Implementation of a Stair Walking Algorithm on the REEM-C Humanoid Robot

by

Kareem Mostafa

A thesis
presented to the University of Waterloo
in fulfillment of the
thesis requirement for the degree of
Master of Applied Science
in
Mechanical and Mechatronics Engineering

Waterloo, Ontario, Canada, 2022

© Kareem Mostafa 2022

Author's Declaration

I hereby declare that I am the sole author of this thesis. This is a true copy of the thesis, including any required final revisions, as accepted by my examiners.

I understand that my thesis may be made electronically available to the public.

Abstract

The recent increased usage of robots in society brings about unique challenges, such as endowing these robots with the ability to operate in a manner similar to humans. These challenges are well structured for humanoid robots, as they provide the complexity that may allow them to replicate human behaviours. Locomotion is a key challenge for humanoid robots, as being able to move from one place to another is crucial in being able to operate in ways that humans do on a daily basis. Complex environments that humans regularly operate in, such as industrial workspaces, may contain uneven terrain or stairs that need to be climbed to operate in hard-to-reach areas or perform tasks at varying heights. Humanoid robots will naturally need the ability perform such tasks if they are meant to work alongside humans in these challenging workspaces. For instance, giving HRP-4 the ability to enter a plane via its stairs [4] is a prime example of challenging human-like locomotion applied to a humanoid robot.

The present work is focused on performing multi-level maneuverability, which is performed on a set of stairs that were assembled in a static environment, and implements an open loop control system in which the planned motion is generated and then tested on the robot. We are targeting the performance of stair walking on the REEM-C Humanoid Robot, which would provide a better understanding of the humanoid's capabilities and accessibility to hard-to-reach locations by a human subject. The evaluation of the humanoid stability would be through its repeatability and consistency when executing the task. The planned motions of the humanoid robot were executed on simulation for varying stair heights, which were then verified by execution on the real robot. The main goal of this study was to improve the planning time of motion generation, in an attempt to integrate it into a real-time controller scenario. Optimizing the C++ code improved the performance, in contrast to using Python libraries, providing us with an improvement of 70% in planning time. The motions were performed on the robot for varying stair heights; 2, 4, 6, and 8 cm with the time of step being 25 seconds, which is faster than the previous implementation of 60 seconds per step at the University of Heidelberg[32]. Although the improvement in computations was successful, implementing it within a real time controller was not achieved yet.

In addition, an update to the mass, center of mass and moment of inertia for each link in the robot model was performed using the combined information from PAL Robotics [24] and Felix Aller from the University of Heidelberg[2], to accurately reflect real-life robot behaviour and to lessen the gap between simulation and real-life. Furthermore, to gather additional information to evaluate the experiments performed by the robot, an implementation of torque computation using the motor current measurements and inverse

dynamics was used to determine if the experiments were performed safely by comparing the computations to the joint torque limits.

Acknowledgements

I would like to thank Katja Mombaur for her support and mentorship throughout my MASc degree as well as guidance in research on humanoid robotics. Her passion and excitement for research and always countering the most exciting ideas has been an inspiration and drive for me to improve my skills in the humanoids robot field.

I would like to thank my readers, William Melek and John McPhee, for providing their valuable feedback on this thesis and their support.

I would like to thank Francisco Andrade Chavez, for his support during my research whenever I got stuck on an issue. I look forward to more collaborations together.

I would like to thank Jonathan Lin, for providing moral support as well as guidance in my MASc journey and his help in providing insights on motion capture capabilities within my MASc courses.

Thank you Anas, Jan, Menna, Pranav, Will and Christian for many of the course collaborations, help with my experiments and the memories that we had together. It has been a great time working with you and learning about your projects.Looking forward to more interactions in the future.

I would like to thank Felix Aller and Sebastian Sontag from University of Heidelberg for providing their results and algorithms for the REEM-C.

I would like to thank my friends for all the help and support, the journey would not have been easy without getting to know you, hope that we get to have more interactions together and build up more memories.

I would also like to thank the Human-centered Robotics and Machine Intelligence lab of Katja Mombaur. It was a pleasure to learn and bond with everyone in the group, looking forward to more interactions and lab wide events to enjoy together.

Finally, I would like to thank my family for all of their support, it wasn't easy doing it without them. Omar and Abdulrahman, my brothers; Lamiaa, my mother; and Ahmed, my father. Thank you for pushing me to becoming my best and for making all of this possible.

Table of Contents

List of Figures ix

List of Tables xiii

1 Introduction 1

1.1 Thesis Scope . . . 2
1.2 Thesis Contribution . . . 2
1.3 Thesis Organization . . . 3
1.4 REEM-C . . . 3
1.4.1 Robot Specification . . . 3
1.4.2 Kinematics . . . 5
1.4.3 Dynamics . . . 9
1.4.4 Motors . . . 9
1.4.5 Sensors . . . 10

2 Literature Review 11

2.1 Robotic Basics . . . 11
2.1.1 Kinematics . . . 11
2.2 Humanoid Robots Motion Planning and Control . . . 14
2.2.1 Important Definitions . . . 14
2.2.2 Walking . . . 16

2.2.3 Uneven Terrain Walking . . . 18
2.3 Related Work . . . 19
2.3.1 iCub walking implementation . . . 19
2.3.2 Stair Climbing using Stereo Vision . . . 21
2.3.3 Optimal Control for Whole-Body Motion Generation for multi-contact configurations . . . 23
2.3.4 REEM-C University of Heidelberg . . . 25
2.4 Summary . . . 27

3 Model Update **29**
3.1 Torque Computation . . . 31
3.1.1 Motor Current Readings . . . 32
3.1.2 Inverse Dynamics . . . 33
3.2 Results . . . 35
3.2.1 Motor Current Reading . . . 35
3.2.2 Inverse Dynamics . . . 37
3.3 Conclusion . . . 38

4 Stair Walking Implementation **40**
4.1 Workspace Design . . . 40
4.1.1 Design Approach . . . 41
4.2 Implementation . . . 43
4.2.1 Pattern Generation . . . 44
4.2.2 Inverse Kinematics . . . 47
4.2.3 Testing . . . 47
4.3 Analysis . . . 48
4.3.1 Static Analysis . . . 48
4.3.2 Kinematic Check . . . 56

4.4 Simulation 61
4.4.1 2 cm Stairs 62
4.4.2 4 cm Stairs 66
4.4.3 6 cm Stairs 69
4.4.4 8 cm Stairs 72
4.4.5 10 cm Stairs 75
4.4.6 Summary 77
4.5 Real-life Experiments 79
4.5.1 2 cm Stairs 79
4.5.2 4 cm Stairs 82
4.5.3 6 cm Stairs 87
4.5.4 8 cm Stairs 90
4.5.5 10 cm Stairs 93
4.5.6 Summary 95
4.5.7 Visualizer Implementations 97

5 Conclusion **99**
5.1 Limitations 100
5.2 Future Work 101

References **102**

List of Figures

1.1 REEM-C. 4
1.2 Upper-body kinematics. 6
1.3 Lower-body kinematics. 7
1.4 Left leg joints (I) 7
1.5 Left leg joints (II) 8
1.6 Left leg joints (III) 8
1.7 Left leg joints (IV) 8

2.1 A chain in a robotic manipulator with revolute joints using D-H parameters.[30, ch. 1.4] 13
2.2 Inverse kinematics representation of a 2 DoF planar chain.[20, ch. 6] 13
2.3 Ground reaction forces and their equivalent force and moment.[13] 15
2.4 3D LIPM, with the CoM on the constraint horizontal plane.[13, ch. 4.3.1] . 16
2.5 Basic control framework in this book: (a)In an ideal condition, a biped robot can walk by following a walking pattern. (b)In a real environment, a biped robot needs a stabilizer. [13] 17
2.6 Cart model of mass M, sliding over a mass-less table.[13] 18
2.7 The CoM is controlled to move on the constraint line (broken line) by the kick force f, while the ankle rotates freely. 19
2.8 Pattern generation workflow. [12] 20
2.9 Stair Recognition workflow. [9] 22

2.10 HRP-2 on the stairs using support from the handrail. The overlay shows the CoM at the pelvis and the contact forces at hand and feet. 23
2.11 Joint trajectories overview with **c** being the COM trajectory, **RF,LF,RH,LH** are the end-effector trajectories and **q** for the generation process. 25
2.12 REEM-C performing the stair walking on the stairs with the help of the lift 26
2.13 Pattern generation pipeline using Python 27

3.1 Output of effort topic. 32
3.2 Filtering effort data 33
3.3 Inverse dynamics 33
3.4 RAW motor readings while standing still. 36
3.5 Filtered motor reading while standing still. 36

4.1 Initial stair top/side view 41
4.2 Stairs in real life 42
4.3 Work flow pipeline 43
4.4 Footholds and CoM trajectory for stair walking. 44
4.5 Pattern generation pipeline 45
4.6 Foot trajectory in z component. 46
4.7 Foot trajectory in x axis. 46
4.8 Foot trajectory in 2D space (sideview). 47
4.9 Shelf dimension 48
4.10 Humanoid robot configuration for the static torque analysis, right before lift-off 49
4.11 Robot model simplified in order to compute the unilateral constraints on the ankle. 54
4.12 Torques required by the ankle joints in the stepping configuration to prevent the lift-off. 55
4.13 Robot Kinematics with one foot on the stair and one on the ground, ankle joint (q1) and knee joint (q2). 56

4.14 The shank CoM position so that the stair would not collide with the stairs. 57
4.15 Robot Kinematics for walking down the stairs, with one foot on the high step and one on the lower step of the stairs. 59
4.16 Robot Kinematics visualization of the angle values for walking down the stairs, with one foot on the high step and one on the lower step of the stairs. 61
4.17 2 cm motion on stairs in simulation. 62
4.18 Joint angles[rad] vs Time[sec] for 2 cm left leg desired vs actual trajectories on simulation. 64
4.19 Joint angles[rad] vs Time[sec] for 2 cm right leg desired vs actual trajectories on simulation. 65
4.20 4 cm motion on stairs in simulation. 66
4.21 Joint angles[rad] vs Time[sec] for 4 cm left leg desired vs actual trajectories on simulation. 67
4.22 Joint angles[rad] vs Time[sec] for 4 cm right leg desired vs actual trajectories on simulation. 68
4.23 6 cm motion on stairs in simulation. 69
4.24 Joint angles[rad] vs Time[sec] for 6 cm left leg desired vs actual trajectories on simulation. 70
4.25 Joint angles[rad] vs Time[sec] for 6 cm right leg desired vs actual trajectories on simulation. 71
4.26 8 cm motion on stairs in simulation. 72
4.27 Joint angles[rad] vs Time[sec] for 8 cm left leg desired vs actual trajectories on simulation. 73
4.28 Joint angles[rad] vs Time[sec] for 8 cm right leg desired vs actual trajectories on simulation. 74
4.29 10 cm motion on stairs with no lip. 75
4.30 Updated stairs model with lip. 76
4.31 10 cm motion on stairs with lip and updated model 76
4.32 Stairs 2 cm. 80
4.33 Steps of 2 cm stair walking. 80

4.34 Torques produced by the left foot throughout the motion on the 2 cm stairs. 81
4.35 Torques produced by the right foot throughout the motion on the 2 cm stairs. 82
4.36 Stairs 4 cm. 83
4.37 Steps of 4 cm stair walking. 84
4.38 Torques produced by the left foot throughout the motion on the 4 cm stairs. 85
4.39 Torques produced by the right foot throughout the motion on the 4 cm stairs. 86
4.40 Stairs 6 cm. 87
4.41 Foot tilt during liftoff. 88
4.42 Steps of 6 cm stair walking. 88
4.43 Torques produced by the left foot throughout the motion on the 6 cm stairs. 89
4.44 Torques produced by the right foot throughout the motion on the 6 cm stairs. 90
4.45 Steps of 8 cm stair walking. 91
4.46 Torques produced by the left foot throughout the motion on the 8 cm stairs. 92
4.47 Torques produced by the right foot throughout the motion on the 8 cm stairs. 93
4.48 Shank too close to the step. 94
4.49 The foot being stuck in the lip. 94
4.50 RVIZ visualization of the recorded motion from the robot during the experiment. 98

List of Tables

1.1 Degrees of Freedom. 5
1.2 Joint Torques at full charged battery. 9
1.3 Motor Torque constants. 10
1.4 Force Torque sensors in the ankles. 10

3.1 Total Mass of different models. 29
3.2 Link properties. 31
3.3 Torques from Motor current measurements in default configuration. 37
3.4 Inverse dynamic results in default configuration. 37

4.1 Initial stairs specifications. 41
4.2 Static test analysis for the 2 cm stairs configuration 50
4.3 Static test analysis for the 4 cm stairs configuration 50
4.4 Static test analysis for the 6 cm stairs configuration 51
4.5 Static test analysis for the 8 cm stairs configuration 51
4.6 Static test analysis for the 10 cm stairs configuration 52
4.7 Static test analysis for the 12 cm stairs configuration 52
4.8 Static test analysis for the 14 cm stairs configuration 53
4.9 Maximum configuration for the rear foot. 58
4.10 Maximum configuration for the front foot. 58
4.11 Maximum configuration for the rear foot. 60

4.12 Maximum configuration for the front foot. 60
4.13 Parameters for 2 cm Stairs in simulation. 63
4.14 Parameters for 4 cm stairs in simulation. 66
4.15 Parameters for 6 cm stairs in simulation. 69
4.16 Parameters for 8 cm stairs in simulation. 72
4.17 Maximum torques produced by the REEM-C in simulation during different stair walking heights 78
4.18 Parameters for 2 cm stairs. 79
4.19 Parameters for 4 cm stairs. 83
4.20 Parameters for 6 cm stairs. 87
4.21 Parameters for 8 cm stairs. 91
4.22 Maximum Torques produced by the REEM-C in real-life during different stair walking heights. 96
4.23 Implemented stair height in real-life. 97
4.24 Success rate of the experiments over 5 repetitions. 97

Chapter 1

Introduction

In the simplest manner of explanation, we may say that we are surrounded by robots on a daily basis. According to the dictionary, a robot is "A machine that can be programmed by a computer and is capable of carrying out a series of complex actions." The word "robot" originated from the Czech word **robota** meaning "forced labour." This implies that everything that performs some labour for an outcome is a robot. These robots have developed over the decades and entered new aspects of life to help humans in their daily tasks, enabling them to integrate with the environment depending on their structures, abilities, and limitations. In the attempt to familiarize the robots and make them human-friendly and relatable, the idea of humanoid robots emerged. Humanoid robots have a strong resemblance to normal human beings as most consist of 2 parts: an upper body with arms, torso and head, and a lower body with legs. To introduce humanoid robots to the public, a couple of projects were initiated by Honda, Toyota, and other research institutes to present the robot in a friendly manner; an example of these projects is the humanoid robot orchestra in which the robots would play the drums, trumpet, and violin in synchronization. By observing these humanoid robots, we can find them divided into two categories with regards to locomotion; legged or wheeled robots, each of which has a different manner of interacting with their environments.

As mentioned earlier, one form of locomotion for a humanoid robot is by utilizing their two legs; this bipedal motion enables these robots to maneuver any area that is accessible to humans. This can even apply to situations where tasks are dangerous or repetitive for humans, and thus require a robot alternative. As a result, robot locomotion is considered one of the most critical problems to be solved. These humanoid robots must perform locomotion efficiently and accurately to reach their next goal position in the workspace. This issue has been introduced and can be solved using the famous Kajita [13] approach,

which allows the robot to travel from one place to another using bent knees. Another problem that emerged when using these robots is the access to various levels of ground and the ability to maneuver over rugged terrains. This would enable the humanoids to have access to more complex workspaces and provide versatility, especially regarding hard-to-reach areas or dangerous surroundings for humans to be in. Naturally, endowing humanoids with the ability to walk up stairs to reach these areas becomes an important locomotion problem in the field of robotics.

Motivated by these causes, the DARPA robotics challenge was developed to exploit and encourage research institutes worldwide to develop creative and efficient solutions to counter certain situations that the humanoid robots would face in the industrial/actual environment[5]. The challenges range from manipulation to locomotion tasks, like navigating rough terrains such as stairs or uneven platforms.

1.1 Thesis Scope

The project's main goal was to enable humanoid robots to access different heights, such that they may operate on varying levels of ground with humans. This project tested the capability of the REEM-C Humanoid robot to perform the stair walking motion and to pinpoint its limitations, as well as what can be improved in further implementations.

The approach taken within this thesis was the implementation of an open loop control system, in which the motion would be generated through the known approach of Kajita [13], in which the footholds are used to generate the center of pressure and center of mass positions as conditions for the motion. Inverse kinematics is then applied to generate the configurations on the robot that satisfy the specified conditions.

1.2 Thesis Contribution

This thesis focuses on implementing the stair walking algorithm on the current version of REEM-C in University of Waterloo and exploiting the capabilities of the robot's motor limits. The main contribution is improving the computation time during pre-planning stage, through optimizing the algorithm in C++. This improvement aims to help implement an online pattern generator, that would compute the steps while the robot executes the action. The experiments are designed as follows, the humanoid robot's motion starts from an initial position in front of the stairs to the final position at the top, with all the points

being generated with reference to the initial position. Finally, the development of the motion, in addition to the torque calculation scheme, provides an evaluation of the robot's capabilities and highlights the areas of development needed to perform better motions.

1.3 Thesis Organization

Chapter 1.4 overviews the REEM-C humanoid robot used throughout this thesis. Chapter 2 provides background on robotics and motion generation, then presents the state-of-the-art implementations in stair walking for humanoids. Chapter 3 overviews the process of updating the robot model using partial information from two sources, then evaluates the torque calculation using both motor current readings and inverse dynamics on the REEM-C model. Chapter 4 displays the implementation of the Kajita [13] stair-walking algorithm as an open loop system and discusses the results for various stair configurations. Chapter 5 presents the conclusion and discusses future work.

1.4 REEM-C

REEM-C is a bipedal humanoid robot developed by PAL Robotics in Spain in 2013. PAL Robotics was founded in 2004 and has been developing humanoid biped and mobile robots over the years[24]. Their mission is to provide robots that interact with humans and support various aspects of their lives. PAL Robotics provides the robots and their integrated software to be used within industries and research institutes worldwide. One of the humanoid robots that were also provided to the University of Waterloo was TALOS, which differs from REEM-C as it is torque controlled rather than position controlled, and is larger in size and in torque limits.

In the upcoming sections, we will cover the specifications, the physical properties, and the actuators of the REEM-C humanoid robot. The highlight would be the legs of the robot, as they are the main focus of locomotion for humanoids.

1.4.1 Robot Specification

This section is based on the handbook provided for the robot [28]. The diagrams and the values provided in this section are taken from the handbook directly. This section provides an overview of the essential features used on the robot during this thesis.



Figure 1.1: REEM-C.

REEM-C is 1.65 m high and weighs around 80 kg (including the battery)[27]. It is slightly heavier than an average human being of that height, which could introduce some difficulties when performing some tasks and would need the use of a crane to ensure safety.

The humanoid can be seen in Fig1.1. REEM-C is supported with 68 degrees of freedom (DOF), allowing various movements to be implemented. The DOF are distributed around the whole body as seen in Table 1.1. Due to the large number of degrees of freedom, the robot has lots of potential for performing different tasks and would be suitable for use in both research and industry in a wide variety of applications.

Table 1.1: Degrees of Freedom.

Link Name	DoF
Head	2
Left Arm	7
Right Arm	7
Left Hand	19
Right Hand	19
Waist	2
Left Leg	6
Right Leg	6

1.4.2 Kinematics

The kinematic structure of the robot is shown in Fig1.2 and 1.3. Each frame is located at the joint origin, with each diagram having the tree structure begin at the **base link** of the robot, which is located at the pelvis where the centre of mass lies. The figures show all the joints as well as their rotation axes. The z-axis indicates the direction of the joint rotation. REEM-C's joints are all revolute with varying rotation directions.

For the intent of this thesis focused on locomotion, the lower body part which includes the legs, is of most importance. Fig 1.4, 1.5, 1.6 and 1.7 shows the position of each joint of the legs of the REEM-C. These joints will be the focus of the stair-walking motion, as will be explained in the upcoming chapter [28].

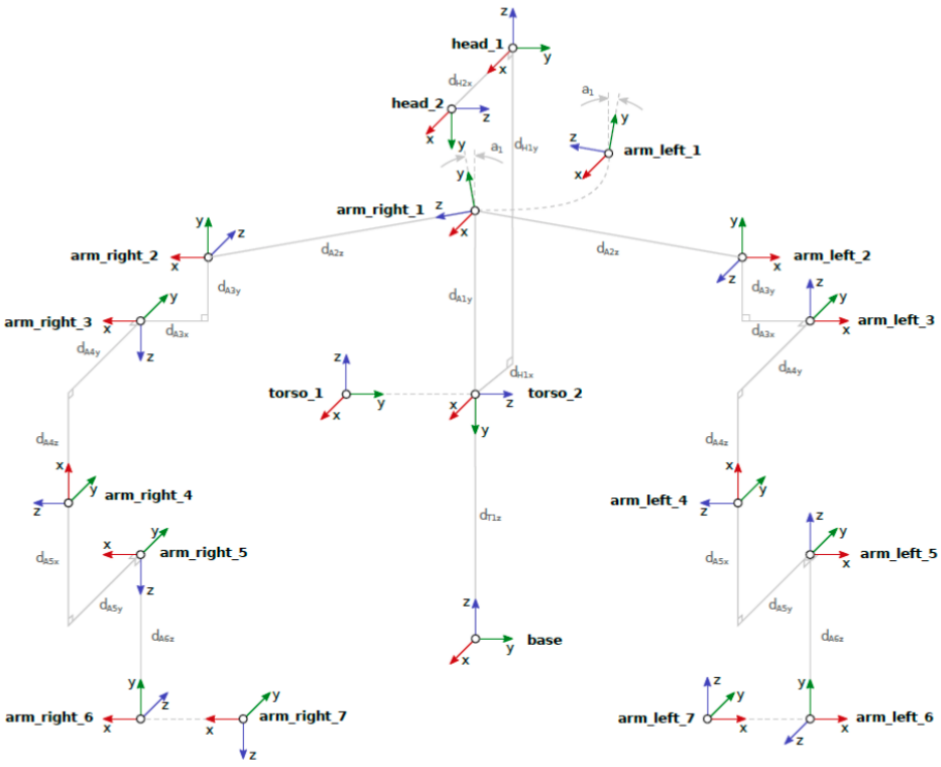


Figure 1.2: Upper-body kinematics.

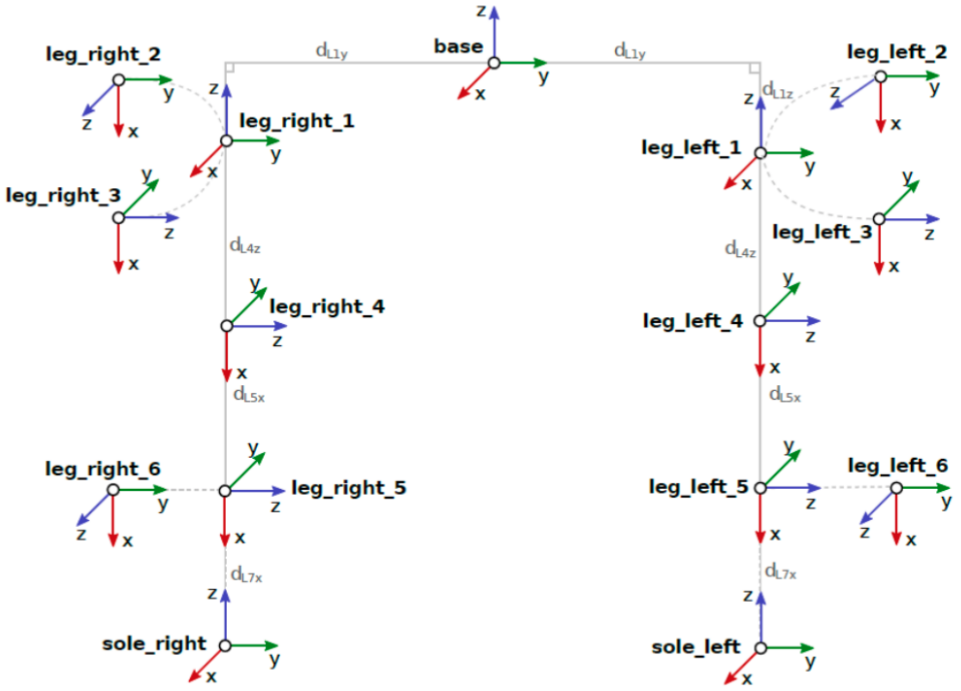


Figure 1.3: Lower-body kinematics.

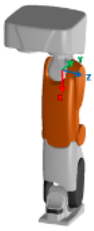


(a) base_link (b) leg_left_1_joint

Figure 1.4: Left leg joints (I)

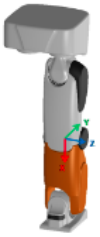


(c) leg_left_2_joint



(d) leg_left_3_joint

Figure 1.5: Left leg joints (II)



(e) leg_left_4_joint



(f) leg_left_5_joint

Figure 1.6: Left leg joints (III)



(g) leg_left_6_joint



(h) sole_left_joint

Figure 1.7: Left leg joints (IV)

1.4.3 Dynamics

This subsection provides a quick review of the robot's dynamics, specifically the output torques that can be provided at each joint for each degree of freedom. The motors in the leg of the robot are either driven by a harmonic gear drive or through a belt system to provide higher torques when needed for various motions. Table 1.2 shows the nominal and maximum torques that each motor can provide during its operation on a full battery.

Table 1.2: Joint Torques at full charged battery.

Joint Name	Nominal Torque[Nm]	Max Torque [Nm]
Leg_joint_1	50.7	127
Leg_joint_2	64	76
Leg_joint_3	62.9	127
Leg_joint_4	140.4	284
Leg_joint_5	91.7	147
Leg_joint_6	64	76

1.4.4 Motors

REEM-C is provided with a large number of motors. These motors are supplied with 48 V from the internal battery of the robot. The robot's degrees of freedom are powered by electrical motors, except for the fingers, which are under-actuated. Table1.3 provides the torque constants of each joint motor, which will be used and discussed in the upcoming chapters.

Table 1.3: Motor Torque constants.

Joint	Torque constants (mN.m/A)
leg_1_joint	185.3
leg_2_joint	149.9
leg_3_joint	254.6
leg_4_joint	154.8
leg_5_joint	254.6
leg_6_joint	149.9

1.4.5 Sensors

In addition, the robot contains several sensors which can be used to capture data about the external environment. These sensors include the IMU, camera and four six-axes force sensors that can be found in the ankle as well as the wrists. These provide us with a better understanding of the robot's state, its position in the workspace, and its velocity and acceleration. The IMU is placed on the hip of REEM-C in the same position as the base link. The specifications of the two force sensors at the ankle can be found in Table1.4.

Table 1.4: Force Torque sensors in the ankles.

Fx,Fy range	2800N
Fz range	6800 N
Tx, Ty range	120 Nm
Tz range	120 Nm
Fx.Fy resolution	3/4 N
Fz resolution	1 N
Tx, Ty resolution	1/50 Nm
Tz resolution	1/80 Nm

Chapter 2

Literature Review

This literature review will cover robotic manipulation, humanoid control and motion generation. This chapter overviews the concepts covered in the thesis in more detail, followed by a glimpse of the state of the art.

2.1 Robotic Basics

This section is dedicated to providing a fundamental background for robotics. The topics discussed would cover open-loop systems that do not include feedback to determine the upcoming actions. The discussion would involve revolute joints, as REEM-C's joints are revolute by design.

2.1.1 Kinematics

From Siciliano and Khatib [30, ch. 1.2], the definition of the position and orientation of a body in space is a common issue that needs to be addressed to have an understanding of where a certain body is located. In the case of robotics, determining the end-effector position is achieved by knowing the joint angles in the single chain of N DoF. This process is referred to as **forward kinematics**. The most common convention used in robotics is the Denavit-Hartenberg (D-H) parameter convention, which requires knowledge of the frames of the robot. The (D-H) parameter is an efficient convention as it only requires four parameters to locate the frame of each joint relative to the other.[17]

The parameters are defined as :

- The link length a_i
- The link twist α_i
- The joint offset d_i
- The joint angle θ_i

The homogeneous transformation matrix [30, ch. 1.4], T_i^{i-1} from frame i-1 to i can be represented as follows for a robot chain seen in Fig 2.1, where the R_i^{i-1} is the rotation matrix from frame i-1 to i and x_i^{i-1} is the translation vector from frame i-1 to i.

$$T_i^{i-1} = \begin{pmatrix} cos\theta_i & -sin\theta_i & 0 & a_i \\ sin\theta_i cos\alpha_i - & cos\theta_i cos\alpha_i & -sin\alpha_i & -sin\alpha_i d_i \\ sin\theta_i sin\alpha_i & cos\theta_i sin\alpha_i & cos\alpha_i & cos\alpha_i d_i \\ 0 & 0 & 0 & 1 \end{pmatrix} \tag{2.1}$$

$$= \begin{pmatrix} R_i^{i-1} & x_i^{i-1} \\ 0 & 1 \end{pmatrix} \tag{2.2}$$

To describe the position of the end-effector of a manipulator of N joints using forward kinematics, concatenate the transformations between frames by multiplying them sequentially as follows: [30, ch. 1.6]

$$^0T_N = {}^0T_1\, {}^1T_2\, {}^0T_3 ...\, {}^{N-1}T_N \tag{2.3}$$

On the other hand, another problem is computing the joint angles needed to reach a desired end-effector position. This problem is the opposite of the previous example and is referred to as **Inverse Kinematics** [20]. For instance, for a 2D planar open chain as shown in Fig 2.2, we take into account the desired end-effector position in addition to the joint lengths L_1, L_2 to compute the required angles for a specific configuration which would satisfy the end-effector position.

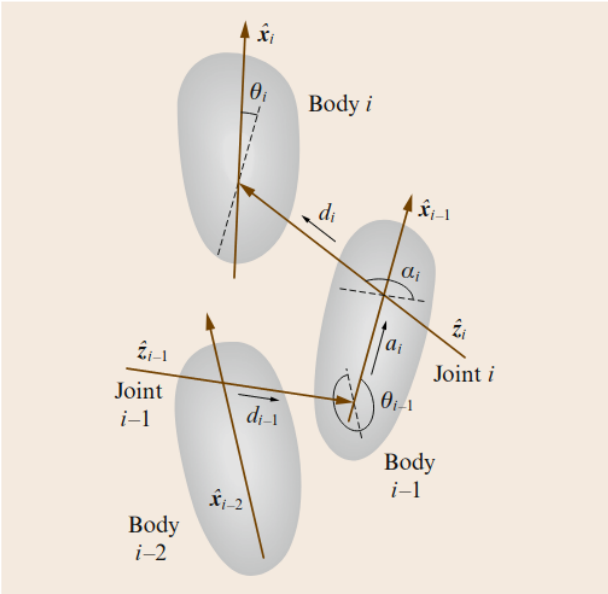
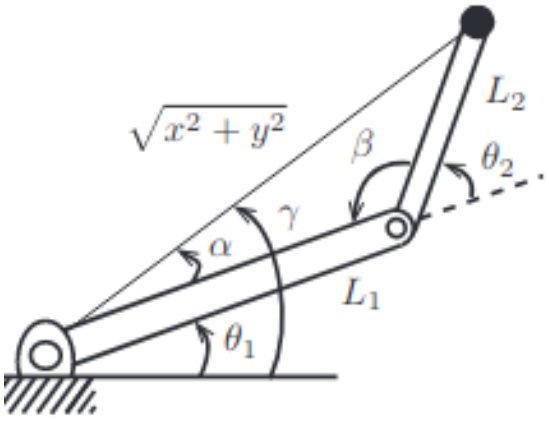


Figure 2.1: A chain in a robotic manipulator with revolute joints using D-H parameters.[30, ch. 1.4]



(b) Geometric solution.

Figure 2.2: Inverse kinematics representation of a 2 DoF planar chain.[20, ch. 6]

2.2 Humanoid Robots Motion Planning and Control

This section will provide the basic knowledge of humanoid robotics motion planning and control. The humanoid robots are in the form of tree kinematics in which the arms and the legs are all attached to the same base. These joints are often designed to be revolute so that all of the joints would behave similarly to a primary manipulator.

2.2.1 Important Definitions

During motion planning, some essential points must be considered to generate a reasonable and stable trajectory for the robot. These points are discussed in this subsection.

Center of Mass (COM)

The Center of Mass (COM) is a common point in humanoid robotics as it simplifies the humanoid model to a more interpretive and easily visualized state, and applies well to scenarios such as the linear inverted pendulum model.

For humanoid robots, it can be calculated as follows:

$$c = \sum_{i=1}^{N} \frac{m_i p_i}{M} \tag{2.4}$$

Where N is the number of masses, m_i is the mass of the i-th point, p_i is the position of each centre of mass, and M is the total mass of the robot [13, ch. 3].

Zero Moment Point (ZMP)

One more essential point for understanding humanoid robots is the zero-moment point (ZMP). ZMP is the point on the support polygon at which the moments of the ground reaction force sum to zero, Fig. 2.3. As prescribed by Kajita et al.[13, ch. 3], the ZMP is a point that always exists within the support polygon, which creates a convex hull depending on the contacts of the feet with the ground. The ZMP is equivalent to the COM projection on the ground, provided that the floor is not slippery.

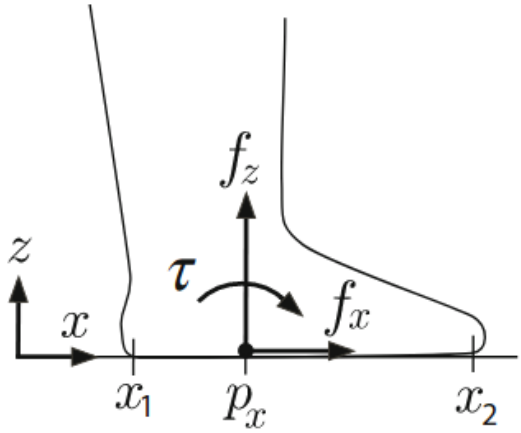


Figure 2.3: Ground reaction forces and their equivalent force and moment.[13]

The ZMP on a 2D plane is calculated as follows:

$$p_x = \frac{Mgx + p_z\dot{P}_x - \dot{L}_y}{Mg + \dot{P}_z} \tag{2.5}$$

$$p_y = \frac{Mgy + p_z\dot{P}_y - \dot{L}_x}{Mg + \dot{P}_z} \tag{2.6}$$

Where M is the mass, g is gravity, p_z is the height of the ground (zero for flat ground), $\dot{P}$ is the change in linear momentum and $\dot{L}$ is the change in angular momentum. Should the robot be standing still, both momentum terms are then zero, and the ZMP simplifies to:

$$p_x = x \tag{2.7}$$

$$p_y = y \tag{2.8}$$

which is equal to the center of pressure of the robot.

2.2.2 Walking

Walking is an essential method of travelling from one place to another to perform any task. Implementing such motion can be achieved using a famous method developed by Kajita et al.[13], which generates a walking pattern based on ZMP and the preview controller, which is mainly referred to as Kajita walking. Since the humanoid robot is moving in 3D space, we can model it as 3D LIPM, which can be seen in Fig. 2.4. The governing equations for the COM position in the horizontal plane (XY) can be written as:

$$\ddot{x} = \frac{g}{z_c} x \tag{2.9}$$

$$\ddot{y} = \frac{g}{z_c} y \tag{2.10}$$

where the z_c is the height of the COM in the horizontal plane and g is the gravitational acceleration.

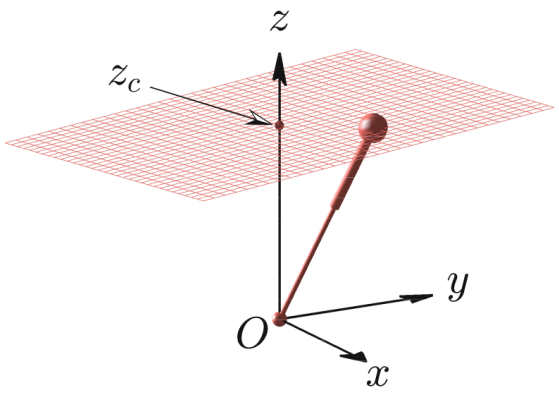


Figure 2.4: 3D LIPM, with the CoM on the constraint horizontal plane.[13, ch. 4.3.1]

The pattern generator is based on the cart-table model shown in Fig.2.6. This helps in calculating ZMP in 3D space for a simplified model as :

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

$$p_y = y - \frac{z_c}{g} \ddot{y} \tag{2.12}$$

The ZMP calculations can be driven from the centre of mass trajectory; however, in the case of the cart model, the pattern of footholds is generated from the ZMP to ensure that the ZMP satisfies the planned stepping pattern.

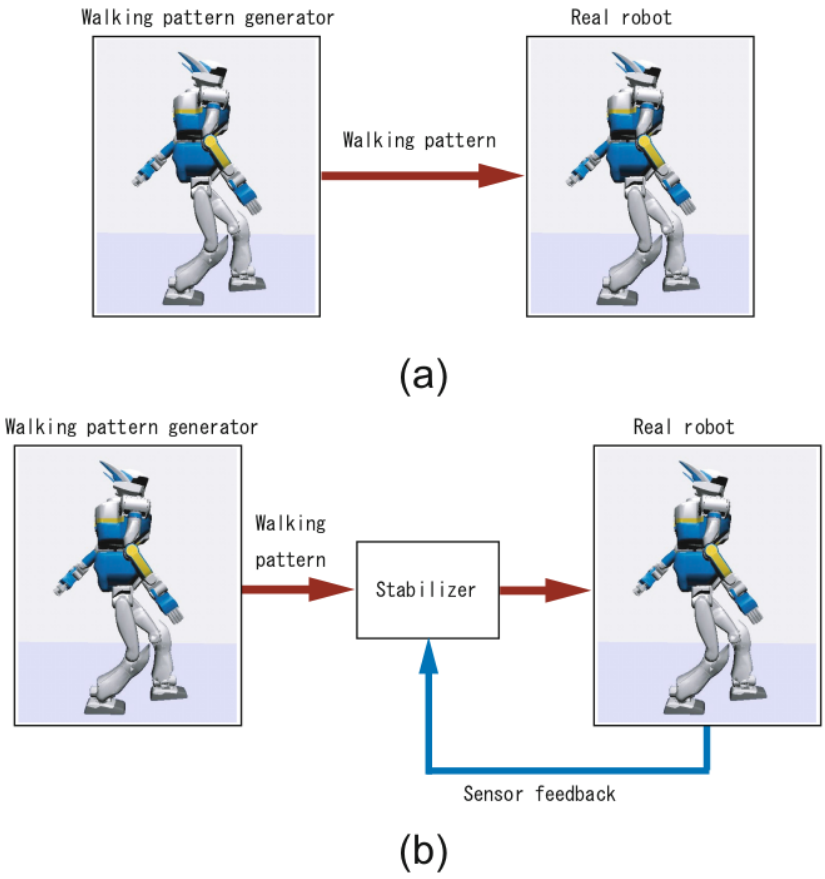


Figure 2.5: Basic control framework in this book: (a)In an ideal condition, a biped robot can walk by following a walking pattern. (b)In a real environment, a biped robot needs a stabilizer. [13]

As seen in Fig. 2.5, there are two scenarios when implementing such motion on the robot. One scenario assumes ideal conditions without vibrations or disturbances, while the other scenario is a natural environment where the robot would need an implementation of a stabilizer to reduce any sudden or unexpected disturbances. In our implementation for

the motion, we assume a static walking motion; therefore, the projection of the centre of mass would never leave the support polygon during the motion [13, ch. 4].

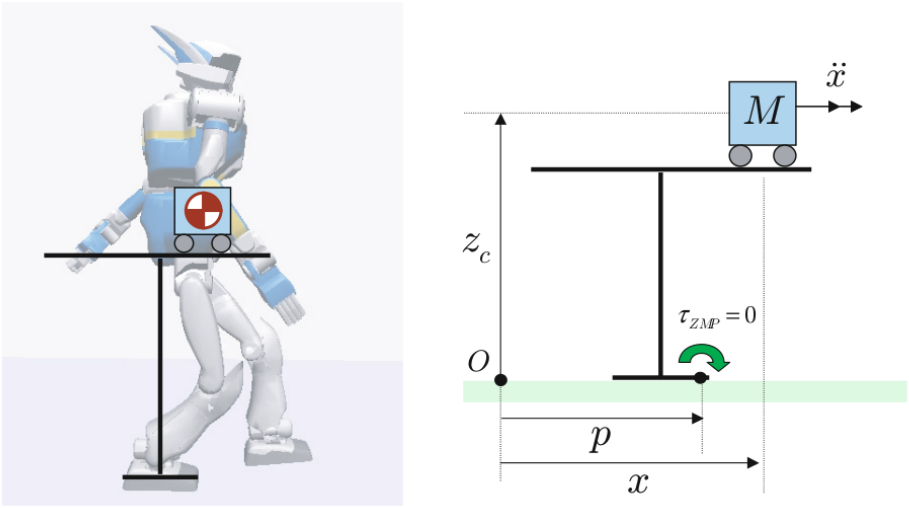


Figure 2.6: Cart model of mass M, sliding over a mass-less table.[13]

An essential feature of this algorithm is allowing for the robot to have bent knees when performing any walking motion to add stability to the robot and reduce the perturbations that can cause the robot to fall.

2.2.3 Uneven Terrain Walking

An extension can occur on the previous method described above. Going back to the inverted pendulum model [14], the COM Z position was a constraint to a certain height. In this approach, we consider the COM to be on a sloped line as shown in Fig.2.7, which is described by:

$$z = kx + z_c \tag{2.13}$$

where k is the slope of the line and z_c is the z intersection of the line. The line that the COM follows is called the *constraint line*.

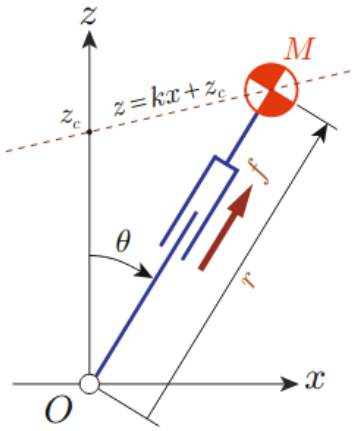


Figure 2.7: The CoM is controlled to move on the constraint line (broken line) by the kick force f, while the ankle rotates freely.

2.3 Related Work

This section discusses some of the implementations done in the field of humanoid robots performing stair walking. As discussed previously, performing stair walking is a vital locomotion problem that needs to be solved, as it introduces the ability to tackle various terrains and heights, enabling access to difficult locations. Various implementations have been done on different robots according to their capabilities. Some of the related works on stair walking of robots are discussed in more detail in this section.

2.3.1 iCub walking implementation

Yue Hu et al.[12] have developed various software modules through the ZMP-based pattern generator utilizing the position control of the iCub, to identify the robot's limitations against various walking scenarios. For simplicity, the experiments were done on a reduced version of the iCub, which was missing the arms and head.

The algorithm workflow can be seen in Fig.2.8. The algorithm states the following; the pattern generation is implemented using MATLAB [21], which takes the walking parameters as an input and outputs the respective footholds. Afterwards, the footholds generate the desired ZMP_d and the desired height COM_z.

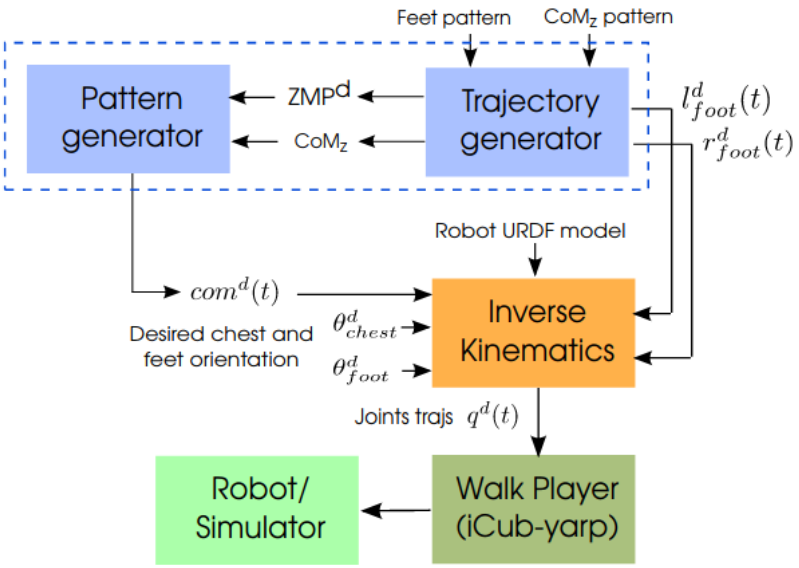


Figure 2.8: Pattern generation workflow. [12]

The ZMP Equation 2.11 is implemented here and can be discretized in the form of $\mathbf{p} = \mathbf{Ax}$:

$$\begin{bmatrix} p_1 \\ p_2 \\ : \\ p_{N-1} \\ p_N \end{bmatrix} = \begin{bmatrix} a_1 + b_1 & c_1 & 0 & & \\ a_2 & b_2 & c_2 & & \\ & : & : & : & \\ & a_N - 1 & b_N - 1 & c_N - 1 & \\ & 0 & a_N & b_N + c_N & \end{bmatrix} \begin{bmatrix} x_1 \\ x_2 \\ : \\ x_{N-1} \\ x_N \end{bmatrix} \tag{2.14}$$

In which the terms of the matrix are defined as follows:

$$a_i = -\frac{z}{(\ddot{z} + g)\Delta t^2} \tag{2.15}$$

$$b_i = -\frac{2z}{(\ddot{z} + g)\Delta t^2} + 1 \tag{2.16}$$

$$c_i = a_i \tag{2.17}$$

The ZMP_d and COM_z generate the entire COM trajectory COM_d. All these data are used as input to the inverse kinematics algorithm implemented in C++ using Rigid Body Dynamics Library (RBDL) to get the desired joint trajectories.

$$e_i(q) = \begin{cases} {}^d p_i - p_i(q) \\ R_i^T(q).a({}^d R_i R_i^T(q)) \end{cases} \tag{2.18}$$

Where **e** is the residuals of error in orientation and position for all end-effectors, ${}^d p_i$ and p_i are the desired and current positions of the joints i, while $a({}^d R_i R_i^T(q))$ calculates the angular velocities based on the current rotation matrix and the desired rotation matrix. The inverse kinematic solver solves the non-linear optimization problem by using the following formulation:

$$\min_q \|e(q)\|_2^2 \tag{2.19}$$

The outcome from the inverse kinematics is then provided to the "Walk" player, which commands the robot in either simulation or real life to execute the planned motion.

The experiments provided an understanding of the stability and limitations of the robot, which was achieved through various measured key performance indicators (KPI). The main KPIs used for stair walking were the cost of transport and the precision of task execution. The cost of transport is defined as follows:

$$E_{CT} = \frac{\sum_{m=1}^{M} \int_{t_0}^{t_f} I_m(t) V_m(t) dt}{m_{robot}.g.d} \tag{2.20}$$

In which M is the total number of motors,V_m and I_m are the voltage and current readings of the motor m respectively, m_{robot} is the total mass of the robot, g and d are gravity, and the total distance travelled by the robot.

The precision of task execution was defined as the square of tracking errors of the COM, ZMP and joint angles. The ZMP is the most important as it ensures the robot's stability.

This algorithm implementation was tested in real life on two stair heights, 0.01 and 0.02 m, each having a stride time of 10 sec and 8 sec, respectively. The experiments were tested over five trials with a success rate of 100%.

2.3.2 Stair Climbing using Stereo Vision

For this implementation, Jens-Steffen Gutmann et al.[9] introduced stereo vision to help humanoid robots identify their environment to perform autonomous navigation. The implementation on the QRIO robot uses the data collected from the stereo vision system,

and processes the segments from that view into planar surfaces to identify the stairs or obstacles in the 3D space.

As seen in Fig.2.9, after the plane segmentation, these different planes are put through an edge detector to identify the presence of stairs. According to these identifications, the stair parameters are extracted and inputted as the data to generate the stair climbing motion.

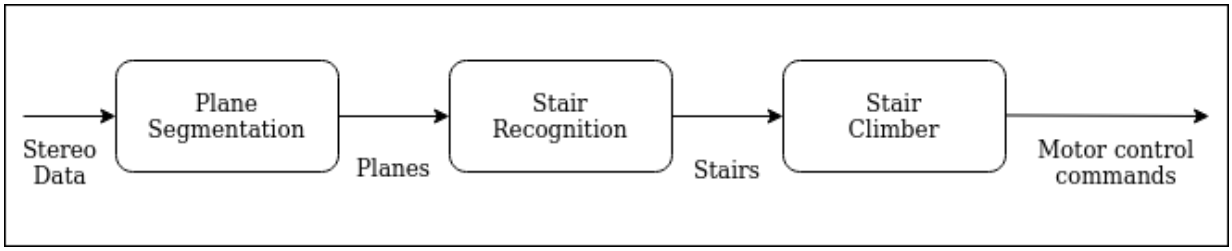


Figure 2.9: Stair Recognition workflow. [9]

The control scheme for the robot motion is implemented in the form of a finite-state machine consisting of the following states:

- Search: scans the 3D space to gather information through moving the head.
- Align: positions itself in a preset position offset from the stairs, then rotates to align with the stairs.
- Approach: move directly in front of the stairs.
- Climb: step up or down depending on the collected data. After stepping perform the search again and repeat till no more steps can be recognized.
- Finish: walk to a safe position on the stairs and turn back to the normal mode of operation.

The key points that need to be verified to determine if the robot would attempt the climbing or not are:

- edge of the stair must be close and the robot aligned to it.
- sufficient space on the sides of the stair for stepping up.

- both feet are placed within the limits of the parameter sets.

As the experiments were performed using the QRIO humanoid robot, the height of the staircases used for testing the algorithm was 3 cm. The robot searches for the stairs for about 5 sec, aligns itself in 5 sec, approaches the stairs in 5 sec and then climbs at a rate of 6 sec per step, completing the four steps in 24 sec.

2.3.3 Optimal Control for Whole-Body Motion Generation for multi-contact configurations

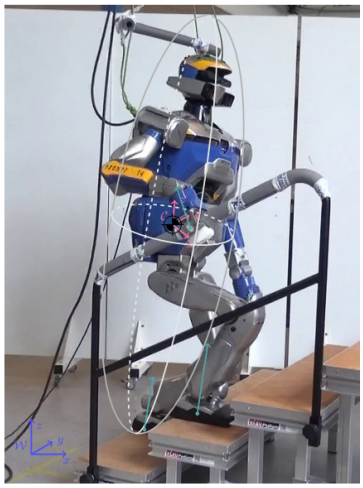


Figure 2.10: HRP-2 on the stairs using support from the handrail. The overlay shows the CoM at the pelvis and the contact forces at hand and feet.

Multi-contact motion is an important problem that would enable humanoid robots to expand on their bipedal locomotion capabilities. This implementation described the formulation of the optimal control problem (OCP) for the reduced multi-contact COM dynamics of the humanoid robot, which is used to generate the whole-body motion implemented for stair walking with the HRP-2 using handrails.

The OCP minimizes the running cost using the Lagrange objective function over a

finite time $t \in [0, T]$ by:

$$\min_{x(),u()} \int_0^T l(x(t), u(t))dt \tag{2.21}$$

$$\tag{2.22}$$

where $l(x(t), u(t))$ is the running cost, $x(t)$ are the states, consisting of COM position, velocity and the angular velocity of the robot; $(c, \dot{c}, \theta, \omega)$, while $u(t)$ are the controls consisting of the contact forces at time t and internal angular momentum variation; $(f_1, ..., f_M, \sigma)$.

The running cost l consists of seven terms which are defined as follows:

- l_0 preserves the COM around the support polygon of the foot contacts

$$l_0 = \|f_{lf}\|_2^2 \cdot \left\|c^{x,y} - p_{lf}^{x,y}\right\|_2^2 + \|f_{rf}\|_2^2 \cdot \left\|c^{x,y} - p_{rf}^{x,y}\right\|_2^2 \tag{2.23}$$

- l_1 tracks the COM reference height dependent on the support foot height

$$l_1 = \left\|(f_{lf,z} + f_{rf,z})(c_z - c_z^{ref}) - f_{lf,z}p_{lf,z} - f_{rf,z}p_{rf,z}\right\|_2^2 \tag{2.24}$$

- l_2, l_3, l_4, l_5 penalize the swinging motion of $\mathbf{c}$ in z-direction and stabilize the rotation.

$$l_2 = \|\dot{c}_z\|_2^2, l_3 = \|\omega_x\|_2^2, l_4 = \|\omega_y\|_2^2, l_5 \|\omega_z\|_2^2 \tag{2.25}$$

- l_6 used as regularization term

$$l_6 = \|\ddot{c}\|_2^2 + \|\dot{\omega}\|_2^2 \tag{2.26}$$

To generate the motion of the robot, these steps need to be followed:

- Locating a set of contact points.
- The OCP computes the trajectory of the COM.
- End effector full trajectory is generated using a B-spline.
- Inverse kinematics computes the joint trajectories of the robot from the previous points.

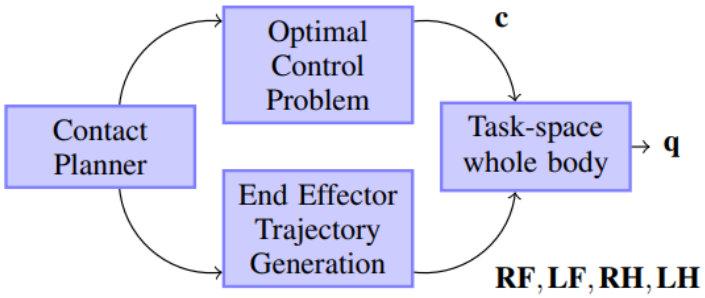


Figure 2.11: Joint trajectories overview with **c** being the COM trajectory, **RF,LF,RH,LH** are the end-effector trajectories and **q** for the generation process.

The climbing task is executed in 3 phases: (i) the right hand accomplishes contact with the handrails, (ii) the right foot is placed at the top of the next step, (iii) the left foot is lifted and aligns next to the right foot on the stair. This sequence is executed twice, coming to a half-sitting stance at the end of the motion. It is important to highlight that throughout the motion, the robot is in a double support phase during the motion of an end effector (support foot and hand) and a triple support phase during the COM transition between feet.

For the experiments, the stair's height, width and depth are 15 cm, 1 m and 30 cm, respectively, while the handrail is cylindrical with a diameter of 3 cm. The experiments have shown that utilizing the handrails reduces the robot's power consumption by 25% compared to performing the motion solely by the feet.

2.3.4 REEM-C University of Heidelberg

This project was conducted at the University of Heidelberg by Sebastian Sontag; the implementation uses the Kajita 2003 implementation for inclined walking similar to the one implemented on iCub 2.3.1. This project has been implemented in Python for ease of prototyping. As seen in Fig. 2.12, the robot is performing the stair walking action on a set of stairs that were designed specifically for this experiment.



Figure 2.12: REEM-C performing the stair walking on the stairs with the help of the lift

The robot uses the following scheme shown in Fig. 2.13 to compute the pattern generation, in which the stair climbing parameters are defined as the bounding conditions for the pipeline. These parameters are used to compute the foot placements, which in turn is used to compute the ZMP. Using the cart-table model from Kajita, the COM trajectory can be computed from the ZMP. Afterwards, the COM and foot placements are imported into the inverse kinematics in order to calculate the joint angles that satisfy these trajectories.

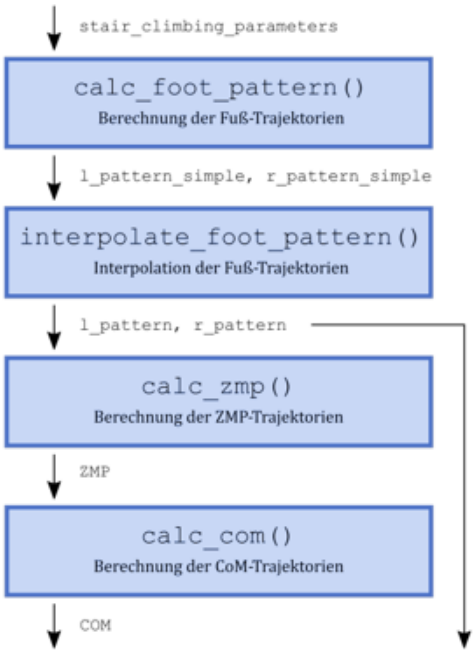


Figure 2.13: Pattern generation pipeline using Python

The robot was tested in simulation as well as real life. The robot was able to perform the task in real life over 4.5, 7, and 9 cm stair heights with the time of step being 60 sec.

2.4 Summary

This chapter provided background on robotics related to this work, such as forward kinematics, inverse kinematics and motion planning. These concepts provide an understanding of techniques used in humanoid robots and their control.

An overview of the related works was also provided to highlight what can be achieved using various methods and robots. Some of them were designed on lightweight or smaller robots making their motion more manageable and energy efficient during execution. These implementations exploited the use of the hands as extra support for the motion. In addition, the stair recognition algorithm presents a possible way to input the stair parameters into the planning pipeline. By looking at the specifications and capabilities of the robot, REEM-C's stereo camera would allow it to recognize and identify the stair using the same

algorithm implemented on the QRIO robot. In addition, exploiting the stabilization of the CoM using optimal control could yield a faster and more stable motion. Finally, through the use of whole-body control, utilizing the upper body as in the use of a handrail would assist the robot in performing the stair walking motion with less torques being exerted by the legs.

This thesis aims to implement an open loop stair walking algorithm from Kajita walking [13] on the REEM-C in which the full trajectory is being pre-planned and played on the robot. Similar implementations were performed, with one on the REEM-C found in the University of Heidelberg in which Python was used for ease of prototyping, and the other on the iCub implemented by Yue Hu et al.[12] in which Matlab was used for the computation. The goal is to improve on the previous implementation executed on the REEM-C robot in the University of Heidelberg, through the exploitation of an optimized programming language such as C++, in order to reduce computation time for the pre-planning. Finally, an improvement in the execution time of the motion on the robot compared to the time performed at the University of Heidelberg.

Chapter 3

Model Update

In this chapter, the process of updating the model of the REEM-C will be discussed in more detail. A proper model that describes the robot is crucial, especially during the implementation process. A model that comes as close as possible to representing the real robot's dynamics and properties would result in a more predictable and repeatable outcome. This would guide us to be more confident in our simulation results and therefore have a reproducible and efficient pipeline for our experiments.

Looking at the model that was provided, it was clear that there were some discrepancies and inconsistencies. This led us to believe that the model needed to be improved, to reflect the results perceived when carrying out the experiments in real life. The noticeable difference between the weight of the whole model and the real robot was of approximately **7 kg**. Table 3.1

Table 3.1: Total Mass of different models.

Model Name	**Model Mass (kg)**
Main PAL Model	67.295
Updated PAL Model	69.917
Heidelberg Model	76.442
Handbook	74.29

PAL provided the lab with an updated model, which increased the robot's mass but was insufficient to reflect the robot's behaviour in real life. The update was made to moments of

inertia of the links being modified to reflect their irregular geometry, rather than assuming they were regular cylinders, which would reflect how the joint dynamics would actually contribute to the motion.

Felix Aller et al. [2] has performed a mass characterization on a previous version of the REEM-C found in Heidelberg, where each link was weighed individually to have an accurate mass representation in the model. It was found that the COM of most of the links was slightly shifted, but a significant difference was found in the TORSO 2 link, i.e. pelvis, which had the COM position shifted sideways to the back of the robot and was discovered to weigh an additional **6 kg**. This was suspected to be because of the weight of the battery, as it is irregular in shape and consists of wiring which cannot be properly modelled and included.

The new model was constructed from the collective data from both the new PAL Model and Heidelberg Model. Since both robots had the same structure, the masses and the COM positions were updated using the values from the Heidelberg Model of REEM-C. Regarding the moments of inertia of the links, the values provided by PAL were considered the most reliable as they were derived more accurately from the CAD model of the robot.

Updating these values yielded a better reflection on the real robot, and some of the issues faced in real life were now reflected in the simulation. For example, the issue of deviation from walking in a straight line or falling backward when performing stair walking was now visible. This update helps lessen the discrepancy between what is observed and what is simulated and plays an important role in ensuring that stair-walking and all future motions played on the robot can be accurately assessed and executed. The properties of the updated links of the model can be found in Table 3.2.

Table 3.2: Link properties.

Link	Mass [kg]	Center of mass location w.r.t link frame [mm] x	y	z
Legs				
base_link	14.89	-29.021	-1.8945	18.427
Leg_link_1	1.02901	-25.71	0.0	25.57
Leg_link_2	0.69621	0.57	-8.81	-11.25
Leg_link_3	4.82	152.1	-9.4	-15.2
Leg_link_4	3.35	136.8	9.9	-17.3
Leg_link_5	0.68213	-0.28	11.19	-8.59
Leg_link_6	2.00843	61.11	-3.36	-4.97
Torso and head				
torso_link_1	1.58341	-0.11	-0.02	-6.69
torso_link_2	14.62	-18.3	-149.8	1.6
head_link_1	1.11967	18.19	2.48	-5.17
head_link_2	1.63134	11.83	-80.98	1.04
Arms				
arm_link_1	1.53614	7.01	-0.11	186.27
arm_link_2	1.66	25.2	-62.7	17.5
arm_link_3	1.71	2.4	-24.5	57.3
arm_link_4	1.27	-34.2	24.1	-16.5
arm_link_5	1.23	6.7	7.1	70.5
arm_link_6	0.41597	-0.05	-2.52	0.07
arm_link_7	0.10051	0.0067314	-0.13277	-29.334

3.1 Torque Computation

Another issue to be resolved is proper evaluation of the joint torques being applied by the robot. The two approaches that have been implemented and discussed later are:

- Motor Current Readings.
- Inverse Dynamic from the model.

Both approaches involve using the joint_states ROS topic to access the joint angles or the efforts these corresponding joints apply. As seen in Fig.3.1, the effort channel can extract the torques produced by the joints. In simulation, the effort output is directly the joints' torque values. In contrast, in real life, the data in the effort channel supplies the motor currents, which need to be processed to compute the motor torques.

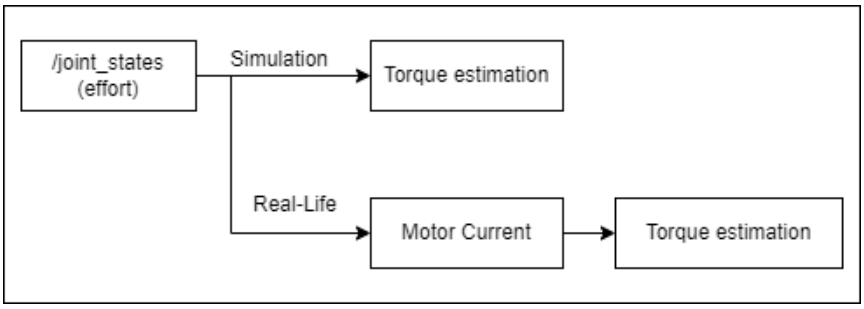


Figure 3.1: Output of effort topic.

3.1.1 Motor Current Readings

As mentioned earlier, from the simulation point of view, the effort topic in the Gazebo environment would directly reflect the generated torque values. On the contrary, when it comes to the real robot, this topic publishes the motor current readings, which in its raw format is quite noisy.

This led to the creation of a pipeline to provide useful data when performing experiments. This pipeline seen in Fig.3.2 was implemented as follows:

- Subscribing to the joint state topic
- Extract the effort of the joints, i.e. motor currents
- Filter the data to remove the noise using the moving average
- Calculate the torques based on the motor constants and gear ratios as follows

$$MotorTorque = \frac{MotorCurrent * MotorTorqueConstant * GearRatio}{1000.0}$$

- Publish the Joint Torques over a topic

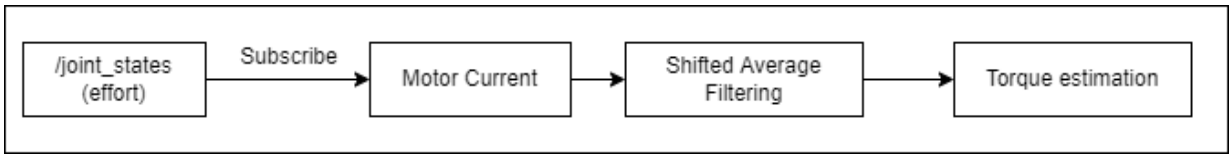


Figure 3.2: Filtering effort data

3.1.2 Inverse Dynamics

A model describing the robot's dynamics is essential to achieve more realistic simulations before running them on the real robot. It was expected that performing inverse dynamics on the model throughout the experiments could yield better torque readings as it would have less noise compared to the motor current readings and would be bound to the joint angle limits.

For this approach, an implementation of the inverse dynamics algorithm using the **RBDL Library** in C++ was used as it was the same library used to perform inverse kinematics for the robot in the later stages. In addition, it would allow more flexibility when applying the forces at different points on the robot links, as seen in Fig. 3.3, and can support the tree structure of the humanoid robot.

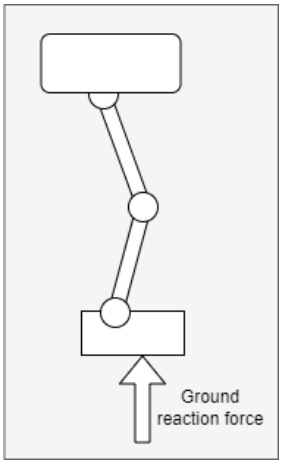


Figure 3.3: Inverse dynamics

To be able to compute the torques we needed to collect:

- Joint angles and velocities
- Torque and Force readings from the sensors in the feet

The readings from the sensors would indicate the external forces due to the ground reaction forces. The humanoid robot is a floating base system, meaning that the robot has to be acting against gravity to be able to stand still.

The RBDL library uses the Featherstone method when performing all of its computations. The solution to the inverse dynamics problem is computed through the recursive Newton-Euler method consisting of three steps:

- computing joint velocities and accelerations
- computation of the net force on each link dependent on its inertia and motion
- calculating the individual joint forces to produce the forces from step 2

By defining $\hat{v}_i$ as the absolute velocity of link i and $\hat{a}_i$ as its absolute acceleration, the recurrence relation of the velocities can be expressed as the sum of velocities of link *i-1* across joint i.

$$\hat{\mathbf{v}}_i = \hat{v}_{i-1} + \hat{s}_i \dot{q}_i. \quad \left(\hat{v}_0 = \hat{\mathbf{0}}\right) \tag{3.1}$$

The acceleration relation is derived by differentiating the velocity equation which is given by:

$$\hat{a_i} = \hat{a_{i-1}} + \hat{v}_i \times \hat{s}_i \dot{q}_i + \hat{s}_i \ddot{q}_i \quad \left(a_0 \hat{=} 0\right) \tag{3.2}$$

The iteration between these equations computes the velocities and accelerations of every link.

Afterwards, the net forces that are acting on link i are represented by the spatial equation of motion with relation to the rate of change in the momentum I.

$$\begin{aligned} \hat{f}_i^* &= \frac{d}{dt}\left(\hat{\mathbf{I}}_i \hat{\mathbf{v}}_i\right) \\ &= \hat{\mathbf{I}}_i \hat{\mathbf{a}}_i + \hat{\mathbf{v}}_i \hat{\times} \hat{\mathbf{I}}_i \hat{\mathbf{v}}_i \end{aligned} \tag{3.3}$$

The total forces transmitted from link*i-1* to link i through joint i can be shown below.

$$\hat{f}_i - \hat{f}_{i+1} = \hat{f}_i^* \tag{3.4}$$

This equation can be rearranged in order to compute the recursive relation and get the individual joint forces.

$$\hat{f}_i = \hat{f}_{i+1} + \hat{f}_i^{\,*} \quad (\hat{f}_n = \hat{f}_n^{\,*}) \tag{3.5}$$

This relation enables us to compute the spatial joint forces from the end effector until the base. If there are external forces acting on the joints, this relation would be modified as follows:

$$\hat{f}_i = \hat{f}_{i+1} + \hat{f}_i^{\,*} - \hat{f}_i^x \tag{3.6}$$

with $\hat{f}_i^x$ being the external forces acting on link i.

3.2 Results

This section discusses the results for each previously described approach. A preliminary test was performed where the robot stood still to have a baseline understanding of its performance and reduce errors when evaluating the torque values.

3.2.1 Motor Current Reading

As explained earlier, there was lots of noise present in the raw readings received directly from the robot shown in Fig.3.4. Due to the lack of time, we have performed the moving average filter approach similar to the implementation by Seng Hansun [10], as it would remove the significant noise in the signal as seen in Fig.3.5.

Looking at the results from the initial approach provides confidence to apply this on a more dynamic motion and verify if a clean torque reading can be captured for safety/control purposes. These are the trial results, shown in Fig.3.5. The readings from both feet during this experiment were symmetrical confirming that this approach is reliable.

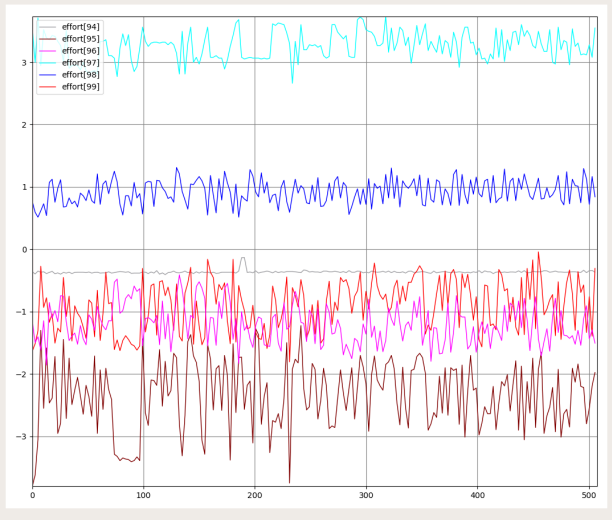


Figure 3.4: RAW motor readings while standing still.

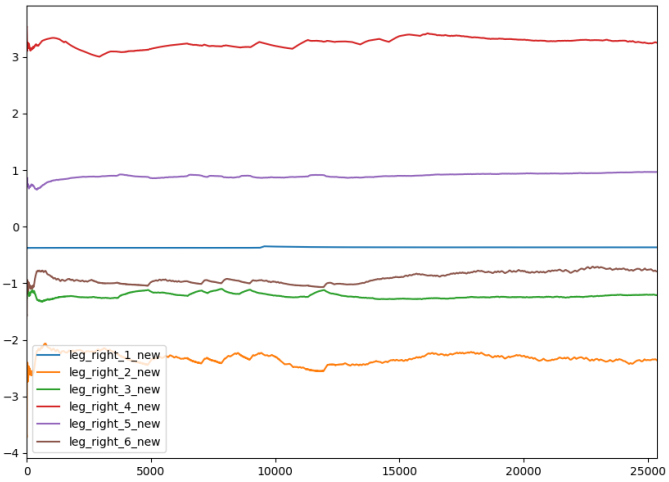


Figure 3.5: Filtered motor reading while standing still.

The output of the standing still pose can be seen in Table 3.3, again confirming that this approach is reliable.

Table 3.3: Torques from Motor current measurements in default configuration.

Joint Names	Joint Torques [Nm]	Joint Names	Joint Torques [Nm]
Left_Leg_link_1	0.72	Right_Leg_link_1	1.03
Left_Leg_link_2	5.76	Right_Leg_link_2	5.70
Left_Leg_link_3	6.02	Right_Leg_link_3	7.20
Left_Leg_link_4	35.21	Right_Leg_link_4	35.35
Left_Leg_link_5	6.02	Right_Leg_link_5	4.83
Left_Leg_link_6	4.02	Right_Leg_link_6	4.00

3.2.2 Inverse Dynamics

Looking at the results from this approach, the values received from the tests on the stationary pose had discrepancies as the torques from both feet were different, although the robot is distributing its weight equally on both legs. The torques produced from the initial tests can be seen in Table 3.4.

Table 3.4: Inverse dynamic results in default configuration.

Joint Names	Joint Torques [Nm]	Joint Names	Joint Torques [Nm]
Left_Leg_link_1	2.03	Right_Leg_link_1	1.38
Left_Leg_link_2	-2.11	Right_Leg_link_2	-45.12
Left_Leg_link_3	-12.11	Right_Leg_link_3	-16.97
Left_Leg_link_4	-5.53	Right_Leg_link_4	-16.81
Left_Leg_link_5	-0.307	Right_Leg_link_5	-16.78
Left_Leg_link_6	-13.77	Right_Leg_link_6	-44.77

These discrepancies are suspected to be due to an issue encountered during the implementation, which was clear during the definition of the floating base of the robot. By importing the URDF model in the library, the base_link definition is not the same as the root_link; this prompts the library to define the root_link location randomly, resulting in a problem when applying the force and torques on the feet, as the reference point is not well defined resulting in these arbitrary readings. By exploiting the transformation relation of the world frame; i.e. root_link, to the floating base a better result can be achieved. A

preliminary test can be achieved through the use of a fixed frame for the base of the robot and computing the torques needed to perform such motions depending on the captured data from the simulation. Finally, converting the model file type from URDF files to LUA files would provide improved handling of this case, as the relation between the world frame and the base frame is more direct. This approach has a lot of potential and therefore needs further investigation in order to make it compute properly, and as mentioned the library is being utilized to perform its computations on LUA models instead of URDF models which could be one of the error sources.

3.3 Conclusion

The robot model was updated with masses and COM positions from various sources to reflect the robot's properties in real-life. With these updates, the current model has a similar mass as the real robot, and the COM positions are in a more accurate location. These updates have resulted in the model reflecting real-life behaviours that only appeared after the update.

In the attempt to compute the joint torques, two approaches were attempted; one being the calculation of the torques using the kinematics and dynamics of the robot via inverse dynamics, while the other extracts motor current sensor readings and computes the torques after filtering.

A recap on the tools used within the experiments, as mentioned before were Gazebo and RBDL. Gazebo was the physics engine used for simulation in order to capture the interactions of the robot with the stairs, as well as the robot dynamics during the motion. We can collect the joint states; joint angles, velocities and torques, as well as the sensor readings measured from the force-torque sensors of in the foot from within the simulation. In addition, Rigid Body Dynamics Library (RBDL) is a C++ Library that uses the data values from Gazebo to perform the necessary evaluations in computation such as the use of either Inverse Kinematics or Inverse Dynamics.

The results from the inverse dynamics approach were not viable as the results received did not provide a sensible outcome during the default standing stance. This was due to the issue of defining the floating base, as discussed earlier, resulting in invalid computation of torques. Further investigations need to be done on this approach, as it may have potential in the future.

The conclusion was to implement the motor current reading approach and deploy it on the real robot to provide a live stream of the torque readings while performing any

operation, as it provides logical and consistent output during experiments. A concern during the implementation was that excess filtering might occur and actual data would be altered too much. Various filtering windows were used until a suitable window was reached, resulting in a more clean reading of the torques. In addition, a concern that needs to be addressed is the motor reaching its hardware limit causing a spike in the motor readings, which may be identified via some form of anomaly detector.

Chapter 4

Stair Walking Implementation

This chapter develops the implementation of stair walking for humanoid robots consisting of test bed design and locomotion. This work is based on the experiments done by Sebastian Sontag on the REEM-C in Heidelberg[32]. To estimate the robot's capability, experiments were performed on different setups to observe the different performances of the humanoid robot. The requirements for this project were as followed:

- Finding a flexible design for the stair
- Implementation of the stair walking algorithm
- Testing the limitations

During the implementation phase, the stair walking motion was implemented on various stair heights to evaluate the robot's capabilities. The results and feasibility of each stair height are discussed later in this chapter.

4.1 Workspace Design

In this section, we will go through the different stairs used. The main criteria for the stairs was to find a robust apparatus that would not deform due to the robot's weight and would provide the flexibility of different stair heights for different implementations on both the humanoid robot and the exoskeleton groups of the lab.

4.1.1 Design Approach

The first approach was to design a generic staircase with a standard stair height and import it into our simulation environment to have a rough estimate of our outline for the dimensions of the stairs. The specifications of the initial design of the stairs can be seen in Table 4.1 :

Table 4.1: Initial stairs specifications.

Stair Parameters	Units
Stair Height	0.145 m
Stair Width	0.6m
No. of Steps	3 steps

A three step stair was designed to provide the required weight needed for the stairs to be rigid in place once the robot steps on them. As it can be seen in Figure 4.1, a top and side view is presented to give a sense of dimensions relative to the robot.

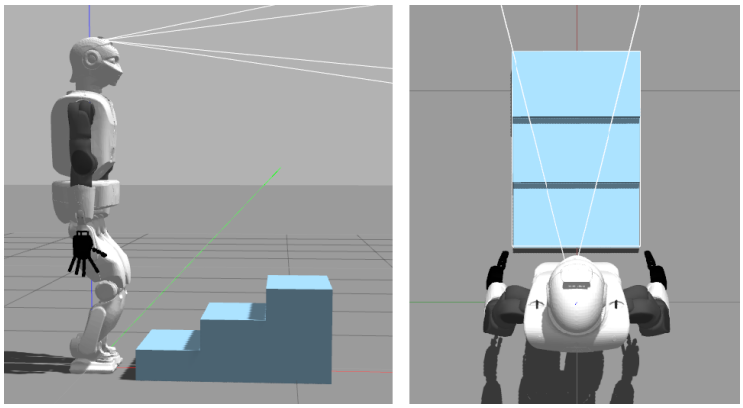


Figure 4.1: Initial stair top/side view

The best solution was to locate a set of stairs that are pre-designed to accommodate the different needs of the lab and provide us with the robustness of the design. The final decision was to get a set of stairs designed for rehabilitation patients, as it would provide us with two sets of heights for the stairs as well as various configurations.

As it can be seen in 4.2, the stair is made of 2 partitions which allows it to be constructed in 2 different configurations, either a linear configuration as can be seen in Fig 4.2 or a 90-degree angle configuration. Fig 4.2 shows that both sides have different height stairs, with one side having a 10 cm stair, while the other has 15 cm stairs.

The linear configuration can be used for simple straight stair walking, allowing the robot to climb up the stairs and then continue its motion to descend on the other side. The other configuration would require the robot to perform an extra rotation in place and then descend.

As the smallest stairs were the primary concern while modelling the stairs to import into our simulation environment, they were the only side to be imported and modelled. The stairs can be seen in the following Fig 4.2.



Figure 4.2: Stairs in real life

4.2 Implementation

In this section, we dive deeper into the implementations to achieve the results with the stair walking on the REEM-C robot and the algorithm that has been set into action.

An overview of the workflow pipeline is shown in Fig.4.3. The stair parameters are first defined as the guiding outline for the whole process. These parameters are used to generate the motion pattern, which includes the foothold generation followed by the ZMP trajectory computation. Finally, the COM trajectory is calculated from the ZMP using the LIPM [15]. Once the feet and COM trajectories are planned, they are used as input to the upcoming stage as constraint sets for the URDF model to compute the inverse kinematics of the whole motion, which produces the joint angles that would be used for motion execution. Finally, the motion is tested in simulation or real-life using the joint trajectory player developed by the lab. For simulation, the joint trajectories are tested on Gazebo to verify the motions in a physical environment where gravity is acting, and interactions between objects are present.

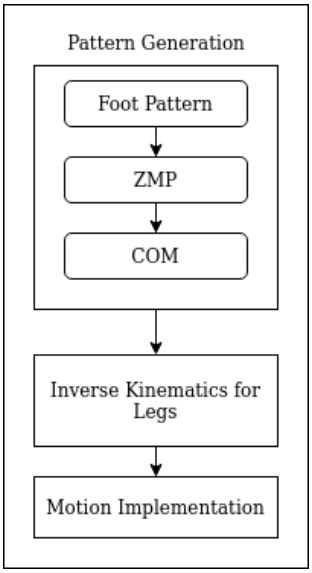


Figure 4.3: Work flow pipeline

The desired motion of the robot while performing the stair walking would be to shift the COM to the center of the support polygon, which would be on the single support foot, then for the lift-off foot to move upward and clear off the stairs. Afterwards, it moves forward by the specified step length then lowers that foot in order to make it the support

foot for the next step. These motions are repeated consecutively until the robot reaches the final step of the stairs in which both feet are next to each other, and the robot returns the COM in between them, as seen in Fig. 4.4.

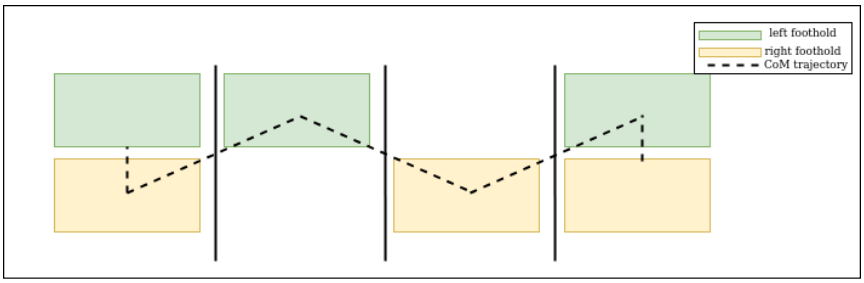


Figure 4.4: Footholds and CoM trajectory for stair walking.

4.2.1 Pattern Generation

As seen in Fig.4.5, the pattern generation algorithm is implemented as follows. The stair and step parameters are entered in the configuration file, which is then used as the guideline for the whole pattern generation. From these parameters, the foot holds are defined and generated; afterwards, an interpolation is performed to produce a smoother trajectory to be tracked. A sampled portion of the generated foot trajectories can be seen in Figures 4.6,4.7 and 4.8, showing the trajectories in both x and z axis with time in addition to the side view trajectory of the foot, respectively. The foot trajectory is defined as follows: the foot lifts off vertically until it reaches the step height, afterwards the foot moves horizontally with the step length once it reaches the clearance height, and finally drops vertically to establish support for the next step as seen in Fig 4.8. As observed in both Fig. 4.6 and 4.7, the motion is generated in a manner such that the left foot starts its motion first, and once it reaches the resting position, on the stair, the right foot begins its motion.

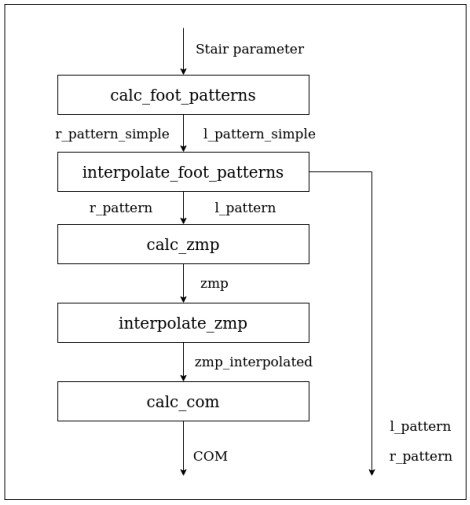


Figure 4.5: Pattern generation pipeline

As soon as the foot holds are generated, they are saved into CSV files to be used later in the pipeline. In addition, the values of the foot pattern are fed to the **calc_zmp** function to calculate the ZMP trajectory, as discussed in 2.2.2, and similarly, it would be interpolated to produce a smoother trajectory to track. Finally, the ZMP points are supplied to the **calc_com** function to produce the COM positions that would be used as a guideline for the pelvis motion. The COM values are also exported into CSV files to be used in the next stage.

Using the same equations mentioned in subsection 2.3.1, the ZMP Equation 2.11 is implemented and can be discretized in the form of $\mathbf{p} = \mathbf{A}\mathbf{x}$:

$$\begin{bmatrix} p_1 \\ p_2 \\ : \\ p_{N-1} \\ p_N \end{bmatrix} = \begin{bmatrix} a_1 + b_1 & c_1 & 0 & \\ a_2 & b_2 & c_2 & \\ & : & : & : \\ & a_N - 1 & b_N - 1 & c_N - 1 \\ & 0 & a_N & b_N + c_N \end{bmatrix} \begin{bmatrix} x_1 \\ x_2 \\ : \\ x_{N-1} \\ x_N \end{bmatrix} \tag{4.1}$$

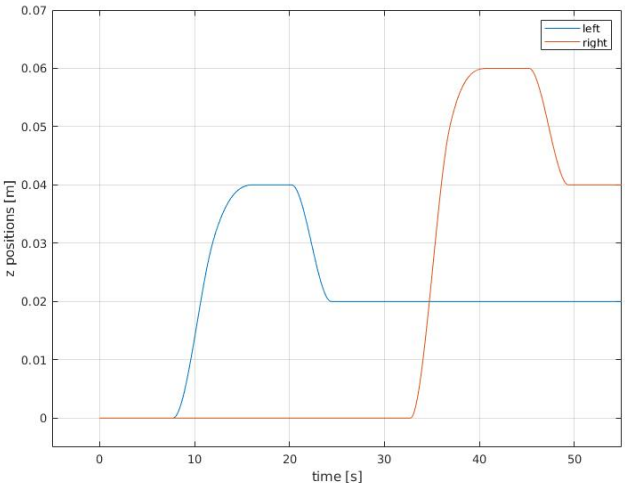


Figure 4.6: Foot trajectory in z component.

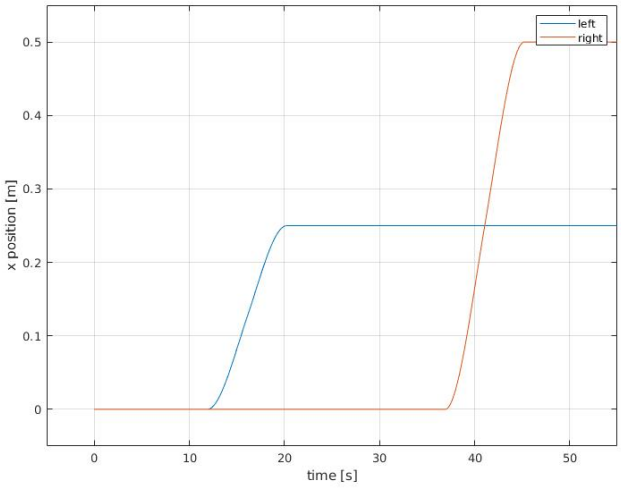


Figure 4.7: Foot trajectory in x axis.

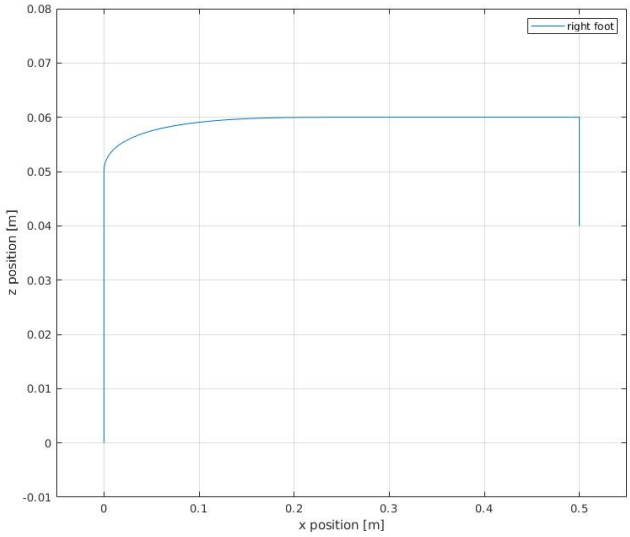


Figure 4.8: Foot trajectory in 2D space (sideview).

4.2.2 Inverse Kinematics

In this stage, a custom Rigid Body Dynamics Library was used[26] to compute the inverse kinematics of the humanoid robot using the points provided from the pattern generation, i.e. CoM and foot positions. As mentioned, the kinematic tree structure is confined by three points, which are on each foot (leg_joint_6) and the pelvis (COM). The initial joint angle values are defined to limit the solutions of the IK. Using the constraints and points streamed from the pattern generation stage, an output of the joint angle values are generated and exported into a CSV file that can be used for the execution stage.

4.2.3 Testing

After the extraction of the joint angles from the previous step [Chap.4.2.2], verification needs to be performed to make sure that they are feasible. The pipeline to implement the motions on the robot were as follows:

- Trials of the motion using RVIZ visualizer 4.50

- Trials of the motion on simulation using Gazebo[8]
- Validation of the torques generated using the *effort* topic
- Trials on the real humanoid robot and analysis of the torques produced

Initially the robot has been tested with a different sets of stairs, constructed from 2 cm wood shelves shown in Fig.4.9, to examine the capabilities of the robot and observe its motion without introducing a critical drop height as we are performing our locomotion on elevated grounds. The implementation as well as the results are discussed further ahead.



Figure 4.9: Shelf dimension

4.3 Analysis

4.3.1 Static Analysis

To verify that the experiments would be viable and to provide a baseline understanding of the torques needed by the robot at the instant after the lift-off, a static test has been conducted in which the robot configuration would be as follows. The robot would have the support feet on the stair while the other is flat on the ground level with the pelvis over the support foot, getting ready for the lift-off. The pelvis represents the robot's upper body, which is considered as a point mass concentrated at leg joint 1 location is space; both links are considered to be aligned horizontally with each other making it possible to

carry on the calculations. With such joint configuration, the static torques of each joint can be computed as the torques would only consider the kinematic configuration without involving any dynamics due to motion. The robot configuration for such computation can be seen in Fig. 4.10.

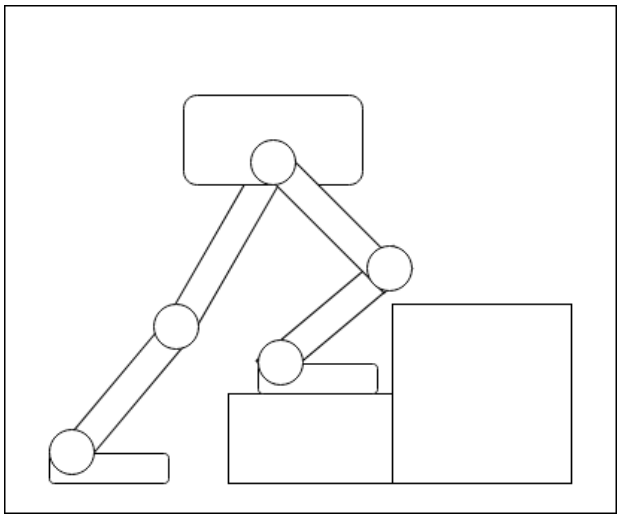


Figure 4.10: Humanoid robot configuration for the static torque analysis, right before lift-off

This static analysis estimates the minimum torques required by the joints under such configuration. This is due to the elimination of the acceleration components; i.e. α, during the torque computation as the acceleration would be equal to zero as there is no motion, resulting in the elimination of the inertia component.

$$\tau = Force * distance + I * \alpha \tag{4.2}$$

The analysis has been performed on all of the stair heights that are discussed later on in the upcoming section. The joint angles as well as the torques required, are represented in the following tables Table 4.2, Table 4.3, Table 4.4 and Table 4.6. As mentioned before, these calculations provide an estimate of the required torques. The torques are being computed using the basic torque equation:

$$\tau = Force * distance \tag{4.3}$$

with Force being the weight of the previous links and distance being the offset from the current joint to the overall CoM of the previous links.

Table 4.2: Static test analysis for the 2 cm stairs configuration

Stair Height	Joint Name	Joint Angle (rad)	Joint Torque (Nm)
	Joint 1 right	-0.05	3.22
	Joint 2 right	-0.18	6.03
	Joint 3 right	0.02	-16.98
	Joint 4 right	0.34	-8.14
	Joint 5 right	-0.63	10.14
	Joint 6 right	0.19	-2.24
	Joint 1 left	-0.05	3.49
	Joint 2 left	-0.17	30.62
	Joint 3 left	-0.79	51.52
	Joint 4 left	1.10	-46.83
	Joint 5 left	-0.57	48.46
2 cm	Joint 6 left	0.18	-22.14

Table 4.3: Static test analysis for the 4 cm stairs configuration

Stair Height	Joint Name	Joint Angle (rad)	Joint Torque (Nm)
	Joint 1 right	-0.05	0.90
	Joint 2 right	-0.19	6.94
	Joint 3 right	-0.22	-14.13
	Joint 4 right	0.72	-6.69
	Joint 5 right	-0.76	11.14
	Joint 6 right	0.19	0.68
	Joint 1 left	-0.05	8.07
	Joint 2 left	-0.19	31.84
	Joint 3 left	-0.94	55.07
	Joint 4 left	1.32	-48.97
	Joint 5 left	-0.67	60.36
4 cm	Joint 6 left	0.19	-17.13

Table 4.4: Static test analysis for the 6 cm stairs configuration

Stair Height	Joint Name	Joint Angle (rad)	Joint Torque (Nm)
	Joint 1 right	-0.05	7.09
	Joint 2 right	-0.20	14.90
	Joint 3 right	-0.21	-14.14
	Joint 4 right	0.71	-4.46
	Joint 5 right	-0.76	-0.72
	Joint 6 right	0.20	5.74
	Joint 1 left	-0.05	2.18
	Joint 2 left	-0.20	38.61
	Joint 3 left	-0.97	48.87
	Joint 4 left	1.41	-55.44
	Joint 5 left	-0.71	43.53
6 cm	Joint 6 left	0.20	-20.0

Table 4.5: Static test analysis for the 8 cm stairs configuration

Stair Height	Joint Name	Joint Angle (rad)	Joint Torque (Nm)
	Joint 1 right	-0.05	1.81
	Joint 2 right	-0.19	7.53
	Joint 3 right	-0.23	-26.64
	Joint 4 right	0.75	-13.63
	Joint 5 right	-0.78	10.30
	Joint 6 right	0.20	5.31
	Joint 1 left	-0.05	2.64
	Joint 2 left	-0.21	25.36
	Joint 3 left	-1.05	62.32
	Joint 4 left	1.55	-68.13
	Joint 5 left	-0.78	52.83
8 cm	Joint 6 left	0.22	-23.29

Table 4.6: Static test analysis for the 10 cm stairs configuration

Stair Height	Joint Name	Joint Angle (rad)	Joint Torque (Nm)
	Joint 1 right	-0.05	2.94
	Joint 2 right	-0.19	15.16
	Joint 3 right	-0.24	-15.22
	Joint 4 right	0.77	-31.63
	Joint 5 right	-0.79	1.99
	Joint 6 right	0.20	-2.17
	Joint 1 left	-0.02	-10.67
	Joint 2 left	-0.21	-59.85
	Joint 3 left	-1.07	64.78
	Joint 4 left	1.62	114.82
	Joint 5 left	-0.79	-50.23
10 cm	Joint 6 left	0.21	-22.73

Table 4.7: Static test analysis for the 12 cm stairs configuration

Stair Height	Joint Name	Joint Angle (rad)	Joint Torque (Nm)
	Joint 1 right	-0.05	2.94
	Joint 2 right	-0.19	15.16
	Joint 3 right	-0.24	-15.22
	Joint 4 right	0.77	-31.63
	Joint 5 right	-0.79	1.99
	Joint 6 right	0.20	-2.17
	Joint 1 left	-0.06	-28.49
	Joint 2 left	-0.22	-69.06
	Joint 3 left	-1.08	-63.8
	Joint 4 left	1.68	120.9
	Joint 5 left	-0.80	-47.63
12 cm	Joint 6 left	0.20	-20.0

Table 4.8: Static test analysis for the 14 cm stairs configuration

Stair Height	Joint Name	Joint Angle (rad)	Joint Torque (Nm)
	Joint 1 right	-0.05	2.94
	Joint 2 right	-0.19	15.16
	Joint 3 right	-0.24	-15.22
	Joint 4 right	0.77	-31.63
	Joint 5 right	-0.79	1.99
	Joint 6 right	0.20	-2.17
	Joint 1 left	-0.06	-28.49
	Joint 2 left	-0.24	-71.4
	Joint 3 left	-1.11	-56.59
	Joint 4 left	1.72	140.4
	Joint 5 left	-0.85	-33.63
14 cm	Joint 6 left	0.23	-19.43

By observing the values from the static test, it can be seen that as the stair height varies, the rear foot; i.e. right foot, configuration is almost consistent. On the other hand, the configuration of the front foot (lift-off foot); i.e. left foot, varies with the varying step height, which is expected as the distance from the base of the foot to the pelvis changes. An important observation seen within the tables, is that the torques on the Knee joint of the lift-off foot; i.e. left foot, increase as the stair height increases, which indicates that the majority of the robots' weight is being supported by that joint. This builds a baseline understanding of the behaviour expected once the motion becomes dynamic, as the acceleration component would then be involved. Through these computations, it is clear that the robot has the ability to supply the required torques to withstand the lift-off motion for the questioned stair heights. Further results on the dynamic motion results; i.e. the whole motion, would be discussed in the upcoming sections.

As seen from Table 4.8, under the prescribed configuration, the left foot joints 2 and 4 have reached their limits within the static analysis once reaching the **14 cm** stairs. This indicates that the robot cannot perform any bigger stair heights as the torques required for such motion would not be feasible to be supplied by the robots' motors.

When looking at the torque limits of the joints, it is not enough to just look at their joint torque limits in these configurations; it is important as well to compute the unilateral

constraints on the ankle joints that are applied to them due to the weight of the whole robot. This is due to the torques being very limited at the ankle, as it would have to counter the toe/heel lift-off that may be caused by larger torques. The computation of such torques can be achieved through the simplification of the whole robot mass into being a point mass and then computing the torques it produces on the ankles to the foot edge. As mentioned, it is a simple torque equation as shown below.

$$\tau = Force * distance \tag{4.4}$$

with the force in this case being the total mass of the robot and the distance being the offset of the mass from the edge of the foot. This can be described similarly to an inverted pendulum as shown in Fig.4.11.

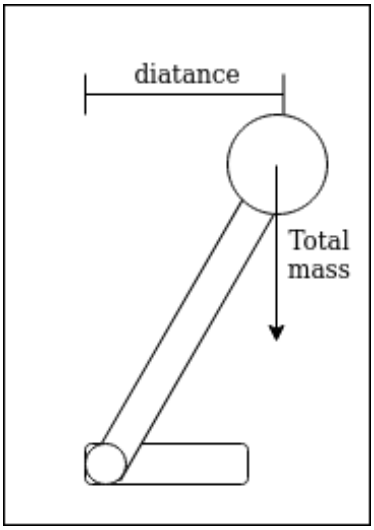


Figure 4.11: Robot model simplified in order to compute the unilateral constraints on the ankle.

Computation of all of the previously discussed stair configurations was made on the ankle joints, and the variation of the torques can be seen in the following figure. In order to simplify the calculations the CoM position is represented as the Pelvis position in space according to the joint angles from the previous discussion and the total mass of the robot was taken as 74.3 kg. The figures show how the ankle torques required changes with the stair height difference.

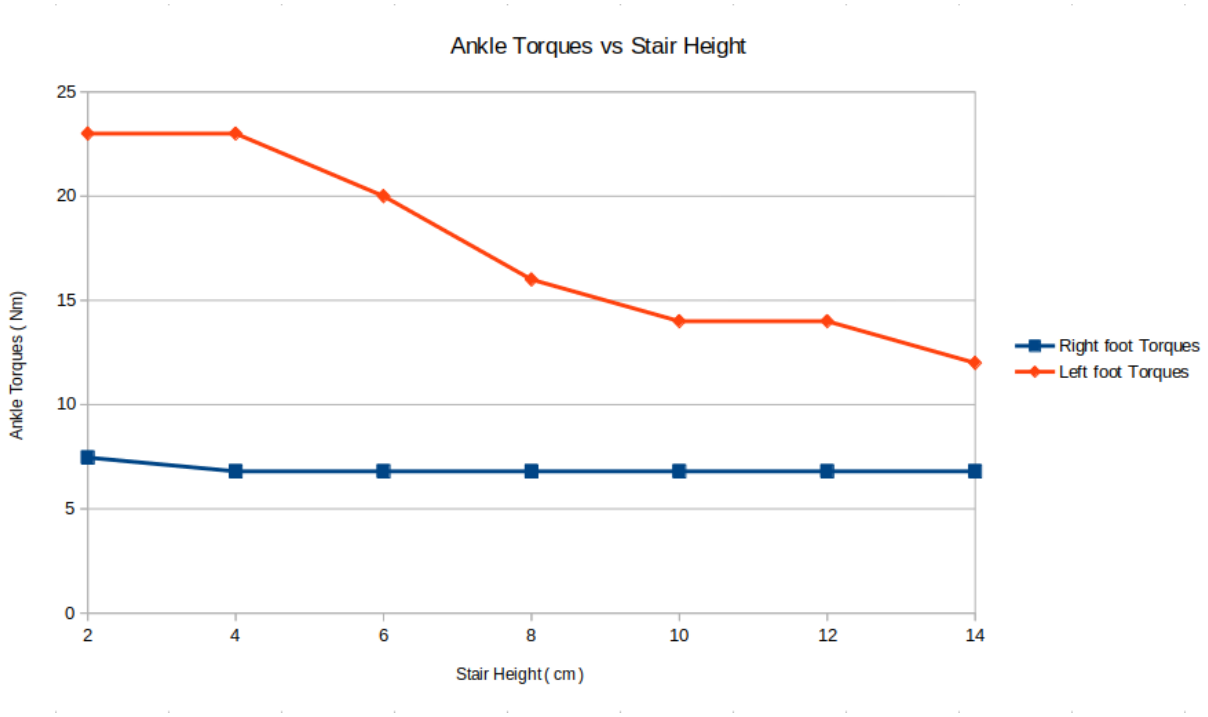


Figure 4.12: Torques required by the ankle joints in the stepping configuration to prevent the lift-off.

As seen from Fig.4.12, the torques generated on the right foot ankle are almost constant throughout the various stair heights, this is due to the robot always being at a consistent distance from the rear foot; i.e right foot, with a value of **6.8 Nm**, the torques on the right foot are so small due to the mass of the robot being supported fully on the left foot, having the weight of the leg as the only load on the ankle joint. On the other hand, the torques required on the left foot can be seen to decrease with the increase of the stair height, this is the result of the CoM getting closer to the ankle joint, therefore, requiring less torque to overcome the weight of the robot.

These torques are added to the torque values generated from the static analysis step and then verified against the joint torque limits from the handbook [28]. When such computation is performed, the ankle joint torques are found to be within the joint torque limits, ensuring that there are no critical torques affecting the ankle at these configurations.

4.3.2 Kinematic Check

To identify the maximum step height achievable by the robot, a kinematic check using the geometry is performed. This computes the feasibility of the motion regardless of the torques needed by the joint.

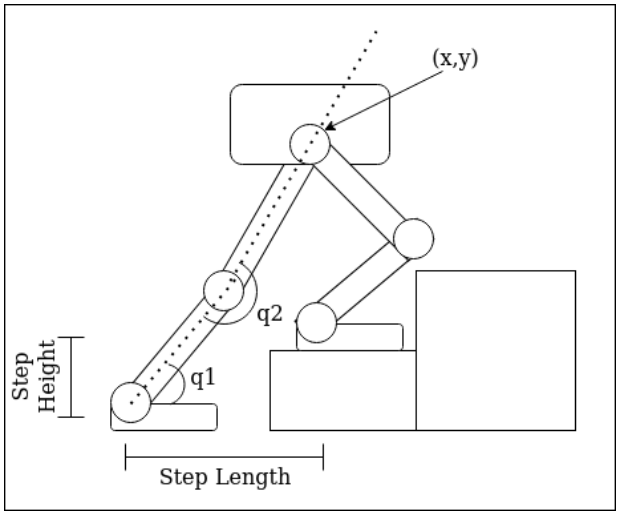


Figure 4.13: Robot Kinematics with one foot on the stair and one on the ground, ankle joint (q1) and knee joint (q2).

The robots' complex DoF can be simplified down into two sections with each having two DoF revolute joints. This allows us to compute the EE position and from the computed data, in addition to the subjected maximum stair height Inverse Kinematics can be performed to validate the viability of the proposed stairs. In order to calculate the common point; i.e. pelvis, at which both feet are connected together, estimate joint angles can be used from the previous subsection as a guideline to the angles reached by the rear foot. Using the joint angles, forward kinematics can be performed using the following equations to capture the EE position in space.

$$x = l1\cos(q1) + l2\cos(q1 + q2) \tag{4.5}$$

$$y = l1\sin(q1) + l2\sin(q1 + q2) \tag{4.6}$$

After the computation of the pelvis position, this data collective with the footstep length and height can be used in order to verify the feasibility of the proposed stair height using

Inverse kinematics which is achieved by the following equations.

$$q1 = \tan^{-1}(\frac{y}{x}) - \tan^{-1}(\frac{l2\sin(q2)}{(l1 + l2\cos(q2))}) \tag{4.7}$$

$$q2 = \cos^{-1}(\frac{x^2 + y^2 - l1^2 - l2^2}{2 * l1 * l2}) \tag{4.8}$$

In order to find the proper lift-off configuration, there is three main characteristics that need to be satisfied in the robot, being a configuration where the shank would not collide with the stair during its maximum leaning position, the joint angles are contained within the joint limits and finally, the position of the overall CoM of the robot being placed on top of the support foot. Using these criteria theoretically, the robot should be able to perform the lift-off motion successfully, in case the torques were not limited.

As observed from the previous subsection the rear foot had similar angles throughout varying stair heights, therefore these angles were used as seen in Table 4.9. From such configuration, the maximum stair height preventing collision with the shank of the robot was computed as 0.22 m. This was computed as follows, the CoM position of the shank was computed under this configuration and the Y component was used as the maximum achievable stair height as seen in Fig.4.14.

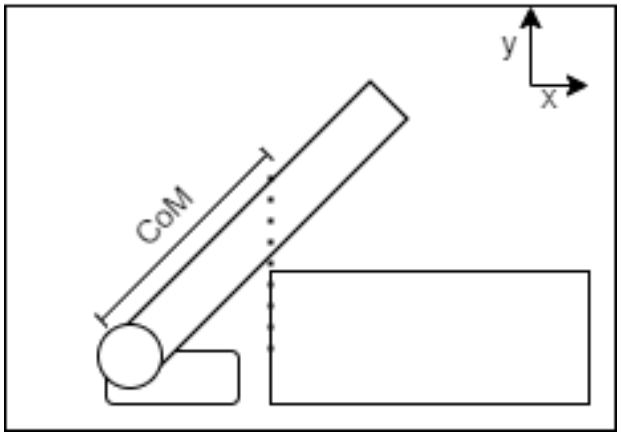


Figure 4.14: The shank CoM position so that the stair would not collide with the stairs.

Table 4.9: Maximum configuration for the rear foot.

Joint	Joint Angle (rad)
q1	0.71
q2	-0.76

The 0.22 m stair height was used to compute the required joint angles using Inverse Kinematics, this resulted in joint angles that would be exceeding the limit. In order to figure out the maximum height, decrements of 0.01 m were performed until the results for the joint angle were within the joint limits. Finally, according to the computations, the maximum stair height that can be achieved in the case of a one-step staircase, by just using the kinematics of the robot is **0.2 m** stairs with a step width of 0.16 m and a stair depth of 0.25 m, and the corresponding joint angles can be seen in Table 4.10 and Table 4.9. All of these computations are having the rear foot with a 0.05 m offset from the stairs in order to eliminate collisions.

Table 4.10: Maximum configuration for the front foot.

Joint	Joint Angle (rad)
q1	0.51
q2	2.05

Regarding the presence of more than two steps in the stair walking scenario, the shank of the front foot should be taken into account in order to ensure no collision with the consecutive stairs. As this scenario is similar to real-life implementations, a check had to be done to provide us with a more realistic outcome. This study was performed on the previous stage, and the results from such test concluded that the 0.2 cm stair heights would not be viable in the case of two or more steps. Similar to what occurred earlier, decrements of 0.01 m were computed with their respective joint angles in order to reach a viable result. Throughout these decrements, the robot step length was 0.22 m with the stair depth being 0.25 m. The first feasible step size in the case of the two-step stairs was found to be the **0.14 m**, with such step height and stair depth the robots' front and rear shanks would not be colliding with the stairs, theoretically insuring that the motion can be executed without the presence of any torque limitations.

To have a relative insight into the future, a preliminary study was performed for the walking down the stairs scenario, which would include the robot being in the following

configuration, as seen in Fig4.15. This configuration would be the critical point of judgment in determining the stair height that can be achieved by the robot going downwards, as the Pelvis has to be on top of the rear foot till the point in which the front foot comes in contact with the upcoming step/ground.

For this study, there are three main characteristics that need to be achieved in order to consider the stair height as feasible. This would be, having the ankle joint within the joint angle limit, a configuration in which the rear side of the shank would not collide with the stairs, and finally the CoM of the robot being on top of the rear foot; i.e. support foot.

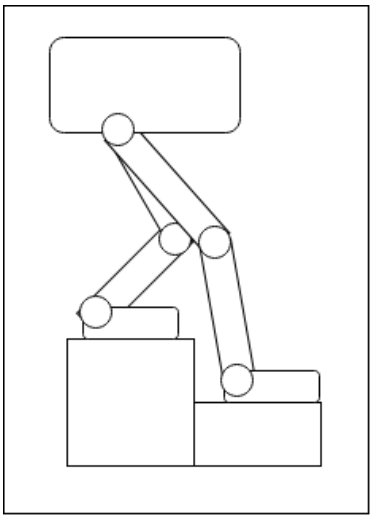


Figure 4.15: Robot Kinematics for walking down the stairs, with one foot on the high step and one on the lower step of the stairs.

Using these equations 4.5 and 4.7, the maximum drop height of the pelvis was computed, in which the joints especially the ankle joint would remain within the limits. With the help of the visualizer from RVIZ, an initial set of joint angles where captured to perform the iterations on. Using these angles the maximum height drop that the robot can perform with the constraints of the Pelvis present on top of the rear foot was found to be 0.42 m; this is the distance from the pelvis to the sole of the foot. Any further drop in the pelvis height would either cause the robot to exceed the ankle joint angles to maintain balance, or the pelvis would be at a critical point that could cause the robot to tip over. The captured joint angles can be seen in the following Table 4.11.

Table 4.11: Maximum configuration for the rear foot.

Joint	Joint Angle (rad)
q1	0.93
q2	1.56

From these joint angles Inverse Kinematics was performed on various stair heights starting with 0.02 m stairs. Increments of 0.02 m were performed till the maximum joint angles would be reached. After various computations the maximum step height for which the robot can perform down stair walking was computed as **0.12 m** with a step length of 0.22 m, that can be achieved using the following joint angles shown in Table 4.12.

Table 4.12: Maximum configuration for the front foot.

Joint	Joint Angle (rad)
q1	1.63
q2	0.71

The visualization of the maximum reachable stair height with the computed joint angles can be shown in the following Fig.4.16.

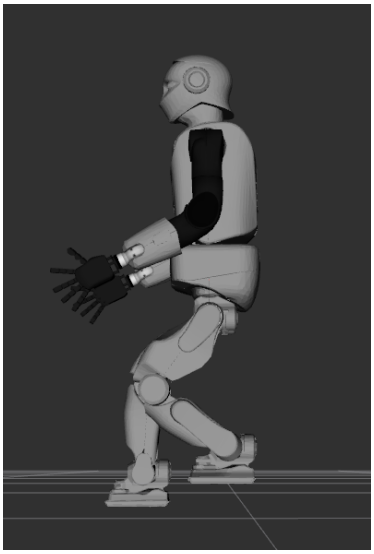


Figure 4.16: Robot Kinematics visualization of the angle values for walking down the stairs, with one foot on the high step and one on the lower step of the stairs.

In conclusion, from the kinematic point of view, the robot can perform up to **20 cm** stairs in the case of a one-step stair and **14 cm** stair in the case of two or more steps, which is a more realistic scenario. On the other hand for the robot to perform the walking down stair motion, it was found that the maximum that can be performed would be the **12 cm** stairs. All of the computations performed in this section has been cross-referenced with the joint angle limits from the PAL handbook [28] to ensure the viability of these configurations.

4.4 Simulation

During the simulation the robot has been kinematic tested on various configurations as well as different stair heights, the simulation results are displayed in this section. Most of the stair height simulations were executed successfully in this stage. Gazebo [8] was the choice of the physics engine used to run the simulations to provide a better understanding of how forces and interactions between the humanoid robot and the stairs affect the motion.

As mentioned earlier increments of 2 cm stairs were used to test the humanoid robot motion and capability until the robot reached its limit. The motions as well as the trajec-

tories of each increment are displayed in the upcoming subsections. In addition to these, the parameters used to generate the motion can be seen in the tables.

4.4.1 2 cm Stairs

The motions are generated with the upper body having fixed joint angles, making the leg joints the only joints that would be moving in an attempt to reach the final pose.

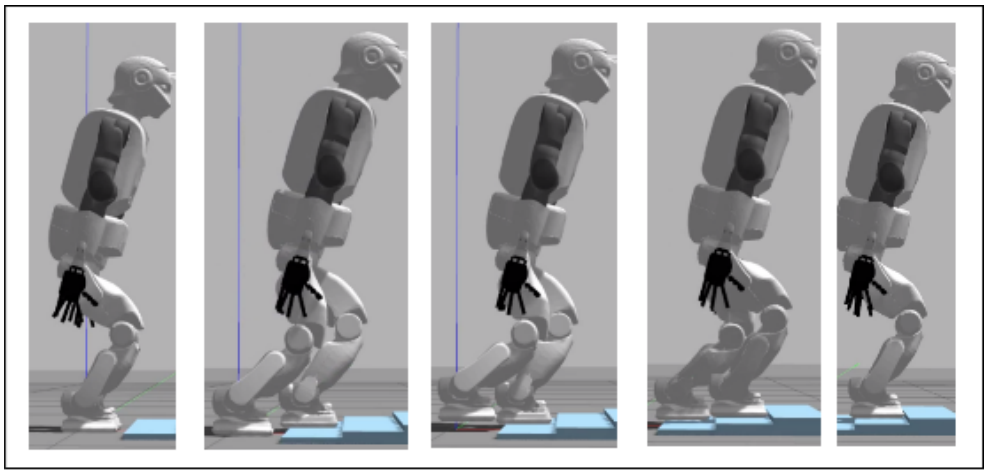


Figure 4.17: 2 cm motion on stairs in simulation.

Table 4.13 shows the parameters used to generate the motion for the 2 cm stairs. The trajectories of this motion can be seen in Figure 4.18 and Figure 4.19.

Table 4.13: Parameters for 2 cm Stairs in simulation.

Parameter	
Stair Height	0.02 m
Stair Width	0.5 m
No. of steps	2 steps
Step Height	0.035 m
Step Drop	0.02 m
Step Length	0.22 m
Step Width	0.16 m
COM Z position	0.76 m
t_step	50 sec

Trajectories

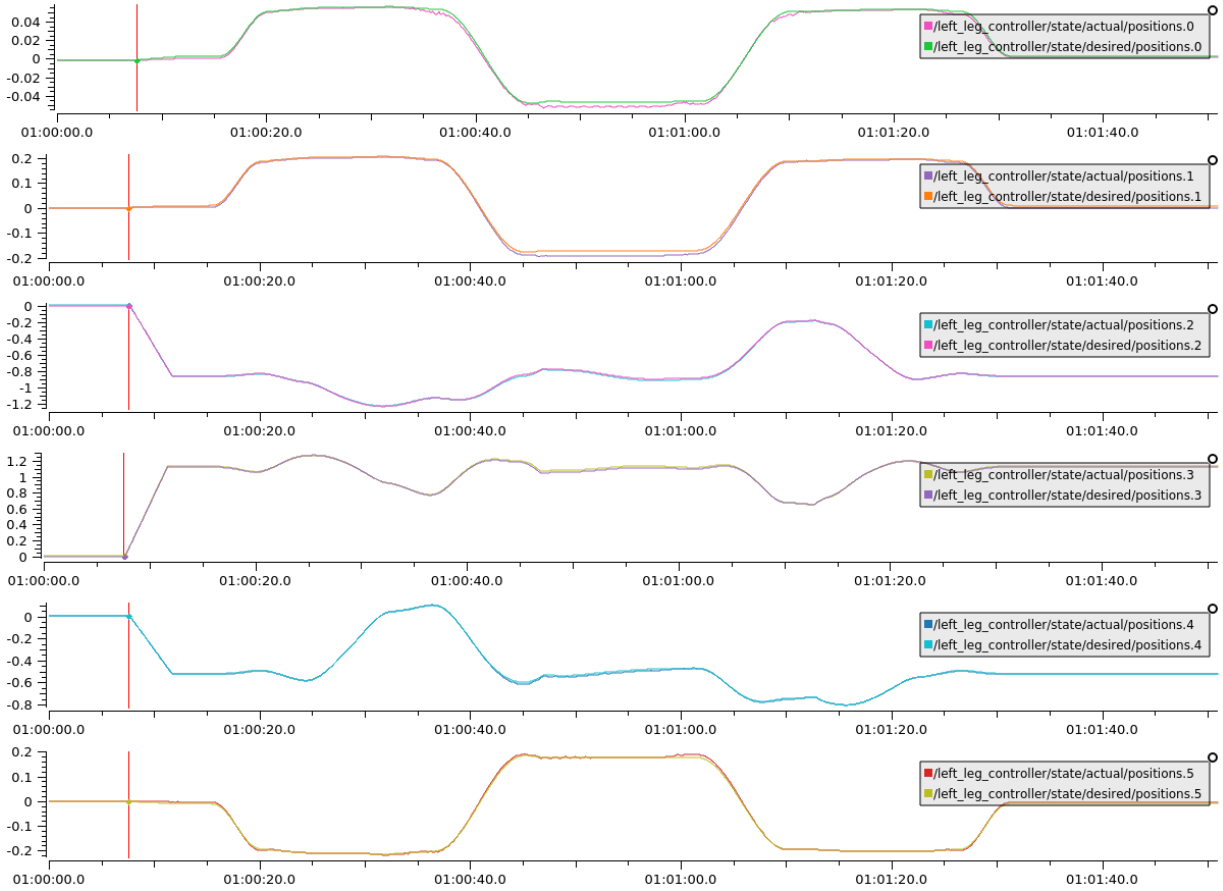


Figure 4.18: Joint angles[rad] vs Time[sec] for 2 cm left leg desired vs actual trajectories on simulation.

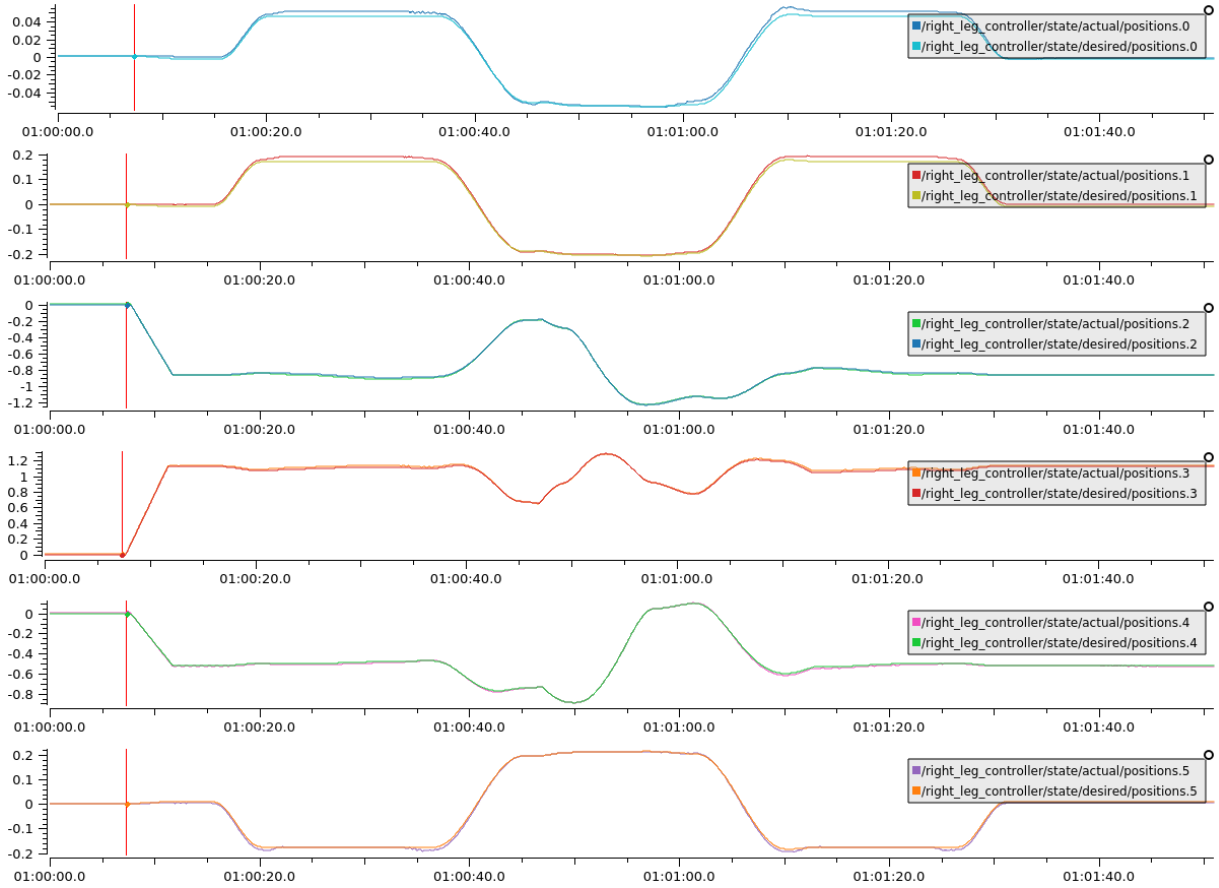


Figure 4.19: Joint angles[rad] vs Time[sec] for 2 cm right leg desired vs actual trajectories on simulation.

4.4.2 4 cm Stairs

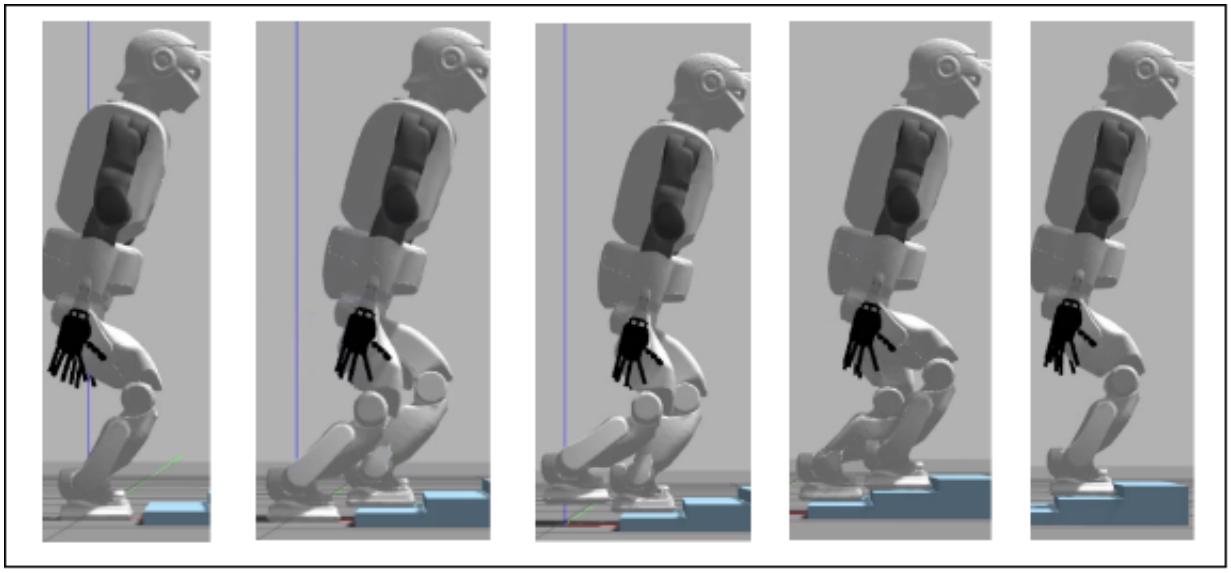


Figure 4.20: 4 cm motion on stairs in simulation.

Table 4.14 shows the parameters used to generate the motion for the 4 cm stairs. The trajectories of such said motion can be seen in Figure 4.21 and Figure 4.22.

Table 4.14: Parameters for 4 cm stairs in simulation.

Parameter	
Stair Height	0.04 m
Stair Width	0.5 m
No. of steps	2 steps
Step Height	0.055 m
Step Drop	0.02 m
Step Length	0.22 m
Step Width	0.16 m
COM Z position	0.76 m
t_step	50 sec

Trajectories

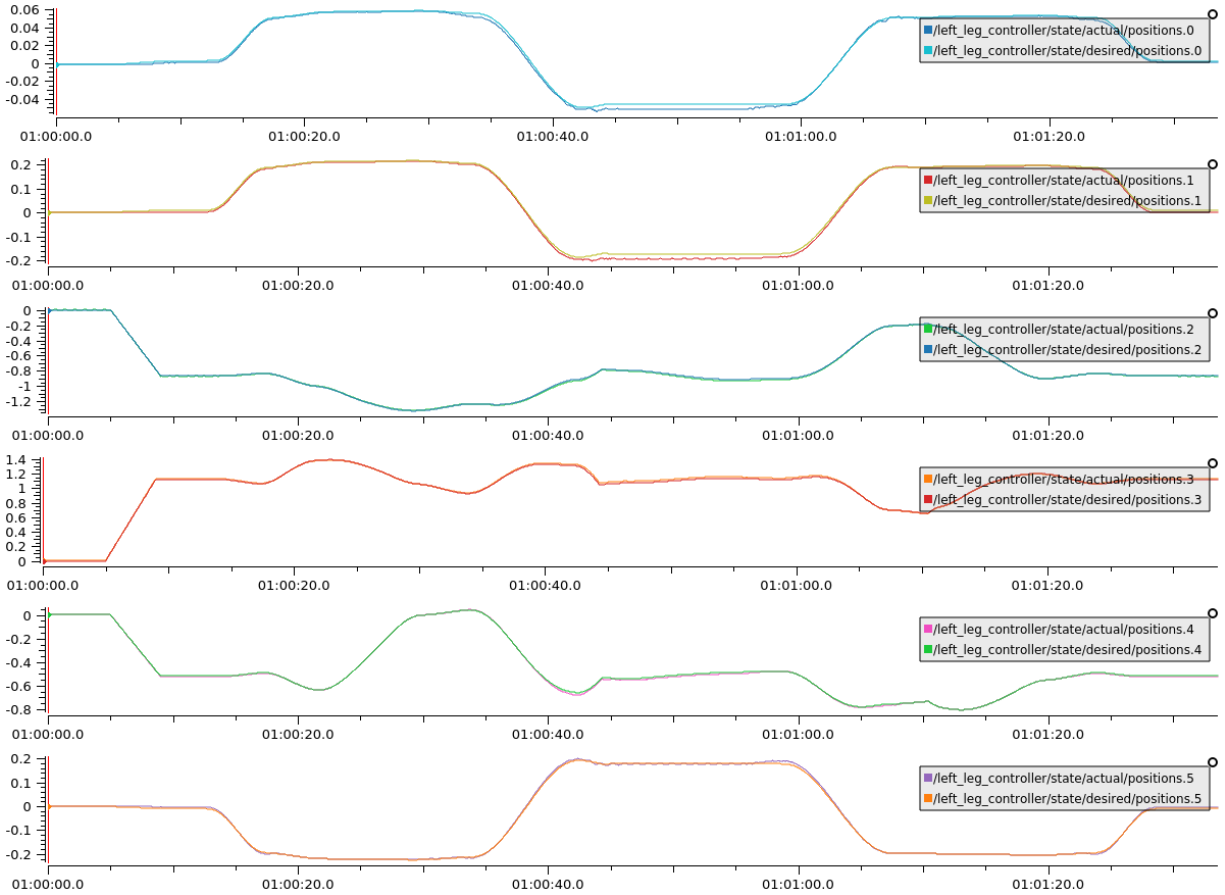


Figure 4.21: Joint angles[rad] vs Time[sec] for 4 cm left leg desired vs actual trajectories on simulation.

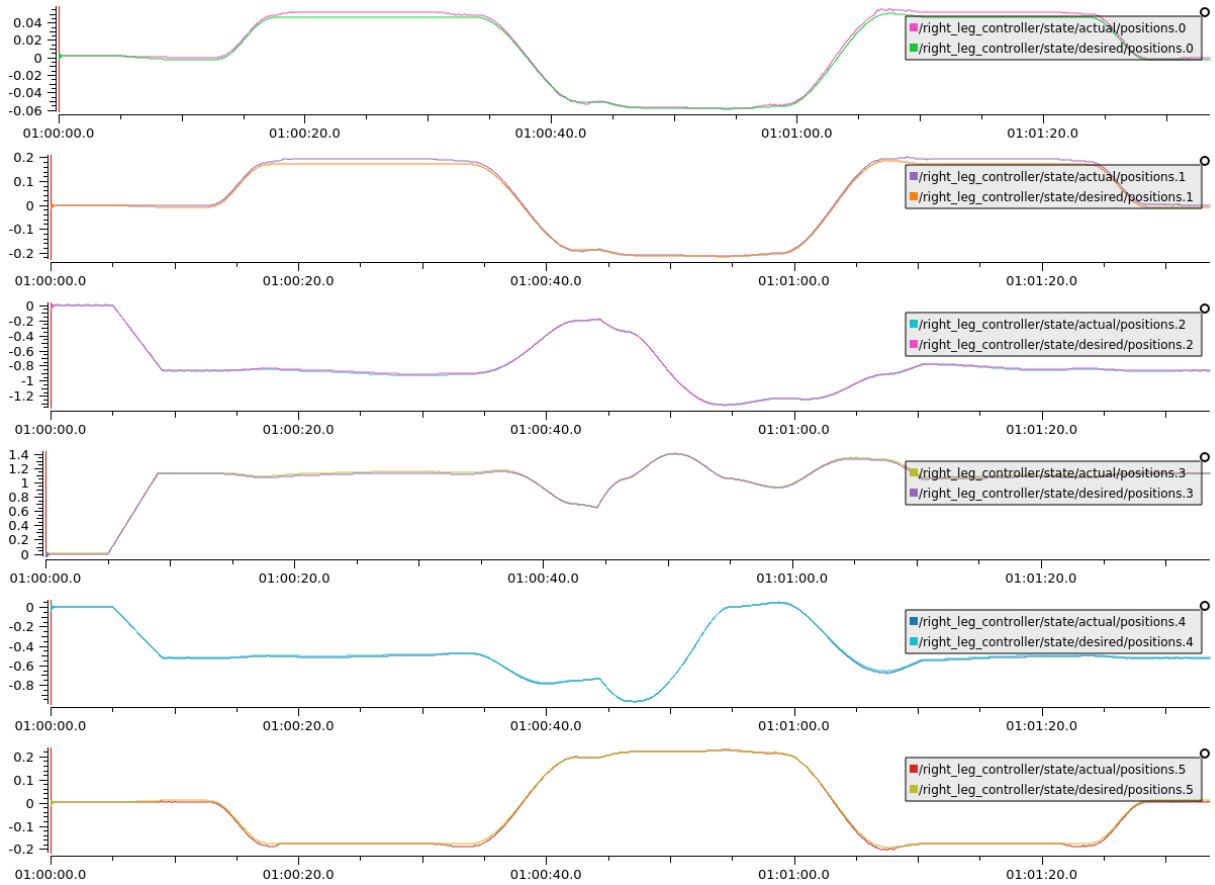


Figure 4.22: Joint angles[rad] vs Time[sec] for 4 cm right leg desired vs actual trajectories on simulation.

4.4.3 6 cm Stairs

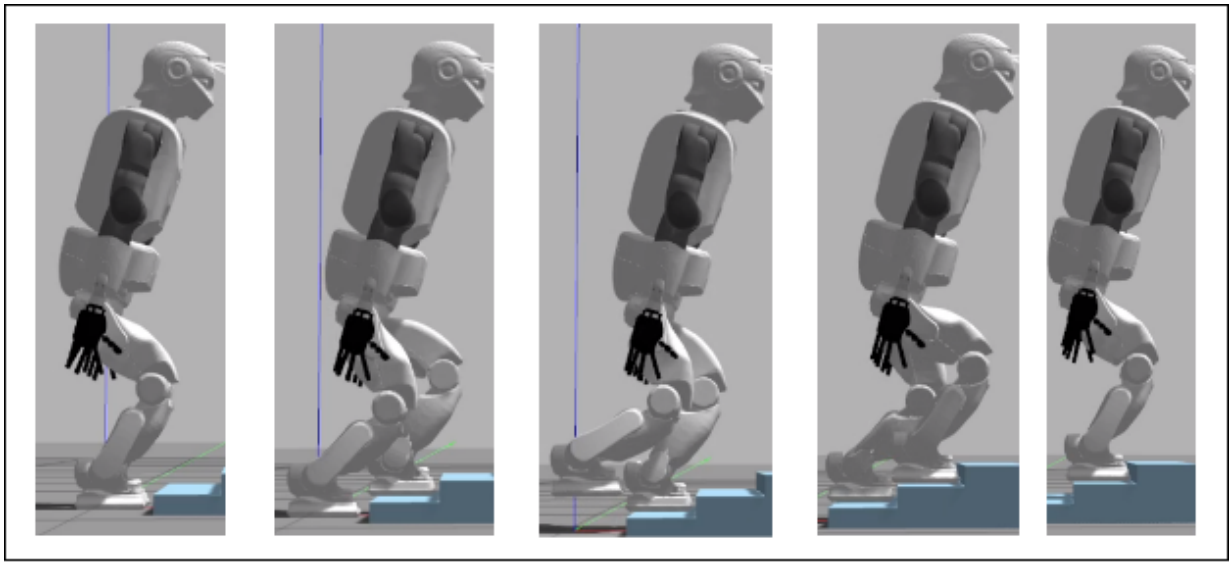


Figure 4.23: 6 cm motion on stairs in simulation.

Table 4.15 shows the parameters used to generate the motion for the 6 cm stairs. The trajectories of such said motion can be seen in Figure 4.24 and Figure 4.25.

Table 4.15: Parameters for 6 cm stairs in simulation.

Parameter	
Stair Height	0.06 m
Stair Width	0.5 m
No. of steps	2 steps
Step Height	0.075 m
Step Drop	0.02 m
Step Length	0.22 m
Step Width	0.16 m
COM Z position	0.76 m
t_step	50 sec

Trajectories

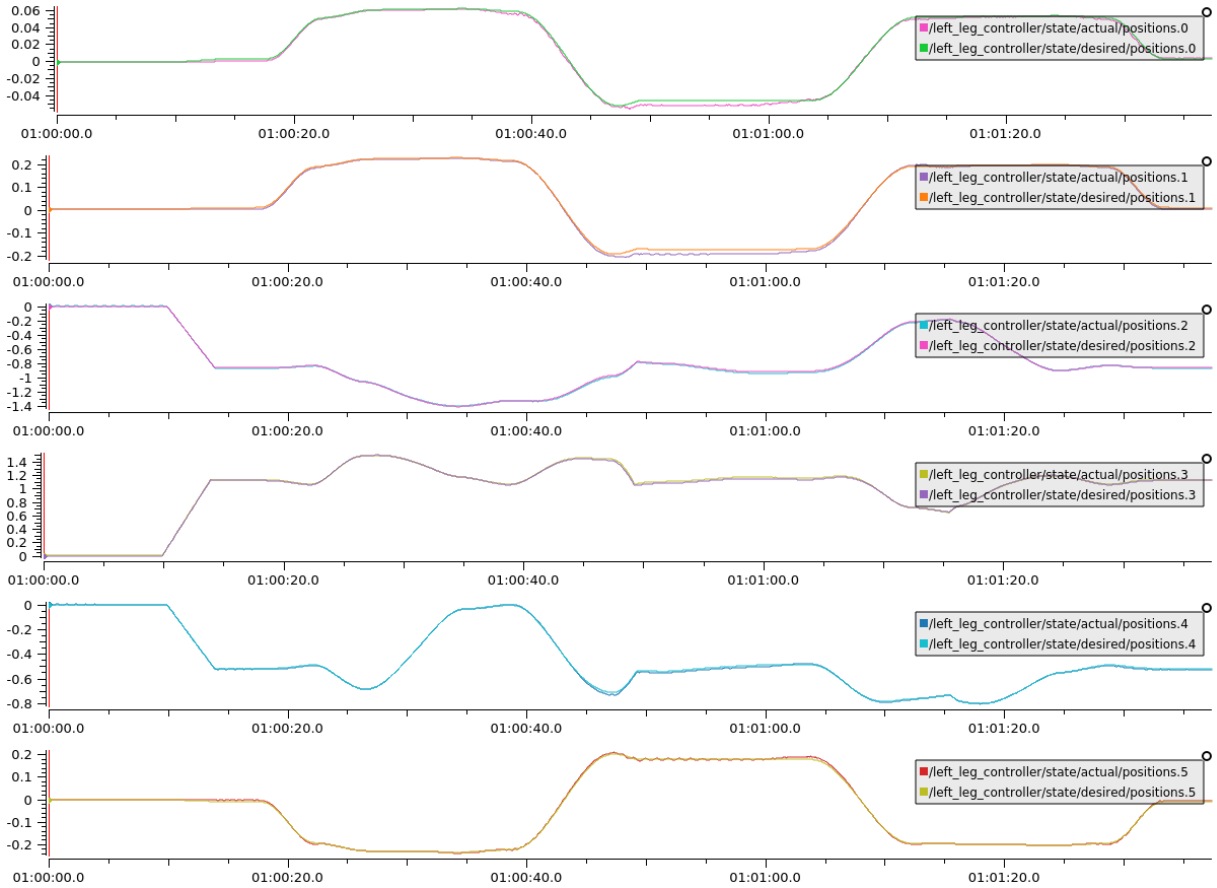


Figure 4.24: Joint angles[rad] vs Time[sec] for 6 cm left leg desired vs actual trajectories on simulation.

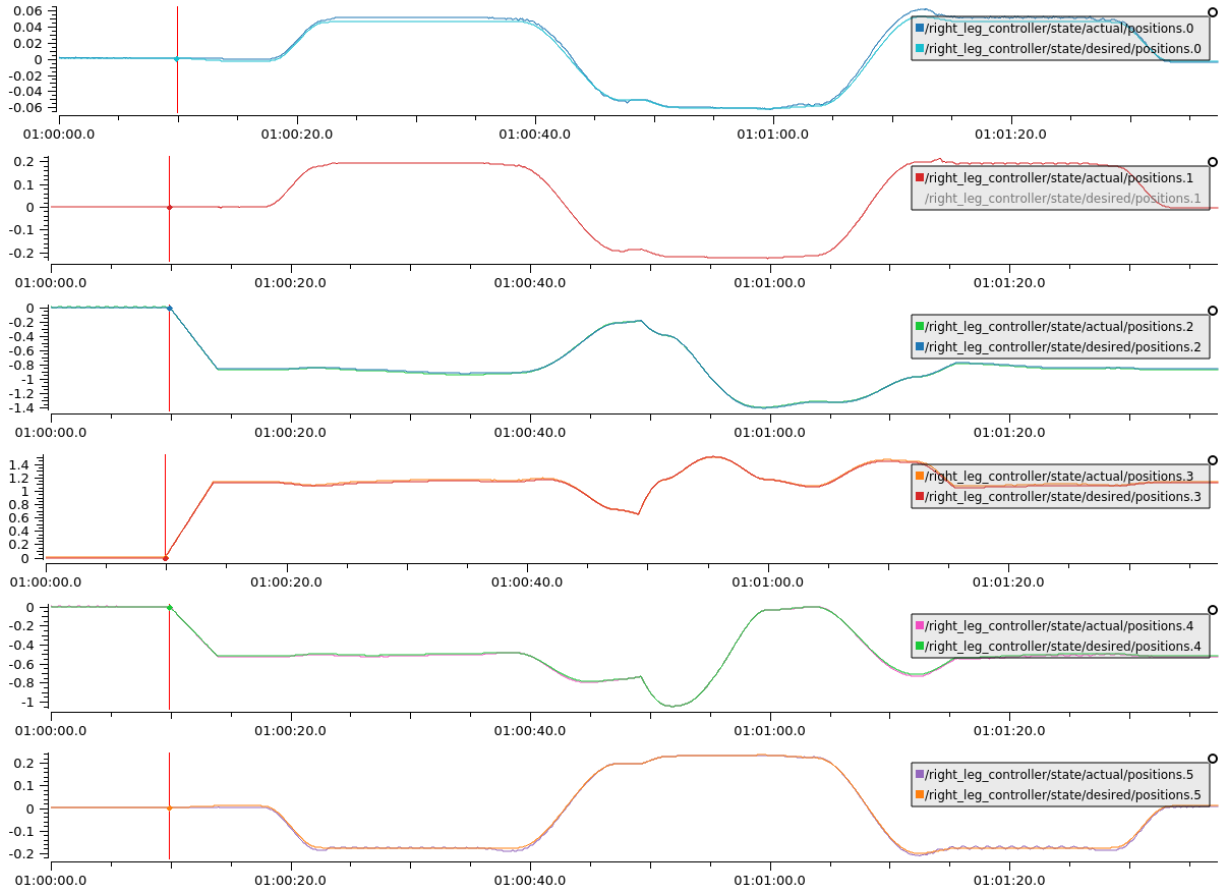


Figure 4.25: Joint angles[rad] vs Time[sec] for 6 cm right leg desired vs actual trajectories on simulation.

4.4.4 8 cm Stairs

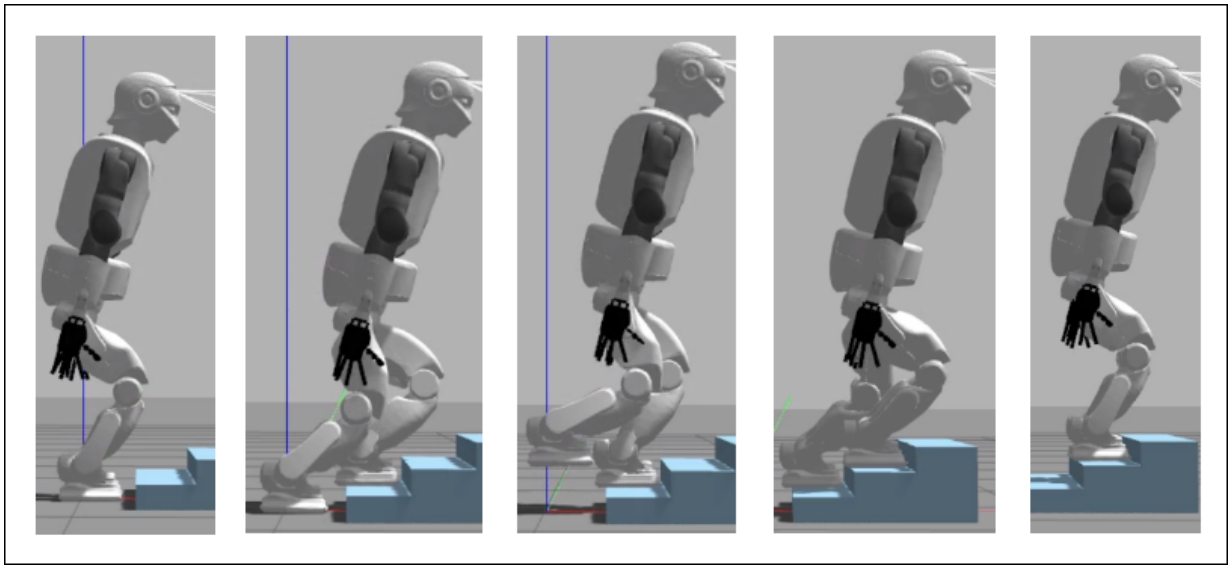


Figure 4.26: 8 cm motion on stairs in simulation.

Table 4.16 shows the parameters used to generate the motion for the 8 cm stairs. The trajectories of such said motion can be seen in Figure 4.27 and Figure 4.28.

Table 4.16: Parameters for 8 cm stairs in simulation.

Parameter	
Stair Height	0.08 m
Stair Width	0.5 m
No. of steps	2 steps
Step Height	0.095 m
Step Drop	0.02 m
Step Length	0.22 m
Step Width	0.16 m
COM Z position	0.76 m
t_step	50 sec

Trajectories

As seen from Fig.4.27 and Fig.4.28, the joint angle tracking done with the robot controller resulted in a well-tracked trajectory which allowed the robot to perform the motion successfully. The differences between the actual and desired trajectories are due to vibrations and inaccuracies that can be found in real-life equipment.

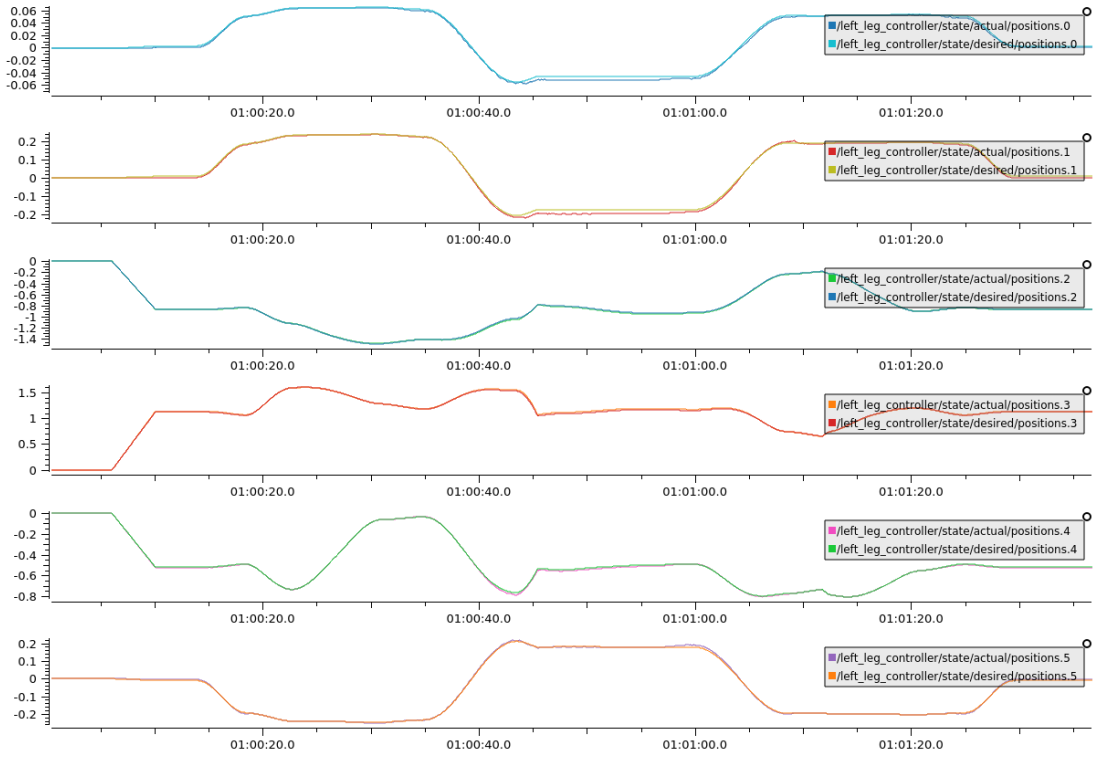


Figure 4.27: Joint angles[rad] vs Time[sec] for 8 cm left leg desired vs actual trajectories on simulation.

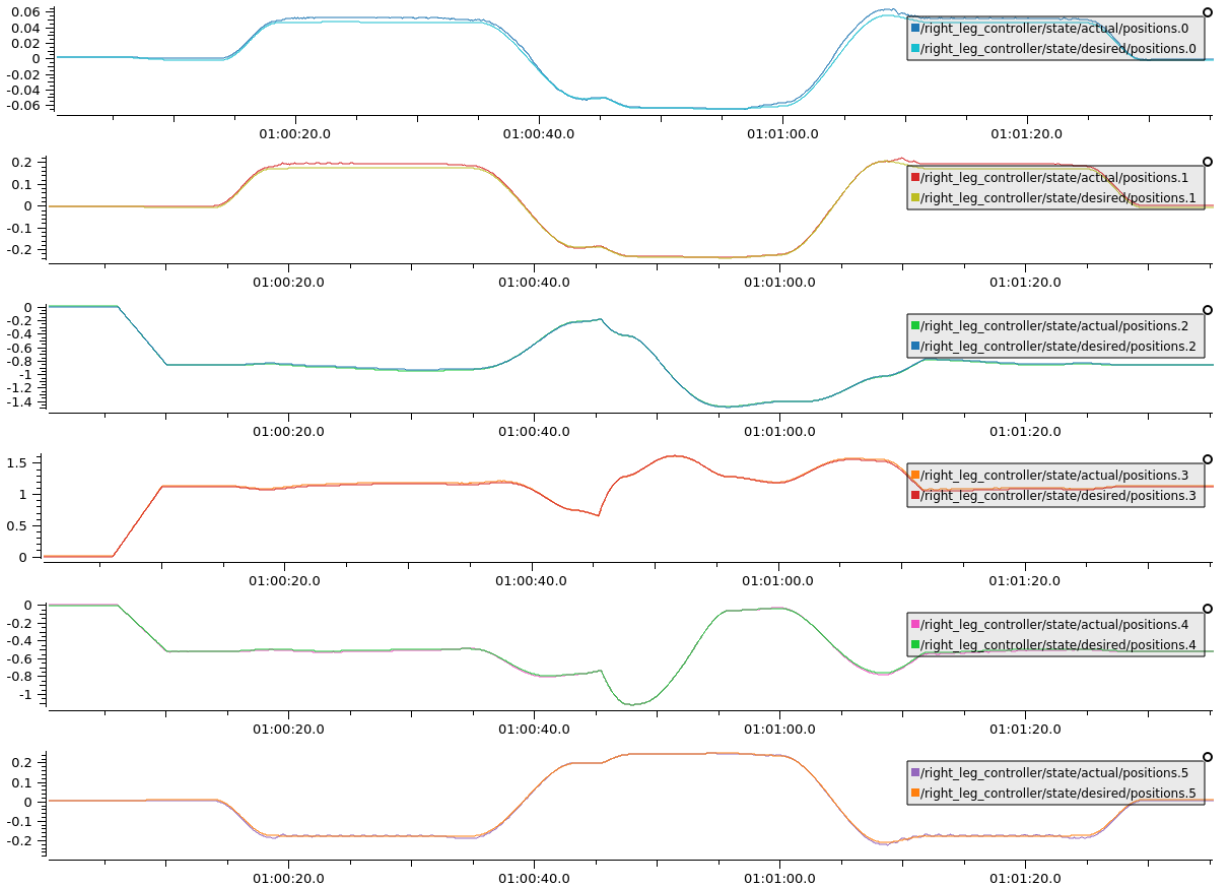


Figure 4.28: Joint angles[rad] vs Time[sec] for 8 cm right leg desired vs actual trajectories on simulation.

4.4.5 10 cm Stairs

The implementation of these stairs went through a number of iterations in which the robot model was in the update phase for the initial experiments and the model of the stairs was trivial.

For this trial, the robot's motion was planned to execute on 2 stair steps. As seen in Figure 4.29 the robot was able to perform the motions with ease on the basic stair model.

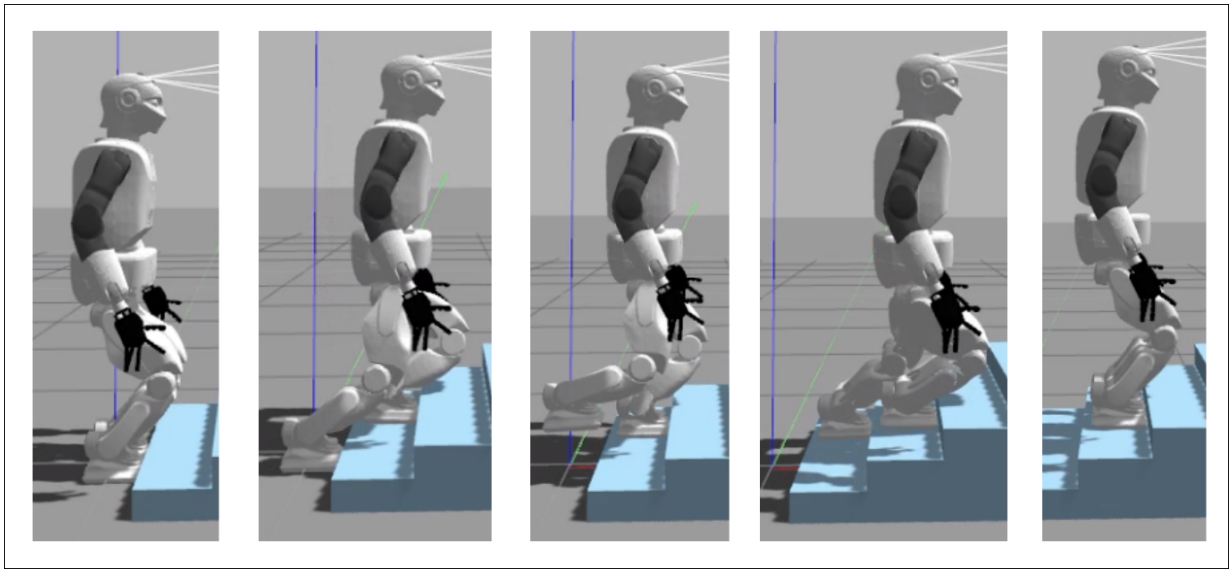


Figure 4.29: 10 cm motion on stairs with no lip.

After a revision regarding both the robot model and the stair model, the robot's model was updated [Ch. 3], and an update to the stairs took place as well to model the real steps that we received. An extra lip was added which would introduce a challenge for the robot as it needs to clear off from the lip during the lift-off phase, which can be seen in Fig.4.30.

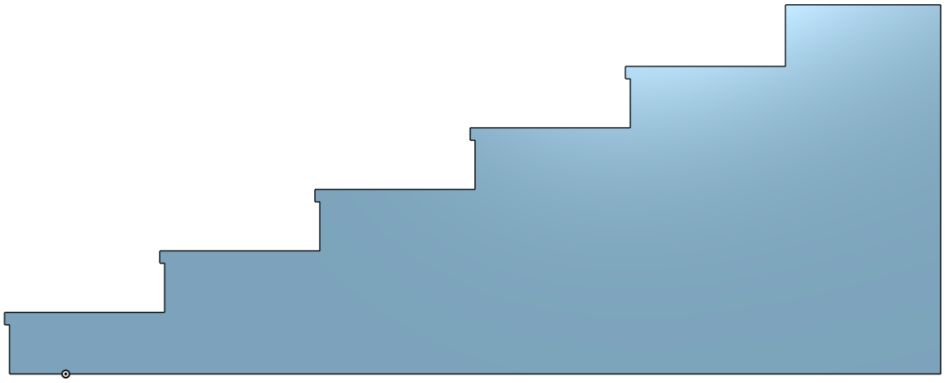


Figure 4.30: Updated stairs model with lip.

Performing the experiments with these new set of changes produced some challenges, the first being an extra mass at the rear of the hip of the robot which would cause it to tilt backward during the motion resulting in the robot losing its balance and falling backward. This issue has been resolved by tilting the torso_1 joint forward making the robot more bent over to the front which shifted the COM and COP near the center of the support polygon rendering the robot more stable. This tilt can be seen in the generated motion in Fig.4.31.

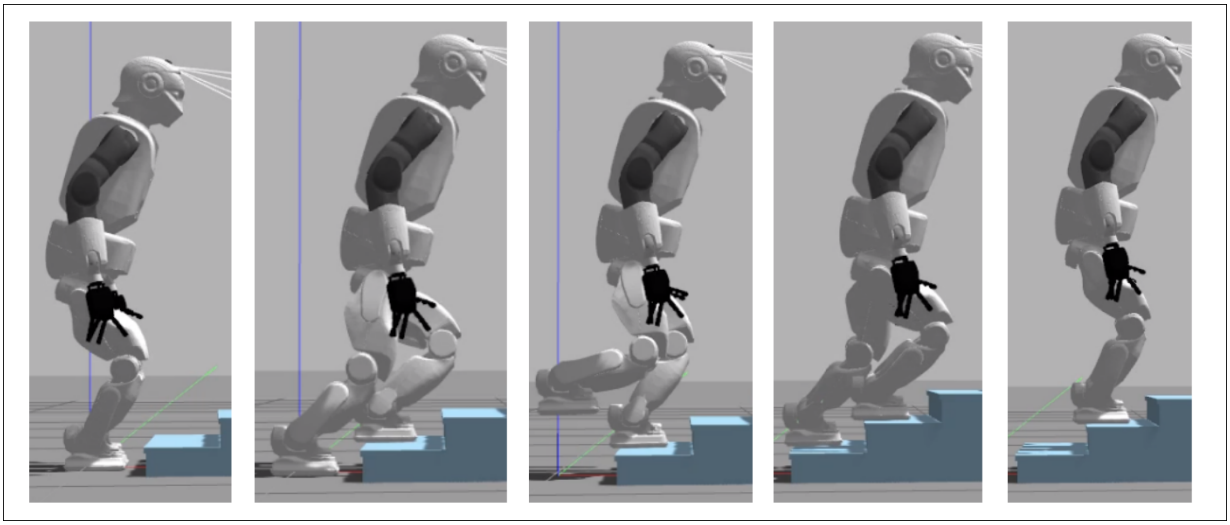


Figure 4.31: 10 cm motion on stairs with lip and updated model

In addition, due to the presence of the lip, the exact placement of the robot in front of the stairs is deemed to be crucial. Placement of the robot close to the stairs would result in the feet being caught in the lip and hindering the robot's motion. On the other hand, placing it far from the stairs would result in the robot's step being too big which would result in extra torques on the joints. Placing it too close requires smaller steps which would result in the feet being on the edge of the stair and the robot losing its balance.

The ideal distance was found to be 30 cm from the rear of the foot, allowing the robot to perform the necessary steps without any difficulties during the motion. In addition, the pelvis height was increased to ensure that the shank of the robot would not collide with the stairs.

4.4.6 Summary

The simulations of the robot were done on Gazebo to introduce the physics to the simulation and introduce the interactions between the robot and the stairs. As can be seen from the figures above, the robot was able to execute the generated motions successfully for the 2,4,6,8 and 10 cm stairs. The trajectories displayed above indicate that the robot was able to successfully track the desired joint angles to perform the motion, although the torques were reaching their limits on some of the joints as can be seen in Table 4.17. A discovery was made during the simulation: generating the motions to be executed in a faster manner always led to the robot tilting forward and colliding with the stairs, as Gazebo was introducing unnecessary noise in the attempt to mimic real-life behaviours.

As expected the knee joint and the ankle joint i.e joint_4 and joint_6 respectively, are contributing a lot of torque to perform the motion. As can be seen in Table 4.17, as the stair height increases a significant increase in these joint torques can be seen, with the joint_2 and joint_6 reaching their maximum for both 8 and 10 cm stairs, indicating that the 10 cm stairs would be difficult to perform in real-life as a number of joints are reaching their limits and may cause damage to the motors in real-life. Finally, for increasing the stair height it seemed that the exact placement of the robot is crucial especially for the 10 cm stairs as it needs some distance to perform the clearance during the lift-off.

As seen in the Table 4.17, the torques in both the right and left foot are not identical, although the motion is symmetric. This difference is due to the COM not being directly in the middle of the pelvis joint, producing close but not exact torques. The torque peaks were produced during the lift-off phase, as the robot was applying all its force to push itself upward.

Table 4.17: Maximum torques produced by the REEM-C in simulation during different stair walking heights

Stair Height	Joint Name	Max Torque (Nm)	Joint Name	Max Torque (Nm)
	Joint 1 right	24.30	Joint 1 left	16.68
	Joint 2 right	63.27	Joint 2 left	63.82
	Joint 3 right	55.16	Joint 3 left	57.23
	Joint 4 right	74.59	Joint 4 left	75.39
	Joint 5 right	56.63	Joint 5 left	52.31
2 cm	Joint 6 right	58.60	Joint 6 left	66.17
	Joint 1 right	17.11	Joint 1 left	16.23
	Joint 2 right	63.42	Joint 2 left	64.05
	Joint 3 right	55.67	Joint 3 left	55.38
	Joint 4 right	78.45	Joint 4 left	76.38
	Joint 5 right	55.21	Joint 5 left	53.26
4 cm	Joint 6 right	60.16	Joint 6 left	64.55
	Joint 1 right	26.58	Joint 1 left	16.61
	Joint 2 right	64.26	Joint 2 left	64.18
	Joint 3 right	56.08	Joint 3 left	62.11
	Joint 4 right	88.14	Joint 4 left	81.59
	Joint 5 right	57.51	Joint 5 left	54.82
6 cm	Joint 6 right	71.48	Joint 6 left	71.48
	Joint 1 right	29.56	Joint 1 left	19.05
	Joint 2 right	71.40	Joint 2 left	71.40
	Joint 3 right	57.66	Joint 3 left	62.9
	Joint 4 right	111.65	Joint 4 left	107.09
	Joint 5 right	69.02	Joint 5 left	71.92
8 cm	Joint 6 right	71.48	Joint 6 left	71.48
	Joint 1 right	22.11	Joint 1 left	25.38
	Joint 2 right	71.40	Joint 2 left	71.40
	Joint 3 right	58.62	Joint 3 left	61.11
	Joint 4 right	117.5	Joint 4 left	122.69
	Joint 5 right	62.67	Joint 5 left	69.85
10 cm	Joint 6 right	71.48	Joint 6 left	71.48

4.5 Real-life Experiments

The next step was to implement the tested trajectories on the humanoid robot in real-life and validate that they can be executed correctly. In addition, this would verify that the model was reflecting real-life behaviour. In all of the experiments done the CoM_z was fixed and the stair parameters as well as the step parameters were changed to accommodate for the stair dimensions.

4.5.1 2 cm Stairs

Initially, the robot was tested to perform the stair climbing motion on the 2 cm stairs only for 2 steps. Afterwards, an extension of this experiment was implemented on 4 steps using the same parameters to generate the motion as seen in Table 4.18.

Table 4.18: Parameters for 2 cm stairs.

Parameter	
Stair Height	0.02 m
Stair Width	0.5 m
No. of steps	4 steps
Step Height	0.03 m
Step Drop	0.01 m
Step Length	0.25 m
Step Width	0.16 m
COM Z position	0.76 m
t_step	25 sec
t_sample	0.02 sec

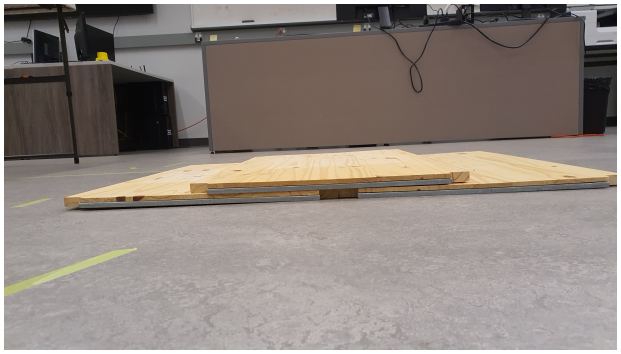


Figure 4.32: Stairs 2 cm.

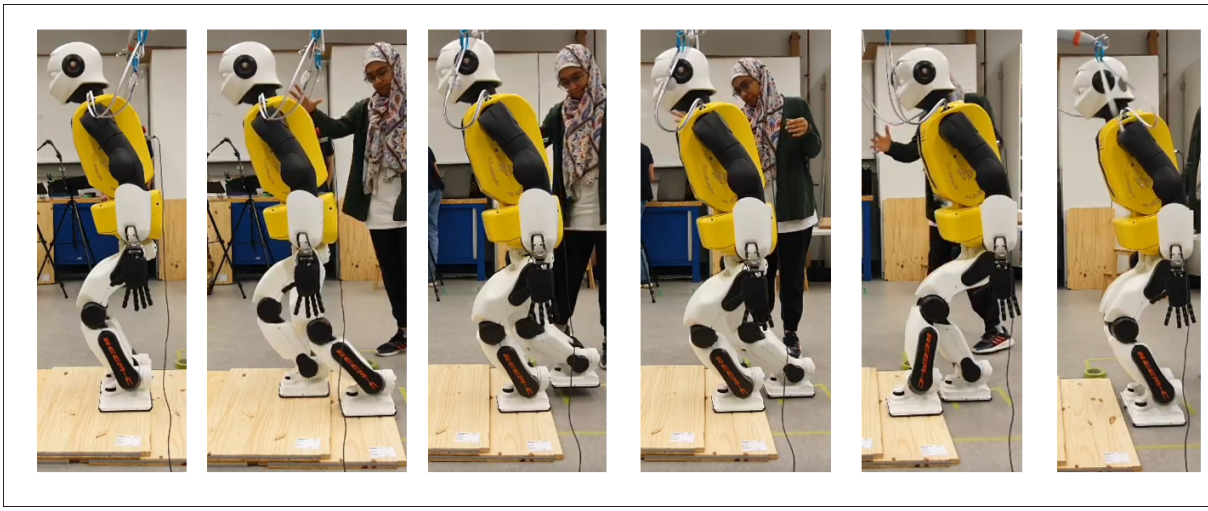


Figure 4.33: Steps of 2 cm stair walking.

During the execution on the real robot, an issue occurred where the rate of the data being passed to the robot was out of sync with the controller frequency. This resulted in the robot executing the motion in a jerky manner and introduced lots of disturbances to the motion itself. Although the motion was executed successfully, the jerky motion was an issue that needed to be handled to provide smoother trajectories. This was achieved by changing the sampling rate from 25 Hz to 50 Hz in an attempt to sync the readings with the refresh rate of the controller. As anticipated the motion became smoother and was performed without jerkiness.

The initial tests had the robot execute one step i.e t_step, in 50 sec resulting in the whole motion to be performed in around 2 min. This t_step was implemented as a guideline

from the instructions provided by Sebastian Sontag to ensure that the robot's motion is stable and feasible. In an attempt to provide a more human-like execution time, the t_step was decreased to 25 sec which reduced the time from 2 min to 1 min, resulting in a more realistic motion that still ensured stability and accuracy.

Torques

As mentioned earlier, the torques are being generated using the filtered motor current measurements. The torques exerted by the robot's joints can be seen in Fig. 4.34 and 4.35. As observed, the maximum torques exerted by the robot during the motion are within the torque limits, ensuring that the robot is performing the motion without exerting any critical torques that may damage the joints.

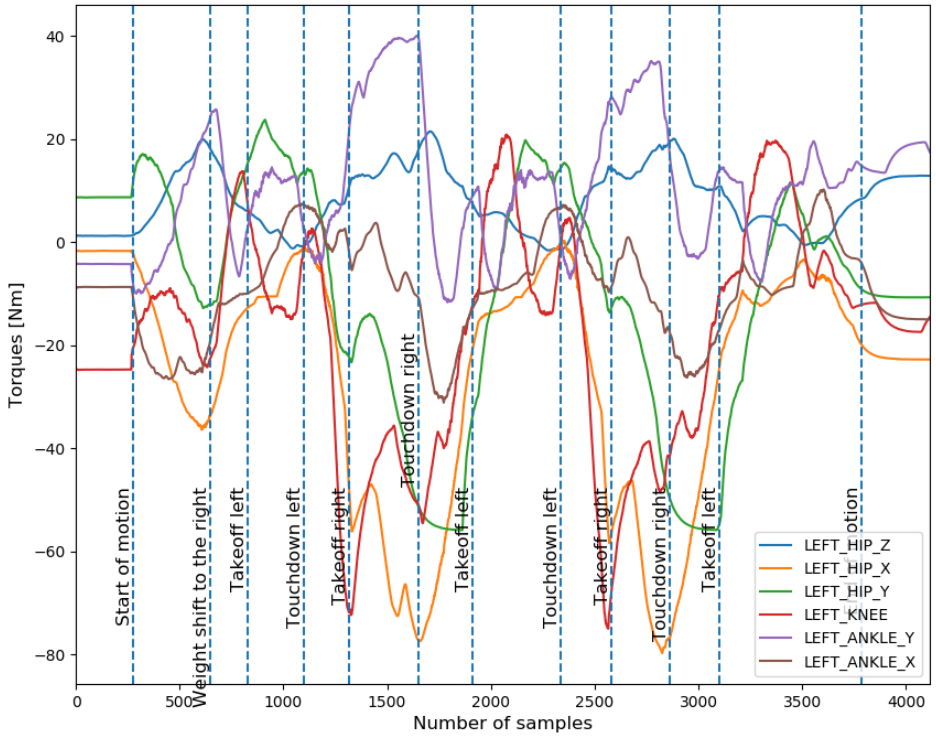


Figure 4.34: Torques produced by the left foot throughout the motion on the 2 cm stairs.

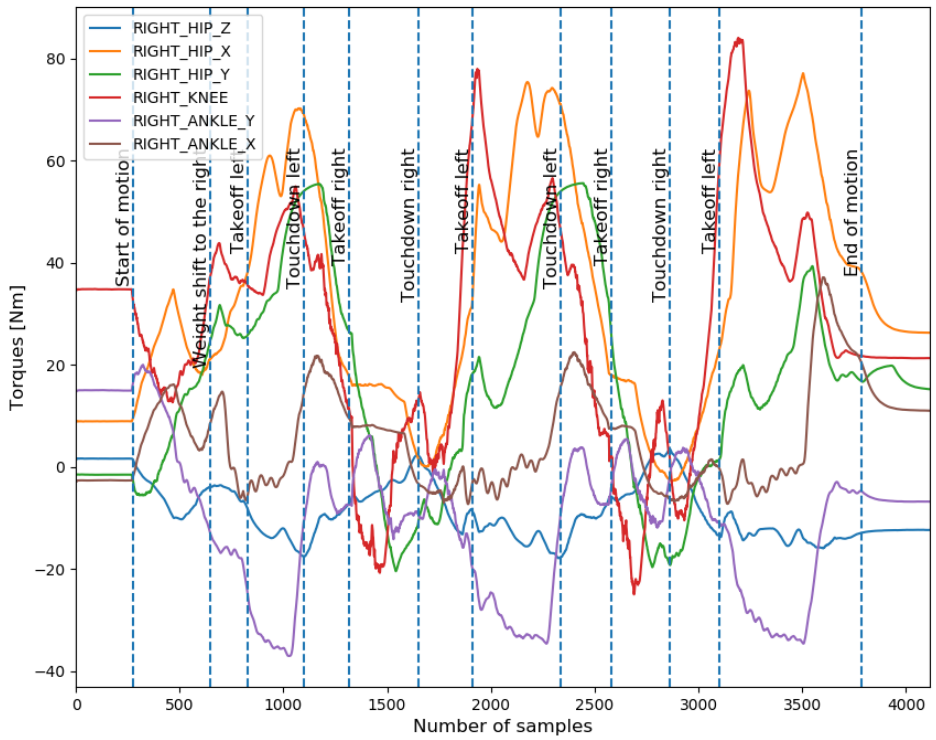


Figure 4.35: Torques produced by the right foot throughout the motion on the 2 cm stairs.

4.5.2 4 cm Stairs

As mentioned earlier, the 2 cm shelves can be seen in Figure 4.36 being stacked over each other to make the 4 cm steps for the robot to climb on them. The parameters that were used in the execution of this motion can be seen in Table 4.19.

Table 4.19: Parameters for 4 cm stairs.

Parameter	
Stair Height	0.04 m
Stair Width	0.5 m
No. of steps	2 steps
Step Height	0.055 m
Step Drop	0.02 m
Step Length	0.25 m
Step Width	0.16 m
COM Z position	0.76 m
t_step	25 sec



Figure 4.36: Stairs 4 cm.

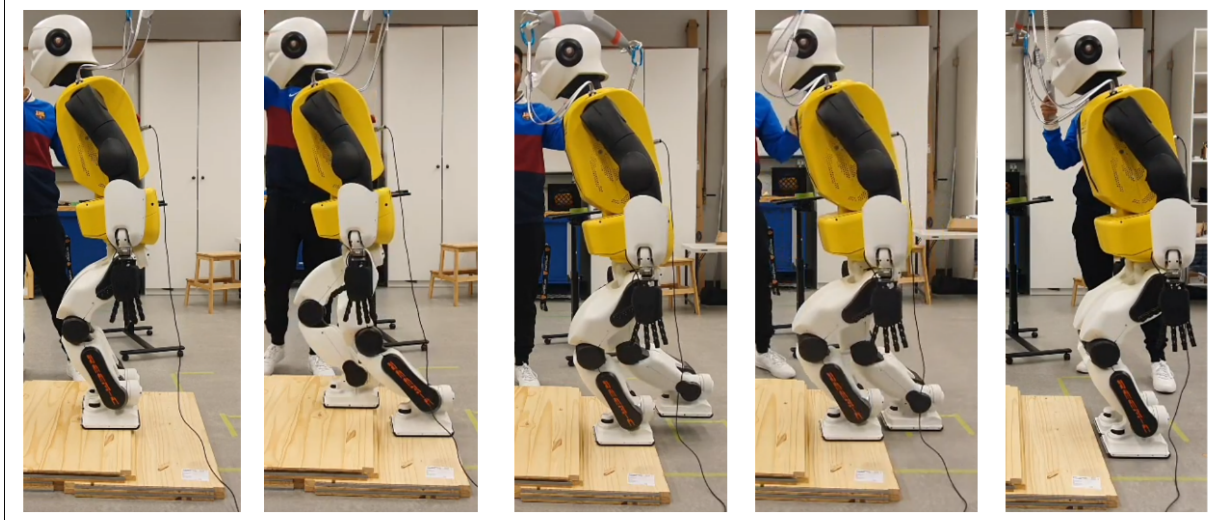


Figure 4.37: Steps of 4 cm stair walking.

During the execution of the motion, it was noticed that the robot was pushing the stairs forward during the weight shift motion which can be possible due to the light weight of the stairs. This issue was solved by providing a weight in front of the stairs to make them more rigid, and the motion was still executed successfully without any issues.

Torques

The torques exerted by the robot's joints can be seen in Fig. 4.38 and 4.39.

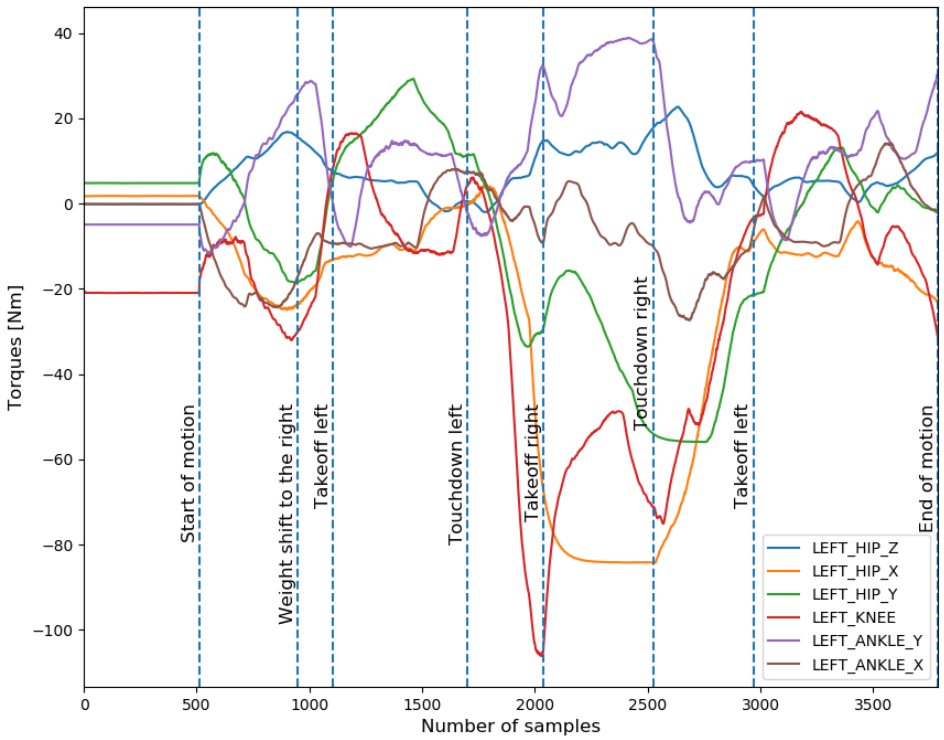


Figure 4.38: Torques produced by the left foot throughout the motion on the 4 cm stairs.

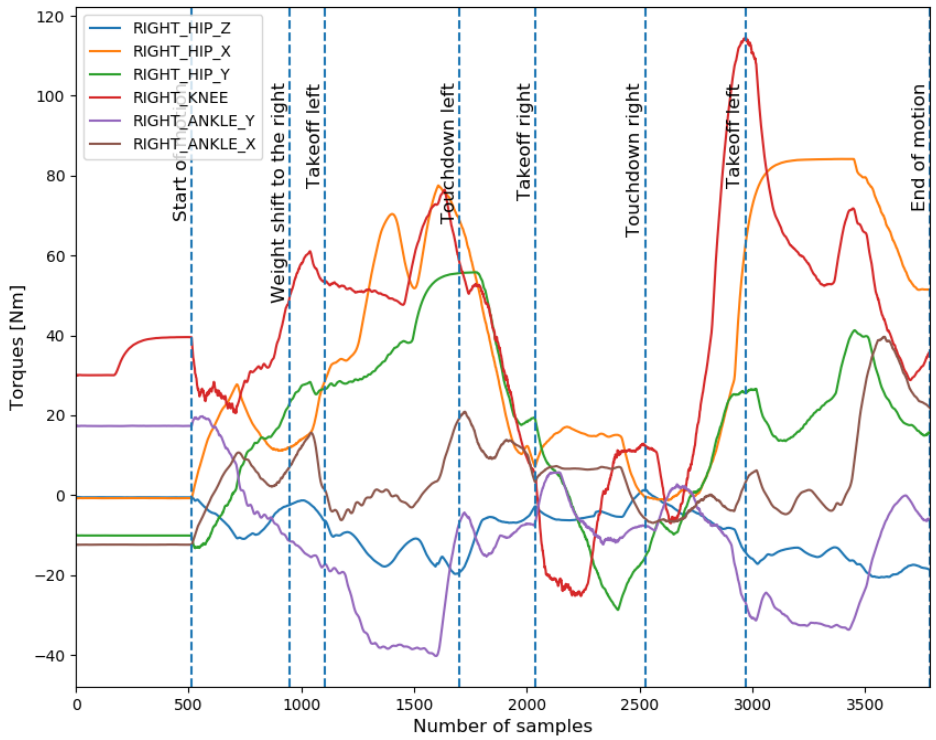


Figure 4.39: Torques produced by the right foot throughout the motion on the 4 cm stairs.

4.5.3 6 cm Stairs

The parameters that has been used to generate the motion can be found in Table4.20. The stair assembly can be seen in Fig 4.40, this was the configuration at which the robot attempted its motion.

Table 4.20: Parameters for 6 cm stairs.

Parameter	
Stair Height	0.06 m
Stair Width	0.5 m
No. of steps	2 steps
Step Height	0.07 m
Step Drop	0.02 m
Step Length	0.25 m
Step Width	0.16 m
COM Z position	0.76 m
t_step	25 sec



Figure 4.40: Stairs 6 cm.

When attempting this stair height the robot's rear foot would tilt during the lift-off phase, as the robot itself would be tilting at the initial part of the motion. This would

be due to the COP of the robot being shifted at the back border of the support polygon, which could have resulted in the robot being unstable. Although this was happening the robot was able to perform the motion successfully, and the results from this attempt are discussed later.



Figure 4.41: Foot tilt during liftoff.

An adjustment to the robot's posture has been made. The arms of the robot have been shifted forward slightly and the experiment was then repeated. This slight adjustment seemed to solve the issue during the lift-off and the motion was executed successfully without any collisions occurring with the stairs. The results from the experiment can be seen in the following section.

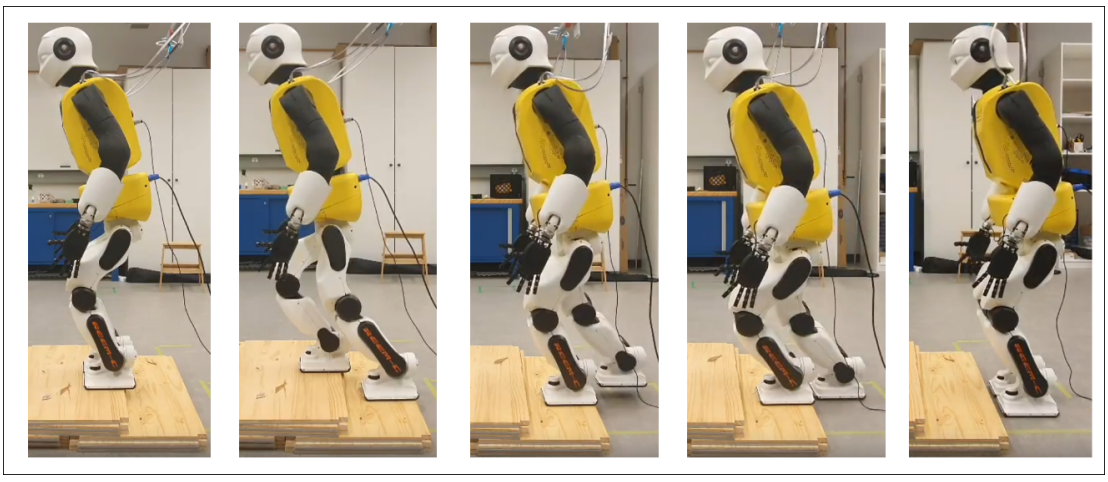


Figure 4.42: Steps of 6 cm stair walking.

Torques

The torques exerted by the robot's joints can be seen in Fig. 4.43 and 4.44. As observed, the maximum torques exerted by the robot are within the torque limits. This ensures that the robot is performing the motion in a safe manner and not reaching any critical points that could be cause for concern.

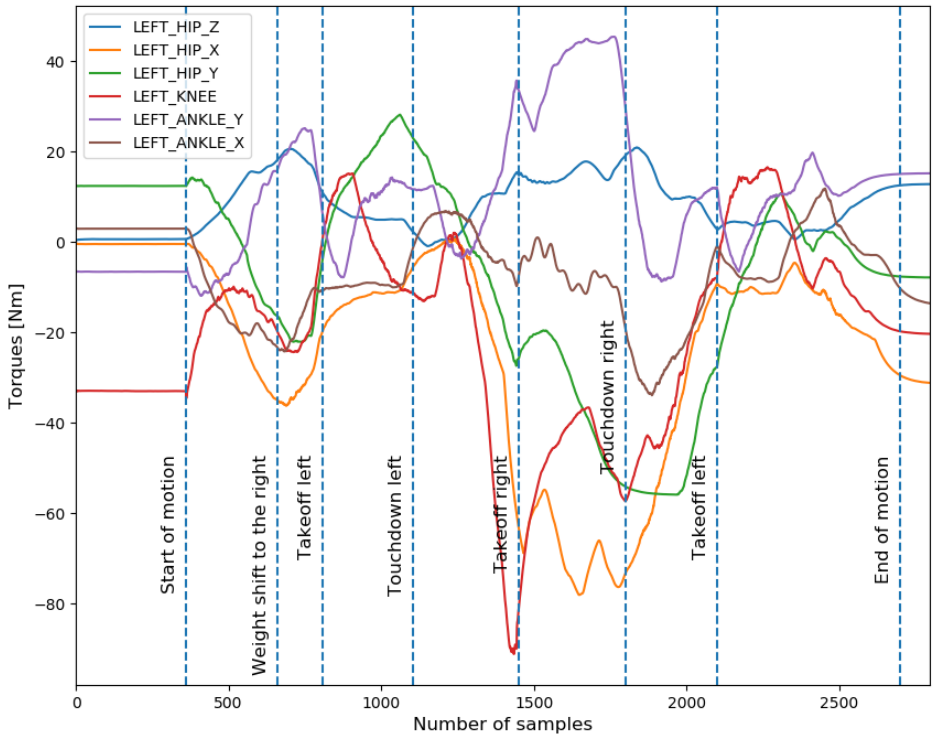


Figure 4.43: Torques produced by the left foot throughout the motion on the 6 cm stairs.

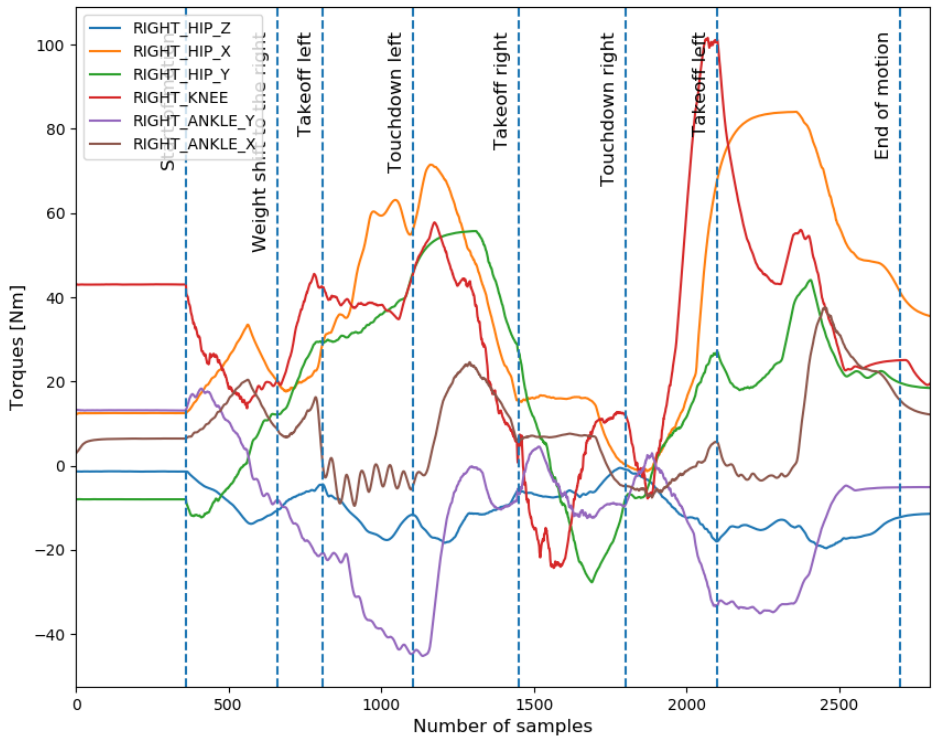


Figure 4.44: Torques produced by the right foot throughout the motion on the 6 cm stairs.

4.5.4 8 cm Stairs

Due to the limitation of the available shelves, performing 8 cm stairs with this configuration was the maximum height to be implemented. The parameters that were used during the generation of the current motion can be seen in Table 4.21.

Table 4.21: Parameters for 8 cm stairs.

Parameter	
Stair Height	0.08 m
Stair Width	0.5 m
No. of steps	2 steps
Step Height	0.09 m
Step Drop	0.02 m
Step Length	0.25 m
Step Width	0.16 m
COM Z position	0.76 m
t_step	25 sec

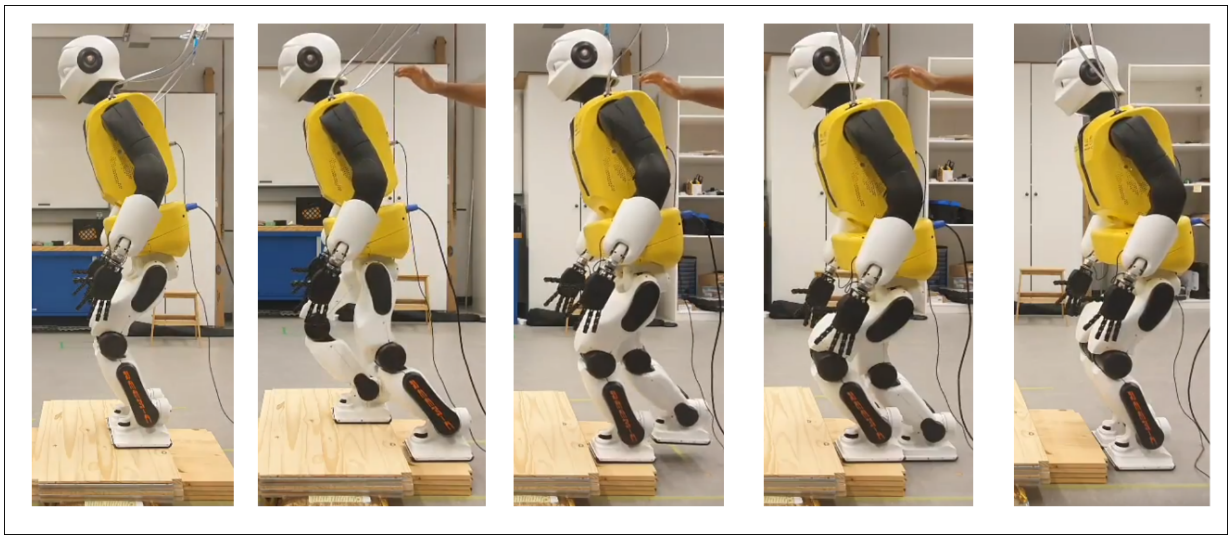


Figure 4.45: Steps of 8 cm stair walking.

Torques

The torques exerted by the robot's joints can be seen in Fig. 4.46 and 4.47. Once more, it can be seen that the maximum torques exerted by the robot are within the torque limits. It is also noticed that joint_2 in Fig.4.47 had a spike that is very close to the limits of that

joint, indicating that the experiments for this implementation are reaching the maximum capabilities of the robot.

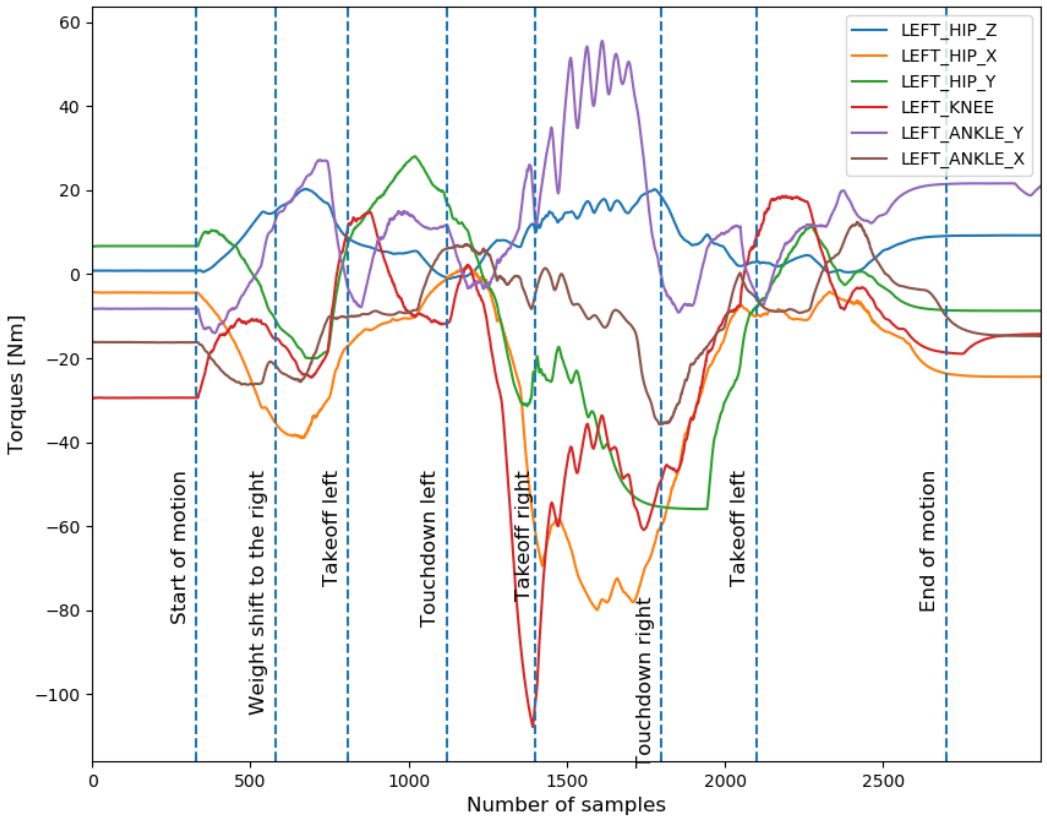


Figure 4.46: Torques produced by the left foot throughout the motion on the 8 cm stairs.

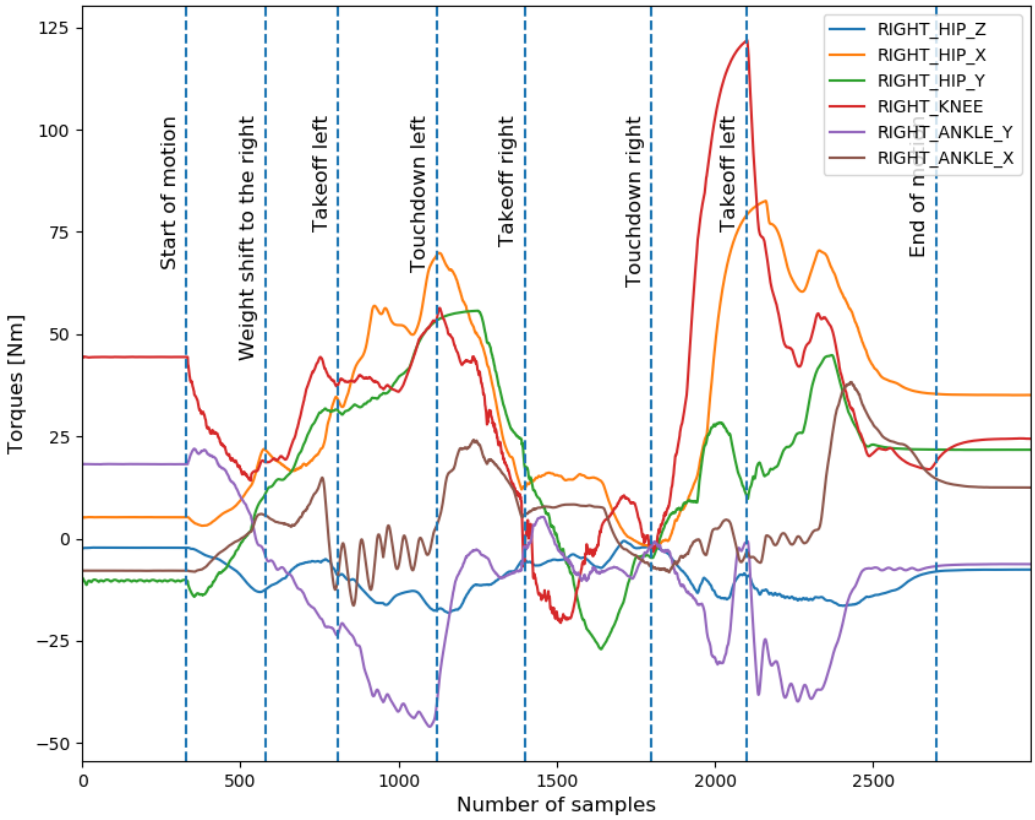


Figure 4.47: Torques produced by the right foot throughout the motion on the 8 cm stairs.

4.5.5 10 cm Stairs

In the implementation of this motion, the shank of the robot was quite close to the stairs during the weight-shifting phase, raising concerns that the robot may collide with the stairs, which can be seen in Fig 4.48. This has led to the increase of the com_z from 0.74 m to 0.76 m as it was the highest height the IK solver would provide a solution for. This action helped with the shank gaining better clearance with regard to the next step.

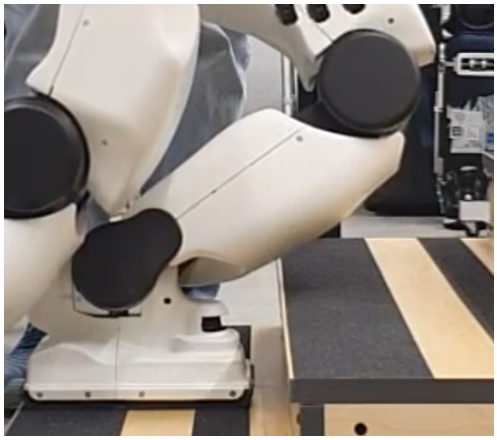


Figure 4.48: Shank too close to the step.

Another observation was that the robot was finding difficulties lifting off. The feet would collide with the lip of the stairs, as shown in Fig 4.49 during lift-off. During the initial experiments, the foot collided with the stairs causing the force sensors to malfunction and rendering the robot unusable for 6 months, delaying all other tests on it. This confirmed that the robot model we had for simulation was not accurate to reflect these critical behaviours, leading to the update process of the model mentioned here chapter 3.



Figure 4.49: The foot being stuck in the lip.

4.5.6 Summary

Table 4.22 shows the torque produced by the robot when performing the various stair heights for the stair-walking motion. As can be seen from the table, the torques increase as the stair height increases. This relationship explains why the motion on the 10 cm stairs was not successful, as these required torques are beyond the robot's capabilities.

The simulation was carried out with different stair heights ranging from 2.0 cm to 8.0 cm, and the humanoid was able to successfully climb them without changing posture or perturbation to the motion. This shows that the calculations for the motion are reliable. During implementation in real life, the faster the motion performed, the more stable the robot was. The speed was also drastically improved from the previous implementation at the University of Heidelberg. Some issues were encountered, for instance, the ankle of the rear foot would tilt due to the higher torques needed. Unfortunately, the 10.0 cm execution was not reproduced on the real robot as it required higher torques which the robot could not provide.

As can be seen from the torque figures above, the knee joint (leg_joint_4) and hip joint (leg_joint_2) have the highest torques, and the motion needs to be modified in an attempt to lower the stress on these joints. In addition, it can be seen that during take-off, the adjacent leg_joint_4 would be exerting high torques to achieve the lift-off of the robot. In the 8 cm stairs test, it can be seen that the torques are near the torque limits of the joint, hindering it from producing additional torque, which is why the robot was unable to perform the 10 cm stair climbing.

Table 4.22: Maximum Torques produced by the REEM-C in real-life during different stair walking heights.

Stair Height	Joint Name	Max Torque (Nm)	Joint Name	Max Torque (Nm)
	Joint 1 right	17.87	Joint 1 left	21.52
	Joint 2 right	77.16	Joint 2 left	79.73
	Joint 3 right	55.65	Joint 3 left	55.87
	Joint 4 right	84.10	Joint 4 left	74.98
	Joint 5 right	37.04	Joint 5 left	40.09
2 cm	Joint 6 right	37.18	Joint 6 left	31.13
	Joint 1 right	20.62	Joint 1 left	22.70
	Joint 2 right	84.18	Joint 2 left	84.19
	Joint 3 right	55.82	Joint 3 left	55.91
	Joint 4 right	114.50	Joint 4 left	106.10
	Joint 5 right	40.25	Joint 5 left	38.86
4 cm	Joint 6 right	39.67	Joint 6 left	27.34
	Joint 1 right	19.62	Joint 1 left	20.92
	Joint 2 right	84.04	Joint 2 left	78.02
	Joint 3 right	55.70	Joint 3 left	55.86
	Joint 4 right	101.60	Joint 4 left	91.11
	Joint 5 right	45.18	Joint 5 left	45.39
6 cm	Joint 6 right	37.59	Joint 6 left	33.89
	Joint 1 right	18.18	Joint 1 left	20.29
	Joint 2 right	82.60	Joint 2 left	79.90
	Joint 3 right	55.73	Joint 3 left	55.90
	Joint 4 right	121.92	Joint 4 left	107.76
	Joint 5 right	46.00	Joint 5 left	55.54
8 cm	Joint 6 right	38.28	Joint 6 left	35.75

Table 4.23: Implemented stair height in real-life.

Stair Height	Status
2 cm	Feasible
4 cm	Feasible
6 cm	Feasible
8 cm	Feasible
10 cm	Infeasible
15 cm	Infeasible

By observing the maximum torque readings produced by the simulation and real-life, we can conclude that the model in simulation provides a reliable outcome as the maximum torques produced in both real-life experiments and simulation are quite similar.

All of the previous experiments were conducted 5 times in real-life and the success rate for each stair height can be seen in Table:

Table 4.24: Success rate of the experiments over 5 repetitions.

Stair Height	Success Rate
2 cm	100%
4 cm	100%
6 cm	100%
8 cm	80%
10 cm	0%

4.5.7 Visualizer Implementations

It is also important to discuss the tool that was implemented to visualize the motion and helped during the debugging of the pipeline.

In the stages of testing the issue of not having any visual feedback of the recorded data on the robot arose, which led to the creation of a ROS package that would allow the visualization of these motions on **RVIZ** at anytime with the help of the **rqt bag publisher**. This implementation allowed us to visualize the robot without needing to

run the Gazebo simulation in the background. In addition, it would allow us to have a better understanding of the state of the humanoid at crucial moments during any executed motions.

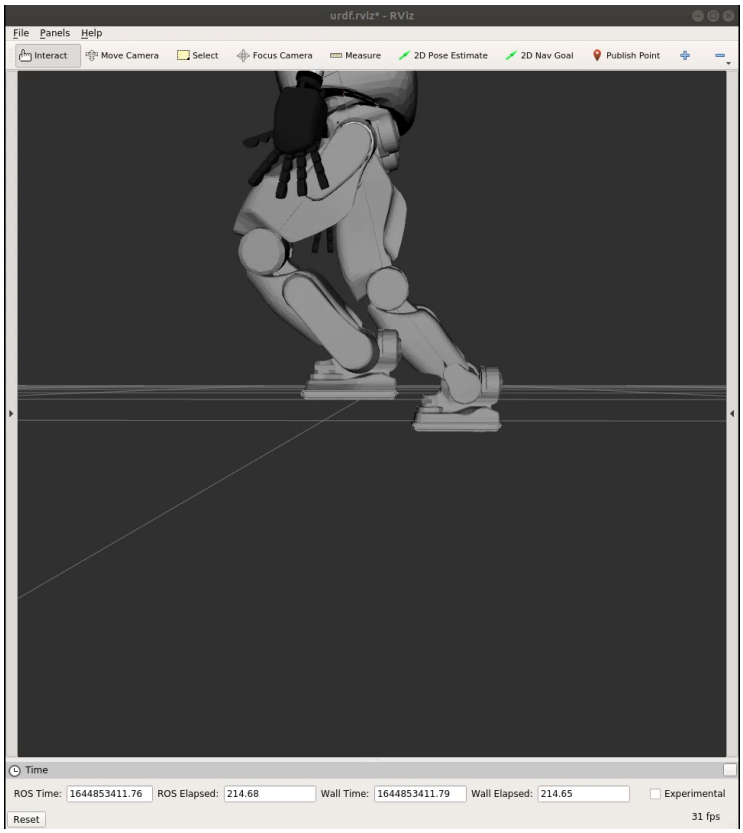


Figure 4.50: RVIZ visualization of the recorded motion from the robot during the experiment.

Chapter 5

Conclusion

This section will go through the conclusion reached from the previous chapter 4, as well as the limitations faced during the process and the future work that can be implemented according to the results and the robot's capabilities under the current circumstances.

Implementing the whole pipeline in C++ provided a faster computation time for the pre-planning by 70% compared to the Python implementation performed by the University of Heidelberg, with the C++ time running at 1.5 sec/step and the Python implementation running at 5 sec/step. In addition, the time of step executed on the robot was faster, having been reduced from 60 sec/step to 25 sec/step in our implementation.

Although the kinematics of the robot can reach 20 cm stairs, the torques required for such configuration would be exceeding the torque limits.Once implemented on simulation the joint would be reaching their joint torque limits at 10 cm implementation, therefore the simulation was only conducted for the 2, 4, 6, 8 and 10 cm stairs. The robot could perform the motions successfully and reach the end pose. It can be seen that the robot was able to perform the stair walking task for the 2 and 4 cm stairs without any difficulty, and from the torque readings as well as the trajectory readings, the robot values were within the operating bounds with a safe margin. Starting from the 6 cm stairs in simulation, the robot was beginning to hit the torque limits in one of the joints, specifically the ankle joint, i.e. joint_6; this was suspected to be due to disturbances as it happened instantaneously for this motion. When the experiment was reflected on the real robot, the first observation was that the robot was tilting backward, causing the feet to collide with the stairs. This issue was solved by moving the arms forward to aid in the movement of the COP forward, rather than being at the edge of the foot. This adjustment resolved the tipping issue, and the motion was executed successfully both in simulation and real life. From there, it

could be seen that the ankle joint, i.e. joint_6, does not reach the torque limits in real life, whereas in simulation it does, confirming that the simulation is adding extra noise to the motor readings. Further experiments were performed on 8 cm, and with the previous adjustment of the arm mentioned, the motion was successful in both simulation and real-life experiments.

Finally, experimenting with the 10 cm stairs was deemed problematic. Initially, the model still needed to be updated, and the readings and simulation results did not reflect the actual behaviours. During that time, the simulation readings were within joint torque limits, which led to the experimentation on the real robot, causing the discovery of the discrepancies, leading to the update of the model using data collected from the extensive experiment performed by Felix Aller in the University of Heidelberg on their version of REEM-C, which in turn reflected the real behaviour in simulation. The robot's motion was adjusted as these stairs were different from the others, being more rigid and containing an extra lip, which required extra considerations for the robot to be able to clear the stairs. The simulation readings showed that the knee and ankle joints, i.e. joint_4 and joint_6, respectively, were arriving at the torque limits they can provide, resulting in the concern about carrying the motion into real life. Once the motion was tested on the real robot, it was clear that the joints could not provide the needed torques to lift off the body and perform the clearance, making the 8 cm stairs the maximum height that can be achieved without complications using this method.

5.1 Limitations

As mentioned earlier, during one of the experiments, the REEM-C foot collided with the lip of the stair step damaging the force torque sensor in one of the feet, which caused the foot to malfunction and hindered further testing from occurring on the real robot for six months. In addition, using the current implementation of the motion, the motors in the knee and ankle joints are reaching their maximum torques, not making it possible to perform the motion on higher stairs. This is due to the robot applying extra force to lift off; an improvement upon the algorithm can be achieved by applying a CoM stabilizer to reduce the vibrations occurring during the motion, maintaining the CoP within a specific boundary inside the support polygon. An extra step to reduce the torques on the feet would be taking into account the upper body motion, enabling the utilization of the arms and a handrail to be a form of support, minimizing the torques on the feet.

5.2 Future Work

In the aspect of improvements that can be made, a verification of the joint torque and kinematic limits should be performed to understand the humanoid robot's capabilities better. This can be achieved by generating critical point poses in which the robot would reach its maximum torques. These poses can be generated using optimal experimental design [31], in which we define our experimental constraints. Then optimization is performed to satisfy such constraints to some extent instead of the use of manual experimentation and brute force approach that may take days or even weeks to reach the desired result.

In addition, an adjustment to the motion generated could be helpful to help in reaching higher stairs and provide a more human-like motion. By the implementation of an extra ankle motion, a toe-off can be achieved during lift-off, which can be resourceful in countering the lips that can be found on some of the stairs, providing extra clearance while hovering over the stairs. A similar implementation can be seen here [34]. Another implementation would generate an optimization to favour joint_4 and joint_6 to perform the most negligible torque possible so that they do not reach their limits.

Finally, an attempt to generate the motions for another humanoid robot, such as Waterloo's TALOS, would be interesting to exploit differences and explore more comparisons regarding their capabilities and structure. An attempt was recently performed on a TALOS, and the results produced showed some elasticity in the joints, which caused some issues regarding the implementation in real life [25]. This issue could be addressed to allow for TALOS to perform stair-walking as well, potentially leading towards a more generalized approach for all humanoid robots.

References

[1] Koroibot project. https://orb.iwr.uni-heidelberg.de/koroibot/. Accessed: 2022-07-15.

[2] Felix Aller, Monika Harant, and Katja D. Mombaur. Motion capture based model identification of the humanoid robot reem-c using static poses. *Proceedings of the 10th ECCOMAS Thematic Conference on MULTIBODY DYNAMICS*, 2021.

[3] Narcis Miguel Baños. *REEM-C 7 Training: "Walking"*. PAL Robotics, Barcelona, Spain, Sept. 16, 2020.

[4] Stéphane Caron, Abderrahmane Kheddar, and Olivier Tempier. Stair climbing stabilization of the hrp-4 humanoid robot using whole-body admittance control. In *2019 International Conference on Robotics and Automation (ICRA)*, pages 277–283, 2019.

[5] Defence Advanced Research Projects Agency. Darpa robotics challenge. http://drc.mit.edu/challenge/. Accessed: 2022-10-08.

[6] Defence Advanced Research Projects Agency. Darpa robotics challenge. https://www.darpa.mil/program/darpa-robotics-challenge. Accessed: 2022-07-15.

[7] Chenglong Fu and Ken Chen. Gait synthesis and sensory control of stair climbing for a humanoid robot. *IEEE Transactions on Industrial Electronics*, 55(5):2111–2120, 2008.

[8] Gazebo. https://gazebosim.org/home. Accessed: 2022-10-15.

[9] J.-S. Gutmann, M. Fukuchi, and M. Fujita. Stair climbing for humanoid robots using stereo vision. In *2004 IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS) (IEEE Cat. No.04CH37566)*, volume 2, pages 1407–1413 vol.2, 2004.

[10] Seng Hansun. A novel research of new moving average method in time series analysis. *International Journal of New Media Technology (IJNMT)*, 1:22, 08 2014.

[11] K. Hirai, M. Hirose, Y. Haikawa, and T. Takenaka. The development of honda humanoid robot. In *Proceedings. 1998 IEEE International Conference on Robotics and Automation (Cat. No.98CH36146)*, volume 2, pages 1321–1326 vol.2, 1998.

[12] Yue Hu, Jorhabib Eljaik, Kevin Stein, Francesco Nori, and Katja Mombaur. Walking of the icub humanoid robot in different scenarios: Implementation and performance analysis. In *2016 IEEE-RAS 16th International Conference on Humanoid Robots (Humanoids)*, pages 690–696, 2016.

[13] S. Kajita, H. Hirukawa, K. Harada, and K. Yokoi. *Introduction to Humanoid Robotics*. Springer Tracts in Advanced Robotics. Springer Berlin Heidelberg, 2014.

[14] S. Kajita, F. Kanehiro, K. Kaneko, K. Yokoi, and H. Hirukawa. The 3d linear inverted pendulum mode: a simple modeling for a biped walking pattern generation. In *Proceedings 2001 IEEE/RSJ International Conference on Intelligent Robots and Systems. Expanding the Societal Role of Robotics in the the Next Millennium (Cat. No.01CH37180)*, volume 1, pages 239–246 vol.1, 2001.

[15] Shuuji Kajita, Fumio Kanehiro, Kenji Kaneko, Kazuhito Yokoi, and Hirohisa Hirukawa. The 3d linear inverted pendulum model: A simple modeling for a biped walking pattern generation. volume 1, pages 239 – 246 vol.1, 02 2001.

[16] Shuuji Kajita, Mitsuharu Morisawa, K. Miura, Shin'ichiro Nakaoka, Kensuke Harada, K. Kaneko, Fumio Kanehiro, and Kazuhito Yokoi. Biped walking stabilization based on linear inverted pendulum tracking. pages 4489 – 4496, 11 2010.

[17] W Author Khalil, E Author Dombre, and ML Nagurka. Modeling, Identification and Control of Robots. *Applied Mechanics Reviews*, 56(3):B37–B38, 05 2003.

[18] Jung-Yup Kim, Ill-Woo Park, and Jun-Ho Oh. Realization of dynamic stair climbing for biped humanoid robot using force/torque sensors. *Journal of Intelligent and Robotic Systems*, 56:389–423, 11 2009.

[19] Tzuu-Hseng S. Li, Yu-Te Su, Cheng-Hsiang Kuo, Chi-Yang Chen, Chia-Ling Hsu, and Ming-Feng Lu. Stair-climbing control of humanoid robot using force and accelerometer sensors. In *SICE Annual Conference 2007*, pages 2115–2120, 2007.

[20] Kevin M. Lynch and Frank C. Park. *Modern Robotics: Mechanics, planning, and Control*. Cambridge University Press, Cambridge, United Kingdom, 2019.

[21] MATLAB. The MathWorks Inc., Natick, Massachusetts, 2010.

[22] Philipp Michel, Joel Chestnutt, Satoshi Kagami, Koichi Nishiwaki, James Kuffner, and Takeo Kanade. Gpu-accelerated real-time 3d tracking for humanoid locomotion and stair climbing. In *2007 IEEE/RSJ International Conference on Intelligent Robots and Systems*, pages 463–469, 2007.

[23] Moveit! motion planning framework. https://moveit.ros.org/. Accessed: 2022-07-15.

[24] Pal robotics website. https://pal-robotics.com/about-us/. Accessed: 2022-10-14.

[25] N. Ramuzat, O. Stasse, and S. Boria. Benchmarking whole-body controllers on the talos humanoid robot. *Frontiers in Robotics and AI*, 9, 2022.

[26] Rigid body dynamics library orb (rbdl). https://github.com/harmooni/rbdl-orb/tree/IK. Accessed: 2022-11-06.

[27] Reem-c. https://pal-robotics.com/robots/reem-c/. Accessed: 2022-10-14.

[28] PAL Robotics. *REEM-C Handbook*. PAL Robotics, Barcelona,Spain, 2020.

[29] Bi Sheng, Min Huaqing, Zhuang Zhongjie, Xia tuo, Mo Huaxi, Chen jian, Xu Jincheng, Zhou Yanping, Li Shaojun, Luo Ronghua, and Dong Min. Gait planning of humanoid robots walking on stairs. In Tzuu-Hseng S. Li, Kuo-Yang Tu, Ching-Chih Tsai, Chen-Chien Hsu, Chien-Cheng Tseng, Prahlad Vadakkepat, Jacky Baltes, John Anderson, Ching-Chang Wong, Norbert Jesse, Chung-Hsien Kuo, and Haw-Ching Yang, editors, *Next Wave in Robotics*, pages 25–33, Berlin, Heidelberg, 2011. Springer Berlin Heidelberg.

[30] Bruno Siciliano and Oussama Khatib. *Springer Handbook of Robotics*. Springer-Verlag, Berlin, Heidelberg, 2007.

[31] Byran Smucker, Martin Krzywinski, and Naomi Altman. Optimal experimental design, Jul 2018.

[32] Sebastian Sontag. Calculation and implementation of a motion trajectory for climbing a staircase with the humanoid robot reem-c, 2020.

[33] Trac-ik kinematics solver. docs.ros.org/en/kinetic/api/moveit_tutorials/html/doc/trac_ik/trac_ik_tutorial.html. Accessed: 2022-07-15.

[34] Ruixiang Zhang and Prahlad Vadakkepat. Motion planning of biped robot climbing stairs. 2003.