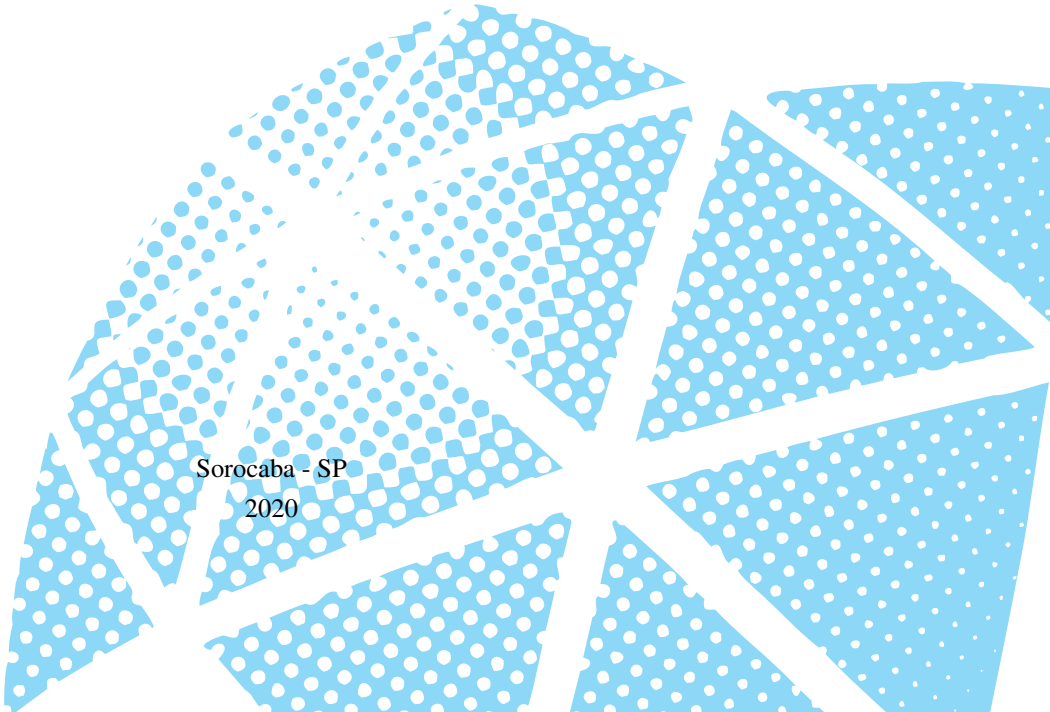
UNIVERSIDADE ESTADUAL PAULISTA
"JÚLIO DE MESQUITA FILHO"
Instituto de Ciência e Tecnologia de Sorocaba

MARIA FERNANDA TEJADA BEGAZO

A LEARNING-BASED MODEL-FREE CONTROLLER FOR DECOUPLED HUMANOID ROBOT WALKING



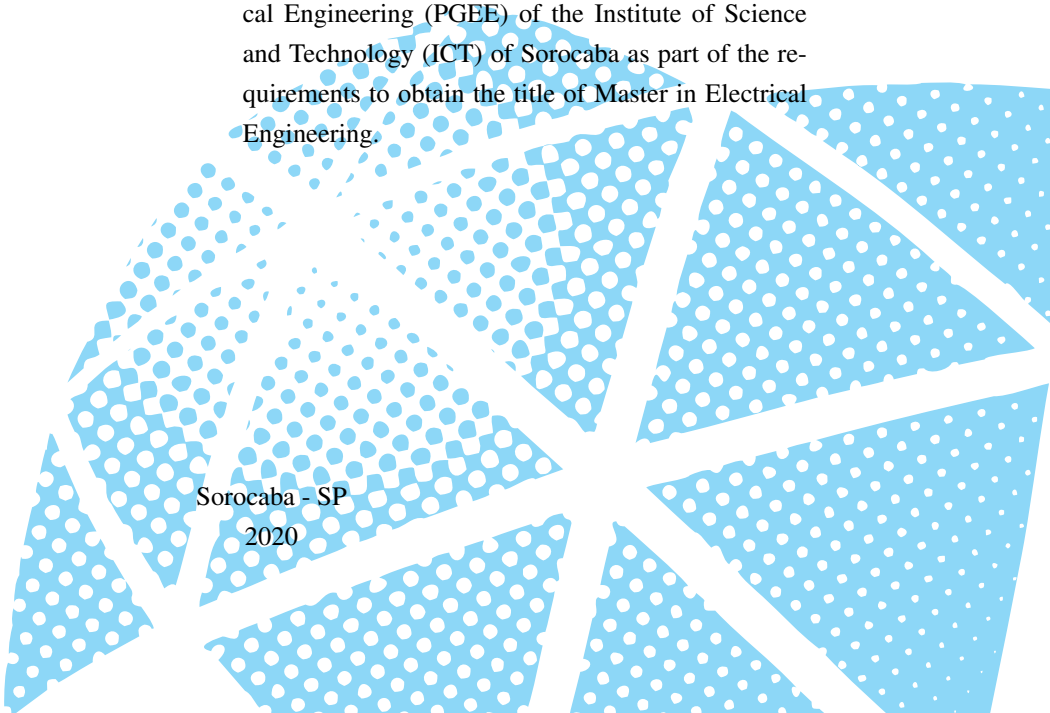
Sorocaba - SP
2020

MARIA FERNANDA TEJADA BEGAZO

A LEARNING-BASED MODEL-FREE CONTROLLER FOR DECOUPLED HUMANOID ROBOT WALKING

MASTER'S THESIS

Text submitted to the Graduate Program in Electrical Engineering (PGEE) of the Institute of Science and Technology (ICT) of Sorocaba as part of the requirements to obtain the title of Master in Electrical Engineering.



Sorocaba - SP
2020

To my advisor, for all his patience, dedication and time in the most difficult moments, and for inspiring me to look outside the bubble of the common to innovation.

B416l

Begazo, Maria Fernanda Tejada
A learning-based model-free controller for decoupled humanoid robot walking / Maria Fernanda Tejada Begazo. -- Sorocaba, 2020
94 p. : il., tabs.

Dissertação (mestrado) - Universidade Estadual Paulista (Unesp), Instituto de Ciência e Tecnologia, Sorocaba
Orientador: Alexandre da Silva Simões

1. Genetic algorithms. 2. Artificial intelligence. 3. Intelligent control systems. 4. Androids. I. Título.

Sistema de geração automática de fichas catalográficas da Unesp. Biblioteca do Instituto de Ciência e Tecnologia, Sorocaba. Dados fornecidos pelo autor(a).

Essa ficha não pode ser modificada.

UNIVERSIDADE ESTADUAL PAULISTA

Câmpus de Sorocaba

CERTIFICADO DE APROVAÇÃO

TÍTULO DA DISSERTAÇÃO: **"A learning-based model-free controller for decoupled humanoid robot waking"**

AUTORA: MARIA FERNANDA TEJADA BEGAZO
ORIENTADOR: ALEXANDRE DA SILVA SIMÕES

Aprovada como parte das exigências para obtenção do Título de Mestra em ENGENHARIA ELÉTRICA, área: Sistemas Eletrônicos pela Comissão Examinadora:

Prof. Dr. ALEXANDRE DA SILVA SIMÕES (Participaçao Virtual)
Departamento de Engenharia de Controle e Automação / Instituto de Ciência e Tecnologia - UNESP - Câmpus de Sorocaba

Professor Titular REINALDO AUGUSTO DA COSTA BIANCHI (Participaçao Virtual)
Centro Universitário Fundação Educacional Inaciana Pe. Saboia de Medeiros - FEI

Prof. Dr. MARCOS RICARDO OMENA DE ALBUQUERQUE MAXIMO (Participaçao Virtual)
Divisão de Ciência da Computação, Departamento de Metodologias de Computação / Instituto Tecnológico de Aeronáutica

Sorocaba, 30 de dezembro de 2020

Prof. Dr. Alexandre da Silva Simões
Orientador

Acknowledgements

I thank all the family, friends, teachers and employees of the Sorocaba Institute of Science and Technology, who contributed directly or indirectly to the realization of this work. In particular, I offer my thanks:

- To my parents Cristina and Armando and my brother Anthony for their love, support and encouragement;
- To my aunt Tania for the support and encouragement at all time;
- To Prof. Dr. Alexandre and Dra. Esther, for all the teaching, encouragement, trust and guidance;
- To my friends and colleagues in the laboratory who helped me directly or indirectly, especially Vitor, Nayari, Murillo and Rafael, for the help and work done together;
- To my friends Renata, Claudia, Kevin and Elizabeth for their support and encouragement.
- This study was financed in part by the Coordenação de Aperfeiçoamento de Pessoal de Nível Superior – Brasil (CAPES) – Finance Code 001.

“O sol é para todos,
mas a sombra é para quem
chega primeiro.”
Geremias Ludu

Abstract

Robotics has advanced rapidly in recent decades. Initially, it focused on automotive industry production after the second industrial revolution; it rapidly expanded to new exciting domains that prompted robots to perform tasks with tools and in environments designed for humans. However, controlling the movements of such robots is becoming more and more challenging due to the growing complexity of their mechanical structure and dynamics. This difficulty has stimulated the investigation of novel methods for robots controlling that may not depend on the kinematic or dynamic models of the robots. The Marta humanoid robot was designed and built by the LaRoCS and GASI research groups for humanoid walking investigation and present some novel concepts like small feet – instead of large ones – and a 3-DOF spherical joint in the waist that allows the design of decoupled controlling strategies. Taking advantage of the mechanical architecture of this robot, this work presents a model-free controller for humanoid robot walking composed by i) a classical controller for the waist spherical joint that focuses on robot stability, and ii) a trajectory controller for the remaining joints based on a Truncated Fourier Series (TFS) its parameters tuned by a Genetic Algorithm (GA) that focus on the implementation of a coordinated robot walking pattern. Results have shown that the decoupled robot control strategy has the potential to confer additional robustness to the bipedal robots walking while compared to traditional techniques.

Keywords: Genetic algorithms. Artificial intelligence. Intelligent control systems. Androids.

Resumo

A robótica avançou rapidamente nas últimas décadas. Inicialmente focada na produção industrial de automóveis depois da segunda revolução industrial; ela rapidamente se expandiu para novos e excitantes domínios que demandaram dos robôs a realização de tarefas com ferramentas e em ambientes projetados para seres humanos. No entanto, controlar o movimentos desses robôs está se tornando cada vez mais desafiador devido ao crescimento da complexidade de sua estrutura mecânica e dinâmica. Essa dificuldade tem estimulado a investigação de novos métodos para o controle de robôs que possam não depender do modelo cinemático ou dinâmico dos robôs. A robô humanóide Marta foi projetada e construída pelos grupos de pesquisa LaRoCS e GASI para a realização de investigações de caminhada de robôs humanóides, e apresenta alguns novos conceitos tais como os pés pequenos - ao invés de pés grandes - e uma junta esférica de 3 DOF na cintura que permite o projeto de estratégias de controle desacopladas. Aproveitando a arquitetura mecânica deste robô, este trabalho apresenta um controlador livre de modelo para a caminhada de robôs humanóides composta por i) um controlador clássico para a junta esférica da cintura que foca na estabilidade do robô e ii) um controlador de trajetória para as demais juntas baseado em uma Série Truncada de Fourier (TFS) com seus parâmetros ajustados por um algoritmo genético (GA) que foca na implementação de um padrão de caminhada coordenada do robô. Os resultados mostraram que a estratégia de controle descomposta tem potencial para conferir robustez adicional à caminhada de robôs bípedes em relação a técnicas convencionais.

Palavras-chave: Algoritmos genéticos. Inteligência artificial. Sistemas inteligentes de controle. Andróides.

List of Figures

List of Tables

List of Abbreviations and Acronyms

3D-FPBIPM 3D force-pattern based inverted pendulum model

AD *Anno Domini*

AI Artificial Intelligence

BC Before Christ

CPU Central Processing Unity

CPG Central Pattern Generator

CoM Center of Mass

DOF Degrees Of Freedom

EA Evolutionary Algorithms

GASI Automation and Integrated Systems Group

GA Genetic Algorithm

GUI Graphical User Interface

IMU Inertial Measurement Unit

IPM Inverted Pendulum Model

LaRoCS Laboratory of Robotics and Cognitive Systems

LIPM Linear Inverted Pendulum Model

MDP Markov Decision Process

ML Machine Learning

NUC Next Unity of Computing

OS Operational System

PID Proportional-Integral-Derivative

RL Reinforcement Learning

TLIPM Triple Linear Inverted Pendulum Model

TFS Truncated Fourier Series

UNESP *Universidade Estuadual Paulista*

UNICAMP *Universidade Estadual de Campinas*

VLIPM Virtual Linear Inverted Pendulum Model

V-REP Virtual Robotic Experimentation Platform

ZMP Zero-Moment Point

List of Symbols

$\sum_w$	World coordinates system of the humanoid robot
$\sum_i$	Local coordinates system of the j-th joint
p_i	Vector position of the j-th joint in the $\sum_w$
p_i^j	Vector position of i-th joint viewed from the local coordinates system of the j-th joint ($\sum_j$)
r	Vector resultant of the p_i and p_j
e_{ic}	The c axis of the coordinate system of $\sum_i$
$R_c(\theta)$	Represent the θ of rotation around axis c
R_i	Rotation matrix of the i-th joint in the $\sum_w$
R_nm	The position n, m in the rotation matrix
T_j	The homogeneous transformation matrix of the j-th joint in the $\sum_w$
T_j^i	The homogeneous transformation matrix of the j-th joint viewed the local coordinates system of the i-th joint
c_*	$\cos(*)$
s_*	$\sin(*)$
b_j	Vector between the $\sum_w$ and $\sum_j$
q_i	Angle of rotation the i-th joint
M	Total mass of the robot
$\mathscr{C}$	Robot's center of mass
m_i	The mass of the i-th joint
$\mathfrak{p}_\mathfrak{i}$	Position of the i-th mass point
$\mathscr{P}$	Momentum of the robot
$\mathfrak{L}$	Angular momentum of the robot
$\mathfrak{L}^r$	Angular momentum about the referent point r
$\dot{\mathfrak{L}}$	Rotational movement about the origin of the joint
f	External force applied to the robot that is not gravity
f_{all}	All external forces applied to the robot
τ	External torque applied to the robot
τ_{all}	All external torque applied to the robot
g	Gravitation force
f_{ij}^{int}	Force applied to the mass of the i-th joint from j-th
f_{ij}^{ext}	Force external applied to the mass of the i-th joint from j-th
v_i	Velocity of the i-th joint
ω	Angular velocity
I	Coefficient matrix
l_c	Length on c axis

w_r	Reference signal of the control system
w_d	External disturbance of the control system
w_n	Sensor noise of the control system
$e(*)$	Error function
K_p	Proportional gain of the PID system
K_i	Factor integral of the PID system
K_d	Factor derivative of the PID system
T_i	Integral time of the PID system
T_d	Derivative time of the PID system
L_z	Delay time of the Ziegler-Nichols based on step response
T_z	Time constant of the Ziegler-Nichols based on step response
K_u	Critical gain of the Ziegler-Nichols based on frequency response
T_u	Critical oscillatory period of the Ziegler-Nichols based on frequency response
$p(i)$	Probability of an i-th individual of the Genetic Algorithm
$f(i)$	Fitness value of an i-th individual of the Genetic Algorithm
$N(0,\sigma)$	Gaussian distribution
π_t	Policy on the t time of the Reinforcement Learning
π_*	Optimal policy of the Reinforcement Learning
S_t	State on the t time of the Reinforcement Learning
A_t	Action on the t time of the Reinforcement Learning
R_t	Reward on the t time of the Reinforcement Learning
$V_\pi(*)$	Value function of the Reinforcement Learning
$p(*)$	Probability distribution of the Reinforcement Learning
G_t	Commutative reward in the t time of the Reinforcement Learning
α	Learning rate factor of the Reinforcement Learning
γ	Discount factor of the Reinforcement Learning

Contents

1 INTRODUCTION

"I've never expected a miracle. I will get things done myself."
(Guts character)
Berserk

Humans have dreamed of creating skillful and intelligent machines, which could help them in tedious or dangerous activities for a long time. This dream began to come true at the end of the second industrial revolution, where machines were typically used in repetitive tasks that harmed human health, especially in automobile factories. These early machines were called manipulator's arms, opening the way for exploration of the field of robotics. Over the past few decades, robotics has spread to many areas allowing machines to perform activities outside of industries in medicine, customer service, military, exploration, rescue, education, etc.

Even considering the dissemination of robotics nowadays, interacting with environments and tools designed for humans still represents a significant challenge for robots. In this context, **humanoid robots** – robots that resemble a human appearance – occupy a prominent place due to their ability to adapt easier in these challenging scenarios. Since such robots must have the ability to move from one point to another as close as possible to humans, **legged robots** have attracted a particular interest of the researchers. This context justifies the interest in researching new controllers for humanoid robots that can expand their capabilities in the bipedal walking task.

The control of the humanoid robot walking is usually divided into two approaches: i) the quasi-static walk and ii) the dynamic walk. The quasi-static walk focuses on a **kinematic modeling** of the robot that is typically based on the manipulator arms theory. The main problem with this approach is the reduction in walking speed since the robot must perform movements slow enough to depreciate the robot's dynamics. On the other hand, **dynamic walk** modeling focuses on obtaining the stability of the robot with the dynamics of the environment. If the controller is based on the robot dynamic model, it typically focuses on the internal and external forces that act on the humanoid during the movement. This technique comes up against the complexity problem of modeling robot dynamics that is not trivial. There are two approaches in this dynamic walk; the **based on model** approach requires the robot dynamic model in the environment. On the other hand, **model-free** is not necessary the robot dynamic, since it empirically discovers the dynamic model. This last approach allows us to work with various types of humanoids with minor modifications to the controller.

Recently, the working group – the Automation and Integrated Systems Group (GASI) at Unesp, and Laboratory of Robotics and Cognitive Systems (LaRoCS) at Unicamp – has designed and built-in physical and simulated environments – a humanoid robot called Marta with twenty-five DOF designed for walking research, which presents at least three particular concepts when compared to most commercial standard robots: i) the small dimension of the feet, ii) the articulation of the toe, and iii) a 3-DOF spherical joint placed in robot waist that allows the design of decoupled controlling strategies. These constructive particularities bring new challenges and possibilities to the humanoid dynamic walk control for maintaining the robot's balance during the walk.

Taking advantage of the mechanical architecture of this robot, this work proposes a model-free controller for humanoid robot walking composed by i) a classical controller for the wast spherical joint that focuses on robot stability, and ii) a trajectory controller for the remaining joints based on a Truncated Fourier Series (TFS) with its parameters tuned by a Genetic Algorithm (GA) that focus on the implementation of a coordinated robot walking pattern.

1.1 OBJECTIVE

1.1.1 MAIN OBJECTIVE

The main goal in this work is to investigate the possibility of controlling the gait of a bipedal robot – particularly the Marta humanoid robot – using a strategy based on the decoupling of the robot joints into two sets: i) waist and ii) remaining DOFs, with each set guided by a distinct controller. A first controller is designed for the 3-DOF spherical joint placed in the robot waist focusing on improving the humanoid's stability during the walk. Simultaneously, a second model-free controller generates a set of equations for controlling the remaining joints' trajectories based on the Truncated Fourier Series (TFS), with its parameters properly tuned through a machine learning process based on a Genetic Algorithm (GA).

1.1.2 SPECIFIC OBJECTIVES

The specific objectives of this work are:

- To implement the computational framework that can allow the work with the simulated version of the Marta humanoid robot;
- To implement a classical controller on the waist of Marta robot and to evaluate the stability of the robot during walk;
- To implement the optimization of the TFS parameters with a Genetic Algorithm for controlling the DOFs of the Marta robot;
- To compare the walking pattern achieved with the implemented algorithm with others available in the literature.

1.2 WORK ORGANIZATION

This work has been organized as follows:

- Chapter 2 details the theoretical foundations of humanoid robots. The section focus on the mathematical models available in the literature to describe a humanoid robot;
- Chapter 3 presents the machine learning control foundations. We explain control theory and Ziegler-Nichols tuning. Also, We detail machine learning in the approaches of the evolution algorithm theory and reinforcement learning;
- Chapter 4 presents a review of the state of the art on biped walking controllers. Two types of controllers are examined: those based on models and those that are model-free;
- Chapter 5 describes the proposed approach in more details;
- Chapter 6 examines the materials that we propose to use in this work: the Marta humanoid robot and the simulator that will be used in the experiments and for the evaluation of the algorithm;
- Chapter 7 presents experiments performed, results and discussions;
- Chapter 8 presents the conclusions in this work and directs future works.

2 FUNDAMENTALS OF HUMANOID ROBOTS

"Once you eliminate the impossible, whatever remains, no matter how improbable, must be the truth."
(Sherlock Holmes)
Arthur Conan Doyle

2.1 HUMANOID ROBOTS

Humanoid robotics is the field of robotics that deals with robots that resemble the human body structure. The first mention of a physical mechanism that can be associated with the concept of robot goes back to Greek mythology with the legend of Talos, a giant bronze built by Zeus to protect the island of Crete from invaders (1500 BC) (SMITH, 1849). This concept is also present in the statues of the Egyptian oracle machines that symbolized a deity able to answer questions. It is similar to a chat-bot with a physical body (HRASTINSKI et al., 2019).

The first automated mechanical device capable of measuring the time by regulating the flow of liquids from one vessel to another, known as the *clepsydra*, existed in Babylon, Egypt, and Persia (TURNER, 1984). This mechanical device was improved with a water and levers system by Hero of Alexandria. He had various mechanisms such as birds that drank water, statues that served wine, automatic doors, in the other (10 AD - 70 AD). The most relevant machine he made was "The Automated Theater"which represents the Trojan War through a mechanical puppet theater.

The development of automaton continued. In Asia, the *Karakuri* is a wooden mechanism that can represent traditional myths and legends (YOKOTA, 2009). In the eighteen century, Pierre Jaquet-Droz built three sophisticated automatons: the musician, the drawer, and the writer (VOSKUHL, 2007).

In 1920 the writer Karel Capek created the term **robot** in his science fiction work "Rossum's Universal Robots (R.U.R.)"(CAPEK, 2004). This term comes from the Czech word *robota* that means forced labor. Furthermore, Capek's work is the starting point for the creation of stories with artificial human bodies.

In the twenty century, the development of integrated circuits, digital computers, and miniaturized components allowed computer-controlled robots to develop. Robotics established itself as the field of science related to the design and production of robots. In 1948, the Argonne National Laboratory developed the first robot that manipulated radioactive elements (PAUL, 1984), inaugurating the branch of *manipulator robots*. These robots became essential components in the industry, especially in the automotive industry. More recently, robots have found new applications outside of the industry (BARRIENTOS et al., 1997), in areas such as cleaning, entertainment, surveillance, search and rescue, underwater, space, and medical applications

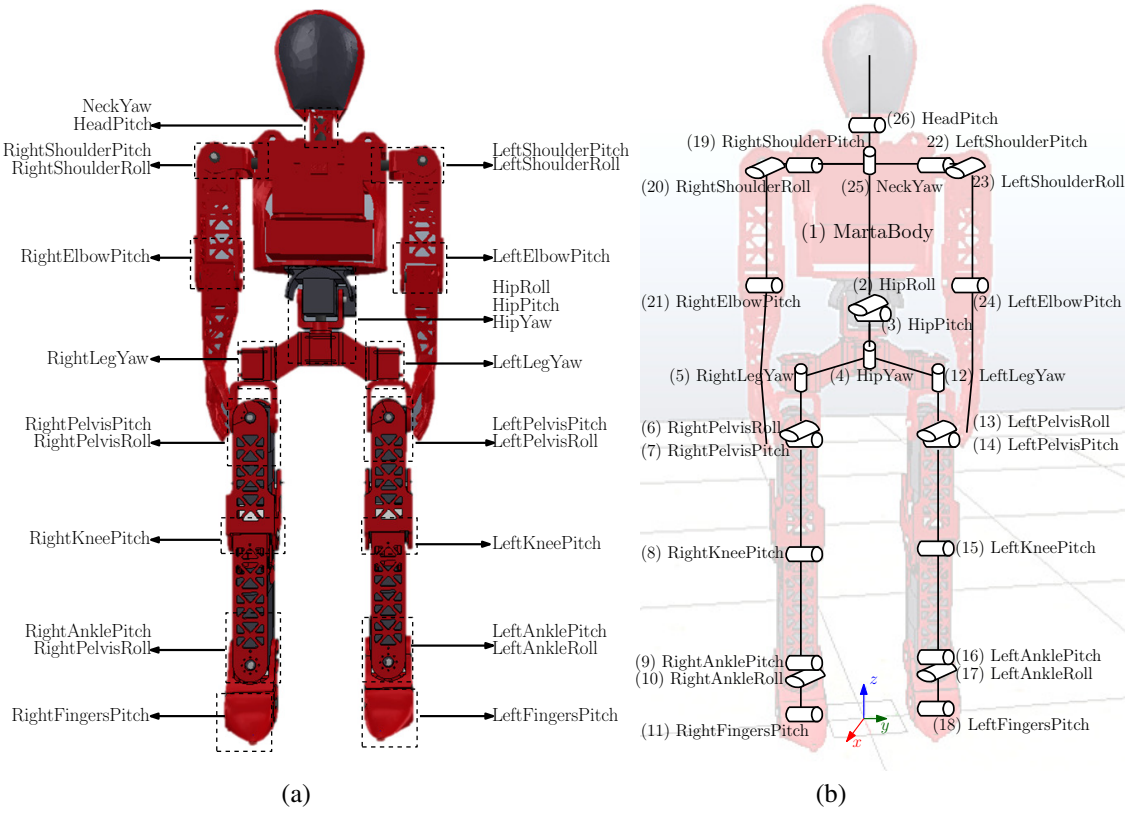
For a long time, human beings have tried to replicate their natural experience in their environment by imitating these events with artificial mechanisms. Those experiences related to their own bodies are some of the most challenging, considering that the functioning of the human body and mind is still a mystery in many ways. Not surprisingly, this field has been an inspiration for artists, engineers, and scientists. In this perspective, we can understand humanoid robotics as the field concerned with the creation of machines that operate in the real world and exhibit the human form and behavior (SICILIANO; KHATIB, 2019).

A robot with a body similar to the human body can most easily adapt to the human environment. In this way, we expect these robots can perform simple tasks like walking in a room without colliding with an object, for example. A task like these involves different challenges such as i) creating a bipedal mechanism with several of DOFs, ii) endow this mechanism with a considerable sensory capacity to make possible its interaction with the environment, iii) developing techniques to control the stability of the robot during the walk, and others.

In modern robotics, the starting point to the project of such mechanisms is *kinematics* and *dynamics*. These concepts will be detailed in the following subsections.

2.1.1 KINEMATICS

Humanoid robots interact and move in 3D space, and some strategy is necessary to model this interaction. A humanoid robot is usually modeled as a collection of rigid parts (links) and joints (the connection between two or more links). The **kinematics** is the theory usually adopted to analyze the relationship between the position and attitude of a link and the joint angles of a mechanism (KAJITA et al., 2014). Figure 1 presents Marta humanoid robot structure with its joints names (CHENATTI et al., 2018).



(a) (b)

Figure 1 - Marta humanoid robot: a) Links; b) Joints (texts) and links (lines connecting joints).

The humanoid robot and its parts have a physical position in the world (absolute position), which can be defined by a fixed coordinate systems (x, y, z) in a *world coordinate system* $(\sum_w)$ (CRAIG, 2006). A *local coordinate system* $(\sum_a)$ can be attached to any moving part, for example, to the left shoulder (Figure 2). The absolute position of the shoulder is defined by vector p_a, and vector r shows the position of the wrist in relation to the shoulder. It can be expressed in the following way:

$$p_h = p_a + r. \tag{1}$$

The absolute wrist end position can be expressed by introducing another vector r' which points the lifted wrist from

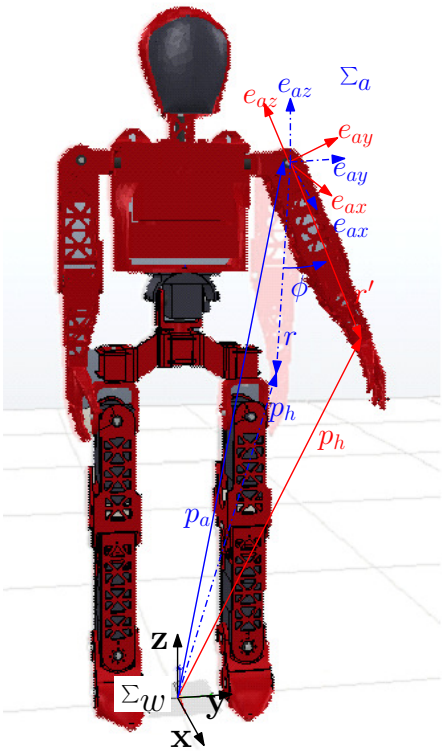


Figure 2 - The world coordinates $\sum_w$ and local coordinate of arm $\sum_a$. Vectors and coordinates system related to the initial (blue) and final (red) position after the rotation of the arm. Inspired by (KAJITA et al., 2014).

the shoulder. The hand lifts by the vector's rotation from r to r' while the shoulder keeps the same position p_a:

$$p_h = p_a + r'. \tag{2}$$

Unlike the world coordinate system which is fixed to the ground, the local coordinate system $\sum_a$ moves together with the attached link. This coordinate system is defined by vectors e_{ax}, e_{ay} and e_{az}, which are parallel to $\sum_w$ in the initial position of the arm. When the arm rotates by an angle ϕ on the axis e_{ax}, the local coordinates also rotates. The new vectors can be obtained by the following equations:

$$e_{ax} = \begin{bmatrix} 1 \\ 0 \\ 0 \end{bmatrix} \quad e_{ay} = \begin{bmatrix} 0 \\ \cos(\phi) \\ \sin(\phi) \end{bmatrix} \quad e_{az} = \begin{bmatrix} 0 \\ -\sin(\phi) \\ \cos(\phi) \end{bmatrix}. \tag{3}$$

These vectors define the matrix R_a ($[e_{ax}, e_{ay}, e_{az}]$), which describes the relationship between r and r':

$$r' = R_a\, r. \tag{4}$$

In the example shown in Figure 2. we can observe:

- The position of the end of wrist p_h viewed from $\sum_w$.
- The position of the end of wrist p_h^a viewed from $\sum_a$, where the upper index of p indicate the local coordinate.

The relationship between p_h and p_h^a can be described from Equations (2) and (4) as:

$$p_h = p_a + R_a\, p_h^a \tag{5}$$

$$\begin{bmatrix} p_h \\ 1 \end{bmatrix} = \begin{bmatrix} R_a & p_a \\ 0\ 0\ 0 & 1 \end{bmatrix} \begin{bmatrix} p_h^a \\ 1 \end{bmatrix}. \tag{6}$$

The homogeneous transformation matrix is obtained from the rotation matrix and the translation matrix, so in this system T_a converts the points described in arm local coordinates to world coordinates:

$$\begin{bmatrix} p \\ 1 \end{bmatrix} = T_a \begin{bmatrix} p_h^a \\ 1 \end{bmatrix}. \tag{7}$$

If we consider the rotation around the x, y and z axes, respectively called Roll (α), Pitch (β) and Yaw (γ), the rotations will be given by (SPONG; HUTCHINSON; VIDYASAGAR, 2004):

$$R_x(\alpha) = \begin{bmatrix} 1 & 0 & 0 \\ 0 & \cos(\alpha) & -\sin(\alpha) \\ 0 & \sin(\alpha) & \cos(\alpha) \end{bmatrix} \tag{8}$$

$$R_y(\beta) = \begin{bmatrix} \cos(\beta) & 0 & \sin(\beta) \\ 0 & 1 & 0 \\ -\sin(\beta) & 0 & \cos(\beta) \end{bmatrix} \tag{9}$$

$$R_z(\gamma) = \begin{bmatrix} \cos(\gamma) & -\sin(\gamma) & 0 \\ \sin(\gamma) & \cos(\gamma) & 0 \\ 0 & 0 & 1 \end{bmatrix}. \tag{10}$$

Now, it is possible to introduce the Z-Y-X Euler angles notation, frequently used to model attitudes of ships, airplanes, and robots (BARRIENTOS et al., 1997). The *Rotation Matrix* will be given by:

$$R_{rpy}(\alpha,\beta,\gamma) = R_z(\gamma)\, R_y(\beta)\, R_x(\alpha), \tag{11}$$

$$R_{rpy}(\alpha,\beta,\gamma) = \begin{bmatrix} c_\gamma c_\beta & -s_\gamma c_\alpha + c_\gamma s_\beta s_\alpha & s_\gamma s_\alpha + c_\gamma s_\beta c_\alpha \\ s_\gamma c_\beta & c_\gamma c_\alpha + s_\gamma s_\beta s_\alpha & -c_\gamma s_\alpha + s_\gamma s_\beta s_\alpha \\ -s_\beta & c_\beta s_\alpha & c_\beta c_\alpha \end{bmatrix}, \tag{12}$$

where: $c_* \equiv \cos(*)$, $s_* \equiv \sin(*)$, and so on.

2.1.1.1 DIRECT KINEMATICS

Direct Kinematics focuses on finding the relative position and orientation of a joint from a reference coordinate system, where the values of each joint and the geometric parameters of the robot are known (PAUL, 1984). In a few words, a robot is a kinematic chain formed by rigid objects joined together by joints, therefore the direct kinematic problem is reduced to finding a *homogeneous transformation matrix T* associating the position and orientation of the joint solicited (CRAIG, 2006).

The kinematic chain can have different representations, the most used is Denavit-Hartenberg (DH). DH defined i-th link by four transformations (BARRIENTOS et al., 1997):

- θ_i: Rotation around the z_{i-1}-axis;

- d_i: Translation along the z_{i-1}-axis;
- a_i: Translation along the x_i-xis;
- α_i: Rotation around the x_i-axis.

The relative position and orientation between the systems associated with two consecutive links of the robot are usually called matrix T_i^{i-1}, so the matrix T_k^0 is the result of the product of the matrices T_i^{i-1} where $i \in [1,k]$, which represents the total or partial kinematic chain that forms the robot.

Illustrate the previous definitions, the matrix T_{21}^1 of the elbow of Marta's left arm (Figure 3) is defined by:

$$T_{21}^1 = T_{19}^1 \; T_{20}^{19} \; T_{21}^{20} \tag{13}$$

$$T_{21}^1 = \begin{bmatrix} c_{\theta_{19}} & -c_{\alpha_{19}} s_{\theta_{19}} & s_{\alpha_{19}} s_{\theta_{19}} & a_{19} c_{\theta_{19}} \\ s_{\theta_{19}} & c_{\alpha_{19}} c_{\theta_{19}} & -s_{\alpha_{19}} c_{\theta_{19}} & a_{19} s_{\theta_{19}} \\ 0 & s_{\alpha_{19}} & c_{\alpha_{19}} & d_{19} \\ 0 & 0 & 0 & 1 \end{bmatrix} T_{20}^{19} \; T_{21}^{20} \tag{14}$$

$$T_{21}^1 = \begin{bmatrix} R_{21}^1 & p_{21}^1 \\ 0 \; 0 \; 0 & 1 \end{bmatrix}, \tag{15}$$

where: T_{20}^1 is the homogeneous transformation matrix of the *shoulder pitch* to the reference coordinate system $\sum_M$, T_{20}^{19} is the transformation matrix of the *shoulder roll* to the *shoulder pitch* coordinate system, and T_{21}^{20} is the transformation matrix of the *elbow pitch* to the *shoulder roll* coordinate system; $c_* \equiv \cos(*)$, $s_* \equiv \sin(*)$; R_{21}^1 and p_{21}^1 is the relative orientation and position of the *elbow pitch*.

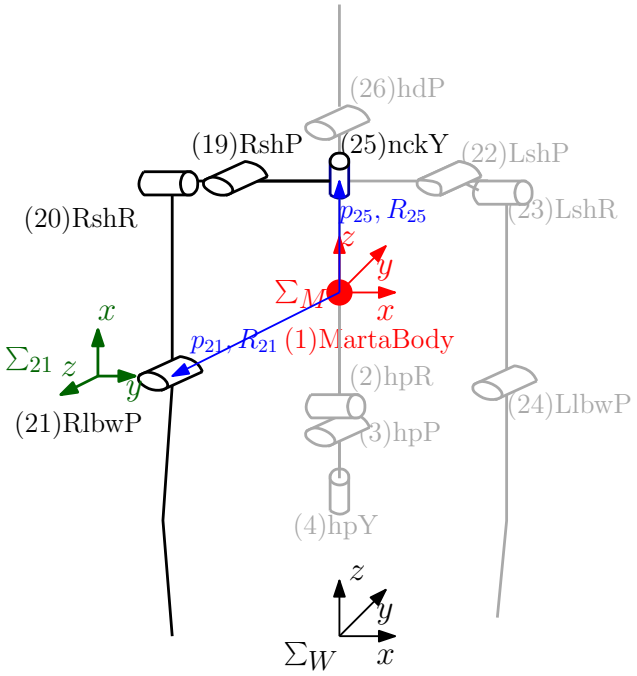


Figure 3 - Direct kinematics model of the Marta humanoid robot arm. The reference coordinate system ($\sum_M$) in the pelvis joint (in red), and the local coordinate system ($\sum_{21}$) in the elbow joint (in green).

2.1.1.2 INVERSE KINEMATICS

Inverse kinematics consists of finding the values that the robot's joints $q = [q_1, q_2, \ldots, q_n]^T$ must adopt so that one end of the robot is positioned p_* and oriented R_* according to a determined spatial location. In the previous section, *direct kinematics* was systematically procedure as it is independent of the robot configuration. However, the *inverse kinematics* obtaining of the equations are strongly dependent on the robot configuration (BARRIENTOS et al., 1997).

Generic procedures have been developed based on the knowledge of the robot kinematics and numerical methods. (KAJITA et al., 2014) exemplifies this method by focusing on the robot's leg, so we focus it on the Marta robot shown in Figure 4. The position and orientation of *HipYaw* (p_4, R_4) is known, and where we want to position and orientation *RightAnkleRoll* (p_{10}, R_{10}). The inverse kinematics following steps:

First, the position of the *RightLegYaw* (p_5) is given by:

$$p_5 = p_4 + R_4 \begin{bmatrix} 0 \\ D \\ 0 \end{bmatrix}, \tag{16}$$

where: D is the distance between *HipYaw* and the *RightLegYaw*. Next, we calculate the position of the crotch viewed from the *RightAnkleRoll* coordinate space.

$$\begin{aligned} r &= R_7^T (p_5 - p_{10}) && (17) \\ r &\equiv [r_x\ r_y\ r_z]^T. && (18) \end{aligned}$$

From this we calculate the distance between the *RightAnkleRoll* and the *HipYaw*, which we will define as:

$$C = \sqrt{r_x^2 + r_y^2 + r_z^2}. \tag{19}$$

In the Figure 4 b), if it used the cosine rule in the triangle ABC it get the angle of the knee (q_8).

$$q_8 = -\arccos\left(\frac{A^2 + B^2 - C^2}{2AB}\right) + \pi. \tag{20}$$

If we define the angle at the lower end f the triangle as α, from the sine rule we get,

$$\begin{aligned} \frac{C}{\sin(\pi - q_8)} &= \frac{A}{\sin\alpha} && (21) \\ \alpha &= \sin^{-1}\left(\frac{A\sin(\pi - q_8)}{C}\right). && (22) \end{aligned}$$

Now, it will focus on the *RightAnkleRoll* local coordinates, as shown in Figure 4, its possible calculate the ankle roll and pitch angles with vector r. So:

$$\begin{aligned} q_{10} &= \text{atan2}(r_y, r_x) && (23) \\ q_9 &= -\text{atan2}\left(r_x, \text{sign}(r_z)\sqrt{r_y^2 + r_z^2}\right) - \alpha. && (24) \end{aligned}$$

And, we still need to compute is the Yaw, Roll and Pitch angles at the base of the leg. From the equations that define each joint.

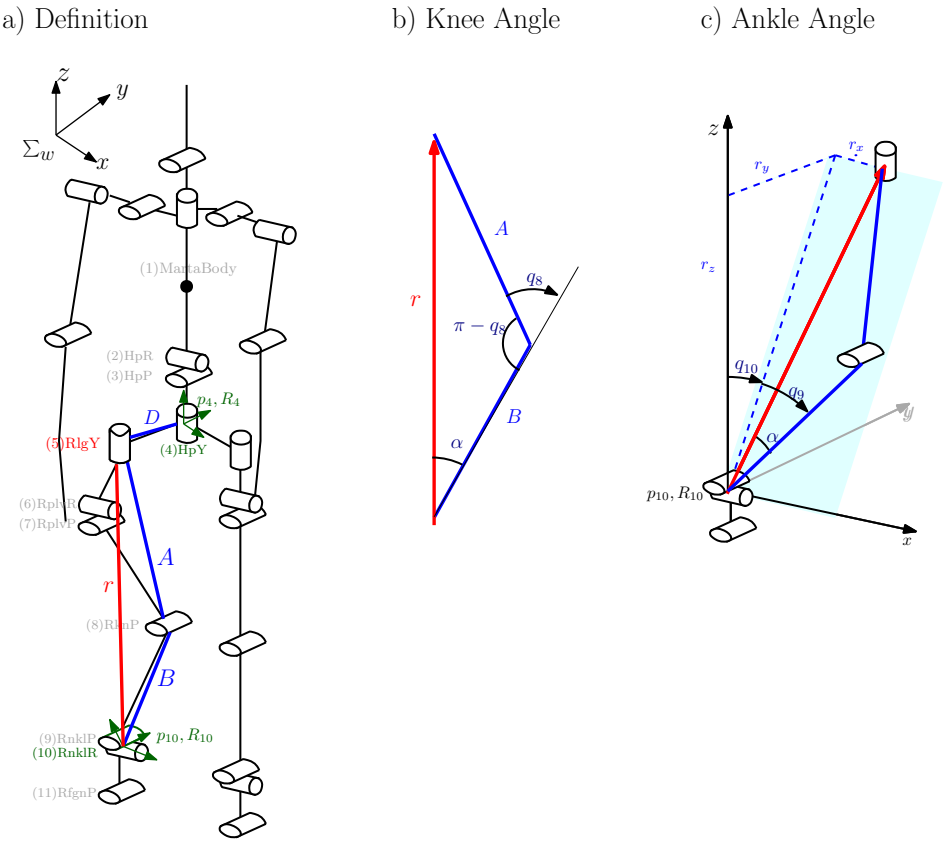


Figure 4 - Calculating Inverse Kinematics of the right leg.

$$R_{10} = R_4 R_z(q_5) R_x(q_6) R_y(q_7) R_y(q_8+q_9) R_x(q_{10}) \quad (25)$$

$$R_z(q_5) R_x(q_6) R_y(q_7) = R_4^T R_{10} R_x(-q_{10}) R_y(-q_8-q_9). \quad (26)$$

When solving the previous expression, we have the following equation:

$$\begin{bmatrix} c_5c_7 - s_5s_6s_7 & -s_5c_6 & c_5s_7 + s_5s_6c_7 \\ s_5c_7 + c_5s_6s_7 & c_5c_6 & s_5s_7 - c_5s_6c_7 \\ -c_6s_7 & s_6 & c_6c_7 \end{bmatrix} = \begin{bmatrix} R_{11} & R_{12} & R_{13} \\ R_{21} & R_{22} & R_{23} \\ R_{31} & R_{32} & R_{33} \end{bmatrix}. \quad (27)$$

Finally, using the left hand side of this equation, we obtain:

$$q_5 = atan2(-R_{12}, R_{22}) \quad (28)$$

$$q_6 = atan2(R_{32}, -R_{12}s_5 + R_{22}c_5) \quad (29)$$

$$q_7 = atan2(-R_{31}, R_{33}). \quad (30)$$

2.1.2 ZERO-MOMENT POINT (ZMP)

Most industrial robots are fixed to the ground, so their workspace is limited to the range that their joints can reach. However, humanoid robots have to move while maintaining an indispensable condition is to keep their feet in contact with the ground. ZMP is usually used to accomplish this purpose.

Zero-Moment Point (ZMP) is a concept introduced by Miomir Vukobratovic in 1968, which denotes a point that the dynamic reaction force produced by contact of the foot with the ground does not produce any moment in the horizontal direction (VUKOBRATOVIC et al., 2012). In other words, ZMP is a point where the sum of all the horizontal forces, the inertia, and the gravity is equal to zero. In 1972, Vukobratovic and Stepaneko wrote the first article on the control of bipedal robots using ZMP (VUKOBRATOVIC; BOROVAC, 2004). The Figure 5 shows the distribution of force across the foot. The charge has the same sign over the entire surface, so it can be reduced to a single resultant force R, so ZMP is the point where the force R acts.

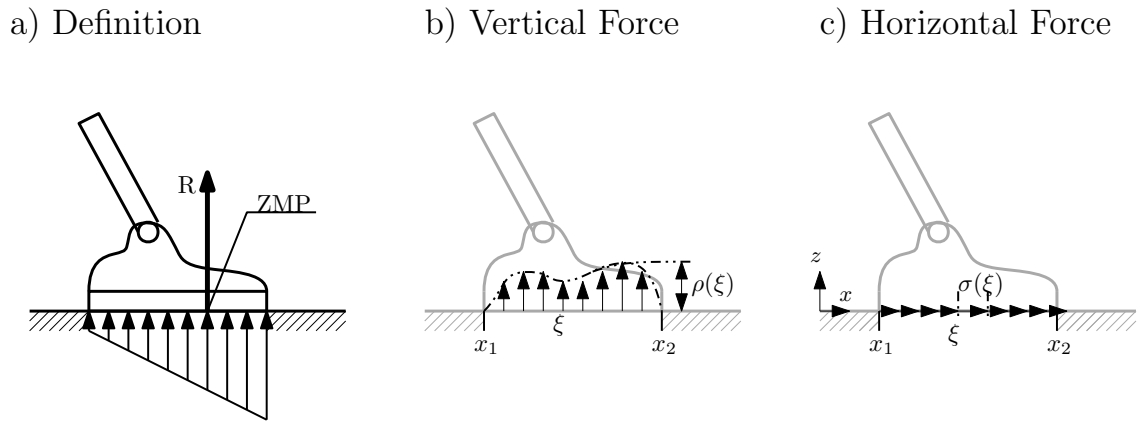


Figure 5 - The definition, vertical force and horizontal force of ZMP (KAJITA et al., 2014).

Another important concept related to ZMP is the support polygon. In Figure 6, the support polygon is the area formed by all the contact points between the robot and the ground. It is relevant to examine Vukovratovic's definition for ZMP: *"the point that exists within the limit of the foot"*. The relationship between Center of Mass (CoM), ZMP, and the support polygon are illustrated in Figure 6. Figure 6a shows a support polygon formed by two feet – are in contact with the ground – and the projection of the CoM on the field coincides with ZMP, which means that the robot is in equilibrium. On the other hand, Figure 6b shows a single support polygon; and the CoM projection is outside the support polygon; consequently, its control is complicated.

2.1.3 DYNAMICS

In the previous subsection, we observed the CoM, ZMP, and the support polygon concepts for the humanoid robot to keep in balance. Now, we analyze the relationship between the reaction force of the ground and the robot's movement, so follow Kajita's deduction (KAJITA et al., 2014).

A humanoid robot has a structure similar to a human, so we are going to define the following ten physical parameters that allow the robot to balance and classified into four groups:

- **Mass:** Total mass of the robot. M[Kg]
- **Center of Mass:** Robot's center of mass. $\mathscr{C} \equiv [x\ y\ z]^T$[m]
- **Momentum:** Measure of a robot's translational motion. $\mathscr{P} \equiv [\mathscr{P}_x\ \mathscr{P}_y\ \mathscr{P}_z]^T$[Ns]
- **Angular Momentum:** Measure of robot's rotational motion about the origin. $\mathscr{L} \equiv [\mathscr{L}_x\ \mathscr{L}_y\ \mathscr{L}_z]^T$[Nms]

The **dynamics** are given by these physical laws, the principle is expressed by:

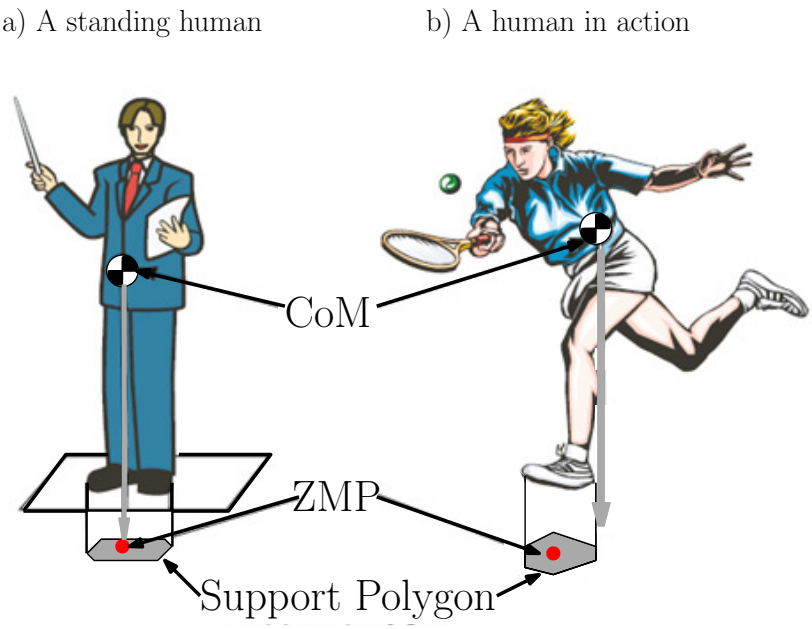


Figure 6 - Relationship between CoM, ZMP and the support polygon (KAJITA et al., 2014).

$$\dot{\mathscr{C}} = \frac{\mathscr{P}}{M} \tag{31}$$

$$\dot{\mathscr{P}} = f_{all} \tag{32}$$

$$\dot{\mathscr{L}} = \tau_{all}, \tag{33}$$

where: f_{all} and τ_{all} denote the sum of all forces and torque applied to the robot from the outside.

The Equation (31) concerning the translational motion gives the relationship between the velocity of the center of mass and the momentum. Meanwhile, Equation (32) for translational motion shows how the momentum changes due to external forces. And, the Equation (33) shows the angular momentum changes according to the sum of the moments.

Analyzing the force applied from outside the robot, we have the gravitational force applied to each component of the robot, and its sum can be considered as the force of Mg applied to the center of mass of the robot $\mathscr{C}$. On earth, the vector of gravitational acceleration is given by $g = [0\ 0\ -9.8]^T [m/s^2]$. So the equation will be given by:

$$f_{all} = Mg + f \tag{34}$$

$$\dot{\mathscr{P}} = Mg + f, \tag{35}$$

where: f denotes the force other than gravity, for example, the ground reaction force. When a robot stands still, the momentum change is zero, as the gravitational force balances out with the ground reaction force. Free fall occurs when the ground reaction force disappears, the moment the robot increases rapidly downward due to gravity.

We take into account the dynamics of the rotational motion that is given by the Equation (33). Furthermore, the moment generated by the gravitational force is given by:

$$\tau_g = \mathscr{C} \times Mg. \tag{36}$$

If τ is without the effect of gravity, the total moment applied to the robot is observed, given by:

$$\tau_{all} = \mathscr{C} \times Mg + \tau. \tag{37}$$

The equation of the rotational movement about the origin can be expressed as follows:

$$\dot{\mathscr{L}} = \mathscr{C} \times Mg + \tau. \tag{38}$$

The τ moment considers only the moment of reaction to the ground applied to the robot. For a robot to stop, the moment must balance with the gravitational force. If the ground reaction moment is not balanced, the angular momentum increases rapidly, causing the robot to fall.

2.1.4 MOMENTUM

A humanoid robot is composed of N points of mass. So, m_i is the mass of the i-th point (KAJITA et al., 2014):

$$M = \sum_{i=1}^{N} m_i \tag{39}$$

$$\mathscr{C} = \sum_{i=1}^{N} m_i \mathfrak{p}_i / M \tag{40}$$

$$\dot{\mathscr{C}} = \sum_{i=1}^{N} m_i \dot{\mathfrak{p}}_i / M \tag{41}$$

$$\mathscr{P} = \sum_{i=1}^{N} m_i \dot{\mathfrak{p}}_i. \tag{42}$$

Equation (40) represents the position of the robot's center of mass, where $\mathfrak{p}_i$ is the position of the i-th mass point. Equation (42) shows the total momentum of the robot where $m_i \dot{\mathfrak{p}}_i$ is the momentum of the i-th mass point. So, by combining Equations (41) and (42):

$$\dot{\mathscr{C}} = \mathscr{P} / M. \tag{43}$$

The dynamics of the robot's momentum is given by the equation of motion of the mass at the i-th point:

$$m_i \ddot{\mathfrak{p}}_i = \sum_{i=1}^{N} f_{ij}^{int} + f_i^{ext}, \tag{44}$$

where: f_{ij}^{int} and f_i^{ext} denote the force applied to the i-th point mass from the j-th one and the force applied to the i-th point mass from the outside the robot. However, due to the law of action and reaction obtain:

$$f_{ij}^{int} = -f_{ji}^{int} \ \ (i \neq j).$$

This equation together to Equation (44) allow us to conclude that the robot's momentum depend only on the external forces.

$$\sum_{i=1}^{N} m_i \ddot{p}_i = \sum_{i=1}^{N} f_i^{ext} \tag{45}$$

$$\dot{\mathscr{P}} = f_{all}. \tag{46}$$

On the other side, the angular momentum of the i-th point mass about the origin is defined by:

$$\mathscr{L}_i = \mathfrak{p}_i \times \mathscr{P}_i. \tag{47}$$

In a robot we also have rigid bodies that are connected by joints. Therefore, it can be taken into account that it is a body that floats in space and rotates without being affected by external force, so the rotational speed is expressed by the angular velocity vector (ω). Assuming that the origin of the reference coordinate system coincides with the center of mass. Therefore, the velocity at a point of the rigid body is expressed by:

$$\upsilon_i = \upsilon(\mathfrak{p}_i) = \omega \times \mathfrak{p}_i. \tag{48}$$

In addition, the total angular momentum of a rigid body about the reference point r can be calculated by:

$$\mathscr{L}^r = \sum_i \mathfrak{p}_i \times (m_i \omega \times \mathfrak{p}_i) \tag{49}$$

$$\mathscr{L}^r = \sum_{i=1}^{NN} m_i \mathfrak{p}_i \times (-\mathfrak{p}_i \times \omega) \tag{50}$$

$$\mathscr{L}^r = (\sum_{i=1}^{NN} m_i \widehat{\mathfrak{p}}_i \widehat{\mathfrak{p}}_i^T)\omega, \tag{51}$$

where: NN is the total of points mass about the reference r. The angular momentum seen above is expressed by the angular velocity vector multiplied by a coefficient matrix (I), called the inertia tensor.

$$\mathscr{L} = (\sum_{i=1} m_i \widehat{\mathfrak{p}}_i \widehat{\mathfrak{p}}_i^T)\omega \tag{52}$$

$$\mathscr{L} = I\omega \tag{53}$$

$$I \equiv \sum_{i=1} m_i \widehat{\mathfrak{p}}_i \widehat{\mathfrak{p}}_i^T, \tag{54}$$

where: $\mathscr{L}$ is the total angular momentum of a rigid body. We do not need to calculate the inertia tensor of any objects with uniform density since the inertia tensor with typical shape. For example, for a cuboid which length of each edge is l_x, l_y and l_z and mass is m:

$$I = \begin{bmatrix} \frac{m}{12}(l_y^2 + l_z^2) & 0 & 0 \\ 0 & \frac{m}{12}(l_x^2 + l_z^2) & 0 \\ 0 & 0 & \frac{m}{12}(l_x^2 + l_y^2) \end{bmatrix}. \tag{55}$$

2.2 INVERTED PENDULUM

The inverted pendulum is one of the most popular systems adopted in the control field. The purpose of the control is to stabilize a naturally unstable open-loop system (ROBERGE, 1960). We should analyze this system in a two-dimensional world since the robot's movement to the *sagittal plane* defined by the axis of the direction of walking and the vertical axis. For this reason, we consider that all of the robot's mass is centralized in Center of Mass (CoM) and that the robot has massless legs that touch the ground at the individual rotary joints (Figure 7).

The robot dynamics control is a challenging task, and the problem can be modeled based on the linear inverted pendulum. It is shown in Figure 7 that the fixed point is ZMP, and the horizontal component of the force f remains

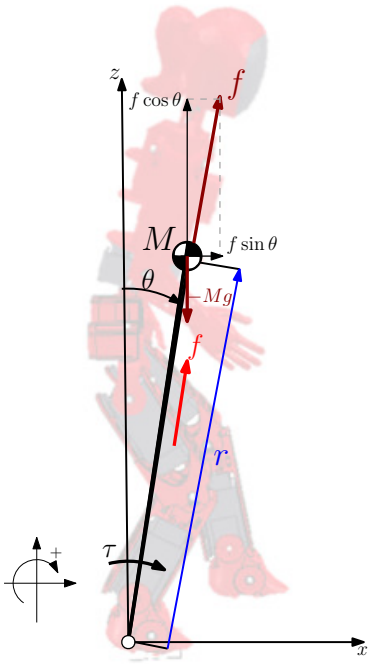


Figure 7 - Bi-dimensional inverted pendulum shows the simplest model for a robot adapted from (KAJITA et al., 2014).

while the vertical component is canceled by gravity. The horizontal component accelerates the CoM horizontally is given by (KAJITA et al., 2014):

$$M\ddot{x} = f\sin\theta \tag{56}$$
$$M\ddot{x} = \frac{Mg}{\cos\theta}\sin\theta \tag{57}$$
$$M\ddot{x} = Mg\tan\theta \tag{58}$$
$$M\ddot{x} = Mg\frac{x}{z}, \tag{59}$$

where: x and z give the CoM of the inverted pendulum. By rewriting the above equation, it is obtain a differential equation for the horizontal dynamics of the CoM:

$$x_{\text{ZMP}} = x - \frac{g}{z}\ddot{x}. \tag{60}$$

Assuming that our inverted linear pendulum model exists the constant z, we can solve the ordinary differential equation:

$$x(t) = x(0)\cosh(t/T_c) + T_c\dot{x}(0)\sinh(t/T_c) \tag{61}$$
$$\dot{x}(t) = x(0)/T_c\sinh(t/T_c) + \dot{x}(0)\cosh(t/T_c) \tag{62}$$
$$T_c \equiv \sqrt{z/g}, \tag{63}$$

where: T_c is the time constant depending on the height of the CoM and the acceleration of gravity. The initial position and velocity are given by $x(0)$ and $\dot{x}(0)$, which together are called *initial conditions*.

In this chapter, we emphasize the kinematic and dynamic concepts of humanoid robots. So, we have the theoretical foundations to understand the complexity of the controllers. In the next chapter, we will focus on the concepts and algorithms of controllers for humanoid robots.

3 MACHINE LEARNING CONTROL

"Keep your love of nature, for that is the true way to understand art more and more"
Vincent Van Gogh

Control can be defined as *"to act, to implement decisions that guarantee that device behaves as desired"* (ANDREI, 2006). This concept is one of the fundamental concepts of robotics and a possible solution to slavery. For example, we can see it in the book "Politics"by Aristotle (384-322 BC) (BENNETT, 1979), where wrote: *"... if every instrument could accomplish its own work, obeying or anticipating the will of others ... if the shuttle weaved and the pick touched the lyre without a hand to guide them, chief workmen would not need servants, nor masters slaves"*. The text shows us the idea of having a slave machine that works automatically, a description similar to the origin of the word **robot** presented in Capek's work (CAPEK, 2004). Therefore, the concept of control an object is inherent in human history. Then, we will detail the relevant investigations of control theory.

The first significant work was on James Watt's centrifugal governor controller in the seventh century. It is a device that automatically controls the speed of steam engines; however, it has an oscillatory behavior as the machines got bigger and more powerful (LEIGH, 2012). In 1868, James Clerk Maxwell developed a formal analysis of the dynamics of the centrifugal governor. He described and analyzed self-oscillation, that is, delays in the system lead to overcompensation and unstable behavior (OGATA, 2010). In 1895, Hurwitz and his colleagues created the Routh-Hurwitz theorem that allows the verification of the stability of the hydroelectric power station in Davos, Switzerland (LEIGH, 2012). This theorem determines whether a linear dynamical system is stable without solving the system.

The works of Minorsky, Hazen, and Nyquist began the foundations of the **control theory**. Minorsky (1922) worked on automatic controllers to pilot ships and demonstrated this stability with differential equations. In 1932, Nyquist developed a simple procedure for determining the closed-loop system stability based on the open-loop response of steady-state sinusoidal inputs. Hazen (1934) introduced the term servomechanisms for position control systems (OGATA, 2010).

The investigators of the Servomechanisms Laboratory group observed the difficulty analyze high-order servomechanisms with the differential equations approach, thus investigated the frequency response methods (BENNETT, 1993). This method allowed the design of a closed-loop linear control system. In the 1940s, Ziegler and Nichols created rules for tuning PID control used to control industrial systems (OGATA, 2010). From 1948 to 1950, Evans worked on his proposal of the root-locus method that allows a rapid system evaluation through the closed-loop characteristics that vary with changes in gain (BENNETT, 1993).

The frequency response methods and the root-locus method are the core of **classical control theory**, providing a stable system that satisfies a set of performance requirements. In the 1960s, digital computers made available time domain analysis in complex systems, modern control theory, based on time-domain analysis and synthesis using state variables. This theory solved the increasing complexity of modern plants and the stringent requirements of precision, weight, and cost (OGATA, 2010).

Modern control theory simplifies the design of the control system based on the physical system model. However, the system stability is sensitive to the error between the model and the system. Therefore, the system may not be stable, which can be avoided by setting an error range and keeping the control system within that range. This method is called robust control (OGATA, 2010).

Robust control designs the system model with a priori uncertainty and guarantees stability; however, the uncertainty doesn't update during the system operation (SKOGESTAD; POSTLETHWAITE, 2007). The field of machine learning is the study of computer algorithms that improve automatically through experience (MITCHELL, 1997). Con-

sequently, the **machine learning control** originates that allows merging these fields. This control focuses on the design control systems as regression problems (BÄCK; SCHWEFEL, 1993), identify and optimizes control parameters (LEE et al., 1997; ALFARO-CID et al., 2008), and determines the control law with continuous updating from the system performance changes (BARTO, 1994).

We have briefly detailed the relevant works on control theory and its evolution. Therefore, we will explain in the following section: control theory and machine learning. Control theory will expound concepts, and PID control. On the other hand, machine learning will explain evolutionary algorithms and reinforcement learning.

3.1 CONTROL THEORY

Control theory defines control rules to achieve a goal in a certain period; their concepts are simple and independent of the application. Consequently, control theory is applied from everyday to complex tasks (LEIGH, 2012). For example, controlling a robot to move involves calculating the robot's position, analyzing possible collisions, estimating movement's trajectory, and executing the movement. To understand the control systems, we must know the following terminologies (OGATA, 2010):

- Plant: physical object to perform a particular operation, such as a mechanical device, an industrial machine, or a robot.
- Process: Method to do or do something. In this case, it will be the objective operation of the control.
- System: Combination of components that act in conjunction and meet an objective. A system can be a physical or abstract phenomenon.
- Disturbance: A signal that negatively affects the output of the system.
- Feedback control: A process of creating a feedback loop to modify the behavior of a system through action that determined by system measurements.

3.1.1 FEEDBACK CONTROL

Figure 8 illustrates a feedback control system. The blue box and the yellow box represented the plant and the control law, respectively. The control law input is the sensor measurement (y), and the actuators (u) modified the plant behavior. The plant is subject to disturbances (w) and allows optimizing a cost function (J) (DURIEZ; BRUNTON; NOACK, 2017). There are two types of control loops: open-loop control and closed-loop control (feedback).

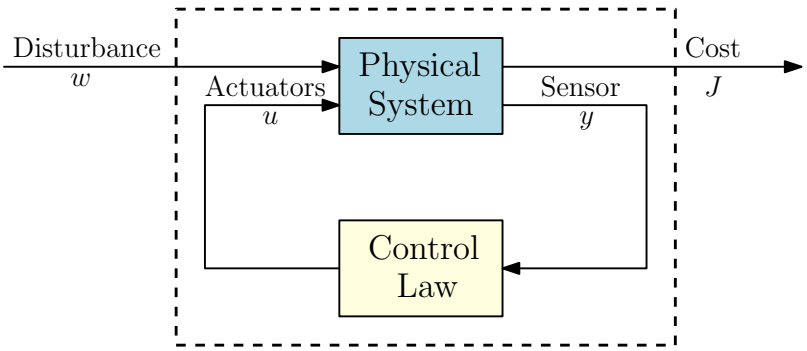


Figure 8 - General framework for feedback control (DURIEZ; BRUNTON; NOACK, 2017).

Figure 9 illustrates the open-loop control architecture. For this strategy, actuation signal b is chosen based on knowledge of the system to produce the desired output that matches the ordered reference signal (w_r) (DURIEZ; BRUNTON; NOACK, 2017). However, the external disturbances (w_d) and sensor noise (w_n) degrade the controller's performance. Time-dependent systems use this architecture. For example, the washing machine controller is determined by periods of soaking, washing, and rinsing.

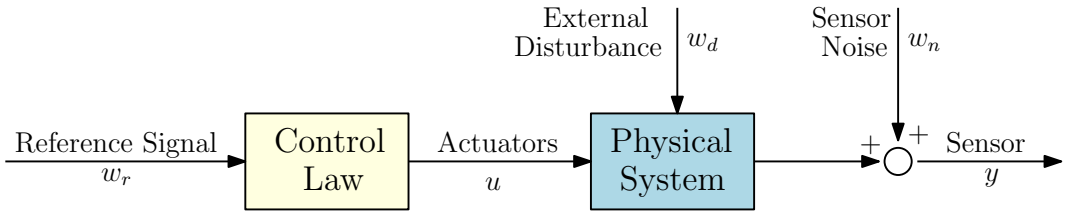


Figure 9 - Open-loop control architecture (DURIEZ; BRUNTON; NOACK, 2017).

The advantages of open-loop control systems are simple construction, easy to maintain, and a feasibly economical system for outputs that are difficult to measure accurately. However, disturbances and calibration changes cause errors in the system output (OGATA, 2010).

On the other hand, the closed-loop system involves sensor feedback, which allows the stabilization of an unstable system by reducing the actuation error (OGATA, 2010). The error e is the difference between the sensor signal y (feedback) and the reference signal w_r (Figure 10). Also, closed-loop control can compensate for external disturbances w_d and attenuate noise w_n (DURIEZ; BRUNTON; NOACK, 2017).

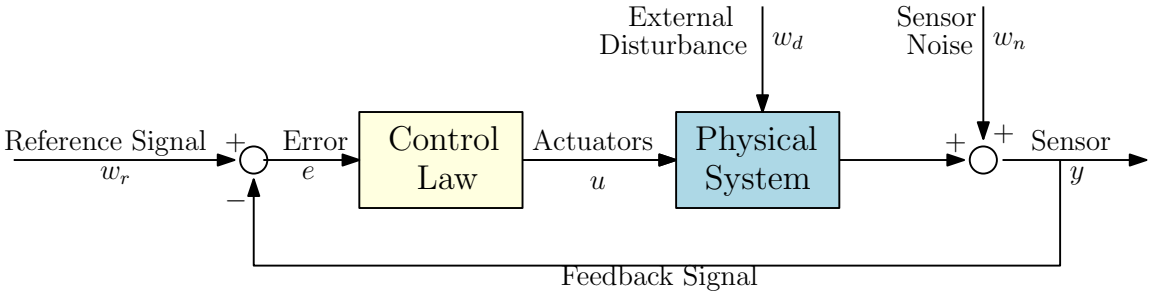


Figure 10 - Closed-loop control architecture (DURIEZ; BRUNTON; NOACK, 2017).

3.1.2 PID CONTROL

The Proportional-Integral-Derivative controller is the most widely used. This controller solved approximately 90 to 95% of control problems. The operation of this controller relies on the feedback signal to eliminate existing drifts (LEVINE, 1996). PID control is generally the sum of three terms (Figure 11). Therefore, this control law is described as:

$$y(t) = u_P(t) + u_I(t) + u_D(t), \tag{64}$$

where u_P is the proportional part, u_I the integral part, and u_D the derivative part.

The proportional part is simple feedback (Equation (65)). This part is proportional to its input, the error signal as a function of time ($e(t)$). The error is the difference between the reference signal (desired value) and the feedback signal (process variable). Furthermore, there is a factor K_p (proportional gain) that allows reversing the error calculation. This

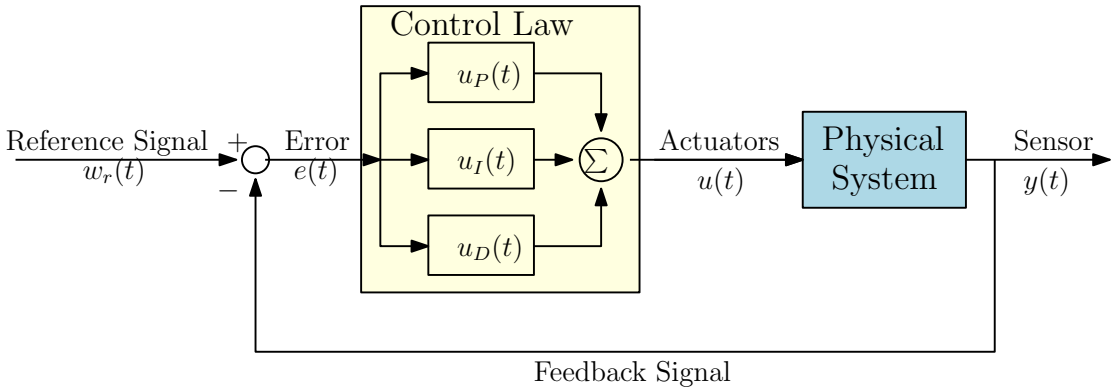


Figure 11 - PID control scheme (DURIEZ; BRUNTON; NOACK, 2017).

action does not modify the shape of the input signal. It only introduces a scaling factor (amplifies or reduces) the value applied at the plant input (PINTO, 2014).

$$u_P(t) = K_p e(t), \tag{65}$$

Proportional control has a steady-state error. A manually adjustable reset term can be added to the control signal to eliminate the steady-state error (LEVINE, 1996). The proportional controller by Equation (65) becomes:

$$u_P(t) = K_p e(t) + u_b(t), \tag{66}$$

where u_b is the reset term.

The integral part is an automatic setting of the reset term. It filters out the low-frequency of the error signal and adds it to the proportional part. This part is given by:

$$u_I(t) = K_i \int_0^t e(t)dt. \tag{67}$$

$$u_I(t) = \frac{K_p}{T_i} \int_0^t e(t)dt, \tag{68}$$

where K_i is the integral factor, and T_i is the integral time.

The derivative part provides anticipated action, given by:

$$u_D(t) = K_d \frac{de(t)}{dt}. \tag{69}$$

$$u_D(t) = K_p T_d \frac{de(t)}{dt}, \tag{70}$$

where K_d is the derivative factor, and T_d is the derivative time.

The parameters of the controller are selecting with controller tuning. That, in the next subsection, we will explain the Ziegler and Nichols method.

3.1.2.1 ZIEGLER-NICHOLS TUNING

Ziegler and Nichols propose rules to determine the proportional gain (K_p), integral time (T_i), and derivative time (T_d). This method is based on the characteristics of the plant dynamics. There are two methods for performing the Ziegler-Nichols adjustment: step response method and frequency response method (OGATA, 2010).

The first method determines K_p, T_i, T_d according to Table 1. Figure 12 shows the input of a unit-step signal that produces the output of an S-shaped curve. This curve is determined by two constant: the delay time L_z and the time constant T_z (OGATA, 2010).

Table 1 - Ziegler-Nichols tuning rule based on step response.

Type of Controller	K_p	T_i	T_d
P	$\frac{T_z}{L_z}$	∞	0
PI	$0.9\,\frac{T_z}{L_z}$	$\frac{L_z}{0.3}$	0
PID	$0.9\,\frac{T_z}{L_z}$	$2\,L_z$	$0.5\,L_z$

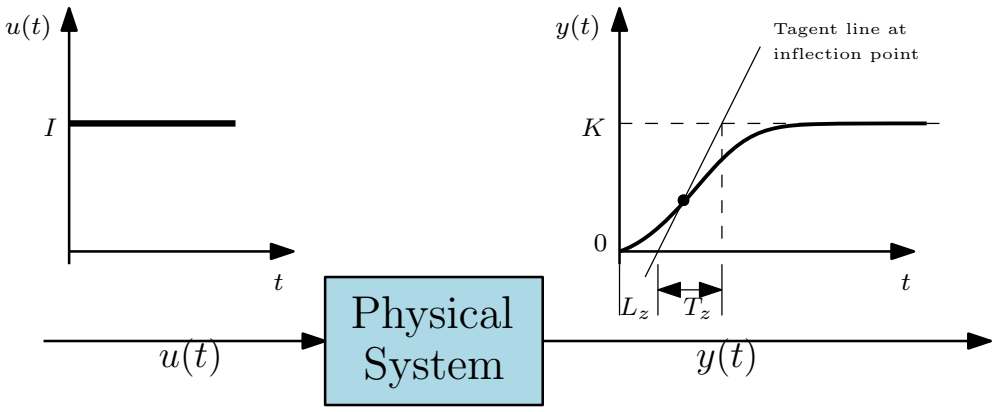


Figure 12 - Unit-step response of a physical system (OGATA, 2010)

The second method is obtained by the proportional gain that is incremental until obtaining an oscillatory response with constant amplitude. At this point, the critical gain (K_u) and the critical oscillatory period (T_u) are determined. Figure 13 shows an example of the process response. The K_u and T_u values are used in Table 2 to obtain the PID settings.

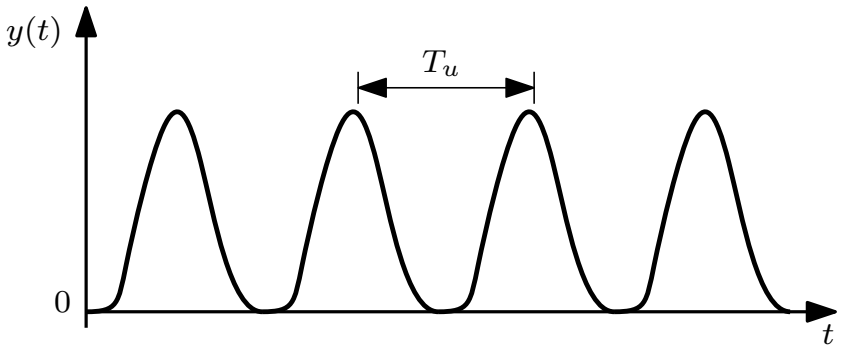


Figure 13 - Examples of the frequency response (PINTO, 2014)

Table 2 - Ziegler-Nichols tuning rule based on frequency response.

Type of Controller	K_p	T_i	T_d
P	$0.5\ K_u$	∞	0
PI	$0.45\ K_u$	$\frac{1}{1.2}\ T_u$	0
PID	$0.6\ K_u$	$0.5\ T_u$	$0.125\ T_u$

3.2 MACHINE LEARNING

The field of **Artificial Intelligence (AI)** emerged at a workshop at Dartmouth College in 1956, making use of concepts from many other science and engineering disciplines. This field is typically concerned with problems like playing games, proving mathematical theorems, writing poetry, driving a car in the street, and diagnosing diseases. Historically, robotics walks side-by-side with AI focusing on building robots that can perform complex tasks in dynamic, unpredictable, and unknown environments.

One of the branches of AI that has experienced a significant increase in the last decades is **Machine Learning (ML)**. This field studies algorithms and techniques for automating solutions to complex problems that are hard to program using conventional programming methods (REBALA; RAVI; CHURIWALA, 2019). ML is based on the premise that systems improve from experience, that is, a computer program is said to **learn** from experience E concerning some class of task T and performance measure P, if its performance at tasks in T, as measured by P, improves with experience E (MITCHELL, 1997).

ML agglutinates different families of algorithms divided according to the input/output nature, data/model, or the algorithms' theoretical nature. Considering the relationship between input and output, there are four types of learning: supervised learning, unsupervised learning, semi-supervised learning, and reinforcement learning.

In **supervised learning**, the program receives a labeled data set, from which it has to learn the fundamental cha-

racteristics in each data set to create a general model so that the next time new data is provided, it can respond with a correct label. For example, the machine receives thousands of images classified as mammals, birds, fish, and reptiles. The machine learns the key characteristics that determine the set to which each image belongs. If an image of a new animal is presented to the machine, it should determine the set to which the animal belongs.

On the other hand, **unsupervised learning** receives unlabeled data set, so this family of ML algorithms usually focuses on identifying similarity trends. For example, a toy store has customer information and needs to know the trend of customers buying teddy bears. So, the machine analyzed the information and determined that male customers between the ages of 18 to 25 are the ones who buy it. In reality, the machine determined a purchase pattern for all customers in the store, which determined this response.

Semi-supervised learning based on between supervised and unsupervised learning. The machine receives a large data set, where only a part of the data is labeled. Typically, an algorithm of this family focus on the use of clustering techniques (unsupervised learning) to determine groups within the data set and then determine the corresponding label using the labeled data for each given group (REBALA; RAVI; CHURIWALA, 2019). The most obvious benefit is the less effort and time employed to label data. For example, a company has 1 billion images of good and bad parts, and it is specified that the machine determines which parts are bad. You would have to label some of these images and run the algorithm for the machine to learn.

Reinforcement learning addresses how an agent learns to make decisions based on perception and execution in its environment. This process does not need a training set and is based on trial and error. For example, a robot needs to go in a straight line from one point to another in a room. The robot will start to move in this environment. If the robot falls, the environment will give it a penalty. Otherwise, it receives a reward.

Another important family of ML algorithms is the one related to **optimization**. Optimization is the process of selecting the hypothesis of models so that an objective function – that describes the quality of the solution – is optimized. The optimal solution is the one in which the variables maximizes or minimizes the objective function. **Evolutionary Algorithms (EA)** are typical representatives of this class of algorithms.

The next sections will describe in more detail two families of ML algorithms that are particularly relevant in this work: i) Evolutionary Algorithms and ii) Reinforcement Learning.

3.2.1 OPTIMIZING WITH EVOLUTIONARY ALGORITHMS

In the last decades, there has been a growing interest in engineering problem-solving systems based on evolution principles since the natural evolution process allows biological systems to be sophisticated, robust, and adaptable (MITCHELL, 1998), (FLOREANO; MATTIUSSI, 2008). Holland and Rechenberg introduced this field of study. They created optimization algorithms that are based on a population of possible solutions, which through a process of selecting the fitness of individuals and some genetic operators allows obtaining an optimal solution (RECHENBERG, 1978), (HOLLAND et al., 1992).

The basic principles of evolutionary systems are derived from the principles of life written by Darwin in 1859, which describes the following (ROTHLAUF, 2006):

- There is a population of individuals with different properties and abilities.
- Nature creates new individuals with properties similar to existing individuals.
- Promising individuals were selected more frequently for reproduction by natural selection.

3.2.2 PRINCIPLES OF THE EVOLUTION

Natural evolution theory is based on four pillars: population, diversity, inheritance, and selection. Two or more individuals define the **population**; an individual is a way of life that is made up of cells formed for genomes. A genome is a complete set of chromosomes, which are the carriers of hereditary biological characteristics. Chromosomes are made up of genes that are responsible for both the structure and processes of organisms. An allele is any variation of a gene, which generally arises through the mutation. It is responsible for the heritable variation (NEAPOLITAN; JIANG, 2018).

In summary, evolutionary systems define an individual through their genes, so there cannot be an individual genetically the same as another (**diversity**). Individuals can reproduce so that individual characteristics are passed on to the next offspring (**inheritance**). Darwinian theory shows that the individual with the greatest aptitude has the highest probability of reproducing due to natural **selection**. The new generations adapt to the changing environment, and the new behavioral characteristics, morphology, functionality, and natural niches allow for greater adaptation to changing environments. Furthermore, natural selection ensures that negative traits are eliminated in future generations, and only beneficial traits are propagated (FLOREANO; MATTIUSSI, 2008).

It is observed that the basic principles of evolutionary theory are sufficiently explanatory and relatively simple to compact them in a model and apply them in several areas. Several formal models have been developed to address specific problems, mainly in the field of optimization (FLOREANO; MATTIUSSI, 2008).

3.2.3 GENETIC ALGORITHM

In 1975, John Holland, in the book *"Adaptation in Natural and Artificial Systems"*, presented the Genetic Algorithm (GA) as an abstraction of biological evolution. Holland relied on the formal study of the phenomenon of adaptation in nature and then developed mechanisms of natural adaptation that could be imported into computer systems (MITCHELL, 1998). GA is a method with a population that can be represented by a string of bits with m individuals. This population uses the ”natural selection” determined by an evaluation function that selects the most suitable individuals for reproduction that, together with operators inspired by genetics - mutation, combination, and inversion - produce individuals' next generation. Figure 14 shows an example of the GA pipeline.

The fundamental part of GA is the selection of individuals from a population for their reproduction. Also, each individual has an associated fitness, which determines how better an individual is. The fitness function $f(*)$ defined the aptitude of the individual. For example, we must obtain the parameters of each joint for a humanoid robot to remain standing. We have as input the position of the robot. Our fitness function is determined by the error between the initial and current positions of our robot. GA's problem arises in choosing an appropriate fitness function.

The selection of the individual that can reproduce is based on the fitness function, and there are several strategies for this selection. The first idea is focused on choosing those who have high aptitudes. However, there is a certain risk of loss of diversity and convergence to a sub-optimal solution in this strategy. Frequently, a proportional selection is applied where the probability ($p(i)$) of an individual i-th reproduce is given by the relationship between its fitness value ($f(i)$) and the sum of the fitness values of all individuals in the population ($\sum_i f(i)$) (MITCHELL, 1998), given by:

$$p(i) = \frac{f(i)}{\sum_{i}^{m} f(i)}. \tag{71}$$

The expected number of individuals of the next-generation (m) is usually the same as the previous population.

Biological mutations inspired the creation of a genetic operator, whereby have a wide variety of operators. The selection of operators depends on the individual's structure. The crossover operator allows exchanging sub-parts of

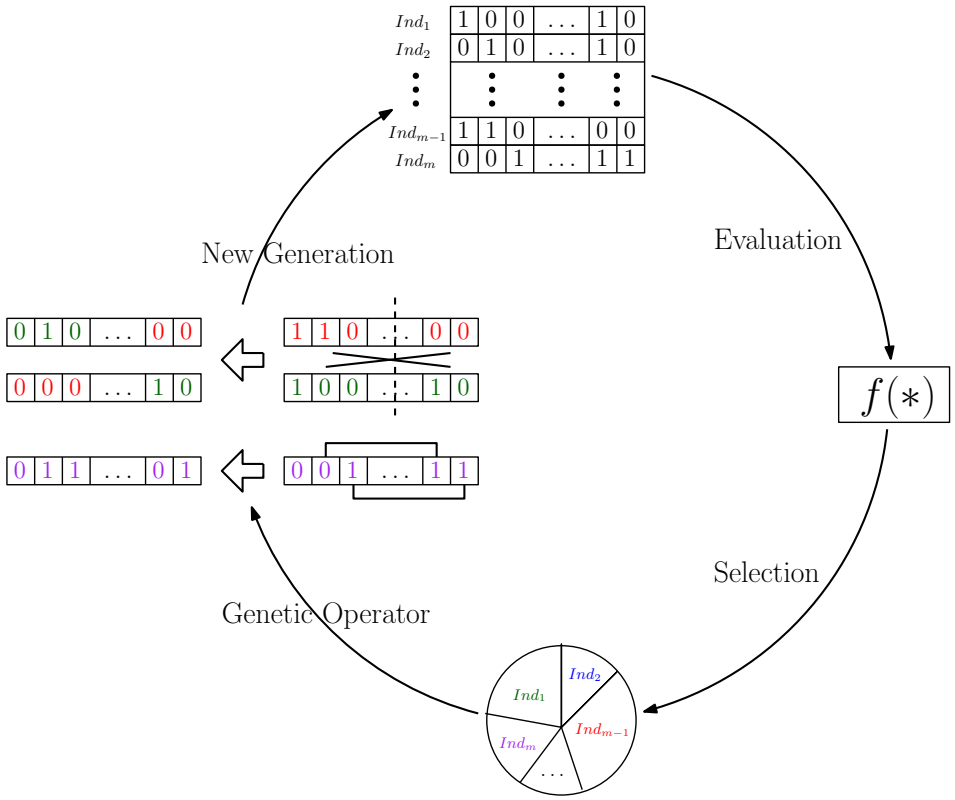


Figure 14 - Main features of the Genetic Algorithm

two individuals, that mimicking biological recombination between two organisms. The mutation operator randomly changes a part of an individual. Finally, the inversion reverses the order of the individual's genes.

3.2.3.1 GENETIC REPRESENTATIONS

A suitable genetic representation must be devised for the genetic operators to be used in the Genetic Algorithm, thus allowing a high probability of generating increasingly better individuals and covering the space of optimal solutions to the problem. There are various representations, so the most used will be detailed (ROTHLAUF, 2006).

The discrete representation describes an individual using the sequence of l discrete values extracted from an alphabet with cardinality k. One of the most used is the binary alphabet $0, 1$ with cardinality $k = 2$. The binary representation can be interpreted directly. In most cases, a different transformation must be performed to be interpreted (FLOREANO; MATTIUSSI, 2008).

Another representation is the floating-point representation that is based on a set of n numbers that belongs to the real numbers. This representation is suitable for solutions that require high precision parameter optimization. The solution space for the problem must be taken into account (MITCHELL, 1998).

The tree representation is suitable for describing hierarchical structures with branching points and conditions, which is used to describe computer circuits, construction procedures, and planning of experiments or compilers. This representation is based on a mapping directly onto a tree (ROTHLAUF, 2006).

3.2.3.2 INITIAL POPULATION

The initial population must be large and diverse enough. Population size depends on the search space's properties and the computational cost of evaluating all individuals over several generations. The literature often finds populations ranging from a hundred to a few thousand individuals. If an individual's evaluation requires real-world experimentation, as is the case with the evolution of a robot, the population size is usually less than one hundred individuals (MITCHELL, 1998).

In the binary and real value representations, a random sequence is generated. However, the real value representation must be within the predefined range. Another way to initialize the population is to seed copy mutations in one or more alleles known to correspond to good or promising individuals. However, this strategy carries the risk that the population is not diverse enough and that evolution remains a sub-optimal solution (FLOREANO; MATTIUSSI, 2008).

3.2.3.3 FITNESS FUNCTION

The fitness function is associated with a numerical score for each individual in the population and indirectly assesses the development of the learning process's quality. There are two critical aspects of designing a fitness function: the choice and combination of fitness components. It is possible to optimize multiple objectives of a problem into a single fitness function. The optimize multiple objectives is based on the properties of the search space and find the relationship between the objectives (REEVES; ROWE, 2002).

The evaluation of fitness is the slowest part of a genetic algorithm since the solution's quality depends on how exhaustive the individual's evaluation has been (RECHENBERG, 1978). The fitness subjective exists when observers rate individual performance through visual inspection. This fitness is often used in the artistic field, architectural structures, and music. It is difficult to formalize aesthetic qualities into objective aptitude functions.

3.2.3.4 SELECTION AND REPRODUCTION

The selection should assign a more significant number of descendants to the best individuals in the population. The high selection pressure means that only a small percentage of individuals will be selected for reproduction. This approach results in a rapid increase in aptitude. However, it risks the diversity of the population. Therefore, a balance must be maintained between selection pressure and diversity generation factors (REEVES; ROWE, 2002).

The proportional selection is defined by the probability of an individual making a copy of their genome based on the relationship between their fitness value and the sum of the entire population's fitness values. The selection of individuals can be visualized as if it were a roulette wheel where the slot size depends on the probability of each individual. This selection method does not work well when all the individuals have almost similar fitness values or obtain a very high probability compared to the others. In the first case, all individuals will have an almost equal probability of having offspring, and evolution is equivalent to a random search. In the second case, almost all the offspring will be copies of the best-fit individual, whose genes will dominate the population and cause premature convergence (FLOREANO; MATTIUSSI, 2008).

There are two solutions to these problems, the first is to scale the fitness values before selection to emphasize or reduce differences; this procedure requires additional parameters. On the other hand, it is possible to classify all the individuals from best to worst and to assign reproductive probabilities proportional to the individual's rank; therefore, no matter how small the difference is between two individuals, the best of the two will always have a greater chance of having offspring (MITCHELL, 1998).

Another type of selection is the truncated selection based on ranges. This selection takes the first n individuals from the list classified by the fitness function and copies these individuals m times until completing the population. It should

be considered that n cannot be small since it would cause premature convergence. This method guarantees it is used in populations that do not need an exhaustive evaluation (FLOREANO; MATTIUSSI, 2008).

Selection by tournament consists of organizing a tournament among a small subset of individuals to generate each offspring. k individuals are randomly selected from a population, where k is known as the tournament size. The individual with the best aptitude of the k individuals generates an offspring. The individuals that were not chosen return to the population to participate again in other tournaments. Tournament selection allows selection pressure and genetic diversity to be maintained (MITCHELL, 1997).

The generational change is based on the fact that the offspring generated replaces the entire previous population. However, in a complex search space, the fitness assessment can be very noisy or that genetic mutations strongly affect the individual, such that a better individual is lost. For this case, there is the strategy of elitism, which consists of keeping the n best individuals from the previous population. However, it is possible to relax the entire generational replacement by inserting only a few descendants into the population, which is called gradual generational change (FLOREANO; MATTIUSSI, 2008).

3.2.3.5 GENETIC OPERATORS

Genetic operators capture the effects of biological mutations. This subsection will describe the most commonly used operators in the genetic representations seen above. The main task of genetic operators is to introduce diversity into the population and the exploration of solutions.

The crossover operator is the most used, which allows the offspring to inherit the parents' characteristics through the recombination of the genomes of two selected individuals. The newly created offspring are randomly paired for gene crossing. This operator is based on the benefit of the synergistic effect of the combination of the solutions found by the parents. However, genetic recombination is equivalent to a large random mutation and has a deleterious effect on the individual's fitness (MITCHELL, 1998).

One-point crossover can be applied to discrete, real-value representations. It consists of randomly selecting a crossing point in each of the two chains and exchanging genetic material between individuals around this point. Multi-point crossing consists of randomly selecting n crossed points in the two chains and exchanging genetic material between these points (REEVES; ROWE, 2002).

For real value representations, smooth and arithmetic crossovers are options. The uniform crossover consists of exchanging the genetic content in n positions chosen at random. Instead, arithmetic crossing creates a unique gene by taking the average of n randomly chosen positions from the two gene chains. For tree-based representations, the crossover randomly selects one node in each parent and changes the two corresponding sub-trees (FLOREANO; MATTIUSSI, 2008).

On the other hand, mutations are small random modifications that allow explorations of variations of existing solutions. Mutations are useful for escaping from local minimal and making further progress in the highly convective population where genetic recombination has little effect. However, the number and size of mutations must be relatively low to avoid the loss of previously discovered solutions (BACK, 1996).

Mutation in binary representations consists of alternating selected bit values. In the real value representation, the individual selected a position and added a random value based on the Gaussian distribution $N(0,\sigma)$, where 0 is the mean and σ is the variance.

In binary representations, the mutation consists of alternating the selected bit values. In real-value representations, a selected position is modified by adding a random value drawn from a Gaussian distribution $N(0,\sigma)$, where 0 is the mean and σ is the variance, to produce few large mutations. In tree-based representations, a selection node swaps its

content with another from the same subset. If the selected node is a terminal, it will be replaced by another element chosen at random from the terminal sets (FLOREANO; MATTIUSSI, 2008).

3.2.3.6 SCALING

Regulation of which individuals reproduce is critical in GA with small populations. At first, we may have some extraordinary individuals in a population with multiple mediocre individuals. The underlying problem is that the early strongest candidates may be in or around a local peak fitness space that can be difficult to get out of. In contrast, the weakest candidates could be closer to the global maximum. The aggressive population pruning compromises genetic diversity, reducing search space and search speed. Therefore, we may want to scale fitness so that selection pressure remains the same throughout the run (GOLDBERG, 1989).

Linear scaling is a simple procedure where a linear relationship is defined between the raw fitness f and the scaled fitness f' ($f' = af + b$). The coefficients a and b can be chosen in several ways; however, in all cases, it is desired that the scaled average fitness f'_{avg} be equal to the average raw fitness f_{avg} to ensure that each member of the average population contributes an expected offspring to the next generation.

Linear scaling misbehaves with negative aptitudes. Sigma truncation scaling can overcome the presence of poor individuals. This method uses information from the population variance to pre-process the objective function's values (f) before scaling. Sigma truncation scaling is given by:

$$f' = f - (f_{avg} - c * f_{dev}), \tag{72}$$

where f_{avg} is the mean of the objective functions in the population. The constant c is chosen as a multiple of the population standard deviation (f_{dev}), and the negative results are arbitrarily set to zero.

3.2.4 REINFORCEMENT LEARNING

In Reinforcement Learning (RL), the agent learns by interacting with the environment to maximize its gain. Imagine a pet learning to bark for food. If we feed our pet every time it barks (action), the pet is likely to repeat this action to get a response from us. The reward determines the agent's behavior. For this example, our pet (agent) learns an action to achieve a goal without any previous teaching. Therefore, Reinforcement Learning resided on the agent behavior who seeks to act positively in its environment to obtain a reward from it.

Formally, the Reinforcement Learning (RL) agent interacts with its environment, producing a sequence of actions $a_1, a_2, \ldots$, over time. These actions affect the agent's environment, resulting in a reward or punishment r_t at each time step t. The algorithm's goal is to learn to act in a way that can maximize some measure of utility in the future (NEAPOLITAN; JIANG, 2018). Reinforcement Learning comprises the following elements (SUTTON; BARTO, 2018):

- **Policy** (π_t): defines the agent's behavior in time t. Policy maps the perceived states of the environment to the actions it must take in that state.
- **Reward** (R_t): define the objective of an RL problem, that is, which events are good and bad for the agent. The agent's goal is to maximize or minimize the total reward he receives in the long run. The reward is unalterable but can be used as a basis for modifying policies.
- **Value Function** ($V_\pi(*)$): determines what is better in the long term, although the reward indicates what is better in the immediate sense. Generally, the total amount of reward that an agent can expect to accumulate in the future, starting from that state.

- **Environment model**: allows it to make inferences about how the environment will behave. The model is used to plan for future situations before they happen. This element may not exist, as there are free-model methods that learn from trial and error.

One of RL's challenges is the trade-off between exploration and exploitation. To obtain a greater reward, an agent must prefer the actions that it has tried in the past, resulting in reward (exploit). However, it falls into an over-exploitation problem, where it only focuses on a particular group of actions falling to a local solution. On the other hand, if the algorithm explores too much, it may never really learn the problem's desired solution. This dilemma is called exploration-exploitation, which has been studied intensively. These two paradigms must be balanced, which is very difficult to carry out (SUTTON; BARTO, 2018).

Besides, we have a strand of RL that has helped in the last decades increasing investigations in this area, which are the deep reinforcement learning algorithms. These methods employ deep neural networks and large-scale computing power to improve the capacity to store the experiences of Reinforcement Learning (RL) agents. This new capability has allowed solving problems in various fields, such as board games or video games. Another field that is strongly using this aspect is robotics so that robots learn complex tasks for dynamic environments (REBALA; RAVI; CHURIWALA, 2019).

The difference between the Evolutionary Algorithms and Reinforcement Learning relies on the fact that RL algorithms can remember and use what it has learned over time. In contrast, evolutionary systems make decisions in each generation based on information from the current generation.

3.2.4.1 MARKOV DECISION PROCESS

Markov Decision Process (MDP)s is a mathematical model that formalizes sequential decision-making, where actions influence not only the immediate reward but also subsequent situations or states, and through future rewards. Thus, MDPs involve delayed reward and the need to compensate for immediate and delayed reward (SZEPESVÁRI, 2010).

Figure 15 shows a robot responsible for making decisions, called *agent*. The thing it interacts with, comprising everything outside the agent, is called the *environment*. The agent continuously selects *actions* and the environment responds to these actions and presents new situations to the agent. The environment also generates *rewards*, which are numerical values that the agent aims to maximize over time through its actions' choice. This process is mathematically defined as a Markov Decision Process (MDP).

Specifically, the agent and the environment interact in each sequence of discrete time steps, $t = 0, 1, 2, 3, \ldots$. At each time step t, the agent receives the representation of the *state* of the environment, $S_t \in S$, and the agent selects an *action*, $A_t \in A(s)$. In the next step, in part as a consequence of the action, the agent receives a *reward* R_{t+1}, and finds a new state S_{t+1}. So the sequence would be as $S_0, A_0, R_1, S_1, A_1, R_2, S_2, A_2, \ldots$ (SUTTON; BARTO, 2018) (DONG et al., 2020).

A finite MDP occurs when the state, action, and reward sets have a finite number of elements. For this case, the variables R_t and S_t have a defined discrete distribution that depends on the state and the previous action. Therefore, given $s' \in S$ and $r \in R$, there is a probability that these values occur at time t, given the particular values of the preceding state and action, given by:

$$p(s', r \mid s, a) \doteq Pr\{S_t = s', R_t = r \mid S_{t-1} = s, A_{t-1} = a\}. \tag{73}$$

$$\sum_{s' \in S} \sum_{r \in R} p(s', r \mid s, a) = 1, \text{ for all } s \in S, a \in A(s), \tag{74}$$

where $s', s \in S$, $r \in R$, and $a \in A(s)$. The function p defines the MDP *dynamics*. It specifies a probability distribution for each choice of s and a (Eq. 74).

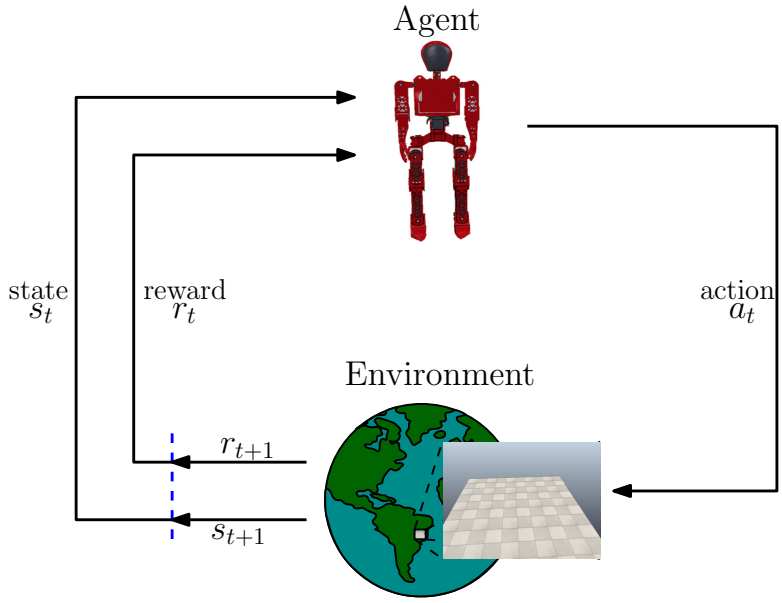


Figure 15 - The agent-environment interaction in Reinforcement Learning (SUTTON; BARTO, 2018)

From p, we can derive the state transition probability, denoted by:

$$p(s' \mid s,a) \doteq Pr\{S_t = s' \mid S_{t-1} = s, A_{t-1} = a\}. \tag{75}$$

$$p(s' \mid s,a) = \sum_{r \in R} p(s', r \mid s,a). \tag{76}$$

In addition, the expected reward of the state-action pair can also be calculated as:

$$r(s,a) \doteq \mathbb{E}\{R_t \mid S_{t-1} = s, A_{t-1} = a\}. \tag{77}$$

$$r(s,a) = \sum_{r \in R} r \sum_{s' \in S} p(s', r \mid s,a). \tag{78}$$

An MDP is an abstraction of the problem of goal-directed learning from interaction. MDP is an independent scheme of the goal, the sensory, the memory, and control details. In conclusion, the RL problems can be reduced to three signals between the agent and his environment: the signal to represent the agent's action, the signal to represent the agent decision (states), and the signal to define the agent's objective (reward). (SUTTON; BARTO, 2018).

3.2.4.2 AGENT'S OBJECTIVE

The agent's objective is to maximize the cumulative reward it receives in the long term. The sequence of rewards received over time t is denoted $R_{t+1}, R_{t+2}, R_{t+3}, \ldots$. The agent can only maximize the future cumulative reward for the rest of the episode since it cannot go back in time to change past rewards. G_t will be defined as the cumulative reward that the agent receives (REBALA; RAVI; CHURIWALA, 2019):

$$G_t \doteq R_{t+1} + R_{t+2} + R_{t+3} + \cdots + R_T, \tag{79}$$

where T is a final time step. This approach makes sense for applications where the agent-environment interaction naturally breaks into sub-sequences (*episodes*), for example, board games where you have an ending that can be winning or losing the game, and again return to the initial state of the board.

On the other hand, other agent-environment interactions are not divided into identifiable episodes but continue continuously without limit. In this case, the maximization of the reward tends to infinity easily. The concept that is added is the *discount*, in which the agent tries to select actions to maximize the sum of the discount rewards that he receives in the future, such that (SUTTON; BARTO, 2018):

$$G_t \doteq R_{t+1} + \gamma R_{t+2} + \gamma^2 R_{t+3} + \gamma^3 R_{t+4} + \ldots, \tag{80}$$

$$G_t \doteq R_{t+1} + \gamma(R_{t+2} + \gamma R_{t+3} + \gamma^2 R_{t+4} + \ldots), \tag{81}$$

$$G_t \doteq R_{t+1} + \gamma G_{t+1}, \tag{82}$$

where γ is a parameter, $0 \leq \gamma \leq 1$, called the *discount rate*. If γ is less than 1, its value decreases exponentially, therefore, future rewards contribute less percentage of their numerical value in the cumulative rewards, and this avoids the problem of infinite rewards.

3.2.4.3 POLICY AND VALUE FUNCTIONS

Reinforcement Learning algorithms need to estimate the value function, that is, state functions that estimate *"how good"* it is for the agent to be in a state and *"how good"* it is to perform an action given that state. The notion of *"how good"* is defined in terms of future rewards that can be expected (expected performance). Value functions are defined concerning a particular way of acting, called policies (SEWAK, 2019).

A policy is a mapping of states to selection probabilities of possible actions. If the agent follows the policy π at time t, then $\pi(a \mid s)$ is the probability that $A_t = a$ if $S_t = s$. Reinforcement learning methods specify how agent policy is changed as a result of their experience. For any π policy and any s state, the following consistency condition between the value of s and the value of its possible successor states (MITCHELL, 1997) (SUTTON; BARTO, 2018):

$$V_\pi(s) \doteq \mathbb{E}_\pi[G_t \mid S_t = s]. \tag{83}$$

$$= \mathbb{E}_\pi[R_{t+1} + \gamma G_{t+1} \mid S_t = s]. \tag{84}$$

$$= \sum_a \pi(a \mid s) \sum_{s'} \sum_r p(s', r \mid s, a)[r + \gamma \mathbb{E}_\pi[G_{t+1} \mid S_{t+1} = s']]. \tag{85}$$

$$= \sum_a \pi(a \mid s) \sum_{s', r} p(s', r \mid s, a)[r + \gamma V_\pi(s')], \text{ for all } s \in S \tag{86}$$

where it is implied that the action, $a \in A(s)$, that the next state, $s' \in S$, and that the reward, $r \in R$.

Solving a Reinforcement Learning task means finding a policy that achieves many rewards. Infinite MDP, an optimal policy can be precisely defined, which the value function defines a partial order on the policy. The π policy is defined as better than or equal to a π' policy if its expected return is greater than or equal to π' for all states. We denote all optimal policies by pi_*, where they share the same state value function, called the optimal state value function (v_*), defined as (SUTTON; BARTO, 2018):

$$V_*(s) \doteq \max_\pi V_\pi(s), \text{ for all } s \in S \tag{87}$$

Optimal policies also share the same *optimal action-value function*, denoted as q_*, and defined by:

$$q_*(s, a) \doteq \max_\pi q_\pi(s, a), \text{ for all } s \in S, a \in A(s) \tag{88}$$

$$q_*(s, a) = \mathbb{E}[R_{t+1} + \gamma v_*(S_{t+1}) \mid S_t = s, A_t = a] \tag{89}$$

The different Reinforcement Learning algorithms solve the problems of learning value functions, improving politics, or performing both simultaneously, such as the actor-critic method.

We detail the theoretical foundations of humanoid robots, and in this chapter, we see the different control strategies. In the next chapter, we summarize the works of literature for the walk of humanoid robots.

4 HUMANOID ROBOT WALKING APPROACHES

"To know more is to be freer"
César Vallejo

Walk can be defined as *"to move or go somewhere by placing one foot in front of the other on the ground, but without running"* (OXFORD, 2015). If we adopt this definition, we must consider that the robot must always have at least one foot in contact with the ground during the walking process. However, how can we make a machine to perform such movements? Bipedal locomotion is approached in the literature in various distinct aspects, including walking patterns, control algorithms, types of controllers, control strategies, optimization techniques, and so on. Some of these aspects will be discussed in this chapter. Figure 16 presents some of the most relevant aspects found in literature considering the state of art.

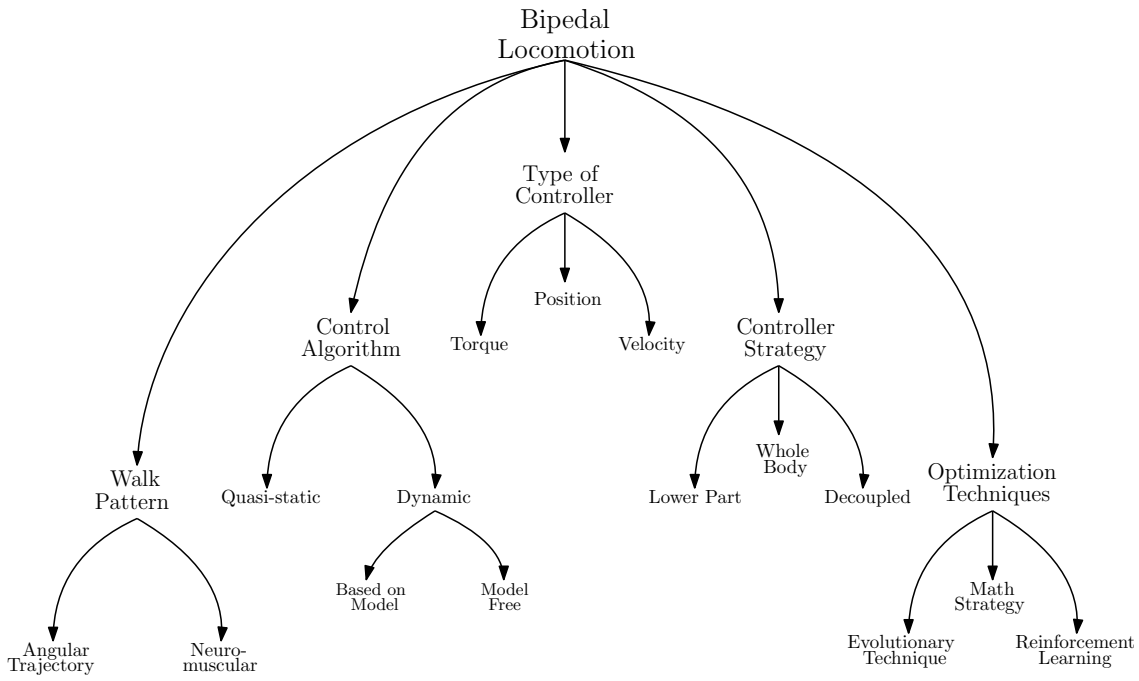


Figure 16 - Classification of State of Art: most relevant aspects in humanoid robot walking literature.

This chapter will discuss the most relevant aspects of humanoid walking focusing on the control algorithms that are grouped into: i) quasi-static walking and ii) dynamic walking. The first group is based on the theory of equilibrium, where the projection of the Center of Mass must never leave the support polygon formed by the robot's feet. Instead, the second group is based on the fact that there are times when the robot's center of mass leaves the support polygon.

4.1 CONTROL ALGORITHMS

4.1.1 QUASI-STATIC WALKING

In the previous section, we defined the quasi-static walk as the one based on the principle that the robot must remain in balance at every moment of time. This approach is composed of two consecutive phases to complete a step that will

be repeated during *n* steps. In the first phase, or *simple support*, the robot lifts one foot, and the other foot leaves it as a support point on the ground. In the second phase, or *double support*, the robot has both feet on the ground, and the center of mass must move towards the space covered by the foot that was previously lifted (CUEVAS; ZALDÍVAR; ROJAS, 2004) (FIGUEROA; MEGGIOLARO, 2016). Figure 17 shows the vertical projection of the center of mass on the floor of the phases of the quasi-static walk, which represent: a) the single support (Figure 17a), the double support (Figure 17b), and loss of balance (Figure 17c). In the Table 3 shows a summary classification of the representative articles of this control algorithm.

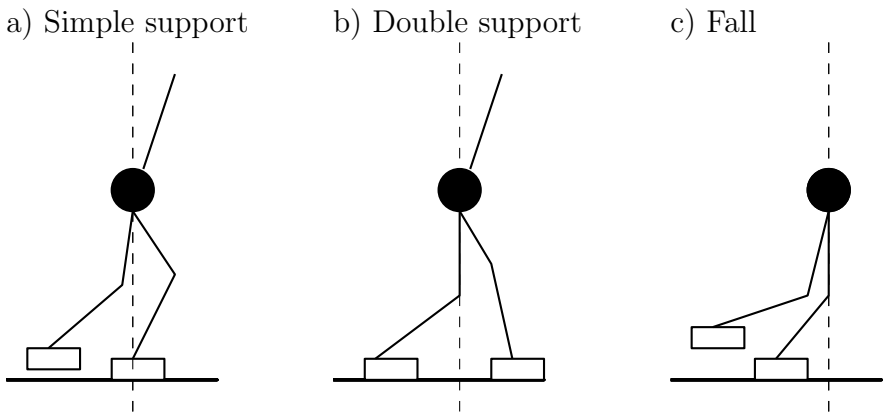


Figure 17 - Phases in the quasi-static walk (CUEVAS; ZALDÍVAR; ROJAS, 2004).

Table 3 - Quasi-static walking works review.

Authors	Optimization Techniques	Type of Controller	Controller Strategy
(NIEHAUS; RÖFER; LAUE, 2007), (FIGUEROA; MEGGIOLARO, 2016), (RAJ; SEMWAL; NANDI, 2019), (LIU et al., 2020)	Evolutionary T.	Position	Lower Part
(DONG et al., 2019)	Math Strategy	Position	Lower Part
(MURAI; KANETA; SUGIHARA, 2013)	Math Strategy	Velocity	Whole Body
(YUAN et al., 2017), (NANDI et al., 2016)	-	Position	Lower Part
(REHER et al., 2016), (GRIZZLE; CHEVALLEREAU, 2017), (STEIN et al., 2017)	-	Torque	Lower Part
(YAMAGUCHI et al., 1999)	-	Torque	Whole Body

The quasi-static walk focuses on optimizing the parameters of a classic controller applied to the abstraction of the kinematic model of the robot (FIGUEROA; MEGGIOLARO, 2016) (SARI; DEWANTO; PRAMADIHANTO, 2019) (RAPETTI et al., 2019). On the other hand, other research focuses on obtaining information from the environment that helps to make a more robust controller (CHEN et al., 2017) (MOHAMED; AWAD; MAGED, 2020).

The quasi-static walking approach involves an analysis of the structure and kinematics of the robot. Therefore, this approach algorithms are tedious to transfer to other robots, as requires a detailed analysis of the humanoid robot. The problem with this approach is that the dynamics do not affect the robot, that the robot's movements are at a low speed.

4.1.2 DYNAMIC WALKING

The dynamic walking is determined by two basic postures: *support*, where a robot foot is in contact with the ground (that usually represents about 60% of the walking cycle), and *balance*, when we have an elevated foot (that usually represents about 40% of the walking cycle) (ROSE; GAMBLE, 1994). So, we can have, for example, simultaneously a left leg support and a right leg in balance. Also, there is the *double support* phase where both feet are in contact with the ground (that usually represents about 20% of the walking cycle). These basic positions are subdivided into (SÁNCHEZ-LACUESTA et al., 1993):

- Support: contact with the heel is subdivided, sole support, medium support, the elevation of the heel, and finally elevation of the foot.
- Balance: it is determined to accelerate, average balance, and deceleration.

The Figure 18. shows the movement of these three positions and their corresponding intervals.

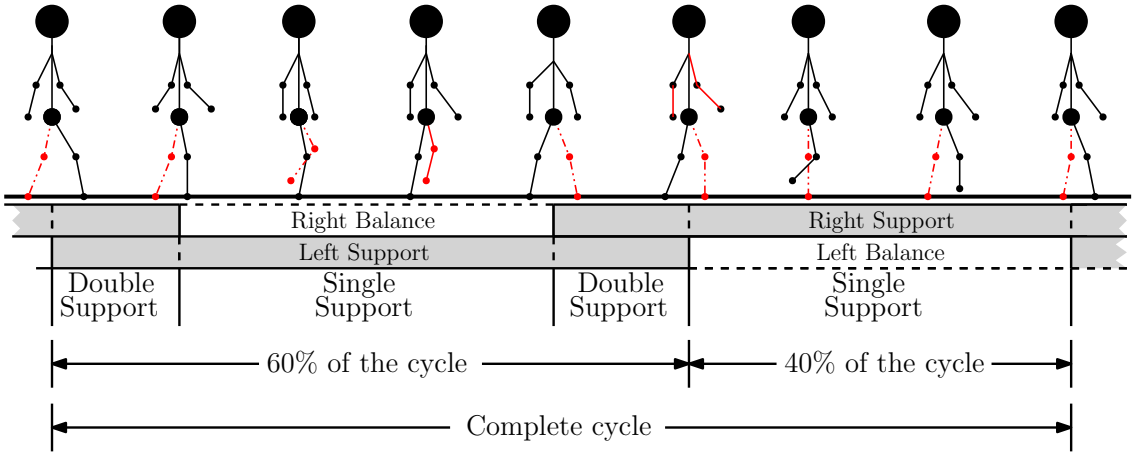


Figure 18 - Human walking cycle (FIGUEROA; MEGGIOLARO, 2016).

The main difference between dynamic gait and quasi-static gait is that it allows the CoM to leave the support polygon in some instants of time. This approach relies on finding a walking pattern that is the robot controller (MAXIMO; COLOMBINI; RIBEIRO, 2017) (VILLAGRA; BALAGUER, 2011). Table 4 shows the most relevant papers found in the recent literature grouped by the following characteristics: i) type of controller, ii) walk pattern, iii) optimization technique, iv) type of controller, and v) controller strategy. It distinguishes two subgroups: i) **based on model** and ii) **model-free**, which will be better explained in the follow subsections.

4.1.2.1 BASED ON MODEL

The main difference between model-based dynamic walking and quasi-static walking is that the former uses the dynamic model of the robot, that is, the internal and external forces that interact with the robot (VILLAGRA; BALAGUER, 2011). The Inverted Pendulum Model (IPM) is one of the most used models and offers a framework to deal with the robot's balance. This model facilitates the analysis to perform the cyclical movement on flat terrain (TANG; ER, 2007) (COROS; BEAUDOIN; PANNE, 2010). Works in this field has focused on improving this model (LIPM, TLIPM, nonlinear IPM) (ERBATUR; KURT, 2006) (KHADIV et al., 2017) (PRISTOVANI; SANGGAR; DADET, 2018) (WANG

Table 4 - Dynamic walking works review.

Authors	Type of Controller	Walk Pattern	Optimization Technique	Type of Controller	Controller Strategy
(CHEN et al., 2017), (GAO; GU, 2019)	Based on Model	-	-	Position	Whole Body
(COROS; BEAUDOIN; PANNE, 2010), (NAVEAU et al., 2016)	Based on Model	-	-	Torque	Whole Body
(KIM; OH, 2004)(KONG et al., 2008), (BUSCHMANN; LOHMEIER; ULBRICH, 2009)	Based on Model	-	-	Torque	Lower Part
(HAMED; GREGG; AMES, 2018), (HEREID et al., 2018)	Based on Model	-	-	Position	Lower Part
(KOLATHAYA et al., 2018)	Based on Model	Neuro-muscular	-	Velocity	Lower Part
(MENG et al., 2018)	Based on Model	Neuro-muscular	-	Torque	Lower Part
(ANTONOVA; RAI; ATKESON, 2016)	Model Free	Neuro-muscular	Evolutionary Technique	Torque	Lower Part
(SHAFII et al., 2009b), (SHAFII et al., 2009a), (ABEDI; ALAMIRPOUR; MIRSHAHVALAD, 2017), (MAXIMO; COLOMBINI; RIBEIRO, 2017), (SILVA; MAXIMO; GÓES, 2019)	Model Free	Angular Trajectory	Evolutionary Technique	Position	Lower Part
(OGINO et al., 2004)	Model Free	-	Reinforcement Learning	-	Lower Part
(TOMAZELA; COLOMBINI, 2019)	Model Free	-	Reinforcement Learning	Torque	Decoupled
(CANIO et al., 2016)	Model Free	-	-	Position	Lower Part
(FARCHY et al., 2013)	Model Free	-	-	Position	Whole Body

et al., 2019), working in the frontal plane of the robot (3D-LIPM) (YUAN et al., 2017) and on the proposition of internal equations to increase model robustness (VLIPM, 3D-FPBIPM, etc.) (MOTOI; SUZUKI; OHNISHI, 2008) (GUO; ZENG; MENG, 2019).

4.1.2.2 MODEL-FREE

The model-free methods attempt to create a controller without any model of the robot dynamics. The stability of the movement is evaluated by trial and error, so it typically requires techniques derived from ML or optimization algorithms.

The basic idea behind model-free control was proposed by a group of researchers who considered that biped walking should not be planned analytically but, instead, should raise as a result of nonlinear oscillations that arise from the feedback and dynamic interaction between the system and the environment (KATOH; MORI, 1984). These researchers proposed a Central Pattern Generator (CPG) inspired in patterns generated by neural activity in human beings that rhythmically drive and stereotype motor movement such as walking, swimming, breathing or chewing. They tested the idea and observed a robot could naturally generate motion that is robust against disturbances (TAGA; YAMAGUCHI; SHIMIZU, 1991) (HYON; MORIMOTO; KAWATO, 2010) (SAYARI; MASMOUDI; ZAIER, 2019)

(KHAN; CHEN, 2019).

Another model-free method is based on the periodic movement of walking. In general, there is an asymmetry between the right and left sides of the body, where one side is always half a period behind the other. This movement was based on the bipedal angular trajectory of two joints: the hip and the knee, which were captured from the human gait shown in Figure 19 (YANG et al., 2007).

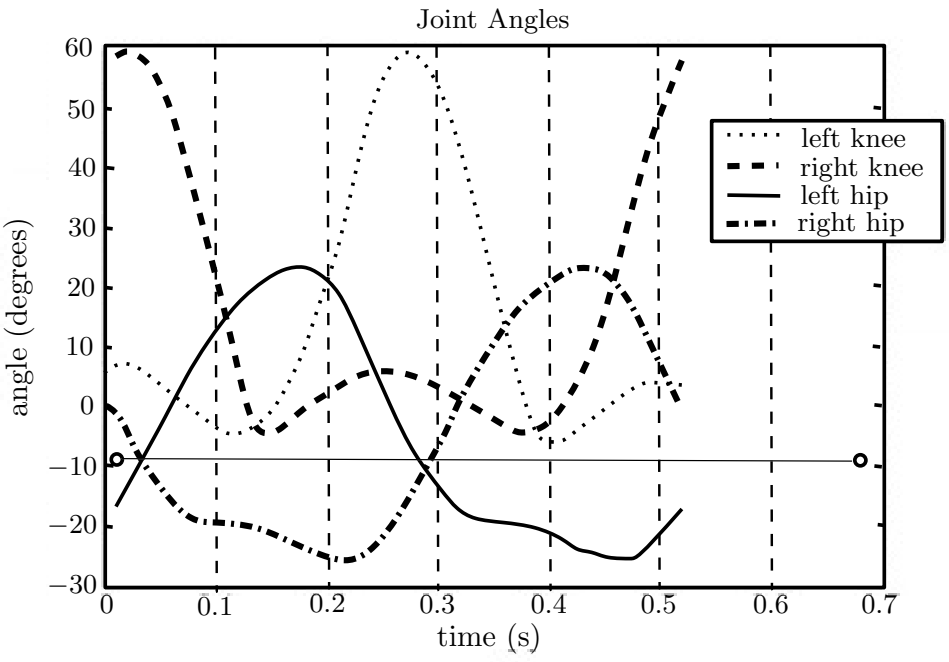


Figure 19 - Human walking angular trajectory of the hip and knee (YANG et al., 2007).

It is observed that the periodic function can be described as a Fourier series given (YANG et al., 2007):

$$F(t) = \frac{a_0}{2} + \sum_{i=1}^{\infty} (a_i \cos(i\omega t) + b_i \sin(i\omega t)), \tag{90}$$

where a_0 is the offset of the signal, ω is the periodic frequency of the signal given by π/L, and L is half of the period of the signal. The Fourier transform formula considers that i is infinite.

Subsequent works proposed the use of the finite Fourier series as a Truncated Fourier Series (TFS), which eliminates the part of the cosine 91 (SHAFII et al., 2009a):

$$F(t) = a + \sum_{i=1}^{n} b_i \sin(i\omega t). \tag{91}$$

The TFS for generating each portion of hip and knee trajectories are formulated as follow (Fig. 20.):

$$\theta_{hip}(t) = \begin{cases} a_{hp} + A_{hp} \sin(i\omega_{hp} t) & t \in R_1 \\ a_{hp} + B_{hp} \sin(i\omega_{hp} t) & t \in R_2 \end{cases} \tag{92}$$

$$\theta_{knee}(t) = \begin{cases} a_{kn} + C_{kn} \sin(i\omega_{kn} t_2) & t \in R_1 \\ a_{kn} & t \in R_2 \end{cases} \tag{93}$$

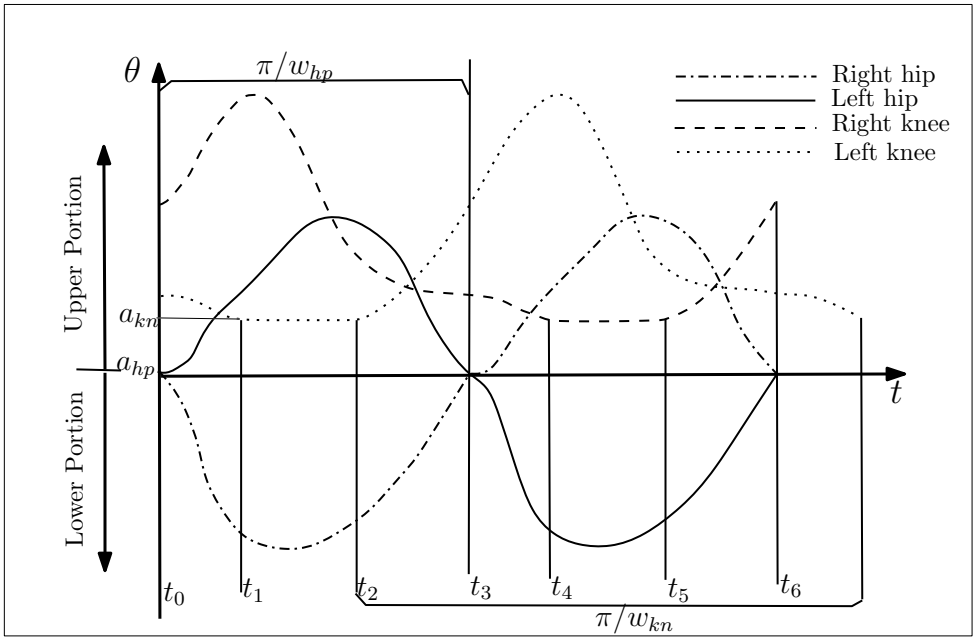


Figure 20 - Generic pattern of walk elaborated from human walks features (YANG et al., 2007)

$$R_1 = [kT, \frac{T}{2} + kT\rangle, k \in \mathbb{Z} \quad (94)$$

$$R_2 = [\frac{T}{2} + kT, (k+1)T\rangle, k \in \mathbb{Z} \quad (95)$$

where: A_{hp}, B_{hp} and C_{kn} are constant coefficients for generating signals; T is the period of the trajectory of the angles; a_{hp} and a_{kn} are signal offset.

If there is an explicit description of the sinusoidal movement of the joints, the TFS equations of the joints relevant to the walk can be obtained. Therefore, the investigations are focused on obtaining the parameters of the TFS equations, in the example of the knee and hip joints would be the parameters of A_{hp}, B_{hp}, C_{kn}, a_{hp}, a_{kn} and t_2. The evolutionary system's algorithms are commonly used (SHAFII et al., 2009a) (SHAFII et al., 2009b) (SHAFII; REIS; LAU, 2010) (MAXIMO; COLOMBINI; RIBEIRO, 2017) (VILLELA et al., 2017) (SALEHI; AZARKAMAN; AGHAABBASLOO, 2018) (ABEDI; ALAMIRPOUR; MIRSHAHVALAD, 2017).

Recent works have successfully applied the model-free technique for generating walking patterns for robots with distinct DOFs and constructive characteristics, like the NAO robot (SHAFII; REIS; LAU, 2010) and the Robonova (MAXIMO; COLOMBINI; RIBEIRO, 2017). The application of the model-free technique to distinct robots or using distinct DOFs require important adaptations both in curve generation and parameters tuning by ML algorithms.

5 PROPOSED APPROACH

"Inspiration unlocks the future."
(The Wind Rises film)
Studio Ghibli

The increase in the number of DOFs in humanoid robots is a trend since they are expected to mimic human movements more and more faithfully. A typical example is the humanoid robot *Marta* developed by the workgroup. Marta's specifications are detailed in Chapter 6, and it has some particular characteristics that are not usually found in conventional humanoid robots:

- The spherical joint designed at the robot waist, that adds 3 new DOFs that are critical to the robot stability;
- The small foot compared to the proportions usually adopted in humanoid robots that significantly reduces the support polygon in the support phase of the walk;
- The presence of articulated toes in the feet, that brings new possibilities to the walking trajectories.

The complexity of the kinematic model of the robot increases with the number of DOFs of the robot. So the robot becomes much more complex to control using classical methods. In practice, the use of controllers based on dynamic models limits the number of free joints of the robot and, therefore, the tasks they can perform. *Marta's* constructive features bring us new challenges and possibilities to control her balance during the walk. While the static stability is reduced, on the one hand, new dynamic control possibilities are available on the other hand. Since *Marta* has a more complex design and dynamic when compared to similar robots. Therefore, the literature fails in providing curves that could guide the robot's movements with this complexity using model-free techniques. In this context, this chapter details the proposed approach for the investigation of new control strategies applicable to *Marta* humanoid robot.

5.1 DECOUPLED CONTROLLER APPROACH

This work proposes decoupling the humanoid robot controller into two distinct controllers: i) the *waist* controller and ii) the TFS controller, each applied to a specific joints subset (Figure 21). The waist controller provides Marta's upper balance actuating on the spherical joint based on some PID tuning technique. A model-free controller actuates over the remaining DOFs (hip, knee, ankle, foot, shoulder, and elbow DOFs) based on the generation of joints movement patterns using Truncated Fourier Series (TFS) equations and expecting to generate a successful walk.

5.1.1 TFS CONTROLLER

This controller focus on the generation of the joints movements trajectory patterns by using TFS, as shown in Equation (91). Since Marta robot has a mechanical structure distinct from those robots modeled by previous works (SHAFII et al., 2009a) (SHAFII et al., 2009b) (SHAFII; REIS; LAU, 2010) (MAXIMO; COLOMBINI; RIBEIRO, 2017), the TFS controller must be reviewed.

In previous works, the following joints were considered: *pelvis pitch*, *knee pitch* and *ankle pitch* (SHAFII et al., 2009a). The movements of these joints were determined by the following equations:

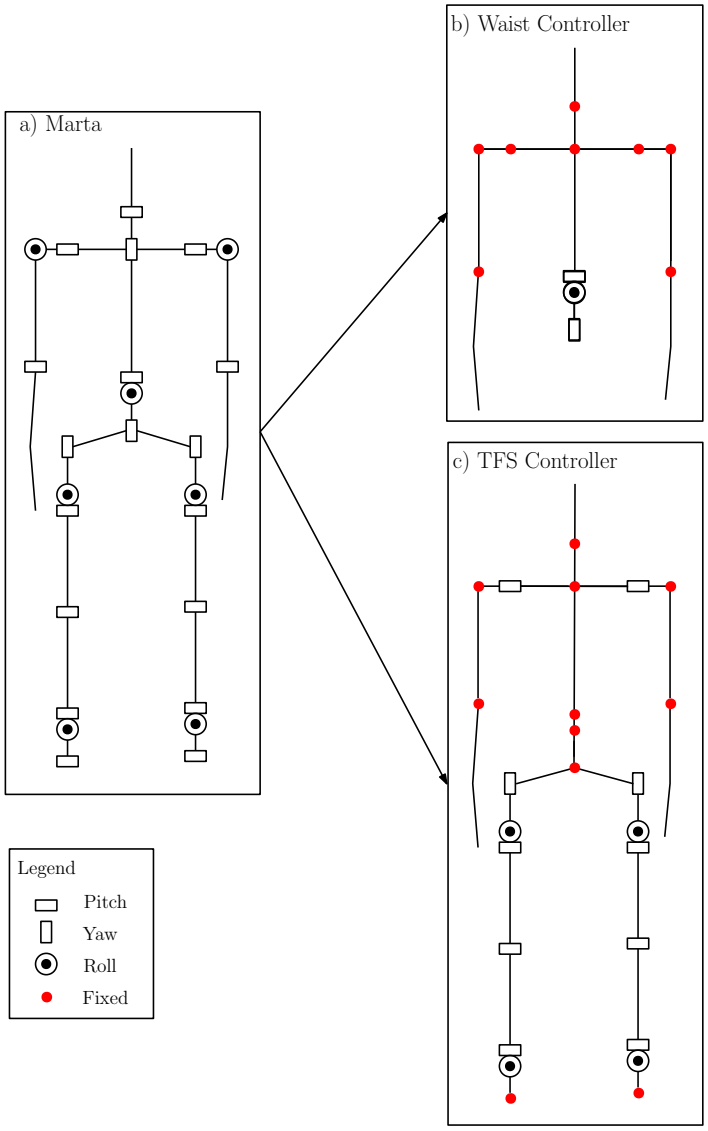


Figure 21 - Controller decoupled approach for the humanoid robot Marta. a) represents all of Marta's joints, b) the upper part determined by the spherical joint (waist controller), and c) the part of the arms and legs (TFS controller). The red dots represent the joints that will not be controlled.

$$\theta_{RplvP}(t) = O_{plvP} + d_f * A_{plvP} \sin\left(\frac{2\pi t}{T}\right). \tag{96}$$

$$\theta_{LplvP}(t) = O_{plvP} - d_f * A_{plvP} \sin\left(\frac{2\pi t}{T}\right). \tag{97}$$

$$\theta_{RknP}(t) = \begin{cases} O_{knP} & t \in R_1, \\ O_{knP} - d_f * C_{knP} \sin\left(\frac{2\pi(t-t_{2k})}{T}\right) & t \in R_2. \end{cases} \tag{98}$$

$$\theta_{LknP}(t) = \begin{cases} O_{knP} + d_f * C_{knP} \sin\left(\frac{2\pi(t-t_{2k})}{T}\right) & t \in R_1, \\ O_{knP} & t \in R_2. \end{cases} \tag{99}$$

$$R_1 = [kT, \frac{T}{2} + kT\rangle, \ k \in \mathbb{Z}, \tag{100}$$

$$R_2 = [\frac{T}{2} + kT, (k+1)T\rangle, \ k \in \mathbb{Z}, \tag{101}$$

where the indexes *plvP* and *knP* are references of the *pelvis pitch* and *knee pitch*, respectively. θ_{plvP} and θ_{knP} represents the angles in the trajectory at time t. O_{plvP} and O_{knP} are the offset of the signal. A_{plvP}, C_{knP} represents the positive amplitude of the signal. Also, d_f is the damping factor that will increase in each cycle from 0.0 to 1.0

In the original equations (SHAFII et al., 2009a), a constant B represented the negative amplitude for the pelvis. In our work this amplitude will be the same as the positive amplitude (A_{plvP}). T represents the period of the movement that determines walking speed, t_{2K} is the time delay of the signal knee. The constant k allows us to determine the walking cycle. Finally, we have the constant d_f which is a damping factor, since due to the complexity of the humanoid, the movements must begin little by little to gain stability in each cycle. Figure 22 shows the *pelvis pitch*, *pelvis roll* and *knee pitch* curves, emphasizing the phase difference between the three curves (gray lines between time 2.0 to 2.5).

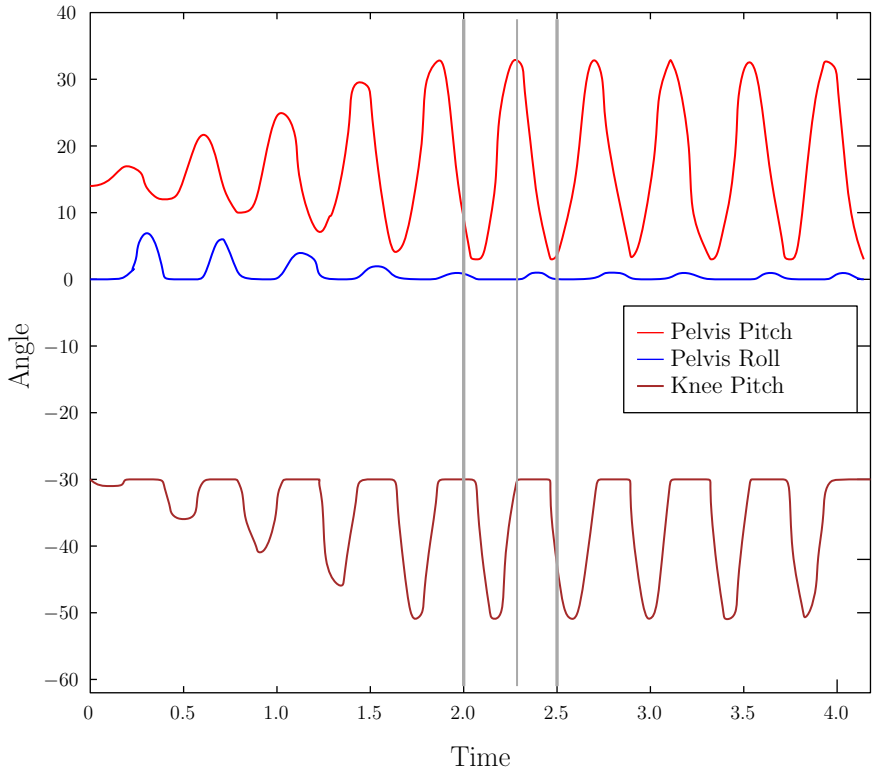


Figure 22 - The *pelvis pitch*, *knee pitch* and *pelvis roll* trajectory (SHAFII; REIS; LAU, 2010).

The equations for the *pelvis roll* were adopted as follows:

$$\theta_{RplvR}(t) = d_f * \left(A_{plvR} \sin\left(\frac{2\pi(t - t_{2p})}{T} \right) \right). \tag{102}$$

$$\theta_{LplvR}(t) = d_f * \left(A_{plvR} \sin\left(\frac{2\pi(t - t_{2p})}{T} \right) \right), \tag{103}$$

where the index *plvR* is reference of the *pelvis roll*. θ_{plvR} represents the angle in the trajectory at time t. A_{plvR} and B_{plvR} represent the positive and negative amplitude of the signal, respectively. T represents the period of the movement that it is same that *pelvis pitch*, *knee pitch* and *ankle pitch*, but we have a time delay of the signal (t_{2p}).

It can be seen in the Equations (96) and (97) of *right pelvis pitch* and *left pelvis pitch*, they have a different sign, since when one of the legs moves forward (to the balance phase) the other remains in the support phase. On the other hand, in Equations (102) and (103) both pelvis joints move to the same side, helping to release the leg when performing the swing.

The *shoulderpitch* joint was subsequently introduced by the following equation (SHAFII et al., 2009b):

$$\theta_{shP}(t) = d_f * \begin{cases} D_+ \ \sin(\frac{2\pi t}{T}) & t \in R_1, \\ D_- \ \sin(\frac{2\pi t}{T}) & t \in R_2, \end{cases} \tag{104}$$

where D_+ and D_- represent the positive and negative amplitude of the *shoulder pitch*. In this case, the *shoulder pitch* does not show the displacement of the signal, since the premise of the arms is that the movement begins with the angle 0°.

The *ankle pitch* provides stability to the robot relative to the ground. We investigated three different approaches::

1. The *ankle pitch* is defined by the sum of the joints of the *pelvis pitch* and *knee pitch*, that is:

$$\theta_{RankP}(t) = -(\theta_{RplvP}(t) + \theta_{RknP}(t)), \tag{105}$$
$$\theta_{LankP}(t) = -(\theta_{LplvP}(t) + \theta_{LknP}(t)). \tag{106}$$

2. The *ankle pitch* follows an orientation sensor places in the robot's feet.

3. The *ankle pitch* depends on the orientation sensor of the robot's pelvis. If the robot's pelvis sensor is greater than the inclination maximum (*MaxIncl*), the *ankle pitch* follows the sum of the robot's feet and pelvis sensors; otherwise, it will only be determined by the feet sensor. Its illustrated in the following equation:

$$\theta_{ankP}(t) = \begin{cases} -1.0 * (FeetSensor + Kp_{ank} * PelvisSensor) & abs(PelvisSensor) \geq MaxIncl, \\ -1.0 * FeetSensor & otherwise. \end{cases} \tag{107}$$

The *ankle roll* is expected to be parallel to the ground. In this context, two distinct approaches were investigated:

1. The *ankle roll* is defined by the inverted angle of the *pelvis roll*, that is:

$$\theta_{RnkR}(t) = -(\theta_{plvR}(t)), \tag{108}$$
$$\theta_{LnkR}(t) = -(\theta_{plvR}(t)). \tag{109}$$

2. The *ankle roll* follows an orientation sensor placed in the robot feet.

5.1.2 WAIST CONTROLLER

The decoupling of the *Marta* robot in upper and lower parts allows the robot's upper parts to act like an inverted pendulum. In this way, the 3-DOFs in the robot waist are expected to improve the robot's stability dynamically. We hypothesize that the insertion of these additional DOFs can allow a greater number of wave patterns generated by the TFS controller to provide a dynamically stable walk.

For the waist controller, we proposed to investigate the traditional Proportional-Integral-Derivative controller. Since this control strategy has been widely adopted to control inverted pendulums, it is also reasonable to expect that it could reach satisfactory results in this domain. For the K_p, K_d and K_i constants tuning, two distinct approaches were investigated: i) The traditional Ziegler-Nichols method, widely used to tune PID parameters, and ii) A Reinforcement Learning

based approach. The RL approach's adoption justifies considering that humanoid robots are subject to complex input patterns, which can make the Ziegler-Nichols method – based on response curves – to have a poor performance in the balance control. A fine-tuning based on the experimentation of the robot in the environment is expected to select more robust controller constants to the robot balance.

5.2 MODEL-FREE PARAMETERS LEARNING

In order to properly tune the set of TFS parameters (T, O_{plvP}, A_{plvP}, O_{knP}, C_{knP}, t_{2k}, A_{plvR} and t_{2p}), a genetic algorithm was proposed. The use of the steady state (JONG, 1990), where some of the best individuals of a population are cloned to the next generation, has proven to be convenient.The steady-state proves to be convenient since the best individuals from the population are cloned for the next generation. In Figure 23, we observed that the fitness function evaluated the population (parents). Then, the roulette selector chooses some individuals based on the value of fitness. These individuals crossed with a one-point crossover operation, and some individuals can mutate (children). Finally, the new population is the best individuals between the parents and children. The steady-state is similar to the basic genetic algorithm, which is discussed in the previous chapters. The main difference is the percentage of the best individuals of the generation stays in the next generation.

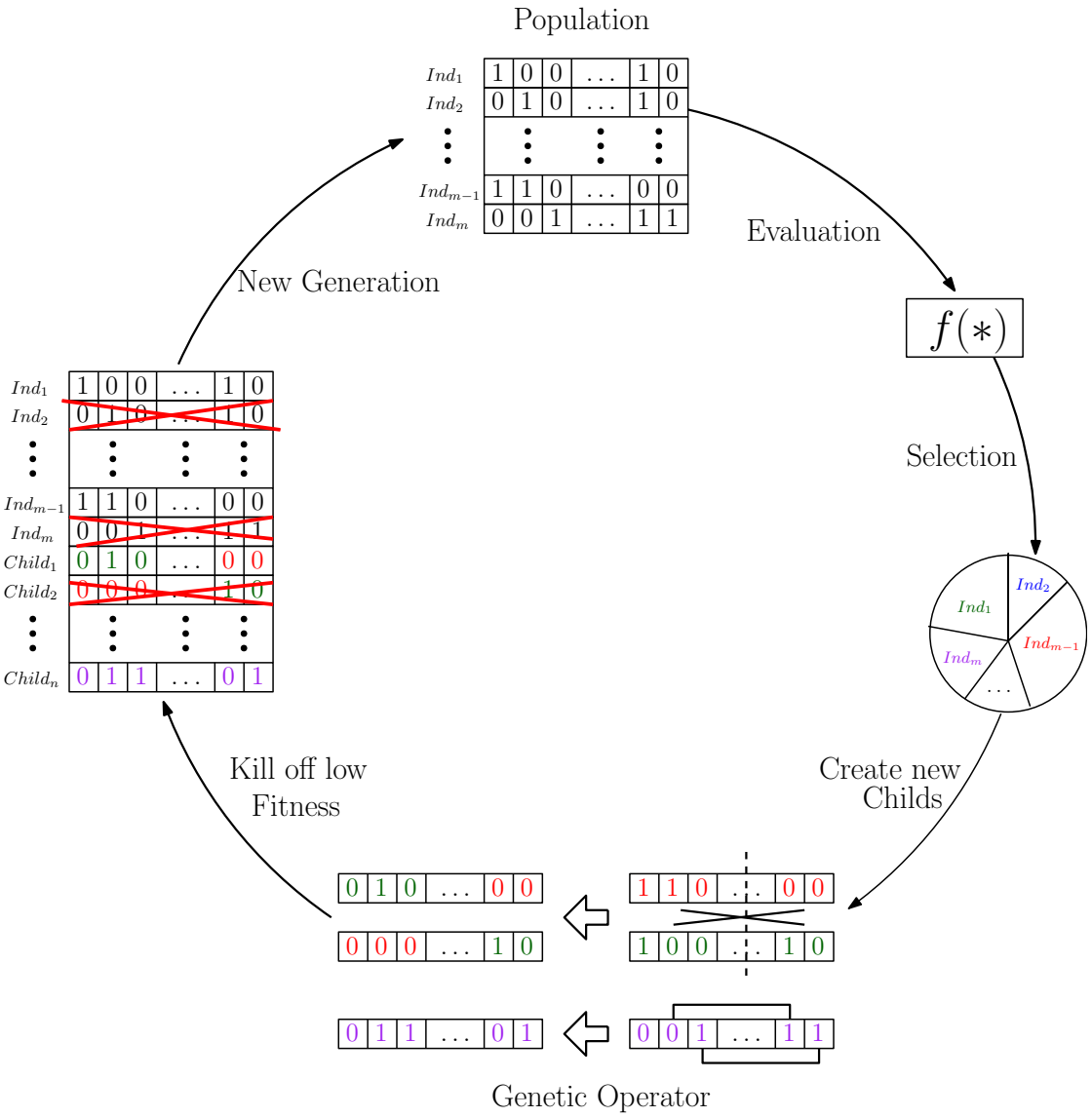


Figure 23 - Genetic algorithm steady state.

6 MATERIALS AND METHODS

"I was taught that the way of progress was neither swift nor easy."
Marie Curie

In this chapter we describe the materials and methods adopted for the experimental procedure. In the following sections are detailed: i) the Marta humanoid robot, ii) the software environments, iii) the robotics simulator, and, finally, iv) the proposed experiments.

6.1 MARTA HUMANOID ROBOT

The main robotics platform adopted in this research is the *Marta* Humanoid robot, designed and built by the work group. Figure 24 presents an image of the physical robot and the robot DOFs, while Table 5 describes the robot specifications. *Marta* has 25 DOF with some innovations in her model that can influence the control process, of which the following are mentioned:

- A spherical joint at the waist inspired in human beings;
- Feet with a small area when compared to other similar robots;
- A joint that simulate the movement of the toes.

Next sections will detail the actuators, sensors and onboard processing capacity of the robot.

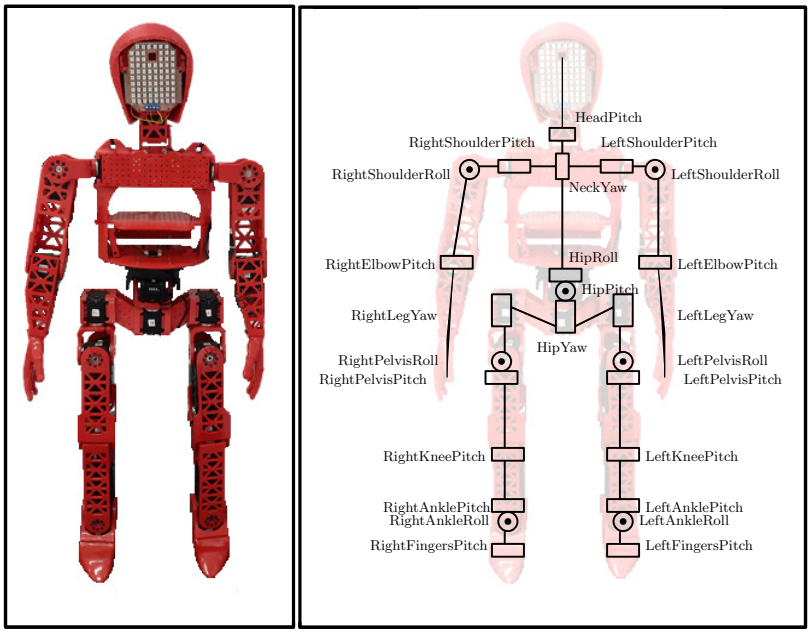


Figure 24 - Marta physical robot (left) and DOFs (right).

Specification	Value
Maximum Height	956mm
Maximum Width	1038mm
Weight	8kg
Leg Width	487mm
Number of Cameras	1 Logitech C920 HD Pro
Degrees of Freedom	25 DOF
Skeleton Materials	ABS plastic
Processor Specifications	Ultratop Brix
Motor Type	Dynamixel MX-64T/AX-12
Sensors	9 axis IMU UM7-LT e Flexi-Force A201
Battery	2 Lipo 4S 14.8V - 4400mAh

Table 5 - Specifications of the dimensions of Marta (CHENATTI et al., 2018).

6.1.1 ACTUATORS

Marta adopts as actuators the Dynamixel AX-12 and Dynamixel MX-64T motors. For the *HeadPitch* and *NeckYaw* motors, Dynamixel AX-12 motors have the following specifications (ROBOTIS, 2019a):

- Degrees of execution: $0^\circ \sim 300^\circ$.
- Weight: $54.6\,g$.
- Size: $32\,mm \times 50\,mm \times 40\,mm$.
- Stop torque: $1.5\,N.m$ (at $12.0\,V$, $1.5\,A$).
- No-load speed: $59\,rpm$ (at $12.0\,V$).

For the remaining twenty-three (23) motors, Dynamixel MX-64T motors were used with the following indications (ROBOTIS, 2019b):

- Degrees of execution: $0^\circ \sim 360^\circ$.
- Weight: $165\,g$.
- Size: $40.2\,mm \times 61.1\,mm \times 41\,mm$.
- Stop torque: $5.5\,Nm$ (at $11.1\,V$, $3.9\,A$), $6.0\,Nm$ (at $12.0\,V$, $4.1\,A$) or $7.3\,Nm$ (at $14.8\,V$, $5.2\,A$).
- No-load speed:$58\,rpm$ (at $11.1\,V$), $63\,rpm$ (at $12.0\,V$) or $78\,rpm$ (at $14.8\,V$).

6.1.2 SENSORS

Marta has two types of sensors: IMU and pressure. The IMU is a UM7 sensor (REDSHIFT-LABS, 2019), which has three-axis accelerometer, rate gyro, and magnetometer. It combines this data using an Extended Kalman Filter (FULA; FERREIRA; OLIVEIRA, 2018) to produce attitude and heading estimates. IMUs were placed at the chest, pelvis and feet of the robot. Also, *Marta* is equipped with seven pressure sensors of the A207 FlexiForce model (TEKSCAN, 2019) placed in each foot. This is a fine and flexible piezoresistive force sensor with the following specifications:

- Thickness: $0.203\,mm$.
- Compromise: $191\,mm$.
- Length: $14\,mm$.
- Detection area: $9.53\,mm$.
- Standard force ranges: $4.4\,N$ $(0-0.45$ kg), $111\,N$ $(0-11.34$ kg) or $445\,N$ $(0-45.36$ kg).

6.1.3 PROCESSOR

We used *Marta*'s computer for this project. This computer is an Ultratop Brix with the following technical specifications:

- Intel Core i7 processor.
- 4GB 1600MHz DDR3 memory expandable to 16GB.
- Connectivity: Two USB 3.0 connectors on the front panel and two USB 3.0 connectors on the back panel.
- Storage: HDD and SSD options.
- Source: Adapter 19v, 65W, 3.2A, automatic bivolt.
- Operative System: Ubuntu 16.04 for 64 bits.
- Graphics: Intel HD graphics 5500, HDMI 1.4 and Mini display port.
- Network: 10/100, 1000Mbps, Wi-Fi 802.11 ac, Bluetooth 4.0.
- Dimensions: $468\,mm \times 107.6\,mm \times 114.4\,mm$.

6.2 PROGRAMMING

All developments were conducted in C++ language using Atom environment running over a Ubuntu Operational System (OS). Figure 25 presents the programming environment and Figure 27 presents the class diagram of the developed code. The Genetic Algorithm (GA) was implemented using GAlib, a object-oriented C ++ library for genetic algorithms created by Matthew Wall (WALL, 1996). GAlib includes various tools for Genetic Algorithm that allow to optimize any representation and genetic operator. An overview of the GALib Class Hierarchy is shown in Fig. 26. Our research is using this library for optimization under the GAlib247 version.

6.3 ROBOTICS SIMULATOR

Considering that ML training processes typically requires the repetition of thousands of millions of movements, it would be impractical to develop such training in the physical robot. In this way, *Marta* has been modeled in a robotics simulator. The main purpose of this modeling is to allow the development of the control algorithms in software before they can be transferred to the real robot. The Virtual Robotic Experimentation Platform (V-REP) was adopted. V-REP is a general-purpose robotic simulator which makes available a free version of the software for research purposes. The platform has portability to Windows, Linux, and Apple OSs. Furthermore, it allows programming in different languages such as Lua, C/C++, Python, Java, Matlab, and Octave. It has different types of physical mechanisms (Bullet

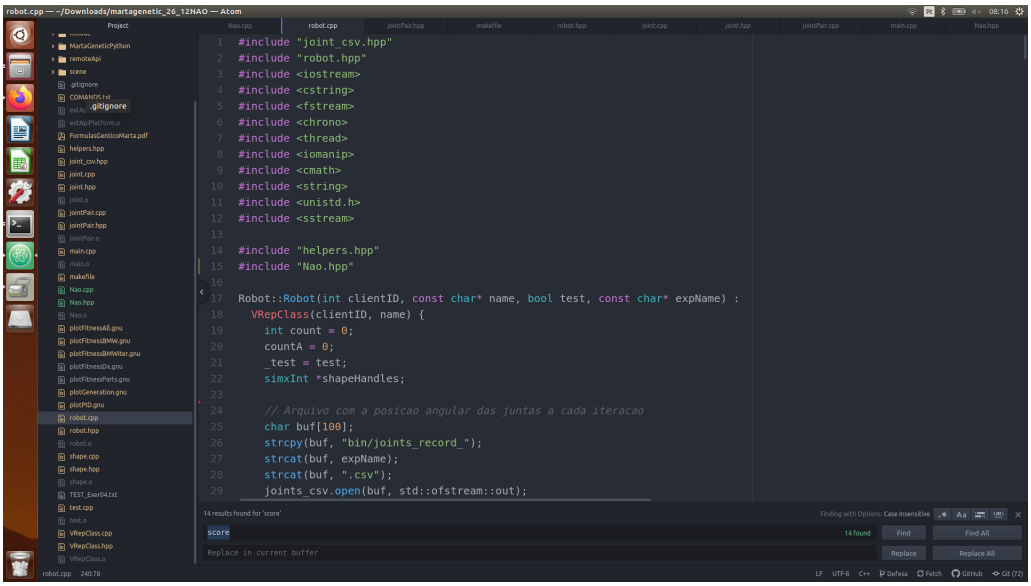


Figure 25 - Programming environment.

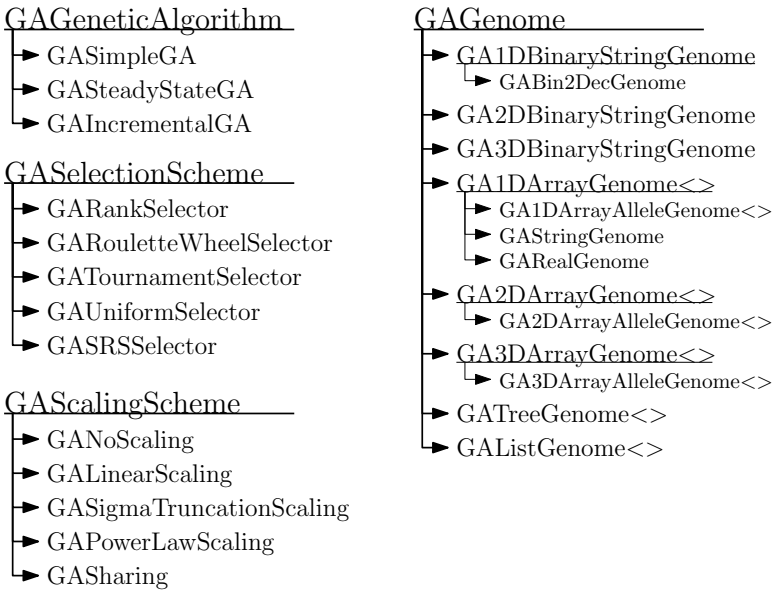


Figure 26 - GALib Class Hierarchy (WALL, 1996).

Open Dynamics Engines (ODE), Vortex Studio and Newton Dynamics). V-REP automatically performs kinematic calculations and interference detection and allows that simulations of visual sensors for image processing (ROBOTICS, 2019) (ROHMER; SINGH; FREESE, 2013). There are different versions of VREP. In this research, *VREP PRO EDU 3.5.0* was adopted. The virtual version of *Marta* was developed by the work group (CHENATTI et al., 2018) with the concern to maintain maximum fidelity to the physical characteristics and limitations of the real robot. Figure 28, presents a simulation of the Marta robot in V-REP and the hierarchy of the components of the robot.

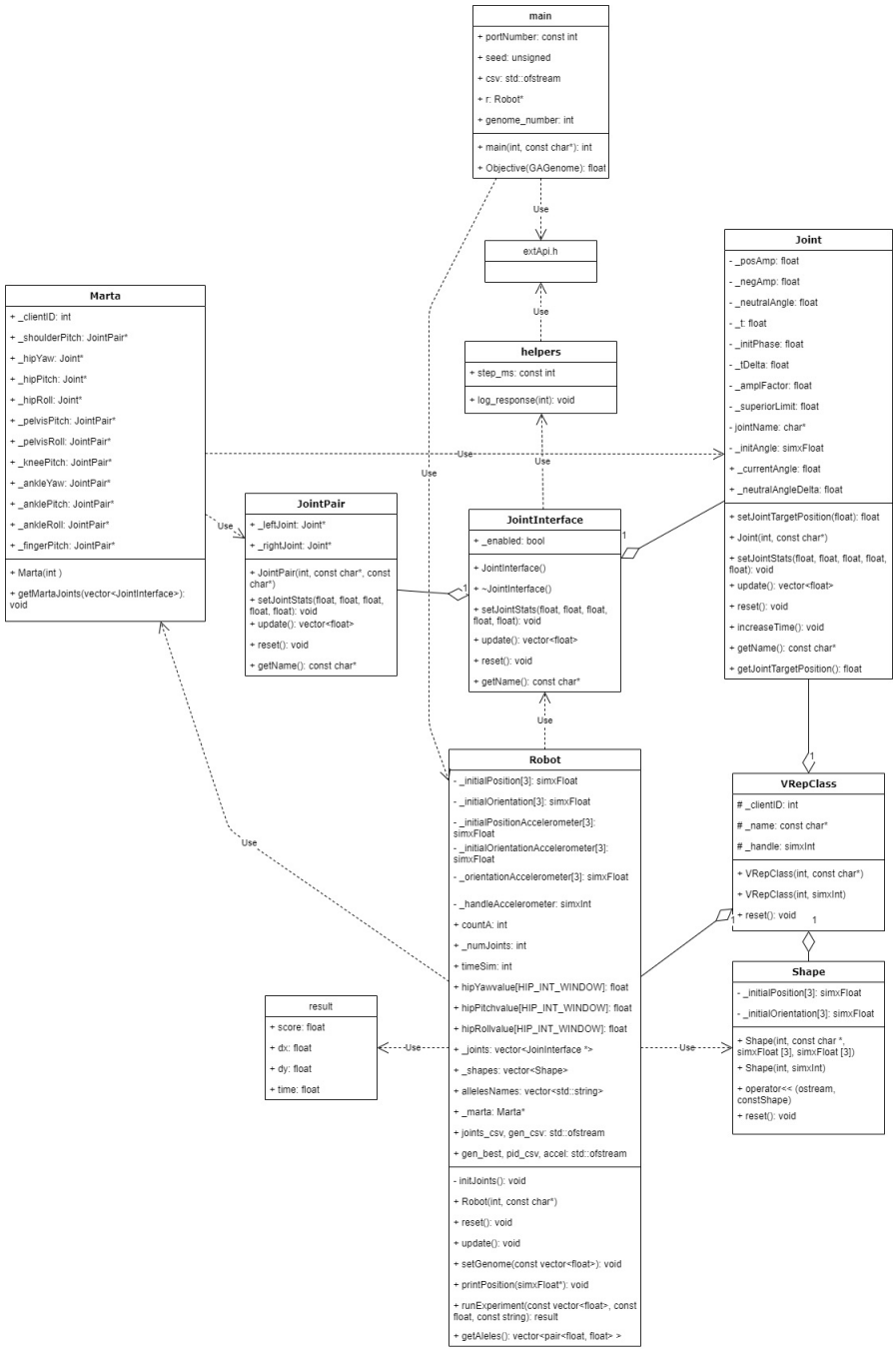


Figure 27 - Class diagram of the developed code.

6.3.1 COMMUNICATION

Since the main code was developed in the C++ programming language, in this research we adopted the remote API to allow the communication between the controller and V-REP. The basic functions used to develop the proposal are:

- *simxStart* and *simxStartSimulation*: establishes the communication channel and initialization of the simulator;
- *simxSetJointTargetPosition*: allows a joint to move to a certain position;

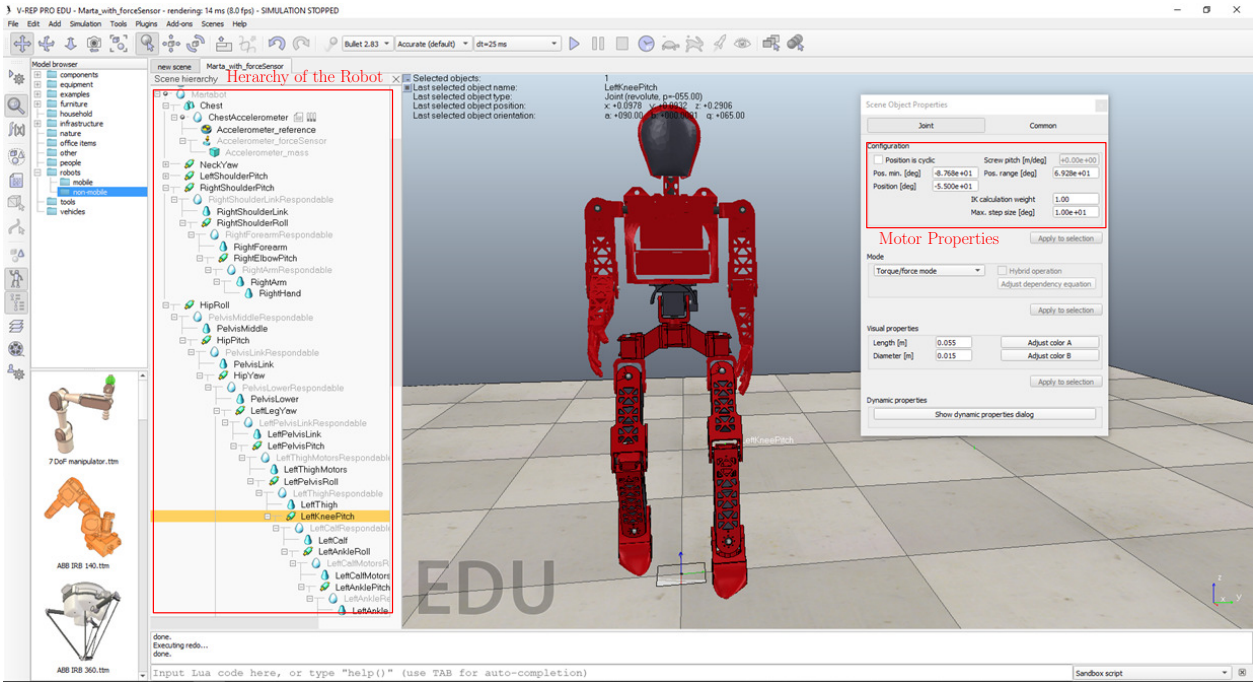


Figure 28 - Marta's virtual model in V-REP.

- *simxSynchronous*: allows the establishment of the communication operation synchronously and block the server;
- *simxGetObjectPosition*: allows to obtain the position of the robot;
- *simxSetJointTargetPosition*: allows to update the position of each active joint of the robot;
- *simxSynchronousTrigger*: sends a signal to run the next step in the simulation.

Figure 29 visually illustrates the communication process in asynchronous update mode between the controller and the simulator. It details each step of the protocol established between the client and server sides of the remote API provided for V-REP simulator. This figure covers the protocol from the moment of the first client request until the beginning of the simulation on the client side.

6.4 EXPERIMENTAL PROCEDURE

A set of experiments were conducted with the virtual version of the *Marta* robot under the V-REP environment in order to evaluate the decoupled controller strategy and the walking patterns obtained in each case. Different controllers for the waist spherical joint were investigated. In the remaining joints, a TFS controller was implemented as discussed in previous sections, adopting the same joints as in previous works (SHAFII et al., 2009a) (SHAFII et al., 2009b) (SHAFII; REIS; LAU, 2010) (MAXIMO; COLOMBINI; RIBEIRO, 2017). The TFS parameters were tuned in each experiment based on a learning processes controlled by a Genetic Algorithm (GA) described in more details in the next section. The experiments had the following configurations:

1. **EXP − 01**. All three DOFs of the spherical joint of the robot waist were fixed. In this way, Marta resembled a traditional robot with a rigid body;

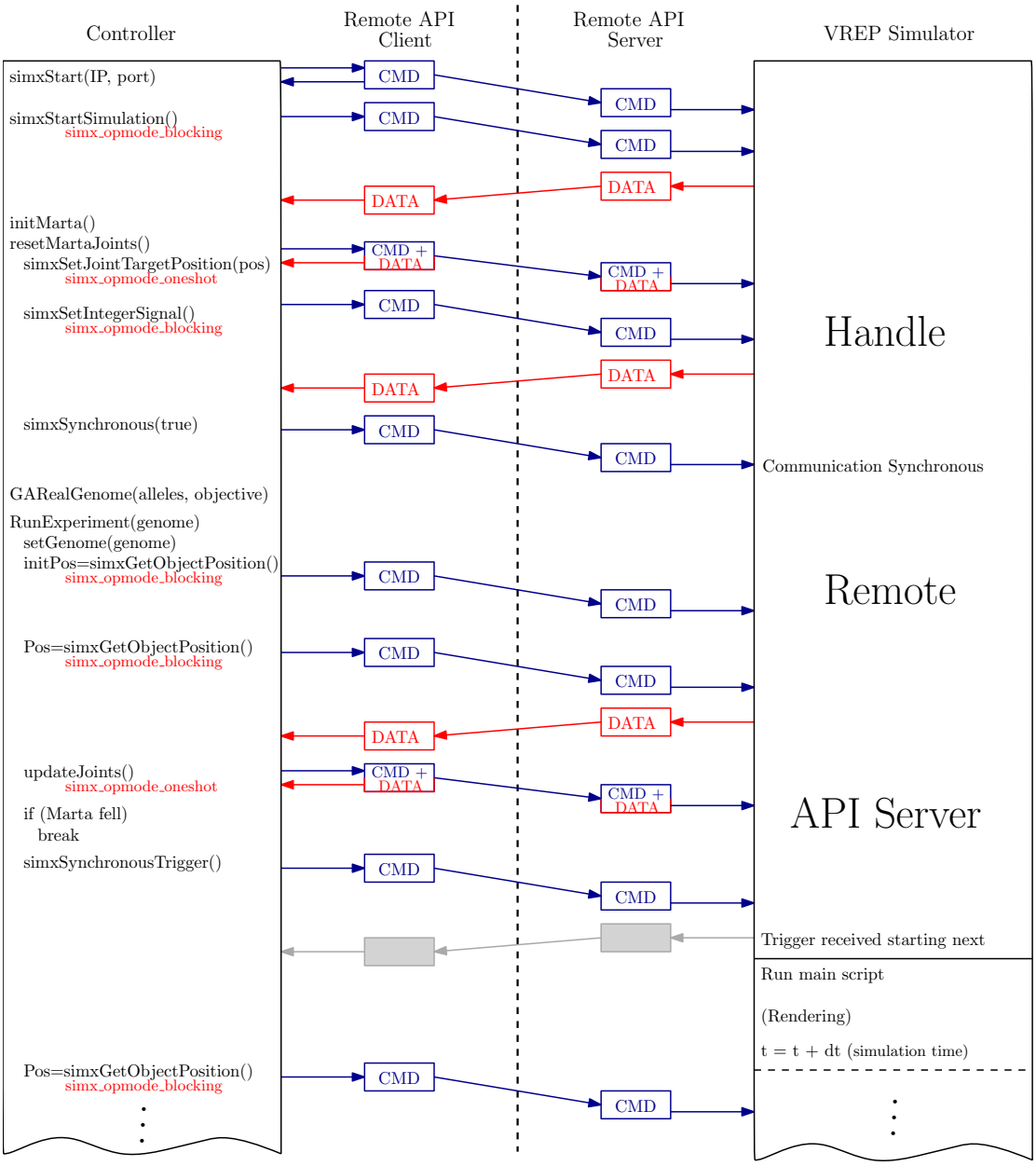


Figure 29 - Sequence diagram between the controller and the simulator.

2. **EXP – 02**. The pitch DOF of the spherical joint was released and controlled by a simple Proportional (P) algorithm with the body reference configured to the vertical position. Roll and Yaw remained fixed;

3. **EXP – 03**. The pitch and yaw DOFs of the spherical joint were released and controlled by a PID algorithm with the body reference configured to the vertical position. Roll remained fixed;

4. **EXP – 04**. The pitch and roll DOFs of the spherical joint were released and controlled by a PID algorithm with the body reference configured to the vertical position. Yaw remained fixed;

5. **EXP – 05**. Pitch, yaw and roll DOFs of the spherical joint were released and controlled by a PID algorithm with

the body reference configured to the vertical position.

In each experiment, the following phases were observed:

1. **Setup phase.** The robot was placed upright and the K_p, K_d and K_i parameters of the PID controller were properly tuned using the Zieglher-Nichols method based on the response curve of the robot's joint;

2. **Learning phase.** For each individual in a population, the robot was automatically placed upright in the V-REP scene configured with a plain surface without obstacles. the ability of the robot to walk without falling (traveled distance) was observed. A **quantitative** analysis was carried out in order to evaluate the ability of the individual to walk. The value of a fitness function based on the traveled distance and time before falling was observed. The simulator was configured to run in headless mode (without GUI) in order to speed up the learning process;

3. **Best individual analysis phase.** A **qualitative** analysis of the walking pattern obtained was carried out observing (using the GUI) the behavior of the best GA individual.

6.4.1 GENETIC ALGORITHM

In the GA a float-type **genome** was adopted, with each allele in the genome representing a distinct value for a TFS curve (amplitude, offset and phase shift) associated to the movements of a particular robot joint. The maximum and minimum values of the alleles were limited according to the range of possible positions for each joint in the real robot respecting their physical limitations. These values are shown in Table 6. The maximum torque of the motors ware configured according to the motor specifications described in the chapter 6.

Parameter	Min (rad)	Max (rad)
T	0.100	2.000
D_+	-0.700	0.700
A_{plvP}	-0.700	0.700
O_{plvP}	-0.700	0.700
A_{plvR}	-0.700	0.700
$t2_{plvR}$	-1.000	1.000
C_{knP}	-1.000	1.000
O_{knP}	-1.000	0.026
$t2_{knP}$	-1.000	1.000

Table 6 - Limits for each genome parameters of the GA.

After some preliminary tests, the following parameters were selected to configure the Galib247 library:

- Population size: 100.
- Maximum generation: 80.
- Seed for population initialization: 100.
- Probability of replacement: 0.8.
- Mutation probability: 0.3.
- Maximum time for each test: 35s.

- Test number for each genome: 3.
- Individuals selection for recombination: Roulette wheel selector.
- Crossover operator: One-point crossover.
- Mutation: Uniform mutation.

6.4.2 FITNESS FUNCTIONS

Some distinct **fitness functions** were investigated in preliminary tests:

1. **FF1** (Forward shift). This fitness function considered only the path (d_x) taken by the humanoid robot forward (negative distances were discarded). The *max* function prevented the robot from learning to walk backwards.

$$FF1 = 100 * max(dx, 0) \tag{110}$$

2. **FF2** (Forward shift and time). This fitness function also considered the time (t, in seconds) the robot remained standing without falling.

$$FF2 = 100 * max(dx, 0) + 0.8 * t \tag{111}$$

3. **FF3** (Discount for diagonal walks). In the third fitness function, displacements in the y axis;

$$FF3 = 100 * (max(dx, 0) - abs(dy)) + 0.8 * t \tag{112}$$

4. **FF4** (Stimulated diagonal). In this case, the diagonal displacement was stimulated and a $penalty$ with value -30 was applied when the robot fell before 15 seconds.

$$FF4 = 100.0 * (d_x + 0.25 * \sqrt{dx^2 + dy^2}) + 0.1 * t + penalty \tag{113}$$

6.4.3 REFERENCE POSITION

Adopting the correct reference for the robot displacement (dx) proven relevant in our preliminary experiments. In our earlier experiments, we considered the difference (in x and y axis) between the initial and final position of the geometrical center of the robot body as the measure of its displacement. However, even when the robot falls immediately in a trial (without walking), it usually received a positive or negative feedback depending on the position in which its body fell to the ground (forward or backward with respect to the initial position). This incorrect score stimulated the learning algorithm to select unwanted behaviors. We fixed this problem adopting the middle point between the left and right ankles – that is, the point as close as possible to the projection of the geometrical center of the robot in the ground – as the robot reference position. With this reference, even when the robot fell immediately (forward or backward), its feet tends to be close to the initial position.

6.4.4 DAMPING FACTOR

Authors in the literature (SHAFII et al., 2009a)(MAXIMO; COLOMBINI; RIBEIRO, 2017) suggested the use of a damping factor in order to attenuate the first cycles of the TFS functions. This procedure may be required since abrupt changes in the joint angles; its can easily cause rapid changes in the actuators' positions, making the system unstable. This problem due to a lack of adaptation of the robot to the first cycles of the TFS; that can not be confused with the instability in the robot's walk caused by unsuitable curve parameters. We obtained better results with a damping factor of 4 cycles; that is, we started the factor with 0% and increased 25% in each cycle until we reached 100%.

6.4.5 ANKLE CONTROL

Ankle control is important in any humanoid robot. However, in Marta (a highly unstable robot), this is a key issue. The main strategy found in the literature focus on keeping the foot parallel to the ground. However, there are different ways to implement this strategy. Some authors (SHAFII et al., 2009a), as mentioned in previous sections, proposed equations for controlling ankle pitch and roll (equations 106 – 109) based on the angles of the pelvis and knee. However, these equations are valid if and only if the robot is stable. When the robot has some significant disturb in its main axis, the use of such angles no longer guarantees that the foot are parallel to the ground. In order to investigate the best ankle control strategy for a naturally unstable robot like Marta, some preliminary tests were conducted:

- **ANK-01.** Equations 106 – 109 were adopted to the ankle control;
- **ANK-02.** Two IMUs were placed in the robot foot (one in each feet), and the ankles were actively controlled based on the IMU orientation signals;
- **ANK-03.** A third IMU was placed in the robot pelvis. The *ankle roll* was controlled based only in the foot IMU signal, while the *ankle pitch* control considered both foot and pelvis IMU, correcting the robot balance (Equation 107) whenever a maximum inclination was exceeded.

7 RESULTS AND DISCUSSION

"Follow your passion and life will reward you."
(Iroh character)
Avatar: The Last Airbender

This chapter presents the results obtained for the experiments with the learning-based model-free controller using the decoupled strategy for humanoid robot walking with the Marta robot.

7.1 PRELIMINARY TESTS

This section presents the results for the preliminary tests conducted with the Marta robot.

7.1.1 WAIST PID TUNING

The results obtained in the configuration phase to adjust the constant PID parameters of the 3-DOF spherical joint on the robot waist using the Ziegler-Nichols method are shown in Table 7. Tuning was performed with the robot in a neutral position. After adjusting the constants, the robot has normally shown itself capable to dynamically keep its balance, as shown in Figure 30.

	K_p	K_i	K_d
Hip Pitch	3.60	0.0262	2.475
Hip Roll	0.36	0.0007	0.900
Hip Yaw	0.60	0.0044	0.413

Table 7 - PID parameters selected for the experiments.

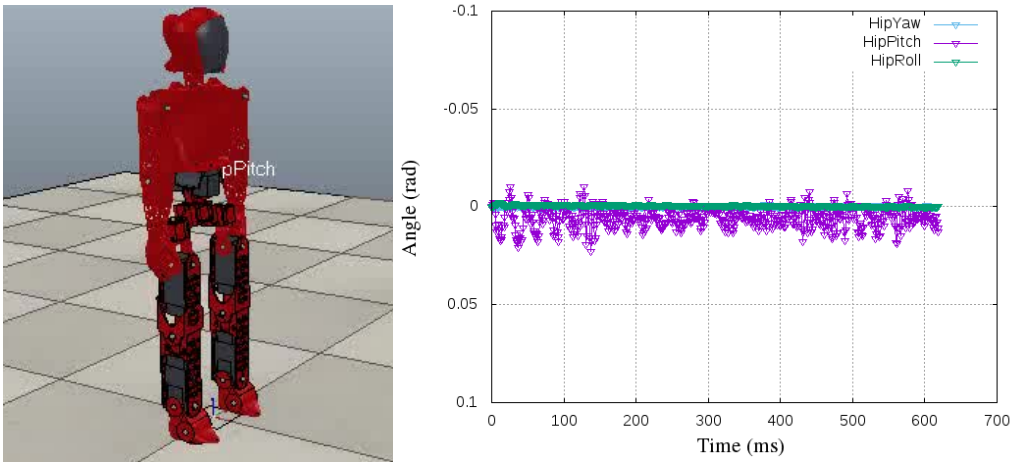
7.1.2 WAIST RL TUNING

Reinforcement learning tuning on *Hip Pitch*, which is the most problematic joint. Therefore, we implemented Q-learning with seven possible actions: stay in the same action, increase or decrease K_p (0.1), K_i (0.001), and K_d (0.1). Our states are 3-dimensional vectors (K_p, K_i, K_d) that are dynamically chosen, depending on the action that increases a new state. We can have n increases or decreases in K_p, K_i, and K_d. Our other parameters of the Q-learning:

- ε: 0.6
- α: 0.1
- γ: 0.9
- Initial state: $[3.6, 0.021, 2.5]$

The result of tuning this joint is given by:

- K_p: 3.8
- K_i: 0.025
- K_d: 2.5



(a) (b)

Figure 30 - Angles of the waist joints after the process of tuning of the PID parameters: a) start position of the robot; b) Angles of the spherical joints over time. Similar to the inverted pendulum, the robot reached the stability with the PID controller after the tuning process.

7.1.3 INFLUENCE OF THE FITNESS FUNCTION

This fitness function FF1 stimulated the robot to walk in small steps (typically $T = 0.1\,s$). The robot never bent its knees as movements with long periods were ruled out when the robot fell. Figure 31 a shows the typical foot trajectories. The fitness function FF2 stimulated the robot to stand up as much as possible and usually caused a frontal fall, that feet trajectory are a small length (Figure 31 b).

On the other hand, the FF3 function led us to a selection of individuals with little variation in the genomes, since the robot at the beginning of the learning process always falls backward or laterally. Figure 31 c shows the typical trajectory pattern stimulated by this function. Finally, in function FF4 with the diagonal is scored. This helps to generate genome diversity for GA. Figure 31 d shows a typical individual selected by GA, capable of taking longer steps.

7.1.4 INFLUENCE OF THE ANKLES CONTROL STRATEGY

Figure 32 shows the experimental results for the different ankle control strategies. Figure 33 shows some examples of the gait pattern obtained in each case. Images 33 a and b showed that one of the feet has a shorter stride length. On the other hand, Image 33 c shows the foot trajectory more stable steps (ANK-03 results). In general, the proposed approach (ANK-03) ensures better convergence when compared with the original equations (SHAFII; REIS; LAU, 2010) (ANK-01) or with the correction based on the IMUs placed on the feet (ANK-02). The strategy in ANK-03 brings more people to the condition of stability.

7.2 DECOUPLED CONTROLLER EVALUATION

Figure 34 presents the average fitness value for each generation during the learning phase for experiments EXP-01 to EXP-05. First of all it is important to observe that the Marta robot is more difficult to control (size, weight and dynamics) when compared to other robots traditionally used in similar experiments like NAO and Robonova. The attempts to tune the TFS parameters even with the ML algorithm without the active spherical joint DOFs in robot waist

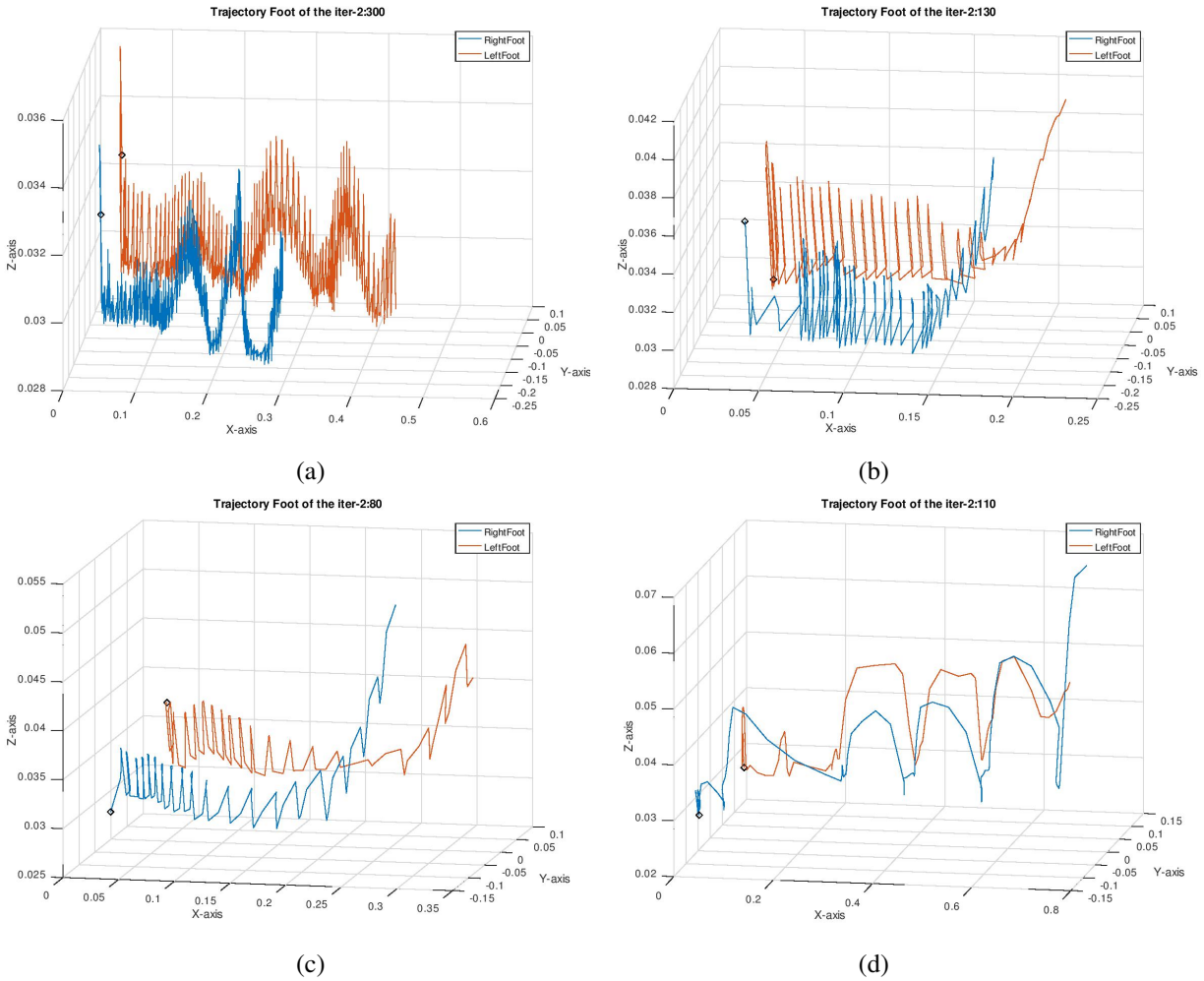


Figure 31 - Body trajectories of the typical individuals selected by the Genetic Algorithm (GA) using distinct fitness functions: a) Forward shift (FF1); b) Forward shift and time (FF2); c) Discount for diagonal shifts (FF3); d) Stimulated diagonal (FF4).

results in loss of stability, and, therefore, in reduced fitness values for the individuals over the generations. The fitness values reached significant values when some of the DOFs were released.

Figures 35 – 39 presents, respectively, the walking patterns generates by the best individuals found by each learning process in the conducted experiments (EXP-01 – EXP-05). The first experiment (EXP-01) resulted in a robot that is unstable for most of the walking patterns generated by the TFS generator. The experiments with some active DOF in the spherical joint in waist resulted in a far more stable robot, and, therefore, in higher fitness values. For all of these control strategies, the robot was able to take a few steps. The better walking pattern was obtained with all DOFs controlled by PIDs.

7.3 DECOUPLED CONTROLLER WITH RL

This test was carried out with the PID correction through reinforcement learning in the Hip Pitch articulation. Image 40 shows Marta's walking pattern and the foot trajectory. This trajectory in the left foot has a maximum step height of

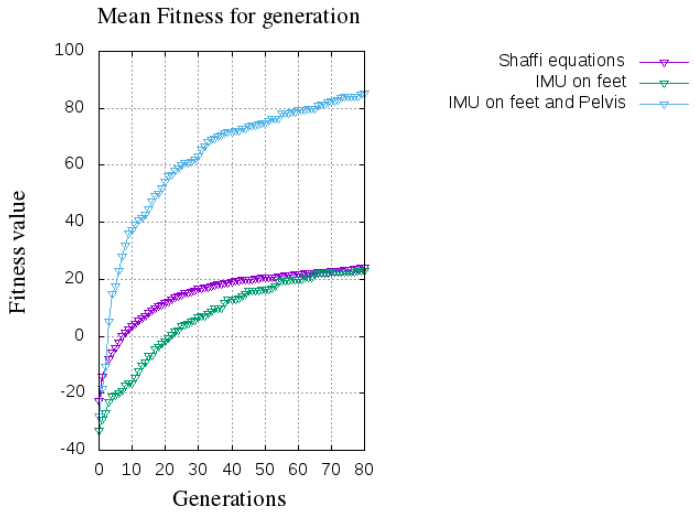


Figure 32 - Average fitness value during the training process for distinct ankles control strategies.

0.09 m., and the right foot has a 0.05 m. before it felt. In this test, we prove that optimize with RL on the Hip Pitch parameters of PID has better results, that Marta walks 0.95m.

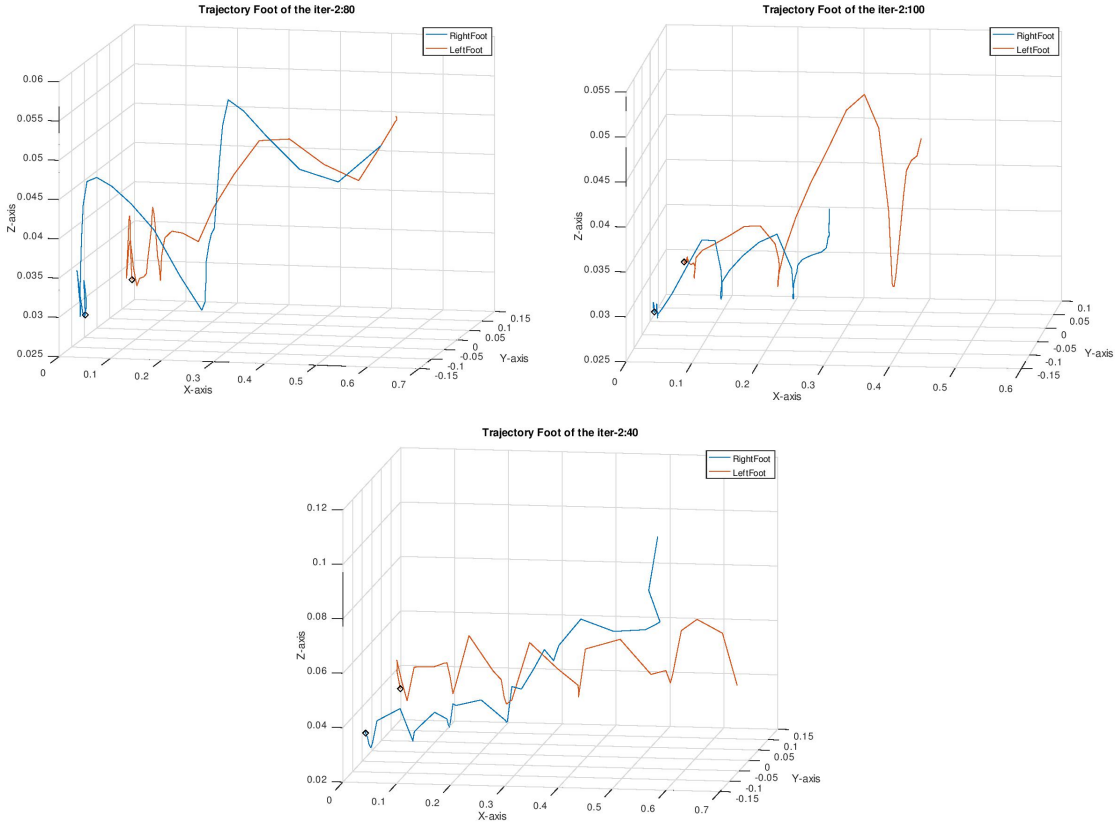


Figure 33 - Walking patterns for distinct ankles control strategy. a) Shafii equations (ANK-01); b) IMUs placed at robot's feet (ANK-02); c) IMUs placed at robot's feet and pelvis (ANK-03).

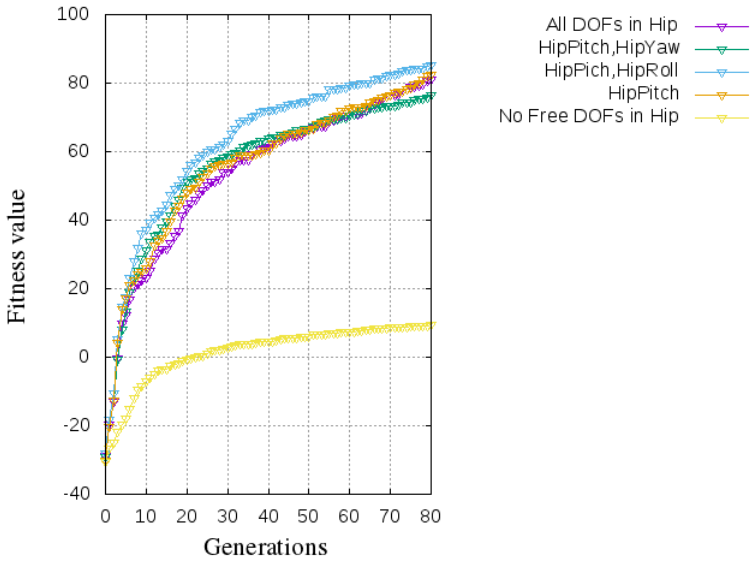


Figure 34 - General evaluation of the decoupled controller strategy. Average values of the fitness function for each generation in the learning phase for EXP-01 to EXP-05.

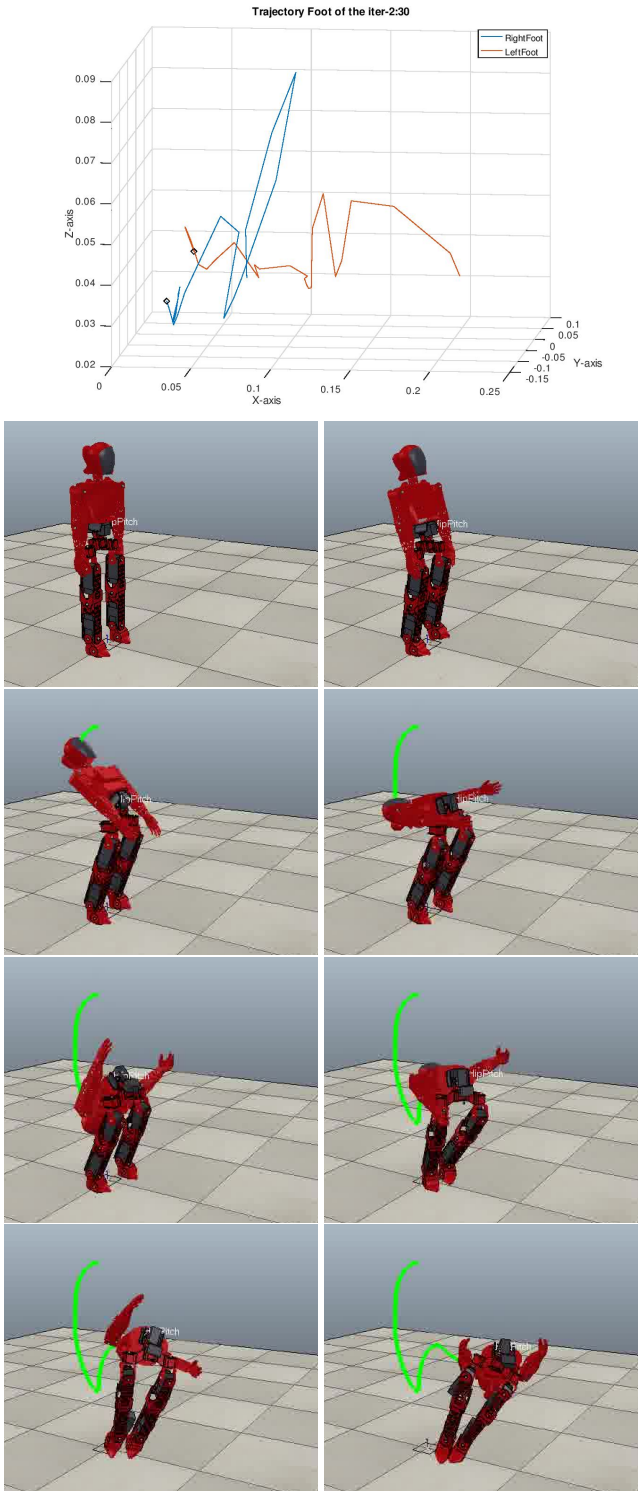


Figure 35 - Movements learned by Marta robot over time in EXP-01 (no DOFs in robot waist). Without an active control the robot is naturally unstable and was unable to learn a walking pattern.

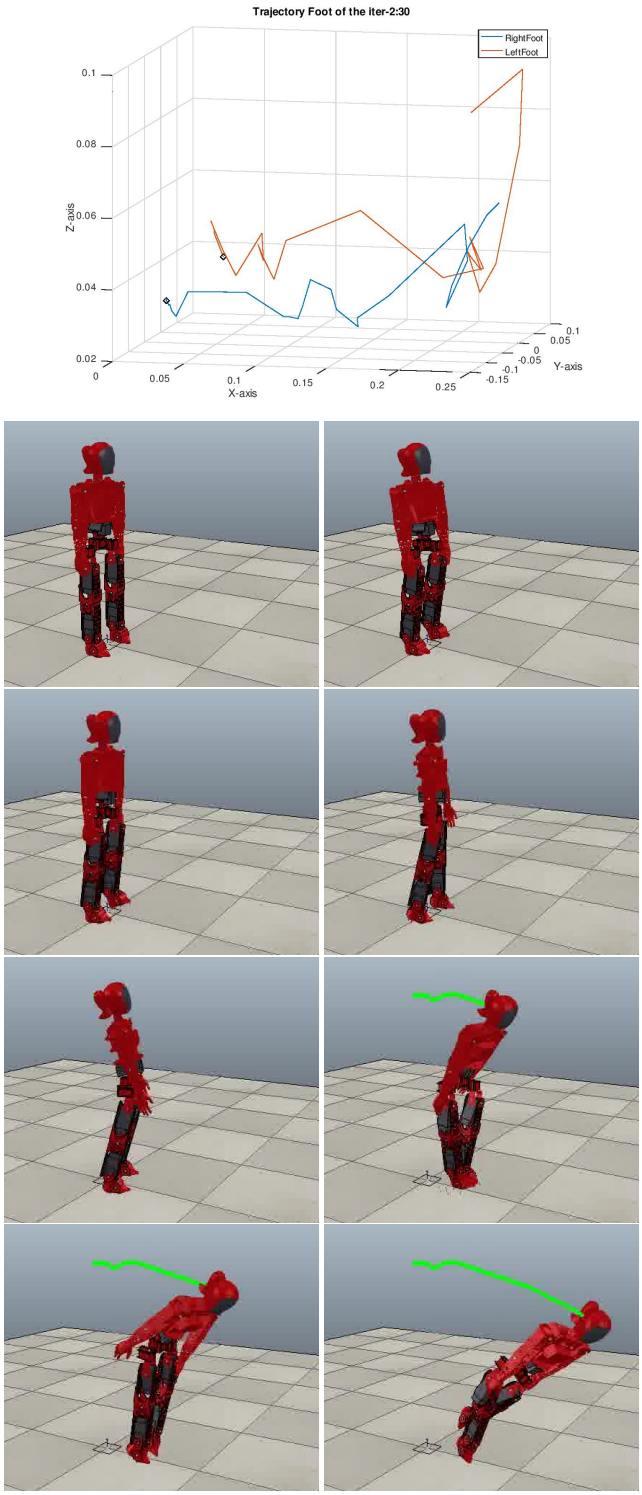


Figure 36 - Movements learned by Marta robot over time in EXP-02 (pitch DOF in waist spherical joint). The robot was able to learn a few steps.

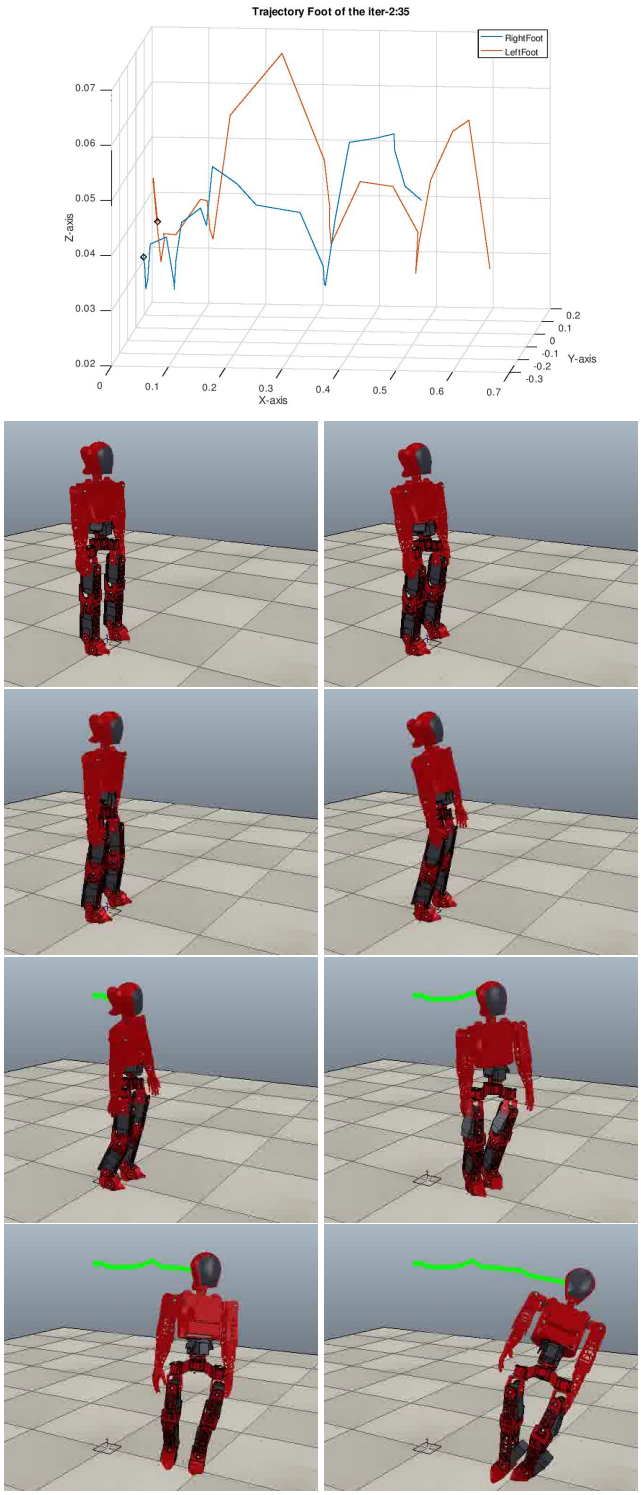


Figure 37 - Movements learned by Marta robot over time in EXP-03 (pitch and yaw DOFs in waist spherical joint). The robot was able to learn a few steps and has a more pronounced lateral displacement.

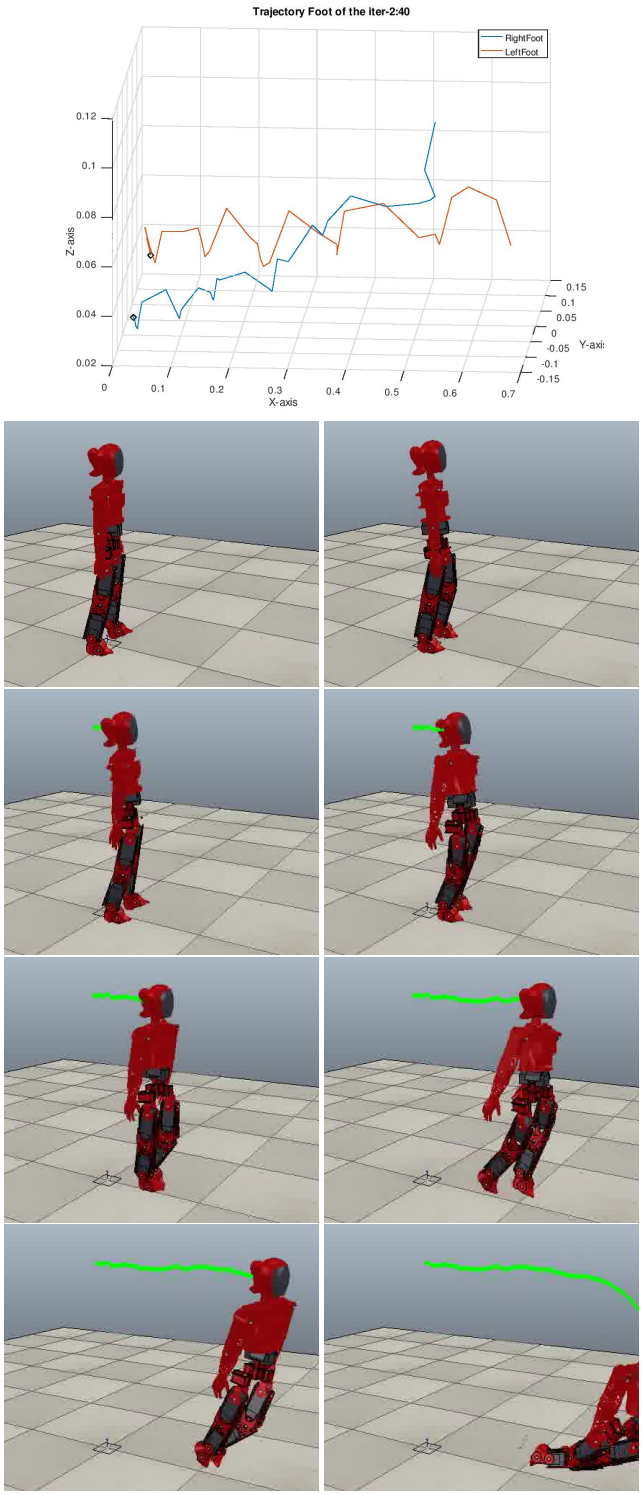


Figure 38 - Movements learned by Marta robot over time in EXP-04 (pitch and roll DOFs in waist spherical joint). The robot was able to learn a few steps.

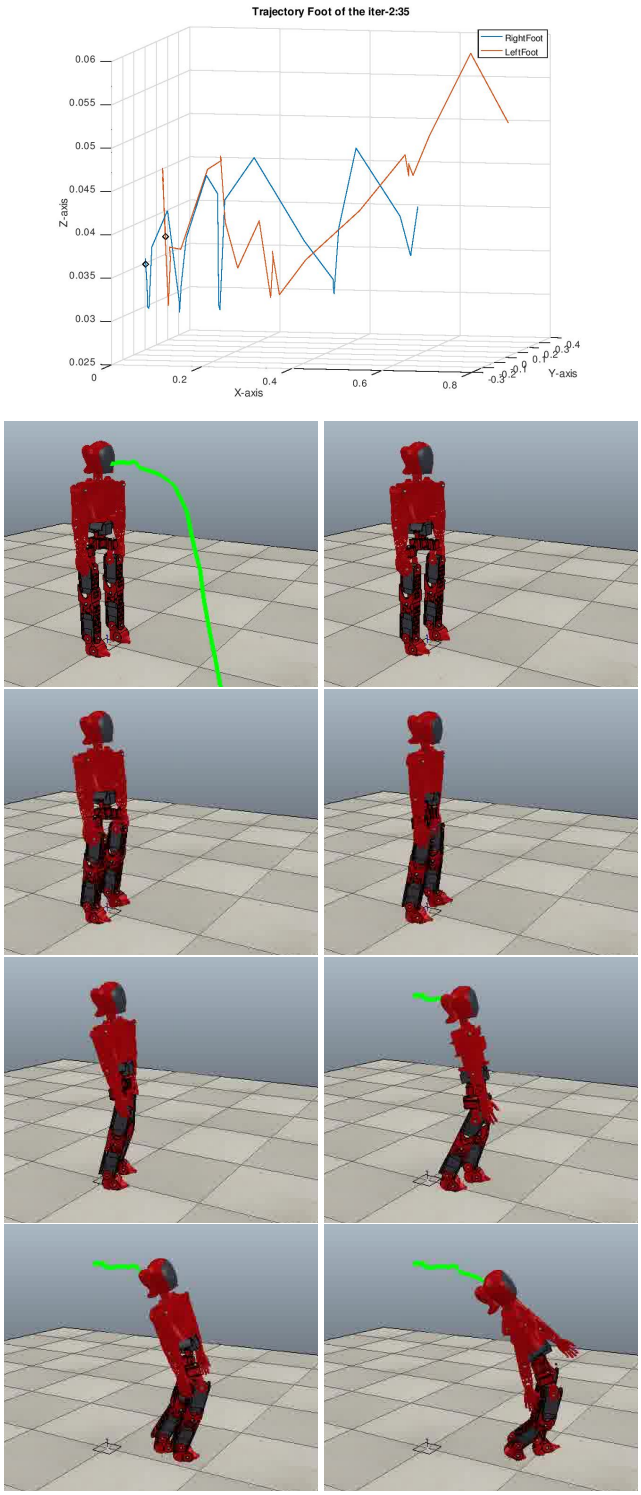


Figure 39 - Movements learned by Marta robot over time in EXP-05 (pitch, yaw and roll DOFs in waist spherical joint). Robot was able to learn a more structured walking pattern before falling.

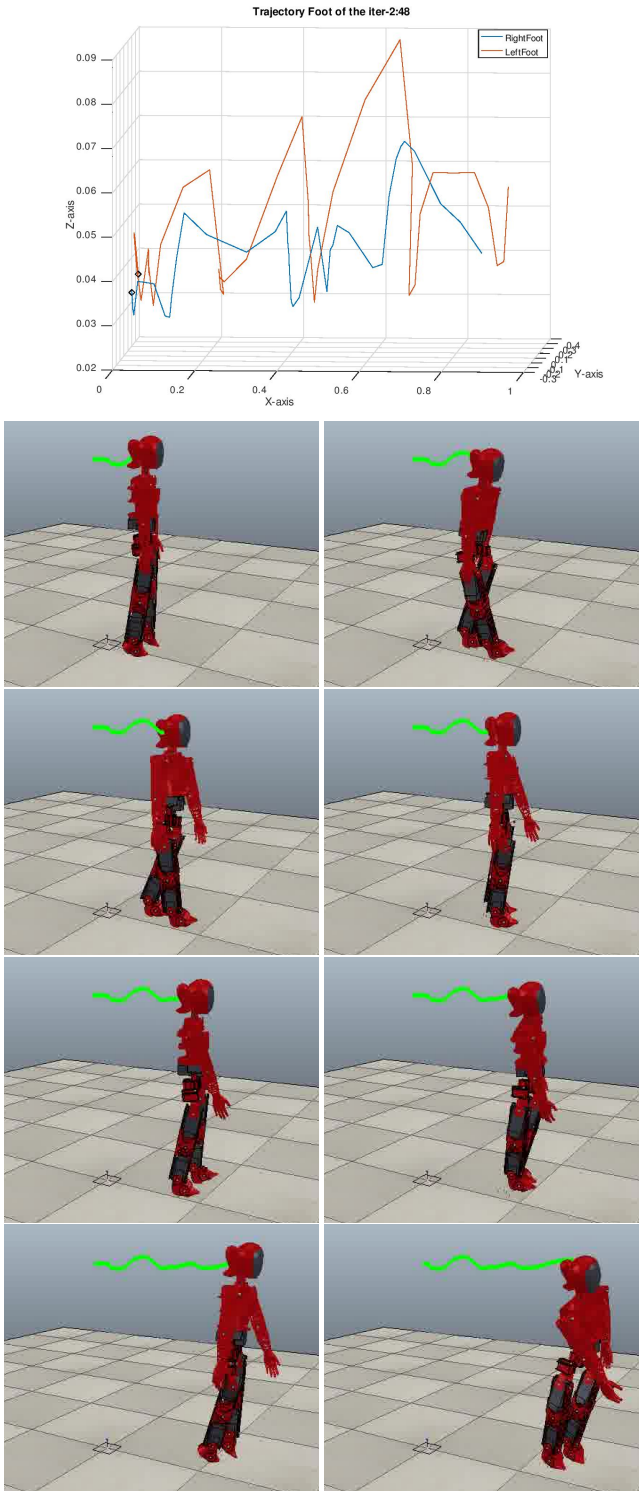


Figure 40 - Movements learned by Marta robot over time in Final Experiment. Robot was able to learn a more structured walking pattern before falling.

8 CONCLUSIONS AND FUTURE WORKS

This work presented a **novel strategy** for controlling humanoid robots walking. This strategy is based on decoupling the robot into two parts: i) an active upper torso with 3-DOFs concerned with keeping the robot's balance, and ii) a model-free controller concerned with generating (or implementing) walking pattern waves for the remaining DOFs. An ML algorithm selects the best walking patterns for a particular robot. Several experiments were carried out, leading us to draw conclusions from different perspectives.

A first aspect is concerning the **evaluation of walking patterns** in humanoid robots. Contrary to what was initially expected, the experiences with the fitness function allow us to conclude that the total distance traveled should not be adopted as a unique parameter for evaluating the evolution of the learning of walking patterns. We obtained better results taking into account the time, and the diagonal distance traveled. Walking patterns derived from these tend to reach greater distances.

Other important aspects concern the feet. According to the experiments carried out, we observed that for a humanoid robot of teenager size, the average position between the heels on the feet is a better **reference position** than the robot's body geometric center. Yet, we also concluded that the adoption of **active control strategies in the ankles** holds great potential for contributing to the maintenance of the robot's balance during walking.

Based on the experiments carried out, we confirm the hypothesis that the insertion of 3 additional DOFs in the humanoid robot waist **allows a more significant number of waveforms generated by the** TFS controller to correspond to stable walks. The control algorithms investigated in this work allowed the Marta robot to walk about 0.6m. In fact, this architecture holds great potential for controlling humanoid robots, and new algorithms build based on these preliminary results can take greater advantage of the new possibilities opened by this architecture.

Among the contributions of this work, we highlight:

- A review of the state-of-the-art regarding approaches to robot walking, presented in Chapter 4;
- The implementation of a computational framework that allowed working with the simulated version of Marta humanoid robot, described in section 6.2;
- The implementation of a classical controller on the waist of the Marta robot, described in section 7.1.1 and the evaluation of the stability in standing and walking described in sections 7.1.1 and 7.2;
- The implementation of reinforcement learning on the hip pith of the Marta robot, described in section 7.1.2, and the evaluation of the walk in the section 7.3;
- The implementation of the optimization of the TFS parameters using Genetic Algorithms for controlling the DOFs of Marta robot described in section 7.2;
- The comparison between the walking pattern achieved by the implemented algorithm to the method proposed by (SHAFII; REIS; LAU, 2010), as show in Figure 34.

As future works, we highlight:

- The investigation of new control strategies for the ankles that can improve the robot's balance during walking;

- The investigation of possible strategies to improve the waist controller after the conclusion of the walking patterns learning phase;
- The investigation of the free joint in the toes, not explored in this work.

⋆⋆⋆⋆⋆

BIBLIOGRAPHY

ABEDI, E.; ALAMIRPOUR, P.; MIRSHAHVALAD, R. Control humanoid robot using intelligent optimization algorithms fusion with fourier series. In: IEEE. *2017 9th International Conference on Computational Intelligence and Communication Networks (CICN)*. [S.l.], 2017. p. 181–185.

ALFARO-CID, E.; MCGOOKIN, E. W.; MURRAY-SMITH, D. J.; FOSSEN, T. I. Genetic programming for the automatic design of controllers for a surface ship. *IEEE transactions on intelligent transportation systems*, IEEE, v. 9, n. 2, p. 311–321, 2008.

ANDREI, N. Modern control theory. *Studies in Informatics and Control*, Citeseer, v. 15, n. 1, p. 51, 2006.

ANTONOVA, R.; RAI, A.; ATKESON, C. G. Sample efficient optimization for learning controllers for bipedal locomotion. In: IEEE. *2016 IEEE-RAS 16th International Conference on Humanoid Robots (Humanoids)*. [S.l.], 2016. p. 22–28.

BACK, T. *Evolutionary algorithms in theory and practice: evolution strategies, evolutionary programming, genetic algorithms*. [S.l.]: Oxford university press, 1996.

BÄCK, T.; SCHWEFEL, H.-P. An overview of evolutionary algorithms for parameter optimization. *Evolutionary computation*, MIT Press, v. 1, n. 1, p. 1–23, 1993.

BARRIENTOS, A.; PEÑÍN, L. F.; BALAGUER, C.; ARACIL, R. *Fundamentos de Robótica*. [S.l.], 1997.

BARTO, A. G. Reinforcement learning control. *Current opinion in neurobiology*, Elsevier, v. 4, n. 6, p. 888–893, 1994.

BENNETT, S. *A history of control engineering 1800-1930*. [S.l.]: IET, 1979. v. 8.

BENNETT, S. *A history of control engineering, 1930-1955*. [S.l.]: IET, 1993.

BUSCHMANN, T.; LOHMEIER, S.; ULBRICH, H. Humanoid robot lola: Design and walking control. *Journal of physiology-Paris*, Elsevier, v. 103, n. 3-5, p. 141–148, 2009.

CANIO, G. D.; LARSEN, J. C.; WÖRGÖTTER, F.; MANOONPONG, P. A combination of central pattern generator-based and reflex-based neural networks for dynamic, adaptive, robust bipedal locomotion. In: SWARM. *1st International Symposium on Swarm Behavior and Bio-Inspired Robotics*. [S.l.], 2016.

CAPEK, K. *RUR (Rossum's universal robots)*. [S.l.]: Penguin, 2004.

CHEN, X.; YU, Z.; ZHANG, W.; ZHENG, Y.; HUANG, Q.; MING, A. Bioinspired control of walking with toe-off, heel-strike, and disturbance rejection for a biped robot. *IEEE Transactions on Industrial Electronics*, IEEE, v. 64, n. 10, p. 7962–7971, 2017.

CHENATTI, S. F.; PREVIATO, G.; TOMAZELA, R.; KOPP, V. G.; BEGAZO, M. F. T.; SALARO, L. G.; ROHMER, E.; COLOMBINI, E. L.; SIMOES, A. da S. Larocs+ unesp team description paper for the ieee humanoid racing 2018. *Latin American Robotics Competition - IEEE Humanoid Racing*, 2018.

COROS, S.; BEAUDOIN, P.; PANNE, M. Van de. Generalized biped walking control. *ACM Transactions on Graphics (TOG)*, ACM New York, NY, USA, v. 29, n. 4, p. 1–9, 2010.

CRAIG, J. J. *Robótica*. 3th. ed. [S.l.]: Pearson Educación, 2006.

CUEVAS, E.; ZALDÍVAR, D.; ROJAS, R. Bipedal robot description. Freire Unversitat Berlin, Fachbereich Mathematik und Informatik, 2004.

DONG, H.; DONG, H.; DING, Z.; ZHANG, S.; CHANG. *Deep Reinforcement Learning*. [S.l.]: Springer, 2020.

DONG, S.; YUAN, Z.; YU, X.; ZHANG, J.; SADIQ, M. T.; ZHANG, F. On-line gait adjustment for humanoid robot robust walking based on divergence component of motion. *IEEE Access*, IEEE, v. 7, p. 159507–159518, 2019.

DURIEZ, T.; BRUNTON, S. L.; NOACK, B. R. *Machine learning control-taming nonlinear dynamics and turbulence*. [S.l.]: Springer, 2017.

ERBATUR, K.; KURT, O. Humanoid walking robot control with natural zmp references. In: IEEE. *IECON 2006-32nd Annual Conference on IEEE Industrial Electronics*. [S.l.], 2006. p. 4100–4106.

FARCHY, A.; BARRETT, S.; MACALPINE, P.; STONE, P. Humanoid robots learning to walk faster: From the real world to simulation and back. In: *Proceedings of the 2013 international conference on Autonomous agents and multi-agent systems*. [S.l.: s.n.], 2013. p. 39–46.

FIGUEROA, A. L.; MEGGIOLARO, M. A. *Modelagem e otimização por algoritmos genéticos de padrões de marcha quase-estática de robôs bípedes planos*. Dissertação (Mestrado) — Pontifícia Universidade Católica do Rio de Janeiro, Rio de Janeiro, Setembro 2016.

FLOREANO, D.; MATTIUSSI, C. *Bio-inspired artificial intelligence: Theories, Methods, and Technologies*. [S.l.]: The MIT Press, 2008.

FULA, J. P.; FERREIRA, B. M.; OLIVEIRA, A. J. Auv self-localization in structured environments using a scanning sonar and an extended kalman filter. In: *OCEANS 2018 MTS/IEEE Charleston*. [S.l.]: IEEE, 2018. p. 1–6.

GAO, Y.; GU, Y. Global-position tracking control of a fully actuated nao bipedal walking robot. In: IEEE. *2019 American Control Conference (ACC)*. [S.l.], 2019. p. 4596–4601.

GOLDBERG, D. E. *Genetic Algorithms in Search, Optimization, and Machine Learning*. [S.l.]: Addison-Wesley Publishing Company, Inc., 1989.

GRIZZLE, J. W.; CHEVALLEREAU, C. Virtual constraints and hybrid zero dynamics for realizing underactuated bipedal locomotion. *arXiv preprint arXiv:1706.01127*, 2017.

GUO, Z.; ZENG, Y.; MENG, C. A trajectory generation method for humanoid robots walking: Optimal force pattern method. In: IEEE. *2019 WRC Symposium on Advanced Robotics and Automation (WRC SARA)*. [S.l.], 2019. p. 103–109.

HAMED, K. A.; GREGG, R. D.; AMES, A. D. Exponentially stabilizing controllers for multi-contact 3d bipedal locomotion. In: IEEE. *2018 Annual American Control Conference (ACC)*. [S.l.], 2018. p. 2210–2217.

HEREID, A.; HUBICKI, C. M.; COUSINEAU, E. A.; AMES, A. D. Dynamic humanoid locomotion: A scalable formulation for hzd gait optimization. *IEEE Transactions on Robotics*, IEEE, v. 34, n. 2, p. 370–387, 2018.

HOLLAND, J. H. et al. *Adaptation in natural and artificial systems: an introductory analysis with applications to biology, control, and artificial intelligence*. [S.l.]: MIT press, 1992.

HRASTINSKI, S.; OLOFSSON, A. D.; ARKENBACK, C.; EKSTRÖM, S.; ERICSSON, E.; FRANSSON, G.; JALDEMARK, J.; RYBERG, T.; ÖBERG et al. Critical imaginaries and reflections on artificial intelligence and robots in postdigital k-12 education. *Postdigital Science and Education*, Springer, v. 1, n. 2, p. 427–445, 2019.

HYON, S.-H.; MORIMOTO, J.; KAWATO, M. From compliant balancing to dynamic walking on humanoid robot: Integration of cns and cpg. In: IEEE. *2010 IEEE International Conference on Robotics and Automation*. [S.l.], 2010. p. 1084–1085.

JONG, K. D. Genetic-algorithm-based learning. In: *Machine learning*. [S.l.]: Elsevier, 1990. p. 611–638.

KAJITA, S.; HIRUKAWA, H.; HARADA, K.; YOKOI, K. *Introduction to Humanoid Robotics*. [S.l.]: Springer, 2014. v. 101.

KATOH, R.; MORI, M. Control method of biped locomotion giving asymptotic stability of trajectory. *Automatica*, Elsevier, v. 20, n. 4, p. 405–414, 1984.

KHADIV, M.; HERZOG, A.; MOOSAVIAN, S. A. A.; RIGHETTI, L. A robust walking controller based on online step location and duration optimization for bipedal locomotion. *arXiv preprint arXiv:1704.01271*, Apr, 2017.

KHAN, U. I.; CHEN, Z. Natural oscillation gait in humanoid biped locomotion. *IEEE Transactions on Control Systems Technology*, IEEE, 2019.

KIM, J.-H.; OH, J.-H. Walking control of the humanoid platform khr-1 based on torque feedback control. In: IEEE. *IEEE International Conference on Robotics and Automation, 2004. Proceedings. ICRA'04. 2004*. [S.l.], 2004. v. 1, p. 623–628.

KOLATHAYA, S.; REHER, J.; HEREID, A.; AMES, A. D. Input to state stabilizing control lyapunov functions for robust bipedal robotic locomotion. In: IEEE. *2018 Annual American Control Conference (ACC)*. [S.l.], 2018. p. 2224–2230.

KONG, J.-S.; LEE, E.-H.; LEE, B.-H.; KIM, J.-G. Study on the real-time walking control of a humanoid robot using fuzzy algorithm. *International Journal of Control, Automation, and Systems*, v. 6, n. 4, p. 551–558, 2008.

LEE, C.; KIM, J.; BABCOCK, D.; GOODMAN, R. Application of neural networks to turbulence control for drag reduction. *Physics of Fluids*, American Institute of Physics, v. 9, n. 6, p. 1740–1747, 1997.

LEIGH, J. R. *Control theory: A guided tour*. 3. ed. [S.l.]: IET, 2012. v. 72.

LEVINE, W. S. *The control handbook*. [S.l.]: CRC press, 1996.

LIU, C.; GENG, W.; LIU, M.; CHEN, Q. Workspace trajectory generation method for humanoid adaptive walking with dynamic motion primitives. *IEEE Access*, IEEE, v. 8, p. 54652–54662, 2020.

MAXIMO, M. R.; COLOMBINI, E. L.; RIBEIRO, C. H. Stable and fast model-free walk with arms movement for humanoid robots. *International Journal of Advanced Robotic Systems*, SAGE Publications Sage UK: London, England, v. 14, n. 3, p. 1729881416675135, 2017.

MENG, L.; MACLEOD, C. A.; PORR, B.; GOLLEE, H. Bipedal robotic walking control derived from analysis of human locomotion. *Biological Cybernetics*, Springer, v. 112, n. 3, p. 277–290, 2018.

MITCHELL, M. *An introduction to genetic algorithms*. [S.l.]: MIT press, 1998.

MITCHELL, T. M. *Machine Learning*. [S.l.]: McGraw-Hill, 1997. (McGraw-Hill International Editions). ISBN 9780071154673.

MOHAMED, S. A.; AWAD, M. I.; MAGED, S. A. Online gait generation for nao humanoid robot using model predictive control. In: IEEE. *2020 International Conference on Innovative Trends in Communication and Computer Engineering (ITCE)*. [S.l.], 2020. p. 205–209.

MOTOI, N.; SUZUKI, T.; OHNISHI, K. A bipedal locomotion planning based on virtual linear inverted pendulum mode. *IEEE Transactions on Industrial Electronics*, IEEE, v. 56, n. 1, p. 54–61, 2008.

MURAI, N.; KANETA, D.; SUGIHARA, T. Identification of a piecewise controller of lateral human standing based on returning recursive-least-square method. In: IEEE. *2013 IEEE/RSJ International Conference on Intelligent Robots and Systems*. [S.l.], 2013. p. 96–101.

NANDI, G. C.; SEMWAL, V. B.; RAJ, M.; JINDAL, A. Modeling bipedal locomotion trajectories using hybrid automata. In: IEEE. *2016 IEEE region 10 conference (TENCON)*. [S.l.], 2016. p. 1013–1018.

NAVEAU, M.; KUDRUSS, M.; STASSE, O.; KIRCHES, C.; MOMBAUR, K.; SOUÈRES, P. A reactive walking pattern generator based on nonlinear model predictive control. *IEEE Robotics and Automation Letters*, IEEE, v. 2, n. 1, p. 10–17, 2016.

NEAPOLITAN, R. E.; JIANG, X. *Artificial intelligence: With an introduction to machine learning*. [S.l.]: CRC Press, 2018.

NIEHAUS, C.; RÖFER, T.; LAUE, T. Gait optimization on a humanoid robot using particle swarm optimization. In: *Proceedings of the Second Workshop on Humanoid Soccer Robots in conjunction with the*. [S.l.: s.n.], 2007. p. 1–7.

OGATA, K. *Modern control engineering*. [S.l.]: Prentice hall, 2010.

OGINO, M.; KATOH, Y.; AONO, M.; ASADA, M.; HOSODA, K. Reinforcement learning of humanoid rhythmic walking parameters based on visual information. *Advanced Robotics*, Taylor & Francis, v. 18, n. 7, p. 677–697, 2004.

OXFORD. *Oxford Advanced Learner's Dictionary*. 9th. ed. [S.l.]: Oxford University Press, 2015.

PAUL, R. P. *Robot Manipulators: Mathematics, Programming, and Control*. 6th. ed. Cambridge, MA, USA, 1984.

PINTO, J. E. M. G. *Aplicação prática do método de sintonia de controladores PID utilizando o método do relé com histerese*. Dissertação (Mestrado) — Universidade Federal do Rio Grande do Norte, 2014.

PRISTOVANI, R. D.; SANGGAR, D. R.; DADET, P. Implementation of push recovery strategy using triple linear inverted pendulum model in "t-flow" humanoid robot. In: IOP PUBLISHING. *Journal of Physics: Conference Series*. [S.l.], 2018. v. 7, n. 1, p. 12–68.

RAJ, M.; SEMWAL, V. B.; NANDI, G. Multiobjective optimized bipedal locomotion. *International Journal of Machine Learning and Cybernetics*, Springer, v. 10, n. 8, p. 1997–2013, 2019.

RAPETTI, L.; TIRUPACHURI, Y.; DARVISH, K.; LATELLA, C.; PUCCI, D. Model-based real-time motion tracking using dynamical inverse kinematics. *arXiv preprint arXiv:1909.07669*, 2019.

REBALA, G.; RAVI, A.; CHURIWALA, S. *An Introduction to Machine Learning*. [S.l.]: Springer, 2019.

RECHENBERG, I. Evolutionsstrategien. In: *Simulationsmethoden in der Medizin und Biologie*. [S.l.]: Springer, 1978. p. 83–114.

REDSHIFT-LABS. *UM7 Orientation Sensor*. <http://www.chrobotics.com/shop/um7-orientation-sensor>, 2019.

REEVES, C.; ROWE, J. E. *Genetic algorithms: principles and perspectives: a guide to GA theory*. [S.l.]: Springer Science & Business Media, 2002. v. 20.

REHER, J.; COUSINEAU, E. A.; HEREID, A.; HUBICKI, C. M.; AMES, A. D. Realizing dynamic and efficient bipedal locomotion on the humanoid robot durus. In: IEEE. *2016 IEEE International Conference on Robotics and Automation (ICRA)*. [S.l.], 2016. p. 1794–1801.

ROBERGE, J. K. *The mechanical seal*. Tese (Doutorado) — Massachusetts Institute of Technology, 1960.

ROBOTICS, C. *Virtual Robot Experimentation Platform - USER MANUAL*. <http://www.coppeliarobotics.com/helpFiles/index.htmle>, 2019.

ROBOTIS. *Dnamixel AX-12A*. <http://emanual.robotis.com/docs/en/dxl/ax/ax-12a/>, 2019.

ROBOTIS. *Dynamixel MX-64T Series (Protocol v2.0)*. <http://emanual.robotis.com/docs/en/dxl/mx/mx-64/>, 2019.

ROHMER, E.; SINGH, S. P.; FREESE, M. V-rep: A versatile and scalable robot simulation framework. In: IEEE. *2013 IEEE/RSJ International Conference on Intelligent Robots and Systems*. [S.l.], 2013. p. 1321–1326.

ROSE, J.; GAMBLE, J. G. *Human walking*. [S.l.]: Williams & Wilkins, 1994.

ROTHLAUF, F. *Representations for Genetic and Evolutionary Algorithms*. 2. ed. [S.l.]: Springer, 2006.

SALEHI, M.; AZARKAMAN, M.; AGHAABBASLOO, M. Optimized joint trajectory model with customized genetic algorithm for biped robot walk. *Journal of Computer & Robotics*, Qazvin Islamic Azad University, v. 11, n. 1, p. 21–29, 2018.

SÁNCHEZ-LACUESTA, J.; PRAT, J.; HOYOS, J. V.; VIOSCA, E.; SOLER-GRACIA, C.; COMÍN, M.; LAFUENTE, R.; CORTÉS, A. Biomecánica de la marcha humana normal y patológica. *Valencia: Instituto de Biomecánica de Valencia*, v. 1, 1993.

SARI, W. P.; DEWANTO, R. S.; PRAMADIHANTO, D. Kinematic and dynamic modelling of"t-flow"humanoid robot for ascending and descending stair. In: IEEE. *2019 International Electronics Symposium (IES)*. [S.l.], 2019. p. 82–87.

SAYARI, M. A.; MASMOUDI, N.; ZAIER, R. Bio-inspired cpg based locomotion for humanoid robot application. In: SPRINGER. *International Conference Design and Modeling of Mechanical Systems*. [S.l.], 2019. p. 910–918.

SEWAK, M. *Deep reinforcement learning*. [S.l.]: Springer, 2019.

SHAFII, N.; ASLANI, S.; NEZAMI, O. M.; SHIRY, S. Evolution of biped walking using truncated fourier series and particle swarm optimization. In: SPRINGER. *Robot Soccer World Cup*. [S.l.], 2009. p. 344–354.

SHAFII, N.; KHORSANDIAN, A.; ABDOLMALEKI, A.; JOZI, B. An optimized gait generator based on fourier series towards fast and robust biped locomotion involving arms swing. In: IEEE. *Automation and Logistics, 2009. ICAL'09. IEEE International Conference on*. [S.l.], 2009. p. 2018–2023.

SHAFII, N.; REIS, L. P.; LAU, N. Biped walking using coronal and sagittal movements based on truncated fourier series. In: SPRINGER. *Robot Soccer World Cup*. [S.l.], 2010. p. 324–335.

SICILIANO, B.; KHATIB, O. Part i: History of humanoid robots. In: GOSWAMI, A.; VADAKKEPAT, P. (Ed.). *Humanoid Robotics: A Reference*. Dordrecht, The Netherlands: Springer Nature, 2019. cap. 1, p. 1–35.

SILVA, C. C.; MAXIMO, M. R.; GÓES, L. C. Height varying humanoid robot walking through model predictive control. In: IEEE. *2019 Latin American Robotics Symposium (LARS), 2019 Brazilian Symposium on Robotics (SBR) and 2019 Workshop on Robotics in Education (WRE)*. [S.l.], 2019. p. 49–54.

SKOGESTAD, S.; POSTLETHWAITE, I. *Multivariable feedback control: analysis and design*. [S.l.]: Citeseer, 2007. v. 2.

SMITH, W. *Dictionary of Greek and Roman biography and mythology*. [S.l.]: CC Little and J. Brown, 1849. v. 3.

SPONG, M. W.; HUTCHINSON, S.; VIDYASAGAR, M. *Robot Dynamics and Control*. Second edition. [S.l.]: John Wiley & Sons, Inc., 2004.

STEIN, K.; HU, Y.; KUDRUSS, M.; NAVEAU, M.; MOMBAUR, K. Closed loop control of walking motions with adaptive choice of directions for the icub humanoid robot. In: IEEE. *2017 IEEE-RAS 17th International Conference on Humanoid Robotics (Humanoids)*. [S.l.], 2017. p. 184–190.

SUTTON, R. S.; BARTO, A. G. *Reinforcement learning: An introduction*. [S.l.]: MIT press, 2018.

SZEPESVÁRI, C. Algorithms for reinforcement learning. *Synthesis lectures on artificial intelligence and machine learning*, Morgan & Claypool Publishers, v. 4, n. 1, p. 1–103, 2010.

TAGA, G.; YAMAGUCHI, Y.; SHIMIZU, H. Self-organized control of bipedal locomotion by neural oscillators in unpredictable environment. *Biological cybernetics*, Springer, v. 65, n. 3, p. 147–159, 1991.

TANG, Z.; ER, M. J. Humanoid 3d gait generation based on inverted pendulum model. In: IEEE. *2007 IEEE 22nd International Symposium on Intelligent Control*. [S.l.], 2007. p. 339–344.

TEKSCAN. *FlexiForce Standard Model A201*. <https://cdn.sparkfun.com/datasheets/Sensors/ForceFlex/FLX-A201-A.pdf>, 2019.

TOMAZELA, R. M.; COLOMBINI, E. L. *A combined model-based planning and model-free reinforcement learning approach for biped locomotion*. Dissertação (Mestrado) — Universidade Estadual de Campinas, São Paulo, Brazil, June 2019.

TURNER, A. J. *The Time Museum. Vol. I: Time measuring instruments. Part 3: Water-clocks, sand-glasses, fire-clocks*. Rockford. [S.l.: s.n.], 1984. v. 1. ISBN 0912947012.

VILLAGRA, J.; BALAGUER, C. A model-free approach for accurate joint motion control in humanoid locomotion. *International Journal of Humanoid Robotics*, World Scientific, v. 8, n. 01, p. 27–46, 2011.

VILLELA, L. F. C.; COLOMBINI, E. L.; TÉCNICO-IC-PFG, R.; GRADUAÇÃO, P. F. de. Humanoid robot walking optimization using genetic algorithms. *Projeto Final de Graduação, Julho*, 2017.

VOSKUHL, A. Producing objects, producing texts: accounts of android automata in late eighteenth-century europe. *Studies in History and Philosophy of Science Part A*, Elsevier, v. 38, n. 2, p. 422–444, 2007.

VUKOBRATOVIC, M.; BOROVAC, B. Zero-moment point – thirty five years on its life. *International Journal of Humanoid Robotics*, World Scientific, v. 1, n. 1, p. 157–173, 2004.

VUKOBRATOVIC, M.; BOROVAC, B.; SURLA, D.; STOKIC, D. *Biped locomotion: dynamics, stability, control and application*. [S.l.]: Springer Science & Business Media, 2012. v. 7.

WALL, M. *GAlib: A C++ Library of Genetic Algorithm Components*. <http://lancet.mit.edu/ga>, 1996.

WANG, R.; HUDSON, S. J.; LI, Y.; WU, H.; ZHOU, C. Normalized neural network for energy efficient bipedal walking using nonlinear inverted pendulum model. In: IEEE. *2019 IEEE International Conference on Robotics and Biomimetics (ROBIO)*. [S.l.], 2019. p. 1400–1406.

YAMAGUCHI, J.; SOGA, E.; INOUE, S.; TAKANISHI, A. Development of a bipedal humanoid robot-control method of whole body cooperative dynamic biped walking. In: IEEE. *Proceedings 1999 IEEE International Conference on Robotics and Automation (Cat. No. 99CH36288C)*. [S.l.], 1999. v. 1, p. 368–374.

YANG, L.; CHEW, C.-M.; ZIELINSKA, T.; POO, A.-N. A uniform biped gait generator with offline optimization and online adjustable parameters. *Robotica*, Cambridge University Press, v. 25, n. 5, p. 549–565, 2007.

YOKOTA, Y. A historical overview of japanese clocks and karakuri. In: SPRINGER. *International Symposium on History of Machines and Mechanisms*. [S.l.], 2009. p. 75–188.

YUAN, Q.; XI, Z.; LU, Q.; LIN, Z. Method and experiment of the nao humanoid robot walking on a slope based on com motion estimation and control. In: SPRINGER. *International Conference on Intelligent Robotics and Applications*. [S.l.], 2017. p. 154–165.