# Development of a Locomotion and Balancing Strategy for Humanoid Robots

Emile Bahdi
*University of Denver*

# Development of a Locomotion and Balancing Strategy for Humanoid Robots

## Abstract

The locomotion ability and high mobility are the most distinguished features of humanoid robots. Due to the non-linear dynamics of walking, developing and controlling the locomotion of humanoid robots is a challenging task. In this thesis, we study and develop a walking engine for the humanoid robot, NAO, which is the official robotic platform used in the RoboCup Spl. Aldebaran Robotics, the manufacturing company of NAO provides a walking module that has disadvantages, such as being a black box that does not provide control of the gait as well as the robot walk with a bent knee. The latter disadvantage, makes the gait unnatural, energy inefficient and exert large amounts of torque to the knee joint. Thus creating a walking engine that produces a quality and natural gait is essential for humanoid robots in general and is a factor for succeeding in RoboCup competition.

Humanoids robots are required to walk fast to be practical for various life tasks. However, its complex structure makes it prone to falling during fast locomotion. On the same hand, the robots are expected to work in constantly changing environments alongside humans and robots, which increase the chance of collisions. Several human-inspired recovery strategies have been studied and adopted to humanoid robots in order to face unexpected and avoidable perturbations. These strategies include hip, ankle, and stepping, however, the use of the arms as a recovery strategy did not enjoy as much attention. The arms can be employed in different motions for fall prevention. The arm rotation strategy can be employed to control the angular momentum of the body and help to regain balance. In this master's thesis, I developed a detailed study of different ways in which the arms can be used to enhance the balance recovery of the NAO humanoid robot while stationary and during locomotion. I model the robot as a linear inverted pendulum plus a flywheel to account for the angular momentum change at the CoM. I considered the role of the arms in changing the body's moment of inertia which help to prevent the robot from falling or to decrease the falling impact. I propose a control algorithm that integrates the arm rotation strategy with the on-board sensors of the NAO. Additionally, I present a simple method to control the amount of recovery from rotating the arms. I also discuss the limitation of the strategy and how it can have a negative impact if it was misused. I present simulations to evaluate the approach in keeping the robot stable against various disturbance sources. The results show the success of the approach in keeping the NAO stable against various perturbations. Finally,I adopt the arm rotation to stabilize the ball kick, which is a common reason for falling in the soccer humanoid RoboCup competitions.

## Document Type
Thesis

## Degree Name
M.S.

## Department
Computer Engineering

## First Advisor
Mohammad Mahoor, Ph.D.

## Second Advisor
Goncalo Martins

Keywords
Humanoid robots, NAO, RoboCup, Robotics, Artificial intelligence

Subject Categories
Artificial Intelligence and Robotics | Robotics

# Development of a Locomotion and Balancing Strategy for Humanoid Robots

A Thesis

Presented to

the Faculty of the Daniel Felix Ritchie School

of Engineering and Computer Science

University of Denver

In Partial Fulfillment

of the Requirements for the Degree

Master of Science

by

Emile Bahdi

March 2018

Advisor : Dr. Mohammad Mahoor

Author: Emile Bahdi
Title: **Development Of A Locomotion and Balancing Strategies For Humanoids Robots**
Advisor: Dr. Mohammad Mahoor
Degree Date: March 2018

# Abstract

The locomotion ability and high mobility are the most distinguished features of humanoid robots. Due to the non-linear dynamics of walking, developing and controlling the locomotion of humanoid robots is a challenging task. In this thesis, we study and develop a walking engine for the humanoid robot, NAO, which is the official robotic platform used in the RoboCup Spl. Aldebaran Robotics, the manufacturing company of NAO provides a walking module that has disadvantages, such as being a black box that does not provide control of the gait as well as the robot walk with a bent knee. The latter disadvantage, makes the gait unnatural, energy inefficient and exert large amounts of torque to the knee joint. Thus creating a walking engine that produces a quality and natural gait is essential for humanoid robots in general and is a factor for succeeding in RoboCup competition.

Humanoids robots are required to walk fast to be practical for various life tasks. However, its complex structure makes it prone to falling during fast locomotion. On the same hand, the robots are expected to work in constantly changing environments alongside humans and robots, which increase the chance of collisions. Several human-inspired recovery strategies have been studied and adopted to humanoid robots in order to face unexpected and avoidable perturbations. These strategies include hip, ankle, and stepping, however, the use of the arms as a recovery strategy did not enjoy as much attention. The arms can be employed in different motions for fall prevention. The arm rotation strategy can be employed to control the angular momentum of the body and help to regain balance. In this master's thesis, I developed a detailed study of different ways in which the arms can be used to enhance the balance recovery of the NAO humanoid robot while stationary and during locomotion. I model the robot as a linear inverted pendulum plus a flywheel to account for the angular momentum change at the CoM. I considered the role of the arms in changing the body's moment of inertia which help to prevent the robot from falling or to decrease the falling impact. I propose a control algorithm that integrates the arm rotation strategy with the on-board sensors of the NAO. Additionally, I present a simple method to control the amount of recovery from rotating the arms. I also discuss the limitation of the strategy and how it can have a negative impact if it was misused. I present simulations to evaluate the approach in keeping the robot stable against various disturbance sources. The results show the success of the approach in keeping the NAO stable against various perturbations. Finally, I adopt the arm rotation to stabilize the ball kick, which is a common reason for falling in the soccer humanoid RoboCup competitions.

# Acknowledgements

Foremost, I would like to express my sincere gratitude to my advisor Dr. Mohammad Mahoor for the continuous support of my master's study and research, for his patience, motivation, enthusiasm, and immense knowledge. His guidance helped me in all the time of research and writing of this thesis.

My sincere thanks also goes to Dr. Goncalo Martins for offering me the help and motivation.

I would also like to thank the rest of my thesis committee, Dr. Matthew Rutherford and Dr. Mark Siemens.

I thank my fellow labmates in the Computer Vision lab at the University of Denver . . .

# Contents

# List of Figures

# Abbreviations

| | |
|---|---|
| **CoM** | Center of Mass |
| **CoP** | Center of Pressure |
| **ZMP** | Zero Moment Point |
| **FZMP** | Fictitious Zero Moment Point |
| **GCoM** | Ground Projection of Center of Mass |
| **DSP** | Double Support Phase |
| **SSP** | Single Support Phase |
| **DoF** | Degeers of Freedom |
| **IMU** | Inertial Measurement Unit |
| **HAL** | Hardware Abstraction Layer |
| **SPL** | Standard Platform League |
| **FSR** | Force Resistive Sensors |
| **API** | Apllication Programming Interface |
| **SDK** | Software Development Kit |
| **ROS** | Robotic Operating System |
| **LIPM** | Linear Inverted Pendulum |
| **IPM** | Inverted Pendulum |
| **VCP** | Virtual Contact Surface |
| **TDMA** | Tridiagonal Matrix Algorithm |
| **D-H** | Denavit–Hartenberg |
| **DCM** | Device Communication Manager |
| **ARS** | Arms Rotation Strategy |

# Physical Constants

| | |
|---|---|
| $\pi$ | 3.14159265359 |
| Gravity acceleration | 9.80665 m/s2 |

# Symbols

| | | |
|---|---|---|
| $a$ | distance | m |
| $P$ | power | W ($\mathrm{Js}^{-1}$) |
| | | |
| $\omega$ | angular frequency | $\mathrm{rads}^{-1}$ |

# Chapter 1

# Introductions

**Robotics**: is "a branch of engineering that involves the conception, design, manufacture, and operation of robots. This field overlaps with electronics, computer science, artificial intelligence, mechatronics, nanotechnology, and bioengineering".

**Artificial Intelligence**: "The capability of a device to perform functions, which are normally associated with human intelligence, such as reasoning and optimisation, through experience".

Wheeled robots are more popular than humanoids because of its stable structure, simple motion and for being cost effective. These robots are moved by merely rotating the wheels in particular direction and speed. However, Wheeled robots require unique environments to work in because it cannot move on stairs or on narrow paths. On the other hand, humanoid robots are structured similarly to the human body. Its motion is achieved using only its two legs which is know as the locomotion. The locomotion ability and high mobility are some of the advantages of humanoids robots over the wheeled robots. Additionally, the human-based structure makes the humanoid robots more acceptable to humans which increases the human-machine interactions. Humanoid robots can be programmed to perform almost any of the human actions without the need for any changes. The potential behind using such robots is that it could replace the human labor is dangerous and risky tasks such as firefighting or mineral mining. Thus, humanoid robots are required to have a reliable, fast locomotion and robust adaptive balancing controllers to accomplish the goal.

Several popular humanoid platforms can be used for research purposes. Such robots include the famous ASIMO, TOPIA,and NAO, see figure 1.1 . The Aldebaran NAO has gained its popularity after it was chosen as the official robot for the RoboCup SPL competition.

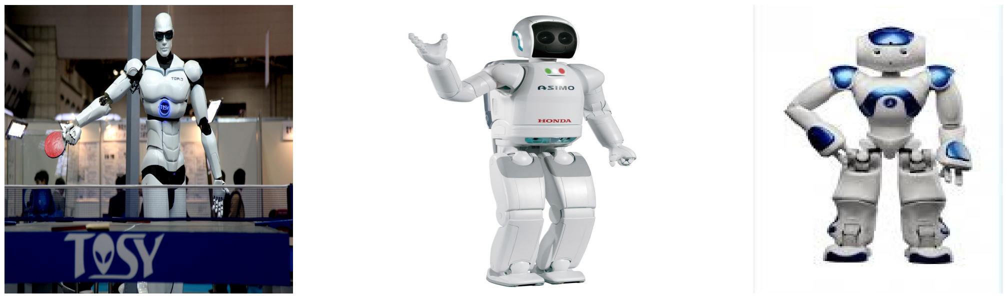


FIGURE 1.1: Different Humanoid models.

Humanoid robots are usually described by the number of degrees of freedom (DOF) which indicate the number of joint actuators. Bipeds robot usually have between 3 to 7 DOF for each leg, that is at least one joint for the hip; one for the knee and one for the ankles. However, the number of DOF in the legs restricts the kinematic space and limit the number of possible motions the can be achieved. In this study I am using The Aldebaran NAO H25 robot has 25 degrees of freedom, five in each leg and hand, two at the head, one in each hand and one for the pelvis.

The robots are usually equipped with various amount of sensors embedded to the CPU. The sensors provide feedback about the surrounding environment allowing the robot to operate robustly. Typical sensors include inertial measurement unit (IMU) that is attached to the body, to detect changes in the orientation. A Force Sensitive Resistors (FSR) to detect the contact of the feet with the ground. Additionally, Several Cameras are used for different vision applications. Feedback from the actuators is conceived by reading the currents, which provide information about the position and torque of the actuators.

A humanoid is expected to perform several tasks in a parallel fashion. The tasks can be programmed as modules that are activated when needed. Programming the robot is accomplished in two separate layers; the low-level layer know as the Hardware Abstract Layer ( HAL) is a low-level library that defines the hardware and software interfaces. It provides the memory address to all the hardware component such as sensors and actuators. These memory addresses contain readings measured by the sensors, as well as, the desired joint values are written to it before it is actuated. The higher level layer contains modules and functions that interact with the hardware. This layer provides high-level APIs to be used by developers. The Aldebaran company provide the required two-layer in its Software Development Kits for the NAO. However, advanced users usually write their own HAL and APIs for sophisticated applications.

## 1.1 Motivation

Humanoid robots are evolving with time to become more capable than the previous generations, because of the availability of newer technologies at reasonable costs. Such technologies include more capable batteries, powerful computers with low power consumptions and more reliable actuators. These evolvements are renewing the biped locomotion and stability researches and challenges.

Wheeled robots are cheaper to implement and more accessible to operate. It can be developed to accomplish the various amount of tasks. However specific environments would require lots of improvement to be suitable for the wheeled robot operators. Raibert et al. [1] mentioned that conventional wheeled robots surpass on surfaces like roads, but present poor mobility in uneven or spongy terrains, thus they can only access half of the earth's landmass.

In contrast humanoid robot moves in term of locomotion, which enhance its mobility allowing it to overcome complex terrains. Huang et al. stated that that biped robots possess higher mobility than wheeled robots, especially when moving in rough terrains [2].

Humanoid robots are designed with a structure similar to the humans and are ready to be programmed to perform most of the human's tasks. Since most of the work-suitable robots are expensive, it is expected that industries would gradually switch its workforce to robots. The transition would start by adopting few robots alongside the human labor until the whole workforce get replaced by robots. During the transition, the robots would have to work alongside humans in human-suitable environments. Moreover, it would not be cost effective to replace a whole working environment just for few robots. Humanoid robots can easily be integrated in a human working environment comparing to other robots. These robot are expected to gain massive popularity shortly, and it would be the most helpful kind of robots for humans. Additionally, humanoids are expected to make a big difference in our lives and would dominate most of the human labor force in the near future.

These desired goals require the humanoids to operate with precise motion and great balancing capabilities. Without having robots that could move in similar skills level as the humans, we can not move closer to the goal. The RoboCup competition is encouraging engineers and developers around the world to improve the humanoids skills so that it could compete with humans [3]. So improving the locomotion and stability is a vital research topic that remains open and is attracting many researchers.

### 1.1.1 RoboCup

RoboCup is an annual robotics competition that aims to promote robotics and AI research with a final long-term goal of having autonomous humanoid robots playing soccer against the winner of the most recent World Cup in according with the FIFA rules. RoboCup aims to achieve its final goal by 2050 which may sound ambitious as building and programming a humanoid soccer player require a significant amount of research and development. The competition is an open source, and the winning teams are expected to publish most of their codes so that new participating organizations would not need to start from the scratch. Every year the competition is held in a different country with new unique challenges. Six major competition domains are divided into leagues and sub-leagues. The RoboCup soccer is the most popular, and it is consists of many leagues such as the Standard Platforms (SPL), Small Size, Middle Size and Simulation Leagues. Other domains include RoboCup Rescue, RoboCup @Home, RoboCup @Work, RoboCup Logistics League and RoboCup Junior. The Standard Platform League restrict the participating teams on using identical robots to focus on the software development rather than the hardware and mechanics of the robots. Another rule of the SPL is that the robots are to operate autonomously without any inputs from humans or remote computers. The Four-legged Sony AIBO robot shown in figure 1.2, was used for the RoboCup SPL until 2006 when its production was discontinued. After that, the humanoid Aldebaran NAO robot replaced AIBO and was assigned to be the official RoboCup SPL robot.

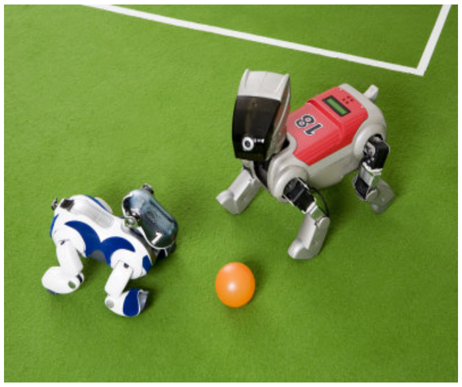


Figure 1.2: Aibo robot by Sony at the RoboCup Competition

The RoboCup Competition is one domain that has been motivating humanoids research and development. The RoboCup SPL is specific is encouraging researchers to improve the locomotion and balance techniques for the humanoid so that it qualifies to compete in against humans soccer teams.

## 1.2 Contributions

The main contribution of the thesis is the following:

1. Develop a study on implementing a multidirectional walking engine for the NAO robot. The study includes realizing different essential modules required to perform a ZMP-based biped locomotion. I analyze and compare different approaches for solving the cart-table/inverted pendulum models in order to produce a smooth CoM trajectory. The active balance module is enhanced to become adaptive instead of constant. Additionally, I create a timing module that is responsible for synchronizing all the engine's together in the correct form while taking the motion frequency in consideration.

2. Investigate how different simulators that support the NAO can be bridged with NAOqi framework. I also developed a comparison between three simulators (Webots, V-rep, and Gazebo) concerning practicality, the ease of use and the supported programming languages and plug-ins flexibility.

3. Develop a study about the different ways in which the arms can be used for balance recovery of the biped robot. The study includes changing the arms position dynamically to enhance the static stability of the robot. Additionally, I purpose a control algorithm that integrate the arm rotation strategy with on-boared sensors of the NAO for balance recovery. The algorithm is responsible for generating arm rotations to prevent falling. The approach uses sensory feedback to evaluate the state of the robot body and employs the arms rotation strategy for fall prevention. I present different simulations scenarios to evaluate our proposed approach. A method to control the angular momentum produced by the arms rotation is presented in addition to the strategy limitation.

4. Develop a powerful ball kick for the NAO, while employing the arm rotation to stabilize the robot body during the kick. I also perform simulations to evaluate the strategy contribution toward stabilizing the robot during the kick.

## 1.3 Thesis Structure

This thesis is divided into five chapters, in which the first one is composed by this introduction. The remainder of this thesis is organized as follows:

**Chapter 2**: Provide a comprehensive review of the literature related to humanoid robots locomotion and balancing strategies. I introduce central concepts related to the gait stability. I discuss models employed to simplify the dynamics of the robot alongside the solution approaches. The gait phases are also analyzed. I review different approaches related to balancing the biped robot. Finally, I address some of the simulators that support the platform model.

**Chapter 3**: Present the process of developing a ZMP based walking engine. Each module of the walking engine is presented with its working mechanism. I present our timing module that is responsible for synchronizing all the modules of the engine together in the correct fashion. Finally, I present the integration of all the modules to produce the desired gait.

**Chapter 4**: Present our contribution to the thesis regarding the fall prevention using arm motions. I present a control algorithm that make use of the developed arm rotation strategy to keep the robot in balance with the presence of external disturbances. A method is proposed to control the amount of recovery from the strategy. Additionally, I examine the approach limits and provide simulations that evaluate our approach. Finally, I examine the use of arm rotation to stabilize the ball kick and prevent the consequence of falling, alongside evaluating simulations for the approach.

**Chapter 5**: summarizes the main conclusions of this thesis. Also, some future perspectives are discussed.

# Chapter 2

# Literature Review and Background

## 2.1 Locomotion Stability Measurements

Biped Robots are distinguished from others by their ability to move using its two legs. The locomotion feature makes the humanoid robots able to work in any human-suitable environment without the need to change any settings. Wheel driven robots are more popular because its more stable and controlling its motion is much easier; however, it can not move over objects or climb a stair. The complex non-linear structure of the bipeds makes it prone to lose balance while in motion. That's said, balancing the humanoid during locomotion remains an active areas of research. The stability of the humanoid gait can be evaluated by measuring the relative distance between specific points on the ground. In this section, I define these points and explain how it could be used to assess the stability of the gait anytime.

### 2.1.1 Center of Mass

The physical definition of the center of mass for a distributed mass body is the unique point where the weighted relative position of the distributed mass sums to zero [4]. In other words, it is a theoretical point where the entire mass of an objected can be assumed to be concentrated. The Center of mass (CoM) is useful for visualizing the motion of a compound multi-link objects such as a humanoid robot. It is also vital for applying Newtonian physics to a complex structure system. The CoM is the particle object equivalent of a given object for an application of the laws of motion [4]. Because of the Complex structure of the humanoids, it is computationally expensive to calculate the position for each of the body links during motion and hence it is easier to use the CoM instead. In regards to the masses distributed rigid body dynamics, it is useful to have the CoM as a reference point for mechanical calculations. The CoM of a multi-link body in 3 dimensions can be calculated using equations 2.1 .

$$x = \sum_{i=1}^{N} \frac{m_i X_i}{M}, y = \sum_{i=1}^{N} \frac{m_i Y_i}{M}, z = \sum_{i=1}^{N} \frac{m_i Z_i}{M} \tag{2.1}$$

Where $x$, $y$ and $z$ are the 3D Cartesian coordinates of the CoM. $X_i$, $Y_i$, $Z_i$ are the Cartesian coordinates of the CoM of the $i_{th}$ link. $m_i$ represents the mass of the ith link $M$ is the total mass of the body and $N$ are the number of the body links.

### 2.1.2 The Ground Projection of the Center of Mass

The Ground Projection of the Center of mass (GCoM) is defined as a point where a vertical line from the CoM intersects with the ground. The GCoM is used to evaluate the static stability of the gait as it will be explained later. The horizontal GCoM for a multi-linked system is given by equations 2.2 and 2.3 .

$$x_{GCoM} = \frac{\sum_{i=1}^{n} m_i x_{ci}}{\sum_{i=1}^{n} m_i} \tag{2.2}$$

$$y_{GCoM} = \frac{\sum_{i=1}^{n} m_i y_{ci}}{\sum_{i=1}^{n} m_i} \tag{2.3}$$

$m_i$ represents the mass of the ith link, $x_{ci}$ and $y_{ci}$ are the position of the Center of mass for the $i_{th}$ link.

### 2.1.3 Zero moment point

The Zero Moment Point (ZMP) is a point on the ground where the total moment generated due to gravity and inertia equals zero [5], [6]. The ZMP is effected by the robot's links position and inertia, which continually changes during motion. The ZMP of a multi-link body can be calculated by equations (2.4) and (2.5) [7], where $p_x$ and $p_y$ are the position of the ZMP in the Cartesian domain:

$$P_x = \frac{\sum_{i=1}^{n} m_i(\ddot{z}_i + g)x_i - \sum_{i=1}^{n} m_i \ddot{x}_i z_i - \sum_{i=1}^{n} I_{iy} \ddot{\Omega}_{iy}}{\sum_{i=1}^{n} (\ddot{z}_i + g) m_i} \tag{2.4}$$

$$P_y = \frac{\sum_{i=1}^{n} m_i(\ddot{z}_i + g)y_i - \sum_{i=1}^{n} m_i \ddot{y}_i z_i - \sum_{i=1}^{n} I_{iy} \ddot{\Omega}_{iy}}{\sum_{i=1}^{n} (\ddot{z}_i + g) m_i} \tag{2.5}$$

In this equation, $x_i, y_i$ and $z_i$ are the CoM vertexes of the $i_{th}$ link . $m_i$ is the mass of the $i_{th}$ link, $g$ is the gravitational constant, $I_{ix}$ and $\Omega_{ix}$ are the moment of inertia and angular displacement about the X-axis, respectively. That is said, the ZMP is a point where the reaction force at the contact with ground does not produce any moment in the horizontal direction, i.e., the inertial and gravity moments result in net zero moments. The term ZMP is prevalent and essential in the robotics world. In the next sections, I explain how the ZMP is related to the robot dynamic stability.

**Extended Zero Moment point** The EZMP was proposed by Sun et al. [8] as an extension to the concept of Zero Moment Point. The ZMP of the robot cannot be defined if the robot's feet are in contact with surfaces on different planes. The EZMP is defined as a point on a virtual contact surface (VCP) in an arbitrary virtual surface with a finite slope [8]. The EZMP is useful in balancing a biped that is walking on complex and rough terrains. Moreover, when the floor surface is smooth flat, the EZMP is the same point as the ZMP. Figure 2.1 shows the concept of VCP.

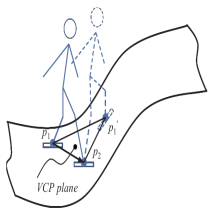


FIGURE 2.1: The virtual contact surface [8]

**Fictitious ZMP** The ZMP only exists inside the support polygon. As I stated before, if the ZMP reaches the polygon edges it becomes marginally stable. However, if the ZMP was pushed outside the support polygon, it becomes a fictitious ZMP (FZMP). The FZMP is used in falling analysis as well as for locomotion on low friction surfaces. Kajtia et al. have used the Fictitious ZMP to sabilize the biped walking on a low friction surfaces [9].

### 2.1.4 Center of Pressure

The Center of pressure (CoP) can be defined as a point on the ground where the ground reaction force acts [7]. The CoP is also the point where the resultant moment generated by inertial and gravity forces is tangential to the ground and the net moment in the horizontal direction is zero [10]. In simpler terms, the CoP is the average pressure distribution over a surface. If the robot is balanced the CoP and the ZMP indicates the same point. During the gait, there are two types of forces that act on the robot; forces exerted by contact and forces transmitted by without contact. The CoP is related to the first type, while the ZMP is linked with the second [11]. The main difference between the CoP and the ZMP is that the CoP never leaves the area covered by the robot feet, while the ZMP can temporary leave that area. The CoP of a multi-link body can be calculated using equation (2.6) [7].

$$OP = \frac{\sum_{i=1}^{N} q_i F_{ni}}{\sum_{i=1}^{N} F_{ni}} \tag{2.6}$$

In this equation OP is the vector from the origin of the coordinate system O to the Center of Pressure position; $q_i$ is the vector to the point where force $F_{ni}$ acts perpendicular to the surface.

### 2.1.5 Support Polygon

Many stability measurement techniques use foot support area in their measurement approaches, which is called the support polygon [7]. The support polygon is the area covered by single foot or the convex hull of two feet area in single and double support phase respectively. Dynamic and static stability can be analyzed based on the distance of a particular point to the boundary of the support area. The size of the support polygon is proportionally related to the stability of the robot. That is a robot with large feet is easier to keep in balance. Moreover, the humanoid robot is more stable in double support phase comparing to a single support as the support polygon area is more extensive. Figure 2.2 shows the support polygon for the NAO in double support phase.

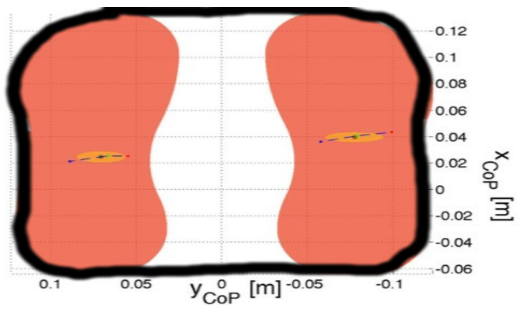


FIGURE 2.2: Support Polygon of the NOA robot in double support phase

### 2.1.6 Static and Dynamic Stability

The biped gait is said to be statically stable and a posture is said to be balanced if the gravity line from its center of mass falls within the convex hull of the the foot support area [10]. In other words , the biped robot is statically stable when the ground projection of the Center of mass (GCoM) is located inside its support polygon. When the robot is not moving, if the GcoM leaves the support polygon area it creates a moment on the supporting foot that causes the robot to fall. It is useful to have a method to evaluate the stability of the robot in a stationary state. The static stability margin assets that by measuring the distance of the GCoM from the edge of the support polygon area. The measured distance is proportionally related to statical stability.

However the situation is different for the dynamic stability, because the GCoM may leave the support polygon area in single support phase to push the robot forward. The robot is said to be dynamically stable if its ZMP remains inside its support polygon. This stability criterion ensures that the robot remains in balance and will not fall during locomotion. The robot stability can be evaluated by measuring the distance of the ZMP to the support polygon boundary. The measured distance is proportionally related to dynamic stability margin. This mechanism can be employed to provide feedback to the robot regarding its balance state. One drawback in using this stability assessment, is that it can not distinguish between the marginally stable and unstable state, because in both cases the ZMP is located at the polygon boundary [7].

### 2.1.7 Summery

Improving the locomotion stability remains one active area of research in the humanoid world. Because of the biped's non-linear structure, balancing the locomotion is a non-trivial task. It is essential to understand the concepts behind the stability assessment to be able to improve the locomotion stability. In this chapter, I have introduced and illustrated various concepts related to locomotion balance, such as the Center of pressure, the Zero Moment Point, the Support polygon and the Center of mass. All these concepts will appear many times in our thesis, and it is crucial that our reader gets familiar with it to be able to follow our work. The ZMP stability criterion is the most popular and was introduced by volcaburic [6]. Using this approach, if the ZMP is kept inside the area of the support polygon the robot remains in balance. Additionally, the ground projection of the CoM is employed in the same manner to ensure the stability of the robot in a stationary state. The distance of the ZMP and GCoM from the support polygon edge can be used to evaluate the dynamic and static stability of the robot respectively.

## 2.2 Biped Locomotion Approaches

### 2.2.1 Overview

Biped locomotion represent the humanoid moving mechanism using its legs. The locomotion is one feature that is exclusive with humanoid robots. There are various robotic locomotion approaches developed with different aspects. In this section, several factors that can be used to classify the walking approaches were discussed.

**Passive walking vs Active walking**

Passive or cyclic walking can be defined as a robot natural walking using only gravitational forces without using any power or joint actuation. Passive walking can be summarized as unaided walking down a slight incline [12]. This kind of walking has the advantage of not requiring energy source or a computer controlling system. However, this walking approach has many disadvantages such as there is no control of the walking speed or direction and the walk can only be performed on a sloped surfaces. All these factors make this type of walking unsuitable for practical use.

Active walking is achieved with joints actuation that is controlled by motors [7]. One of the main differences between passive and active walking is that the swing leg does not fall on landing motion, that way the impact force is reduced considerably, so stable motion is obtained [12]. In contrast with passive walking, the joint trajectory needs to be calculated and the walking process is achieved by following the calculated trajectory. The speed and direction of the walk are controlled and can be changed anytime. This type of walking is more suitable for the everyday application and it is more popular than the passive walking.

**Static locomtion vs Dynamic locomotion**

Static locomotion is achieved based on the CoM position only without considering the CoM acceleration. A static walker is assumed to remain in balance if the GCoM is kept inside the support polygon all the time. The CoM trajectory for this approach is almost the same as the ZMP [7]. Graph et al presented a static closed-loop walking approach for soccer humanoid using only the CoM trajectory [13]. A static gait is much slower than a dynamic gait because it restrict the GCoM from the leaving the support polygon, which makes it less popular than the dynamic locomotion.

In contrast, the Dynamic locomotion considers the CoM position and acceleration. The dynamic walking approach produces a natural human-like gait. Moreover, dynamic locomotion is faster than the static approach because it allows the GCoM to leave the support polygon to push the robot forward. The Dynamic walking is much popular and practical for various tasks.

**Model-Based and Model-Free locomotion**

Model-based locomotion are valid for a specific kind of humanoid robot and require changes to be used for different robot models. The dynamics of a robots are usually simplified using a physical model, because using the actual robot dynamics can be complicated and computationally heavy. A hierarchal control logic is employed to interact multiple modules together and perform the required calculation to produce the gait.

However, in model-free approaches, the gait is achieved by mapping between sensors and actuators without the need to model any dynamics [7]. The mode-free gait can be used for different robots models without the need for many changes. In this study, I am going to concentrate on the model-based locomotion.

### 2.2.2 ZMP Based Approaches

The Zero Moment Point introduced by vukobratovic [6], specifies a point on the ground where the tipping moment acting on the robot, due to gravitational and inertial forces, equals zero [14]. The ZMP criterion is very popular because of its ability to generate a dynamically stable gaits [15]. The stability of the dynamic gait is proportionally related to the distance from the ZMP to the support polygon edges. If the ZMP remains inside the support polygon area, the robot will be physically stable and will not fall. For a given a walking pattern, it is desired to keep the ZMP in the middle of the footsteps to ensure stability. This criterion is not restricted to locomotion and can be used for other robotic actions.

On the other hand, The ZMP can be used independently of walking engine as a feedback control mechanism that indicates if the robot will remain stable or not. When the robot in balance, the ZMP equals the Center of pressure (CoP). The ZMP can be measured with sensors and be employed to indicate instabilities. The ZMP criterion is limited and cannot include complex actions such as running, jumping or stairs climbing, because the ZMP is only defined on a continues support polygon surface. In this master's thesis, the gait will be obtained by modeling the ZMP and the CoM trajectories, as well as the ZMP will be employed a feedback mechanism.

#### 2.2.2.1 Cart-Table Model

The cart-table is a physical model that simplifies the robot's dynamics and provides a mathematical relation between the CoM position and the ZMP. As shown in figure 2.3, the humanoid walking can be represented by cart-table model during the single support phase.

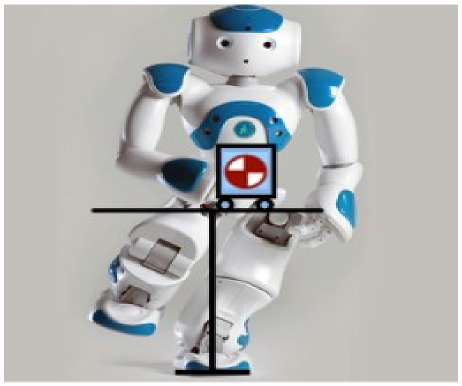


Figure 2.3: The Cart-table Model For The NAO [7]

The model assumes all the masses are concentrated in a cart and the robot support leg is massless. These assumptions are far from reality but could still provide a good approximation because most of the robot mass is concentrated in its trunk and the legs weight effects are small in comparison with the trunk. The principle of the model is that it balances an object by supporting its Center of mass and the ZMP can move in the foot which is modeled by the table contact with the floor. During locomotion the CoM moves along both planes. If the CoM position along a plane exceeded the supported area, then a horizontal moment will be exerted on the axis and the table would topple. Figure 2.4 provides a schematic view of the model.

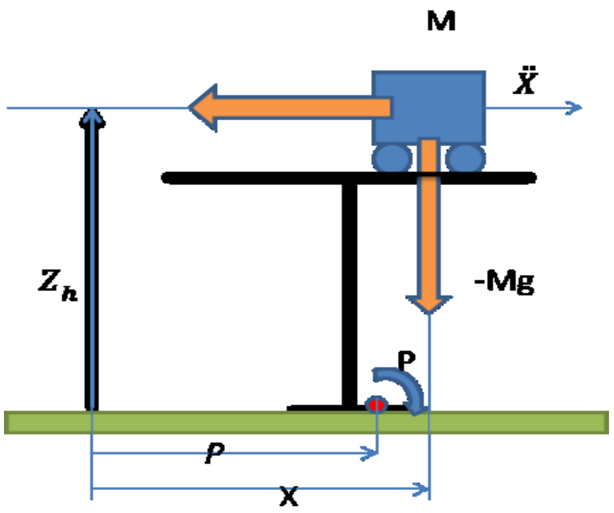


FIGURE 2.4: Schematic view of Cart-Table model [7]

$x$ and $z_h$ represent the CoM position in x-z plane. $\ddot{x}$ and $g$ are the CoM and gravitational acceleration respectively. $T_p$ is the tipping moment around the CoP, and M is the CoM mass. The moment around point P can be calculated using equation 2.7 .

$$T_p = Mg(x - P) - M\ddot{x}Z_h \tag{2.7}$$

As I mentioned before, the dynamic stability criterion requires the moment at the ZMP to equal zero. By substituting $T_p$ =0 in equation 2.7, equation 2.8 and 2.9 describes the ZMP on the $X$ and $Y$ axis repetitively.

$$P_x = x - \frac{Z_h}{g}\ddot{x} \tag{2.8}$$

$$P_y = y - \frac{Z_h}{g}\ddot{y} \tag{2.9}$$

The cart-table describes the CoM movement in one plane only and two augmented cart-table models are needed to describe the robot motion in three dimensions. The Cart-table has a draw back of not considering any angular momentum change at the CoM.

**Linear Inverted Pendulum**

The Linear Inverted Pendulum (LIPM) is another physical model that can be used to generate humanoid motion. It an extension to the Cart-table and has the same equations. As in the Cart-table, the LIPM simplifies the robot as a single mass point on its CoM. It restricts the pendulum motion to the horizontal axis without any vertical movement which results in a linear space state equations. Figure 2.5 provide a schematic view of the model. The LIPM assumes that the ZMP is at the origin O, which corresponds to the robot's ankle, and the ankle's torque is Zero [16].

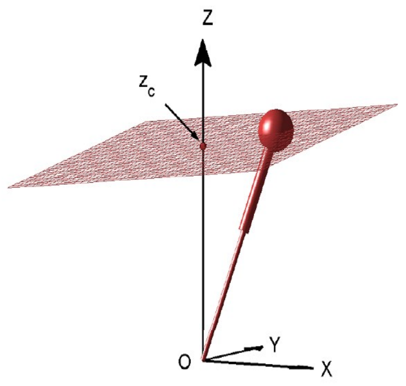


FIGURE 2.5: Linear Inverted Pendulum [16]

In contrast with LIPM, the Cart-Table model allows the ZMP to move in the foot that is modeled by the table contacts with the ground, and the torque of ankle is not required to be zero [7]. The LIPM also has the draw back of not considering any angular momentum associated with the motion .

**Cart-Table Solution approaches**

The physical models discussed above provide a set of differential equations describing the ZMP as a function of the CoM position and acceleration. The Cart-table and LIPM equations are linear. The task of calculating the CoM position from the ZMP by solving the model's differential equations is nontrivial and require some work. Researchers have addressed this problem using different approaches. In this section, I provide a literature review on related work for solving the model's differential equations to provide the CoM position as a function of the ZMP. The approaches to solving this problem can be divided into two groups, Analytical and numerical approaches.

### Analytical approach

The analytical approaches had more attention than the numerical ones. Nishiwaki et al. developed an analytical solution using a method to modify the walking pattern by connecting the actual ZMP trajectory to a new reference trajectory [17]. Takanishi et al. generated the CoM trajectories using Fourier transform of the reference ZMP [18]. Erbatur et al. Solved the cart-table model equations using Fourier representation of the ZMP [19], [20].

Other works addressed the solution by time segmenting the ZMP trajectory. These approaches referred to as the Segmentation based approaches. Herada et al. Solved the differential equations analytically by representing the ZMP using spline function [21]. Park et al. Provided another approach in which he presented the time segmentized ZMP using Fourier series [22]. N.Shaffi et al., also used Fourier series with a different time segmentation approach in which the double support periods can be varied [23].

Table 2.1 presents a comparison between Herada, Park and Shafii's criteria to generate a biped walking. The methods are compared based on computation time for ten steps, trajectories representation and the improvement to the time segmentation approach. The approaches were tested on a machine running on Intel I5 3.0 GHZ with 8 GB of physical memory. Once can notice that Shaffi's approach is the most advanced one, as it has the fastest computation time and the ability to dynamically change the double support time (DSP) and single support time (SSP) give more control on the walking process.

| Approach | ZMP Representation | Improvement | Computation time |
|---|---|---|---|
| Park | Spline Function | Change the walking direction online | 0.000561 s |
| Haerda | Fourier series | Smoother ZMP-CoM trajectories | 0.001654 s |
| Shafii | Fourier series | parameterize different double support periods | 0.000402 s |

Table 2.1: Comparison Between three of the segmentation based approaches

**Numerical approach** For the numerical approach, Kajita et al. Solved the differential equation for the CoM position using the preview control of the reference ZMP [24]. His numerical approach has gained popularity in the robotics field and was adapted by the Aldebaran company for NAO SDK walking module. Kagami et al. Discretized the equations in time segments and provided CoM position from the given ZMP [25].

#### 2.2.2.2 Inverted Pendulum

Both of the discussed models restrict a fixed CoM height during the motion which makes the gait unnatural and energy inefficient. The restriction require the robot to move with a bent knee which exerts a massive amount of torque on the knee joint and prevents the robot from performing advanced motions such as running.

The inverted pendulum model tackles this issue by allowing CoM vertical motion. In this model, a connection between the pivot point and the CoM is assumed to be a massless telescopic rod [7]. Figure 2.6 shows a schematic view of the Inverted pendulum in the sagittal plane with the CoM attached to a telescopic rod.

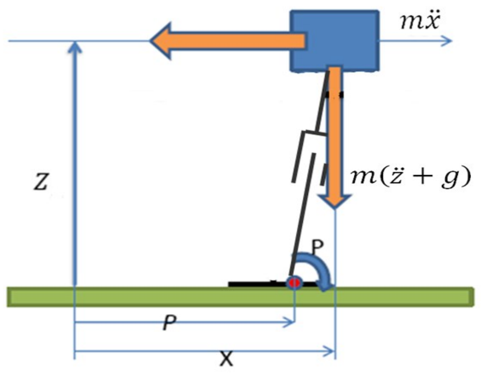


FIGURE 2.6: A schematic view of the inverted pendulum [7]

$x$ and $z$ represent the horizontal and vertical vertices of the CoM. the Gravity $g$; $\ddot{x}$ and $\ddot{z}$ denote the CoM horizontal and vertical accelerations respectively. As in the LIPM, the inverted pendulum describes a movement in one plane, and hence two models are required for a three dimensions motion. Equation 2.10 describes the tipping moment around the point p.

$$T_p = M(g + \ddot{z})(x - p_x) - M\ddot{x}z \tag{2.10}$$

Similar to the LIPM, the robot is stable when the CoP equals the ZMP. This condition requires a zero tipping moment around P. With the left-hand side of the equation equals zero, equation 2.11 and 2.12 present the ZMP as a function of the CoM position and acceleration for the sagittal and frontal plane respectively.

$$p_x = x - \frac{z}{g + \ddot{z}}\ddot{x} \tag{2.11}$$

$$P_y = y - \frac{z}{g + \ddot{z}} \tag{2.12}$$

The vertical CoM position and acceleration are augmented to the model as a sinusoidal function with a particular amplitude and phase.Figure 2.7 shows the NAO robot with the Inverted Pendulum model.

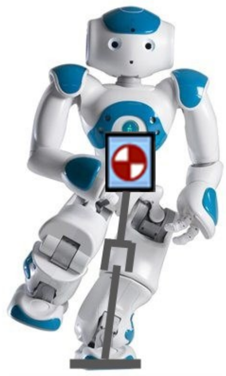


FIGURE 2.7: NAO Robot with the Inverted Pendulum Model [7]

**Solving the IPM differential equations** Kagami et al. Presented a numerical approach to solve the IPM differential equations [26]. Kajita et al. combined this numerical approach with the IPM to generate the CoM trajectory [27]. Kagami's approach discretized the position and acceleration of the CoM using a time step iteration delta t. The IPM differential equation 2.11 can be written as a tridiagonal system as follows:

$$P_x = a_i x(i-1) + b_i x(i) + c_i x(i-1) \tag{2.13}$$

The following linear system can be obtained from equation 2.13 and can be employed to generate a horizontal CoM trajectory.

$$\overbrace{\begin{bmatrix}P_{(1)}\\P_{(2)}\\ \\ \\ \vdots\\P_{(n)}\end{bmatrix}}^{ZMP_x} = \overbrace{\begin{bmatrix} b_1 & c_1 & & & \cdots & 0\\ a_2 & b_2 & c_2 & & & \\ & a_3 & b_3 & c_3 & & \vdots \\ & & \ddots & \ddots & \ddots & \\ \vdots & & & a_{n-1} & b_{n-1} & c_{n-1}\\ 0 & & & \cdots & a_n & b_n\end{bmatrix}}^{K} \overbrace{\begin{bmatrix}x_{(1)}\\x_{(2)}\\x_{(3)}\\ \\ \vdots\\x_{(n)}\end{bmatrix}}^{CoM_x}$$

Tomas algorithm [28] can be employed to quickly solve the tridiagonal matrix algorithm (TDMA).

This approach suffers from drawbacks such as the statical balance is not ensured at the end of each walking step. Moreover, the initial velocity of the CoM is not considered, and the CoM position is not defined at the begging and end of the step.

Shaffi et al. Improved Kagami's approach to overcoming the mentioned drawbacks and to generate a more dynamic gait in which the velocity connectivity of the CoM trajectory is assured [29]. They defined a boundary condition as follows:

$$P_1 = a_1 x(0) + b_1 x(1) + c_1 x(2) \tag{2.14}$$

$$P_n = a_n x(n-1) + b_1 x(n) + c_1 x(n+1) \tag{2.15}$$

Since the CoM position of the current step is the same as the last CoM position of the previous step, equation 2.16 can be retrieved from equation 2.14 by assuming X(0)= $CoM_{initial}$.

$$P_1 = a_1 CoM_{initial} + b_1 x(1) + c_1 x(2) \tag{2.16}$$

Additionally, the assumption that the CoM velocity is zero at the last position. By assuming that x(n) = x(n+1) the following equations is given:

$$P_n = a_n x(n-1) + (b_n + c_n) x(n) \tag{2.17}$$

The advantage of using this approach is that it provides a quick and computationally light solution to the IPM while having the CoM velocity connected between the single and double support phases. On the other hand, this approach has the beginning and the end of the walk is not statically balanced. I managed to handle this drawback using our stability approach that will be discussed in chapter 4.

#### 2.2.2.3 Inverted Pendulum Plus a Flywheel

The Linear Inverted Pendulum Model (LIPM), Inverted Pendulum Model (IPM) and Cart-Table models, introduced in chapter 2, were employed to simplify the robot's dynamics to describe the movement of the CoM. However, these models assume that the motion of the CoM is restricted to the horizontal plane or to a horizontal-vertical plane in the case of the IPM. With such restrictions, the ground reaction force can not generate angular momentum and always passes through the CoM [30]. However, a disturbance source or the motion of the upper body can exert significant amount of torque to the CoM. If a balancing strategy did not manage to handle the received torque, the robot might lose balance and fall as a result. Since the discussed models do not consider any rotational inertia, it is necessary to modify these models to account for the angular momentum change at the CoM.

The LIPM was first extended by [31] and [32] as the Angular Momentum Pendulum Model (AMPM). Part et al. presented a linear inverted pendulum plus a flywheel as an extension of the LIPM that considers the angular momentum associated with the concentrated mass [33]. In their work, they replaced the angular momentum generator for the AMPM by a flywheel. The flywheel is a mechanical device designed to store rotational energy efficiently [34] and can be used to turn objects in space such as satellites. Pratt's model, shown in figure 2.8, uses a flywheel with a centroidal moment of inertia and rotational angle limits instead of a traditional flywheel. That is because their model is intended for humanoids with limited rotational angle and velocities. The NAO is an example of a robot with joints and actuators that have a limited rotational angle and velocities.

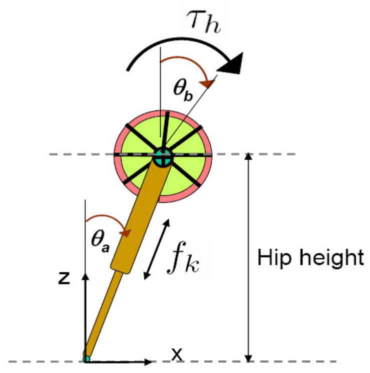


FIGURE 2.8: A model of a biped in the single support phase with a flywheel body and massless legs [33]

The model is shown in figure 2.8, is merely an inverted pendulum with a flywheel attached to the CoM. Similar to the IPM, the leg of the robot is assumed to be massless and expendable. The motion equations of the model in single support phase are :

$$m\ddot{x} = f_k sin(\theta_a) - \frac{\tau_h}{l} cos(\theta_a) \tag{2.18}$$

$$m\ddot{z} = -mg + f_k cos(\theta_a) + \frac{\tau_h}{l} sin(\theta_a) \tag{2.19}$$

$$J\ddot{\theta}_b = \tau_h \tag{2.20}$$

$g$ is the gravity acceleration, $m$ and $J$ are the mass and rotational inertia of the flywheel; $x$ and $z$ are the horizontal and vertical coordinates of the CoM. $l$ is the CoM hight,$\theta_a$, and $\theta_b$ are the leg and flywheel angles to the vertical axis. $\tau_h$ is the motor torque at the flywheel and $f_k$ is the actuation force applied to the leg.

The linear inverted pendulum plus a flywheel is derived from the above equations, by setting the vertical height to be constant $z = z_0$ and hence the vertical acceleration to zero. The equations of motion for linear inverted Pendulum plus a flywheel become :

$$\ddot{x} = \frac{g}{z_o}x - \frac{1}{mz_o}\tau_h \tag{2.21}$$

$$\ddot{\theta}_b = \frac{1}{J}\tau_h \tag{2.22}$$

The LIPM plus flywheel, discussed in the previous section, has a constant vertical height constraint that kept the model's equations of motion linear. However, a consistent CoM height model produces a gait with a bent knee that is unnatural, energy inefficient and exerts a significant amount of torque on the knee joints. The fixed CoM height restriction must be released from the model to overcome the mentioned disadvantages and produce a gait that allows a 3D motion for the upper body. As I mentioned in chapter 4, The IPM was extended from LIPM by augmenting a CoM vertical trajectory to the LIPM. The vertical trajectory of the CoM is a sinusoidal function with a particular amplitude and phase. Kasaei et al. [30] have enhanced the LIPM with a flywheel and released the fixed CoM height constraint to produce a natural walk while accounting for the angular momentum change at the CoM. In their recent work, the equations of the LIPM with the flywheel (2.21 and 2.22) was improved to include the CoM vertical trajectory function as follows:

$$\ddot{x}(t) = \frac{g + \ddot{z}(t)}{z(t)}x(t) - \frac{1}{m \times z(t)}\tau(t) \tag{2.23}$$

$$z(t) = z_c + A_z sin(t/T) \tag{2.24}$$

$$\tau(t) = \tau_a(t) - \tau_b(t) \tag{2.25}$$

$$\tau_b(t) = J\ddot{\alpha}(t) \tag{2.26}$$

Where $z(t)$ is the CoM vertical trajectory, $z_c$ is the height of CoM during double support phase and $A_z$ is the adjustment amplitude. $\tau_b$ the torque from a disturbance source, $\tau_a$ is the recovery torque from a recovery strategy.

### 2.2.3 Summery

In this section, I have reviewed and compared different humanoid locomotion classifiers that are important to understand the characteristics of the gait. In this study, I chose to use a model-based, active and dynamic walking approach because it results in a fast and natural omnidirectional locomotion. I have reviewed the ZMP Stability criterion and provided a detailed explanation of how the benchmark can ensure the physical balance of the robot as well as to evaluate the robot's stability level at any time. I also reviewed the most popular physical models that can be used to simplify the dynamics of the robot and provide computationally feasible solutions. I discussed the related works on solving the model's differential equations numerically and analytically. Since the angular momentum plays a role in the stability of the humanoid robots, I have discussed an enhanced physical models that allows to account for the angular momentum change during locomotion. These models will be employed in the next chapter 4, to measure the torque at the CoM and detect instabilities.

## 2.3 Gait Analysis

In this section, I present and define several leading terms that are vital to understanding the development of a walking engine.

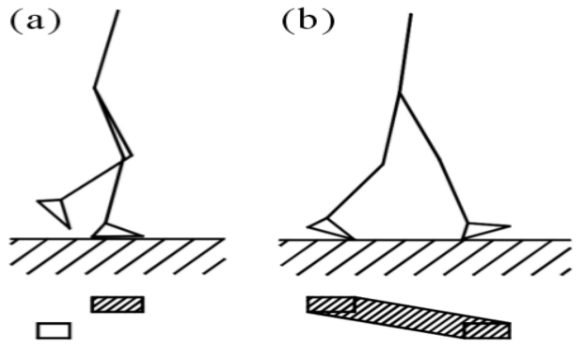


FIGURE 2.9: Support Polygon during single-support phase (a) and double support phase (b) [35]

### 2.3.1 Double and Single Support Phase

**Double Support Phase (DSP)** describe an instant where both of the robot feet are in contact with a surface on the floor. When a biped is in double support phase, it is supported by both of its feet, i.e., the two feet are on the floor surface or on an object that is placed on the floor.

**Single Support Phase (SSP)** define a situation when one the robot feet are in constant with a surface on the floor. During SSP the biped uses only one foot to support its weight. DSP put the robot in a more stable state as the support polygon area is enlarged when the two feet are in contact with the floor. Figure 4.1 shows both the DSP and SSP and its relation to the support polygon area.

### 2.3.2 Swing and Stance Leg

The **Swing leg** is the leg that is performing a step by moving or swinging forward through the air. In contrast, the **Stance leg** denotes the leg that is fully supported by the floor and supports all the robot's weights.

### 2.3.3 Single Direction and Omnidirectional Walking Engine

#### Single direction walking engine

A single direction walking engine allows the robot to walk in straight lines. This approach is not very practical for humanoid robots in dynamic environments, because the robots need to change its walking direction continually as the surrounding environment changes. In terms of RoboCup, the playing robots need to follow the ball and maneuver around others robots which makes having an omnidirectional walking engine vital. Moreover, Humans continually changes their walking speed and direction in everyday tasks and since it is desired to have humanoids robots working at the humans level skills, improving the omnidirectional walking engine remains an active research area.

As it was mentioned above, Erbatur et al. have used Fourier series to represent the ZMP and solved the cart-table equations [36]. His approach resulted in a single direction gait without the ability to perform a curved or diagonal locomotion; because they failed to connect the CoM trajectories smoothly while changing walking direction. The approach was improved later to generate curved walking [37]. Ferreira et al. [38] improved Erbatur method to generate diagonal walking, in which they decomposed the walking trajectory into time segments and added the double support phase to formulate the ZMP trajectory.

#### Omnidirectional walking engine

An omnidirectional walking engine has the ability to change the robot walking speed and direction. Generating omnidirectional walking comprising straight, curved, sideways and diagonal walking. The ability to change the direction of the walking motion improves the robot's maneuverability in dynamic environments [39]. All these advantages make the omnidirectional approach a better choice for humanoids working in complex environments. However, changing the walking direction requires connecting the current CoM trajectory with the new direction trajectory. Changing the direction endanger the robot balance because the ZMP could reach the marginally stable regions of the support polygon, thus more stability controllers are be needed for the omnidirectional approach. In the next chapter, I will discuss the process of developing a modular omnidirectional walking engine. The main modules of the engine will be presented in a hierarchal form to reduce the complexity.

## 2.4 Humanoid Balancing Approaches

### 2.4.1 Introduction

The locomotion ability and high mobility are the main advantages that make bipeds more useful than wheeled robots for certain tasks. There are also some disadvantages that accompany the bipeds such as its complex non-linear design which makes it prone to losing balance. The resulted locomotion from the walking engine is based on simplified physical models which may induce errors in the walk due to the simplifications. Additionally, there is existing noise in the robot's legs actuators that might reduce the walking accuracy. Modern bipeds have various amounts of functionalities and skills which can be used in different areas. However, without having a great ability to remain in balance, we can not take full advantage of these robots. It is desired to keep the robots from falling over because humanoid robots are expensive and fragile. All the reasons mentioned above shows that improving the balancing capabilities of the biped is vital for the sake of practicality.

Developing balancing controllers for the bipeds locomotion are an open problem that attracted many researchers. During motion, there are expected and unexpected disturbance sources that exist in the environment. Balancing the robot during locomotion can be implemented in different forms such as an active balancing controller or as event triggered controllers. The active balancing controllers are always online and working to keep the robot in balance. But being active all the time keep robot resources busy and reduces the computational power of the robot. In contrast, event-triggered controllers remain off and are only activated for some time duration when needed. It saves the computational power of the robot and keeps the robots sources available to be used by other applications. In this section, I provide a review of related work on balancing the robots while in motion.

### 2.4.2 Active Balance

Active balance is required to keep the robot stable against expected perturbations such as a small changes in the terrain. The Generated locomotion will have errors due to the simplifications in modeling and the inherent noise in the legs actuators. Thus some feedback is required to perform slight adjustments to the gait. Humans nervous system actively adjust the torso pitch angle by measuring the tilting angle using the vestibular system [40]. The same phenomenon can be carried with robots by using onboard sensors to estimate the body state. Shaffi et al. [7] introduced the concept of active balancing to keep the robot's trunk upright and reduce the risk of falling. Their balancing module adjusts the trunk inclination to the upright by decreasing the variation of the trunk angle. The trunk inclination angle is measured using the inertial measurement unit (IMU) implemented inside the robot body. The module calculates the feet position and orientation relative to coordinates frame attached to the robot CoM in order to keep the Z-axis perpendicular to the floor. Figure 2.10 illustrates the idea of their active balance approach.

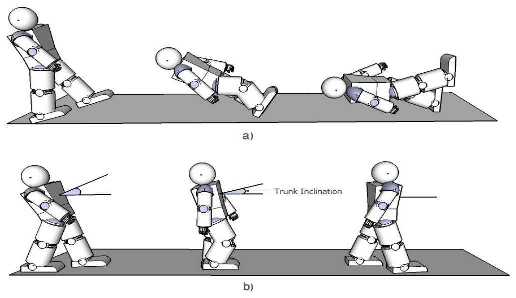


FIGURE 2.10: Illustration of the active balance mechanism by showing two different scenarios of walking with and without using the active balance in section A and B respectively [41]

Liu et al. Have developed a control method to compensate the robot's upper body inclination by employing the hip and ankle joints in real time. Their controller uses the torso's orientation and angular velocity error to modify the reference trajectory of the hip and ankle joints to keep the robot's upper body upright [42]. Equations 2.27-2.30 describe how the hip and ankle joints can be modified online to correct the upper body inclination.

$$\delta\theta = K_{p1}(\theta_{des} - \theta_m) + K_{d1}(\dot{\theta_{des}} - \dot{\theta_m}) \tag{2.27}$$

$$\phi = K_{p2}(\phi_{des} - \phi_m) + K_{d2}(\dot{\phi_{des}} - \dot{\phi_m}) \tag{2.28}$$

$$\delta\vee_{mod} = \vee_{ref} + \delta\theta \tag{2.29}$$

$$\rho_{mod} = \rho_{ref} + \delta\phi \tag{2.30}$$

Where $K_p$ and $K_d$ are the proportional and derivative gains of the PD controller; $\theta_{des}$ and $\phi_{des}$ are the desired torso inclination about the Y-axis and X-axis, $\ddot{\theta}$ ,$\ddot{\phi}$ represent the desired torso angular velocity, which are set to zero; ($\theta_m$ , $\phi_m$) and ( $\dot{\theta_m}$, $\dot{\phi_m}$ ) represent the measured inclination and angular velocity of the torso, respectively; $\theta_{ref}$ , $\phi_{ref}$ denote the reference trajectories of the hip/ankle joints about the Y and X axis respectively; While ( $\theta_{mod}$, $\phi_{mod}$ ) are the modified angles sent to the robot.

Figure 2.11 illustrate their posture control system in a Hierarchal structure. it can be noticed that the controller is augmented to the walking engine.

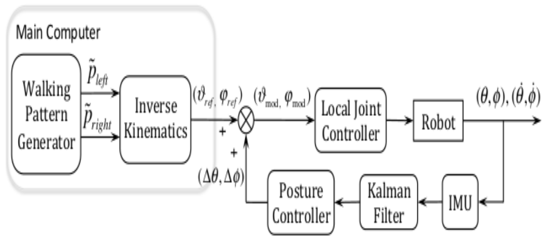

Main Computer
Walking Pattern Generator
$\tilde{p}_{left}$
$\tilde{p}_{right}$
Inverse Kinematics
$(\vartheta_{ref}, \varphi_{ref})$
$(\vartheta_{mod}, \varphi_{mod})$
+
+
$(\Delta\theta, \Delta\phi)$
Local Joint Controller
Robot
$(\theta,\phi), (\dot{\theta},\dot{\phi})$
Posture Controller
Kalman Filter
IMU

FIGURE 2.11: Posture Control system [42]

### 2.4.3 Balance Recovery

When the robot is moving in a complex terrain or dynamic environment, there are different disturbance sources and non-linear forces that exist and should be accounted for. Recovery balance is the process of having the robot performing certain actions to avoid falling. This kind of balancing control is used against unexpected disturbance sources that can not be avoided. Recovery balancing controllers are developed in separate from the walking engine and integrated to operate with it. The controllers are triggered using sensory feedback that detects instabilities. Then a certain sequence of actions is performed to enhance the robot's balance and prevent falling. In contrast with the active balance, the event triggering fashion keeps the recovery controllers from occupying most of the robot limited computational sources. In this section, I review three of the most popular recovery strategies used among humanoid robots, which are the Hip, Ankle and stepping strategies. These strategies are inspired by humans, as studies on humans recovery actions have highlighted these three strategies as the most effective [43].

#### 2.4.3.1 Ankle Strategy

Disturbance sources can exert non-linear forces on the CoM. These forces can move the ground projection of the CoM (GCoM) outside the supported area which will make the robot statically instable. The ankle strategy attempts to keep the GCoM from leaving the support polygon. Ankle strategy can be explained as adding torque to the ankle joints in order to create angular momentum that recover balance [44]. By increasing the robot's ankles stiffness, the changes in the CoP position can be managed to keep it firmly held inside the support polygon area. Ankle strategy is suitable for smaller disturbances because the ankle joint's maximum torque is relatively small. The amount of recovery achieved by using this strategy can vary between different humanoids models. The Force resistive sensors attached to the humanoid feet can estimate the recovery torque needed by the ankles in order to managed the CoP changes. Figure 2.12 describes the working mechanism of the Ankle Strategy.

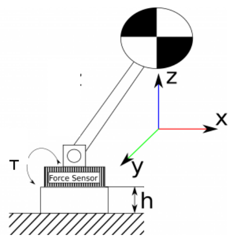


FIGURE 2.12: Ankle Strategy

Hermami and Katbab [45] presented ankle and hip strategies with an inverted pendulum model. Hofmann et al. [46] combined the Ankle strategy with other recovery strategies to produce high balancing capabilities for the humanoid robot. SP. Prahlad et al. [42] have improved the robot stability during locomotion by varying the torque at the ankle joint.

### 2.4.3.2 Hip Strategy

The momentum (hip) strategy involves the use of upper body momentum to compensate for intermediate external force [45]. Similar to the ankle strategy, the hip strategy desire to regulate the changes in the CoP position and move it towards the center of the support polygon. However, the hip strategy depends on the waist and hip joints instead of the ankle. The waist and hip joints usually have higher torque than the ankle joints, as it is responsible for moving the whole upper body. The hip strategy is more suitable for severe disturbances that the ankle strategy fails to handle.

As I mentioned before, when the gait is actuated and the robot is in balance, the CoP and the GCoM represents the same point. In dynamic gaits, the GCoM position can advance the CoP, which makes the robot unstable. The strategy uses the hip and waist joints to change the CoM position so that GCoM and CoP become very close to each other or identical.

Lee et al. proposed an optimized hip strategy for the external force disturbance on elaborate nonstationary floor [46]. Asmar et al. have used the hip and ankle strategies with virtual mode control, such that the hip strategy is deployed when the disturbance is more extensive than what the ankle strategy can handle [42].

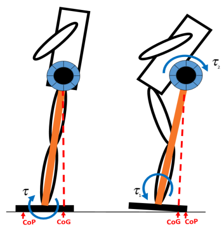


FIGURE 2.13: Hip Strategy [30]

#### 2.4.3.3 Stepping Strategy

The stepping strategy can be defined as taking a step forward step to recover from an externally applied force. The stepping strategy is used when the disturbance is more substantial than what the ankle and hip strategy can manipulate. A forceful push can accelerate the CoM away from the support polygon.and induce instability. Humans tend to take a step forward, when subjected to massive push, to avoid falling.

When the applied force is too large, the robot might fail to regain stability using the hip or ankle strategies. In order to reacquire stability, the robot must take a step forward so that the GCoM final position will be located inside the support polygon area. If the GCoM final position is outside the kinematic workspace of the swing foot, the robot will need to take two or more steps to balance itself. There are different procedures for the stepping strategy such as remaining in double support phase or taking multiple steps. [47].

Goddard et al.[48] Discussed the stepping action to prevent falling. Pratt et al. presented the concept of a capture point approach for the stepping strategy- figure 2.14 . He described when and where to take the step to achieve balance recovery. He also explained that if the capture point was inside the support polygon, the robot could regain stability without taking a step. Otherwise, the capture point can be determined and the robot can take a step to it to reach a stable state [33].

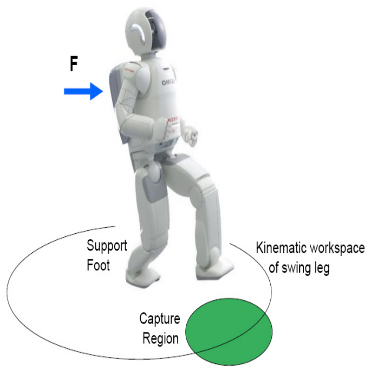


FIGURE 2.14: Capture Point for the stepping strategy [33]

Yasin et al. [49] presented a step out control strategy for biped robot in the presence of externally applied forces, using the capture point. Yi et al. 10,11 developed a balance recovery strategy using the capture point and machine learning [50],[51].

### 2.4.4 Summery

In this chapter, the ankle, hip and stepping strategies were discussed. All of the mentioned strategies are used for balance recovery in the presence of forces that causes the robot to fall. These strategies consist of procedures to actuate the CoP changes and regulate the externally applied forces to satisfy the stability criterion. The ankle strategy is suitable for small forces or disturbance; The stepping strategy is optimized for larger forces that can not be managed by the ankle and hip strategies; The Hip strategy comes in the middle and is useful for medium level perturbations. The Ankle strategy can be summarized to balancing the CoP position, while the Hip strategy is directed toward regulating the angular momentum at the CoM. The stepping strategies can be encapsulated as moving the support area position to follow the CoP. The capture point concept for the stepping strategy is widely used to retrieve the position and timing of the stepping. It is also useful in determining whether the perturbation is inevitable or not.

The three strategies can be combined in one controller. When a distribution source effects the motion, the controller can try employing each of the strategies in series from the least capable (ankle) to the most capable (Stepping) while advancing to the next one if the current is unable to stabilize the robot.

Hofmann et al. Used the three strategies to balance the robot while walking. He aimed to control the horizontal movement of the CoM using the strategies [52],[53]. In their work, the Ankle strategy were employed for small disturbances to reposition the CoP which effects the CoM motion. The Hip strategy was used to produce a counteracting moment about the CoM to change the CoP position. In the presence of severe perturbations, they have employed the stepping strategy to move the support polygon toward the CoP. Stephens et al. Have compared the three balancing strategies. They presented an analytic decision surface to determine if the fall can be avoided using any of the strategies [47].

## 2.5 Humanoid Simulators

Using a simulator is vital for humanoid robotics research because the actual robots are expensive and fragile to some extent. Simulators have many benefits, such as saving the actual robots from being damaged, allowing the user to develop when the real robot is not available, and it does not require reviving the batteries nor dealing with hardware problems.

Different simulators support the NAO robot and choosing to work with one simulator can be frustrating to new users. Various factors should be considered when selecting a simulator over others, such as the hardware requirements, graphics and the available programming languages that can be used. Additionally, some of the advanced simulators not free and their license can be expensive. In this section I provide a summary on some of the famous simulators for the NAO, comparing their advantages and disadvantages.

### 2.5.1 V-rep

V-rep is a general robot simulator provided by Coppelia Robotics. It is based on a distributed control architecture with an integrated, developed environment. Different robot models can be controlled using an embedded script, a Plug-in, ROS node or a remote API client. V-rep allows writing functions and controllers in various programming languages such as C/C++, Python, Java, Lua, Matlab or Octave. The simulator is optimized for fast algorithm development, automation simulation, quick prototyping, and verification. V-rep is also a cross-platform that can be used on Windows, Linux and Mac Os. There are four different physics engines available in V-rep that includes Bullet Physics, Open Dynamics, Newton and Vortex Dynamics allowing fast and customizable dynamics calculation to simulate real-work physics and objects interactions. The NAO H25 model is available in V-rep, and there are bridging programs available to integrate V-rep with NAOqi. There are several advantages for V-rep in comparison with other simulators such as a free license, easy installation and not requiring a specific graphics card or GPU.

### 2.5.2 Webots

Webots is the official NAO simulator provided by the Swiss company Cyberbotics. It has a development environment to model, program and simulates various types of mobile robots. Webots is more suitable for designing complicated robotics setups than the other mentioned simulators. The robots can be programmed and controlled using a built-in IDE or using a third-party development environment. The simulation programs can be easily transferred from the simulator into the real robots to reduce the development time. All these factors made this simulator desirable for research and development projects, such that it is being used by over 1214 universities and research centers around the world. The NAO robot can be simulated in Webots using the Webots C programming language APIs or by using the Aldebaran simulator SDK package. Using the Webots C programming language has a major disadvantage, that different body motions are produced using a predefined joints trajectories. Moreover, the Cyberbotics company holds a yearly NAO challenge to encourage research to develop and program the robot using their simulator.

### 2.5.3 Gazebo

Gazebo is a free simulator that has a robust physics engine with high-quality graphics and a convenient programmatic and graphical interfaces. It is an open source and available under Apache 2.0 licenses. Unlike the above-mentioned simulators, Gazebo is not a multi-platform and only runs on Linux. Gazebo features Dynamic simulation with multiple physics engines that include open Dynamics, Bullet, Simbody and Dart, physics engines. Unlike the simulators mentioned above, Gazebo can be operated using the command line tools that facilitate simulations introspection and control. The most important advantage of Gazebo is that it is the default simulator used in Robot Operation System ( ROS) and have a broad base of community-developed plugins and codes that can benefit a developer. Even though simulators like V-rep have integrated some ROS features, Gazebo is still the most advantaged om terms on ROS integration.

### 2.5.4 Simspark

SimSpark is a physical multi-agent simulator that simulates agents in 3 three-dimensional environments. It is the official RoboCup 3D simulation server. It differs from the other simulators as it allows users to create new simulations using a scene description language as well as being an optimized for multi-agent research. It aims to provide a generic and flexible simulator for different types of simulations. Figure 2.9 shows the NAO robot in SimSpark simulator.

FIGURE 2.15: The NAO robot in SimSpark simulator

### 2.5.5 Summery

In this section, I provided a summary of available simulators that support the humanoid robot, NAO. I presented a comparison while illustrating the advantages and disadvantages of each simulator. Additionally, I discussed some of the objectives on using simulation for testing unpredictable, risky behaviors that might damage the robots.

# Chapter 3

# Development of walking Engine

## 3.1 Walking Engine

A walking engine is a set of modules combined together to produce a set of joints trajectories, that generate a desired walking pattern for a biped robot. The desired walking pattern is specified by the input velocity vector as well as the gait configurations.

## 3.2 Aldebaran Gait

Aldebaran robotics provides a walking API that allows the robot to perform a multi-directional walk. However, the resultant walking is a black box that does not provide the user with full control on the walking process. The API allows the user to adjust some parameters such as the step height and torso orientation. Other parameters can be specified as maximum desired, such as the step in X and Y direction, frequency. However, those parameters can change during the gait. additionally, there are many parameters that can not be set by the user, such as feet separation and step period. During the walk, the user can not tell when the robot is double and single support phase because it keeps changing according to unknown feedback mechanisms. The Aldebaran walk is based on the card-table model, which produces unnatural gait in which the robot walks with a bent knee. The maximum achieved gait speed is 0.119 m/s, which is relatively slow in comparison with other reported speed achieved on the NAO. All these reseasons makes the Aldebaran SDK gait not practical for dynamic environments such as the RoboCup competition.

## 3.3 Engine Modules

In this section, I cover the basic modules needed to obtain a ZMP-based omnidirectional walking engine for the biped robot. These modules are required to achieve a basic locomotion, however other modules can be added to enhance the motion quality and balance. Each module takes several inputs from either another module or from a higher level control engine. The presented modules are not unique and can be obtained using different ways while maintaining the structure of the module inputs and outputs. For example, the ZMP or CoM generators are required to solve ZMP-CoM trajectories, however, there are different approaches to the solution and it is up to the user to choose which method to use. In MS thesis, I simplified the robot dynamics using the IPM because of the advantages discussed before, that the model has over the simple cart-table model. Figure 3.8 illustrates the walking engine working mechanism in a hierarchal modular form.

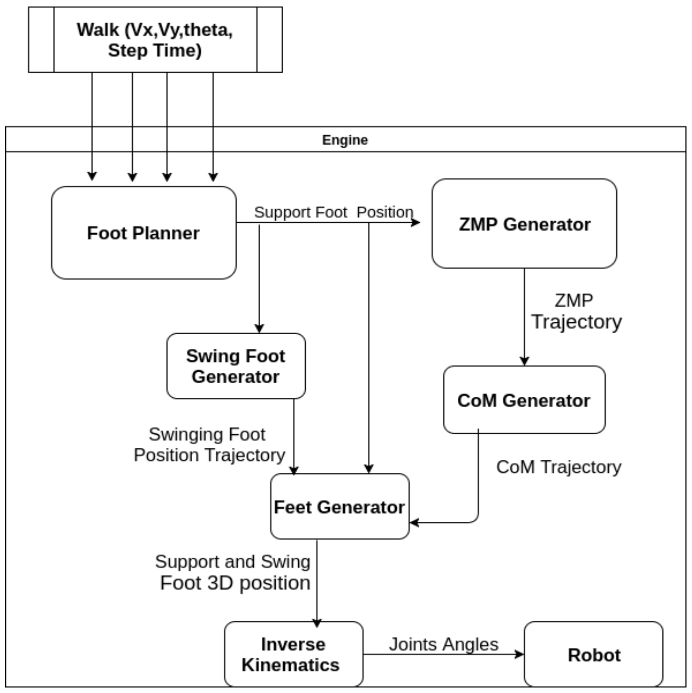


FIGURE 3.1: Modular ZMP based Walking Engine

### 3.3.1 Foot Planner

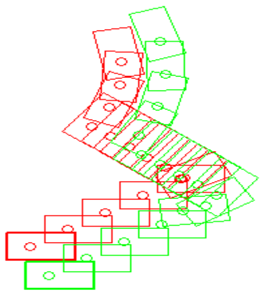


FIGURE 3.2: Planning Next Support foot positions [54]

The foot planner is the start of the walking engine. The future support foot positions are planned based on the engine inputs, such as the velocity vector, the step length as well as the current state of the feet. These planned positions are required to obtain the ZMP trajectory from the ZMP generator module.

Several research papers have presented the development of a omni-directional walking engines [55],[56]. However, They only focused on explaining the ZMP and CoM generator modules functionalities and failed to explain the other modules working mechanism. N.shafi et. al were among a few to include a description of the foot planner structure and functionality [39].

The foot planner plans the next steps as the following: the robot starts the walking process in double support phase in which the CoM is located in the middle of the line connecting the feet centers [39]. After that, the CoM will be shifted towards a new position at the next planned support foot. A reference point can be calculated using the input step length or velocity vector in the X-Y plane. In case of the velocity vector, the reference point is retrieved by multiplying the velocities along the axises by the step time duration. The next support foot is placed away from the reference point by a distance that equals to half of the desired leg separation.

There are three constraints that needed to account for when planning the future support steps which are **Foot reachability**,**Feet Collision** and **Singularities**.

#### 3.3.1.1 Foot reachability

Foot reachability indicates whether the robot is able to move its foot to the desired position. The constrained can be accounted for by checking the minimum and maximum foot separation and step length against the ones calculated by the foot planner, as shown in figure 3.2 . The minimum and maximum step length and foot separation are model-specific factors which are different for different bipeds models.

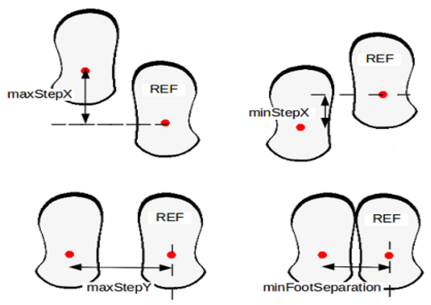


Figure 3.3: Foot steps Clipping against minimum and maximum outreach values [54]

#### 3.3.1.2 Feet Collision

Feet collision constraint indicate whether the robot feet will collide with each other during the motion. This restriction can be accounted for by checking the new step position against a minimum and maximum predefined separation values. Alternatively, the restriction can be accounted for by using a clipping algorithm that defines a box for each foot and chooses a suitable orientation, for the new foot position, so that it will not interfere with the previous foot box.

#### 3.3.1.3 Singularities

The output of a walking engine is a set of joint trajectories that produces the desired gait. These trajectories are calculated using the inverse kinematic module, that is discussed next. The inverse kinematic does not guarantee to find a solution for any reachable foot position, as the space state equations solution may results in a singularity for a certain step. To avoid having singularities at the inverse kinematics, another clipping algorithm can be employed to reduce the singularities.

Aldebaran SDK provides a singularities clipping function that takes into account the minimum and maximum separation and steps length to define safe zone that the foot can move in without resulting in a singularity solution. Figure 3.3 illustrates the ellipse clipping function, where the blue color represents the allowed Zone.

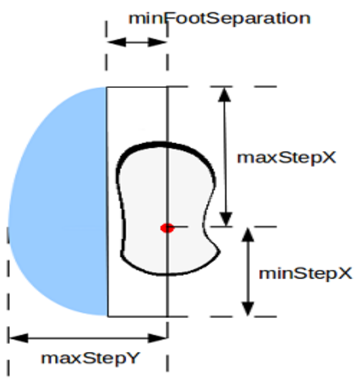


FIGURE 3.4: Ellipse clipping the planned support foot to avoid singularities [54]

### 3.3.2 ZMP Generator

Recall that the ZMP stability criterion ensures the robot stability if the ZMP is kept inside the support polygon represented by the contact area of the robot's feet with the ground. To comply with this criterion, the ZMP should be positioned in the middle of the planned support foot. That is after planning the next steps positions, the reference ZMP trajectory can be retrieved as a set of points located inside the planned footsteps. Figure 3.4 represent the ZMP generator module, which takes the planned footsteps from the foot planner, as input and produces a ZMP trajectory as an output. This module requires the step planner module to operate before it can be called.

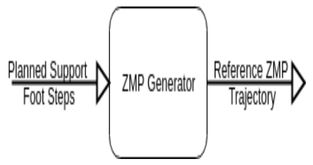


FIGURE 3.5: ZMP Generator Module

### 3.3.3 Center of Mass generator

After generating the ZMP reference trajectory the next step is to calculate the CoM trajectory that produces the reference ZMP. This task is done by the CoM trajectory Generator module, shown in figure 3.5.

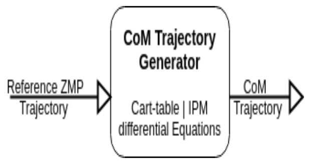


Figure 3.6: ZMP Generator Module

This module produces a CoM trajectory that if it was followed, the actual ZMP will be close to the reference ZMP. The CoM position can be calculated from the reference ZMP by solving the differential equations of a physical model that simplify's the robot dynamics, such as the cart-table or LIPM. As it was mentioned in the previous chapter, there are different numerical and analytical approaches employed to solve the cart-table and inverted pendulum models. The choice of which model to use depends on different factors such as the desired walking characteristics, the robot's computational capabilities, and energy consumption.

In this master's thesis, I have chosen to use the IPM model to simplify the robot's dynamics because the IPM can handle the CoM movement in the vertical axis and produces a natural human-like locomotion. The IPM's differential equation for the frontal and sagittal plane is given in equation 2.11 and 2.12 respectively. I have adopted the approach presented in [29] to calculate the horizontal CoM trajectory using the IPM.

### 3.3.4 Swing Foot Generator

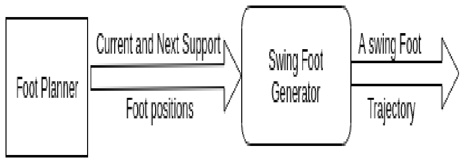


Figure 3.7: Swing Foot Generator

The Swing foot connects the current and next support foot position together. The Swing foot generator module generates a trajectory that takes the swinging foot from the current to the next support foot position, while taking into account control points and speeds to ensure a smooth trajectory that respects the constraints. The trajectory can be described using different parametric functions such as spline interpretation or Bezier curve. These parametric functions takes control points to ensure that swing foot will reach its next position smoothly, without colliding with the ground or the other leg. In this master's thesis, Beizer curve was used to describe the swinging foot trajectory using three constraint points; the current support foot, the next planned support foot and a middle point between the current and the next support foot position.

### 3.3.5 Feet Generator

This module has the role of calculating the position and orientation of both robot feet with respect to the CoM frame. In the world of robotics, joint trajectories are calculated relative to a base frame. The frame can be a fixed world frame or attached to the robot such as Torso or ankle frame. It is common to use CoM frame in a locomotion approach that depends on a concentrated mass models such as the card-table or IPM as in our work.

After computing the support and swing footsteps in world frame, the steps have to be recalculated regarding the CoM frame. This step is vital in order to achieve a correct inverse kinematic calculation for the legs end effector. The CoM position provided by the CoM generator module is used as the base frame for the target footsteps. After that, the end effector position (Target feet 3D position) become ready for the inverse kinematics calculations.

### 3.3.6 Inverse Kinematics

Inverse kinematics provide closed-form solutions to finding joint configurations that drive the end effectors of the robot to desired target positions in the three-dimensional physical space [57]. The Inverse Kinematics module takes the target positions of both feet relative to CoM frame and converts it into joint angles. The Joint angles are then sent to the robot using a PID controller that actuate the angles and move the feet to the desired position.

The Denavit–Hartenberg (D-H) convention is used for selecting frames of reference in robotics applications [58]. The D-H are considered a basic representation that robotics researchers understand for inverse kinematics calculation. The D-H convention for the NAO robot has been investigated before [59],[60] and various inverse kinematic solver have been achieved. Having a system of equations that solves the end effector position in joint space is not a hard task, but having a high accuracy, fast and singularity free solution is a challenging problem that remains an open area of research. Many popular open-sourced IK solvers are being used by NAO researchers [57], [61],[62]. The Aldebaran SDK provides a blocking and non-blocking IK solver APIs that can be employed for simple applications. The Aldebaran solver does not provide access to the solved joint angles but instead, it sends the unknown angles to the actuators. By doing so, it does not allow to have the angles written directly to the device communication manager (DCM) to be executed at the maximum frequency.

The CoM, swing and support foot trajectories are all a function of time. In each time iteration $\delta t$ of the walking process, the inverse kinematics needs to be fed with two vectors; one for the swing foot position relative to CoM and the other is for the support foot position relative the CoM.

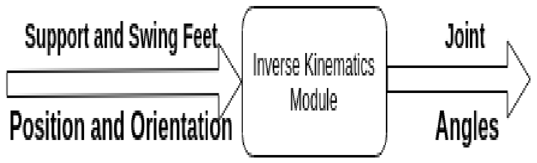


Figure 3.8: Inverse Kinematics Module

### 3.3.7 Engine Timing

The walking engine consists of several calculations and trajectories that are computed and supplied to other modules in an organized fashion. Each step consists of a single and double support phases. So it is important to make sure that the time iteration for the step is occurring at the correct point. The Step time T is divided into delta t segments which are called step time iteration $\delta_t$.

There are different ways to increment the step iteration, such as using the FSR to detect the foot contact with the ground or after the joint finish actuating the angle. There other factors that should be accounted for regarding the engine timing, such as the rate at which the system can update its joints with the new angles. The NAO robot motion runs at a maximum rate of 100 HZ and has internal PID controllers that take care of the joint speeds. The Step time iteration should equal to the step time multiplied by the minimum time required to update the motion, which is 0.01 s in the case of the NAO. In our walking engine, I have chosen 0.8s as our step time iteration is 0.008s. The time iteration is incremented in the walking engine right after the solved angles is sent to the actuators to be executed.

### 3.3.8 Engine integration

After illustrating the working mechanism of every module of the walking engine alongside its inputs and outputs, I describe how these modules interact together to generate a biped gait. The walking engine starts with input parameters which can be the desired walking velocity vector ($V_x$, $V_y$, $\theta$) or walking distance. After that, the step planner plans the future support foot positions according to the given input. The ZMP generator takes the planned support footsteps and calculate the reference ZMP trajectory so that the ZMP remains inside support polygon. Next, the CoM generator takes the reference ZMP trajectories and solve the Cart-table equations for a CoM trajectory that would produce the given reference ZMP. After having the support footsteps, ZMP and CoM trajectory for one step calculated, a step can be actuated. The swing foot generator plans the swinging foot trajectory using the current and next support foot position as control points to a function that produces a trajectory connecting the steps together. Before calling the inverse kinematic solver module, the feet generator module calculates the feet 3D position relative to the CoM reference frame. Finally, the support and swinging foot trajectories are sent to the inverse kinematics module in each time iteration. The inverse kinematics transforms the feet target positions to joint angles and sends it to the actuators PID controller to be executed. At the end of each step, the support and swinging foot are exchanged. The step planner plans the next steps, while taking the currently executed step as a starting point so that it can change the walking direction simultaneously.

## 3.4 Results

The resultant gait from the presented approach is natural, energy efficient and relatively fast. The robot is able to changes it walking direction while maintaining balance. Figure 3.9 and 3.10 shows ZMP and CoM trajectory during the walking in X and Y directions respectively. It can be noticed that the CoM trajectory can successfully follow the ZMP as desired.

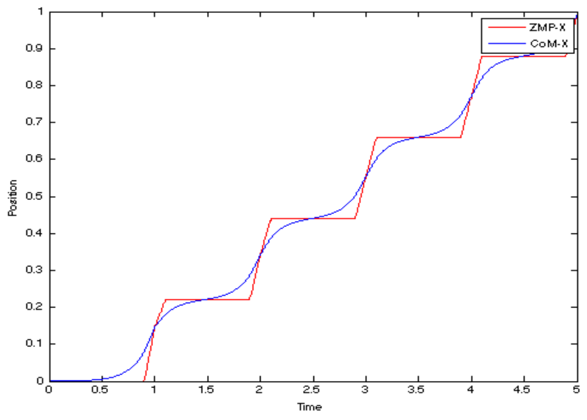


FIGURE 3.9: Inverse Kinematics Module

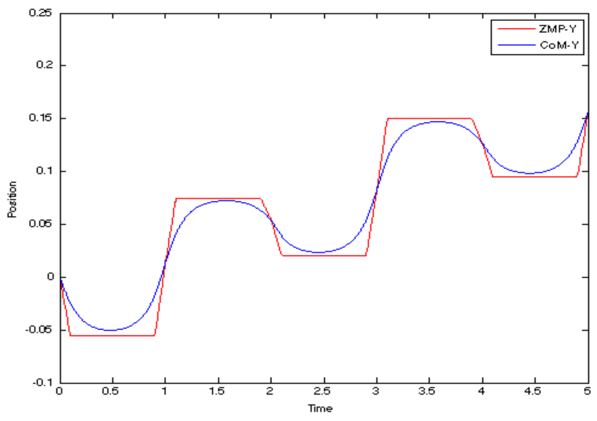


FIGURE 3.10: Inverse Kinematics Module

The time required to plan the ZMP and CoM trajectories for the walk was 0.001177 seconds and 0.078823 seconds to execute the step. The maximum speed achieved in the simulation was 0.160 m/s, which faster than the standard gait provided by Aldebaran SDK. Table 3.1 summarize the gait parameters used to generate the walk with the reported speed. The parameters were chosen by experiment to allow the robot to stably walk, with maximum possible velocity.

| Gait parameter | Chosen value |
|---|---|
| Step period | 0.8s |
| Step Height | 0.03 m |
| Stepx | 0.02 m |
| DSP percentage | 33% |
| Torso inclination | 1 rad |

TABLE 3.1: Gait parameters

In contrast with the Aldebaran walk, the robot was able to walk while stretching its knees, which makes the gait natural and energy efficient. Table 3.2 provides a comparison of our gait with the Aldebaran gait.

| Gait parameter | Aldebaran Gait | Our Gait |
|---|---|---|
| Model | Cart-table | Inverted pendulum |
| Max speed | 0.119 m/s | 0.160 m/s |
| Swinging Foot trajectory | Spline Approximation | Beizer Curve |
| CoM vertical motion | Constant | Variable |

TABLE 3.2: Gait parameters

## 3.5 Summery

The walking engine presented in this chapter is the most basic ZMP based engine. The engine consisted of several modules that are the minimum required to produces a ZMP based locomotion for the biped. There are many other modules that can be integrated to improve the quality and stability of the walk. The engine discussed in this chapter is an open loop which means that it will not adjust its parameters if the walked was disturbed by external sources. There are many perturbation sources that exist in the working environment of the robot, some are expected while the others are not. Some disturbance's effects are large enough to require recalculating the planned steps. Moreover, the torso is assumed to remain at a certain angle and the feet sole are assumed parallel to the floor. Any simple inclination in the ground can make these assumptions invalid and cause the robot to fall. These factors can be managed by making the engine a closed loop, by adding different feedback mechanisms. The humanoid robots are usually equipped with many sensors that can be employed to provide a feedback so that the walking can be adjusted to keep the gait concurrent. As it was mentioned before, humanoids are expected to work in dynamic environments that contain many perturbations and hence it is always urged to have a closed loop walking engine able to adjust itself according to with the surrounding changes.

Chapter 4

# Arms Role In Locomotion Stability

## 4.1 Introduction

Humanoid robots are expected to operate in dynamic environments alongside other robots, so by improving the balance recovery, we improve the overall robustness of the robot and decrease its dependencies on the human operator.

The three strategies mentioned above, ankle, hip and stepping, have gained a considerable amount of attention and research. These strategies are inspired by human actions when regaining balance and preventing a fall. Since humans have a great ability to maintain balance during a fast locomotion or while performing complex actions, then a key to improve the overall stability of humanoid robots can be obtained from humans balancing behaviors. On the same hand, the robotics community has a constant desire to make the motion of humanoid robots natural and human-like to increase humans-machine interactions. So by adopting more of humans balancing actions, we move closer towards the goal.

Another strategy that is employed by humans to enhance stability is the arm motion. Arm motions is a vital balance recovery strategy that plays a role in balancing the human motions and prevent falling. It also enhances the locomotion dynamics and makes the gait energy efficient. However, adopting the arm motions to stabilize the humanoid during locomotion did not receive as much investigation as the other strategies.

In this chapter, I discuss the idea of employing the arm moves towards the biped stability. The related literature is reviewed. I also present a new approach on utilizing arms rotation for balance recovery of NAO. The proposed method is simple, robust and independent of the walking engine. I present its structure and explain its working mechanism. I also present simulations that test and evaluate the proposed approach.

## 4.2 The arms position role in static and dynamic stability

Static stability informs us if the robot is stable in a stationary state. The gait is said to be statically stable, and a posture is balanced if the gravity line from its center of mass (GCoM) falls within the convex hull of the foot support area [10]. The stability margin, defined as the distance of the GCoM from the edge of the support polygon, is proportionally related to the static stability level for a posture. The further the GCoM is from the edges of the support polygon, the more statical stability achieved. The horizontal GCoM of a multi-linked robot is given by equations 2.2 and 2.3.

From the given equations, it can be noticed that if a link with a significant mass percentage changes its position, the GCoM of the whole body will be shifted towards the new position of the link. The arms of the humanoids contain a weighty amount of mass. Thus by changing the arms position, the GCoM of the robot will be shifted towards the arm's new position and will affect the overall statical stability of the robot. Figure 4.1 demonstrates the idea of using the arms position to balance a posture. As shown in the figure, the NAO robot can achieve a statical balance while standing on one leg. The robot was able to bring its GCoM over the support polygon area by distributing its arms and leg as shown. It can be noticed that if the robot changes the position of one link while keeping the other in the same position, the robot will lose stability and will fall toward the side.

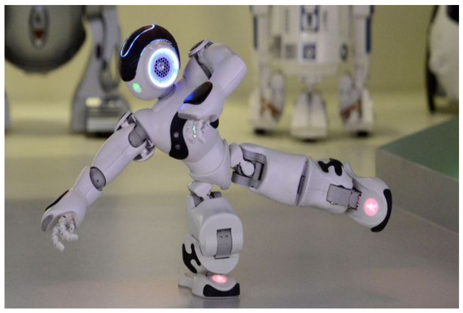


Figure 4.1: The NAO robot able to stand on one foot while in balance by using its leg and arms to position the CoM over the support polygon area

The dynamic stability apprises us about the robot stability while in motion. The dynamic stability margin, which is the distance between the ZMP from the support polygon edges [35], is proportionally related to the stability level of the dynamic gait. Equations 2.4 and 2.5 give the X and Y component of the ZMP. If we ignore the applied torques for now; one can notice that by changing the arms position or acceleration, the ZMP of the robot will be modified. So repositioning the arms influence the ZMP which in turn affects the dynamic stability of the robot.

## 4.3 Arm Effects on The Rotational Inertia

The rotational inertia can be defined as a physical property that combines the mass and distribution of the particles around the rotation axis [63]. The rotational inertia of an object is proportionally related to the amount torque needed to rotate an object with a certain acceleration or change its rotational velocity.

If a humanoid received a sinister force trying to rotate it about an axis, the robot could avoid falling by increasing its moment of inertia alongside the axis. The locomotion on uneven hard floors can generate oscillations that can put the gait in an unstable situation. By expanding the rotational inertia of the robot around the oscillation axis, the rotational velocity would be reduced, and the oscillations would be eventually damped.

Whenever a fall cannot be prevented, the moment of inertia can be used to reduce the rotational speed during the fall and changes its direction to minimize the damages. The same phenomenon is used by gymnast, where they tend to change their arms position control their spinning direction and speed.

For a multi-object body like a robot, the rotational inertia can be calculated as follows:

$$I = \int_0^Q r^2 dm \tag{4.1}$$

Where $r$ is the object distance from an axis; $Q$ is the entire mass. From equation 4.1 one can notice that the moment of inertia is related to the robot's body mass distribution around an axis, the further away the links are, the higher the moment of inertia. Thus the robot can increase its moment of inertia by moving its arms away from the axis. Figure 4.2 shows a robot expanding its arms in T-shape to increase its moment of inertia along the Y-axis which can damp a side oscillations or avoid falling towards the left or right side of the robot.

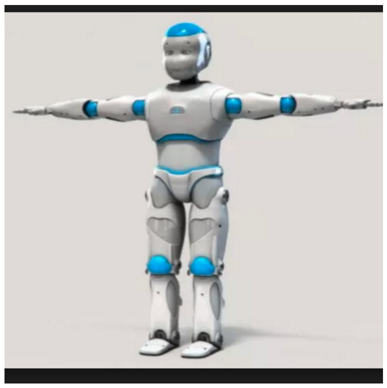


FIGURE 4.2: A Humanoid increasing its rotational inertia by spreading its arms alongside the Y-axis

For a body that can be rotated around the three axises such as a humanoid robot, the rotational inertia can be described using a symmetric 3x3 matrix; A general form of the inertial matrix for the NAO robot is shown below:

$$\begin{matrix} I_{xx} & I_{xy} & I_{xz} \\ I_{yx} & I_{yy} & I_{yz} \\ I_{zx} & I_{zy} & I_{zz} \end{matrix}$$

## 4.4 Arms rotations Strategy

The robot upper body motion and external perturbations can exert torque to the CoM. If the received torque was not managed, it could let the ZMP to leave the support polygon area, which put the robot in an unstable situation. Modeling the robot dynamics with the enhanced LIPM allows us to account for the angular momentum change at the CoM. The ankle, hip and stepping strategies consist of a sequence of actions which the robot performs to counteract the torque at the CoM. The magnitude of the disturbances indicates which of these strategies need to be deployed.

The angular momentum $L$ of a rigid body is proportional o rotational inertia of the body $I$ and the angular velocity $\omega$.

$$L = I\omega \tag{4.2}$$

For a multi-link object, the total angular momentum equals the sum of the angular momentum for every link. The total angular momentum of a humanoid robot equal the angular momentum of the torso, legs, arms, and head as shown below:

$$L_{body} = I_{torso}\omega_{torso} + \sum_{i=0}^{2} I_{leg_i}\omega_{leg_i} + \sum_{i=0}^{2} I_{arm_i}\omega_{arm_i} + I_{head}\omega_{head} \tag{4.3}$$

Equation 4.3 shows that the rotational velocity of the arms affects the overall angular momentum of the body and hence the stability of the robot. This demonstrates the importance of the arm rotation strategy regarding the balance of the humanoid robots.

The arms can be set in different motions to achieve the same goal. The arms of human or humanoid can be thought of as two flywheels attached to the shoulder. Swinging the arm in a particular direction exerts torque to the CoM. Thus by rotating the arms in a specific direction and time, we can counteract the disturbing torque at the CoM and avoid falling.

Humans nervous system employs the arms in rotations to prevent a fall. When a fall is occurring, the arms are engaged in rapid rotations toward the falling direction, trying to bring the upper body back to the upright position as shown in figure 4.3 .

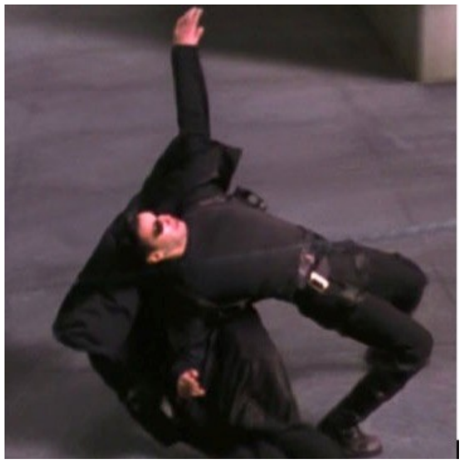


FIGURE 4.3: Rotating the arms to regain balance

This action can be reasoned with Newton's third law of motion which states that in a closed and isolated system, no torque can be exerted in any matter without the exertion on some other matter of an equal and opposite torque. If two rotatable objects are attached, then by rotating one object towards a particular direction, the other object will be rotated in the opposite direction. The same phenomenon is used to position satellites in space, where a system of flywheels are rotated in the opposite direction of the desired satellite orientation. Figure 4.4 illustrates the idea.

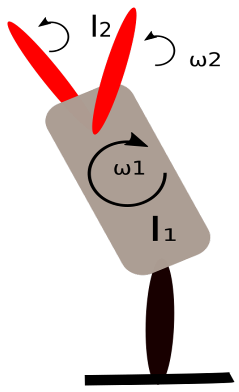


FIGURE 4.4: Arms Rotation Effect On The Body

The robot may be able to prevent a fall by rotating both its arms towards the falling direction. The rotation will produce a counteracting torque at the CoM that will push the upper body to the upright position. The amount of the counteracting torque generated by the arms rotation is proportionally related to the arms weight, length and the shoulder joint rotational velocity. Thus the amount of recovery from using the arms rotation strategy is limited to the robot design specifications.

### 4.4.1 Previous approaches

Thee use of the arm's motion to prevent the robot from falling is still new area of research that had not enjoyed as much attention as the ankle, hip and stepping strategies. Nakada et al. [64] introduced the Arms Rotation Strategy (ARS) to maintain the gait balance. In their work, they employed machine learning to discover how to react properly depending on the circumstances, instead of attempting a physical computation on the direction, timing, and strength of the arm rotations. Their final reinforcement learning goal was a stable balance, which was used to evaluate each trial. They tested their approach on a physically simulated robot. The outcomes of the machine learning used by [64] were not discussed to allow others to utilize the strategy with a walking engine. Additionally, Kasaei et al. [30] have used arm rotations to add torque to the ankle strategy. Their approach improved the Ankle strategy's balancing capabilities which is a part of their push recovery controller.

In this thesis, I present a model- specific study on adopting the arm rotations to the NAO. I integrate the strategy with the on-bored sensors to achieve balancing recovery against unexpected disturbance sources. Our method is, robust, computationally cheap and independent of the walking engine. The NAO is currently being used different RoboCup competitions such as the SPL and Rescue, our approach can be important to the contests. I open the box on how the strategy is applied by specifying the logic of the used algorithms to allow others to use it and build on it. I also present a simple method to control the torque from the arm rotations and the approach limitations is discussed. Additionally, I consider applying the strategy to balance a ball kick as it is a common source for falling during the RoboCup SPL. Finally, various simulations to evaluate our approach are provided.

### 4.4.2 Proposed Methodology

Figure 4.5 shows the proposed arms rotation strategy control algorithm in a hierarchal structure. The algorithm has the role of rotating the arms when required to counteract the angular momentum associated with the CoM to keep the ZMP from leaving the support polygon area.

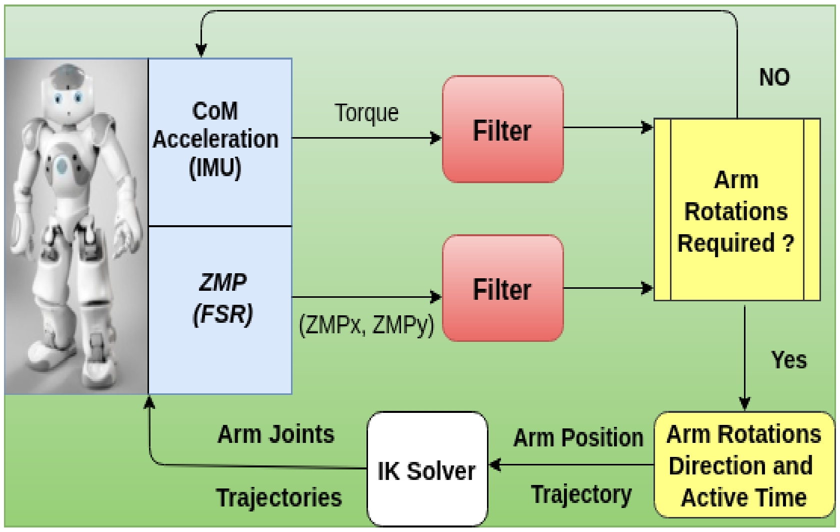


FIGURE 4.5: Arm Rotations Strategy Control Algorithm Flowchart

The control algorithm takes the measured ZMP and the flywheel torque as inputs and produces a proper arm rotations in a specific direction and active time. The ZMP is estimated using the Force Sensitive resistors (FSR) attached at the robot feet and flywheel torque is calculated using the CoM acceleration measured using the inertial measurement Unit (IMU).

Both signals are passed through a moving average filter to remove the existing noise. The moving average filter is a simple low pass finite impulse response (FIR) filter that is popular for signal smoothing and noise removal applications. The filter takes L-size samples from the input at a time and calculates the average of it to produce a single value output. Equations 4.4 provides a mathematical representation of the moving average filter.

$$Y_k = \frac{X_k + X_{k-1} + X_{k-2} + X_{k-3} + .. + X_{k-N}}{N} \tag{4.4}$$

Where $Y_k$ is the filter output, $X_k$ is the input and N is the sample size. In this MS thesis, the filter sample size was set to five by experiment, as further increasing of the window size will produce severe smoothing and smaller perturbations may not be detected. On the other hand, a small window size will not remove all the undesired noise from the signal. The transfer function of the moving average filter in the Z-domain can be obtained from equation 4.4 using the conversion tables as shown in equation 4.5.

$$\frac{Y_k}{Z_k} = \frac{1}{N}(\frac{1 - Z^{-N}}{1 - Z^{-1}}) \tag{4.5}$$

Substituting N=5, we get the following transfer function:

$$H(z) = \frac{z^4 + z^3 + z^2 + z + 1}{5z^4} \tag{4.6}$$

The left part of Figure 4.6 shows the pole-zero diagram for the filter transfer function. There are four poles at $z_1 = -(-1)^{1/5}$, $z_2 = -(-1)^{2/5}$, $z_3 = -(-1)^{3/5}$, $z_4 = -(-1)^{4/5}$.

The frequency response, in terms of normalized frequency ($\omega$), is obtained by substituting $z = e^{j}w$ as shown in equation 4.7.

$$H(e^{j\omega}) = \frac{1}{5} + \frac{1}{5}z^{-j\omega} + \frac{1}{5}z^{-2j\omega} + \frac{1}{5}z^{-3j\omega} + \frac{1}{5}z^{-4j\omega} \tag{4.7}$$

The right part of figure 4.6 shows the magnitude component of $h(e^{j\omega})$. The plot indicates that the moving-average filter passes low frequencies and attenuates high frequencies.

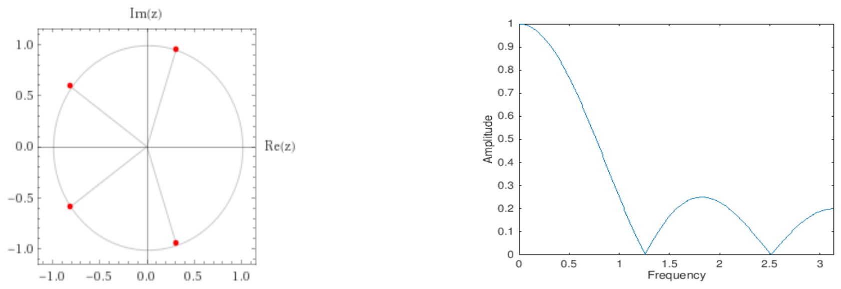


FIGURE 4.6: Pole-zero diagram (left) and the magnitude of the frequency response (right) of a $4^{th}$ order MA filter

If the ZMP reaches any of the support polygon boundaries, we rotate the arms towards the same boundary. The rotation exerts a torque to the flywheel that pushes the ZMP away from the polygon boundary to a stable position. On the same hand, if the ZMP is inside the support polygon and the robot received disturbing torque above a certain threshold, then the torque will rotate the flywheel and push the ZMP to an unstable position. By rotating the arms to the same direction of the flywheel rotation, we exert a counteracting torque that reduces the disturbing torque and keep the ZMP from leaving the support polygon area. The torque threshold is determined by experiment. The rotation active time is determined based on the magnitude of the measured flywheel torque. Keeping the arms in the final swing position for short time duration allows the disturbance source to be changed while the ZMP is kept away from the polygon boundary.

### 4.4.3 Experimental Simulations and Results

The proposed methodology was tested in simulation using Webots, for a precise computation and examination, as well as to avoid the risk of damaging the real robot.

An experiment was set to measure the effect of arm rotations of the NAO on the ZMP. In this experiment, the robot was set to stand in place rotating the arms towards the front of the coronal plane, while we monitor the ZMP. The graph in figure 4.7 shows the ZMP, to the ankle frame, as a function of the arm rotation. A front arm rotation was able to shift the ZMP by 17 mm towards the front of the support polygon, while a rear rotation has shifted the ZMP by 12 mm. This result provides an insight into how much the arms rotation can alter the ZMP and hence affect the stability.

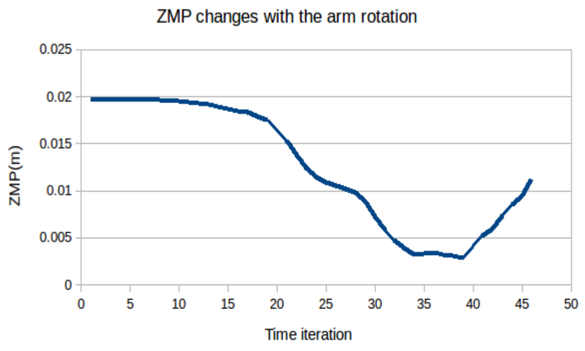


FIGURE 4.7: ZMP changes associated with Arms rotation

To judge the effectiveness of the proposed methodology, three experiments were conducted while all the other balancing strategies and fall managers were switched off. In the first experiment, The robot was set to walk on a flat floor while we apply external forces to it from one direction. The disturbance forces were propelling the robot toward the back with incre- menting magnitudes. The trial aimed to measure the amount of force that the robot could manage using the algorithm. After that, the forces were applied from 0, 90, 180 and 270 degrees. Figure 4.8 and 4.9 present the outcome of the simulation, which shows the amount of the forces that the NAO can resist is enlarged when the strategy is employed.

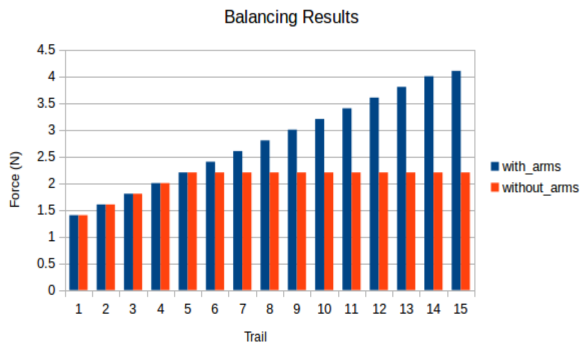


Figure 4.8: The result of the experiment when various magnitudes of disturbance forces, applied towards the back, with and without the arm rotations.

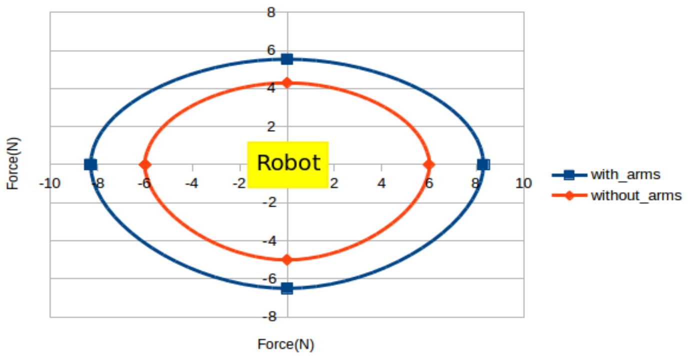


Figure 4.9: The result of the experiment when various magnitudes of disturbance forces, applied from different directions, with and without the arm rotations.

In the second experiment, the robot was made to walk on rough terrain, in which it fails to pass without using a recovery controller. The strategy enabled the robot to robustly recover from the perturbations represented by rocks on the floor. Figure 4.10 presents the ZMP trajectory during the experiment, with and without using the strategy. It is clear that the arm rotations bounded the ZMP in the middle of the support polygon and enhanced the overall stability.

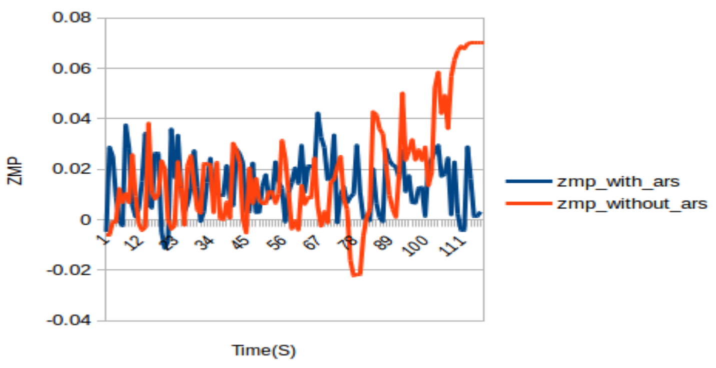


FIGURE 4.10: ZMP while walking on Complex terrain containing sharp edged rocks- with and without the arm rotations.

The third experiment aimed to highlight the role of the static arm posture in keeping the ZMP away from the support polygon boundary. The robot was made to walk into a 15-degree inclined ramp while having the arms extended to the front. The static arm posture kept the ZMP away from the rear boundary of the support polygon and allowed the robot to walk over the ramp without falling.

A video demonstrating the experiments is available at: https://drive.google.com/file/d/0BzhovByakWnnaFpqdFNHOTlvYXc/view

## 4.5 Balancing The Kicking Action Using the Arm Rotation

FIGURE 4.11: NAO robot performing a ball kicking action

Soccer humanoid is gaining popularity as the RoboCup competition is spreading around the world. The area intends to encourage software development for humanoid robots in different categories such as Computer Vision, Motion, and Networking. The choice of soccer environment comes from the fact that soccer played in a dynamic environment where many robots are moving fast in different directions and performing soccer actions like a kick. The environment complexity encourages the development of balancing strategies and optimizing the locomotion so that the robot can frequently change its walking direction to avoid obstacles.

Concerning RoboCup soccer, the robots are required to perform powerful kicks in different directions. A strong kicking motion can induce instabilities and causes the robot to fall. In fact, collision with other robots and powerful ball kicks, are the most common hazards that cause the humanoids to fall during the RoboCup SPL. A frequent falling robots lower the participating team performance and waste the battery power of the robot.

In this section, I present our work on stabilizing the ball kicking actions for the soccer humanoid using the arm rotation strategy. I expose our developed ball kick that is inspired by human soccer players.I integrate the arm rotations to the kicking action to improve its stability, and I present simulations to evaluates the approach.

### 4.5.1 Generating a ball kick

The kicking process starts by realizing the current and target ball position. In this phase, the robot finds the relative position of the ball, while having a desired target position set. After that, the robot performs several steps to propel the ball forward to its final spot. The first step to generate a ball kick is to set the robot in a stable position within a reachable distance to the ball. The robot is required to be in standing posture with a defined leg separation. The next step is to plan the kicking foot trajectory. The trajectory is specified by a set of points that the kicking foot needs to follow to perform the kicking action. After that, the planned path of the kicking leg is sent to the inverse kinematics module to be executed. During the kicking motion, a stability module is required to keep the robot in balance.

The process of propelling the ball starts with leaning action, in which the robot shifts its center of mass (CoM) towards the support leg. After that, the robot lifts its kicking leg off the ground to be ready to perform the kicking action. Next, the kicking foot is swigged over the planned trajectory to propel the ball. Then the kicking foot is returned to its initial lifted position. Finally, the robot put the kicking leg to the ground which shifts the CoM back between the legs.

### 4.5.2 Kicking leg path planning

The kicking foot trajectory describes the foot motion from its initial position to the ball. The trajectory specifies the exact route that the foot follows to reach its final destination. Beizer curve was employed before to define the swinging leg trajectory in the walking engine. In the same manner, Beizer curve is used to plan the kicking foot trajectory. The foot position after the preparation stage is given as the initial control point and the ball edge is used as the final control point. The two points are given as inputs to the Beizer function which produce a smooth path connecting the two points. Beizer function is computationally cheap as it uses simple math to determine a trajectory, which makes suitable for such an application.

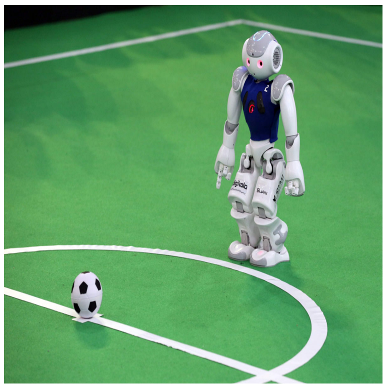


FIGURE 4.12: Robot planning the kicking leg path

### 4.5.3 Stabilizing the ball kick

During the ball kick, the kicking foot moves fast with specific torque and speed towards the ball. Since the ball is flexible, its reaction force is smaller than to affect the robot balance. However, the swinging motion of the leg induces angular momentum changes at the CoM that can make the robot unstable. The situation is similar to having a disturbance force applied to the CoM.

The arms rotation strategy presented in the previous section can be employed to regulate the angular momentum change associated with the kick. The arm can be rotated in the opposite direction of the kick to decrease the applied torque at the CoM. The arm rotation strategy exerts a torque to the CoM to counteract the disturbing torque resulted from the kick. Since the legs have more mass and higher joint torques than the arms, the arm rotations can not cancel all the exerted torque from the kick, but it helps to reduce the overall angular momentum change at the CoM and prevent the ZMP from leaving the support polygon.

### 4.5.4 Results

Our research is related to the RoboCup competition and since falling during a ball kick is very common to the contest, one experiment was conducted to evaluate the role of the arms motion in balancing a ball kick. In the experiment, the robot performed powerful ball kick with and without using the arm rotations. The arm rotations was able to enhance the kick stability, as it regulates the torque produced by the swinging leg and the upper body movement. Figure 4.13 shows the ZMP trajectory during the ball kick with and without using the stability approach.

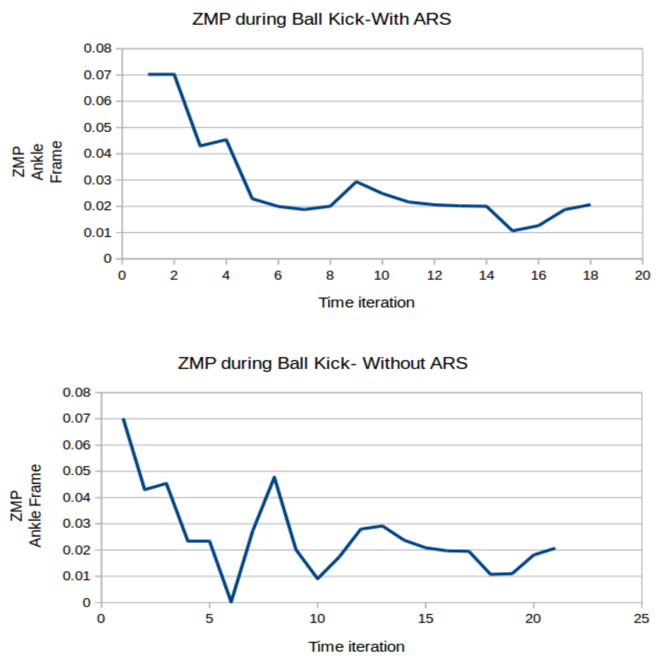


FIGURE 4.13: ZMP trajectory during ball kick with and without using ARS

## 4.6 Summery

This chapter presented a model-specific study on using the arms rotation to maintain the balance of the humanoid robot, NAO, during locomotion with the presence of external perturbations. I have considered the use of arms motion and static posture to achieve stability. Enhancing the robot's static and dynamic balance were considered. The effect of the arms position on the ZMP and the body's moment of inertia was reasoned and employed to achieve stability. The role of the arms rotation strategy in preventing fall was examined and illustrated to regulating the angular momentum at the CoM in regard to keep the ZMP inside the support polygon. Our robotic system was modeled using the LIPM with a flywheel to account for the torque at the CoM during locomotion.

One arms motion control algorithm was presented which employs sensory feedback to monitor the ZMP alongside the torque at the CoM and make use of the arms rotations strategy to reduce the disturbing torque and hold the ZMP inside the support polygon. The approach is simple, computationally inexpensive and can be adapted to any walking without the need for changes. The proposed arms rotations are human inspired and results from different studies on the human reactions to prevent falling were employed to instruct the robot on how to rotate its arms in different situations. The presented algorithm was implemented and tested using the Aldebaran NAO H25 robot in a simulation without the use of any other balancing strategy. Experimental simulations showed that the given approach is competent and capable of keeping the robot stable against external disturbances that the robot fail to maintain balance against it.

Employing the arm rotation during a ball kick kept the ZMP bounded away from the instability regions and smoothed out the trajectory. The approach enhanced the dynamic stability of the robot as the arm rotations produced a counteracting torque to balance the kicking action.

# Chapter 5

# Conclusion and Future work

This Chapter provides an overview of the work discussed in this thesis with its contributions. Additionally, I present some of the future work.

## 5.1 Conclusion

I introduced all the hardware platforms that were used in this study. The Aldebaran NAO is the humanoid robot used to implement and test all of our work. I provided a detailed review of the NAO H25 model in appendix A, illustrating its featured hardware and structure. I also presented all the available insights to program the NAO as well as the permissible programming languages, while discussing the advantages of each insight.

Since humanoid robots are expensive and delicate, the use of simulators is necessary and familiar to the field. In addition, simulators provide a precise way to apply and test new methodologies. Different simulators that support the NAO models are available, however, choosing which simulator to use can be tricky and may require an in-depth research. I have reviewed and compared some of the poplular simulators such as Webots, Gazebo, V-rep, and SimSprak. After that, I introduced the RoboCup Competition as it is one contest that encourages the research on humanoid motion and balance.

I introduced and defined several terms that are related to the locomotion stability. The concepts include the Center Of Mass (CoM), Center Of Pressure (CoP), Zero Moment Point (ZMP) and support polygon. I next discussed the static and dynamic stability criteria. The locomotion ability is one distinguishable feature of the bipeds and humanoids. I reviewed the primary locomotion classifiers, such as model-based, model-free, passive and active walker. After that, I present the most popular stability criterion for humanoid robots which is the ZMP criterion. I also discussed three physical models that can be used to simplify the robot's dynamics, which are the Cart-Table, Linear Inverted Pendulum (LIPM), and The Inverted Pendulum (IPM). These models provide a set of differential equations that describe the relationship between the CoM trajectory to the ZMP. There are several approaches presented to solve these equations which can be categorized into numerical and analytical methods.

After introducing all the concepts regarding the biped locomotion, I present the development of a walking engine in chapter 3. A typical ZMP based walking engine consists of several primary modules. I present each of the modules in details while explaining its working mechanism alongside its inputs and outputs. I address the process of integrating these modules together to produce a multidirectional walking engine able to make the biped walk with the desired gait. I have adopted some developed modules, that was provided by other researchers in the field, to our walking engine. Our contribution to the chapter is the development of timing module that is responsible for synchronizing the modules of the walking engine together. The timing module takes into consideration the maximum motion frequency of the hardware platform and desired time incremental $\delta t$.

Producing a quality, fast and reliable locomotion is a complicated task that is essential for adopting humanoid robots into the human's environments. However, without an excellent balancing abilities, the humanoid will not be as useful as we desire. Different sources of perturbations exist in dynamic environments. To make the robot robust and adaptive, we need to account for the expected and unexpected disturbances that the robot might face while walking. Active balancing is activated all the time and is used against the expected perturbations such as regulating the torso inclination to avoid falling. On the other hand, dynamic environments contain unexpected disturbances such as rough terrains or a pushing forces. These disturbances can cause a change of the angular momentum at the CoM. If a strategy did not manage to regulate the momentum changes, it could let the robot to lose balance. The ankle, hip, and stepping strategies are the most popular strategies employed for managing the external torques.

Using the arm motion as another recovery strategy did not receive as much attention. The approach is human inspired and is highlighted sports that require balancing the body such as gymnast and tightrope walkers. The arms can be employed in different motions to enhance the static and dynamic balance of the robot as well as to counteract any external torques at the CoM. There are few published approaches on using the arms rotations as a balancing strategy. However, the previous works lacked in details and did not provide precise results allowing others to adopt the strategy to their work. Our contribution to the chapter consisted of a study on the effect of the arms position on the static and dynamic balance of the humanoid robot. Additionally, I discussed how the arms can be used to adjust the rotational inertia of the robot and how that can be employed to prevent a fall or to decrease the falling impact.

I propose a simple control algorithm that integrates the arm rotation strategy with the on-board sensors of the humanoid robot, NAO, for balance recovery. The Control algorithm is responsible for rotating the arms in different directions to counteract the angular momentum changes at the CoM. The approach aims to stabilize the gait and prevent falling from external perturbations. In contrast with other work, our method is entirely independent of the walking engine and can be integrated to different gaits without the need for changes. The algorithm employs sensory feedback to estimate the robot body state and detect instabilities. I present a simple method to control the amount of torque from the arm rotation and the strategy limitations were discussed. Our proposed methodology was tested in simulation, and the results indicate the effectiveness of the approach in preventing the fall of NAO during the gait.

In the RoboCup contest, falling during a powerful ball kick is common because the leg swing produces torque at the CoM that tend to rotate the robot about the Y-axis. The arm rotations was employed to stabilize the ball kick by counteracting the angular momentum change during the kick. The simulations experiments proved the success of the approach in balancing the ball kick. Finally, a conclusion on the study is provided, and the future work direction is discussed.

## 5.2 Future Work

In future work, I would like to investigate the possibility of using prediction functions to determine losing stability in a faster manner. Given the current and past trajectories of the ZMP and CoM, a mathematical prediction function can be employed to guess whether the ZMP will leave the support polygon of the robot or not shortly. With predictions, we might be able to speed the push recovery controller to produce faster arms rotation responses and achiever higher stability.

# Bibliography

[1] R. Marc, “Legged robots that balance,” *Massachusetts Institute of Technology, mitpress.mit.edu/books/legged-robots-balance*, 1986.

[2] Q. Huang, S. Kajita, N. Koyachi, K. Kaneko, and K. Yokoi, “high stability, smooth walking pattern for a biped robot.,” *IEEE International Conference on Robotics and Automation*, 1999.

[3] RoboCupSoccer, “Standard platform,” 2017. `http://spl.robocup.org/`.

[4] Wikipedia, “Center of mass,” 2018. `https://en.wikipedia.org/w/index.php?title=Center_of_mass&oldid=826373631`.

[5] G. A., “Postural stability of biped robots and the foot rotation indicator (fri) point,” *International Journal of Robotics Research*, 1999.

[6] V. M, “Biped locomotion: Dynamics, stability, control and application,” *Scientific Fundamentals of Robotics*, 1990.

[7] N. Shafii, “Development of an optimized omnidirectional walk engine for humanoid robots,” *PhD diss, University of Porto, Portugal*, 2015.

[8] S. G, W. H, and L. Z., “A novel biped pattern generator based on extended zmp and extended cart-table model.,” *Inter- national Journal of Advanced Robotic Systems, 12(7): 1–15. DOI: 10.5772/60783*, 2015.

[9] S. Kajita, K. Kaneko, K. Harada, F. Kanehiro, K. Fujiwara, and H. Hirukawa, "Biped walking on a low friction floor," in *2004 IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS) (IEEE Cat. No.04CH37566)*, vol. 4, pp. 3546–3552 vol.4, Sept 2004.

[10] A. Goswami, "Postural stability of biped robots and the foot-rotation indicator point," *Int. J. Rob. Res*, vol. 18, no. 6, p. 523–533, 1999.

[11] P. Sardain and G. Bessonnet, "Forces acting on a biped robot. center of pressure-zero moment point," *IEEE Transactions on Systems, Man, and Cybernetics - Part A: Systems and Humans*, vol. 34, pp. 630–637, Sept 2004.

[12] M. Arbulu, "Stable locomotion of humanoid robots based on mass concentrated model," 04 2009.

[13] C. Graf and T. Rofer, "A closed-loop 3d-lipm gait for the robocup standard platform league humanoid," *IEEE-RAS International Conference on Humanoid Robotics*, pp. 15–22, 2010.

[14] J. Alcaraz, D. Herrero-Perez, and H. Barberá, "Robust feedback control of zmp-based gait for the humanoid robot nao," *The International Journal Of Robotics Research*, vol. 32, pp. 1074–1088, 09 2013.

[15] M. Dekker, "Zero-moment point method for stable biped walkin," *Eindhoven, University of Technology*, 2009.

[16] S. Kajita, F. Kanehiro, K. KANEKO, K. Yokoi, and H. Hirukawa, "The 3d linear inverted pendulum model: A simple modeling for a biped walking pattern generation," vol. 1, pp. 239 – 246 vol.1, 02 2001.

[17] K. Nishiwaki, S. Kagami, Y. Kuniyoshi, M. Inaba, and H. Inoue, "Online generation of humanoid walking motion based on a fast generation method of motion pattern that follows desired zmp," in *IEEE/RSJ International Conference on Intelligent Robots and Systems*, vol. 3, pp. 2684–2689 vol.3, 2002.

[18] A. Takanishi, H. ok Lim, M. Tsuda, and I. Kato, "Realization of dynamic biped walking stabilized by trunk motion on a sagittally uneven surface," in *EEE International Workshop on Intelligent Robots and Systems, Towards a New Frontier of Applications*, pp. 323–330 vol.1, Jul 1990.

[19] K. Erbatur and O. Kurt, "Natural zmp trajectories for biped robot reference generation," *IEEE Transactions on Industrial Electronics*, vol. 56, pp. 835–845, March 2009.

[20] Y. Choi, B.-J. You, and S.-R. Oh, "On the stability of indirect zmp controller for biped robot systems," in *2004 IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS) (IEEE Cat. No.04CH37566)*, vol. 2, pp. 1966–1971 vol.2, Sept 2004.

[21] K. Harada, S. Kajita, K. Kaneko, and H. Hirukawa, "An analytical method on real-time gait planning for a humanoid robot," in *4th IEEE/RAS International Conference on Humanoid Robots, 2004.*, vol. 2, pp. 640–655 Vol. 2, Nov 2004.

[22] I. W. Park and J. Y. Kim, "Fourier series-based walking pattern generation for a biped humanoid robot," in *2010 10th IEEE-RAS International Conference on Humanoid Robots*, pp. 461–467, Dec 2010.

[23] N. Snafii, A. Abdolmaleki, N. Lau, and L. P. Reis, "Development of an omnidirectional walk engine for soccer humanoid robots," *International Journal of Advanced Robotic Systems*, vol. 12, no. 12, p. 193, 2015.

[24] S. Kajita, F. Kanehiro, K. Kaneko, K. Fujiwara, K. Harada, K. Yokoi, and H. Hirukawa, "Biped walking pattern generation by using preview control of zero-moment point," in *2003 IEEE International Conference on Robotics and Automation (Cat. No.03CH37422)*, vol. 2, pp. 1620–1626 vol.2, Sept 2003.

[25] S. Kagami, T. Kitagawa, K. Nishiwaki, T. Sugihara, M. Inaba, and H. Inoue, "A fast dynamically equilibrated walking trajectory generation method of humanoid robot," *Autonomous Robots*, vol. 12, pp. 71–82, Jan 2002.

[26] S. Kagami, T. Kitagawa, K. Nishiwaki, T. Sugihara, M. Inaba, and H. Inoue, "A fast dynamically equilibrated walking trajectory generation method of humanoid robot," *Autonomous Robots*, vol. 12, pp. 71–82, Jan 2002.

[27] S. Kajita, T. Nagasaki, K. Kaneko, and H. Hirukawa, "Zmp-based biped running control," *IEEE Robotics Automation Magazine*, vol. 14, pp. 63–72, June 2007.

[28] Wikipedia, "Tridiagonal matrix algorithm — wikipedia, the free encyclopedia," 2017. `https://en.wikipedia.org/w/index.php?title=Tridiagonal_matrix_algorithm&oldid=813646304`.

[29] N. Shafii, N. Lau, and L. Reis, "Learning to walk fast: Optimized hip height movement for simulated and real humanoid robots," *Journal of Intelligent Robotic Systems*, vol. 80, pp. 1–17, 02 2015.

[30] S. M. Kasaei, N. Lau, A. Pereira, and E. Shahri, "A reliable model-based walking engine with push recovery capability," pp. 122–127, April 2017.

[31] T. Komura, H. Leung, S. Kudoh, and J. Kuffner, "A feedback controller for biped humanoids that can counteract large perturbations during gait," pp. 1989 – 1995, 05 2005.

[32] T. Komura, A. Nagano, H. Leung, and Y. Shinagawa, "Simulating pathological gait using the enhanced linear inverted pendulum model," *IEEE Transactions on Biomedical Engineering*, vol. 52, pp. 1502–1513, Sept 2005.

[33] J. Pratt, J. Carff, S. Drakunov, and A. Goswami, "Capture point: A step toward humanoid push recovery," in *2006 6th IEEE-RAS International Conference on Humanoid Robots*, pp. 200–207, Dec 2006.

[34] Wikipedia, "Flywheel," 2018. `https://en.wikipedia.org/w/index.php?title=Flywheel&oldid=820253493`.

[35] B. P. Mrozowski J, Awrejcewicz J, "Analysis of stability of the human gait," *Journal of Theoretical And Applied Mechanics ,Warsaw*, 2007.

[36] K. Erbatur and O. Kurt, "Natural zmp trajectories for biped robot reference generation," vol. 56, pp. 835 – 845, 04 2009.

[37] M. Yilmaz, U. Seven, K. C. Fidan, T. Akbaş, and K. Erbatur, "Circular arc-shaped walking trajectory generation for bipedal humanoid robots," in *2012 12th IEEE International Workshop on Advanced Motion Control (AMC)*, pp. 1–8, March 2012.

[38] F. Rui, S. Nima, L. Nuno, L. Reis, and A. Abdolmaleki, "Diagonal walk reference generator based on fourier approximation of zmp trajectory," pp. 1–6, 04 2013.

[39] N. Snafii, A. Abdolmaleki, N. Lau, and L. P. Reis, "Development of an omnidirectional walk engine for soccer humanoid robots," *International Journal of Advanced Robotic Systems*, vol. 12, no. 12, p. 193, 2015.

[40] D. E Angelaki and T. Yakusheva, "How vestibular neurons solve the tilt/translation ambiguity comparison of brainstem, cerebellum, and thalamus," vol. 1164, pp. 19–28, 06 2009.

[41] N. Shafii, A. Abdolmaleki, R. Ferreira, N. Lau, and L. P. Reis, *Omnidirectional Walking and Active Balance for Soccer Humanoid Robot*, pp. 283–294. Berlin, Heidelberg: Springer Berlin Heidelberg, 2013.

[42] D. C. Asmar, B. Jalgha, and A. Fakih, "Humanoid fall avoidance using mixture of strategies," *International Journal of Humanoid Robotics*, vol. 09, no. 01, p. 1250002, 2012.

[43] M. Mohades Kasaei, H. Kasaei, E. Shahri, A. Ahmadi, N. Lau, and A. Pereira, "How to select a suitable action against strong pushes in adult-size humanoid robot: Learning from past experiences," 05 2016.

[44] Q. Huang, K. Kaneko, K. Yokoi, S. Kajita, T. Kotoku, N. Koyachi, H. Arai, N. Imamura, K. Komoriya, and K. Tanie, "Balance control of a piped robot combining off-line pattern with real-time modification," 02 2000.

[45] J.-Y. Lee and J.-J. Lee, "A torso-moving balance control strategy for a walking biped robot subject to external continuous forces," *International Journal Of Humanoid Robotics*, vol. 12, p. 1550003, 01 2015.

[46] S.-H. Lee and A. Goswami, "A momentum-based balance controller for humanoid robots on non-level and non-stationary ground," *Autonomous Robots*, vol. 33, pp. 399–414, Nov 2012.

[47] B. Stephens, "Humanoid push recovery," in *2007 7th IEEE-RAS International Conference on Humanoid Robots*, pp. 589–595, Nov 2007.

[48] R. E. Goddard, H. Hemami, and F. C. Weimer, "Biped side step in the frontal plane," in *1981 20th IEEE Conference on Decision and Control including the Symposium on Adaptive Processes*, pp. 147–155, Dec 1981.

[49] A. Yasin, Q. Huang, Q. Xu, and W. Zhang, "Biped robot push detection and recovery," *Information and Automation (ICIA), 2012 International Conference*, pp. 993–998, June 2012.

[50] S.-J. Yi, B.-T. Zhang, D. Hong, and D. D. Lee, "Learning full body push recovery control for small humanoid robots," in *2011 IEEE International Conference on Robotics and Automation*, pp. 2047–2052, May 2011.

[51] S. J. Yi, B. T. Zhang, D. Hong, and D. D. Lee, "Online learning of a full body push recovery controller for omnidirectional walking," in *2011 11th IEEE-RAS International Conference on Humanoid Robots*, pp. 1–6, Oct 2011.

[52] A. Hofmann, M. Popovic, and H. Herr, "Exploiting angular momentum to enhance bipedal center-of-mass control," in *2009 IEEE International Conference on Robotics and Automation*, pp. 4423–4429, May 2009.

[53] A. Hofmann, "Robust execution of bipedal walking tasks from biomechanical principles ," *Institute of Technology. Dept. of Electrical Engineering and Computer Science, Massachusetts*, 08 2007.

[54] Aldebaran, *Locomotion Control*, 2013 (accessed December 3, 2017). `http://doc.aldebaran.com/2-1/naoqi/motion/control-walk.html#control-walk`.

[55] D. Gouaillier, C. Collette, and C. Kilner, "Omnidirectional closed-loop walk for nao," *2010 10th IEEE-RAS International Conference on Humanoid Robots*, 12 2010.

[56] S. Hong, Y. Oh, Y.-H. Chang, and B.-J. You, "A walking pattern generation method for humanoid robots using least square method and quartic polynomial," 01 2009.

[57] N. Kofinas, E. Orfanoudakis, and M. Lagoudakis, "Complete analytical inverse kinematics for nao," *2013 13th International Conference on Autonomous Robot Systems* 04 2013.

[58] W. contributors, "Denavit–hartenberg parameters — wikipedia, the free encyclopedia," 2018. [Online; accessed 16-January-2018].

[59] J. Denavit and R. S. Hartenberg, "A kinematic notation for lower-pair mechanisms based on matrices.," *Trans. of the ASME. Journal of Applied Mechanics*, vol. 22, pp. 215–221, 1955.

[60] Hartenberg, R. Scheunemann, and J. Denavi, "Kinematic synthesis of linkages," vol. 56, 1964.

[61] T. Rofer, T. Laue, and A. Muhlenbrock, "Bhuman Team description for Robocup 2017," 2017.

[62] B. Hengst, "Runswift Walk2014 Report robocup Standard Platform League," 2014.

[63] W. contributors, "Moment of inertia — wikipedia, the free encyclopedia," 2017. `https://en.wikipedia.org/wiki/Moment_of_inertia`.

[64] M. Nakada, B. Allen, S. Morishima, and D. Terzopoulos, "Learning arm motion strategies for balance recovery of humanoid robots," *2010 International Conference on Emerging Security Technologies*, vol. 0, pp. 165–170, 09 2010.

BibTeX Website citatations with the biblatex package

# Appendix A

# An Appendix

## A.1 Aldebaran NAO

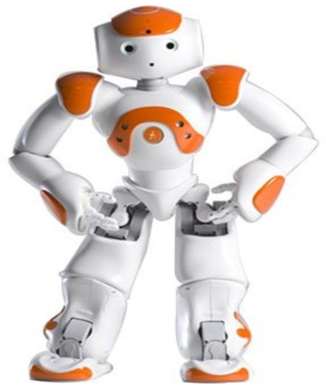


FIGURE A.1: NAO Robot By Aldebaran

The NAO is one of the famous humanoid robots available for research and development. The French company Aldebaran introduced it in 2004. There are four models of the NAO that differs in structure, such as the H25, H21, T21 and T25 – See figures A.2, A.3, A.4 and A.5. The H model is a full body humanoid, while the T model is made only of the upper body parts. The H25 has 25 degrees of freedom, while the H21 has just 21 DOF and there are different versions of each model. In this study, we have used the newest available version of the H25 model which is Version 5 (V5). It weighs 4.3 kg, and its height is 58 cm.

The NAO has two cameras located between the eyes, four microphones at the head and two loud speaks at the ears. The NAO is equipped with different sensors allowing it to receive feedback from the surrounding environment. There are four Force sensitive resistors (FSR) at the bottom of each foot that measures the resistance change in correspondence with the applied pressure. The robot is also equipped with two sonar emitters and receivers located around the middle of its torso. Additionally, there is an inertial measurement unit (IMU) in the middle of the trunk, that consists of a three-axis Gyroscope and a 3-axis accelerometer. Additionally, three capacitive sensors are added to the head and each hand to sense any touches. Each foot of the H25 model has two bumpers at the front, which act like an on/off switch to detect collisions. Figure A6 shows a complete hardware specification summary for the H25 model. The robot is operated using a 1.6 GHz Atom CPU running a Linux based operating system with a total of 9 GB memory. Its battery allows it to work for 60 to 90 min, depending on the tasks being performed. All the mentioned hardware specifications make the NAO very desirable for different research.

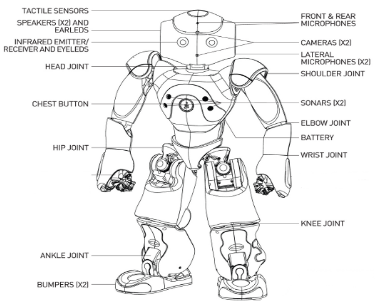


Figure A.2: Aldebaran NAO H25

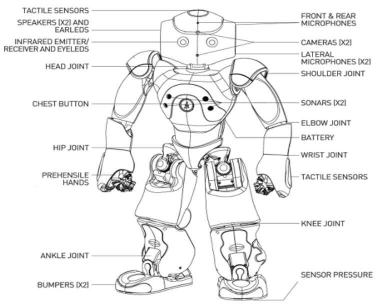


Figure A.3: Aldebaran NAO H21

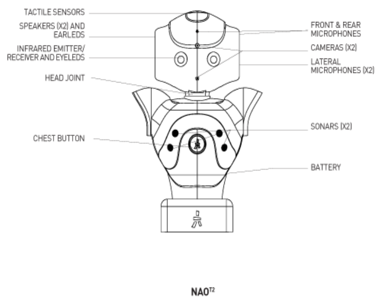


Figure A.4: Aldebaran NAO T21

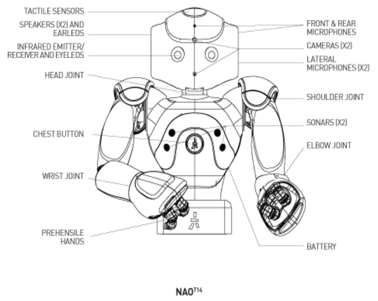
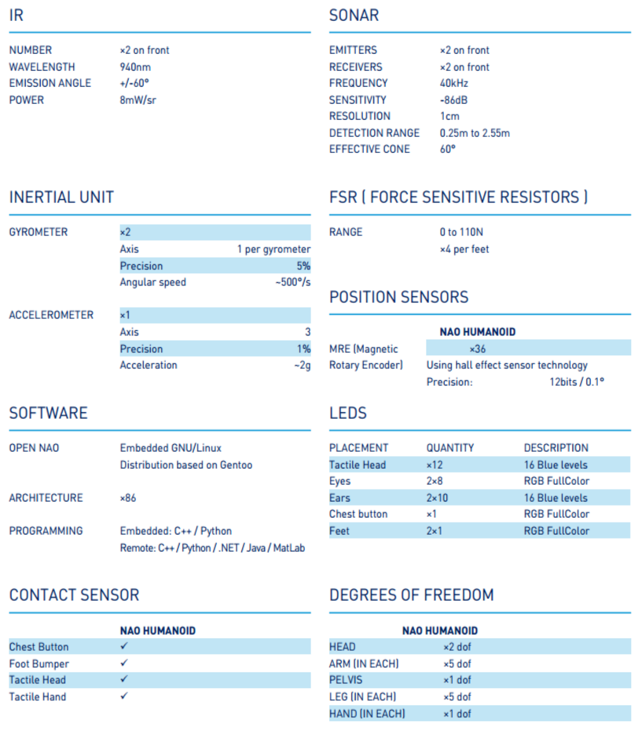


Figure A.5: Aldebaran NAO T25



IR

| | |
|---|---|
| NUMBER | ×2 on front |
| WAVELENGTH | 940nm |
| EMISSION ANGLE | +/-60° |
| POWER | 8mW/sr |

SONAR

| | |
|---|---|
| EMITTERS | ×2 on front |
| RECEIVERS | ×2 on front |
| FREQUENCY | 40kHz |
| SENSITIVITY | -86dB |
| RESOLUTION | 1cm |
| DETECTION RANGE | 0.25m to 2.55m |
| EFFECTIVE CONE | 60° |

INERTIAL UNIT

| | | |
|---|---|---|
| GYROMETER | ×2 | |
| | Axis | 1 per gyrometer |
| | Precision | 5% |
| | Angular speed | ~500°/s |
| ACCELEROMETER | ×1 | |
| | Axis | 3 |
| | Precision | 1% |
| | Acceleration | ~2g |

FSR ( FORCE SENSITIVE RESISTORS )

| | |
|---|---|
| RANGE | 0 to 110N |
| | ×4 per feet |

POSITION SENSORS

| | NAO HUMANOID | |
|---|---|---|
| MRE (Magnetic Rotary Encoder) | ×36 | |
| | Using hall effect sensor technology | |
| | Precision: | 12bits / 0.1° |

SOFTWARE

| | |
|---|---|
| OPEN NAO | Embedded GNU/Linux Distribution based on Gentoo |
| ARCHITECTURE | ×86 |
| PROGRAMMING | Embedded: C++ / Python |
| | Remote: C++ / Python / .NET / Java / MatLab |

LEDS

| PLACEMENT | QUANTITY | DESCRIPTION |
|---|---|---|
| Tactile Head | ×12 | 16 Blue levels |
| Eyes | 2×8 | RGB FullColor |
| Ears | 2×10 | 16 Blue levels |
| Chest button | ×1 | RGB FullColor |
| Feet | 2×1 | RGB FullColor |

CONTACT SENSOR

| | NAO HUMANOID |
|---|---|
| Chest Button | ✓ |
| Foot Bumper | ✓ |
| Tactile Head | ✓ |
| Tactile Hand | ✓ |

DEGREES OF FREEDOM

| | NAO HUMANOID |
|---|---|
| HEAD | ×2 dof |
| ARM (IN EACH) | ×5 dof |
| PELVIS | ×1 dof |
| LEG (IN EACH) | ×5 dof |
| HAND (IN EACH) | ×1 dof |

FIGURE A.6: Aldebaran NAO Hardware Specifications

## A.2 Programming the NAO

In this section, we provide an introduction to programming the NAO robot. The NAO platform can be programmed in several ways depending on the desired application and the user programming skills. There is a software suite available for beginner programmers who prefer not to write code. Additionally, the robot can be coded to perform complex applications using the NAOqi framework. On the same hand, there are a handful of incredible simulators available, in case the robot is unavailable or to test unexpected behaviors.

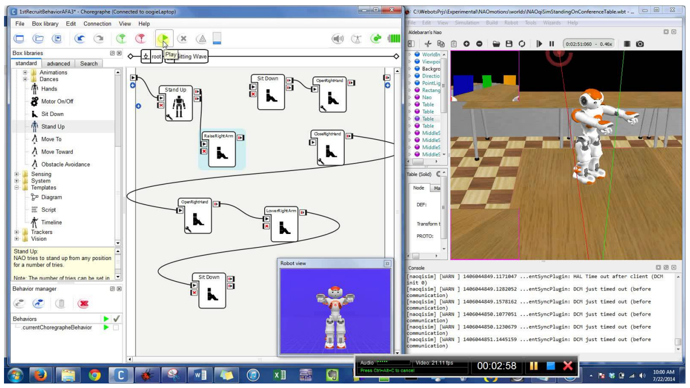
### A.2.1 Naoqi

NAOqi is the primary software running on the NAO robot. The NAOqi Framework is the programming skeleton used to program the NAO. The framework provides pragmatic functionalists as it answers common robotic needs such as parallelism, resources, synchronization, and events. NAOqi is cross-platform, which means that it can be used on Windows, Linux and Mac OS. In addition, NAOqi is a cross-language that allows creating software that runs on the robot using C++ and python. The framework allows creating distributed programs that run remotely from a computer or locally on the robot. It also allows homogeneous communication between different modules like motion, audio, and Video as well as similar programming and information sharing. Naoqi comes with a list of core modules as a part of the framework that provides all the application programming interfaces (API) needed to program the NAO. There are software development kits (SDK) available in different programming languages, which includes Python, C++, Java, MATLAB and .Net. The SDK contains a set of APIs and programming tools that help to program the robot to do various tasks. However, Only C++ and Python can be used to run code on the robot and the other language can be used only for testing. Python is the most straightforward programing language to start with; because it does not require compiling the code on a computer before sending the executable to the robot as in the case of C++.

There are two methods to program the NAO with coding. The first one is by using the NAOqi framework's APIs to call any of the available modules and organize it in the desired structure. This method requires common programming knowledge. The second way is to write all the code entirely from scratch, by creating individual libraries and modules as well as a hardware abstraction layer (HAL) that communicate with NAOqi. The later method features many benefits, such as a complete knowledge and control of the modules in contrast with the first approach, in which the modules are just a black boxes.

### A.2.2 Chrographe

Choregraphe software is a part of the Aldebaran's suit software. It is a drag and drops GUI, which allows using a sequence of preprogrammed modules to create animations, behaviors, and dialogs - figure A.7. These modules can be arranged in any combinational or sequential structure as desired and can be edited in Python. Choregraphe can be connected to the actual robot and simulators, using a broker with a network IP and Port, for testing. This software is an excellent tool for beginner programmers as it allows creating applications with dialogues, services, and different behaviors without having to write a single line of code.

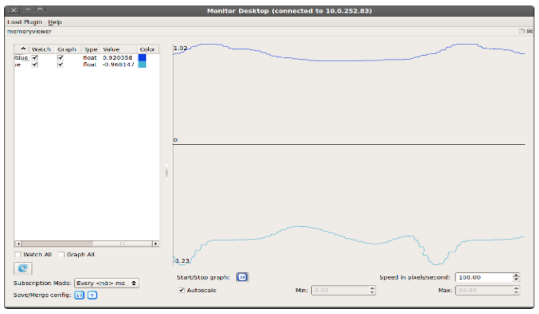


```
[naoqisim] [WARN ] 1406044849.1171047 ...entSyncPlugin: HAL Time out after client (DCM
init 0)
[naoqisim] [WARN ] 1406044849.1282052 ...entSyncPlugin: DCM just timed out (before
communication)
[naoqisim] [WARN ] 1406044849.1578162 ...entSyncPlugin: DCM just timed out (before
communication)
[naoqisim] [WARN ] 1406044850.1077051 ...entSyncPlugin: DCM just timed out (before
communication)
[naoqisim] [WARN ] 1406044850.1230679 ...entSyncPlugin: DCM just timed out (before
communication)
[naoqisim] [WARN ] 1406044851.1445159 ...entSyncPlugin: DCM just timed out (before
communication)
```

FIGURE A.7: Choregraphe By Aldebaran Software Suite

### A.2.3 Monitor

Monitor is also a part of the Aldebaran's suit software allowing the user to visualize the robot behavior in real time. Like Choregraphe it connects to the actual robot or simulator using the broker IP and port. As shown in figure A.8, Monitor provides an elementary feedback from the robot, allowing the user to read sensors values, view camera and read memory while executing a behavior. It is an excellent tool for debugging programs in real time.



Monitor Desktop (connected to 10.0.252.83)

| | Watch | Graph | Type | Value | Color |
|---|---|---|---|---|---|
| Glue | ✓ | ✓ | float | 0.920358 | 🟦 |
| ue | ✓ | ✓ | float | -0.968147 | 🟦 |

FIGURE A.8: Monitor By Aldebaran Software Suite

### A.2.4 Simulator SDK

The Aldebaran Robotics company provides a simulator SDK package, allowing users to simulate the any of the Aldebaran robots in a 3D simulator. Even though NAOqi executable can be bridged with a simulator, the sensors and currents values will return null since there is no actual hardware running. However, Using the simulator SDK package developers can use all NAOqi APIs program the simulated robot as well as get all sensors and currents readings accurately. This SDK package is available to download from the Aldebaran's website. A plug-in is required to use the SDK package in any 3D simulator so that the simulator knows which libraries to use and where to find it. Figure A.9 explains the simulator SDK working mechanism.

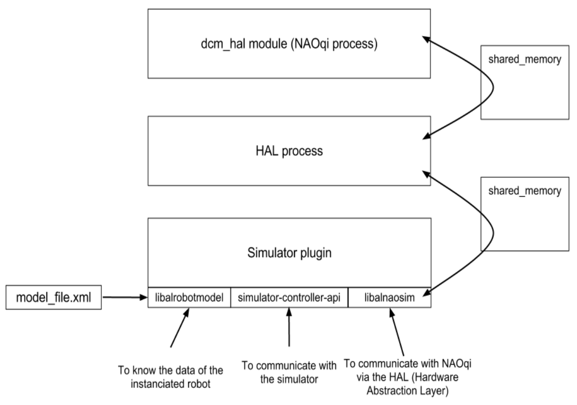

dcm_hal module (NAOqi process)
shared_memory
HAL process
shared_memory
Simulator plugin
model_file.xml
libalrobotmodel
simulator-controller-api
libalnaosim
To know the data of the instanciated robot
To communicate with the simulator
To communicate with NAOqi via the HAL (Hardware Abstraction Layer)

FIGURE A.9: simulator SDK NAOqi interface