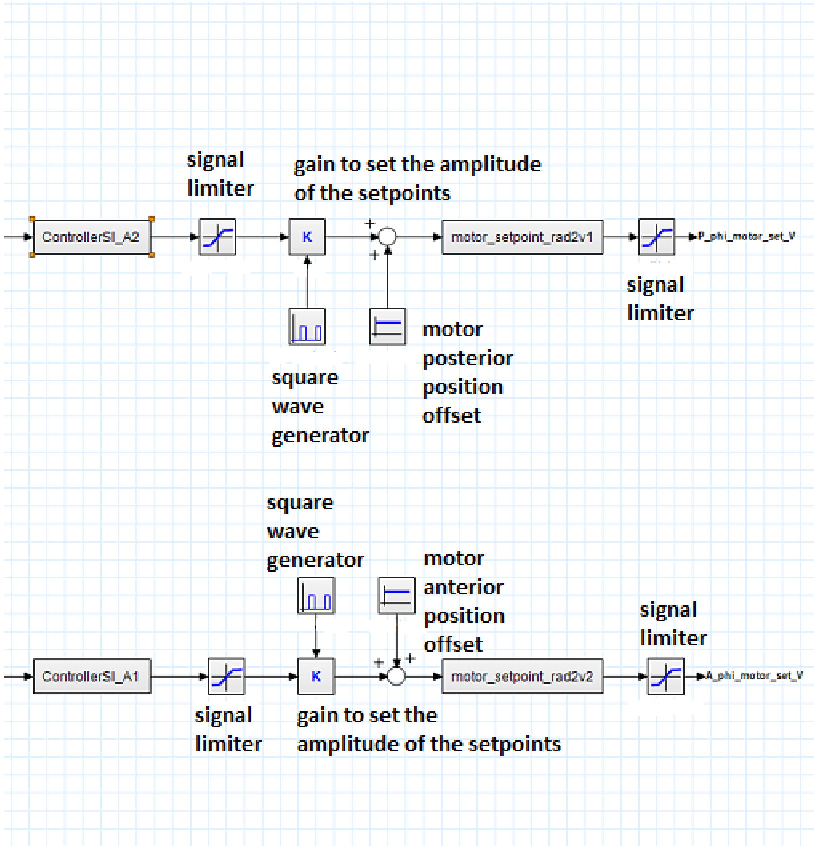

Figure 3.18: Control scheme (part 2). The control action pass through a signal limiter in order to avoid that the correction changes the working of the motor. We only want to compress the springs and not overextend them. Then the correction generated will change the amplitude of the square wave. The offset of the motors them is summed in order to fix the suitable initial conditions. Eventually we convert the signal to PWM in order to feed the motors.

The only real measures that has been used are the angular positions of two sensors, two encorders, placed on the "knees" of the cheetah. Then, throught the 20-sim blocks "amplitude sensor" it has been possible to figure out the maximum peak of the "knees" sensors signals and by just squared and divided, by the constant 2, those signals, we have been able to obtain the "measured-energy" term. These new signals are compared with a constant "energy setpoint" in order to generate errors that feed the controllers. Then, two conceptually different proportional controllers have been developed:

- A proportional controller with constant gain
- A proportional controller with variable gain

It is worth to notice that the initial condition was choosen with the cheetah that is still on the ground, of course, so a problem arises: how can we excite the cheetah to put it in motion? The proposed controllers solve in different ways the problem of the initial energy transient i.e. it is necessary to provide the cheetah a bigger amount of energy than in the "running" phase.

3.6.3 Proportional controller with constant gain

Because we can't choose a relatively big gain for the controller because it would drive the system to instability, we need to find a way to excite the system such that the springs potential energy is close to the setpoints. In particular in the simulations it happens that the cheetah overturns, event hardly found on the prototype. The idea is to get close to the energy setpoint by injecting for the initial four pulses a bigger amount of energy through four fixed-amplitude pulses, then the controllers, even with a relatively small gain can track well the setpoints. The generated corrections directly act and adjust the amplitude of the pulses.

3.6.4 Proportional controller with variable gain

To improve the previous controller, in particular to remove the initial fixed amplitude pulses and to have more flexibility on the initial conditions, we choose to use a variable gain, proportional to the error, i.e. for big errors we will have big gains and viceversa.

$$K = k(error) \tag{3.7}$$

where K is the variable gain and k is a suitable defined constant.
The advantages of these controller with respect to the previous one are mainly two:

1. The control of the very initial energy injected i.e. the removal of the initial open-loop big injection of energy.

2. The controller now can adjust its own energy so the starting is automatic and totally due to the energy controller work. The initial position does not matter anymore and it is also possible to stop it during its run then freed it and it can keep on running.

The only disadvantage is that the run is slightly more instable.

3.6.5 Controllers tuning

It deserves attention the tuning of the controllers. The system is hightly chaotic due to the presence of many passive non-linear elements and because of its interaction with the environment through the ground. Because of that, the errors at steady state will not be null, but if they are small enough, so if the errors lie in a zone enough close to zero without big variations i.e. the energy of the system is held almost constant, the cheetah can move forward pretty well. The tuning of the controllers was done mainly by heuristical methods and paramenters sweep methods. Surprisingly the tuning of the prototype controllers were much easy than the simulation controllers setup. The hypothesis may be many, but the most likely ones are:

- Energy dissipation differences: the mechanic dissipation of the prototype are larger than the ones of the model. This leads to a small tuning-margin, in the simulations, because even a small surplus of energy can let the cheetah fall down, while in the prototype this was not noticed because, for example, an higher jump generated by an higher amount of energy injected will be compensated with more losses.

- Integration errors of the solver algorithm due to the ground contact model: every time the feet touch or better penetrate the ground, they feel a force due to the reaction of the ground itself. Because the position of the feet change every step it means that the forces changes every step and in particular the initial condition for the integration algorithm changes every step. Different initial conditions and different "background" roundings in the simulations can lead to big variation of the behaviour and this make the simulations more unpredictable than the prototype. You can see more details in Appendix C.

3.7 Simulations results for the energy control

In this section, the simulation and the prototype results will be presented, taking in mind that the controllers tuning in the simulation, oddly, required much more time that the controller tuning in the prototype for the reason discussed above. In particular in this section, the results of the previous controllers will be shown, considering that the tuning in the simulations is different, of course, from the tuning on the prototype. In order to understand the results of the simulations and, in particular, the charts of the position of the feet and the bodies we need to remember that the "half-cheetah" is constrained to run on a circumference by the boom, attached to it, for evident standing problems, so the coordinates on the 2D-motion plane, $x(t)$ and $y(t)$, will vary almost as sinusoids, if the cheetah runs properly.

3.7.1 Constant gain controller

In figures 3.19, 3.20, 3.21 and 3.22, you can see the results of the simulations using the energy controller with the constant gain.

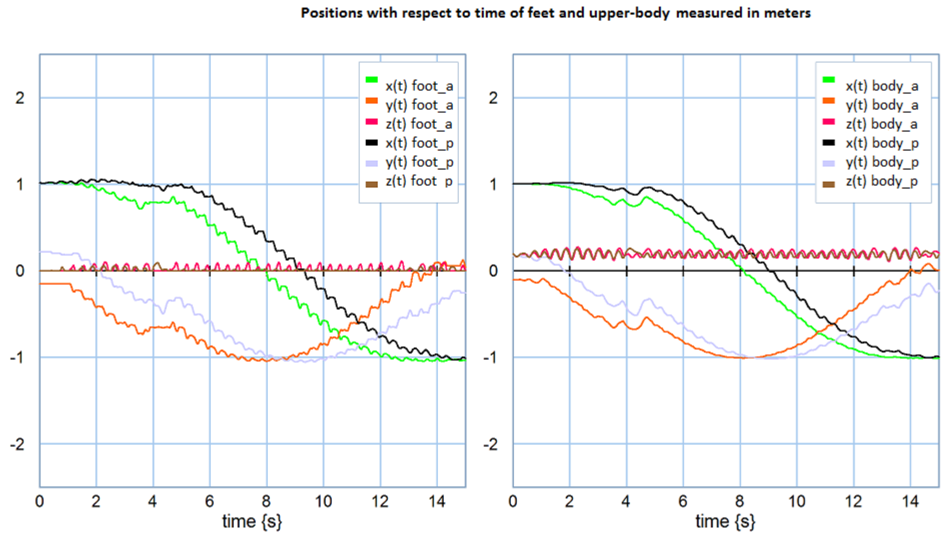

Figure 3.19: Simulation results. Positions with respect to time of the feet and the upper bodies. You can see $x(t)$ of the anterior foot and upper-body in green (left chart and right chart respectively), $y(t)$ of the anterior foot and upper-body in orange (left chart and right chart respectively), $z(t)$ of the anterior foot and upper-body in red (left chart and right chart respectively)$x(t)$ of the posterior foot and upper-body in black (left chart and right chart respectively), $y(t)$ of the posterior foot and upper-body in grey (left chart and right chart respectively), $z(t)$ of the posterior foot and upper-body in brown (left chart and right chart respectively)

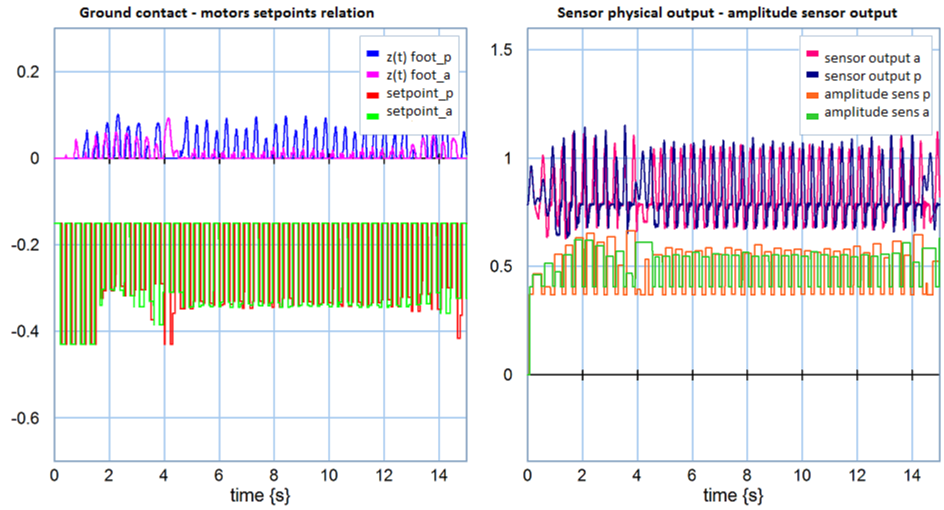

Figure 3.20: Simulation results. In the left chart, you can see the $z(t)$ positions of the anterior and posterior feet (pink and blue respectively) and the anterior and posterior motors setpoints (green and red). In the right chart you can see instead the anterior and posterir "knee" sensors measuraments (pink and blue) and the output elaborated by the amplitude sensors (green and orange).

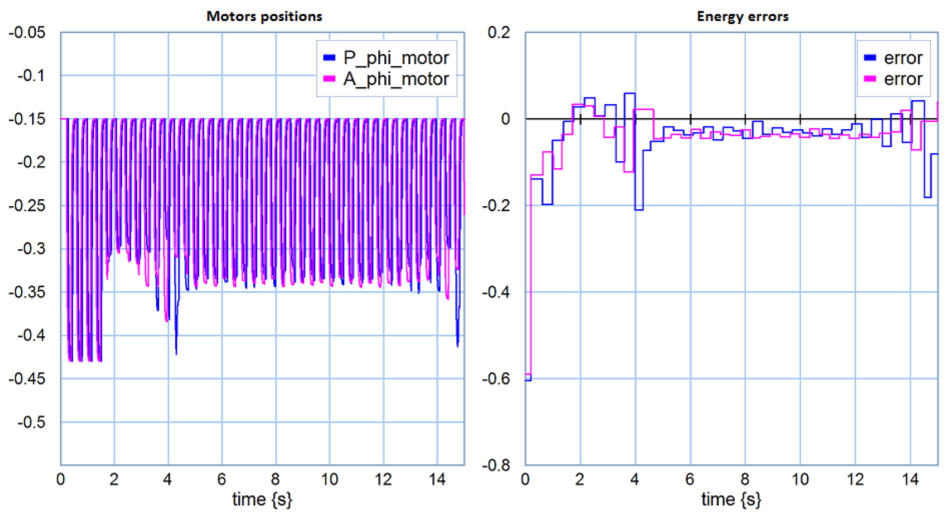

Figure 3.21: Simulation results. In the left chart you can see the anterior motor position (pink) and the posterior one (blue), while on the right you can see the energy error of the posterior leg (blue) and of the anterior one (pink).

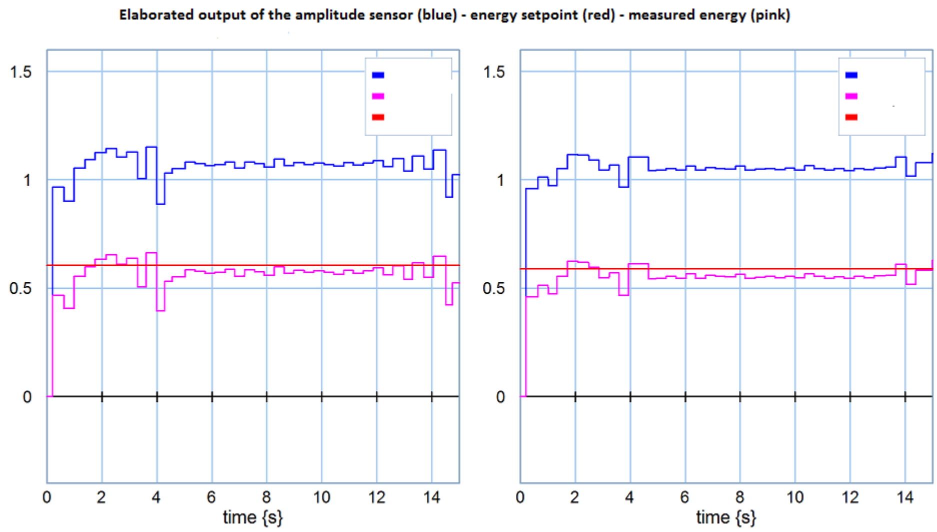

Figure 3.22: Simulation results. In the left chart you can see the elaborated signal coming from the amplitude sensors of posterior half (blue), the energy setpoint (red) and the measured energy (pink), while in the right chart you can see the elaborated signal coming from the amplitude sensors of anterior half (blue), the energy setpoint (red) and the measured energy (pink).

Because of the high complexity of the tuning, we were able to achieve a "good-run", i.e. we were able to keep the energy level on the springs quite constant, for just 15 s for a global distance of half a round, but it was enough to be confident on the potentialities of the controller.

3.7.2 Variable gain controller

In figures 3.23, 3.24, 3.25 and 3.26 we show the results with the variable gain controller.

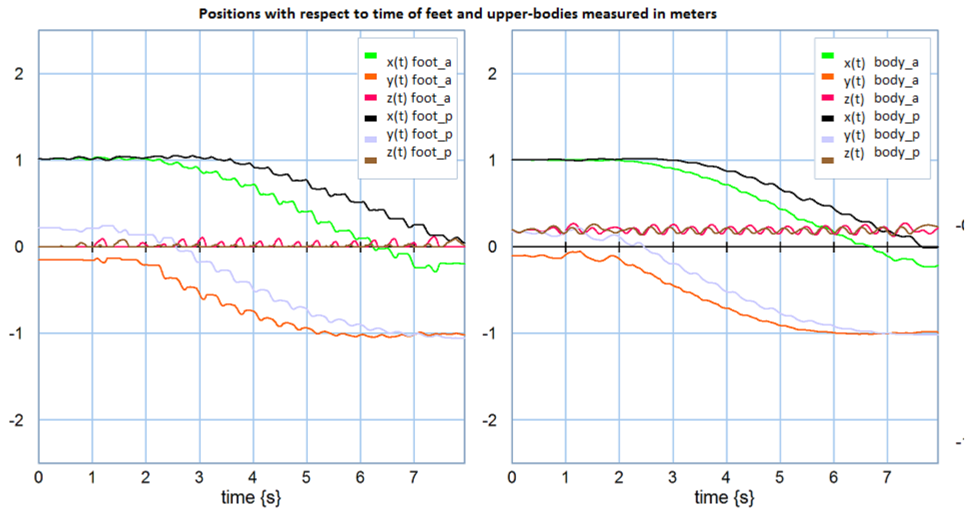

Figure 3.23: Simulation results. Positions with respect to time of the feet and the upper bodies. You can see $x(t)$ of the anterior foot and upper-body in green (left chart and right chart respectively), $y(t)$ of the anterior foot and upper-body in orange (left chart and right chart respectively), $z(t)$ of the anterior foot and upper-body in red (left chart and right chart respectively)$x(t)$ of the posterior foot and upper-body in black (left chart and right chart respectively), $y(t)$ of the posterior foot and upper-body in grey (left chart and right chart respectively), $z(t)$ of the posterior foot and upper-body in brown (left chart and right chart respectively)

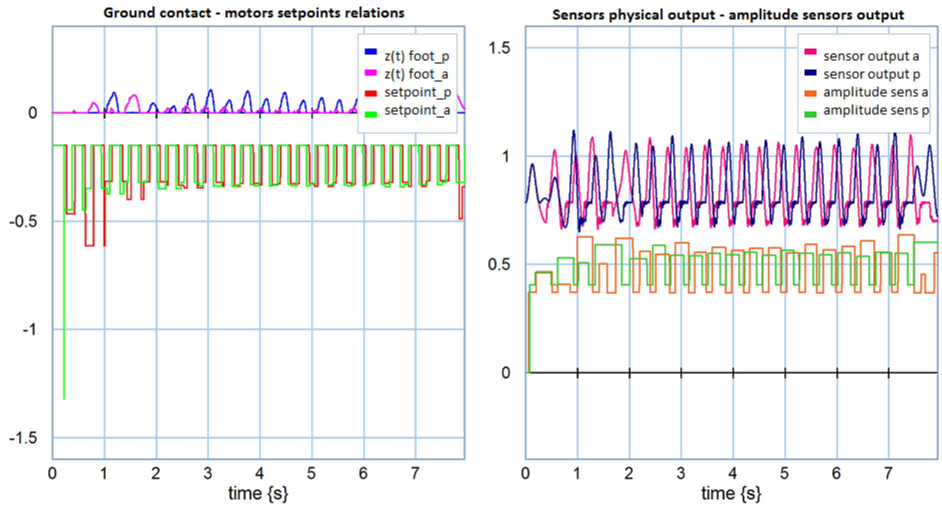

Figure 3.24: Simulation results. In the left chart, you can see the $z(t)$ positions of the anterior and posterior feet (pink and blue respectively) and the anterior and posterior motors setpoints (green and red). In the right chart you can see instead the anterior and posterir "knee" sensors measuraments (pink and blue) and the output elaborated by the amplitude sensors (green and orange).

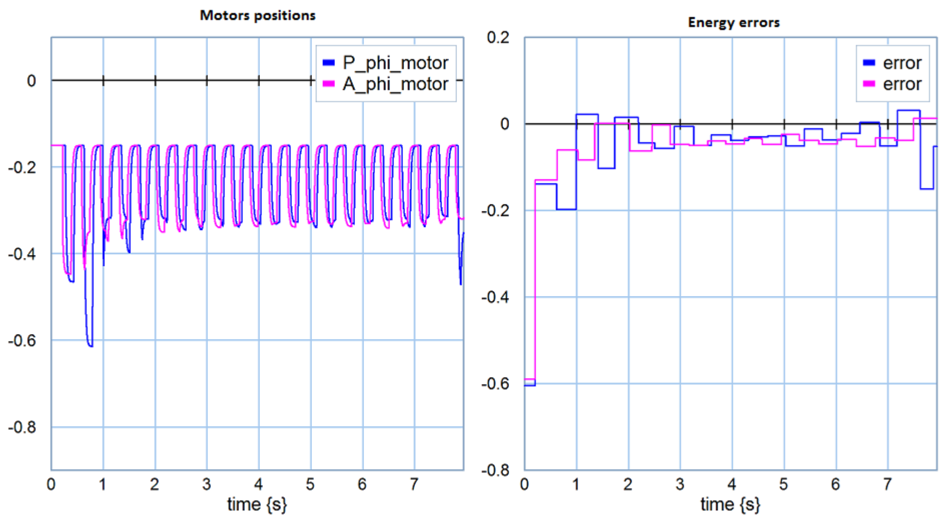

Figure 3.25: Simulation results. In the left chart you can see the anterior motor position (pink) and the posterior one (blue), while on the right you can see the energy error of the posterior leg (blue) and of the anterior one (pink).

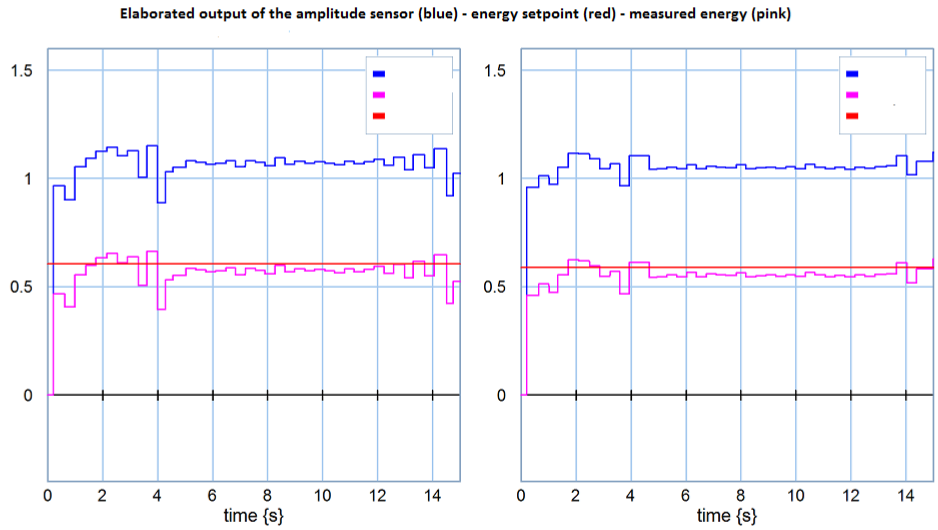

Figure 3.26: Simulation results. In the left chart you can see the elaborated signal coming from the amplitude sensors of posterior half (blue), the energy setpoint (red) and the measured energy (pink), while in the right chart you can see the elaborated signal coming from the amplitude sensors of anterior half (blue), the energy setpoint (red) and the measured energy (pink).

In this case we were able to provide just 7 s of "good-run" for a global distance of a quarter of a round, but it deserved to be prensented due to the "controlled-start" and for flexibility reasons that were already introduced above.

3.8 Prototype results for the energy control

With the prototype we were able to achieve much better results, indeed, for both of the controllers proposed, the half-cheetah can run for a full round and we are confident even more, but for wires lenght problems it was not possible to test it. In figure 3.27 is shown the position with respect to time measured by the encoder on the pole (in blue) and linearly mapped on the circumference in order to assess the real runned distance in meters (in green), while in figure 3.28 it is shown the average speed in m/s.

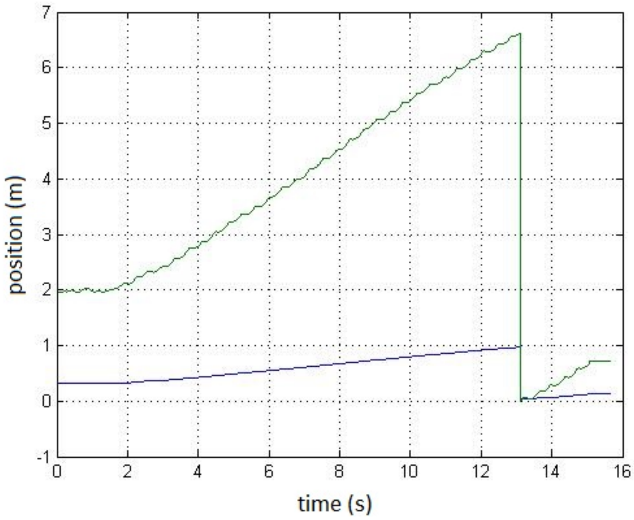

Figure 3.27: Prototype position with respect to time. In blue you can see the output of the encoder placed on the pole around which the cheetah runs, while in green the actual position measured in meters.

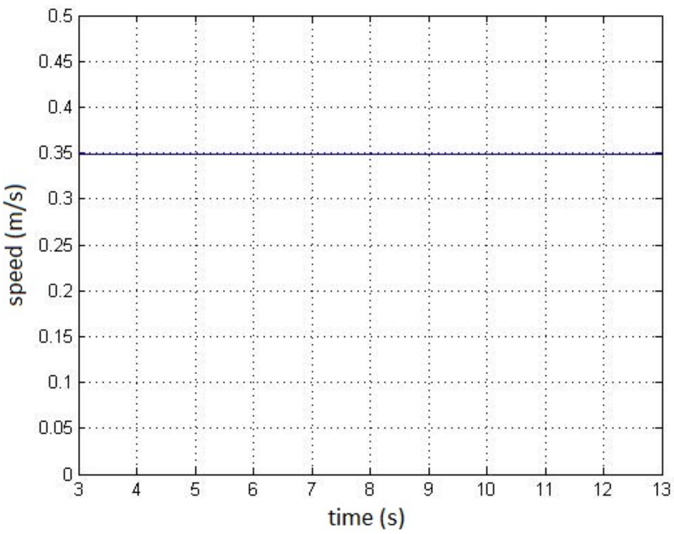

Figure 3.28: Prototype average speed with respect to time in m/s. We used the mean value to make the speed more readable.

Furthermore in figure 3.29 is sequentially shown the lap running.

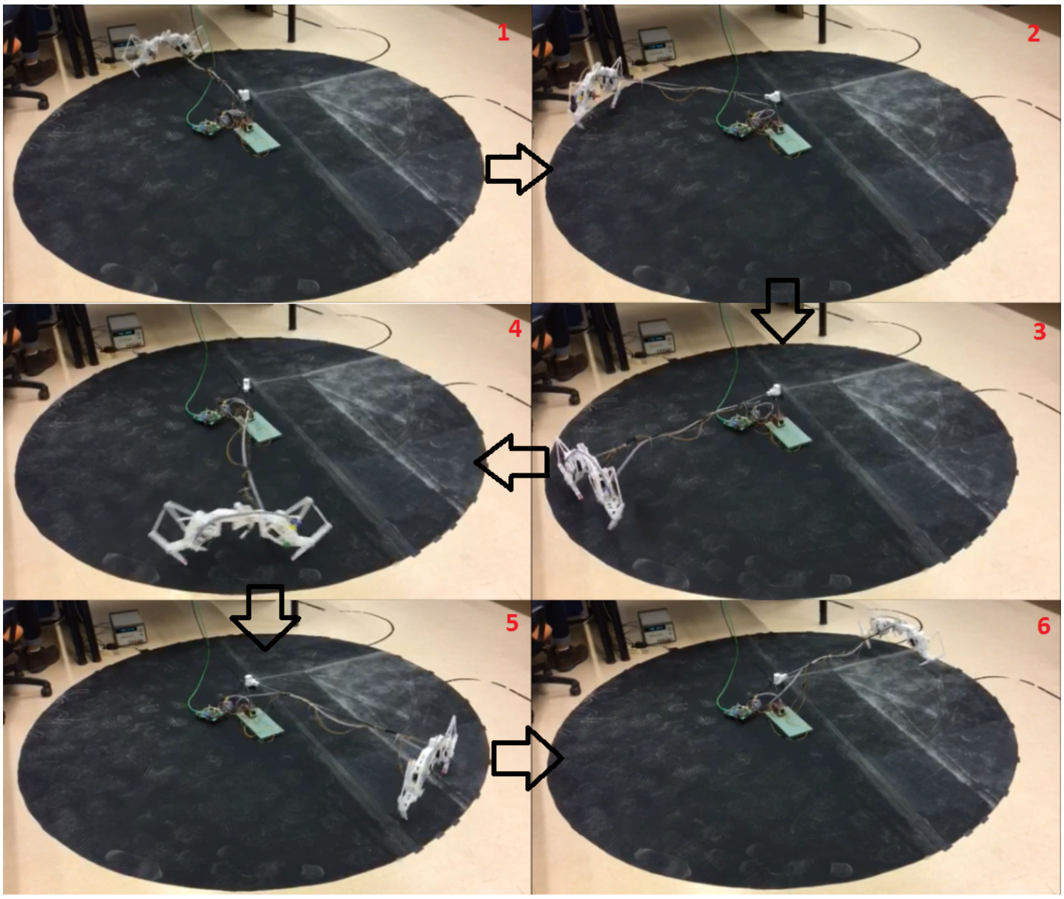

Figure 3.29: Prototype results. It's a 6-steps sequence that shown the setup and the running phase.

3.8.1 Constant gain controller

The run with, this kind of controller, is very smooth and stable so much that we like to think that it can keep running indefinitely. Because of its goodness it is the controller implemented on the four-legged cheetah.

3.8.2 Variable gain controller

With this controller too, the cheetah can run, but slightly roughter and in a more instable way than with the other controller. However it can still complete the full round and it could even run more. This controller offers the possibility of stopping the run of the cheetah, for example by lifting it and when freed it can keep on running. This thing was not always possibile with the previous controller.

3.9 Cost of transport for the half cheetah

Eventually we computed the cost of transport [8], or briefly CoT, of our robot. The CoT is a dimensionless quantity that measures and quantifies the energy efficiency of transporting an animal, a vehicle or a robot from a position to another. It can be computed as:

$$CoT = \frac{E}{mgd} = \frac{P}{mgv} \tag{3.8}$$

where:

- E is the input energy
- m is the mass
- g is the gravity constant
- d is the traveled distance
- P is the input power
- v is the speed

It is easy to foresee that we would like a value as lower as possible because that means that the power or the energy injected are relatively small with respect to the velocity or the distance travelled. With our setup we are able to achieve an high energy efficiency and an average CoT equal to 0.36, measured in a time interval of 10 s, because, although we are not able to achieve high speed, we inject a very low amount of energy inside our system thanks to its mechanics and in particular to the numerous passive elements present, figure 3.30.

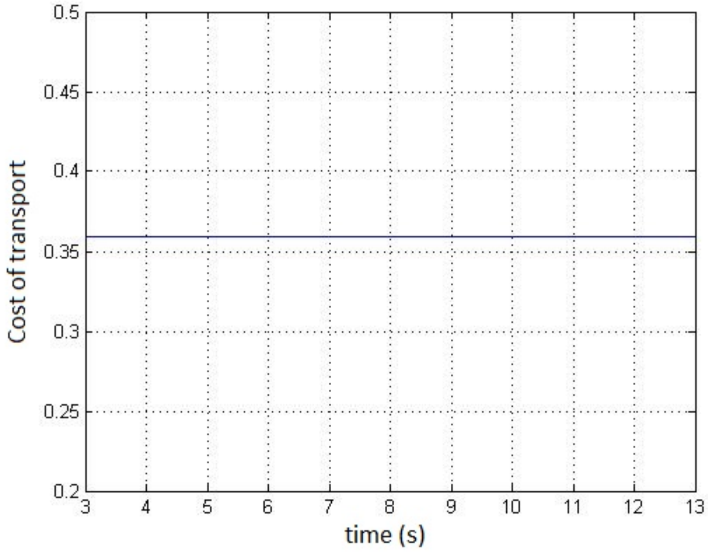

Figure 3.30: Cost of transport of the two-legged robot measured as $\frac{P}{mgv}$

In order to compute the cost of transport, using in particular the second formulation, we need to figure out what is the actual electrical power input, i.e. the power that the cheetah absorbes. The electrical power is the product of the voltage and the corrent and its equation can be written as:

$$P = V * I \tag{3.9}$$

To measures these two values we used an oscilloscope, tool to measure voltages and a very small resistor, typically called shunt. The input voltage, V, will be equal to the voltage measured at the beginning of the voltage wire in a position close to the power supply, while for the current absorbed, I, although we cannot measure it directly, we can measure the potenzial drop due to the resistor and then compute the current by dividing this value for the resistance itself. We chose to record data in an interval of 10 s with a resolution of $400\mu s$, so in the end we collected 25000 samples of our physical quantities, enough for a good accuracy. In figure 3.31 is shown the average input power in function of the samples collected, while in figure 3.32 and 3.33 the input voltage and the average current in function of the samples.

To conclude we scaled the robot to measure its mass, equal to 1240.5 g.

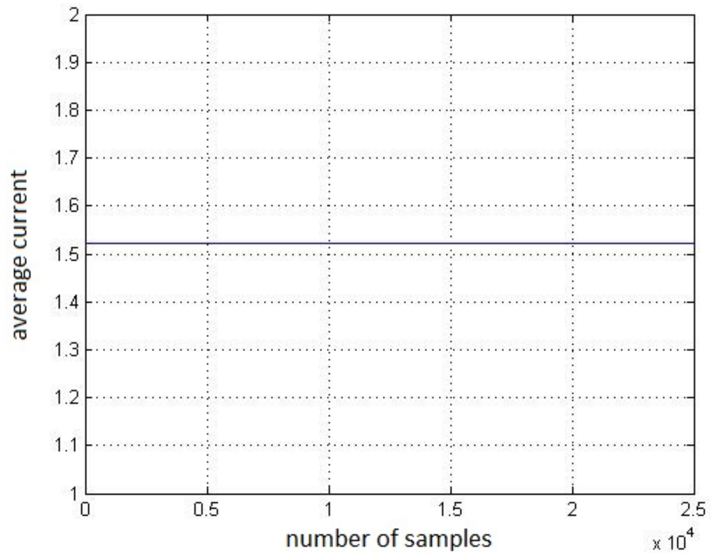

Figure 3.31: Average input power measured in Watt. We computed and show it mean value to increase the readability of the chart.

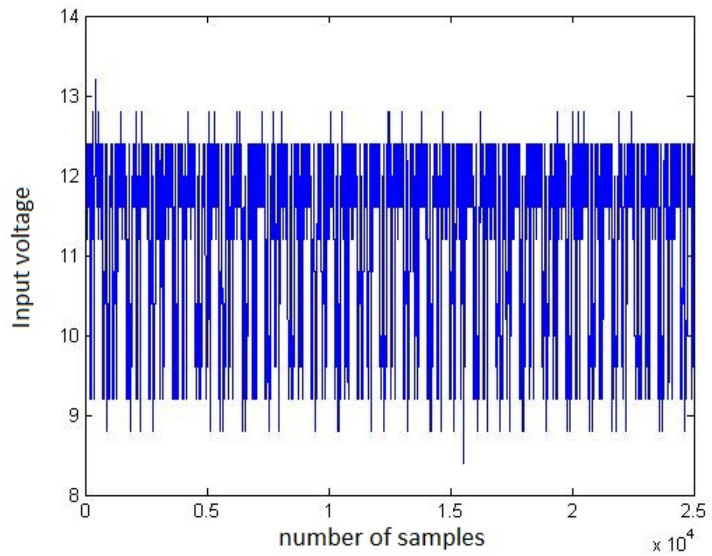

Figure 3.32: Input voltage measured in Volt. The motor are powered with 12 V DC (see Appendix B for more details).

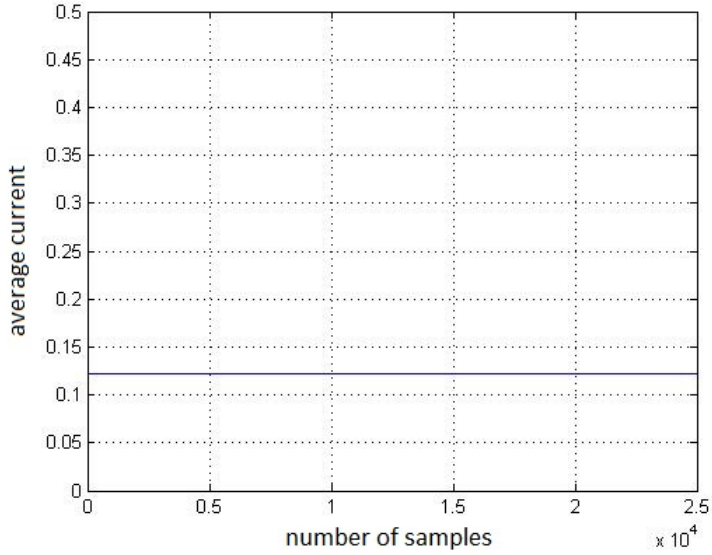

Figure 3.33: Average input current measured in Amphere. We computed and show it mean value to increase the readability of the chart.

Chapter 4

Four-legged cheetah robot

The natural further step of an "half-cheetah" robot is to make a four-legged one. This chapter will be divided in two principal sections:

1. Work on the simulations

2. Work on the prototype

4.1 Work on the simulations

In this section we will present the work on the simulations and in particular the new model structure, the controllers implemented and the results achieved.

4.1.1 Simulation model

The most straightforward way to extend the model of the two-legged robot to the four-legged one was to add another half, perfectly equal in terms of mass and dynamic properties to the first one, as shown in figure 4.1.

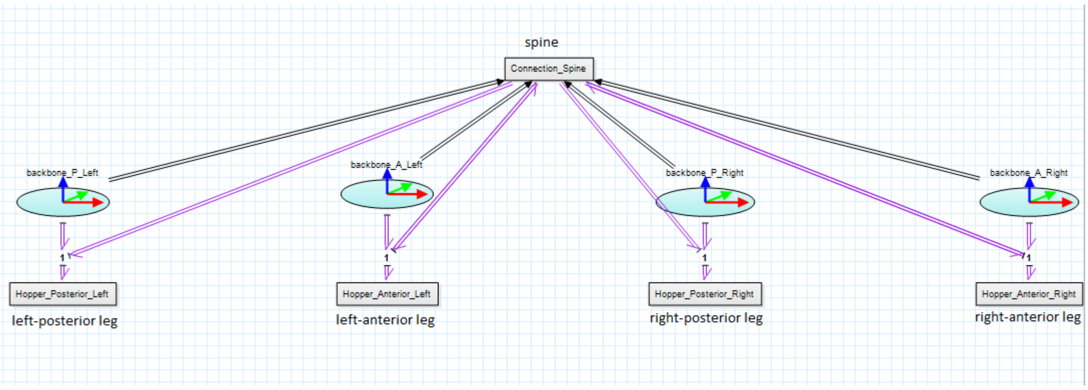

Figure 4.1: Four-legged cheetah model. We can highlight the presence of the four legs, for "backbones" and a single spine to connect the two halves together.

The model was structured in a similar way of the previous one, we can highlight nine principal blocks:

- Connection Spine: it includes all the parts related to the connection of the halves.
- Backbone Posterior Left: rigid body that connects left posterior leg to the spine.
- Backbone Anterior Left: rigid body that connects left anterior leg to the spine.
- Backbone Posterior Right: rigid body that connects right posterior leg to the spine.
- Backbone Anterior Right: rigid body that connects right anterior leg to the spine.
- Hopper Posterior Left: left posterior leg.
- Hopper Anterior Left: left anterior leg.
- Hopper Posterior Right: right posterior leg.
- Hopper Anterior Right: right anterior leg.

While most of the blocks were kept the same, in order to connect the two halves, the new block called "Connection spine" was designed. In particular, using the Bond-Graph notation, we improved the connection between front and rear halves throught the "spine" joint. Then we connect the boom to the "spine" joints in order to keep the halves as free as possible to have relative motion between each other. It is worth to highlight what the "spine" joint is and how it was modelled. Starting from the video analysis of a running cheetah and studying its physical model, it is straightforward to model the spine as a rotational joint with passive elements, springs in particular, to mimic the compression and extension phases. In figure 4.2 is shown its model on 20-sim in which you can recognize the power flow from the anterior to the posterior bodies, purple arrows and the addition of a compliant element, C-type and a dissipative element, R-type, intrinsic in every mechanical device.

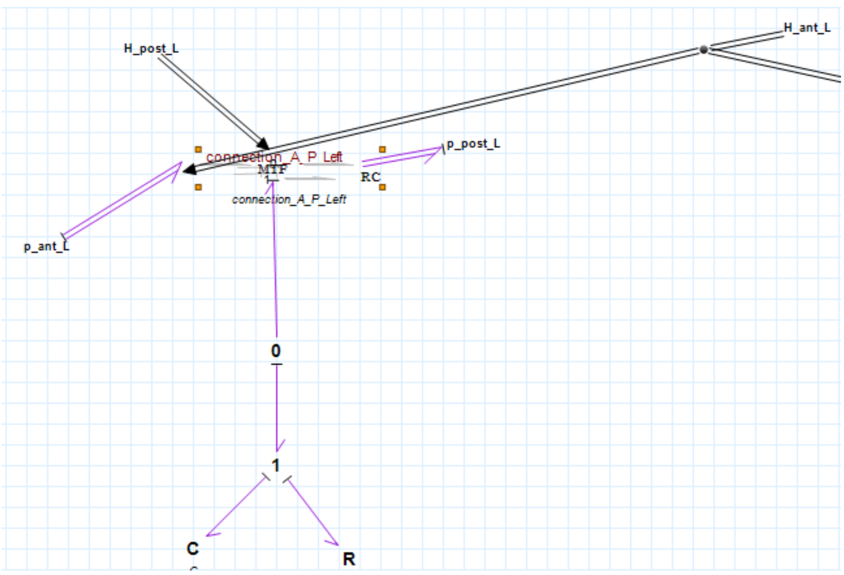

Figure 4.2: "Spine" joint model. The joint is a rotational joint, modeled as the one showed in Chapter 3. It is connected to a C-type and to a R-type element in order to apply the desired compliance. This joint connects anterior and posterior bodies of the robot.

Futhermore, the starting boom lenght was choosen equal to 0.5 m because in this configuration we obtained the maximum stability of the cheetah. This lenght will be reduced when the running sessions would be sufficiently good. In figure 4.3 you can see the whole model of the spine for the four-legged cheetah, in which it is possible to add extra degrees of freedom in the boom-"spine" joint connection. Several trials were done by changing the degrees of freedom of the spine.

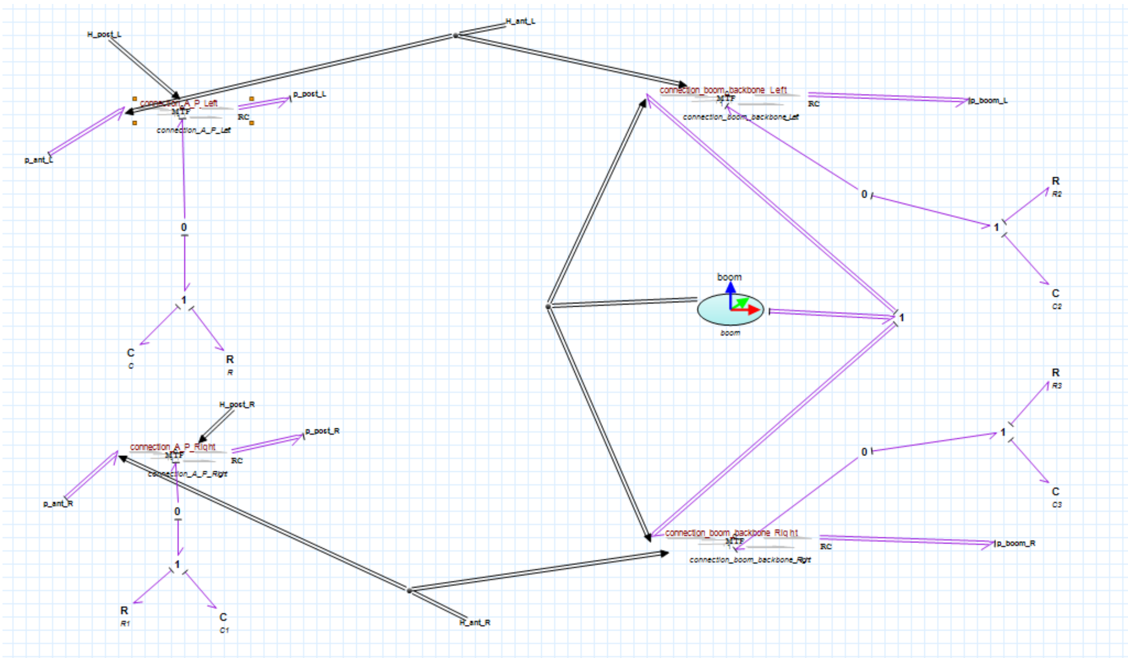

Figure 4.3: Overall spine model. On the left you can see the two "spine" joints that connect front and rear of each half, while on the left the connection of the boom with the two halves. It is still an elastic-dissippative connection, but with very high stiffness. The boom is modelled as a rigid body. The "spine" joints are the same as the ones explained in Chapter 3

We made a simple 3D visualization of the robot to improve the readability of the results, but also it helps to understand where the model simplification lies on. In figure 4.4 you can see easily the bodies that compose the cheetah, from down to top:

- Feet
- Main bodies

- Backbones
- Boom

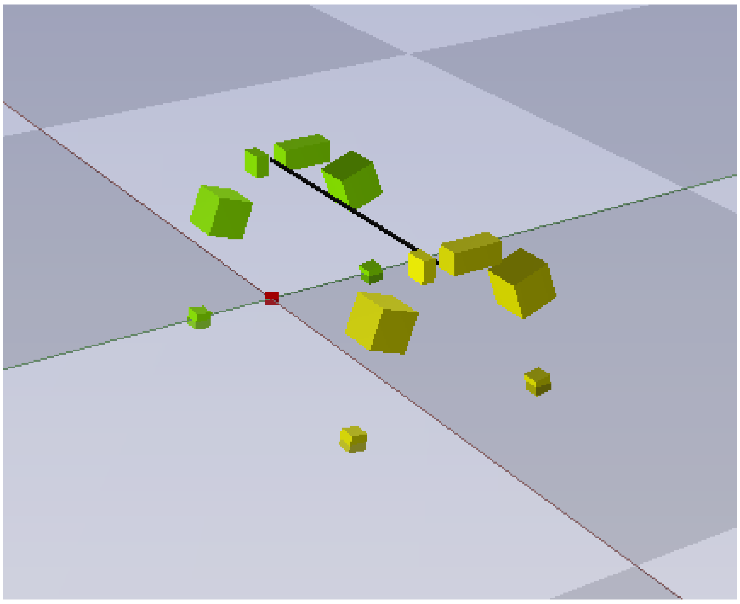

Figure 4.4: 3D visualization of the cheetah. On the left side (yellow) and on the right side (green) you can see from bottom to top feet, the upper-bodies, the backbones and the boom (black). The front is distinguishable from the rear because the "backbones" are significantly smaller.

4.1.2 Work on the controller

Now we are considering a four-legged robot so it is worth to highlight an interesting topic: the type of gait [16]. The gait is basically the syncronization of the legs necessary in order to provide locomotion, in Appendix D you will find more details about. There exists a lot of different gaits in nature, but in our case we will focus our work on the bound gait. The bound gait is one of the easiest gait to implement because the two halves are mirrowed in terms of movements, the two front legs have the same touchdown, liftoff and flight times as well as the rear ones, figure 4.5.

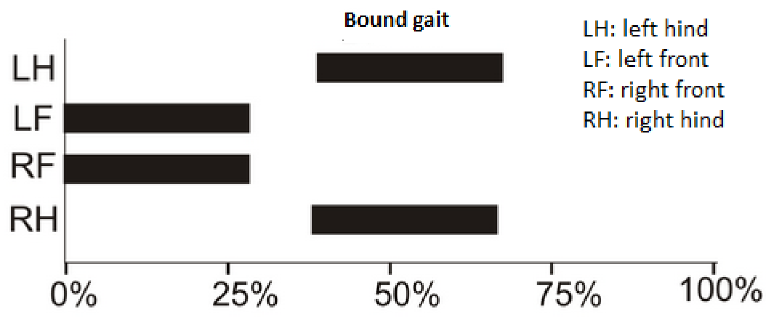

Figure 4.5: Bound gait footsteps. The black strips represent the time in which the feet are in contact with the ground.

If you would look carefully at figure 4.5, actually, in nature, the lift-off and consequently the flight times of the hind legs are not perfectly the same, but in our case we neglected this difference because it did not introduce limitations of the motion. Starting from the energy control with the constant gain of the half-cheetah, we implemented three different controller configurations:

1. Single controller for both halves.
2. Independent controllers for every half.
3. Independent controllers for every half with an extra supervisor to reduce the energy levels errors of the two sides.

Single controller for both halves

Because the two halves are perfectly equal and because the bound gait is a symmetrical gait, the idea is to copy the control action signals from the controller of one side directly as motor setpoints of the other. So we have a single controller, with feedback just from the sensors of one half, that controls the four legs. In figure 4.6, 4.7 and 4.8 you can see some charts that shows the behaviour of the cheetah with this kind of controller.

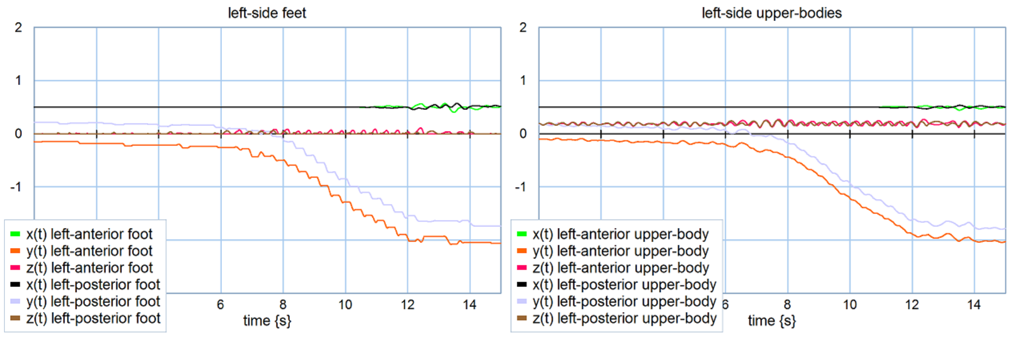

Figure 4.6: Single controller results. Positions with respect to time of the left feet and left upper bodies. You can see $x(t)$ of the anterior foot and upper-body in green (left chart and right chart respectively), $y(t)$ of the anterior foot and upper-body in orange (left chart and right chart respectively), $z(t)$ of the anterior foot and upper-body in red (left chart and right chart respectively), $x(t)$ of the posterior foot and upper-body in black (left chart and right chart respectively), $y(t)$ of the posterior foot and upper-body in grey (left chart and right chart respectively), $z(t)$ of the posterior foot and upper-body in brown (left chart and right chart respectively)

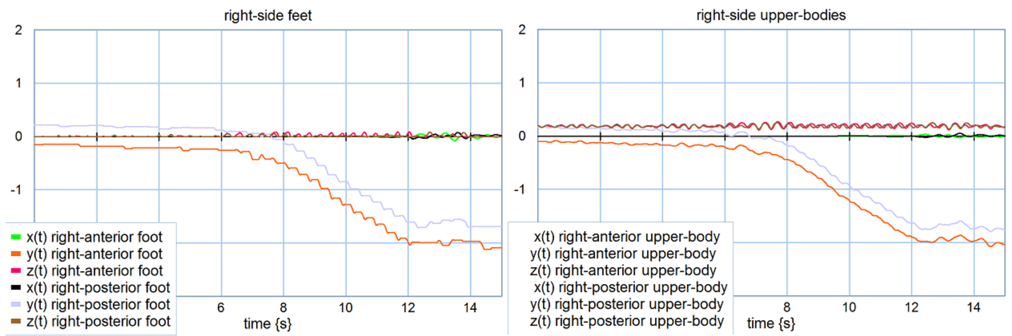

Figure 4.7: Single controller results. Positions with respect to time of the right feet and right upper bodies. You can see $x(t)$ of the anterior foot and upper-body in green (left chart and right chart respectively), $y(t)$ of the anterior foot and upper-body in orange (left chart and right chart respectively), $z(t)$ of the anterior foot and upper-body in red (left chart and right chart respectively)$x(t)$ of the posterior foot and upper-body in black (left chart and right chart respectively), $y(t)$ of the posterior foot and upper-body in grey (left chart and right chart respectively), $z(t)$ of the posterior foot and upper-body in brown (left chart and right chart respectively)

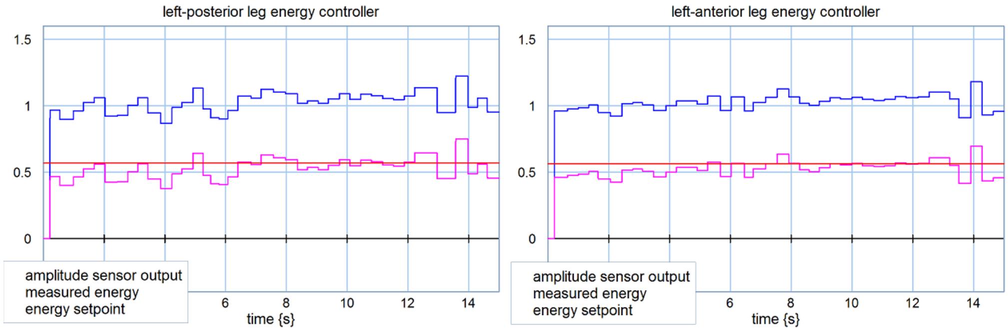

Figure 4.8: Single controller results. In the left chart you can see the elaborated signal coming from the amplitude sensors of posterior half (blue), the energy setpoint (red) and the measured energy (pink), while in the right chart you can see the elaborated signal coming from the amplitude sensors of anterior half (blue), the energy setpoint (red) and the measured energy (pink).

In this configuration the cheetah can run for almost two meters then misalignments and consequently lost of synchronization degrade its race. In this case we are defenseless against these problems because one side of the robot is "open-loop". Again, during the "good" run, from 8 s to 12 s, you can notice that the energy levels are quite constant and close to the setpoints, the knee sensors peaks are quite close each other as well as the motors setpoints. Although the results for this kind of controller are quite interesting, it's necessary to consider this controller unpractical for the prototype because the two halves will not be identical even though they will be built to be very similar. However this tests on the single controller were useful to prove the effectiveness and the flexibility of the energy controller. Although developed for the two-legged cheetah, with a new tuning it works pretty well in the four-legged case.

Independent controllers for every half

In order to partially solve the problems of the single controller, a new control scheme was developed. Every side has its own energy control that computes a controller action starting from the feedback of the sensors of its own half. With this kind of controller we are able to achieve a faster initial transient with respect to the single controller case, i.e. a shorter time to reach the "good" run in which again the energy levels are close to their respective setpoints and the forward locomotion is guaranteed. In figures 4.9, 4.10 and 4.11 we show the best behaviour achieved with this configuration.

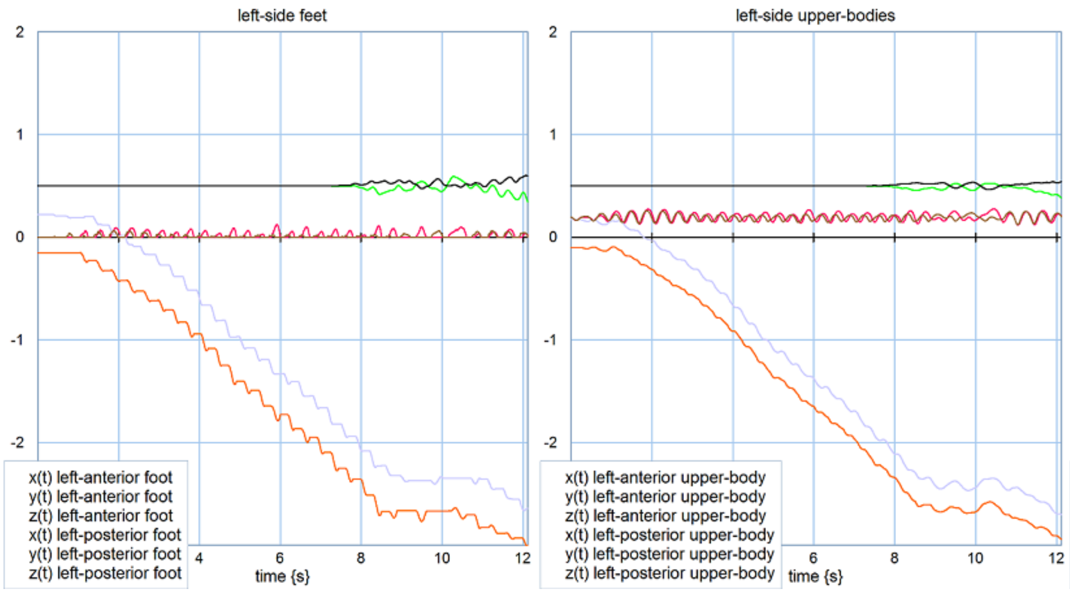

Figure 4.9: Independent controllers results. Positions with respect to time of the left feet and left upper bodies. You can see $x(t)$ of the anterior foot and upper-body in green (left chart and right chart respectively), $y(t)$ of the anterior foot and upper-body in orange (left chart and right chart respectively), $z(t)$ of the anterior foot and upper-body in red (left chart and right chart respectively), $x(t)$ of the posterior foot and upper-body in black (left chart and right chart respectively), $y(t)$ of the posterior foot and upper-body in grey (left chart and right chart respectively), $z(t)$ of the posterior foot and upper-body in brown (left chart and right chart respectively)

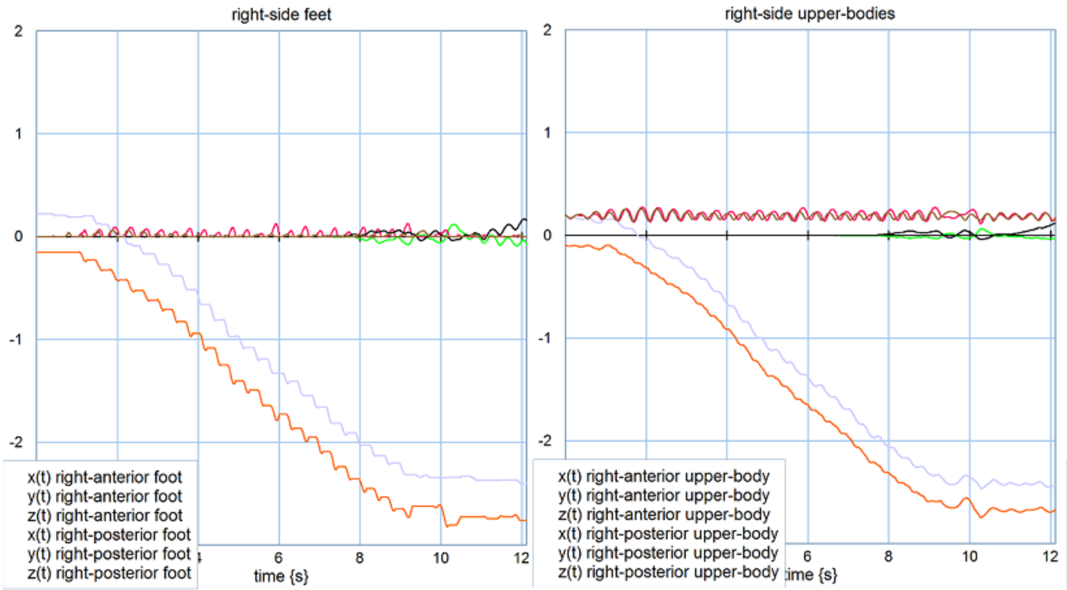

Figure 4.10: Independent controllers results. Positions with respect to time of the right feet and right upper bodies. You can see $x(t)$ of the anterior foot and upper-body in green (left chart and right chart respectively), $y(t)$ of the anterior foot and upper-body in orange (left chart and right chart respectively), $z(t)$ of the anterior foot and upper-body in red (left chart and right chart respectively)$x(t)$ of the posterior foot and upper-body in black (left chart and right chart respectively), $y(t)$ of the posterior foot and upper-body in grey (left chart and right chart respectively), $z(t)$ of the posterior foot and upper-body in brown (left chart and right chart respectively)

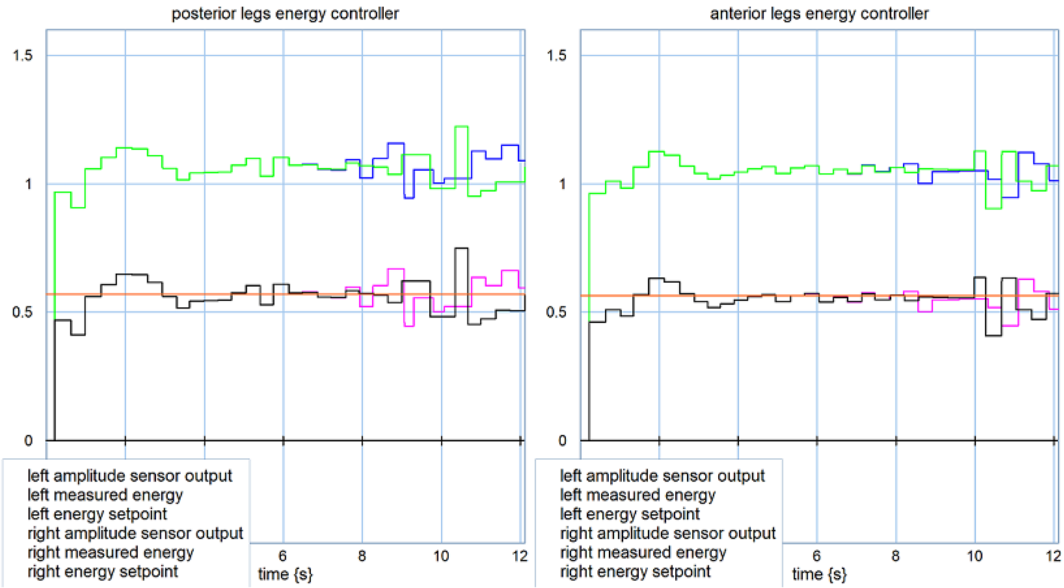

Figure 4.11: Independent controllers results. In the left chart you can see the elaborated signal coming from the amplitude sensors of posterior half (blue), the energy setpoint (red) and the measured energy (pink), while in the right chart you can see the elaborated signal coming from the amplitude sensors of anterior half (blue), the energy setpoint (red) and the measured energy (pink).

In this case the cheetah can run for 2.5 meters and that is a very beautiful result, yet the problem of the lost of synchronization occurs starting from 9 s and the difference tends to increase until the fall, without any possibility to regain it. That problem is pretty normal because the two controllors are totally independent so when the synchronization is lost the controller can't do anything about it.

Independent controllers for every half with an extra supervisor to reduce the energy levels errors of the two sides

In order to solve or actually reduce the lost of syncronization problem another controller block called "Supervisor" was designed. This new controller, as shown in figure 4.12 , is in charge of reduce the energy level difference between the two halves starting from the comparison between the energy mean values.

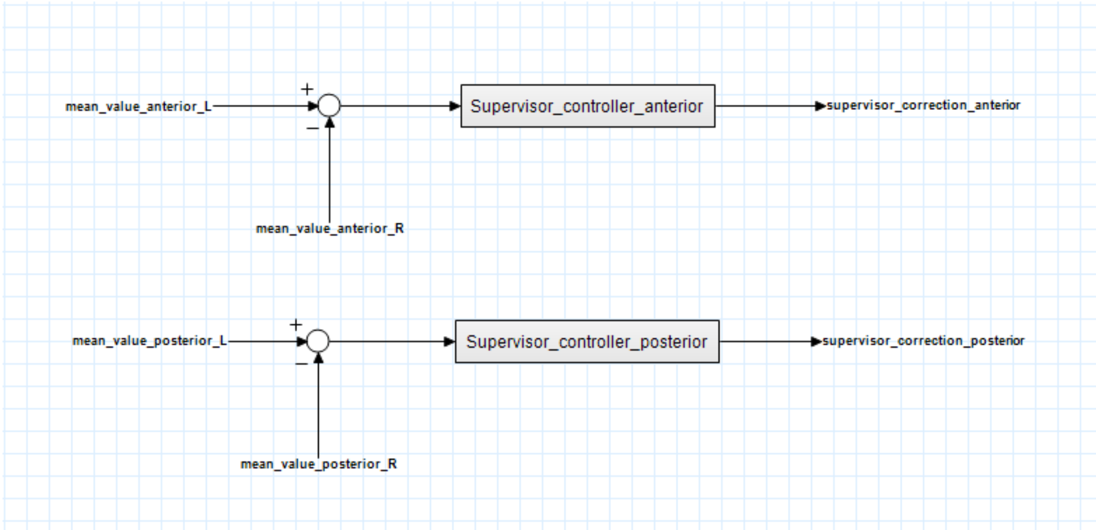

Figure 4.12: "Supervisor" controller. The mean values of the energy of each side is compared in order to generate an error that feeds a controller that generates an extra correction.

This extra controller computes the difference between the mean values of the left anterior leg energy and the right anterior leg one, then it generates a correction, thanks to a suitable tuned proportional controller, for the right anterior motor setpoint. The same procedure is applied to generate a correction for the right posterior motor. In figures 4.13 and 4.14 we instead show the new structure of the controller of the right side of the cheetah, in which it is introduced the measurements of the mean value of the energy error, term naturally proportional to the energy level. With this structure the amplitudes of our setpoints are generated by the following law:

$$Amplitude = Constant\ term - Energy\ controller\ term + Supervisor\ term \tag{4.1}$$

$$Setpoint = Amplitude * Square\ wave \tag{4.2}$$

Where:

- *Constant term*: it is an offset necessary to guarantee that the motors work properly only in compression of the springs and not in extension.

- *Energy controller term*: it is the correction generated by the energy controller.
- *Supervisor term*: it is the correction generated by the supervisor.
- *Square wave*: it is the square wave with the suitable frequency with amplitude from 0 to 1, properly shifted to guaranteed the correct initial condition of the motors.

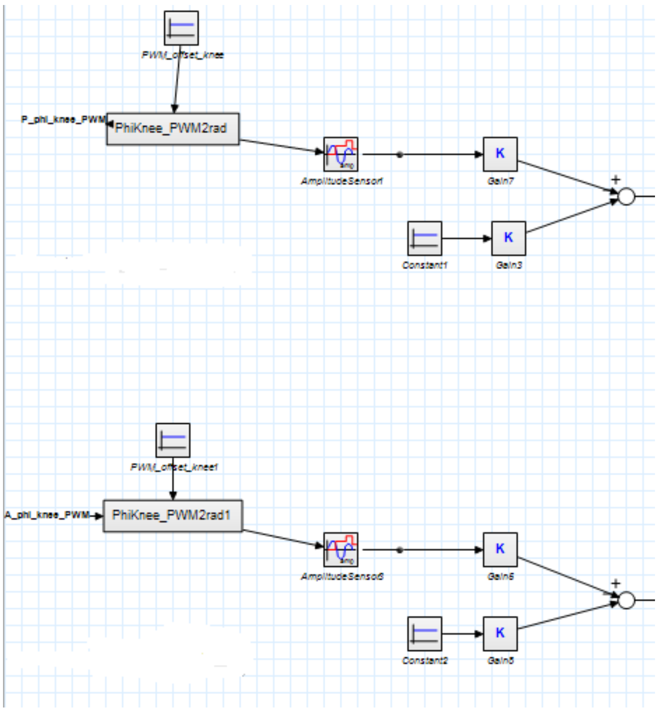

Figure 4.13: "Supervisor" controller structure (part 1). Starting from the reading of the "knee" sensor, we measure through the block "amplitude sensor" the maximum values of the signal coming from the encoders. Then we generate using a gain, suitably defined, the "measured energy" term. We compare this signal with a setpoint to generate and error and to feed the controller. It is worth to notice that the signal form the sensors is a PWM wave so we need to convert it in radians before elaborating it.

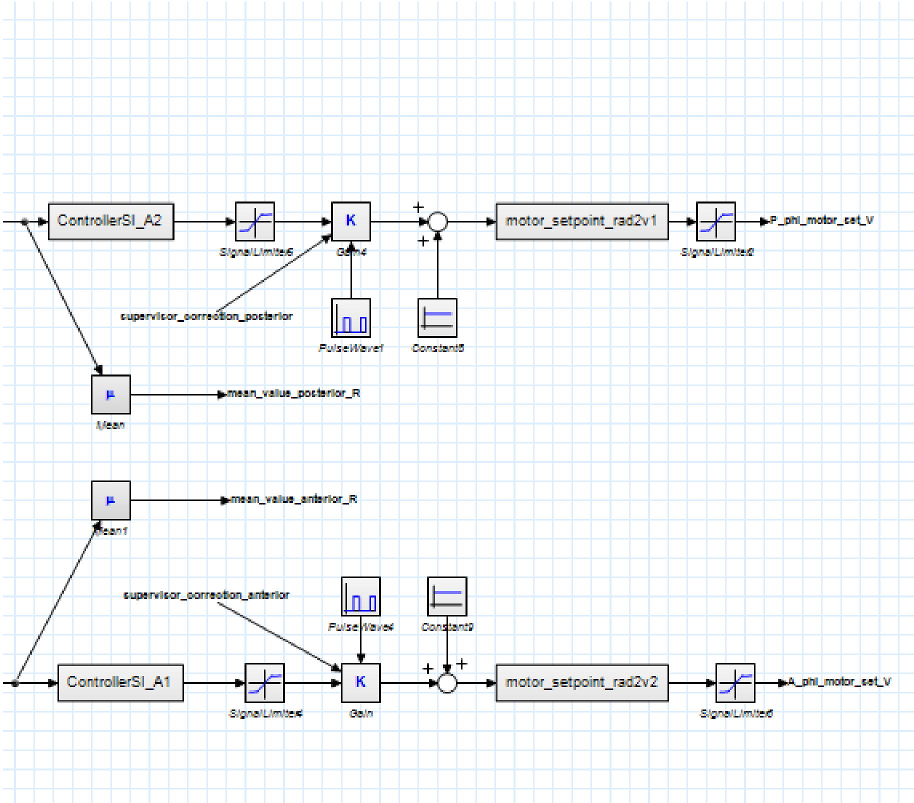

Figure 4.14: "Supervisor" controller structure(part 2). The control action pass through a signal limiter in order to avoid that the correction changes the working of the motor. We only want to compress the springs and not overextend them. Then the correction generated will change the amplitude of the square wave. The offset of the motors them is summed in order to fix the suitable initial conditions. Eventually we convert the signal to PWM in order to feed the motors. We also compute the mean value of the error in order to feed the "supervisor" controller.

The cheetah can still run for more than 2.5 meters and also the degradation due to the losing of the phase is limited. Unfortunately this controller cannot stop completely the losing synchronization problem, but considering what happened for the half-cheetah, in which the prototype was more tolerant than the simulation model, we will implemented it. However we still have the possibility in the prototype to constrain more the two halves with some suitable mechanical devices in the worst case scenario. In figures 4.15, 4.16 and 4.17 we show the global behaviour.

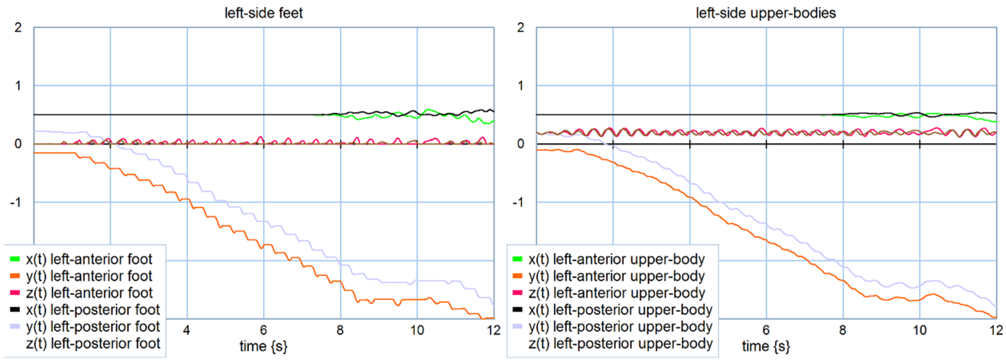

Figure 4.15: Controller results. Positions with respect to time of the left feet and left upper bodies. You can see $x(t)$ of the anterior foot and upper-body in green (left chart and right chart respectively), $y(t)$ of the anterior foot and upper-body in orange (left chart and right chart respectively), $z(t)$ of the anterior foot and upper-body in red (left chart and right chart respectively), $x(t)$ of the posterior foot and upper-body in black (left chart and right chart respectively), $y(t)$ of the posterior foot and upper-body in grey (left chart and right chart respectively), $z(t)$ of the posterior foot and upper-body in brown (left chart and right chart respectively)

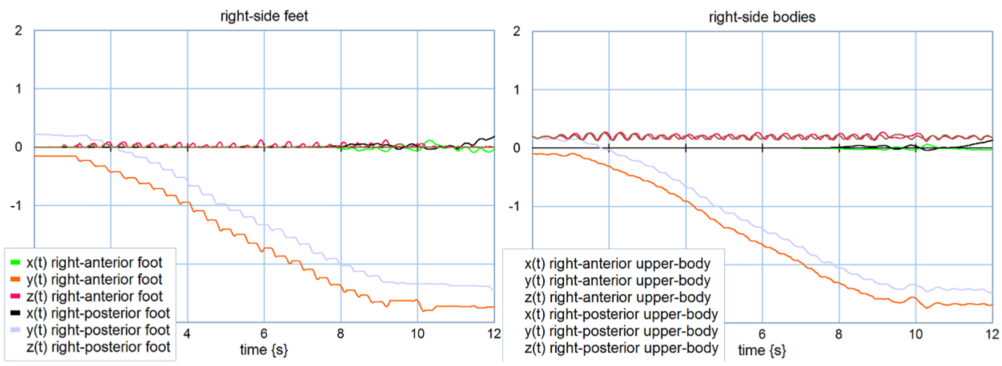

Figure 4.16: Controller results. Positions with respect to time of the right feet and right upper bodies. You can see $x(t)$ of the anterior foot and upper-body in green (left chart and right chart respectively), $y(t)$ of the anterior foot and upper-body in orange (left chart and right chart respectively), $z(t)$ of the anterior foot and upper-body in red (left chart and right chart respectively)$x(t)$ of the posterior foot and upper-body in black (left chart and right chart respectively), $y(t)$ of the posterior foot and upper-body in grey (left chart and right chart respectively), $z(t)$ of the posterior foot and upper-body in brown (left chart and right chart respectively)

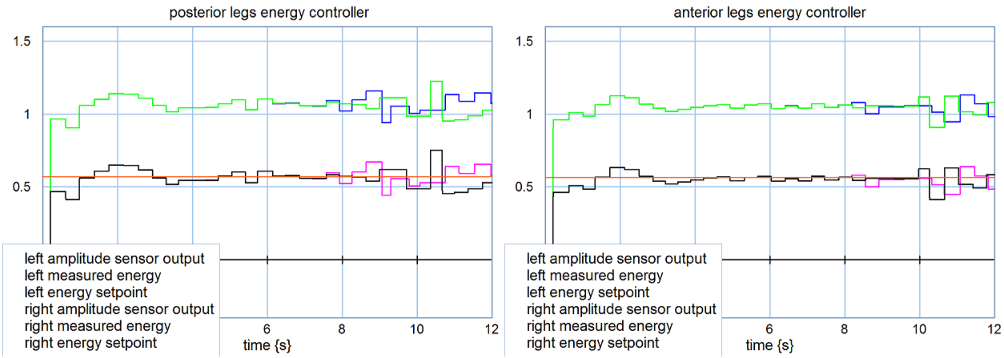

Figure 4.17: Controller results. In the left chart you can see the elaborated signal coming from the amplitude sensors of posterior legs (blue and green), the energy setpoints (red and orange, they overlap) and the measured energies (pink and black), while in the right chart you can see the elaborated signal coming from the amplitude sensors of anterior legs (blue and green), the energy setpoint (red and orange,they overlap) and the measured energy (pink and black).

4.1.3 Results achieved during the simulations

Thanks to the proposed controller we were able to implement the bound gait for a maximum runned distance of 3 m with a speed around 0.42 m/s. This result could not seem amazing, but it's quite reassuring considering the problems of the simulation mentioned in Chapter 2 and deepen in Appendix C. However we were able to achieve a very good mechanical energy efficiency in the simulations as the mechanical cost of transport shows. In figures 4.18 and 4.19, you can see the instantaneous mechanical cost of transport and its mean value measured on an interval of 4 s. The average cost of transport in the time span considered is slightly smaller than 0.25. Although the result is very good, we need to remember that this value was computed starting from the simulations so we can't conclude anything about the actual cost of trasport, but we can consider this value a good estimation of the mechanical cost of transport of our robot.

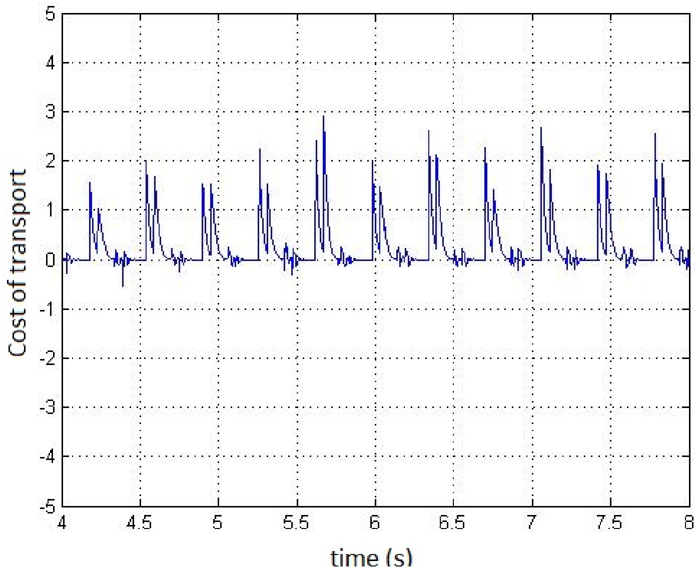

Figure 4.18: Mechanical cost of transport. It was measured by extracting the mechanical power from the motors and by dividing this value for $mass * speed * g$ (with g equal to the gravity acceleration).

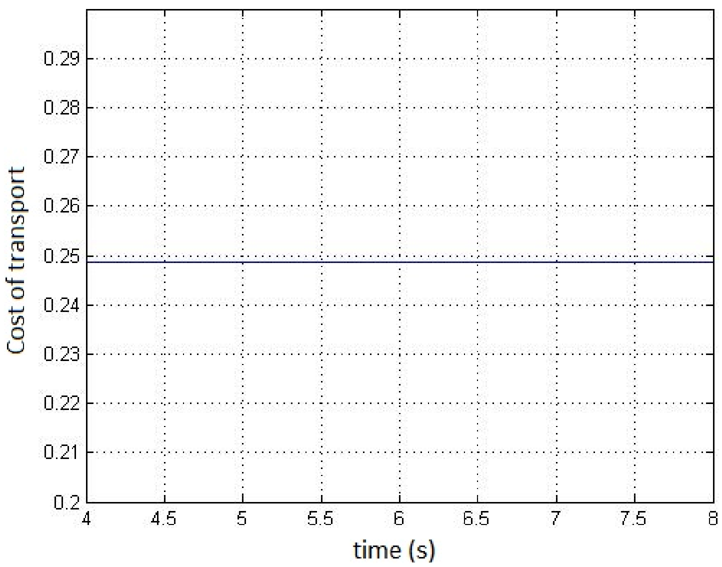

Figure 4.19: Mean value of the MCot. In order to increase the readability of the results we computed the mean value.

Another simulated topic was to add more degrees of freedom to our spine. The idea was to build a center of compliance in the middle of the boom to figure out if new spine configurations could help the locomotion with different gaits. A lot of simulations were also performed in order to implement two more kind of gaits and in particular the transverse gallop and the rotary gallop. Unfortunately two big problems emerged: although the simplifications introduced, moving to the four-legged model slows down a lot the simulations, so much that now fifteen seconds of simulation last fifteen minutes and also, because of the four legs, the unpredictability of the system due to the ground contact model increases exponentially. Because of these problems and because the goal of the project is to built a running prototype it was not possible to implement those two concepts in the simulations. However the same trials will be done on the prototype, hopefully with better results.

4.2 Work on the prototype

In order to conclude the research, we built a four-legged prototype. In this section we will present three different configurations of the robot and the analysis of the results obtained, including the measuraments of the cost of transport.

4.2.1 Prototype design

Starting from the two-legged robot, we built another half, equal to the previous one, but mirrored in terms of motors and SEA springs position in order to balance as much as possible the whole system. We used a rigid metal boom to connect the two halves from spine to spine in order to force the parallelism, but allowing relative rotation. Since the halves could slide on the boom, we fixed them with four nuts. We left the possibility to change the distance between the two sides of the robot and we studied what may happen, with respect to the cost of transport and stability of the run, if we change this distance. In particular, we focused on three configurations:

1. Distance between the two halves equal to 40 cm.

2. Distance between the two halves equal to 30 cm.

3. Distance between the two halves equal to 24 cm.

The configurations were tested using two different controllers. In particular we used an "open-loop" control law in which we injected energy though sending to the motors fixed frequency and amplitude setpoints forgetting about the feedback from the knee sensors, then we implemented the energy controller shown in the previous sections. We chose to implement the "open-loop" controller because we would like to test if exciting the system with its natural frequency was enough to provide locomotion, while we used the energy control to generate locomotion too and to improve the stability of the run if necessary and possible. We started with a distance of 40 *cm* since, theoretically, it should be more stable than shorter ones. In figure 4.20 we show the configuration design.

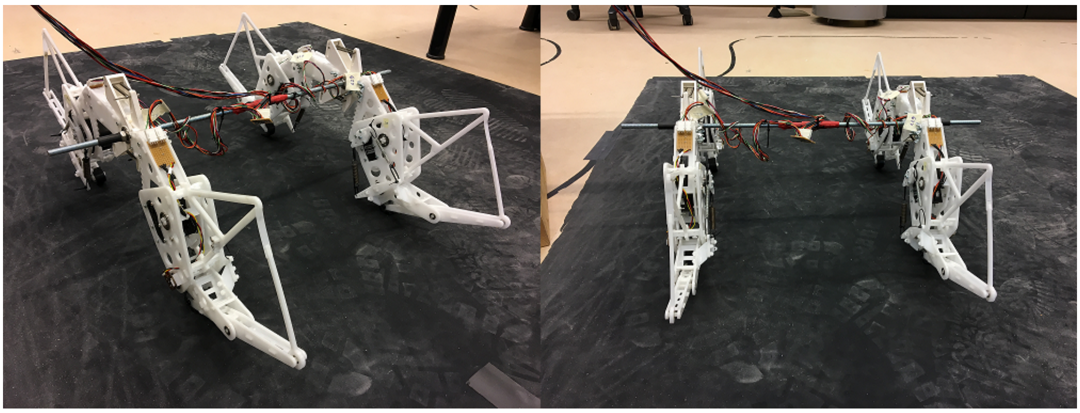

Figure 4.20: Second configuration of the robot. The distance between the two halves now it's equal to 40 cm.

Because the behaviour was good enough, we tried to reduce the lenght of the boom of 10 cm and see if we still could achieve locomotion. In figure 4.21 we show the configuration design.

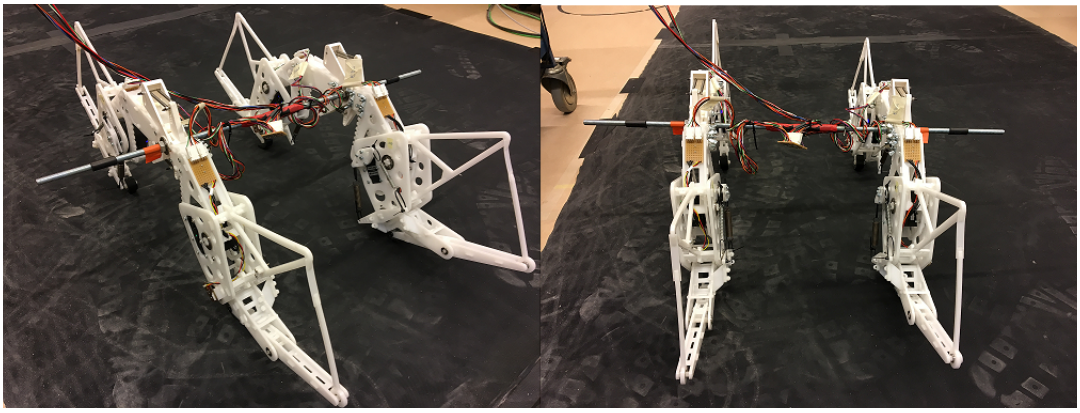

Figure 4.21: Second configuration of the robot. The distance between the two halves now it's equal to 30 cm.

Eventually we reduced again the gap up to 24 cm. In figure 4.22 we show the appearance of the robot.

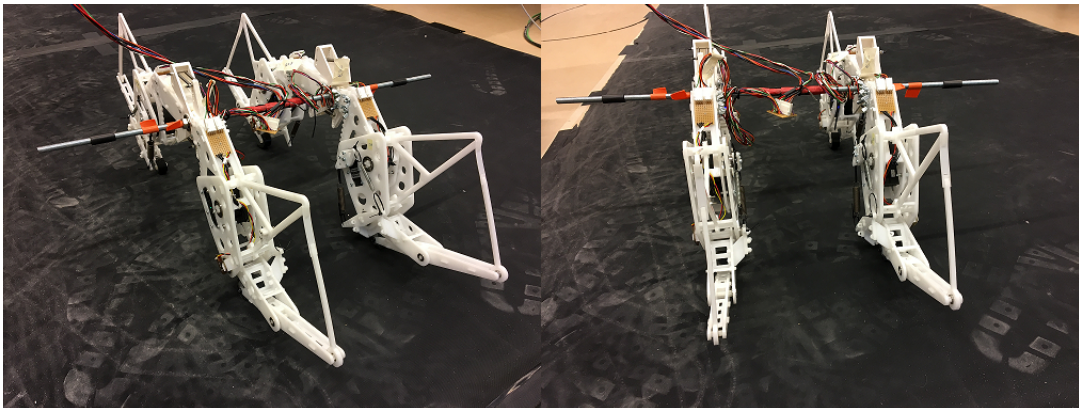

Figure 4.22: Third configuration of the robot. The distance between the two halves now it's equal to 24 cm.

4.2.2 Results achieved on the prototype

In this section we are going to present the results achieved with respect to the three robot configurations and the analysis of the cost of transport for each configuration.

First configuration

We started by trying to generate locomotion by using the "open-loop" control law in which we send to the motors fixed frequency and amplitude square waves as setpoints. Of course, the frequency chosen is the running frequency discovered during the studies on the two-legged robot. Surprisingly, this was enough to achieve a very stable running phase on an almost-straight line. The cost of transport was measured in order to have an index of the efficiency of the robot, taking in mind that no particular optimizations were done to reduce the energy injected and to measure the speed we measured the distance travelled with respect to the time, considering that the robot would run on a perfect straight line even though it was not really true, so the speed values should be likely increased by at least 10-15%. The cost of transport in this case is equal to 1.54, taking into account that the mass of the robot is equal to 2.66 kg and the average speed measured is 0.27 m/s. In figures 4.23, 4.24, 4.25, we show respectively the cost of transport, the power and the current absorbed in a time window of 5 s.

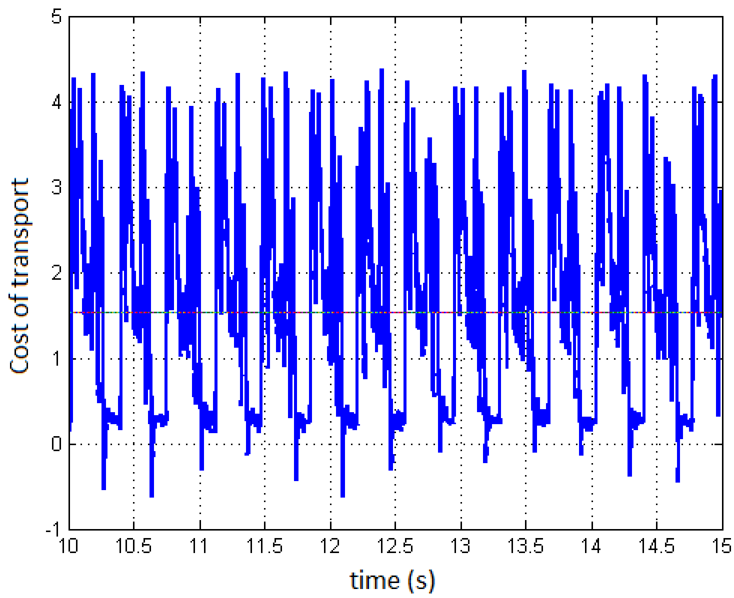

Figure 4.23: Cost of transport (blue line) and its mean value (black line) measured in a time window of 5 s. The Cot is measured as: $\frac{Power_{in}}{mgv}$.

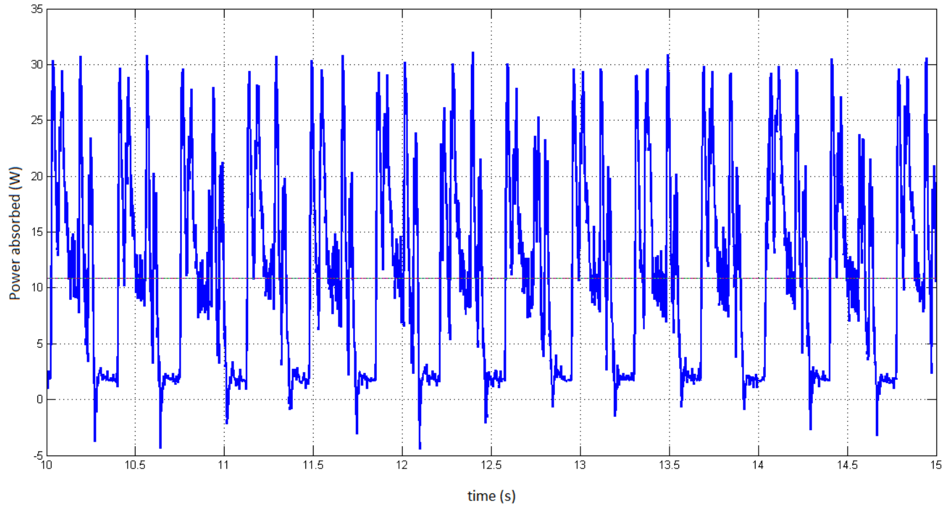

Figure 4.24: Power absorbed in Watt (blue line) and its mean value (black line). The power was measured as: $(V_{dc} - Ri) * i$ where V_{dc} is considered constant and equal to 12 V and the currend is measured by measuring the voltage drop on a resistor and then divided by the resistor value itself.

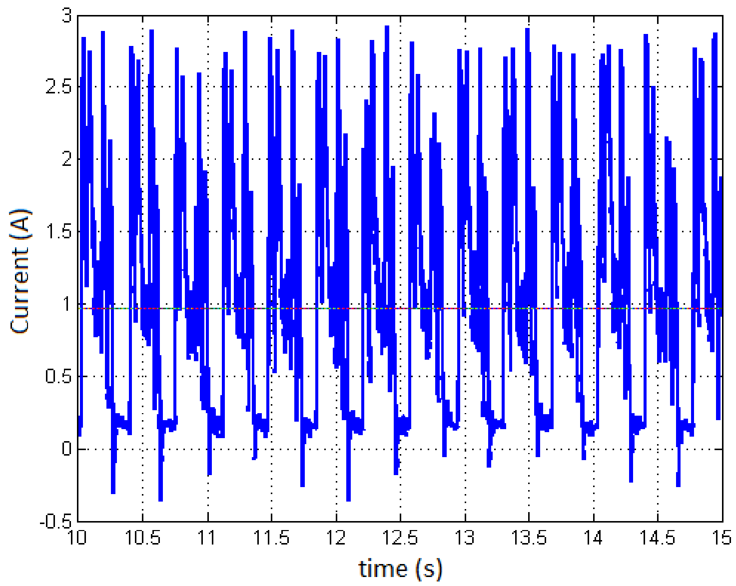

Figure 4.25: Current absorbed in Amphere (blue line) and its mean value (black line). The current was measured as: $\frac{voltage\ drop}{R}$ where R is equal to 0.47 Ω.

While in figure 4.26, we show the outputs of the encoders placed on the knee joints. They are an index of the stability of the run: if the run is stable the peaks of the waves have not big variations.

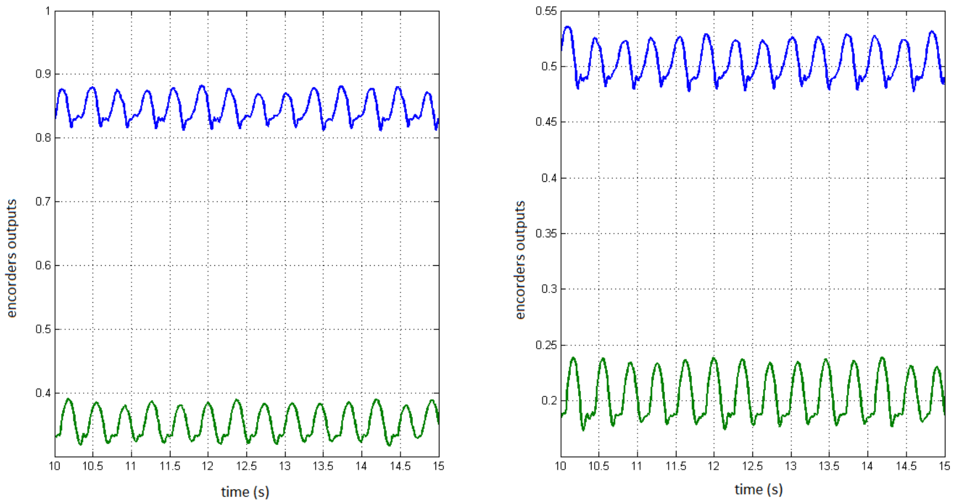

Figure 4.26: Knee sensors outputs of the robot left side (left plot) and right side (right plot). The encoders measure the rotation of the knee joints.

Because the running phase was really stable, we tried to see what would happen if we would increase the amplitude of the setpoints, i.e. increasing the springs compressions. We noticed that up to certain limit values the run was still stable and the speed increased too, up to 0.32 m/s, but more important the cost of tranport decreased to 1.4. This is a very interesting result because it means that by compressing more the springs we actually increase the energy efficiency. In figures 4.27, 4.28 and 4.29, we show respectively the cost of transport, the power and the current absorbed measured in a time window of 5 s.

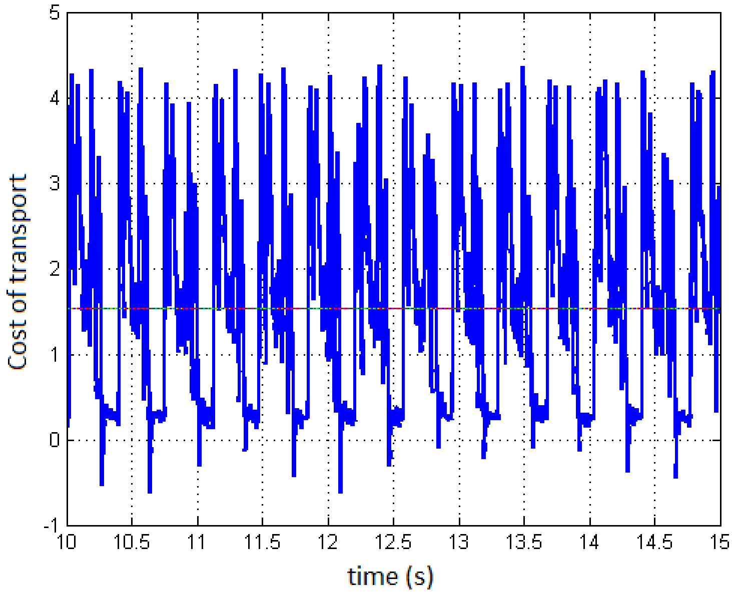

Figure 4.27: Cost of transport (blue line) and its mean value (black line) measured in a time window of 5 s. The Cot is measured as: $\frac{Power_{in}}{mgv}$.

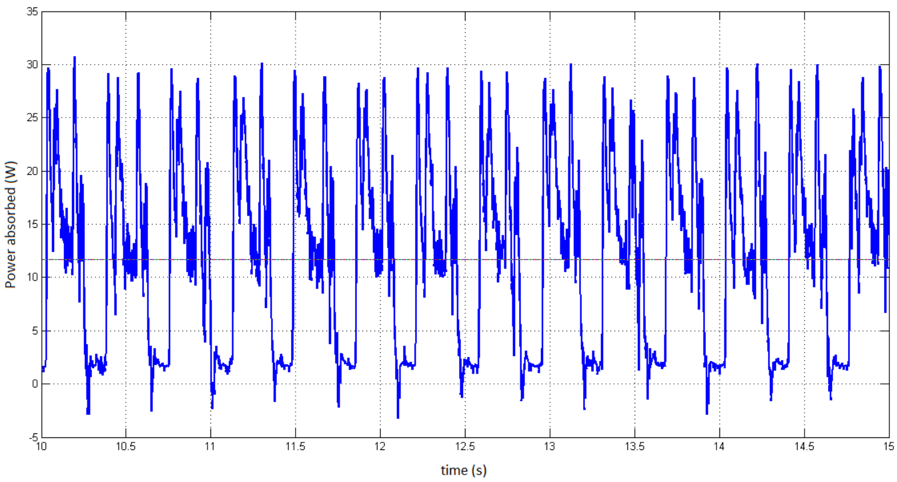

Figure 4.28: Power absorbed in Watt (blue line) and its mean value (black line). The power was measured as: $(V_{dc} - Ri) * i$ where V_{dc} is considered constant and equal to 12 V and the currend is measured by measuring the voltage drop on a resistor and then divided by the resistor value itself.

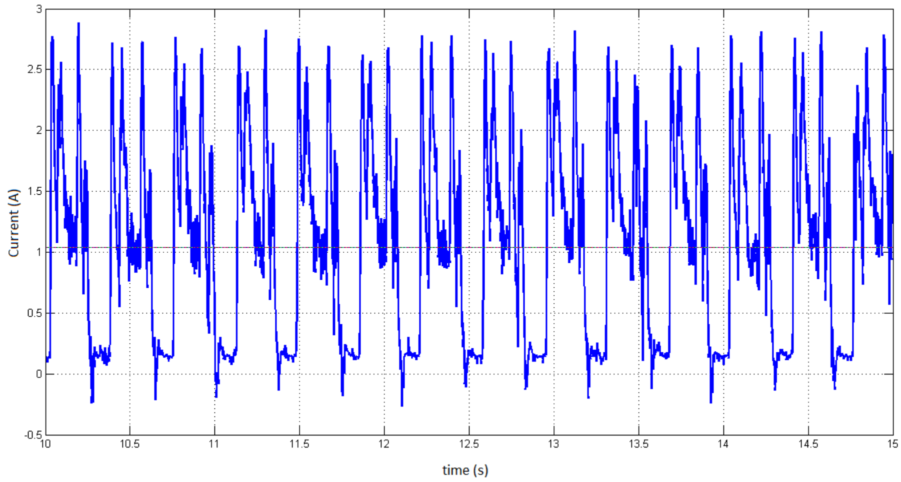

Figure 4.29: Current absorbed in Amphere (blue line) and its mean value (black line). The current was measured as: $\frac{voltage\ drop}{R}$ where R is equal to 0.47 Ω.

While in figure 4.30, we show the outputs of the encoders placed one the knee joints. In this case too, we can observe that the peaks are quite similar each other.

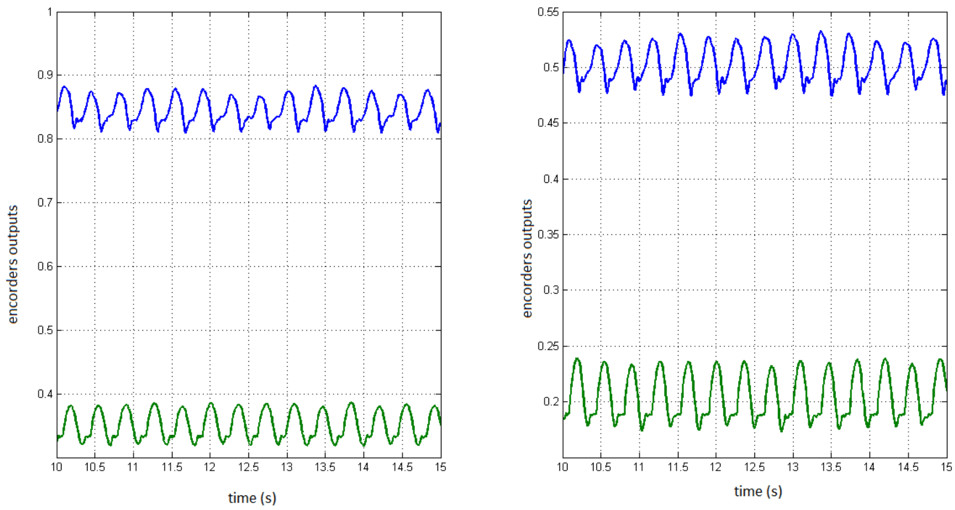

Figure 4.30: Knee sensors outputs of the robot left side (left plot) and right side (right plot). The encoders measure the rotation of the knee joints.

Increasing the amplitude of the setpoints, so increasing the amplitude of the motors movements and compressing more the springs reduces the cost of transport, unfortunately the drawback is the increment of the instabilty of the run. As last trial, we tried to compress even more the springs and use the energy controller to stabilize the run. With this configuration we were able to achieve a cost of transport of 1.3 with an average speed of 0.35 m/s. In figures 4.31, 4.32, 4.33, we show respectively the cost of transport, the power and the current absorbed measured in a time window of 5 s.

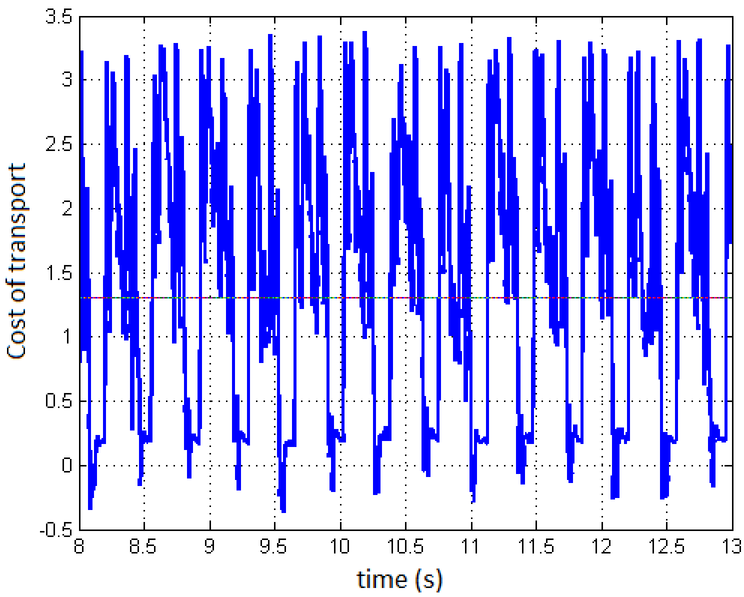

Figure 4.31: Cost of transport (blue line) and its mean value (black line) measured in a time window of 5 s. The Cot is measured as: $\frac{Power_{in}}{mgv}$.

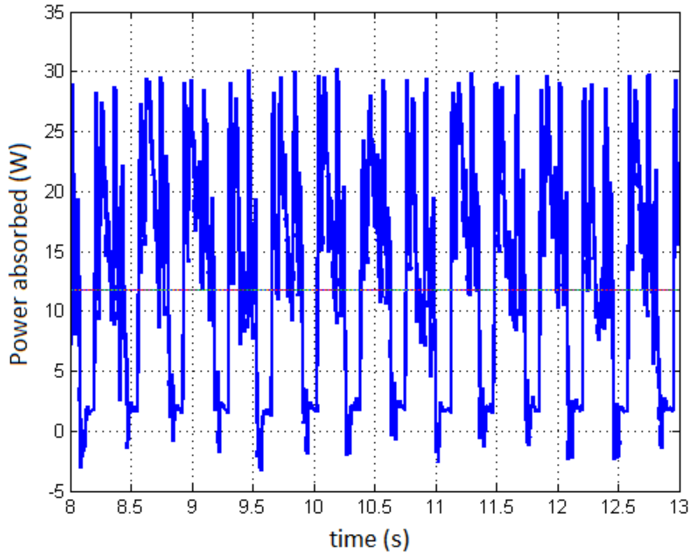

Figure 4.32: Power absorbed in Watt (blue line) and its mean value (black line). The power was measured as: $(V_{dc} - Ri) * i$ where V_{dc} is considered constant and equal to 12 V and the currend is measured by measuring the voltage drop on a resistor and then divided by the resistor value itself.

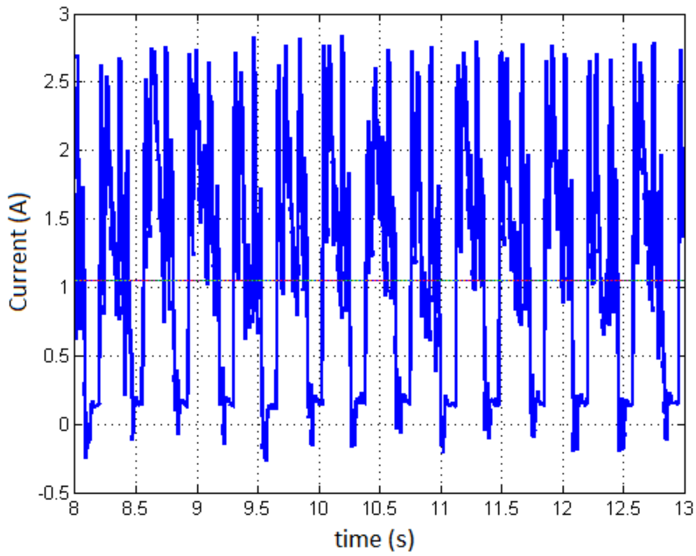

Figure 4.33: Current absorbed in Amphere (blue line) and its mean value (black line). The current was measured as: $\frac{voltage\ drop}{R}$ where R is equal to 0.47 Ω.

In figure 4.34, we show the sensors outputs. We can notice now that the peaks, especially of the right side, tend to vary more than the previous cases, but still not to much to compromize the global stability of the run.

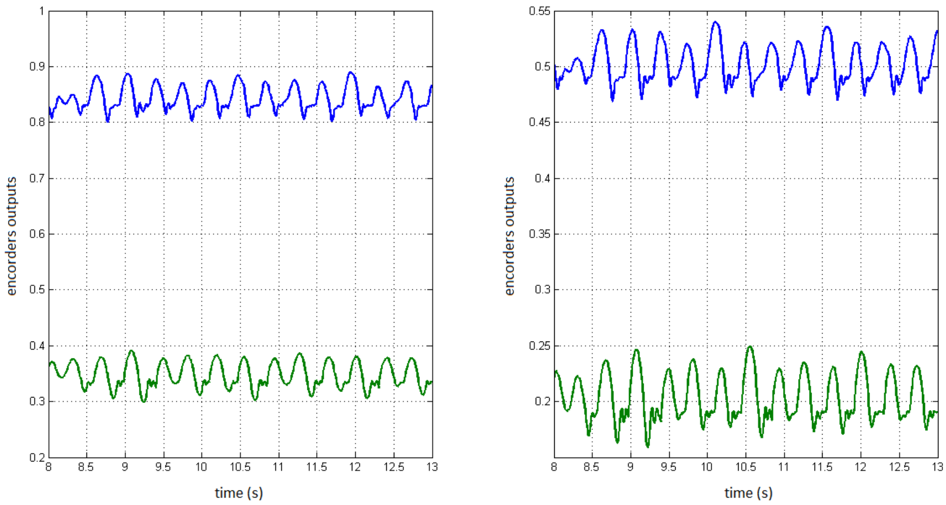

Figure 4.34: Knee sensors outputs of the robot left side (left plot) and right side (right plot). The encoders measure the rotation of the knee joints.

Second configuration

With this configuration, in which the distance between the two halves is 30 cm, we implemented just the "open-loop" control with low energy injected. The behaviour is more instable than the first configuration and consequently the cost of transport is higher and equal to 1.64. We didn't try to use the energy controller because we preferred to work on the first configuration, but nothing avoids us to think that the energy controller may improve the stability of the run in this case too. In figures 4.35, 4.36, 4.37 and 4.38, we show respectively the cost of transport, the power and the current absorbed and the knee sensors outputs measured in a time window of 5 s.

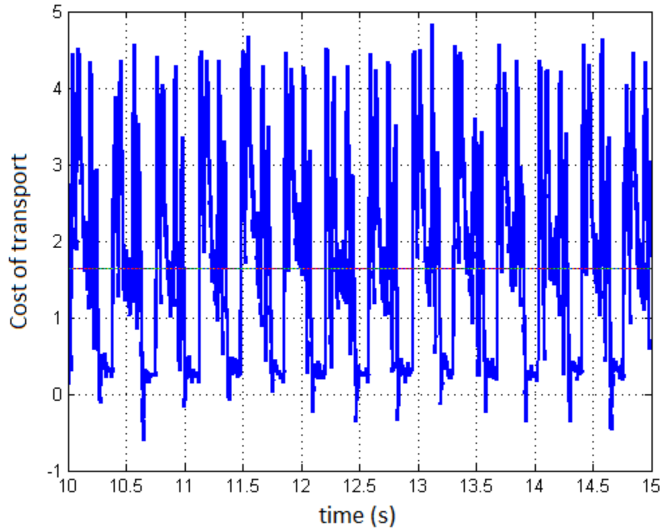

Figure 4.35: Cost of transport (blue line) and its mean value (black line) measured in a time window of 5 s. The Cot is measured as: $\frac{Power_{in}}{mgv}$.

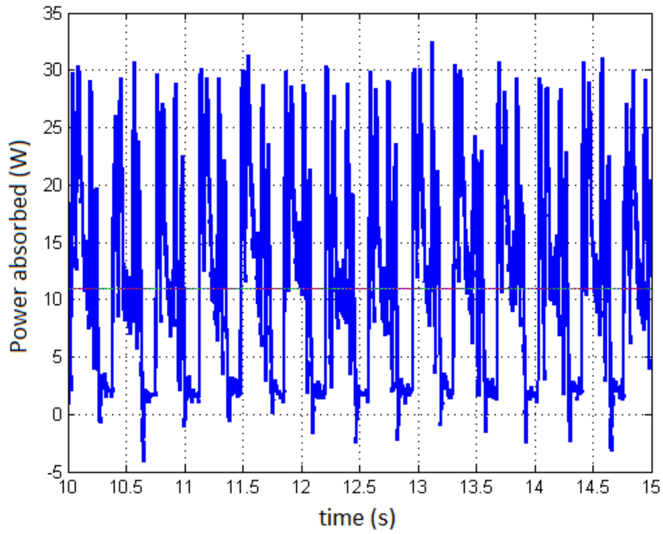

Figure 4.36: Power absorbed in Watt (blue line) and its mean value (black line). The power was measured as: $(V_{dc} - Ri) * i$ where V_{dc} is considered constant and equal to 12 V and the currend is measured by measuring the voltage drop on a resistor and then divided by the resistor value itself.

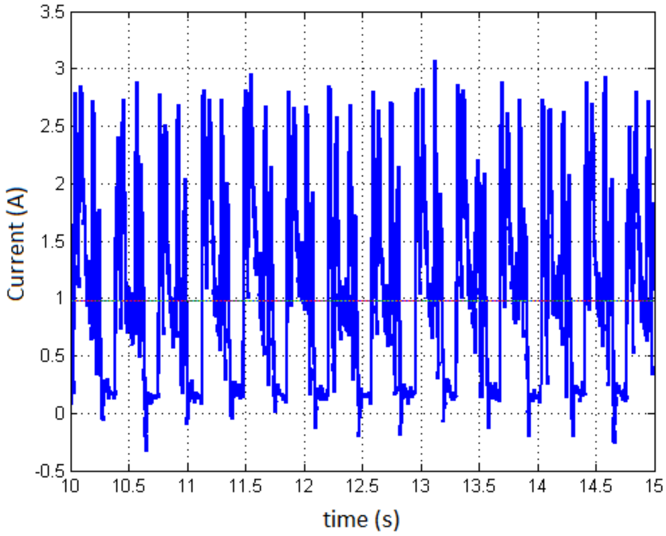

Figure 4.37: Current absorbed in Amphere (blue line) and its mean value (black line). The current was measured as: $\frac{voltage\ drop}{R}$ where R is equal to 0.47 Ω.

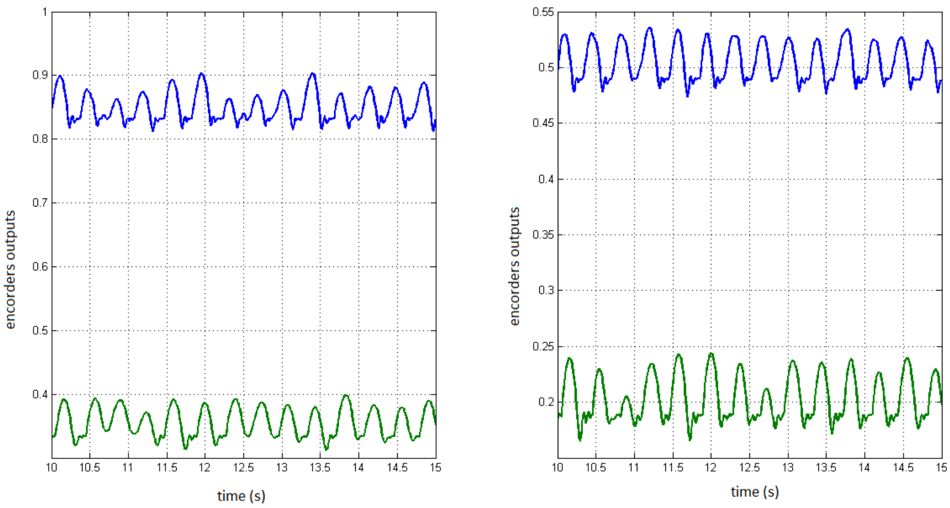

Figure 4.38: Knee sensors outputs of the robot left side (left plot) and right side (right plot). The encoders measure the rotation of the knee joints.

From figure 4.38, we can observe the instability increment due to the different configuration.

Third configuration

The last cofiguration is the most instable one and the one with the worse cost of transport, 1.87, but it still deserves to be prensented since locomotion was achieved. In figures 4.39, 4.40, 4.41 and 4.42, we show respectively the cost of transport, the power and the current absorbed and the knee sensors outputs measured in a time window of 5 s. As in the previous case, we implemented just the "open-loop" control law since we preferred to focus on the most naturally stable one. In figure 4.42, we can notice the instability problem in the peaks of the waves since they are quite difference.

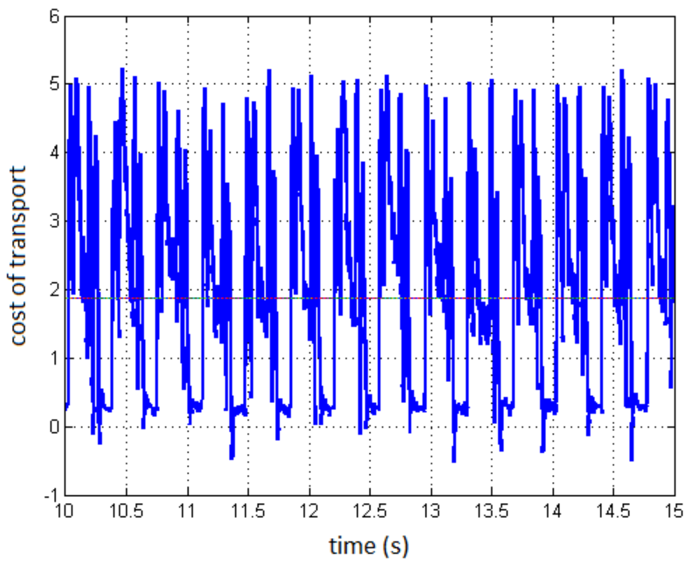

Figure 4.39: Cost of transport (blue line) and its mean value (black line) measured in a time window of 5 s. The Cot is measured as: $\frac{Power_{in}}{mgv}$.

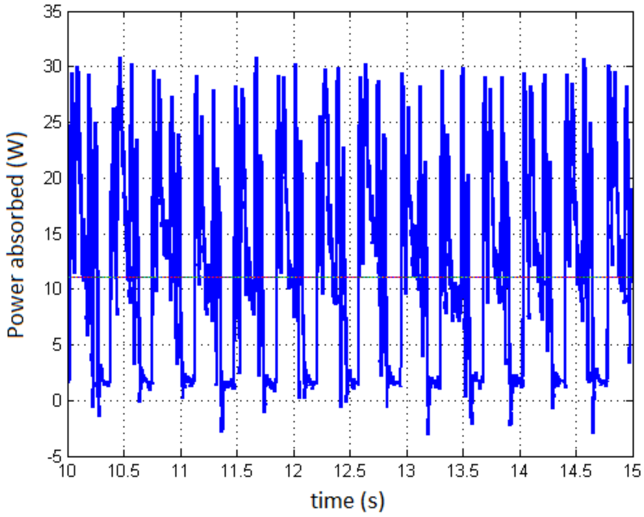

Figure 4.40: Power absorbed in Watt (blue line) and its mean value (black line). The power was measured as: $(V_{dc} - Ri) * i$ where V_{dc} is considered constant and equal to 12 V and the currend is measured by measuring the voltage drop on a resistor and then divided by the resistor value itself.

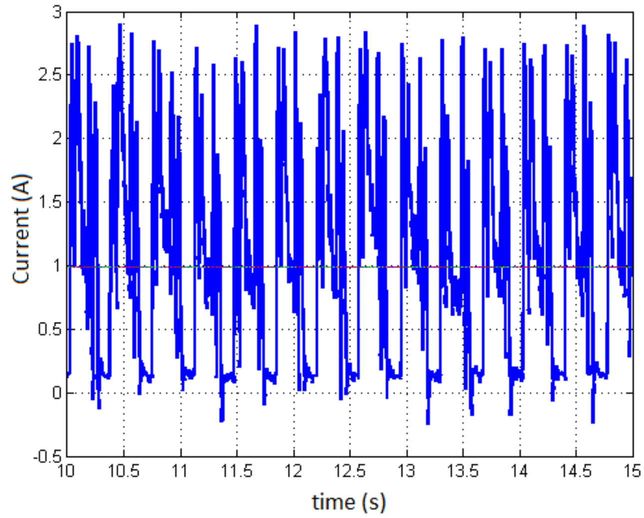

Figure 4.41: Current absorbed in Amphere (blue line) and its mean value (black line). The current was measured as: $\frac{voltage\ drop}{R}$ where R is equal to 0.47 Ω.

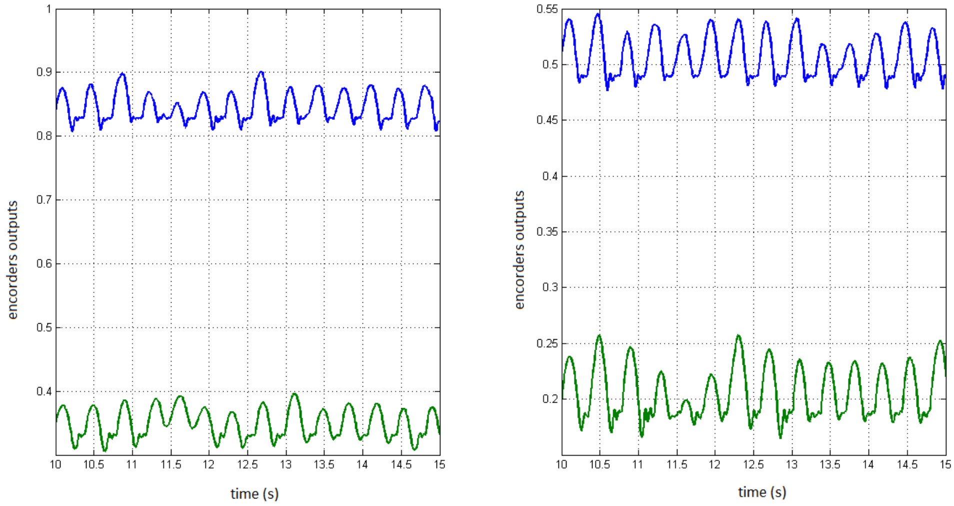

Figure 4.42: Knee sensors outputs of the robot left side (left plot) and right side (right plot). The encoders measure the rotation of the knee joints.

Conclusive summary

In table 4.1, we present the cost of transport values obtained with respect to the configurations of the robot and the controllers implemented in order to summarize the results.

Configuration	Control law	Best Cot	Average speed
1	Open-loop	1.4	0.32 m/s
1	Energy controller	1.3	0.35 m/s
2	Open-loop	1.64	0.26 m/s
3	Open-loop	1.87	0.23 m/s

Table 4.1: Cost of transport values obtained for the different configurations of the robot.

4.2.3 Manual navigation

We tried also to implement different gaits than the bound, unfortunately with no success. Nevertheless we noticed that when the phase between the two halves grew, the bound gait implicitly assumes phase equal to 0, the robot, rather than changing the gait, started to turning in relation of which side is delayed. With this idea in mind using a simple potentiomenter as input we can steer the robot to right or left. Starting from the signal from the potentiometer, from 0 to 5 V, we map it from -0.025 to 0.025 since the maximum delay allowed to generate locomotion is 0.05 and because we don't want to delay totally one side with respect to the other, but rather we want to delay and anticipate both of half the value. In figure 4.43, we show the procedure to generate the square waves that we can dalay manually through the potentiometer.

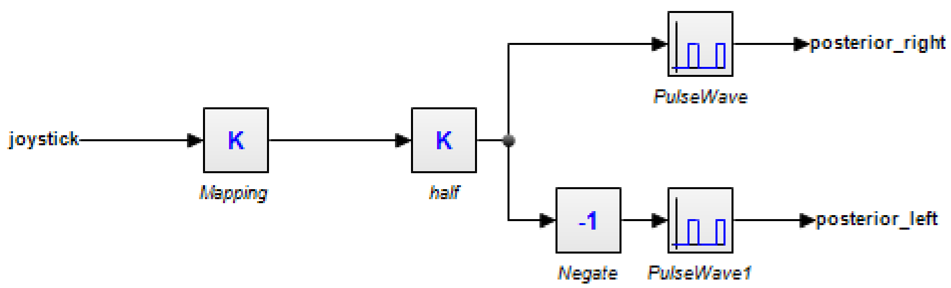

Figure 4.43: Generation of the variable delay square waves. The signal from the potentiometer is mapped from -0.025 to 0.025 and then it is sent to the pulse generator blocks as input in order to change the phase.

In figure 4.44, we show a left turning sequence. After the turn, it is still possible to keep running on a straight line or turning again in both directions.

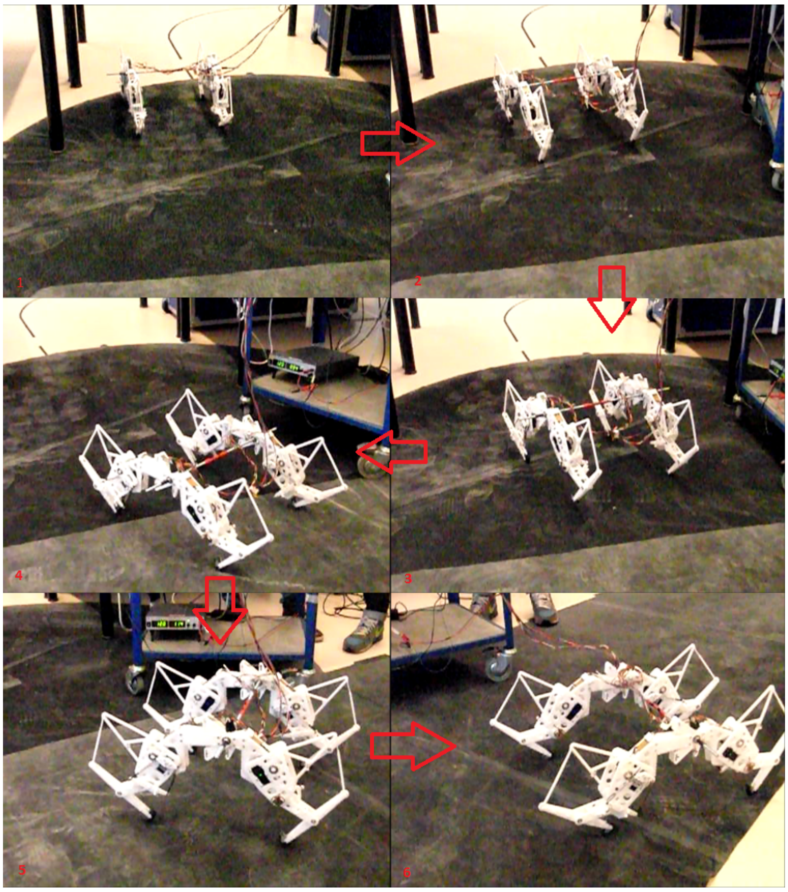

Figure 4.44: 90 degrees turning sequence.

4.2.4 Final considerations

The best cost of transport achieved is equal to 1.3, but nothing forbids us to think that it may be even lowered by just slightly increasing the compression of the springs and by correcting the stability with a stronger control action. We need to remember that for the two-legged robot the cost of transport was equal to 0.36, since it was easier to control just two legs even if we would compress much more the springs. In the four legs case we were more limited in doing that because the instability grows exponentially. Another possible reason for its high value is that the servomotors, beacuse of the gears that have inside, are not very energy efficient. We need more electrical power that the ideal one in order to have the right mechanical power to move the robot. We don't have also to forget that the speed measuraments are not very accurate, but also very pessimistic since we consider the runnnig perfectly along a straight line, thing that doesn't actually happen. Last, but not least, we need to remember that no optimization operations were performed to find the optimal frequency, width or amplitude of the pulses. If we sum all together, we can claim that we still have margins of improvement for the cost of transport and the energy efficiency of the robot even though the results are already quite good. Possible improvements could be to optimize the running frequency with repect to the cost of transport to decrease it even more and to improve the mechanical structure by redesigning it in order to build something more cheetah-resembling in terms of joints and movements and to achieve much higher running speeds.

Chapter 5

Conclusions

The goal of the project was to build and control a quadruped robot that can run fast and in an energy efficiency. Although we have not been able to achieve fast running, the average speed is equal to 0.35 m/s, mainly due to the limits of the mechanical structure, since the legs in the end can just jump on the same spot, we have been able to achieve running with the bound gait in a quite efficient way, the cost of transport is equal to 1.3, and the possibility to steer the robot to left or right through a simple manual input. In order to do that, we don't have to forget the previous steps, necessary to complete the research and in particular the theoretical background (Chapter2) and all the work done on the two-legged robot (Chapter 3), with which we achieved a very good value for the cost of transport, 0.36, with average running speed equal to 0.35 m/s, fundamental for the design of the control concepts used in the final prototype (Chapter 4). It is relevant to notice that combining an elastic spine with jumping legs and a controller that, through a clock-based pulses motors setpoints, excites the system with its own running frequency is enough to provide energy efficient locomotion. If we combine it with a control system that changes the amplitude of these pulses in relation to the energy stored in the SEA springs, we achieve very stable locomotion even with lower cost of transport. This work proves once more that in order to increase the efficiency of legged robots it is possible to add passive elements that can store and release energy during the walk or the running in order to mimic what the animal muscles do. Mimic the main features of the animal world help us to build robots that are much more energy efficient.

Bibliography

[1] 20-sim. http://www.20sim.com/.

[2] Arduino. https://www.arduino.cc/en/main/arduinoboardmega2560.

[3] K. Jayarajan D.C. Kar, K. Kurien Issac. Gaits and energetics in terrestrial legged locomotion, 2003.

[4] Vincent Duindam, Alessandro Macchelli, Stefano Stramigioli, and Herman Bruyninckx. *Modeling and control of complex physical systems: the port-Hamiltonian approach*. Springer Science & Business Media, 2009.

[5] G. A. Folkertsma, S. Kim, and S. Stramigioli. Parallel stiffness in a bounding quadruped with flexible spine. In *2012 IEEE/RSJ International Conference on Intelligent Robots and Systems*, pages 2210–2215, Oct 2012.

[6] G. A. Folkertsma, Arjan J. van der Schaft, and S. Stramigioli. Morphological computation in a fast-running quadruped with elastic spine.

[7] Bart L. Hoorweg. Morphological computation of a robot cheetah focussing on the role of the spine. Master's thesis, University of Twente, 2016.

[8] P. M. Krishna, R. P. Kumar, and S. Srivastava. Energetics of level walking trot gaits in quadruped robots. In *2012 IEEE International Conference on Robotics and Biomimetics (ROBIO)*, pages 61–65, Dec 2012.

[9] College of Veterinary Medicine University of Minnesota. http://vanat.cvm.umn.edu/gaits/index.html.

[10] H. W. Park, M. Y. Chuah, and S. Kim. Quadruped bounding control with variable duty cycle via vertical impulse scaling. In *2014 IEEE/RSJ International Conference on Intelligent Robots and Systems*, pages 3245–3252, Sept 2014.

[11] H. W. Park, Sangin Park, and S. Kim. Variable-speed quadrupedal bounding using impulse planning: Untethered high-speed 3d running of mit cheetah 2. In *2015 IEEE International Conference on Robotics and Automation (ICRA)*, pages 5163–5170, May 2015.

[12] RaMstix. https://www.ram.ewi.utwente.nl/ecssoftware/ramstix/docs/index.html.

[13] Boston Dynamics Robot. http://www.bostondynamics.com/.

[14] S.Gracovetsky and S. Iacono. Energy transfers in the spine engine, 1987.

[15] T.A.McMahon. The role of compliance in mammalian running gaits, 1985.

[16] Wikipedia. https://en.wikipedia.org/wiki/gait.

Appendix A

Dirac structures

Because we dealt with Port-Hamiltonian system and Bound-Graph notation, it is worth to introduce the concept of Dirac structure [4]. In order to do that, we have to define two abstract finite dimension structures:

- F: space of flows. Its elements are called flow vectors $f \in F$
- E: space of efforts. Dual space of $F \Leftrightarrow E = F^*$

The resulting space will have dimension $F \times E \Leftrightarrow F \times F^*$ and we will call it the space of port variables. On the space of port variables we can define the power as:

$$P = \langle e|f \rangle \qquad (f, e) \in F \times F^* \tag{A.1}$$

where $\langle e|f \rangle$ indicates the dual product. If f is a column vector and e is a row vector then the power is simply the product ef. So because also the effort is a column vector, the power can be written eventually as:

$$P = e^T f \tag{A.2}$$

Now we are able to give a definition of Dirac structure: *a Dirac structure on $F \times F^*$ is a subspace $D \subset F \times F^*$ such that:*

1. $\langle e|f \rangle = 0 \qquad (f, e) \in D$

2. $dim D = dim F$

The first property corresponds to the power-conservation and expresses the fact that the total power entering or leaving a Dirac structure is 0. It can also be shown that the maximal dimension of any subspace $D \subset F \times F^*$ that satisfies property 1 is equal to $dim F$.

Appendix B

Electronic structure

In this section the electronic structure used for the prototype will be presented. We will focus our attention on three aspects:

1. The motors
2. The sensors
3. The boards

B.1 The motors

The motor used are HerkuleX servomotors model DRS-0602 of Dongbu Robot with parameters shown in table B.1.

Dimensions	56mm(W) X 35mm(D) X 46mm(H)
Weight	145g
Input voltage	9.5 - 14.8 V
Rated voltage	12 V
Stall torque	7.6 Nm
Max speed	0.164s/60 degrees
Encoder resolution	12960 steps/360 degress

Table B.1: Servomotor main parameters.

These servomotors can be progammed easily through the suitable software in order to choose the type of control, proportional, proportional-integral or proportiona-integral-derivative, the gain of the controller, the range of positions allowed and other motor parameters. We choose to work with only a proportional controller because we don't require particular dynamic performances. As shown in figure B.1, every motor has four input/output wires:

- GND: connected to the ground pin of the power supply.
- VDD: connected to the positive voltage pin of the power supply.
- TXD: output signal of the servomotor.
- RXD: input signal of the servomotor.

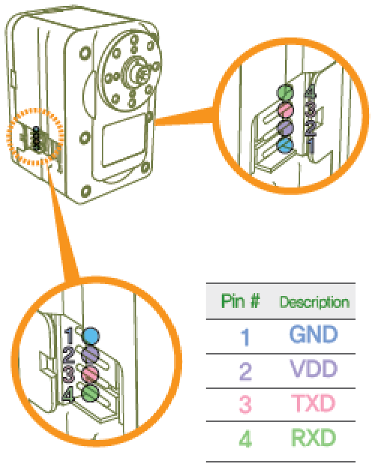

Figure B.1: DRS-0602 sketch in order to hightlight input and output pins of the motor.

In order to communicate with the motors, we wrote a code on the Arduino Mega 2560. This code will be explained in the next section.

B.2 The sensors

The sensors used, AS5600 by ams, one for each leg, measure the rotation of the "knee" rotational joint. These sensors are magnetic rotary position sensors with a resolution of 12-bits, 4096 steps over 360 degress. You can see a schematic draw to highlight the pins in figure B.2.

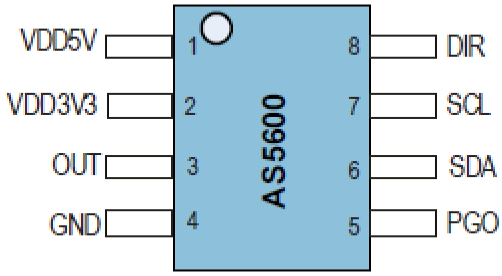

Figure B.2: AS5600 schematic draw to highlight the pins.

The only three pins that we need are the voltage, VDD, the ground, GND and the sensor output, OUT. In figure B.3 is shown the sensor placed on its board, the printed circuit and the three wires above mentioned.

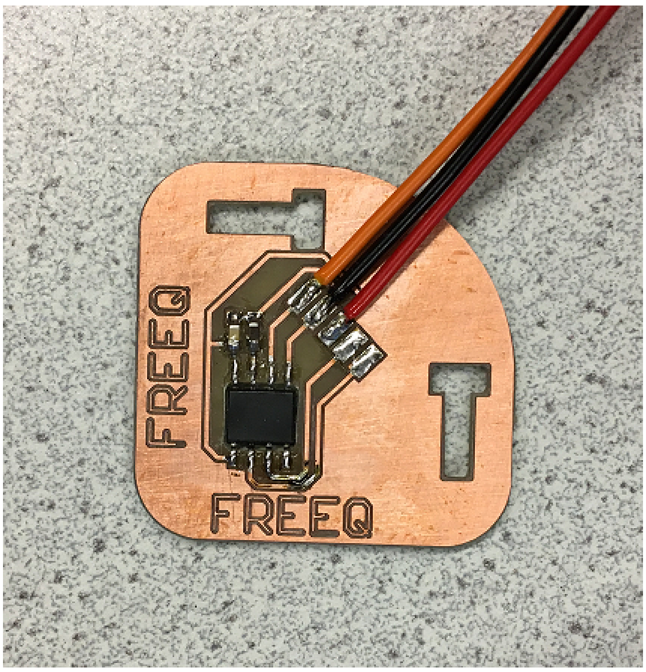

Figure B.3: AS5600 installed on its board. The sensors are placed on the knee joints.

B.3 The boards

The brain of the cheetah is essentially composed by two boards:

- Arduino Mega 2560
- RaMstix

The Arduino Mega 2560 [2] is used to communicate with the motors. Herkulex motors have a library of function in order to be controlled through and Arduino. Below we show the code written ad in particular the loop reading and writing action performed. The Arduino recieves the input signals for the motors from the RaMstix, board in which all the computations are done. Unfortunately the RaM stix has just two DAC outputs, so for the other two motors we use the PWM pins. In the first four line of code you can see the functions "analogRead()" and "pulseIn()" that read and acquire the signal from the RaMstix.

```
adcval = analogRead(adcpin);
adcval2 = analogRead(adcpin2);
pwmleftanterior = pulseIn(adcpin3,HIGH);
pwmleftposterior = pulseIn(adcpin4,HIGH);
```

Then, using the function "Herkulex.getPosition()", the position of each motor is read and assigned to a runtime variable.

```
motorpos = Herkulex.getPosition(motorid);
motorpos2 = Herkulex.getPosition(motorid2);
motorpos3 = Herkulex.getPosition(motorid3);
motorpos4 = Herkulex.getPosition(motorid4);
```

Now we map the control signals, coming from the RaMstix, to motors setpoints in suitable ranges and assign those values to the variables "motorsetpoint", while the motors positions are mapped in a PWM signal from 0 to

255.

```
motorsetpoint = map(adcval,minadcval,maxadcval,minmotorposright,maxmotorposright);
motorsetpoint2 = map(adcval2,minadcval,maxadcval,minmotorposright,maxmotorposright);
motorsetpoint3 = map(pwmleftanterior,0,2000,minmotorposleft,maxmotorposleft);
motorsetpoint4 = map(pwmleftposterior,0,2000,minmotorposleft,maxmotorposleft);
```

```
pwmval = map(motorpos,minmotorposright,maxmotorposright,0,255);
pwmval2 = map(motorpos2,minmotorposright,maxmotorposright,0,255);
pwmval3 = map(motorpos3,minmotorposleft,maxmotorposleft,0,255);
pwmval4 = map(motorpos4,minmotorposleft,maxmotorposleft,0,255);
```

Eventually the motors position are written on the used pins through "analogWrite()" and the commands to the motors are sent through the function "Herkulex.moveOne()".

```
analogWrite(pwmpin,pwmval);
analogWrite(pwmpin2,pwmval2);
analogWrite(pwmpin3,pwmval3);
analogWrite(pwmpin4,pwmval4);
```

```
Herkulex.moveOne(motorid, motorsetpoint, 1, LEDGREEN);
Herkulex.moveOne(motorid2, motorsetpoint2, 1, LEDBLUE);
Herkulex.moveOne(motorid3, motorsetpoint3, 1, LEDBLUE);
Herkulex.moveOne(motorid4, motorsetpoint4, 1, LEDGREEN);
```

Furthermore, the Arduino is powered by 5V DC and, as we said before, it is connected to motors and RaMstix. The control system is instead managed by the RaMstix [12], figure B.4 and in particular by the FPGA installed on it. The sensors are directly connected to the RaMstix and we communicate with it in order to start and stop the robot.

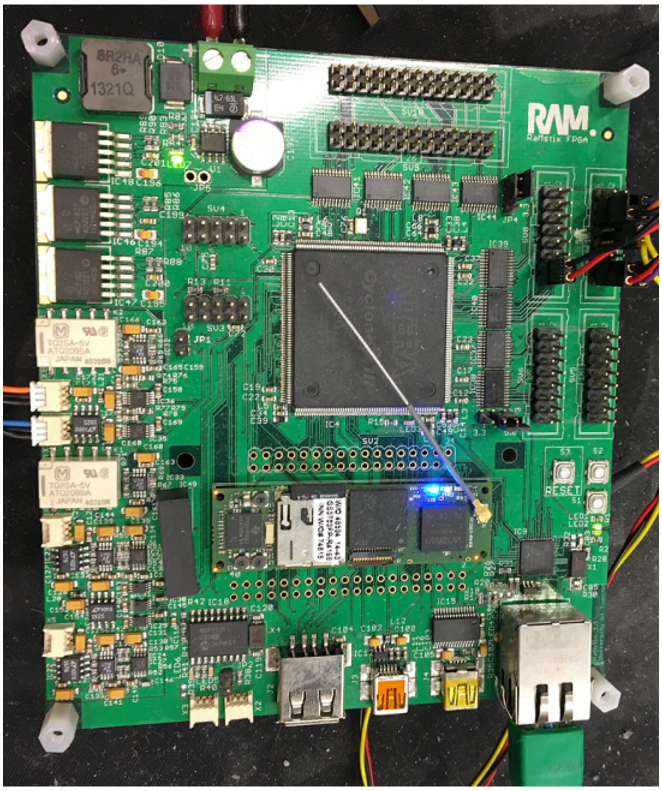

Figure B.4: RaMstix.

Appendix C

Ground contact model

It is worth to spend few words on the ground contact model used in our simulations and understand why it is the main reason of their slowness and unpredictability. The foot-ground contact is modelled with elastic-dissipative relations in the 3-dimensional space, so we will consider more than one contributes in our studies. If we study effort vector:

$$P = \begin{pmatrix} 0 \\ 0 \\ 0 \\ Fc_x \\ Fc_y \\ Fn_z \end{pmatrix} \tag{C.1}$$

We neglect the rotations contributes, while along the three main axis of translation, x, y and z we have three different terms dependend of the vertical position of the foot. In particular if the event contact occurs:

$$\begin{cases} Fc_x = -\mu * (-K * z) * tanh(slope * v_x) \\ Fc_y = -\mu * (-K * z) * tanh(slope * v_y) \\ Fn_z = -K * z - D * abs(z) * v_z \end{cases} \tag{C.2}$$

Elsewhere:

$$
\begin{cases}
Fc_x = 0 \\
Fc_y = 0 \\
Fn_z = 0
\end{cases}
\tag{C.3}
$$

where:

- K: elastic coefficient of the ground. It is equal to $1e6$ in order to have maximum deflection for 1 mm of displacement.
- D: damping coefficient of the ground. It is set equal to $1e6$.
- μ: Coulomb friction, an empirical property of the materials in contact. It is equal to 1.
- *slope*: coefficient choosen to detect the sign of v_x and v_y. It is set equal to $1e3$.

The last think we need to analyse is the ground contact event and its occurence. When z, vertical position of the foot, is less than zero the boolean variable *contact* is set to its positive value and the feet immediately start to feel a force described by relations D.2. The feet will be pushed back by an high order-dynamic spring with a very big stiffness and damping coefficients, while, reaching a positive z position, they will be freed by these forces because the contact does not exist anymore. The slowdown of the simulations is due to the fact that, in the moment in which the ground contact event is activated, the foot becomes in contact with a high order-dynamic spring so the solver algorithm requires much more time to solve those equations because much more complex. This explains the slowness of the simulations. Consequently moving to the four-legged cheetah will slow down everything much more because we need to consider the contact between four feet and the ground. The unpredictability is instead explained by the fact that the z position measured that activates the ground contact event changes every step, so our forces will change at every step and the starting point of the algorithm that solves the system will change at every step. This leads to an high unpredictability of the model that makes the simulations much more complex and unreadable that the results obtained on the prototype.

Appendix D

Gait analysis

The gait [9] is the sequence of movements of the limbs of animals, humans and legged-robots necessary to provide locomotion. There exists a huge number of different gaits, in particular for quadrupeds and the animal can choose among them in relation with their speed, the type of the ground and the energy consumption. We are going to analyse in details five different quadrupeds gaits:

1. Bound
2. Walk
3. Trot
4. Transverse gallop
5. Rotary gallop

D.1 The bound

The bound is one of the most easy-patter gaits. The two anterior legs are coupled as the posterior ones and the period is composed by a simultaneous touchdown of the rear limbs, a flight phase, the simultaneous touchdown of the front feet and a final flight phase, see figure D.1.

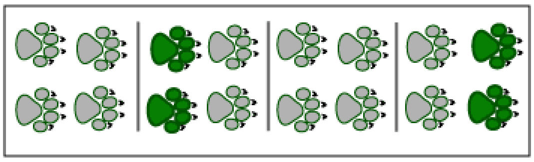

Figure D.1: Bound pattern. Figure reproduced from [9].

D.2 The walk

The walk is the least tiring-gait for every animal in which they alternate a two-limbs ground contact to a three-limbs one as shown in figure D.2. Of course the speed of the animals while they walk is low and limited.

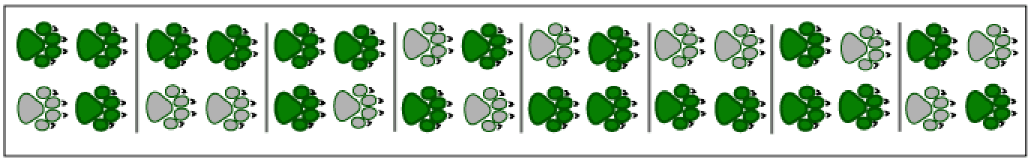

Figure D.2: Walk pattern. Figure reproduced from [9].

The walk has two main variants:

- Power walk: three feet simultaneously touch the ground and the two-limbs ground contact phase is not present anymore.
- Quick walk: only two feet touch the ground simultaneously while the three-limbs ground contact phase is not present anymore.

D.3 The trot

The trot is a two-limbs ground contact gait with a flight phase in the middle in which the feet on the diagonal touch the terrain simultaneously. It is typically use by animal on irregular and rough land or to travel very long distances at a quide good speed. There are a lot of type of trot depepnding on the animal, but they will not be analysed. In figure D.3 is shown the trot pattern.

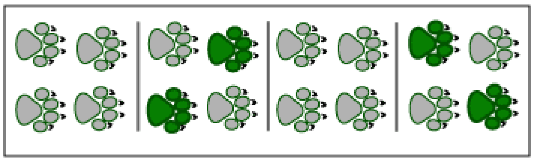

Figure D.3: Trot pattern. Figure reproduced from [9].

D.4 The transverse gallop

The transverse gallop allows reaching high speed, but it is very energy-comsumer. It is a kind of gait in which just one foot touches the ground every time with a flight phase at the end of every period, i.e. the touchdown of the four feet. In figure D.4 is shown this pattern.

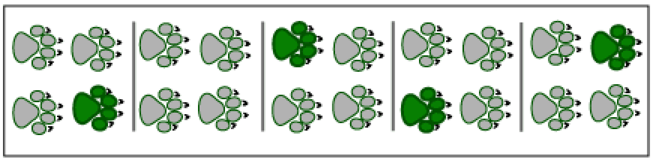

Figure D.4: Transverse gallop pattern. Figure reproduced from [9].

D.5 The rotary gallop

Rotary gallop is the gait that allows reaching the highest speed among the other gaits, but it is also the most energy-consumer. As in the transverse gallop only one foot touches the ground every time, but here we have twice the flight phases, i.e. two flight phases every period. It is the typical gait of cheetahs, figure D.5.

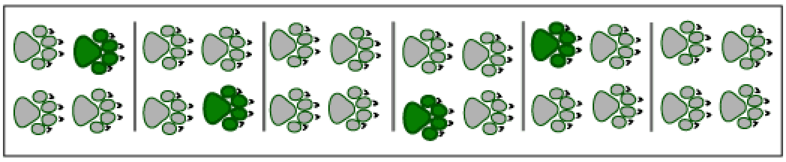

Figure D.5: Rotary gallop pattern. Figure reproduced from [9].

D.6 Energy consumption and energy efficiency

It's well-known that animals and humans are very energy efficient in the field of locomotion [3], that's why legged robot try to mimic their energy efficient features to achieve energy efficiency too. Although there are a lot of tools that the nature uses to achieve energy efficiency we would like to focus on the relation among type of gait, energy required, in term of oxigen consumption and moving speed. An exhaustive plot is shown in figure D.6.

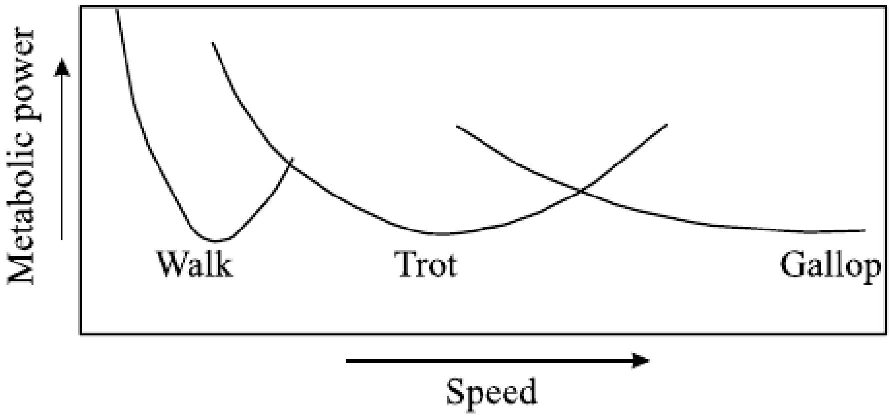

Figure D.6: Relation between energy consumpion of the gait and moving speed Figure reproduced from [3].

From experiments on a horse, some researchers collected data and draw the graph above. It's worth to notice that the velocity increment leads the animal to change its gait. However the most interensting thing is the moment in which its gait changes. It was discovered that the horse, but more in general animals, moves with the type of gait that is the least energy consuming for their travel speed. In the plot you can see exactly this concept, with the speed that keeps increasing, in the moment in which the walk becomes more energy consuming than the trot, the horse starts to trot rather than walk and the same thing happen in the moment in which the trot becomes more energy consuming than the gallop.