ResearchGate

See discussions, stats, and author profiles for this publication at: https://www.researchgate.net/publication/357791580

"CONTROL OF A QUADRUPED LEG USING AN ANALYTICAL APPROACH" Thesis for obtaining the degree of Master in Optomechatronics

Thesis · January 2022

DOI: 10.13140/RG.2.2.29418.24006

CITATIONS
0

READS
254

2 authors, including:

Rodrigo Murillo Aranda
Centro de Investigaciones en Optica
1 PUBLICATION **0** CITATIONS

SEE PROFILE

Some of the authors of this publication are also working on these related projects:

Quadruped Control View project

All content following this page was uploaded by Rodrigo Murillo Aranda on 13 January 2022.

The user has requested enhancement of the downloaded file.

CENTRO DE INVESTIGACIONES
EN OPTICA, A.C.

"CONTROL OF A QUADRUPED LEG USING AN ANALYTICAL APPROACH"

Final Version. Including the corrections marked by the reviewers.

Thesis for obtaining the degree of Master in Optomechatronics

Presents: Eng. Rodrigo Murillo Aranda

Thesis Director: Dr. Gerardo Ramón Flores Colunga

Visto Bueno
10 de diciembre de 2021

Gerardo Flores
Director de tesis

León · Guanajuato · México
December 2021

"Everybody is a genius. But if you judge a fish by its ability to climb a tree, it will live its whole life believing that it is stupid."
-Albert Einstein

Contents

List of Tables

List of Figures

Acknowledgements

I would like to thank God for giving me life in order to fulfill everything that I have achieved in my life so far and for illuminating my mind in order to achieve this work.

There are many people that I would like to thank for the development of this work. The most important of all are my parents Arnoldo and Olga Lilia, that have supported me through my whole life.

I would like to thank my grandmother Ciria for being supportive all these years, and for giving her love and her time to my family since I have memory.

I would also like to thank my siblings Arnoldo, Alberto, Carolina and Gerardo for being my life partners because we have shared many memories and without your support this couldn't be possible.

I would like to thank to my professors, because they shared with me their knowledge, and helped me develop myself, especially to M.O. Diego Torres Armenta for sharing me his knowledge in control, because it helped me with the basics of the development of this project.

I would like to thank my LAPYR colleague Hugo, who introduced me to the subject, guided me within his knowledge of quadruped robots, and helped me with his experience and his friendship in the development of this thesis.

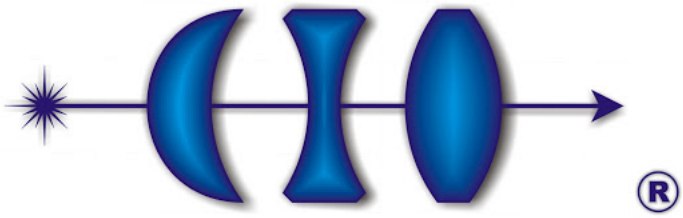
I would also like to thank all of my other LAPYR colleagues, for their knowledge and their friendship, especially Andrés, Alex and Verdín, because they encouraged me through tough times in order to give the best of myself.

I would also like to thank my colleagues from the master degree, for the good and the bad experiences that we shared through the master degree, that helped us get through the adversities that we came across.

I would like to thank specially my advisor Dr. Gerardo Ramón Flores Colunga, because he did everything that was on his hands to introduce me in doing this master degree, and encouraged me with his knowledge and his patience in the development of this thesis.

I would like to thank Dr. Noé Aldana Murillo, because in spite of the pandemic, we had a good work environment, where we encouraged each other with our knowledge and our friendship in the development of our work.

I would like to thank Centro de Investigaciones en Óptica "CIO" for giving me the opportunity to study in their buildings, for supporting me with the knowledge of their professors, and for giving me the opportunity to work in the Laboratory of Automation, Perception and Robotics "LAPYR".



CENTRO DE INVESTIGACIONES
EN OPTICA, A.C.

I would also like to thank to the Congreso Nacional de Ciencia y Tecnología "CONACYT" into giving me their economical support to the CVU 1013522, because they helped me achieve this degree with their scholarship and I was able to dedicate full time into doing this work, providing something to science and society.

Abstract

This thesis presents the control of a simulation of a quadruped robot that mimics the locomotion of a cheetah. This robot has the capability of mimicking many gait patterns of real quadrupeds such as trot, pronk, gallop, and pace. In order to mimic the gaits of the quadruped, a robot was used for the software in the loop (SIL) visualization, this robot was the Laikago[19] robot from the Chinese company Unitree Robotics which has proved to demonstrate an outstanding performance. This robot is capable of responding to external disturbances using the control technique of Model Predictive Control (MPC). The majority of quadruped robots have used the Model Predictive Control technique for their control scheme because of their feasibility in their performance and their capability to respond to external disturbances.

In the state of the art, in simulation and in real quadrupeds, quadruped robots use two controllers for the locomotion of their legs, one for the stance phase and another one for the swing phase. For the stance phase, they have used MPC, and its variations in order to test the locomotion of quadrupeds in simulation as well as with the real quadrupeds. On the other hand for the swing phase, there has been used a PD controller with a feedforward term. In order to test something different, a quadruped leg was designed, as well as a new proposed controller for the locomotion of its swing phase. The objective of this work is to test its performance in simulation using the Simulink software. The main contribution of this controller is that in literature it has been tested a controller with another approach, while it will be proposed another controller with a feedforward term as well, using a different approach. This controller was tested in simulation in order to prove its behavior and its performance. With these tests it will be possible to contribute to the controller of a quadruped robot using a different approach, leading to an analytical solution. In the future, the proposed controller can be tested firstly in a real robot leg, and after testing its performance, in a whole quadruped using a platform with the SIL technique, and later, after having a good functionality in simulation, test the controllers in a real quadruped.

List of Acronyms

- COM-Center of Mass
- DOF-Degrees of Freedom
- MPC-Model Predictive Control
- SIL-Software in the Loop
- UAV-Unmanned Aerial Vehicle
- FR-Front Right
- FL-Front Left
- BR-Back Right
- BL-Back Left
- DH-Denavit Hartenberg
- OSIM-Operational Space Inertia Matrix
- rpy-roll pitch yaw
- w.r.t.-with respect to

Chapter 1: Introduction

Robots have been used by mankind in order to help them to solve problems that are very hard to solve by a single person. As the needs and technology advanced, a new kind of system appeared, which corresponded to systems rigidly connected to the ground such as a robotic arm; this is the conventional kind of robot used in the industrial field. As technology progressed, a new class of systems surfaced in the form of floating-based systems. The classic example of floating base systems is the Unmanned Aerial Vehicle (UAV) robots, which were popularized with the usage of drones. With investigation and tests, it was feasible to build another kind of floating base system in the form of quadruped robots. The objective of making these kinds of robots is to help to aid the humans in tasks that are not as feasible to do for the humans themselves, but there are some challenges that each kind of robot faces in order to achieve its goal. For example, wheeled robots have difficulty traversing uneven terrain or climbing stairs, whilst legged robots have difficulty adapting to uneven terrains, so it is important to keep improving their structure and their control to have a good performance.

For usage of quadruped robots, the user gives them a commanded trajectory and in order to maintain this given trajectory, these systems use control techniques in order to reach stability, with the objective that the robot follows the commanded trajectory in spite of the disturbances. Many enterprises have developed legged robots in order to fulfill the necessities that emerge from the environment. Each enterprise has developed a robot with a control scheme for its robot. For example, one of the applications of quadruped robots in the last years is that due to the pandemic caused by the SARS CoV-2, a legged robot with a surveillance camera was developed by the company Boston Dynamics[7] in order to measure the distance between two persons with a sensor, in order to keep them apart enough from each other with the goal to help the diminishing of the spread of the virus. In the following subsection, it is going to be described some of the robots that these companies have developed.

1.1 State of art

For the last two decades, enterprises have made quadruped robots with different energy supplies and different actuation systems. These enterprises have improved the efficiency of both the energy and the actuation systems, so it will be mentioned the first quadruped robots that were in the form of hydraulic robots, and progress through the most recent robots that are in the form of electric robots.

1.1.1 Hydraulic robots

One of the first hydraulic actuated quadruped robot was the Boston Dynamics Big Dog[15] shown in Figure 1.1, which used the Inverted Pendulum[2] approach in its control. This robot had 4 Degrees of Freedom (DOF) in each leg, and used a gasoline system as its energy source. Because it required a lot of maintenance and had a heavy motor, it was discontinued in 2015.

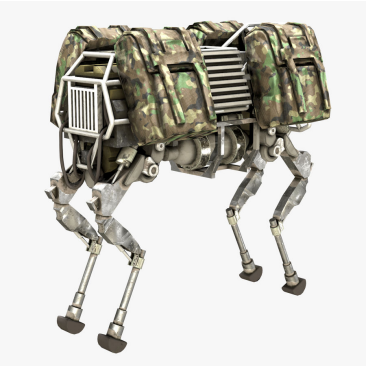


Figure 1.1: Big Dog robot.

Other hydraulic actuated robot was developed in Europe from the Italian Institute of Technology. It proposed a feedforward-feedback strategy that uses the inverse dynamics approach in the robot HyQ[17] shown in Figure 1.2. This robot has 3 revolute joints in each leg, finishing with a prismatic joint.

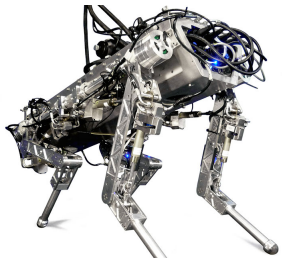


Figure 1.2: HyQ robot.

With further advances in technology and robotics, it emerged the brushless motors and the battery supply for the robots which gave them a better capability to move and to have more efficiency in its energy. With the energy improvements, many companies began developing more robots with the use of more complex control techniques in the form of electric robots.

1.1.2 Electric robots

One of the electric robots was developed in the MIT, with the name of MIT Cheetah[10] shown in Figure 1.3 which uses the hierarchical locomotion. This robot has three motors in each leg, giving it three degrees of freedom on each leg and a large variety of movements. Due to the low-inertia of the robot, it is capable of making fast, dynamic maneuvers, even being capable of making a complete backflip.

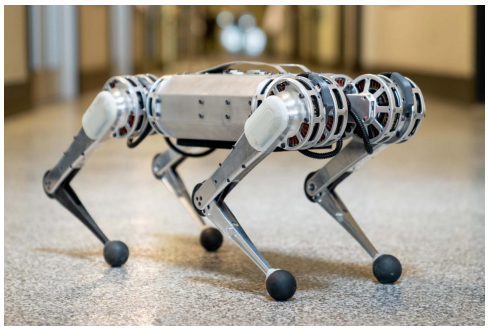


Figure 1.3: MIT cheetah.

Other electric robot was developed in the ETH, with the name of ANYMal[8] shown in Figure 1.4. This robot also uses a hierarchical quadruped locomotion. It has torque-controllable actuators and it is capable of climbing difficult terrains and stairs.

Figure 1.4: ANYMal.

More recently, Boston Dynamics added innovation in its electric actuated robot, incorporating in its robot Spot[7] shown in Figure 1.5, a robotic arm on the top of the robot with five DOF capable of grabbing and manipulating objects. This arm is capable of doing complex movements being capable of manipulating a doorknob to open the door, being one of the most advanced floating base robots of modern time. This robot is capable of even doing autonomous navigation for surveillance purposes and has recently been put on sale for commercial purposes.

It has recently appeared an electric actuated robot from the Chinese company Unitree Robotics with the name of Laikago as seen in Figure 1.6, this robot has 3 DOF en each leg,

Figure 1.5: Spot.

and has the ability to reach up to 0.8 m/s with a battery that could last up to 4 hours. This robot uses for its locomotion the motion imitation from the movements of real animals, it stores this motion in a database called motion data. This data is the reference motion of the robot, which will be reproduced by a robot in a simulation doing the motion imitation. And finally, it can do Reinforcement Learning, in which the real robot imitates the locomotion of the simulation.

Figure 1.6: Laikago.

There are many control techniques that are used in robots, but a predominant technique that has been used in quadruped robots is the Model Predictive Control, which has been a success in both the simulations and in the real robots, so due to its importance in quadruped robots, in the following subsection it is going to be deepened.

1.2 Model Predictive Control

Model Predictive Control[14] (MPC) uses an optimizer, which has the task to compute the lowest quantity of the inputs that are able to achieve the desired output. In the case of the locomotion of quadruped robots, the optimized variables are the contact forces, corresponding to the inputs of the system, that will affect directly in the state of the outputs of the system.

In figure 1.7 is shown a simplified functionality of MPC using one input and one output, which will help the understanding of MPC.

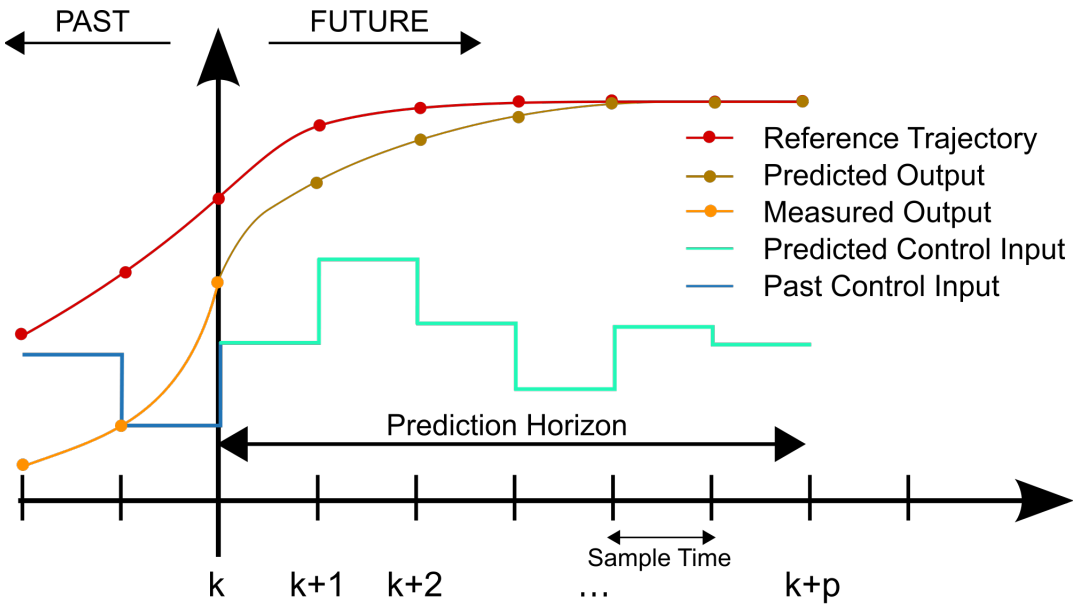


Figure 1.7: Basic functionality of Model Predictive Control.

Even though there are complex dynamics in the legged robots, with the usage of predictive optimization techniques, they can handle the kinematic and dynamic limitations of the robot such as torque limits, self collisions, and contact friction through constraints and cost functions. In simulation many software is being developed to test the locomotion of a given robot, using the software in the loop (SIL) technique, where it integrates the compiled production source code into a model simulation. One of the SIL [4] environments that are being used in robotics is the platform Pybullet. This platform integrates the compiled production of source code into a mathematical model simulation, it eases a virtual simulation environment for developing and testing control strategies of complex systems.

With the usage of the Pybullet platform, it is possible to mimic the locomotion of a real quadruped robot using the MPC technique. One robot that can be used in SIL is the Laikago from Unitree Robotics, described in the previous subsection. In the SIL simulation,

the user can manipulate the linear velocity and the angular velocity of the robot´s COM in previously established time intervals.

In figure 1.8 there can be visualized an image of a simulation of the Laikago Robot using the Pybullet software. It is worth mentioning that many quadruped robots can be used in the platform, but for this thesis purposes, the Laikago from Unitree Robotics was the main used quadruped.

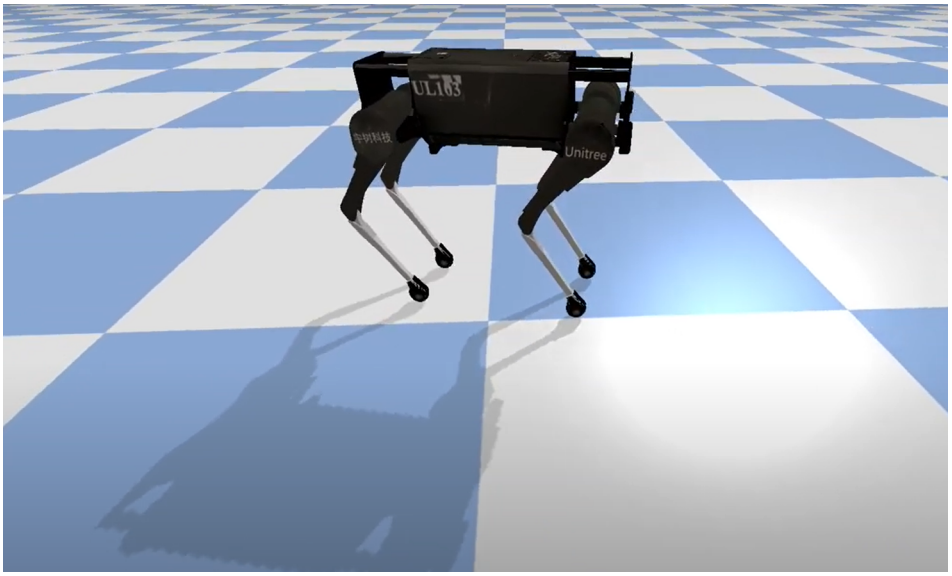


Figure 1.8: Pybullet Software in the loop using the Laikago quadruped from Unitree Robotics.

1.3 Contribution, objectives and structure of the thesis

The main contribution of this work is to propose a controller that calculates an analytical solution for the swing phase controller of the leg of a quadruped robot, differing from the one that has been used in literature.

1.3.1 Main Objective

The main objective of this work is to develop and test a swing phase controller that differs from the ones that are proposed in the literature.

1.3.2 Specific Objectives

- Test the SIL performance that uses the MPC control technique for its development.
- Test different gaits of the quadruped using the SIL.
- Test the newly proposed adaptive PD control with a feedforward torque term for the swing phase of the quadruped.

1.3.3 Structure of the thesis

This thesis is structured with chapter 1 describing the background that quadruped robots have developed in the past, followed by chapter 2 describing the models and the theoretical bases that were involved to work in this thesis. In chapter 3 are described the experiments and simulations that were developed, using the theory previously described in chapter 2.

In chapter 4, the conclusions that were gotten in this thesis are described, in chapter 5 is enlisted the tasks that can be achieved in future work, and finally, an appendix is referred at the end of this thesis where the reader can review the proper references.

Chapter 2: Models and theoretical bases used

2.1 Kinematics model

For the kinematics model, quadruped robots can be modeled as a robot with a solid base with four legs with three degrees of freedom in each leg[1]. For the locomotion of each leg, it has three motors that effectuate the torques in the joints of the legs to provide the locomotion of the whole body. Due to the relatively low mass of the legs, which is roughly 10% of the whole robot, the legs have very low inertia, and the motor could rapidly swing the legs in the air.

The majority of the quadruped robots can be modeled as floating base systems with linked rigid bodies that are attached to the shoulder and the hip joints. The quadruped kinematics can be modeled as a tree structure to have a relation between all the links that conform to the robot. Basically, the whole robot can be modeled as a floating base connected to four legs, visualizing each leg as a Kinematic Chain as it can be seen in figure 2.1.

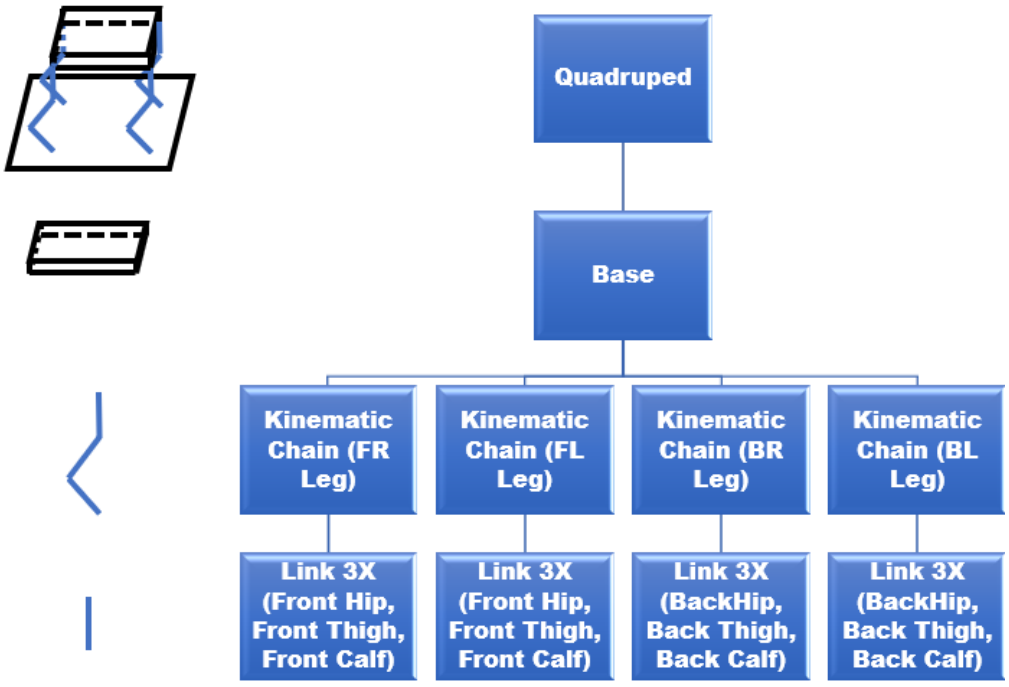


Figure 2.1: The quadruped can be defined as a floating base with a Kinematic Chain representing each leg.

2.2 Frames of the quadruped robot

In order to have a work space on which the robot is referred, there are necessary to have reference frames of the robot; these frames are the geometric points whose position is identified mathematically and physically.

There are two frames considered for the point of view of the robot, the world frame, which has a reference outside of the body of the robot, and the body frame which its reference is in the COM of the robot as shown in figure 2.2. Its important to mention that in simulation and in the real robot, the user visualizes and manipulates the robot from the world frame, while the body frame is often to refer to the robot itself.

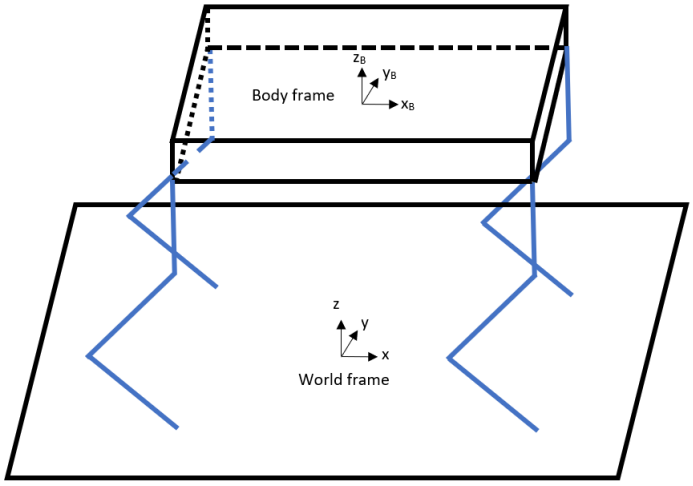


Figure 2.2: Frames of the quadruped robot.

2.2.1 Single Leg Frames

A robotic leg can be viewed as a Kinematic Chain, to analyze it, it is important to have a proper convention of the frames that are involved in the Kinematic Chain. To have a relationship between the joints of a single leg, its important to take into account the parts of which a single leg is composed. For a quadruped, each leg is composed basically of three links, which are the hip, the thigh and the calf, as well as three joints corresponding to the horizontal hip joint, the vertical hip joint and the knee joint. In figure 2.3 is shown the main parts of a quadruped leg.

For a robotic leg, the end effector is located at the bottom of the leg, corresponding to the foot, its important to mention this, because the horizontal hip joint should be identified

as the start of the kinemakic chain and not the foot, as it is often done.

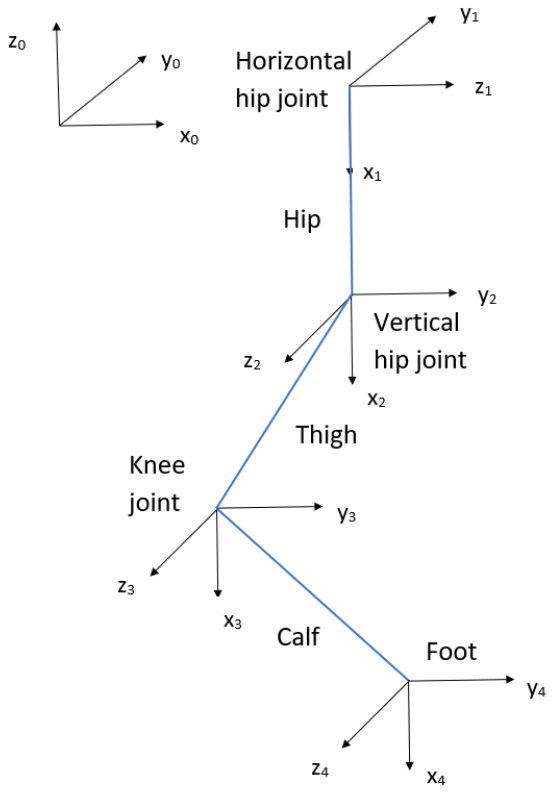


Figure 2.3: Frames of the leg of the robot.

For each leg there are 3 DOF, 1 joint makes the abduction and adduction movements, such as in figure 2.4. These movements are made by the horizontal hip joint.

The vertical hip joint makes the flexion and extension movements of the thigh, while the knee joint makes the flexion and extension movements of the calf. These movements can be viewed in figure 2.5.

Between each frame of the system, there can be two movements between the frames which are rotation movements and translation movements. To have a mathematical relationship between these frames there is a tool called the transformation matrix. To have a transformation matrix that involves all the rotations and translations in every frame of the robot it is important to have a convention that involves the rotations as well as the translations in every frame.

The rotation around the x axis is determined by the α angle, the rotation around the y axis is determined by the ϕ angle and the rotation around the z axis is determined by the θ angle.

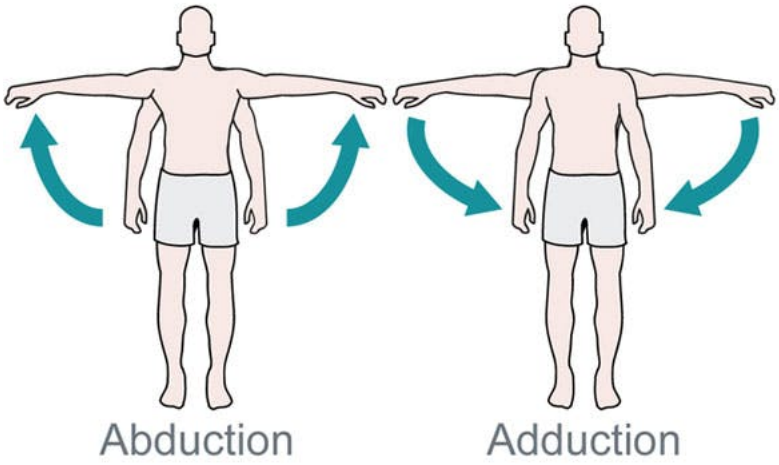


Figure 2.4: Abduction and adduction movements. This image was taken from [13].

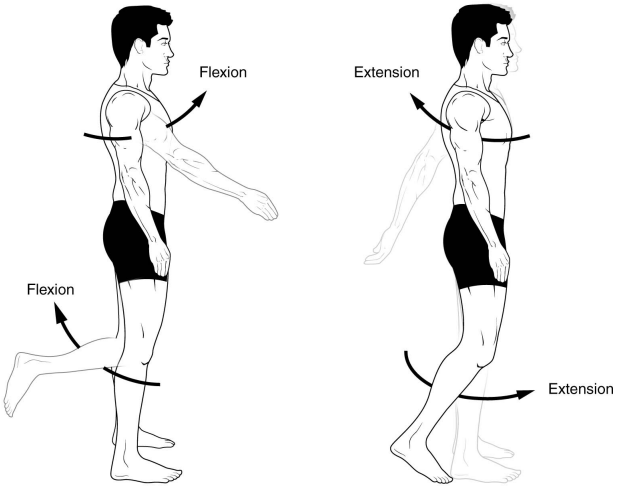


Figure 2.5: Flexion and extension movements. This image was taken from [9].

The rotation matrix of the robot for the α angle is with equation 2.1 as:

$$R_x(\alpha) = \begin{pmatrix} 1 & 0 & 0 \\ 0 & cos(\alpha) & -sin(\alpha) \\ 0 & sin(\alpha) & cos(\alpha) \end{pmatrix} \tag{2.1}$$

The rotation matrix of the robot for the ϕ angle is with equation 2.2 as:

$$R_y(\phi) = \begin{pmatrix} cos(\phi) & 0 & sin(\phi) \\ 0 & 1 & 0 \\ -sin(\phi) & 0 & cos(\phi) \end{pmatrix} \tag{2.2}$$

And finally the rotation matrix for the θ angle is with equation 2.3 as:

$$R_z(\theta) = \begin{pmatrix} cos(\theta) & -sin(\theta) & 0 \\ sin(\theta) & cos(\theta) & 0 \\ 0 & 0 & 1 \end{pmatrix} \tag{2.3}$$

For the translation matrices, they are represented with the letter a for the translation across the x axis, with the letter c for the translation across the y axis, and the letter d for the translation across the z axis.

The translation matrix for the a distance is defined with equation 2.4 as:

$$P_x = \begin{pmatrix} a \\ 0 \\ 0 \end{pmatrix} \tag{2.4}$$

The translation matrix for the c distance is defined with equation 2.5 as:

$$P_y = \begin{pmatrix} 0 \\ c \\ 0 \end{pmatrix} \tag{2.5}$$

The translation matrix for the d distance is with equation 2.6 as:

$$P_z = \begin{pmatrix} 0 \\ 0 \\ d \end{pmatrix} \tag{2.6}$$

With the rotation and the translation matrices, it is possible to build a homogeneous transformation matrix along each axis, yielding to the transformation matrices for each axis.

The transformation matrix along the x axis, incorporating the θ angle and the d distance

is defined by equation 2.7:

$$T_x(\alpha) = \begin{bmatrix} 1 & 0 & 0 & a \\ 0 & cos(\alpha) & -sin(\alpha) & 0 \\ 0 & sin(\alpha) & cos(\alpha) & 0 \\ 0 & 0 & 0 & 1 \end{bmatrix} \tag{2.7}$$

The transformation matrix along the y axis, incorporating the ψ angle and the c distance is defined by equation 2.8:

$$T_y(\phi) = \begin{bmatrix} cos(\phi) & 0 & sin(\phi) & 0 \\ 0 & 1 & 0 & c \\ -sin(\phi) & 0 & cos(\phi) & 0 \\ 0 & 0 & 0 & 1 \end{bmatrix} \tag{2.8}$$

The transformation matrix along the z axis, incorporating the α angle and the a distance is defined by equation 2.9:

$$T_z(\theta) = \begin{bmatrix} cos(\theta) & -sin(\theta) & 0 & 0 \\ sin(\theta) & cos(\theta) & 0 & 0 \\ 0 & 0 & 1 & d \\ 0 & 0 & 0 & 1 \end{bmatrix} \tag{2.9}$$

With these three transformations, a transformation matrix involving the three rotations and the three translations involved within every frame of the system can be computed with equation 2.10.

$$T_{i-1}^{i} = T_z(\alpha)T_y(\phi)T_x(\theta) = \begin{bmatrix} T_{11} & T_{12} & T_{13} & T_{14} \\ T_{21} & T_{22} & T_{23} & T_{24} \\ T_{31} & T_{32} & T_{33} & T_{34} \\ T_{41} & T_{42} & T_{43} & T_{44} \end{bmatrix} \tag{2.10}$$

In Appendix A it can be reviewed the developed transformation matrix of the multiplication of the matrices involving the three angles.

To have a proper relationship between the frames that are involved in the Kinematic Chain, it is important to have a convention that integrates the frames that are involved in the Kinematic Chain. The rules that involve the relationship between each frame are the following:

- Establish a right-handed orthonormal coordinate frame O_0 at the base of the system with Z_0 lying along joint 1 motion axis.
- The Z_i axis is directed along the axis of motion of joint $(i+1)$, which means that link $(i+1)$ rotates about or translates along Z_i.
- Locate the origin of the *ith* coordinate at the intersection of Z_i and Z_{i-1} or the intersection of common normal between the Z_i and the Z_{i-1} axis.
- The X_i axis lies along the common normal from the Z_{i-1} axis to the Z_i axis.
- Assign the Y_i axis to complete the right-handed coordinate system.

In the following figure the frames of a Kinematic Chain can be referred to such as in figure 2.6:

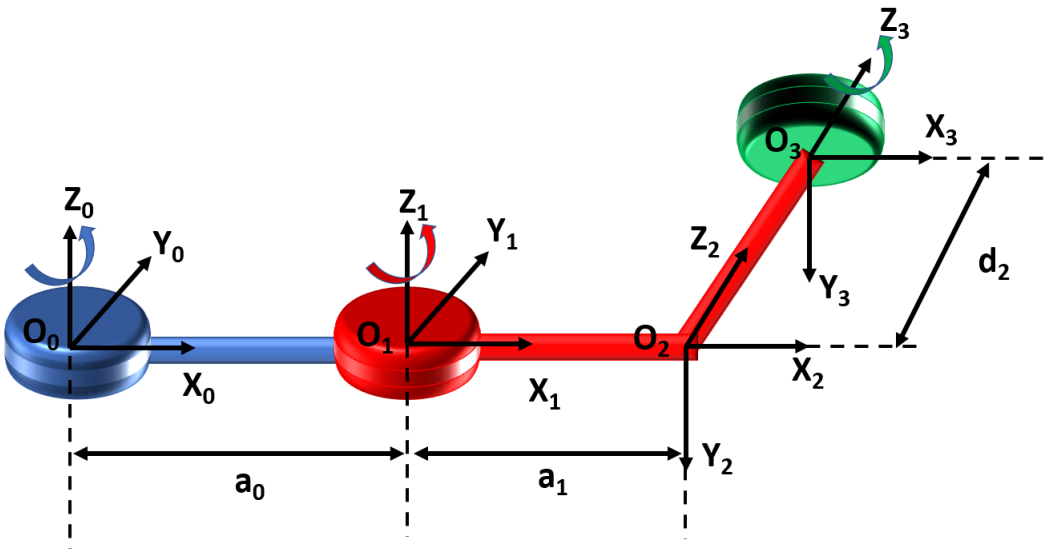


Figure 2.6: Frames of a Kinematic Chain.

After defining the link coordinate frames, it is important to have a relationship of the link and joint parameters that are used in the Denavit Hartenberg (DH) convention. These parameters are the following:

- Joint angle θ_i: This is the angle of rotation from the X_{i-1} axis to the X_i axis about the Z_{i-1} axis.
- Joint distance d_i: This is the distance from the origin of the $(i-1)$ coordinate system to the intersection of the Z_{i-1} axis and the X_i axis along the Z_{i-1} axis.

- Link length a_i: This is the distance from the intersection of the Z_{i-1} axis and the X_i axis to the origin of the ith coordinate system along the X_i axis.
- Link twist angle α_i: This is the angle of rotation from the Z_{i-1} axis to the Z_i axis about the X_i axis.

For the figure that was used, the DH parameters that represent the Kinematic Chain of figure 2.6 are the following:

Denavit Hartenberg Parameters of example				
Joint	$\boldsymbol{\theta_i}$	$\boldsymbol{d_i}$	$\boldsymbol{\alpha_i}$	$\boldsymbol{a_i}$
A_0^1	θ_0	0	0°	a_0
A_1^2	θ_1	0	-90°	a_1
A_2^3	θ_2	d_2	0	0

Table 2.1: DH parameters that involve the frames of the example of the Kinematic Chain of figure 2.6.

With the usage of the DH parameters and a proper relationship between the frames of the robotic leg it is possible to relation the end effector and frame 0, which corresponds to the frame that is attached to the main body. Beginning with the frame of A_0^1, equation 2.8 is applied with a ϕ angle of 90° and a c distance of 0. It´s important this rotation matrix, because normally the end effector of a Kinematic Chain is on the top of its base, while working on a robot leg, its end effector is at the bottom of its base. For the rest of the frames, table 2.2 was used to work applying the DH convention in order to have the linkage coordinate system[16] where the leg of the robot is developed.

Denavit Hartenberg Parameters of Quadruped Leg				
Joint	$\boldsymbol{\theta_i}$	$\boldsymbol{d_i}$	$\boldsymbol{\alpha_i}$	$\boldsymbol{a_i}$
A_1^2	θ_1	0	90°	L_1
A_2^3	θ_2	0	0°	L_2
A_3^4	θ_3	0	0°	L_3

Table 2.2: DH parameters that involve the frames of the leg of the quadruped robot.

Its important to remark that the frames of the robot, are referenced to the frame 0 of the robot, which corresponds to the horizontal hip joint, as it can be seen in figure 2.7:

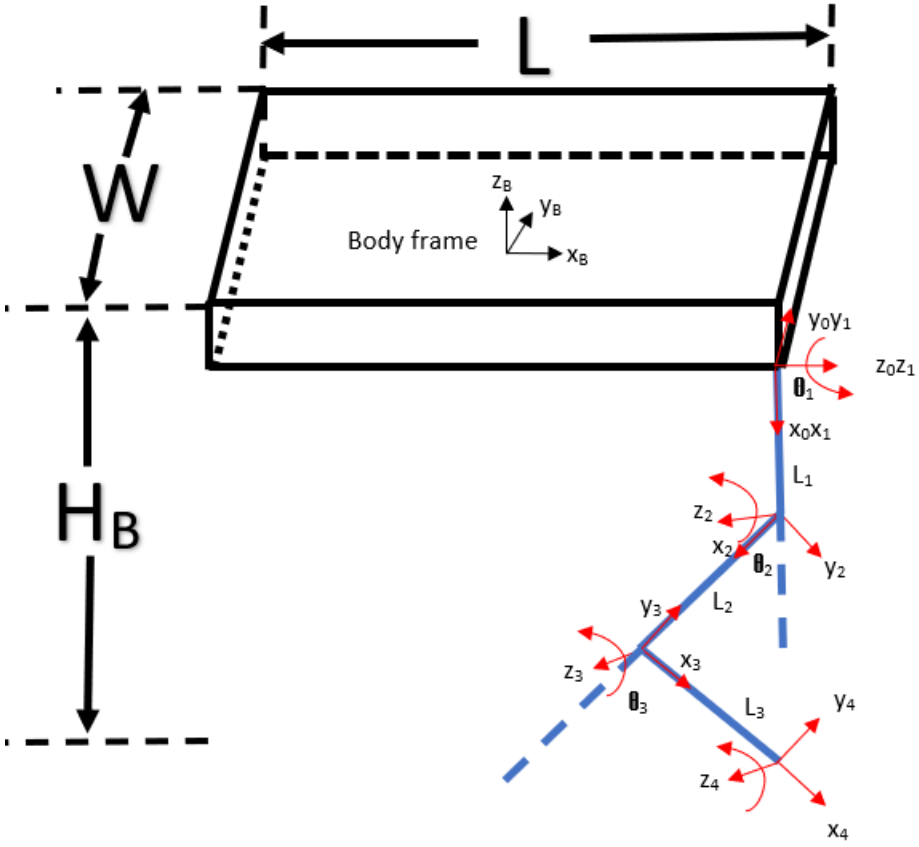


Figure 2.7: Frames of a single leg.

Using equation 2.10, it is possible to calculate the transformation matrices for each frame of the leg of the robot, using the D-H parameters of Table 2.2. Beginning with the first transformation matrix, there is a rotation of 90° along the y axis, yielding to equation 2.11.

$$A_0^1 = \begin{bmatrix} 0 & 0 & 1 & 0 \\ 0 & 1 & 0 & 0 \\ -1 & 0 & 0 & 0 \\ 0 & 0 & 0 & 1 \end{bmatrix} \tag{2.11}$$

For the second transformation matrix, there is a rotation of 90° along the x axis, a translation of the length L_1 along the x axis, corresponding to the length of the hip, and a

rotation of θ_1 along the z axis, which yields to equation 2.12.

$$A_1^2 = \begin{bmatrix} cos(\theta_1) & 0 & sin(\theta_1) & L_1 cos(\theta_1) \\ sin(\theta_1) & 1 & -cos(\theta_1) & L_1 sin(\theta_1) \\ 0 & 1 & 0 & 0 \\ 0 & 0 & 0 & 1 \end{bmatrix} \tag{2.12}$$

For the third transformation matrix, there is a rotation along the z axis and a translation of the length L_2 along the x axis, corresponding to the length of the thigh, which yields to equation 2.13.

$$A_2^3 = \begin{bmatrix} cos(\theta_2) & -sin(\theta_2) & 0 & L_2 cos(\theta_2) \\ sin(\theta_2) & cos(\theta_2) & 0 & L_2 sin(\theta_2) \\ 0 & 0 & 1 & 0 \\ 0 & 0 & 0 & 1 \end{bmatrix} \tag{2.13}$$

For the fourth transformation matrix, there is a rotation along the z axis and a translation of the length L_3 along the x axis, corresponding to the length of the calf, which yields to equation 2.14.

$$A_3^4 = \begin{bmatrix} cos(\theta_3) & -sin(\theta_3) & 0 & L_3 cos(\theta_3) \\ sin(\theta_3) & cos(\theta_3) & 0 & L_3 sin(\theta_3) \\ 0 & 0 & 1 & 0 \\ 0 & 0 & 0 & 1 \end{bmatrix} \tag{2.14}$$

With these equations, a matrix composition can be obtained, multiplying the 4 transformation matrices from the foot frame to the base frame with equation 2.15.

$$T_0^4 = A_0^1 A_1^2 A_2^3 A_3^4 \tag{2.15}$$

After doing the multiplication of these matrices, a rotation matrix, as well as a position matrix involving the end effector with respect to frame (w.r.t.) 0 is obtained like equation 2.16.

$$T_0^4 = \begin{bmatrix} R & P \\ 0 & 1 \end{bmatrix} \tag{2.16}$$

In equation 2.16, the resulting rotation matrix R is a R^{3X3} matrix, while the position matrix P is a R^{3X1} matrix.

Where:

$$\mathrm{R}=\begin{bmatrix} sin(\theta_2+\theta_3) & cos(\theta_2+\theta_3) & 0 \\ cos(\theta_2+\theta_3)sin(\theta_1) & -sin(\theta_2+\theta_3)sin(\theta_1) & -cos(\theta_1) \\ -cos(\theta_2+\theta_3)cos(\theta_1) & sin(\theta_2+\theta_3)cos(\theta_1) & -sin(\theta_1) \end{bmatrix}$$

$$\mathrm{P}=\begin{bmatrix} P_x \\ P_y \\ P_z \end{bmatrix}=\begin{bmatrix} L_3sin(\theta_2+\theta_3)+L_2sin(\theta_2) \\ sin(\theta_1)(L_1+L_3cos(\theta_2+\theta_3)+L_2cos(\theta_2)) \\ -cos(\theta_1)(L_1+L_3cos(\theta_2+\theta_3)+L_2cos(\theta_2)) \end{bmatrix}$$

Obtaining P matrix, the Forward Kinematics of the system is obtained with a relationship of the end effector w.r.t. frame 0, these matrices will be used in the following section to calculate the dynamic model of the leg. Contrary to the Forward Kinematics, an Inverse Kinematics analysis could be applied in which the main objective is to have a reference of the trajectory of the end effector where Cartesian references are used to have resulting angles, after using Inverse Kinematics. The necessary parameters that are needed to calculate the Inverse Kinematics are represented in table 2.3, these parameters can be visualized also in figure 2.7.

Inverse kinematics parameters	
Parameter	**Meaning**
P_x	Position x_4 of the end-effector w.r.t. frame 0.
P_y	Position y_4 of the end-effector w.r.t. frame 0.
P_z	Position z_4 of the end-effector w.r.t. frame 0.
L_1	Length of the hip.
L_2	Length of the thigh.
L_3	Length of the calf.
L	Length of the body.
W	Width of the body.
H_B	Height of the body.

Table 2.3: Parameters used to calculate the Inverse Kinematics.

With the usage of Inverse Kinematics algorithm, the leg can compute the angles to trace the reference trajectory with its foot. After doing some algebraic operations which are out of the purpose of this work, but can get a reference in [3], the joint angles which result for

this leg are represented by $\Theta = [\theta_1 \ \theta_2 \ \theta_3]^T$ with equation 2.17:

$$\Theta = IK({}^{B}PFOOT) = FK(q) = \begin{bmatrix} tan^{-1}\left(\frac{P_y+\frac{W}{2}}{P_z+\frac{H_B}{2}}\right) \\ -\xi + tan^{-1}\frac{P_x-\frac{L}{2}}{\sqrt{\left(P_y+\frac{W}{2}\right)^2+\left(P_z+\frac{H_B}{2}\right)^2-L_1}} \\ \frac{L_2+L_3}{L_3}\xi \end{bmatrix} \tag{2.17}$$

Where:

$$\xi = cos^{-1}\frac{L_2^2+\left(P_x-\frac{L}{2}\right)^2+\left(P_y+\frac{W}{2}-L_1 sin\theta_1\right)^2+\left(P_z+\frac{H_B}{2}+L_1 sin\theta_1\right)^2-L_3^2}{2L_2\sqrt{\left(P_x-\frac{L}{2}\right)^2+\left(P_y+\frac{W}{2}-L_1 sin\theta_1\right)^2+\left(P_z+\frac{H_B}{2}+L_1 sin\theta_1\right)^2}}$$

After having expressions that have used the Forward Kinematics and the Inverse Kinematics is important to have an expression that links the position of the end effector with each angle in which the leg rotates. This expression is obtained by partially deriving the position of the end effector w.r.t. each of the angles. This expression is called the contact Jacobian and is calculated with equation 2.18:

$$J_c = \begin{bmatrix} \frac{\partial x}{\partial \theta_1} & \frac{\partial x}{\partial \theta_2} & \frac{\partial x}{\partial \theta_3} \\ \frac{\partial y}{\partial \theta_1} & \frac{\partial y}{\partial \theta_2} & \frac{\partial y}{\partial \theta_3} \\ \frac{\partial z}{\partial \theta_1} & \frac{\partial z}{\partial \theta_2} & \frac{\partial z}{\partial \theta_3} \end{bmatrix} \tag{2.18}$$

After doing the partial derivatives of the Θ angles that are obtained from equation 2.17, it yields to equation 2.19:

$$J_c = \begin{bmatrix} 0 & L_{2c2} + L_3 c_{23} & L_3 c_{23} \\ L_1 c_1 + L_2 c_1 c_2 + L_3 c_1 c_{23} & -L_2 s_1 s_2 - L_3 s_1 s_{23} & -L_3 s_1 s_{23} \\ L_1 s_1 + L_2 s_1 c_2 + L_3 s_1 c_{23} & L_2 c_1 s_2 - L_3 c_1 s_{23} & L_3 c_1 s_{23} \end{bmatrix} \tag{2.19}$$

In equation 2.19 s_i, c_i stand for $sin\theta_i$ and $cos\theta_i$ respectively, while s_{ij}, c_{ij} stand for $sin(\theta_i + \theta_j)$ and $cos(\theta_i + \theta_j)$ respectively. The contact Jacobian is a matrix that was used in the experiments in order to test the controllers, so it is important to have a proper calculation of this matrix in order to have a proper behavior of the controllers.

2.3 Phases of the legs

To have proper control of the locomotion of the legs, it is important to consider the phases [20] of the legs that are involved during the locomotion of the quadruped. These phases are the swing phase and stance phase, both phases need to have proper coordination to have proper locomotion of the whole body.

The most important difference between the swing phase and the stance phase is that during the swing phase, it doesn't make contact with the ground, meaning that there is no contact force during this phase of the locomotion, while during the stance phase there is a contact force. It is very important to consider this difference because there needs to be applied a different controller in each phase to have proper control of each phase, meaning a control for the whole phases of locomotion. In figure 2.8 it can be seen the main phases of locomotion.

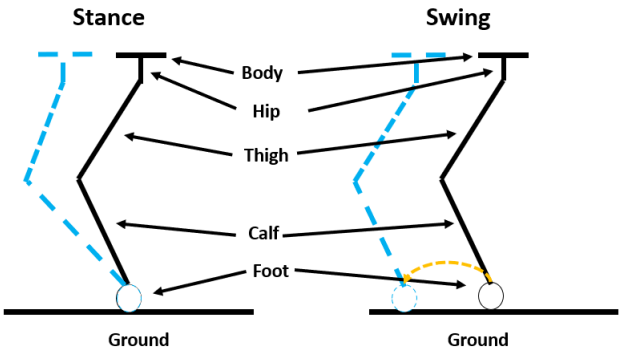


Figure 2.8: Locomotion of a robot leg during stance phase and swing phase. The blue dashed line represents the finish position at the end of swing phase and stance phase respectively. The yellow dashed line represents the trajectory of a foot during swing phase.

As it can be seen in figure 2.8, during the Stance Phase, the foot doesn't move from its position at the beginning of the Stance Phase, but the main body moves. On the other hand during Swing Phase, both the foot and the body move from their original position at the beginning of the Swing Phase.

2.4 Force Distribution

So that the whole quadruped has stability in its position, it is important to have a correct force distribution in each leg that is involved in making contact with the ground. When the

leg is in the stance phase it computes the correct amount of the contact force, to reach stability for the whole quadruped. The force on each foot has 3 components[21] which are represented in equations 2.20, 2.21 and 2.22 considering only the feet that are making contact with the ground as n_c at a specific time:

$$\sum F_x = \sum_{i=1}^{n_c} Fx_i \tag{2.20}$$

$$\sum F_y = \sum_{i=1}^{n_c} Fy_i \tag{2.21}$$

$$\sum F_z = \sum_{i=1}^{n_c} Fz_i \tag{2.22}$$

Where:

$f_i = [Fx_i \;\; Fy_i \;\; Fz_i]^T$

With this relation, when the robot is making contact with its four legs, four forces are obtained, which can be visualized in figure 2.9:

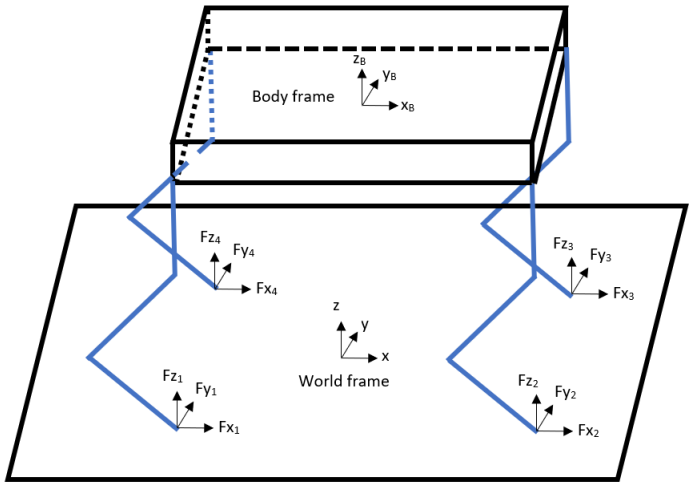


Figure 2.9: Force Distribution of 4 contact forces along the quadruped robot.

It´s very important to consider these contact forces because these forces affect the locomotion of the whole robot, which will be discussed deeper in the following section, in which f_i has a very important role.

2.5 Leg Dynamics Model in Swing Phase

During the swing phase, there needs to be a dynamic model to compute the joint torques. Accounting that there are no contact forces during the swing phase, the dynamics model can be described by the model of equation 2.23:

$$M(q)\ddot{q} + C(q,\dot{q})\dot{q} + G(q) = \tau \tag{2.23}$$

Here $M(q) \in R^{3X3}$ represents the torque caused by the inertia tensors of the links of the robot, $C(q,\dot{q}) \in R^{3X3}$ is a matrix that represents the torque due to the Coriolis forces and the centrifugal forces, $G(q) \in R^{3X1}$ is a matrix that represents the torque due to the Gravity force, and finally $\tau \in R^{3X1}$ is a matrix that represents the torques of the joint motors of the robotic leg. Each leg has 3 DOF, in which the main function of the motor from the horizontal hip joint is to make the movements of abduction and adduction of the leg, the angle of these movements is represented with θ_1, while the motors of the vertical hip joint and the knee joint make the function of flexion and extension of the thigh and the calf respectively, these angles are represented with θ_2 and θ_3 respectively. The three angles in which the leg makes its movements can be represented as a vector-like in equation 2.24:

$$q = \begin{bmatrix} \theta_1 \\ \theta_2 \\ \theta_3 \end{bmatrix} \tag{2.24}$$

With their respective velocities as in equation 2.25:

$$\dot{q} = \begin{bmatrix} \dot{\theta_1} \\ \dot{\theta_2} \\ \dot{\theta_3} \end{bmatrix} \tag{2.25}$$

To compute the matrices of the dynamic model, the first step is to calculate the position COM of the 3 links w.r.t. frame 0, which corresponds to the horizontal hip joint. The COM of the links can be obtained with the aid of the previously computed transformation matrices. For the hip link, the COM of the hip link can be obtained by multiplying the transformation matrices $A_0^1 A_1^2$ and taking the first, second, and third elements of the fourth column, after dividing them by two it yields to the position of the COM of the hip link w.r.t.

frame 0, like in equation 2.26:

$$r1c = \begin{bmatrix} 0 \\ \frac{L_1 \sin(\theta_1)}{2} \\ -\frac{L_1 \cos(\theta_1)}{2} \end{bmatrix} \tag{2.26}$$

Similarly, the coordinates of the COM of the thigh can be obtained multiplying the transformation matrices $A_0^1 A_1^2 A_2^3$, taking as well the first three rows of the fourth column of the resulting matrix. It's important to mention that for the COM of the thigh, it's considered the full length of the hip plus the terms involving L_2 divided by 2 resulting in the position of the COM of the thigh w.r.t. frame 0, which yields to equation 2.27:

$$r2c = \begin{bmatrix} \frac{L_2 \sin(\theta_2)}{2} \\ L_1 \sin(\theta_1) + \frac{L_2 \cos(\theta_2) \sin(\theta_1)}{2} \\ -L_1 \cos(\theta_1) - \frac{L_2 \cos(\theta_1) \cos(\theta_2)}{2} \end{bmatrix} \tag{2.27}$$

Finally, the coordinates of the COM of the calf, are obtained by multiplying the matrices $A_0^1 A_1^2 A_2^3 A_3^4$, considering the first three rows of the fourth column of the resulting matrix. Also, it´s important to mention that for the COM of the calf, it considered the full length of the hip, and the full length of the thigh plus the terms involving L_3 divided by 2 resulting in the position of the COM of the calf w.r.t. frame 0, which yields to 2.28:

$$r3c = \begin{bmatrix} \frac{L_3 \sin(\theta_2+\theta_3)}{2} + L_2 \sin(\theta_2) \\ \frac{\sin(\theta_1)\,(2\,L_1 + L_3 \cos(\theta_2+\theta_3) + 2\,L_2 \cos(\theta_2))}{2} \\ -\frac{\cos(\theta_1)\,(2\,L_1 + L_3 \cos(\theta_2+\theta_3) + 2\,L_2 \cos(\theta_2))}{2} \end{bmatrix} \tag{2.28}$$

The next part is to calculate the linear velocities of the COM of the links referred to frame 0, which can be calculated by a partial derivative of the position of the COM w.r.t. each of the angles of the vector q which can be seen in equation 2.29:

$$drn = \frac{\partial rc}{\partial q} \tag{2.29}$$

After applying equation 2.29 into equation 2.26, it is possible to represent the velocity of

the COM of the hip as in 2.30:

$$dr1c = \begin{bmatrix} 0 \\ \frac{L_1\,\dot{\theta}_1\,\cos(\theta_1)}{2} \\ \frac{L_1\,\dot{\theta}_1\,\sin(\theta_1)}{2} \end{bmatrix} \tag{2.30}$$

Similarly, equation 2.29 can be applied into equation 2.27, to obtain the velocity of the COM of the thigh as in 2.31:

$$dr2c = \begin{bmatrix} \frac{L_2\,\dot{\theta}_2\,\cos(\theta_2)}{2} \\ \dot{\theta}_1\,\left(L_1\,\cos\left(\theta_1\right) + \frac{L_2\,\cos(\theta_1)\,\cos(\theta_2)}{2}\right) - \frac{L_2\,\dot{\theta}_2\,\sin(\theta_1)\,\sin(\theta_2)}{2} \\ \dot{\theta}_1\,\left(L_1\,\sin\left(\theta_1\right) + \frac{L_2\,\cos(\theta_2)\,\sin(\theta_1)}{2}\right) + \frac{L_2\,\dot{\theta}_2\,\cos(\theta_1)\,\sin(\theta_2)}{2} \end{bmatrix} \tag{2.31}$$

Finally, equation 2.29 can be applied into equation 2.28, to obtain the velocity of the COM of the calf as in 2.32:

$$dr3c = \begin{bmatrix} \dot{\theta}_2\,\left(\frac{L_3\,\cos(\theta_2+\theta_3)}{2} + L_2\,\cos\left(\theta_2\right)\right) + \frac{L_3\,\dot{\theta}_3\,\cos(\theta_2+\theta_3)}{2} \\ \frac{\dot{\theta}_1\,\cos(\theta_1)\,(2\,L_1+L_3\,\cos(\theta_2+\theta_3)+2\,L_2\,\cos(\theta_2))}{2} - \frac{\dot{\theta}_2\,\sin(\theta_1)\,(L_3\,\sin(\theta_2+\theta_3)+2\,L_2\,\sin(\theta_2))}{2} - \frac{L_3\,\dot{\theta}_3\,\sin(\theta_2+\theta_3)\,\sin(\theta_1)}{2} \\ \frac{\dot{\theta}_2\,\cos(\theta_1)\,(L_3\,\sin(\theta_2+\theta_3)+2\,L_2\,\sin(\theta_2))}{2} + \frac{\dot{\theta}_1\,\sin(\theta_1)\,(2\,L_1+L_3\,\cos(\theta_2+\theta_3)+2\,L_2\,\cos(\theta_2))}{2} + \frac{L_3\,\dot{\theta}_3\,\sin(\theta_2+\theta_3)\,\cos(\theta_1)}{2} \end{bmatrix} \tag{2.32}$$

After obtaining the positions and the velocities of the COM of the links of the system, an Euler-Lagrange method was developed to obtain the kinematic and the potential energy of the system. The Lagrangian can be calculated with the sum of the kinematic energy, minus the potential energy of the system as expressed with equation 2.33:

$$L = T - U \tag{2.33}$$

In the case of the kinematic energy, is calculated considering the translational, as well as the rotational contributions for each link such as in equation 2.34:

$$T = \frac{1}{2} m_i dr_n^T dr_n + \frac{1}{2} w_i^T I_\Delta w_i \tag{2.34}$$

It's important to mention that the links of the system are considered as rigid bars with a circular section, considering their inertia tensors I_Δ as in equation 2.35.

$$I_\Delta = \begin{bmatrix} I_{xx} & 0 & 0 \\ 0 & I_{yy} & 0 \\ 0 & 0 & I_{zz} \end{bmatrix} = \begin{bmatrix} \frac{mL^2}{12} & 0 & 0 \\ 0 & \frac{mL^2}{12} & 0 \\ 0 & 0 & \frac{mL^2}{12} \end{bmatrix} \tag{2.35}$$

Each link contributes to the Kinematic Energy for the whole system. With equation 2.34 it is possible to calculate the contribution of the Kinematic energy for each link. After calculating the kinematic energy of the three links it is possible to have the kinematic energy of the whole system with equation 2.36:

$$T = T_1 + T_2 + T_3 \tag{2.36}$$

The developed matrix T can be viewed in Appendix B, with the contribution of the kinematic energy of the three links.

The next part is to calculate the contribution for the Potential Energy of each link, which can be calculated with equation 2.37:

$$U = -m_i g^T r_n \tag{2.37}$$

With the contribution of the three links it is possible to sum the whole potential energy U of the whole leg with equation 2.38:

$$U = U_1 + U_2 + U_3 \tag{2.38}$$

The developed matrix U can be viewed in Appendix C, with the contribution of the potential energy of the three links.

With the whole contribution of the kinematic energy, as well as of the potential energy, the next step is to calculate the matrices of the Dynamic Model that is presented in equation 2.23. Beginning with the M matrix, this matrix contains the inertia tensor terms of the links of the leg, this matrix is calculated doing a derivative of the kinematic energy such as in equation 2.39:

$$M(q) = \frac{\partial}{\partial dq}\left(\frac{\partial T}{\partial dq}\right) \tag{2.39}$$

The developed M matrix involving all the terms of the M matrix can be reviewed in Appendix D.

After calculating the M matrix, the next matrix to be calculated is the C matrix. This matrix takes into account the torques due to the Coriolis and Centrifugal terms, this could be calculated using the Christoffel Symbols of the first kind; to calculate them, it is necessary the M(q) matrix that was recently obtained, the $C(q, \dot{q})$ matrix can be calculated with equation 2.40:

$$C(q, \dot{q}) = \frac{1}{2} \left(\frac{\partial M_{kj}}{\partial q_i} + \frac{\partial M_{ki}}{\partial q_j} - \frac{\partial M_{ij}}{\partial q_k} \right) \tag{2.40}$$

The developed C matrix involving all the terms of the C matrix can be reviewed in Appendix E.

Finally, to calculate the torques due to the gravitational forces, the G matrix can be calculated deriving the total potential energy w.r.t. each angle of the vector q, resulting in equation 2.41:

$$G(q) = \frac{\partial U}{\partial q_i} \tag{2.41}$$

The developed G matrix involving all the terms of the G matrix can be reviewed in Appendix F.

2.6 Models used for the whole body

2.6.1 Lumped Mass Model

After having a dynamic model for each leg of the quadruped during the swing phase, it's important to have a model that involves the whole quadruped. The first step to find this model is to use the second law of Newton to account for the reaction forces of the robot considering the legs that are making contact with the ground. The reaction forces are found with the lumped mass model[6] with equations 2.42 and 2.43:

$$m\ddot{p} = \sum_{i=1}^{n_c} f_i - c_g \tag{2.42}$$

$$\frac{d}{dt}(I_R w) = \sum_{i=1}^{n_c} r_i \times f_i \tag{2.43}$$

Where:

$p \in R^3$: Vector representing the position of the robot w.r.t. the world frame.

$\ddot{p} \in R^3$: Vector representing the acceleration of the robot w.r.t. the world frame.

$f_i \in R^3$: Vector representing the reaction forces of the robot w.r.t. the world frame.

n_c: Is the number of legs that are in contact with the ground.

$c_g \in R^3$: Vector representing the gravitational acceleration w.r.t. the world frame.

m: Is the total mass of the robot.

$I_R \in R^{3\times3}$: Is the rotational inertia tensor.

$w \in R^3$: Is the angular velocity of the body of the robot.

$r_i \in R^3$: Is the position of the i-th contact point w.r.t. the COM of the robot.

It is assumed that the $w \times (Iw)$ is small for bodies with small angular velocities, so it can be neglected. Accounting this, equation 2.43 can be approximated with equation 2.44:

$$\frac{d}{dt}(I_R w) = I_R \dot{w} + w \times (I_R w) = I_R \dot{w} \tag{2.44}$$

With the last assumption the acceleration and the angular velocity of the robot's COM can be represented with the equations

$$\ddot{p} = \frac{\sum_{i=1}^{n_c} f_i - c_g}{m} \tag{2.45}$$

$$\dot{w} = I_R^{-1} \sum_{i=1}^{n_c} r_i \times f_i \tag{2.46}$$

The reaction forces found by each contact leg f_i are then used in the Multi-Body Dynamics Model to compute the torques of the joints of each leg.

2.6.2 Multi-body Dynamics Model

The reaction forces f_i of the contact legs obtained from the lumped mass model are used to calculate torque commands of each leg with the usage of the Multi-Body Dynamics Model. The vector of acceleration obtained from the lumped mass model $\ddot{p}$ matches with the vector $\ddot{q}$ from the multibody dynamics model.

In the case of the movement of each leg, a parabola is used to determine the position of each foot during the swing phase, where the swing leg moves faster and finishes the first half of the full swing trajectory in the first half of the full swing trajectory. By doing this each foot has an acceleration vector q_j where it is considered in the multi-body dynamics model to calculate the torques of the joints.

The multi-body dynamics model [11] is described as equation 2.47:

$$S \begin{bmatrix} \tau_f \\ \tau_j \end{bmatrix} = A\ddot{q} + b + g - J_c^T f_i \tag{2.47}$$

Where:

$S = \begin{bmatrix} 0_{n_j x n_f} & I_{n_j} \end{bmatrix}$: This selection matrix separates the n_j joint coordinates from the unactuated floating base coordinates n_f.

$q = \begin{bmatrix} q_j \\ q_f \end{bmatrix}$.

$q_f = (x, y, z, \phi, \theta, \psi)^T$.

$q_f \in R^6$: Unactuated floating base coordinates.

$q_j :\in R^{n_j}$: Actuated joint coordinates.

n_j: Number of joints.

$A \in R^{3\times3}$: Generalized mass matrix.

$\ddot{q} \in R^3$: Vector representing the acceleration of the robot w.r.t. the world frame, and the vector of joint accelerations w.r.t. the world frame.

$b \in R^3$: Coriolis force.

$g \in R^3$: Gravitation force.

$J_c \in R^{3\times3}$: Contact Jacobian.

$f_i \in R^3$: Reaction force obtained from the lumped mass model.

$\tau_f \in R^3$: Floating base torque.

$\tau_j \in R^3$: Joint torque.

2.7 COM orientation and angular velocity

The orientation of the COM of the robot can be expressed with the Euler angles $\Theta = [\phi \ \theta \ \psi]^T$. Here ϕ is roll, θ is pitch and ψ is yaw. In figure 2.10 it can be visualized the COM with its rotation angles. These angles are crucial for building the rotation matrix in which the body coordinates could be transformed into world coordinates.

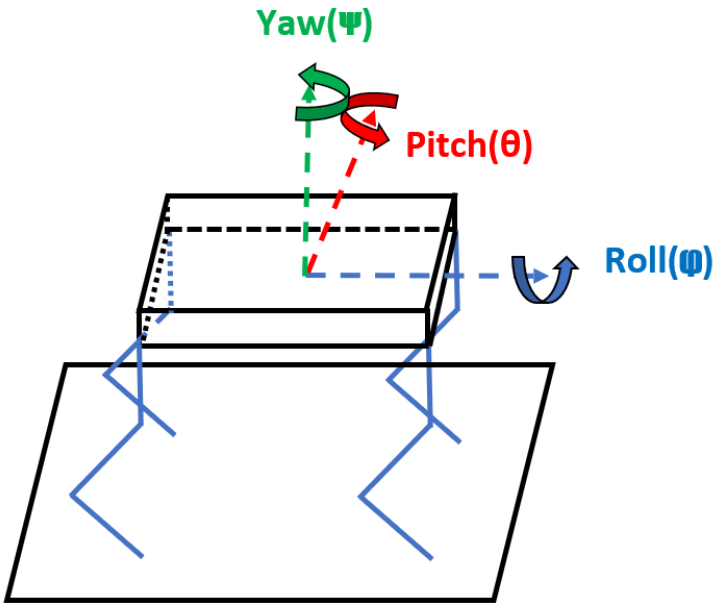


Figure 2.10: COM orientation.

With these matrices, a rotation matrix for every axis can be defined for every angle in which the robot performs its rotations. The rotation matrix of the robot for the ϕ angle is defined with equation 2.48 as:

$$R_x(\phi) = \begin{bmatrix} 1 & 0 & 0 \\ 0 & cos(\phi) & -sin(\phi) \\ 0 & sin(\phi) & cos(\phi) \end{bmatrix} \tag{2.48}$$

The rotation matrix of the robot for the θ angle is defined with equation 2.49 as:

$$R_y(\theta) = \begin{bmatrix} cos(\theta) & 0 & -sin(\theta) \\ 0 & 1 & 0 \\ sin(\theta) & 0 & cos(\theta) \end{bmatrix} \tag{2.49}$$

And finally the rotation matrix for the ψ angle is defined with equation 2.50 as:

$$R_z(\psi) = \begin{bmatrix} cos(\psi) & -sin(\psi) & 0 \\ sin(\psi) & cos(\psi) & 0 \\ 0 & 0 & 1 \end{bmatrix} \tag{2.50}$$

With these rotations matrices, the rotation matrix can be defined, where it transforms from body to world coordinates, such as in equation 2.51:

$$R = R_z(\psi)R_y(\theta)R_x(\phi) \tag{2.51}$$

With the first derivative of the COM orientation, it is possible to obtain the COM angular velocity. To obtain the COM angular velocity viewed from the world frame, the previously calculated rotation matrix R has to be transposed in the form of R^T. This can be expressed with equation 2.52 where the R^T matrix multiplies the COM angular velocity.

$$\begin{bmatrix} \dot{\phi} \\ \dot{\theta} \\ \dot{\psi} \end{bmatrix} = \begin{bmatrix} cos(\psi)/cos(\theta) & sin(\psi)/cos(\theta) & 0 \\ -sin(\psi) & cos(\psi) & 0 \\ cos(\psi)tan(\theta) & sin(\psi)tan(\theta) & 1 \end{bmatrix} w \tag{2.52}$$

Normally in practice roll, and pitch angles (ϕ, and θ) are really small, so the terms involving $cos(\theta)$ and $cos(\phi)$ can be simplified to 1, while $sin(\theta)$ and $sin(\phi)$ can be simplified to 0. Also because of the relationship of the tangent function is $tan(x) = sin(x)/cos(x)$, the terms involving $tan(\theta)$ and $tan(\phi)$ are simplified to 0. With these simplifications, the COM's angular velocity can be expressed using all the mentioned simplifications, reaching a simpler

equation for COM's angular velocity viewed from the world frame with equation 2.53:

$$\begin{bmatrix} \dot{\phi} \\ \dot{\theta} \\ \dot{\psi} \end{bmatrix} = \begin{bmatrix} cos(\psi) & sin(\psi) & 0 \\ -sin(\psi) & cos(\psi) & 0 \\ 0 & 0 & 1 \end{bmatrix} w \tag{2.53}$$

This is equivalent to equation 2.54:

$$\begin{bmatrix} \dot{\phi} \\ \dot{\theta} \\ \dot{\psi} \end{bmatrix} = R_z(\psi)^T w \tag{2.54}$$

As mentioned before, the quadruped can be viewed from the body frame, and from the world frame, to represent the inertia tensor viewed from the world frame, the rotation matrix can be used to have a relationship between these two frames which can be represented in equation 2.55:

$$\hat{I_w} = R_z(\psi)_B I R_z(\psi)^T \tag{2.55}$$

2.8 Simplified Robot Dynamics

With the usage of the obtained equations of the lumped mass model and the equations of the COM orientation and angular velocity, it is possible to have the dynamics of the quadruped. With the usage of a state-space representation the dynamics of the whole robot can be written in the form of equation 2.56:

$$\frac{d}{dt}\begin{bmatrix} \hat{\Theta} \\ \hat{p} \\ \hat{\omega} \\ \dot{\hat{p}} \end{bmatrix} = \begin{bmatrix} 0_3 & 0_3 & R_z(\psi)^T & 0_3 \\ 0_3 & 0_3 & 0_3 & 1_3 \\ 0_3 & 0_3 & 0_3 & 0_3 \\ 0_3 & 0_3 & 0_3 & 0_3 \end{bmatrix} \begin{bmatrix} \hat{\Theta} \\ \hat{p} \\ \hat{\omega} \\ \dot{\hat{p}} \end{bmatrix} + \begin{bmatrix} 0_3 & ... & 0_3 \\ 0_3 & ... & 0_3 \\ \hat{I}_w^{-1}[r_1]\times & ... & \hat{I}_w^{-1}[r_n]\times \\ 1_3/m & ... & 1_3/m \end{bmatrix} \begin{bmatrix} f_1 \\ : \\ : \\ f_n \end{bmatrix} + \begin{bmatrix} 0 \\ 0 \\ 0 \\ g \end{bmatrix} \tag{2.56}$$

The developed dynamic model involving the expanded matrices of equation 2.56 can be viewed in Appendix G.

To find the term of $\hat{I}_w^{-1}[r_1]\times$ the first thing to do is to calculate the inertia matrix viewed from the world frame, as referred to equation 2.55, the inertia matrix viewed from the world frame can be calculated with equation 2.57:

$$\hat{I}_w = \begin{bmatrix} cos(\psi) & -sin(\psi) & 0 \\ sin(\psi) & cos(\psi) & 0 \\ 0 & 0 & 1 \end{bmatrix} \begin{bmatrix} I_{xx} & 0 & 0 \\ 0 & I_{yy} & 0 \\ 0 & 0 & I_{zz} \end{bmatrix} \begin{bmatrix} cos(\psi) & sin(\psi) & 0 \\ -sin(\psi) & cos(\psi) & 0 \\ 0 & 0 & 1 \end{bmatrix} \tag{2.57}$$

After doing the appropriate matrix multiplications, the inertia matrix from the world frame can be viewed like in equation 2.58:

$$\hat{I}_w = \begin{bmatrix} I_{xx}cos(\psi)^2 + I_{yy}sin(\psi)^2 & I_{xx}cos(\psi)sin(\psi) - I_{yy}cos(\psi)sin(\psi) & 0 \\ I_{xx}cos(\psi)sin(\psi) - I_{yy}cos(\psi)sin(\psi) & I_{yy}cos(\psi)^2 + I_{xx}sin(\psi)^2 & 0 \\ 0 & 0 & I_{zz} \end{bmatrix} \tag{2.58}$$

Then, after getting the inverse transform viewed from the world frame, yields to equation 2.59:

$$\hat{I}_w^{-1} = \begin{bmatrix} I_{xx} - I_{xx}sin(\psi)^2 + I_{yy}sin(\psi)^2 & \frac{sin(2\psi)(I_{xx}-I_{yy})}{2} & 0 \\ \frac{sin(2\psi)(I_{xx}-I_{yy})}{2} & I_{yy} + I_{xx}sin(\psi)^2 - I_{yy}sin(\psi)^2 & 0 \\ 0 & 0 & I_{zz} \end{bmatrix} \tag{2.59}$$

The next step is to calculate the skew matrix for a given position of a foot r_1, given its position with the vector $r_1 = [a_1\ b_1\ c_1]$, its skew matrix can be represented like in equation 2.60:

$$[r_1]\times = \begin{bmatrix} 0 & -c_1 & b_1 \\ c_1 & 0 & -a_1 \\ -b_1 & a_1 & 0 \end{bmatrix} \tag{2.60}$$

Finally, after making the multiplication of the inverse inertia matrix viewed from the world frame with the skew matrix of the foot position, it is obtained an expression that appears in Appendix H:

It is important to mention that a new state of gravity was added to have a convenient

state-space representation equation, which expanded in all its components, yields to equation 2.61:

$$\dot{x}(t) = A_c(\psi)x(t) + B_c(r_1, ..., r_n, \psi)u(t) \tag{2.61}$$

In this equation $A_c \in R^{13\times13}$ and $B_c \in R^{13\times3n}$. Here it can be seen that this state-space representation depends on the ψ angle and the foot locations. With the proper dynamics model, it is possible to apply control and make the experiments using diverse platforms.

Chapter 3: Experiments and Simulation

For the experiments, it was done mainly doing a simulation of the whole quadruped using a previously proposed control, and then a simulation of a single leg using the newly proposed controller. For the simulation of the whole quadruped robot, it was tested in SIL using the Pybullet Platform. After testing the quadruped in the Pybullet platform, a Simulation with Simulink was made for the swing phase.

3.1 Software in the Loop

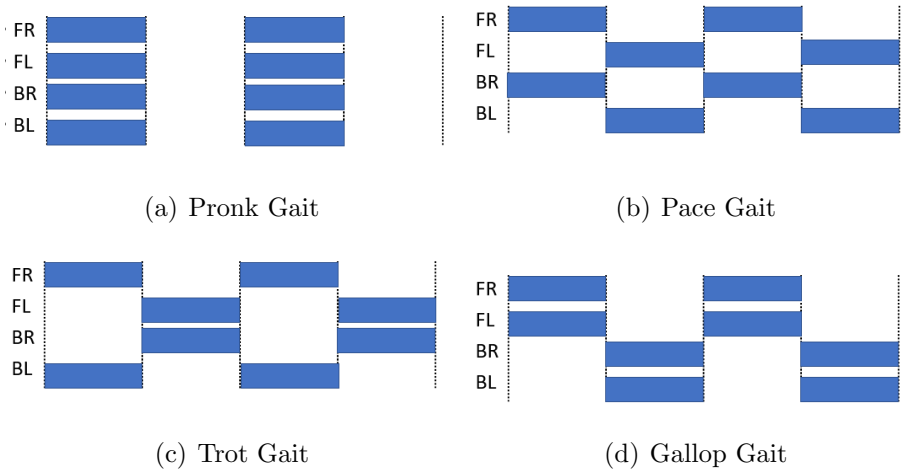
For the simulation of the robot, the quadruped imitates the gaits of real animals, so it is very important to analyze the pattern of locomotion of the legs to have a good simulation of the quadruped. To have proper locomotion of the robot, each leg must have two phases of locomotion. The first analyzed phase is the stance phase; in this phase, the leg is in direct contact with the ground, where a reaction force is generated. The other phase of the leg is the swing phase, in which the leg is not in contact with the ground, contrary to the stance phase, with this phase there is no contact force with the ground.

To perform these gaits some parameters are needed to take into account to alternate between the two phases. One of these parameters is the nominal stance duration, which is the duration of the stance phase w.r.t. the total cycle, which for the experiment is 0.3, representing 30 % of the cycle. Another important parameter that needs to be considered is the duty factor, which has a relationship that involves the stance duration divided by the total gait cycle. This parameter is important to achieve the desired gaits of locomotion of the quadruped. With these parameters, the swing duration can be calculated with equation 3.1, in which it considers the stance duration as well as the duty factor.

$$Swingduration = Stanceduration/Dutyfactor - Stanceduration \tag{3.1}$$

It is possible to implement in a robot the gaits, alternating correctly the stance phase and the swing phase of each leg. In figure 3.1 it is shown the four main gaits of locomotion which are trot, gallop, pace, and pronk in which the blue spaces represent the contact points of each leg during the stance phase, while the blank spaces represent the swing phase. The first letter of the leg determines the frontal, or back position of the leg, for a frontal leg is designated the F, and for a back leg it is designated the B; whereas for the position of the right or the left leg is referenced in the second letter, for the right leg is designated with the R, and for the left leg is designated with the L.

Considering the two main phases of locomotion of the legs of the quadruped, the torques



(a) Pronk Gait (b) Pace Gait

(c) Trot Gait (d) Gallop Gait

Figure 3.1: Quadruped gaits.

of the motor can be computed using a PD controller like in equation 3.2 with the parameters of table 3.1.

$$\tau_i = K_p(\theta_d - \theta)_i + K_d(\dot{\theta}_d - \dot{\theta})_i + additional\ torques_i \tag{3.2}$$

Gains applied to each quadruped leg		
Joint	K_p	K_d
Horizontal Hip Joint	220	1
Vertical Hip Joint	220	2
Knee Joint	220	2

Table 3.1: Gains used in the swing leg controller of the quadruped.

It is important to consider that during the swing phase, the motor torques are computed using the PD controller of equation 3.2, while during the stance phase, the term additional torques are calculated with the aid of MPC, which uses a Quadratic Programming (QP) solver to compute the contact forces of the legs of the robot. The QP solver helps to optimize the correct contact forces to obtain the minimal forces that are necessary to keep the quadruped in the desired state of its COM. The QP solver used in the simulation is the OSQP solver[18] which solves convex quadratic programming problems of the form:

minimize $\frac{1}{2}x^T P_x + q^T + x$

subject to $l \leq Ax \leq u$

Here $x \in R^n$ is the variable that is planned to be optimized. The matrix that is the objective of the optimization is defined as a positive semidefinite matrix $P \in S^n$ and a

vector $q \in R^n$. The linear constraints of the equation are defined by a matrix $A \in R^{mxn}$ and also by the vectors l and u in order to let $l_i \in R-\infty$ and $u_i \in R+\infty$ for all $i \in 1, ..., m$. The objective of OSQP is to compute the optimal contact forces for a given center of mass trajectory and a predefined gait pattern. These contact forces are really important because, with them, the motor torques are calculated in the stance phase of the locomotion. The parameters that are considered for computing the contact forces are: COM position, COM velocity, COM rpy, COM angular velocity, foot contact states, foot positions in the body frame, foot friction coefficients, desired COM position, desired COM velocity, desired COM rpy and desired COM angular velocity.

Through the simulation, some matrices need to be updated to have an accurate optimization process. These matrices are the state, the desired states, the contact states of the legs, foot positions from the base frame, foot friction coefficients, rotation matrix, inertia world frame, A matrix, and B matrix from the dynamic model of equation 2.61. To update these matrices, the MPC plans 10 horizon steps with a planning timestep of 0.25, which means that the QP solver plans within a frame of 2.5 seconds the correct quantity of contact forces so that the quadruped has the desired position and velocity of its COM. Also, it is worth mentioning that the bullet solver makes 30 iterations with a simulation time step of 0.001 seconds, which means that to execute the SIL simulation, the platform has to make 30,000 iterations in one second.

As it was mentioned before, the quadruped has mainly two frames, the world frame, and the body frame, so the user can appreciate the locomotion of the robot within the world frame, where it can rotate the camera to view the robot in different angles. The user can input a desired trajectory of the COM, that is a velocity in the x axis vx, a velocity in the y axis vy or an angular velocity in the z axis wz in 5 seconds for each change of trajectory. To watch the SIL simulation, it was used the platform Pybullet, with the robot Laikago from Unitree Robotics .

One test of the SIL is to compare the desired COM position in the z axis with the actual COM position in the z axis. In figure 3.2 is displayed the position of the COM of the quadruped during a simulation.

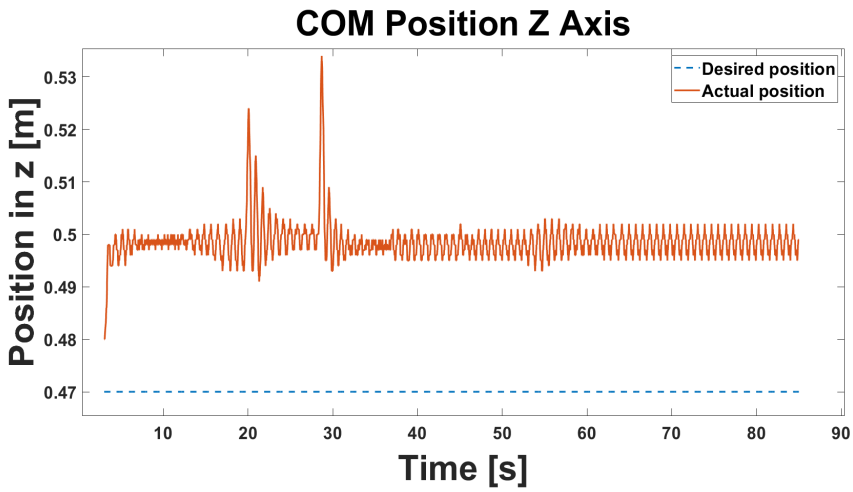


Figure 3.2: Actual COM position vs desired COM position of the z axis.

It is important to note that during the change of the trajectory of the COM, the actual COM Position changes as can be seen in the graph around second 20. Even though that the COM changes from its desired state with a change of trajectory, it still manages to keep close to the desired COM position. The objective of this control is to have stability in the COM, as it can be seen in figure 3.3 when the quadruped is subjected to a disturbance.

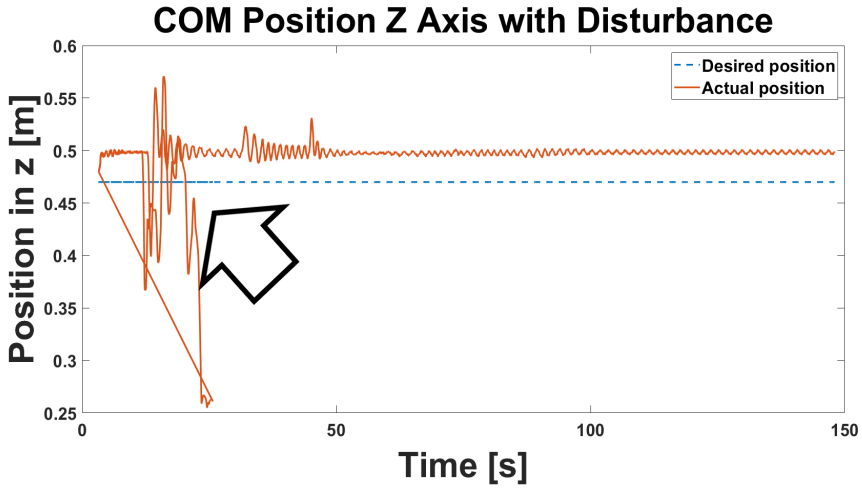


Figure 3.3: Actual COM position vs desired COM position of the z axis when the quadruped is subjected to a disturbance.

Another state that was observed using SIL was the COM velocity, comparing the state of the desired COM velocity in the z axis with the actual COM velocity in the z axis. In figure 3.4 is displayed the velocity of the COM of the quadruped during a simulation.

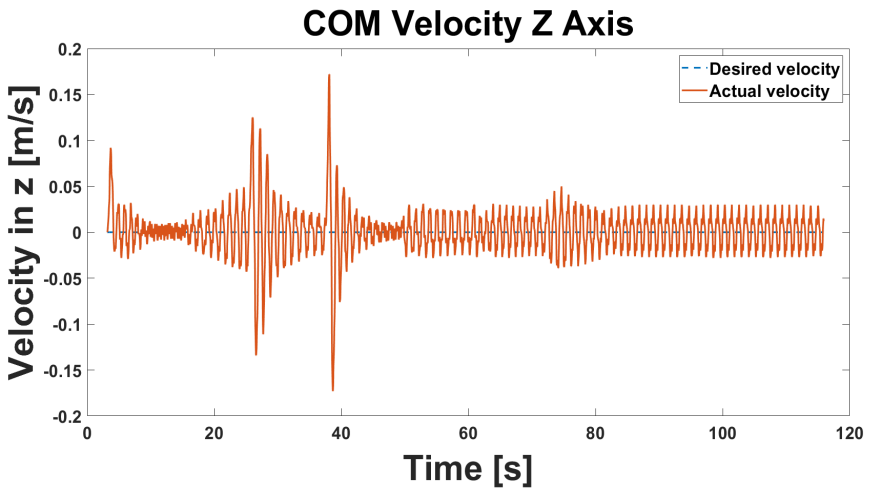


Figure 3.4: Actual COM velocity vs desired COM velocity of the z axis.

It can be seen that even though it oscillates through all the simulation, it still handles to be around the desired velocity of the z axis which is 0 m/s. With the usage of this control, it can still respond to disturbances, as it can be seen in figure 3.5, after an applied disturbance, the control actuates to keep the quadruped in the desired velocity.

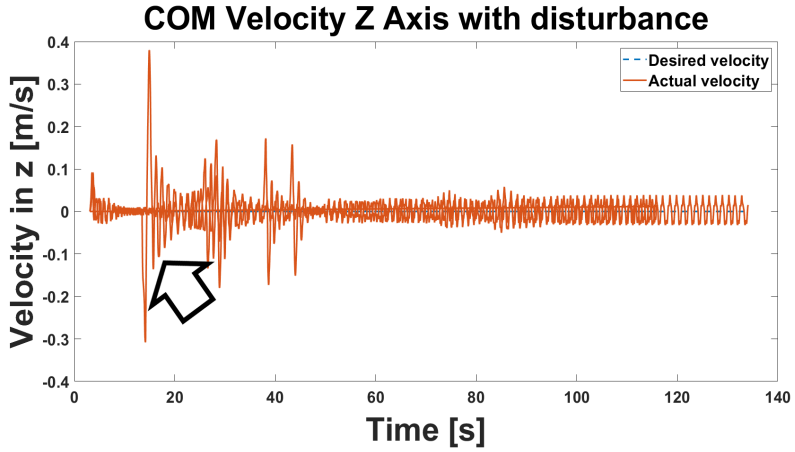


Figure 3.5: Actual COM velocity vs desired COM velocity of the z axis after applying a disturbance.

It can be seen that with a disturbance, the COM velocity still oscillates in the z plane, but it still manages to be around the desired COM velocity in z, which is 0 m/s.

To keep the robot stable is important that each of the legs contribute to the control to keep the robot stable. For a leg with 3 motors, each of the motors computes the right torques to keep the quadruped stable. In figure 3.6 it is compared the desired position vs the actual position of the motors of a leg. Each motor, compute the right torques to keep the robot stable according to equation 3.2.

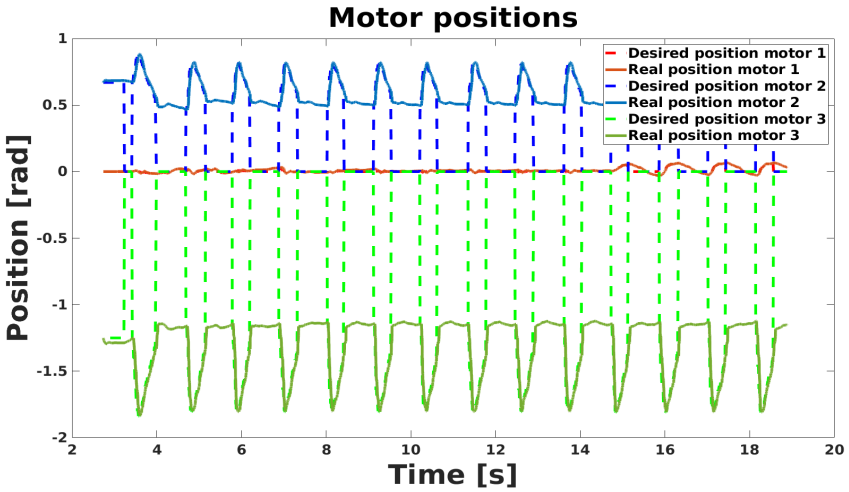


Figure 3.6: Actual position vs desired position of the motors of a leg of the quadruped in simulation.

It is important to mention, after the control law of equation 3.2 is applied to the motors, they compute the torques to have stability in the quadruped. In figure 3.7 it is shown the torques of 3 motors of a robot leg over a period of time during a simulation.

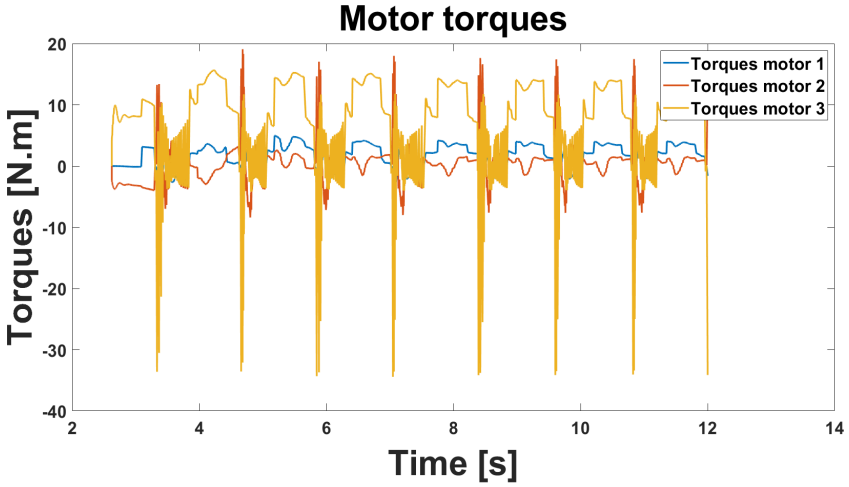


Figure 3.7: Motor torques of a leg of the quadruped in simulation.

With the usage of the Hybrid motor model, it is feasible that each of the motors reaches the desired position and the desired velocities to compute the right torques that are needed to have the proper states of the COM. With the correct appliance of motor torques, the robot is robust to disturbances where the user can input them in the body or any of the legs of the quadruped.

For the SIL simulation, it can be used an even terrain, where the quadruped can walk

freely without any irregularities in the ground as can be seen in figure 3.8. This is the most recommended ground to begin to work with the quadruped and to test the gaits of the quadruped.

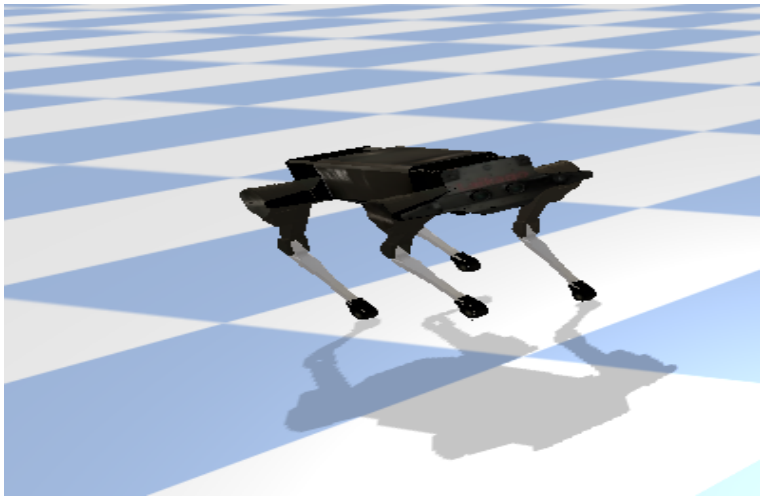


Figure 3.8: Quadruped in even terrain.

In the SIL simulation, the quadruped also has an algorithm in which it reacts to the terrain disturbances, in which the quadruped may behave correctly in uneven terrain. When it is in Swing Phase and it detects an obstacle before it finishes executing the Swing Phase, it enters an Early Contact detection State, where it changes immediately to Stance Phase. Also in the Stance Phase, when the leg loses contact before the scheduled time for the Stance Phase it enters a Loose Contact Detection State, where the leg enters the Swing Phase immediately. In figure 3.9 it can be seen the test that was made with the quadruped Laikago in uneven terrain.

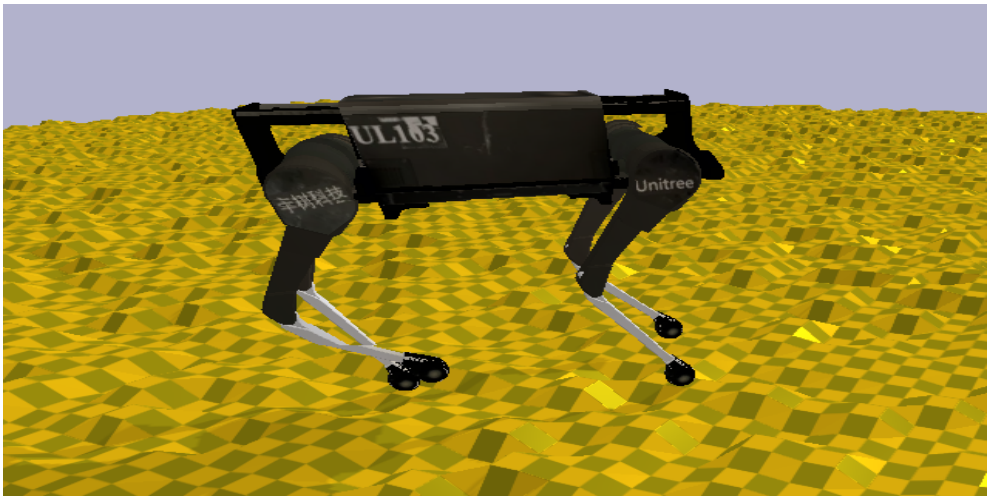


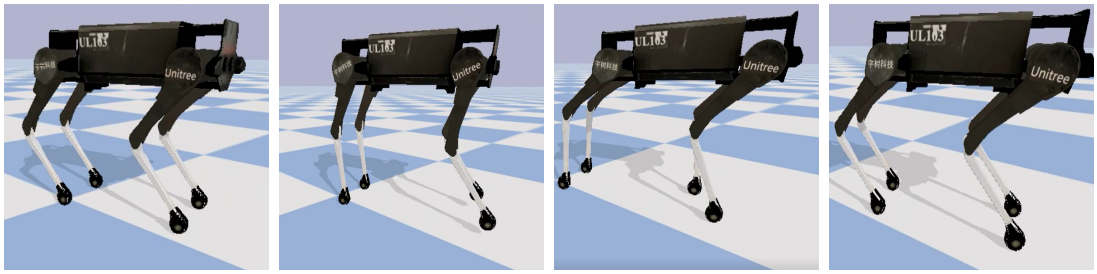
Figure 3.9: Quadruped in uneven terrain.

After testing the robot in even and in uneven terrain, the next part is to test the walking gaits of the quadruped. To achieve each of the gaits of the quadruped, it is necessary to alter the initial gait state of each leg, its initial phase of the cycle, and the duty factor. In table 3.2 it is summarized the parameters that were used to achieve the desired gaits of the quadruped in an even terrain. The leg which is referenced for each parameter is in the following order: FR, FL, BR, and BL. In the initial state of the leg, SW stands for Swing, while ST stands for Stance.

Gait parameters			
Gait	**Duty factor**	**Initial phase**	**Initial State**
Pronk	0.8, 0.8, 0.8, 0.8	0.9, 0.9, 0.9, 0.9	SW, SW, SW, SW
Pace	0.6, 0.6, 0.6, 0.6	0.9, 0, 0.9, 0	SW, ST, SW, ST
Gallop	0.73, 0.73, 0.73, 0.73	0.9, 0.9, 0, 0	SW, SW , ST, ST
Trot	0.6, 0.6, 0.6, 0.6	0.9, 0, 0, 0.9	SW, ST, ST, SW

Table 3.2: Parameters used to achieve the different gaits of the quadruped.

For the pronk gait, it was tested with its corresponding parameters from table 3.2, where a SIL simulation was previously made to test the performance of the gait. In figure 3.10 it can be seen a series of images taken from a video of a simulation of the pronk gait. A distinctive quality of this gait is that the four legs change of phases at the same time.



(a) FR, FL, BR, BL legs are in stance phase

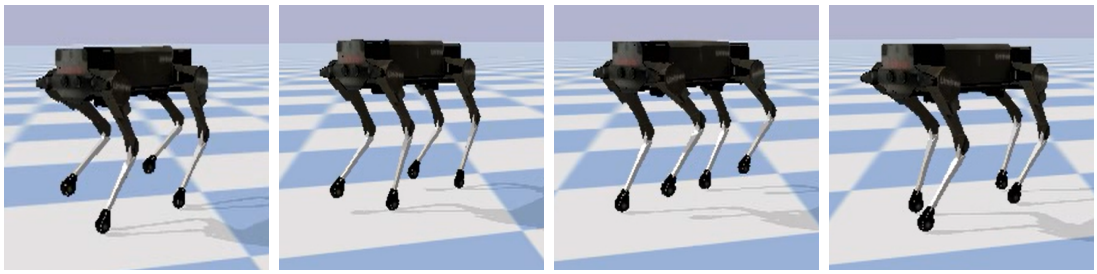
(b) FR, FL, BR, BL legs are in swing phase

(c) FR, FL, BR, BL legs in stance phase

(d) FR, FL, BR, BL legs are back in stance phase

Figure 3.10: Pronk gait simulation using Pybullet software where it shows the significant frames of the simulation.

For the pace gait, it was tested with its corresponding parameters from table 3.2, where a SIL simulation was previously made to test the performance of the gait. In figure 3.11 it can be seen a series of images taken from a video of a simulation of the pace gait. A characteristic of this gait is that it changes of phases between the legs that are located in the left, from the legs that are located on the right.



(a) FL, and BL are in stance phase, FR, and BR legs are in swing phase

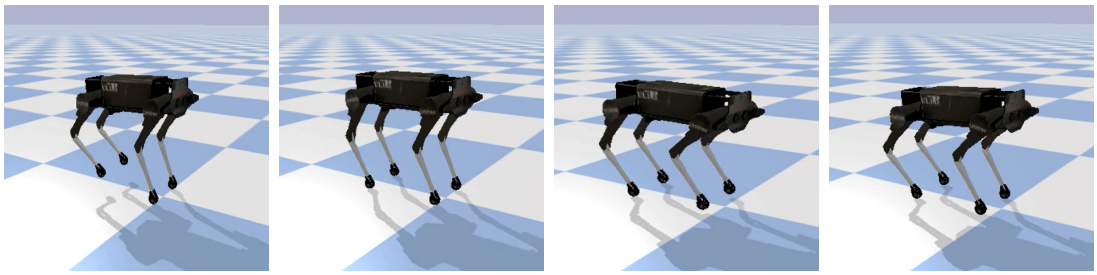
(b) FR and BR legs finish the swing phase

(c) FR, and BR are in stance phase, FL, and BL legs are in swing phase

(d) FL and BL legs finish the swing phase

Figure 3.11: Pace gait simulation using Pybullet software where it shows the significant frames of the simulation.

For the gallop gait, it was tested with its corresponding parameters from table 3.2, where a SIL simulation was previously made to test the performance of the gait. In figure 3.12 it can be seen a series of images taken from a video of a simulation of the gallop gait. A characteristic of this gait is that it changes of phases between the legs that are located in the front, from the legs that are on the back.



(a) FR, and FL are in stance phase, BL, and BR legs are in swing phase

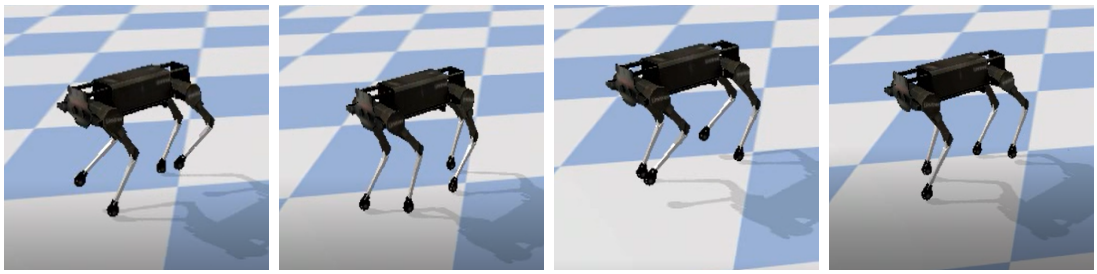
(b) BR and BL legs finish the swing phase

(c) BR, and BL are in stance phase, FR, and FL legs are in swing phase

(d) FR and FL legs finish the swing phase

Figure 3.12: Gallop gait simulation using Pybullet software where it shows the significant frames of the simulation.

For the trot gait, it was tested with its corresponding parameters from table 3.2, where a SIL simulation was previously made to test the performance of the gait. In figure 3.13 it can be seen a series of images taken from a video of a simulation of the trot gait. A characteristic of this gait is that it changes of phases between the legs that are located crisscrossed.



(a) BR, and FL are in stance phase, BL, and FR legs are in swing phase

(b) FR and BL legs finish the swing phase

(c) FR, and BL are in stance phase, BR, and FL legs are in swing phase

(d) BR and FL legs finish the swing phase

Figure 3.13: Trot gait simulation using Pybullet software where it shows the significant frames of the simulation.

3.2 Simulation of swing phase controller

For this experiment, a Swing control was proposed to test the leg in the swing phase of the locomotion. For the simulation of a robotic leg, it is used the dynamic model of the leg of 3 DOF with 3 links. With the proposed model of section 2.5 for the dynamics of the leg, the vector of $\ddot{q}$ can be obtained to implement it on the control, the equation of the actual acceleration of the torques of the motors is obtained with equation 3.3:

$$\ddot{q} = M(q)^{-1}(\tau - C(q,\dot{q})\dot{q} - G(q)) \tag{3.3}$$

With the vector of the actual acceleration, an integration can be applied to obtain a vector of the actual velocity of the motors represented by $\dot{q}$, and an integration to the vector of the velocities can be applied as well, to obtain a vector of the actual position of the motors represented by the vector q, in figure 3.14 it can be seen how these vectors are obtained using Simulink. It is important to consider this vectors, because they are used in the model and in the proposed control.

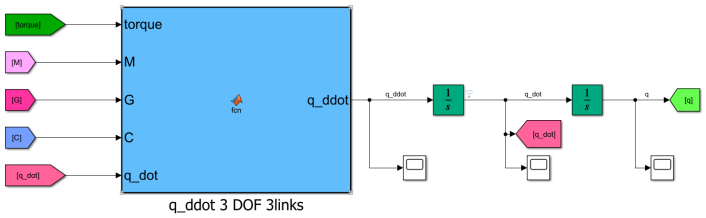


Figure 3.14: Representation of the acceleration, velocity and position in Simulink.

To begin the experiment a robot leg was designed using the Denavit Hartenberg convention of table 2.2 using the Peter Corke Toolbox[5] to design the leg. The parameters for the dimensions of the leg were chosen to resemble the leg used in simulation, which can be seen in table 3.3.

Leg parameters		
Link	**Length [m]**	**Mass [kg]**
Hip L_1	0.08	0.1
Thigh L_2	0.2	0.5
Calf L_3	0.2	0.5

Table 3.3: Parameters used to design the leg of the quadruped.

With the usage of the Peter Corke Toolbox, the parameters from table 3.3 can be used to design a FR leg, which can be visualized as in figure 3.15.

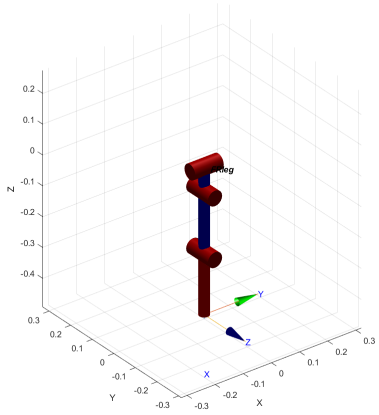


Figure 3.15: Robot leg created using the Peter Corke library.

For the experiment, a reference trajectory was used of the foot w.r.t. frame 0. The proposed reference simulates the robot making a swing trajectory with the form of a circle. The reference trajectory can be visualized in figure 3.16. It is worth mentioning that a circular trajectory was selected because it was observed that during an abrupt change of direction of a reference trajectory, a large acceleration was needed in order to reach the desired trajectory. The proposed trajectory is a circle of 2.5 cm of radius, and the end effector has to complete the proposed trajectory in $\simeq 1 second$.

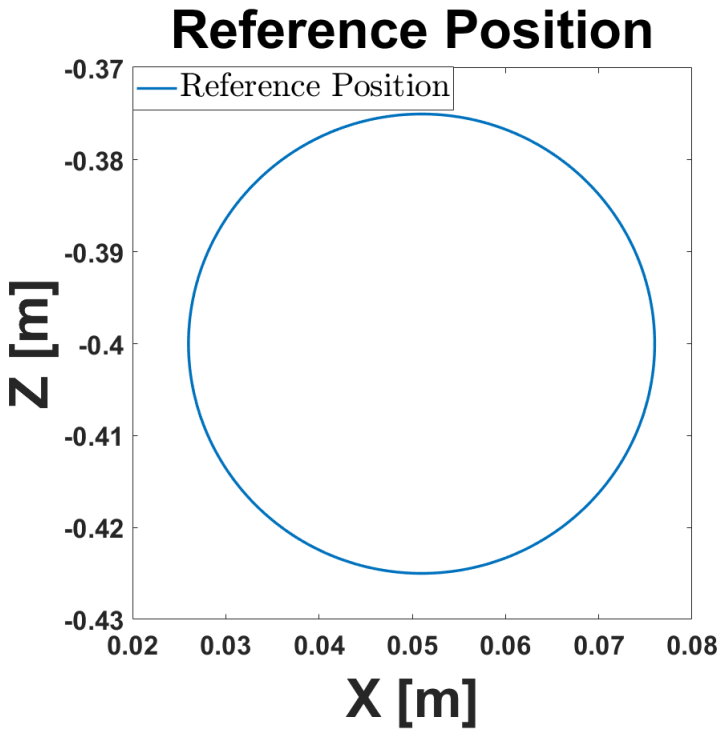


Figure 3.16: Reference of trajectory of the end effector in the X-Z plane.

To follow the given trajectory of the end effector, an Inverse Kinematics algorithm must be implemented in order that the angles are calculated and the joints make their respective angle. It is important to mention that in order to reach a point for a 3 DOF system there are multiple solutions that the joints can make in order to reach the desired point. In order to compute the desired trajectory, an Inverse Kinematics function used from the Peter Corke Robotics Toolbox was implemented to calculate the joint angles to follow a trajectory. With the Inverse Kinematics equations, the reference of the angles can be acquired with a qref vector that can be used to test the control. It's important to mention that in order to acquire these reference angles, all the points of 3.16 were taken into account and the Inverse Kinematics command of the Peter Corke library was used to get the reference angles.

By deriving the q_{ref} vector, a vector of reference velocities can be obtained designated by $\dot{q}_{ref}$, and by having a derivative reference velocity, a vector of reference accelerations can be obtained, which is designated by $\ddot{q}_{ref}$. It is observed that if the reference of the velocity $\dot{q}_{ref}$, and of the acceleration $\ddot{q}_{ref}$ is of a small quantity, it is easier to have a controllable system. In figure 3.17 it can be seen the reference of the angles.

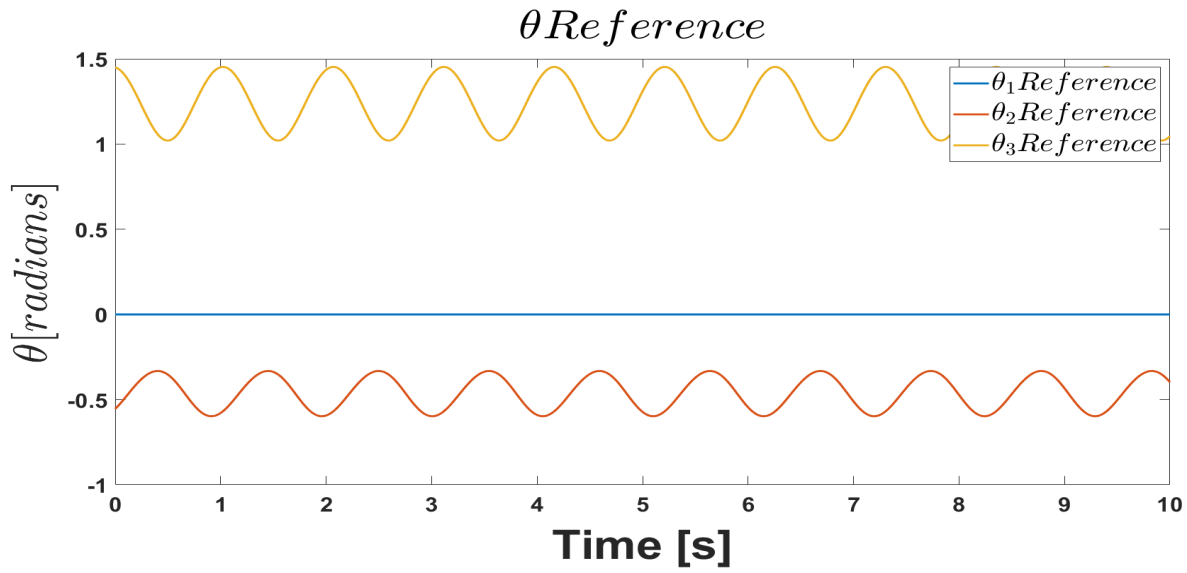


Figure 3.17: Reference of the angles of the θ vector.

In figure 3.18, after deriving the reference position of the angles, the vector of reference velocities can be obtained, it can be seen the reference of the velocities of the motors used in the simulation.

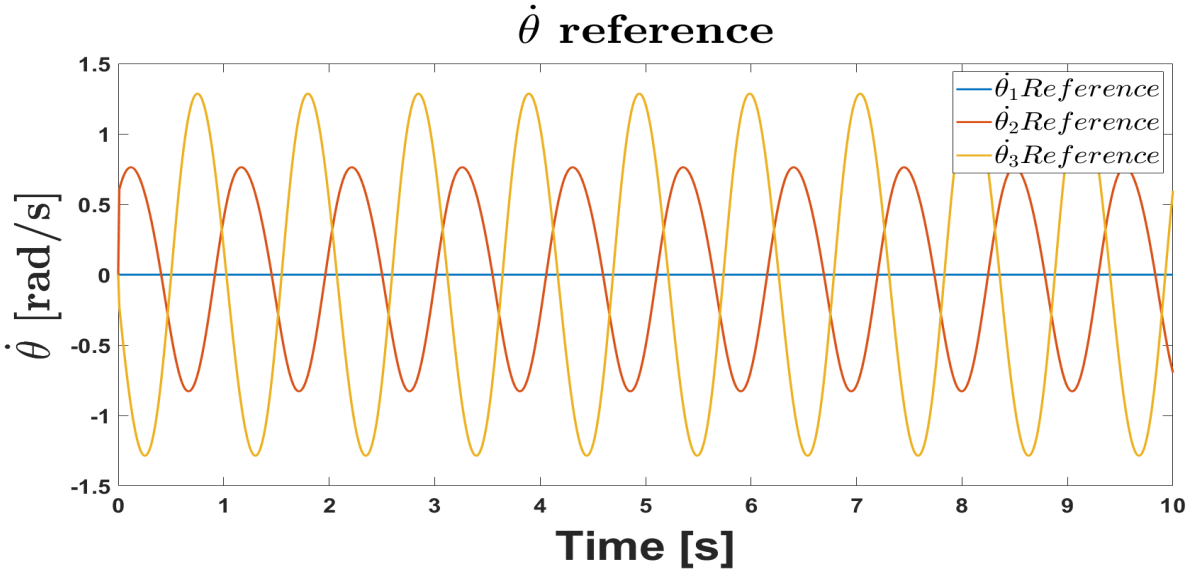


Figure 3.18: Reference of the velocities of the $\dot{\theta}$ vector.

The proposed model for the dynamics is derived from equation 2.56, and its implementation in Simulink can be seen in figure Appendix I.

Originally a control proposed in [6] with the form of $\tau = J(q)^T(-Kp(q_{ref}-q)-Kd(\dot{q}_{ref}-\dot{q}))+FFT$ was proposed, but after tests, the control was divergent leading to results like in figure 3.19. In this part of the experiment, it was compared the $\dot{\theta}_1$ with the reference of the $\dot{\theta}_1$.

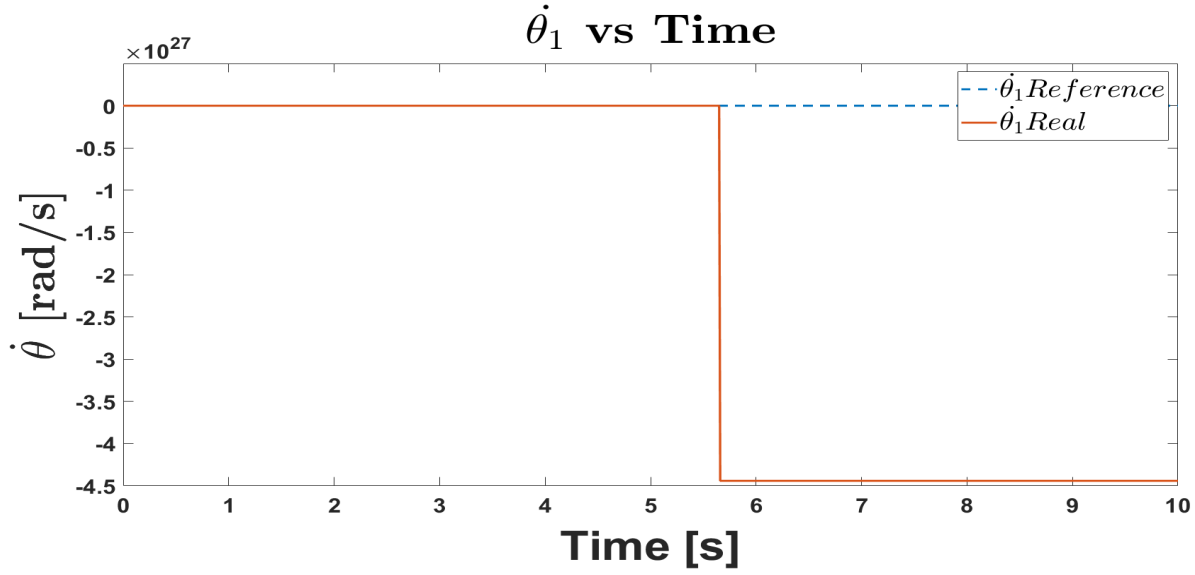


Figure 3.19: Divergent system of the control proposed in [6].

To correct this, a control applied to the model can be applied using equation 3.4. It is important to mention that the control proposed in [6] was changed so that the M matrix multiplies the control instead of the transposed Jacobian.

$$\tau = M(q)(-Kp(q_{ref} - q) - Kd(\dot{q}_{ref} - \dot{q})) + FFT \tag{3.4}$$

The FFT is a feedforward term that is used to compensate the torques due to the acceleration, these torques can be calculated with the following equation 3.5:

$$FFT = (J(q)^T * \Delta)(\ddot{q}_{ref} - \dot{J(q)}\dot{q}) + C(q,\dot{q})\dot{q} + G(q) \tag{3.5}$$

The Operational Space Inertia Matrix $OSIM$[12] matrix can be represented with letter Δ is determined by the equation 3.6 in which it involves the $M(q)$ matrix previously calculated, the Jacobian $J(q)$, and the transposed Jacobian $J(q)^T$:

$$\Delta = (J(q)M^{-1}(q)J^T(q))^{-1} \tag{3.6}$$

It's important to mention that during the experiments in Simulink, when the system finds a singularity, the Δ matrix becomes extremely large and the control can't converge, so it´s important to take into account a correct trajectory for the end effector to have a feasible system that can be controlled. To track the trajectory of the end effector, the right torques must be computed to follow the desired trajectory. The control implemented in the Simulink software can be visualized in Appendix J.

After the simulation, there were displayed some graphs of the angles of the system comparing the reference with the real angle that is being followed during the simulation. The θ_1 angle during simulation represented the behavior of the horizontal hip joint, where it can be seen that even though the reference angle is zero for the whole simulation, it still makes small movements in reality as can be seen in figure 3.20.

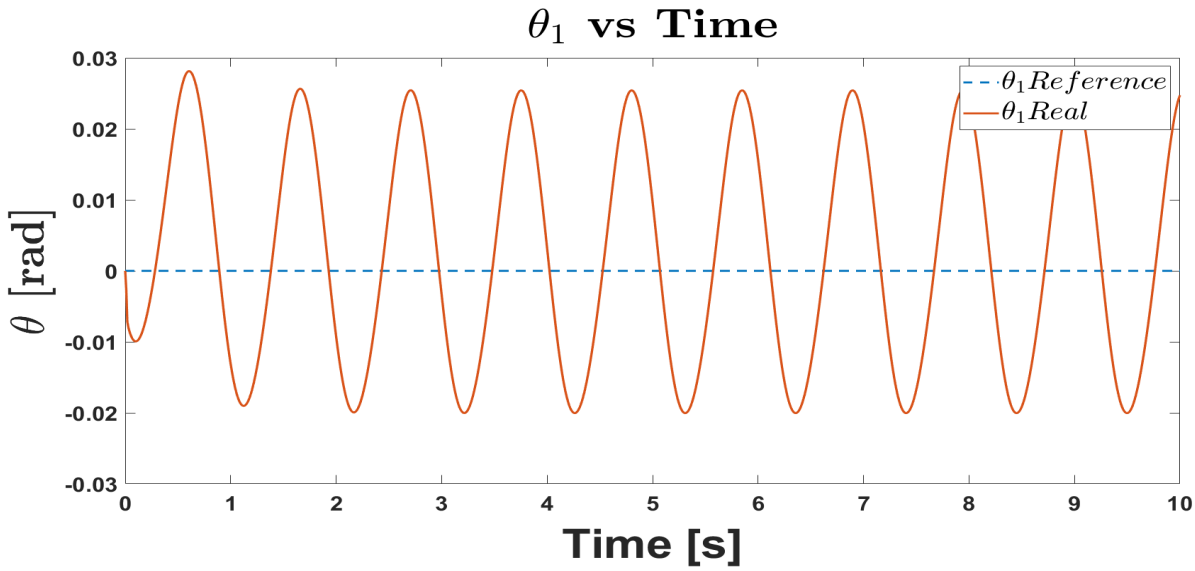


Figure 3.20: Real θ_1 angle vs the reference θ_1 during simulation.

The θ_2 angle during simulation represents the behavior of the vertical hip joint, following the desired reference trajectory as it can be seen in figure 3.21.

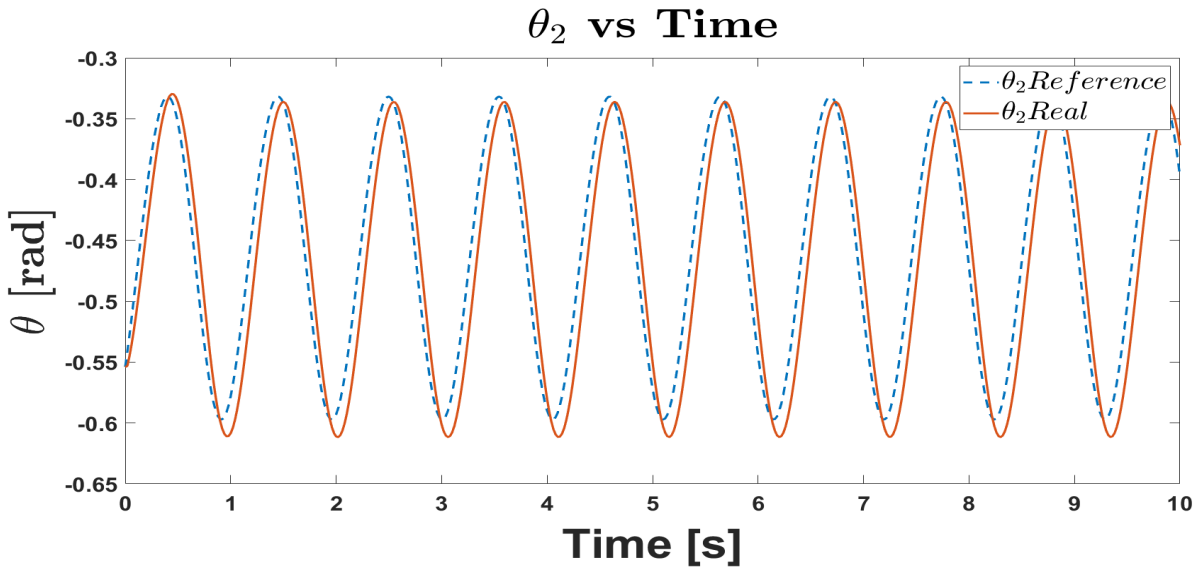


Figure 3.21: Real θ_2 angle vs the reference θ_2 during simulation.

The θ_3 angle during simulation represents the behavior of the knee joint, following the desired reference trajectory as it can be seen in figure 3.22.

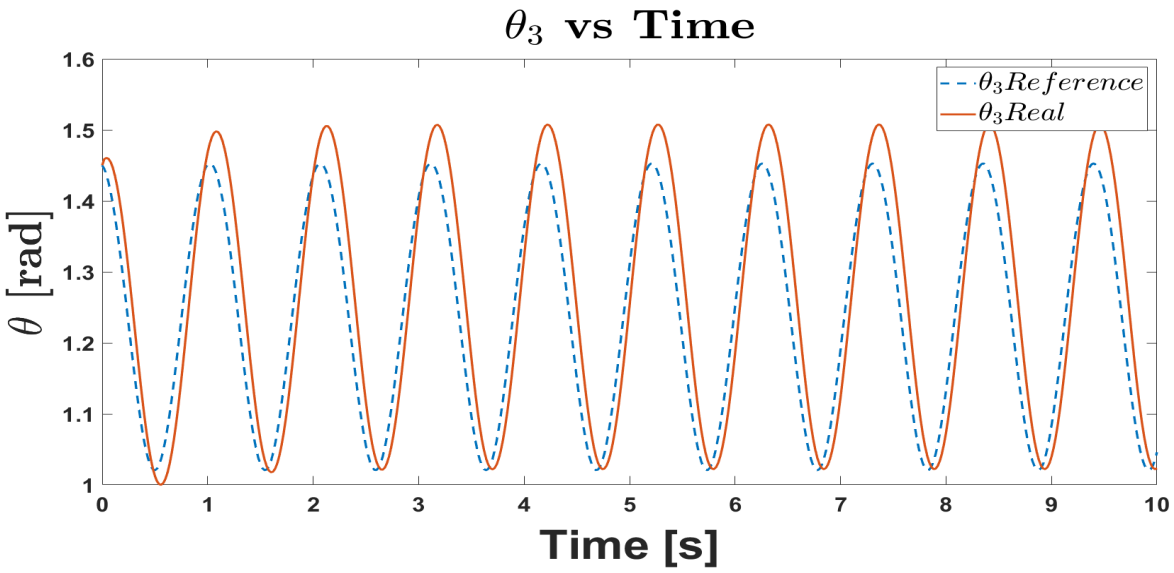


Figure 3.22: Real θ_3 angle vs the reference θ_3 during simulation.

The $\dot{\theta_1}$ angular velocity during simulation represented the angular velocity of the horizontal hip joint, where it can be seen that even though the reference angular velocity is zero

for the whole simulation, it still makes small movements in reality as it can be seen in figure 3.23.

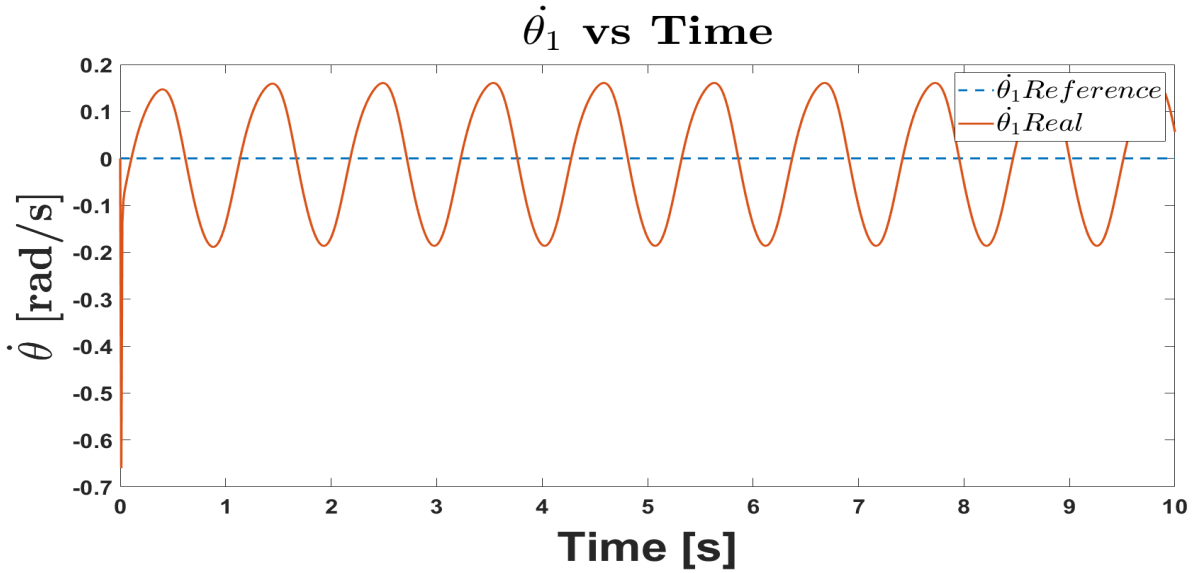


Figure 3.23: Real $\dot{\theta}_1$ vs the reference $\dot{\theta}_1$ during simulation.

The $\dot{\theta}_2$ angular velocity during simulation represents the angular velocity of the vertical hip joint, following the desired reference trajectory as it can be seen in figure 3.24.

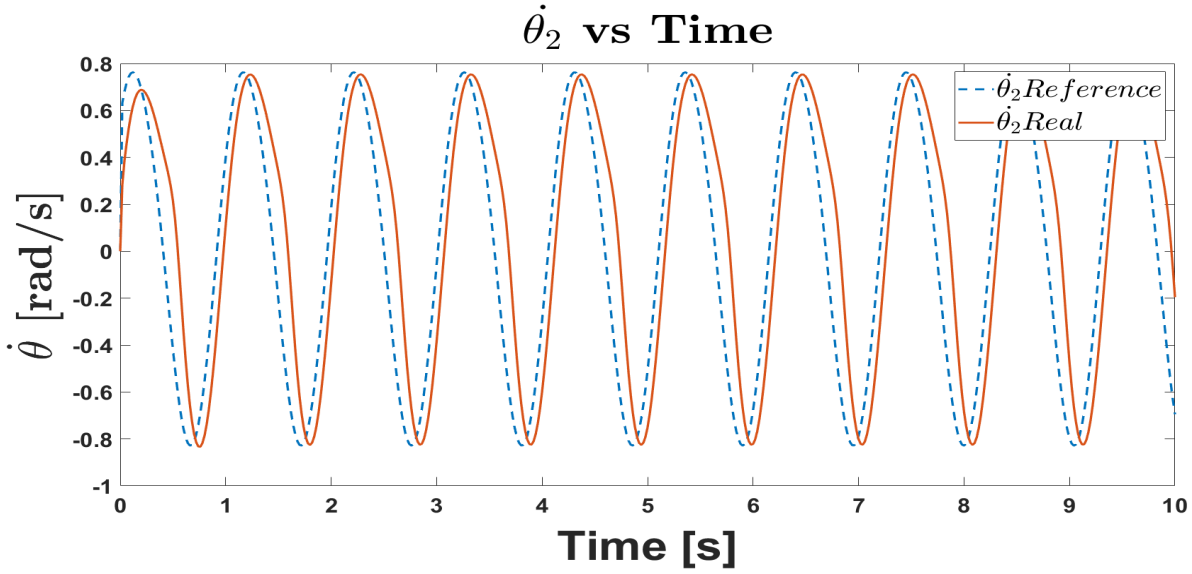


Figure 3.24: Real $\dot{\theta}_2$ vs the reference $\dot{\theta}_2$ during simulation.

The $\dot{\theta}_3$ angular velocity during simulation represents the angular velocity of the knee joint, following the desired reference trajectory as it can be seen in figure 3.25.

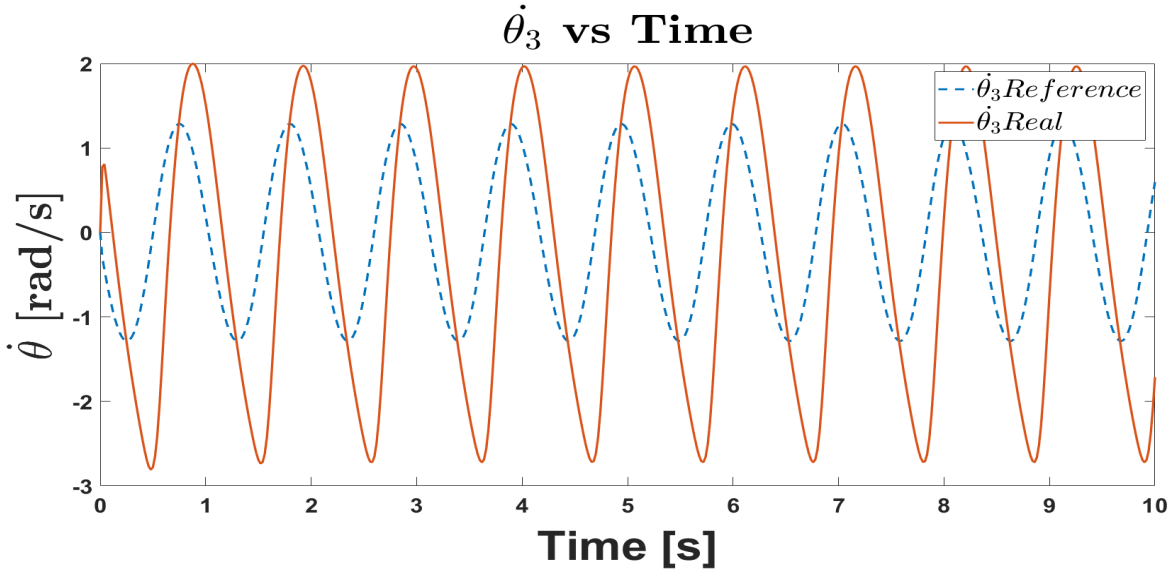


Figure 3.25: Real $\dot{\theta}_3$ vs the reference $\dot{\theta}_3$ during simulation.

After applying the control to the system and having a good track following the reference angles for each of the joints, the resulting torques can be seen in figure 3.26.

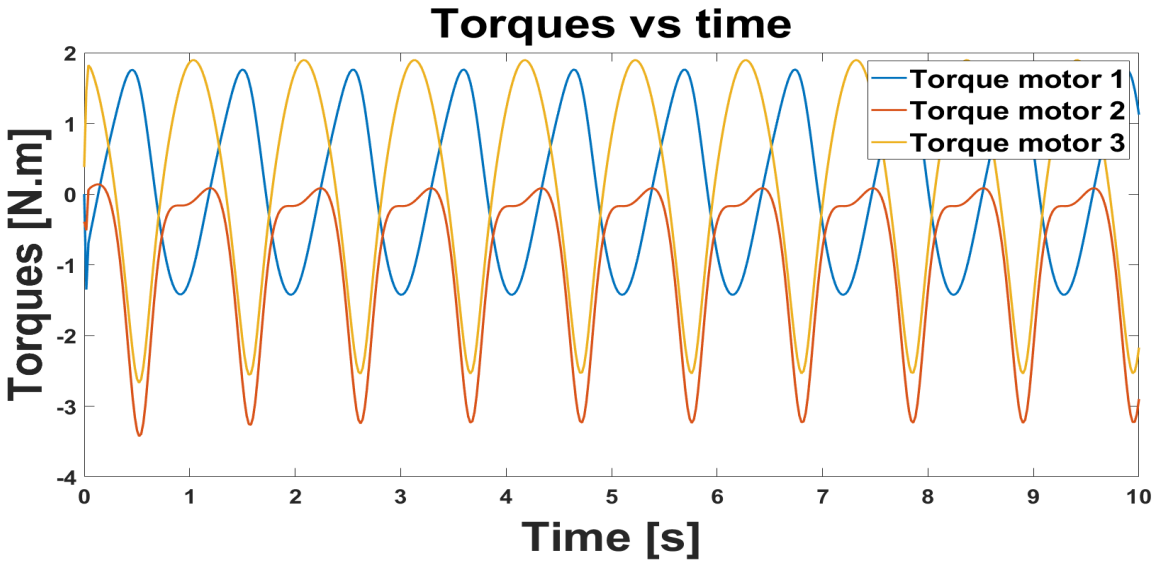


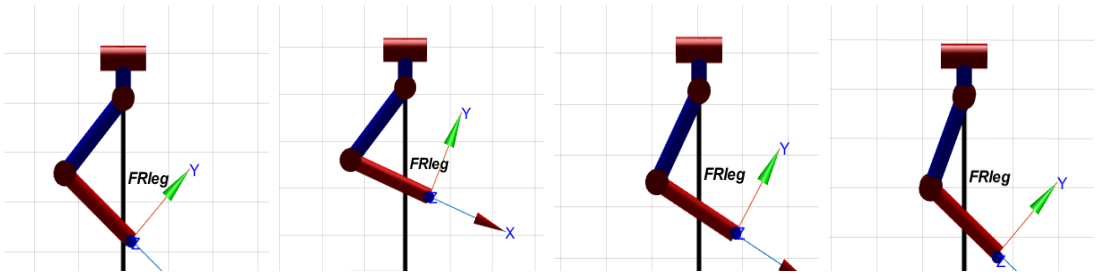
Figure 3.26: Motor torques during simulation.

In order to reach these desired positions and to have a correct tracking of the position of the joints, the gains of the PD controller had to be adjusted to in order to have a proper track of the control. These parameters can be shown in table 3.4.

Gains applied to the quadruped leg		
Joint	K_p	K_d
Horizontal Hip Joint	400	180
Vertical Hip Joint	700	190
Knee Joint	350	250

Table 3.4: Gains used in the swing leg controller of the quadruped.

To prove the correct locomotion of the leg, after obtaining the real angles in simulation, the Forward Kinematics equation obtained in equation 2.16 of the position P of the end effector was applied at the leg using the Peter Corke Toolbox, to visualize the proper track of the angles of the joints, as it can be seen in figure 3.27.



(a) FR leg at the beginning of the simulation

(b) FR leg reaching its maximum position in the Z plane during simulation

(c) FR leg descending during simulation

(d) FR leg finishing its trajectory during simulation

Figure 3.27: FR leg of a robot during simulation in the Simulink software.

Also after applying the Forward Kinematics equations it can be compared the proposed reference trajectory with the actual trajectory after the control is applied. In figure 3.28 it can be seen that the maximum displacement between the reference trajectory and the actual trajectory is around $\simeq$ to 1 cm, which can be seen signaled with an arrow in figure 3.28.

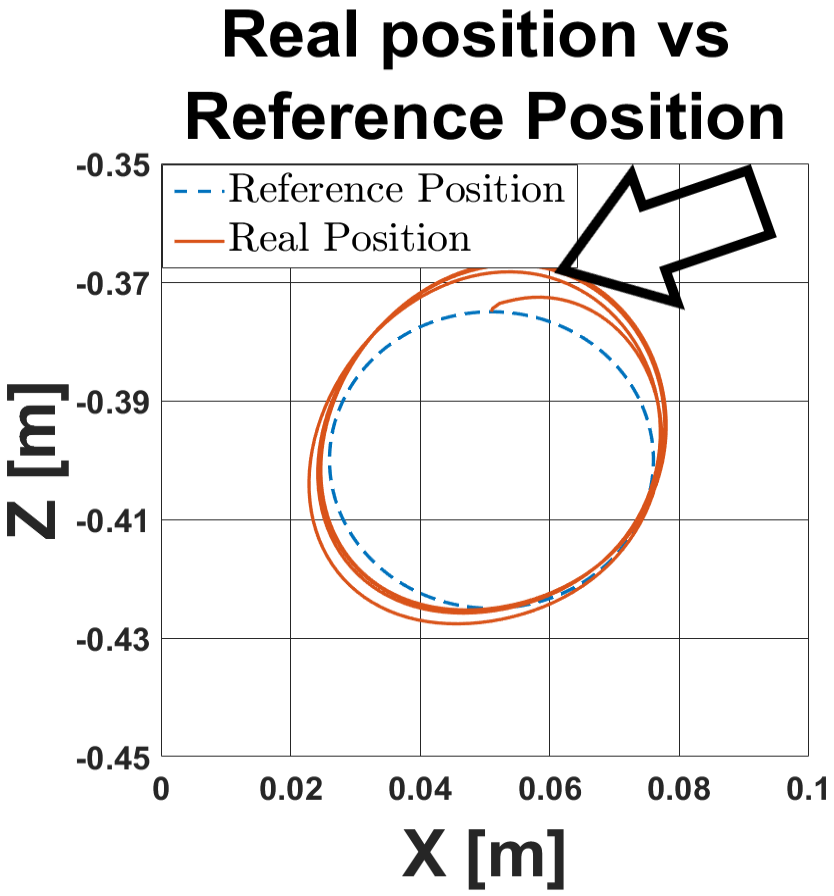


Figure 3.28: Reference position vs Real position of the end effector during simulation.

Chapter 4: Conclusions

It was observed that using the SIL simulation, a quadruped could be controlled using the Pybullet platform. It is feasible to prove the locomotion of a quadruped robot using the MPC control technique for the whole quadruped. It was seen that the controllers worked well in the SIL simulation, alternating between the phases of the locomotion of the robot, reaching stability in both phases. With the appropriate usage of the parameters of the duty factor, as well as the initial phase of the state of the leg, the basic gaits of the quadruped robot could be visualized using the SIL technique.

In the case of the Simulink simulation, it is important to mention that to have a proper model of the leg, the most appropriate approach to obtain the model is with the usage of the Euler-Lagrange method. It is observed that to avoid singularities it is important to command for a proper trajectory that avoids these singularities because if a matrix involving the control becomes non-invertible it will lead to a singularity and will lose the track of the control. After applying the controller to the quadruped leg, it was observed that it was more difficult to track the reference trajectory as the reference velocity increased, so the gains of the control needed to be adjusted more precisely to track the reference trajectory. It was observed that if there is given a reference trajectory with a sudden change, it will generate a large acceleration and it is more difficult to keep track of the control, so to have a proper track of the control, a trajectory with smooth changes must be followed to have a good track of the control and for the trajectory.

It is important to mention that to test the full performance of the locomotion of a leg it's important to design and implement another controller for the stance phase of the leg, and to alternate both controllers to see the full performance. With the usage of both controllers, they could be implemented in other walking robots such as humanoid robots.

Chapter 5: Future Work

There are many opportunity areas in which it is possible to prove the designed controllers in a real robot. In order to do this, there are some steps in order to make this possible:

- Design and implement a controller for the stance phase, alternating the controllers for the stance phase and the swing phase.
- Design a real quadruped robot, that is going to be rendered in a platform using the SIL technique.
- Once the quadruped is simulated, the designed controllers can be tested in the quadruped using a simulator like Pybullet.
- After being tested in simulation, the next step is to test the controllers in a real robot leg, in order to test the motors of the robot leg.
- After being tested in the motors of the legs, the final step is to test it in the whole robot, in order to prove the stability of the controllers in a real quadruped.
- After having good results in the real robot, it will be possible to make publications to international journals and congress papers.

Bibliography

[1] Gerardo Bledt. *Policy regularized model predictive control framework for robust legged locomotion*. PhD thesis, Massachusetts Institute of Technology, 2018.

[2] Petr Chalupa and Vladimír Bobál. Modelling and predictive control of inverted pendulum. In *22nd European Conference on Modelling and Simulation*, pages 531–537, 2008.

[3] Guangrong Chen, Sheng Guo, Bowen Hou, and Junzheng Wang. Virtual model control for quadruped robots. *IEEE Access*, 8:140736–140751, 2020.

[4] Xiang Chen, Meranda Salem, Tuhin Das, and Xiaoqun Chen. Real time software-in-the-loop simulation for control performance validation. *Simulation*, 84(8-9):457–471, 2008.

[5] Peter I Corke. A simple and systematic approach to assigning denavit–hartenberg parameters. *IEEE transactions on robotics*, 23(3):590–594, 2007.

[6] Jared Di Carlo, Patrick M. Wensing, Benjamin Katz, Gerardo Bledt, and Sangbae Kim. Dynamic locomotion in the mit cheetah 3 through convex model-predictive control. In *2018 IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS)*, pages 1–9, 2018.

[7] Boston Dynamics. Spot. `https://www.bostondynamics.com/spot`, 2020.

[8] Péter Fankhauser and Marco Hutter. Anymal: a unique quadruped robot conquering harsh environments. *Research Features*, (126):54–57, 2018.

[9] Oliver Jones. Anatomical terms of movement. `https://teachmeanatomy.info/the-basics/anatomical-terminology/terms-of-movement/`, 2020.

[10] B. Katz, J. Carlo, and Sangbae Kim. Mini cheetah: A platform for pushing the limits of dynamic quadruped control. *2019 International Conference on Robotics and Automation (ICRA)*, pages 6295–6301, 2019.

[11] Donghyun Kim, Jared Di Carlo, Benjamin Katz, Gerardo Bledt, and Sangbae Kim. Highly dynamic quadruped locomotion via whole-body impulse control and model predictive control. *arXiv preprint arXiv:1909.06586*, 2019.

[12] KW Lilly and DE Orin. O (n) recursive algorithm for the operational space inertia matrix of a robot manipulator. *IFAC Proceedings Volumes*, 23(8):275–279, 1990.

[13] MachineDesign. What's the difference between abduction and adduction? `https://www.machinedesign.com/markets/medical/article/21831782/whats-the-difference-between-abduction-and-adduction-biomechanics`, 2014.

[14] Manfred Morari and Jay H Lee. Model predictive control: past, present and future. *Computers & Chemical Engineering*, 23(4-5):667–682, 1999.

[15] M. Raibert, Kevin Blankespoor, G. Nelson, and R. Playter. Bigdog, the rough-terrain quadruped robot. *IFAC Proceedings Volumes*, 41:10822–10825, 2008.

[16] Xue Wen Rong, Yi Bin Li, Jiu Hong Ruan, and Hong Jun Song. Kinematics analysis and simulation of a quadruped robot. In *Applied Mechanics and Materials*, volume 26, pages 517–522. Trans Tech Publ, 2010.

[17] C Semini, N G Tsagarakis, E Guglielmino, M Focchi, F Cannella, and D G Caldwell. Design of hyq – a hydraulically and electrically actuated quadruped robot. *Proceedings of the Institution of Mechanical Engineers, Part I: Journal of Systems and Control Engineering*, 225(6):831–849, 2011.

[18] B. Stellato, G. Banjac, P. Goulart, A. Bemporad, and S. Boyd. OSQP: an operator splitting solver for quadratic programs. *Mathematical Programming Computation*, 12(4):637–672, 2020.

[19] Unitree. Laikago. `http://www.unitree.cc/e/action/ShowInfo.php?classid=6&id=1`, 2020.

[20] Xingming Wu, Long Teng, Weihai Chen, Guanjiao Ren, Yan Jin, and gwei Li. Cpgs with continuous adjustment of phase difference for locomotion control. *International Journal of Advanced Robotic Systems*, 10(6):269, 2013.

[21] Xianpeng Zhang, Lin Lang, Jian Wang, and Hongxu Ma. The quadruped robot locomotion based on force control. In *The 27th Chinese Control and Decision Conference (2015 CCDC)*, pages 5440–5445. IEEE, 2015.

Appendix

Appendix A

Developed transformation matrix involving the three angles

$$T_{i-1}^{i} = T_z(\alpha)T_y(\phi)T_x(\theta) = \begin{bmatrix} T_{11} & T_{12} & T_{13} & T_{14} \\ T_{21} & T_{22} & T_{23} & T_{24} \\ T_{31} & T_{32} & T_{33} & T_{34} \\ T_{41} & T_{42} & T_{43} & T_{44} \end{bmatrix}$$

Where:

$T_{11} = \cos(\phi)\ \cos(\theta)$

$T_{12} = \sin(\alpha)\ \cos(\theta)\ \sin(\phi) - \cos(\alpha)\ \sin(\theta)$

$T_{13} = \sin(\alpha)\ \sin(\theta) + \cos(\alpha)\ \cos(\theta)\ \sin(\phi)$

$T_{14} = a\ \cos(\phi)\ \cos(\theta) - c\ \sin(\theta)$

$T_{21} = \cos(\phi)\ \sin(\theta)$

$T_{22} = \cos(\alpha)\ \cos(\theta) + \sin(\alpha)\ \sin(\phi)\ \sin(\theta)$

$T_{23} = \cos(\alpha)\ \sin(\phi)\ \sin(\theta) - \sin(\alpha)\ \cos(\theta)$

$T_{24} = c\ \cos(\theta) + a\ \cos(\phi)\ \sin(\theta)$

$T_{31} = -\sin(\phi)$

$T_{32} = \cos(\phi)\ \sin(\alpha)$

$T_{33} = \cos(\alpha)\ \cos(\phi)$

$T_{34} = d - a\ \sin(\phi)$

$T_{41} = 0$

$T_{42} = 0$

$T_{43} = 0$

$T_{44} = 1$

Appendix B

Kinematic Energy of the three links

For the hip the Kinematic energy can be calculated with the following expression:

$$T_1 = \frac{\dot{\theta}_1^2 \left(m_1 {L_1}^2 + 4\,\mathrm{I_{xx1}}\right)}{8}$$

For the thigh the Kinematic energy is calculated with the following expression:

$$T_2 = \frac{\mathrm{I_{xx2}}\,\dot{\theta}_1^2}{2} + \frac{\mathrm{Iyy_2}\,\dot{\theta}_2^2}{2} + \frac{m_2(\dot{\theta}_1(L_1 cos(\theta_1) + \frac{L_2 cos(\theta_1) cos(\theta_2)}{2}) - \frac{L_2 \dot{\theta}_2 sin(\theta_1) sin(\theta_2)}{2})^2}{2}$$
$$+ \frac{m_2((\dot{\theta}_1(L_1 sin(\theta_1) + \frac{L_2 cos(\theta_2) sin(\theta_1)}{2}) + \frac{L_2 \dot{\theta}_2 cos(\theta_1) sin(\theta_2)}{2})^2 + \frac{L_2^2 \dot{\theta}_2^2 cos(\theta_2)^2}{4})}{2}$$

For the calf, the kinematic energy is calculated with the following expression:

$$T_3 = \frac{\mathrm{I_{xx3}}\,\dot{\theta}_1^2}{2} + \frac{\mathrm{Iyy_3}\,\dot{\theta}_2^2}{2} + \frac{\mathrm{Iyy_3}\,\dot{\theta}_3^2}{2} + \frac{{L_1}^2\, m_3\,\dot{\theta}_1^2}{2} + \frac{{L_2}^2\, m_3\,\dot{\theta}_1^2}{4} + \frac{{L_2}^2\, m_3\,\dot{\theta}_2^2}{2} + \frac{{L_3}^2\, m_3\,\dot{\theta}_1^2}{16} + \frac{{L_3}^2\, m_3\,\dot{\theta}_2^2}{8} +$$
$$\frac{{L_3}^2\, m_3\,\dot{\theta}_3^2}{8} + \mathrm{Iyy_3}\,\dot{\theta}_2\,\dot{\theta}_3 + \frac{{L_3}^2\, m_3\,\dot{\theta}_2\,\dot{\theta}_3}{4} + \frac{{L_2}^2\, m_3\,\dot{\theta}_1^2\,\cos(2\,\theta_2)}{4} + \frac{{L_3}^2\, m_3\,\dot{\theta}_1^2\,\cos(2\,\theta_2 + 2\,\theta_3)}{16} + L_1\,L_2\,m_3\,\dot{\theta}_1^2\,\cos(\theta_2) +$$
$$\frac{L_2\,L_3\,m_3\,\dot{\theta}_1^2\,\cos(\theta_3)}{4} + \frac{L_2\,L_3\,m_3\,\dot{\theta}_2^2\,\cos(\theta_3)}{2} + \frac{L_2\,L_3\,m_3\,\dot{\theta}_1^2\,\cos(2\,\theta_2 + \theta_3)}{4} + \frac{L_1\,L_3\,m_3\,\dot{\theta}_1^2\,\cos(\theta_2 + \theta_3)}{2} + \frac{L_2\,L_3\,m_3\,\dot{\theta}_2\,\dot{\theta}_3\,\cos(\theta_3)}{2}$$

Appendix C

Potential Energy of the three links

The contribution of the hip, the potential energy can be calculated with the following expression:

$$U_1 = \frac{L_1\,g\,m_1\,\cos(\theta_1)}{2}$$

For the contribution of the thigh, the potential energy can be calculated with the following expression:

$$U_2 = -g\,m_2\,\left(L_1\,\cos(\theta_1) + \frac{L_2\,\cos(\theta_1)\,\cos(\theta_2)}{2}\right)$$

For the contribution of the calf, the potential energy can be calculated with the following expression:

$$U_3 = -\frac{g\,m_3\,\cos(\theta_1)\,(2\,L_1 + L_3\,\cos(\theta_2 + \theta_3) + 2\,L_2\,\cos(\theta_2))}{2}$$

Appendix D

M matrix of the dynamic model

$$M(q) = \begin{bmatrix} m_{11} & m_{12} & m_{13} \\ m_{21} & m_{22} & m_{23} \\ m_{31} & m_{32} & m_{33} \end{bmatrix}$$

With:

$m_{11} = I_{xx1} + I_{xx2} + I_{xx3} + (L_1^2 * m_1)/4 + L_1^2 * m_2 + L_1^2 * m_3 + (L_2^2 * m_2)/8 + (L_2^2 * m_3)/2 + (L_3^2 * m_3)/8 + (L_2^2 * m_2 * cos(2 * \theta_2))/8 + (L_2^2 * m_3 * cos(2 * \theta_2))/2 + (L_3^2 * m_3 * cos(2 * \theta_2 + 2 * \theta_3))/8 + L_1 * L_3 * m_3 * cos(\theta_2 + \theta_3) + L_1 * L_2 * m_2 * cos(\theta_2) + 2 * L_1 * L_2 * m_3 * cos(\theta_2) + (L_2 * L_3 * m_3 * cos(\theta_3))/2 + (L_2 * L_3 * m_3 * cos(2 * \theta_2 + \theta_3))/2$

$m_{12} = 0$

$m_{13} = 0$

$m_{21} = 0$

$m_{22} = I_{yy2} + I_{yy3} + (L_2^2 * m_2)/4 + L_2^2 * m_3 + (L_3^2 * m_3)/4 + L_2 * L_3 * m_3 * cos(\theta_3)$

$m_{23} = (m_3 * L_3^2)/4 + (L_2 * m_3 * cos(\theta_3) * L_3)/2 + I_{yy3}$

$m_{31} = 0$

$m_{32} = (m_3 * L_3^2)/4 + (L_2 * m_3 * cos(\theta_3) * L_3)/2 + I_{yy3}$

$m_{33} = (m_3 * L_3^2)/4 + I_{yy3}$

Appendix E

C matrix of the dynamic model

$$C(q,\dot{q}) = \begin{bmatrix} c_{11} & c_{12} & c_{13} \\ c_{21} & c_{22} & c_{23} \\ c_{31} & c_{32} & c_{33} \end{bmatrix}$$

With:

$c_{11} = -\dot{\theta}_2 * ((L_2^2 * m_2 * sin(2*\theta_2))/8 + (L_2^2 * m_3 * sin(2*\theta_2))/2 + (L_3^2 * m_3 * sin(2*\theta_2 + 2*\theta_3))/8 + (L_1 * L_3 * m_3 * sin(\theta_2 + \theta_3))/2 + (L_1 * L_2 * m_2 * sin(\theta_2))/2 + L_1 * L_2 * m_3 * sin(\theta_2) + (L_2 * L_3 * m_3 * sin(2*\theta_2 + \theta_3))/2) - (L_3 * m_3 * \dot{\theta}_3 * (L_3 * sin(2*\theta_2 + 2*\theta_3) + 4 * L_1 * sin(\theta_2 + \theta_3) + 2 * L_2 * sin(\theta_3) + 2 * L_2 * sin(2*\theta_2 + \theta_3)))/8$

$c_{12} = -\dot{\theta}_1 * ((L_2^2 * m_2 * sin(2*\theta_2))/8 + (L_2^2 * m_3 * sin(2*\theta_2))/2 + (L_3^2 * m_3 * sin(2*\theta_2 + 2*\theta_3))/8 + (L_1 * L_3 * m_3 * sin(\theta_2 + \theta_3))/2 + (L_1 * L_2 * m_2 * sin(\theta_2))/2 + L_1 * L_2 * m_3 * sin(\theta_2) + (L_2 * L_3 * m_3 * sin(2*\theta_2 + \theta_3))/2)$

$c_{13} = -(L_3 * m_3 * \dot{\theta}_1 * (L_3 * sin(2*\theta_2 + 2*\theta_3) + 4 * L_1 * sin(\theta_2 + \theta_3) + 2 * L_2 * sin(\theta_3) + 2 * L_2 * sin(2*\theta_2 + \theta_3)))/8$

$c_{21} = \dot{\theta}_1 * ((L_2^2 * m_2 * sin(2*\theta_2))/8 + (L_2^2 * m_3 * sin(2*\theta_2))/2 + (L_3^2 * m_3 * sin(2*\theta_2 + 2*\theta_3))/8 + (L_1 * L_3 * m_3 * sin(\theta_2 + \theta_3))/2 + (L_1 * L_2 * m_2 * sin(\theta_2))/2 + L_1 * L_2 * m_3 * sin(\theta_2) + (L_2 * L_3 * m_3 * sin(2*\theta_2 + \theta_3))/2)$

$c_{22} = -(L_2 * L_3 * m_3 * \dot{\theta}_3 * sin(\theta_3))/2$

$c_{23} = -(L_2 * L_3 * m_3 * sin(\theta_3) * (\dot{\theta}_2 + \dot{\theta}_3))/2$

$c_{31} = (L_3 * m_3 * \dot{\theta}_1 * (L_3 * sin(2*\theta_2 + 2*\theta_3) + 4 * L_1 * sin(\theta_2 + \theta_3) + 2 * L_2 * sin(\theta_3) + 2 * L_2 * sin(2*\theta_2 + \theta_3)))/8$

$c_{32} = (L_2 * L_3 * m_3 * \dot{\theta}_2 * sin(\theta_3))/2$

$c_{33} = 0$

Appendix F

G matrix of the dynamic model

$$G(q) = \begin{bmatrix} G_1 \\ G_2 \\ G_3 \end{bmatrix}$$

With:

$G_1 = (g * sin(\theta_1) * (L_1 * m_1 + 2 * L_1 * m_2 + 2 * L_1 * m_3 + L_3 * m_3 * cos(\theta_2 + \theta_3) + L_2 * m_2 * cos(\theta_2) + 2 * L_2 * m_3 * cos(\theta_2)))/2$

$G_2 = (g * cos(\theta_1) * (L_2 * m_2 * sin(\theta_2) + 2 * L_2 * m_3 * sin(\theta_2) + L_3 * m_3 * sin(\theta_2 + \theta_3)))/2$

$G_3 = (L_3 * g * m_3 * sin(\theta_2 + \theta_3) * cos(\theta_1))/2$

Appendix G

Developed dynamic model

$$\frac{d}{dt}\begin{bmatrix}\hat{\phi}\\ \hat{\theta}\\ \hat{\psi}\\ \hat{x}\\ \hat{y}\\ \hat{z}\\ \hat{w}_x\\ \hat{w}_y\\ \hat{w}_z\\ \dot{\hat{x}}\\ \dot{\hat{y}}\\ \dot{\hat{z}}\\ \hat{g}\end{bmatrix} = \begin{bmatrix}0&0&0&0&0&0&cos(\psi)&-sin(\psi)&0&0&0&0&0\\ 0&0&0&0&0&0&sin(\psi)&cos(\psi)&0&0&0&0&0\\ 0&0&0&0&0&0&0&0&1&0&0&0&0\\ 0&0&0&0&0&0&0&0&0&1&0&0&0\\ 0&0&0&0&0&0&0&0&0&0&1&0&0\\ 0&0&0&0&0&0&0&0&0&0&0&1&0\\ 0&0&0&0&0&0&0&0&0&0&0&0&0\\ 0&0&0&0&0&0&0&0&0&0&0&0&0\\ 0&0&0&0&0&0&0&0&0&0&0&0&0\\ 0&0&0&0&0&0&0&0&0&0&0&0&0\\ 0&0&0&0&0&0&0&0&0&0&0&0&0\\ 0&0&0&0&0&0&0&0&0&0&0&0&1\\ 0&0&0&0&0&0&0&0&0&0&0&0&0\end{bmatrix}\begin{bmatrix}\hat{\phi}\\ \hat{\theta}\\ \hat{\psi}\\ \hat{x}\\ \hat{y}\\ \hat{z}\\ \hat{w}_x\\ \hat{w}_y\\ \hat{w}_z\\ \dot{\hat{x}}\\ \dot{\hat{y}}\\ \dot{\hat{z}}\\ \hat{g}\end{bmatrix} +$$

$$\begin{bmatrix}0&0&0&...&0&0&0\\ 0&0&0&...&0&0&0\\ 0&0&0&...&0&0&0\\ 0&0&0&...&0&0&0\\ 0&0&0&...&0&0&0\\ 0&0&0&...&0&0&0\\ -\frac{c_1\sigma_2}{\sigma_1}&-\frac{c_1\sigma_3}{\sigma_1}&-\frac{a_1\sigma_2}{\sigma_1}+\frac{b_1\sigma_3}{\sigma_1}&...&-\frac{c_n\sigma_2}{\sigma_1}&-\frac{c_n\sigma_3}{\sigma_1}&-\frac{a_n\sigma_2}{\sigma_1}+\frac{b_n\sigma_3}{\sigma_1}\\ \frac{c_1\sigma_4}{\sigma_1}&\frac{c_1\sigma_2}{\sigma_1}&-\frac{b_1\sigma_2}{\sigma_1}-\frac{a_1\sigma_4}{\sigma_1}&...&-\frac{c_n\sigma_4}{\sigma_1}&\frac{c_n\sigma_2}{\sigma_1}&-\frac{b_n\sigma_2}{\sigma_1}-\frac{a_n\sigma_4}{\sigma_1}\\ -\frac{b_1}{I_{zz}}&\frac{a_1}{I_{zz}}&0&...&-\frac{b_n}{I_{zz}}&\frac{a_n}{I_{zz}}&0\\ \frac{1}{m}&0&0&...&\frac{1}{m}&0&0\\ 0&\frac{1}{m}&0&...&0&\frac{1}{m}&0\\ 0&0&\frac{1}{m}&...&0&0&\frac{1}{m}\\ 0&0&0&...&0&0&0\end{bmatrix}\begin{bmatrix}f_{1x}\\ f_{1y}\\ f_{1z}\\ :\\ :\\ :\\ :\\ :\\ :\\ :\\ f_{nx}\\ f_{ny}\\ f_{nz}\end{bmatrix}$$

Where:

$\sigma_1 = I_{xx}I_{yy}cos(\psi)^4 + 2I_{xx}I_{yy}cos(\psi)^2sin(\psi)^2 + I_{xx}I_{yy}sin(\psi)^4$
$\sigma_2 = I_{xx}cos(\psi)sin(\psi) - I_{yy}cos(\psi)sin(\psi)$
$\sigma_3 = I_{yy}cos(\psi)^2 + I_{xx}sin(\psi)^2$
$\sigma_4 = I_{xx}cos(\psi)^2 + I_{yy}sin(\psi)^2$

Appendix H

Developed inertia tensor viewed from the world frame multiplied by the skew matrix

$$\hat{I_w^{-1}}[r_1]\times = \begin{bmatrix} i_{11} & i_{12} & i_{13} \\ i_{21} & i_{22} & i_{23} \\ i_{31} & i_{32} & i_{33} \end{bmatrix}$$

Where:

$i_{11} = (c * sin(2 * psi) * (I_{xx} - Iyy))/2$

$i_{12} = -c * (I_{xx} - I_{xx} * sin(psi)^2 + Iyy * sin(psi)^2)$

$i_{13} = b * (I_{xx} - I_{xx} * sin(psi)^2 + Iyy * sin(psi)^2) - (a * sin(2 * psi) * (I_{xx} - Iyy))/2$

$i_{21} = c * (Iyy + I_{xx} * sin(psi)^2 - Iyy * sin(psi)^2)$

$i_{22} = -(c * sin(2 * psi) * (I_{xx} - Iyy))/2$

$i_{23} = (b * sin(2 * psi) * (I_{xx} - Iyy))/2 - a * (Iyy + I_{xx} * sin(psi)^2 - Iyy * sin(psi)^2)$

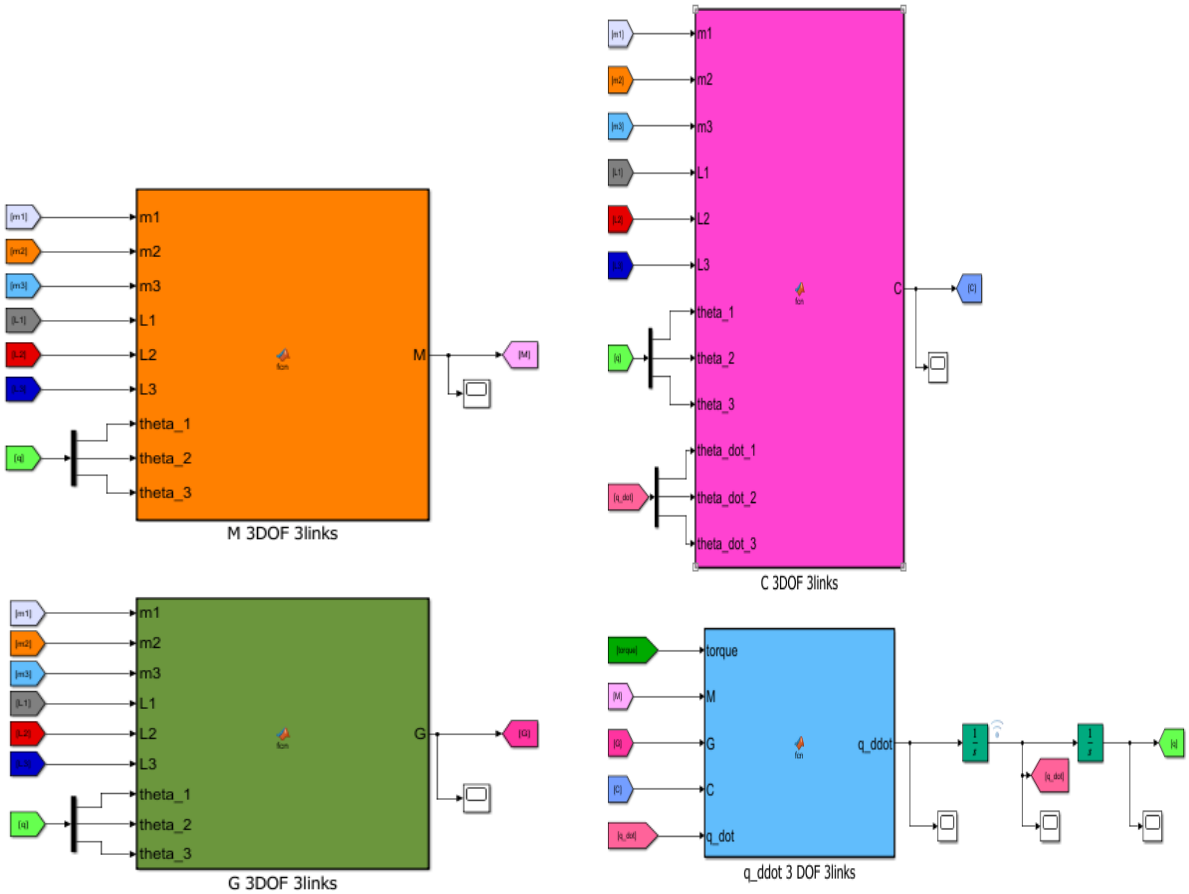
$i_{31} = Izz * b$

$i_{32} = Izz * a$

$i_{33} = 0$

Appendix I

Control of the Swing Leg Controller used in Simulink

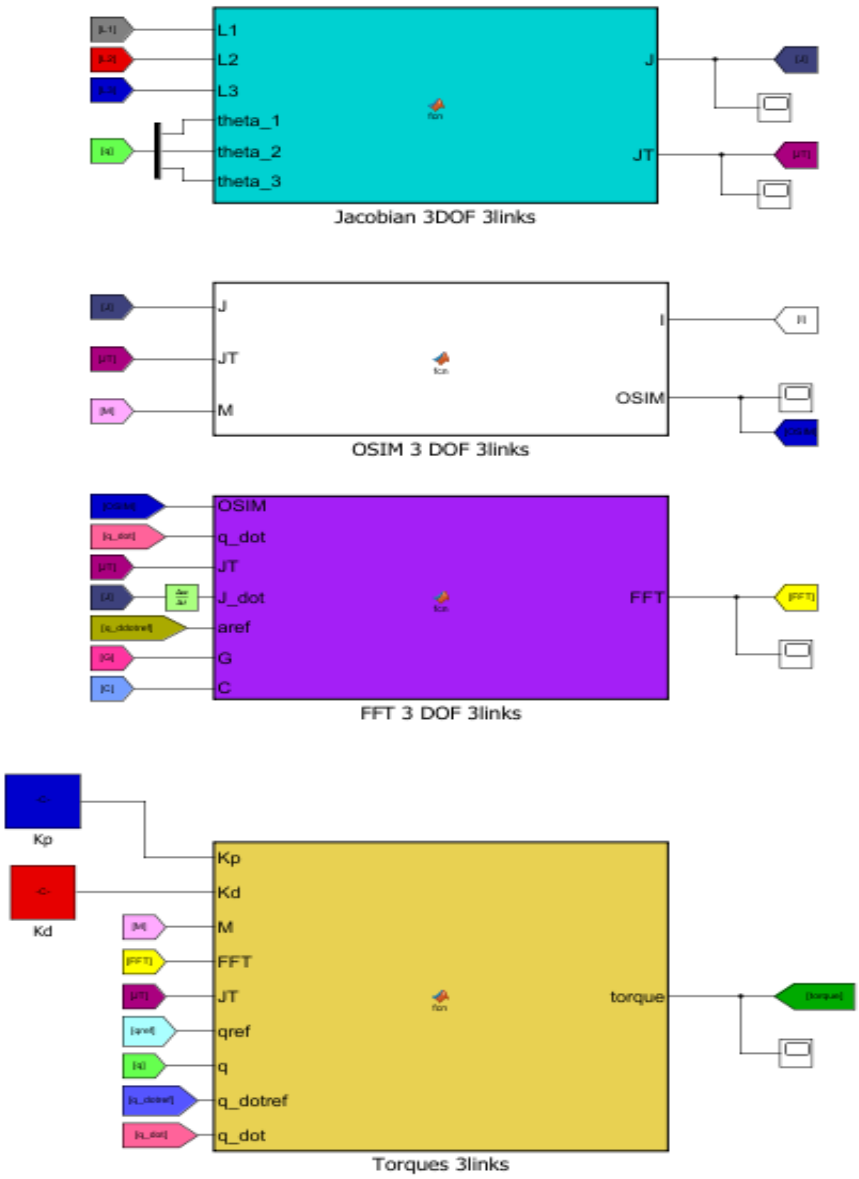


(a) (b)

Model from equation 10 implemented in the Simulink software.

Appendix J

Model of the Swing Leg Controller used in Simulink



Control implemented in the Simulink software in order to control the torques of the links.

View publication stats