POLITECNICO DI TORINO

Faculty of Engineering

Master degree in Mechatronic Engineering

Master Thesis

Simulation of a reduced scale of a flexible-biped-wheeled robot

Tutor : Prof. Luigi Mazza
Following Tutor : Prof. Giovanni Gerardo Muscolo
Candidate : Mounif Tiba

Academic Year 2020/21

Table Of Contents

List of Figures

Acknowledgements

I would like to thank my advisor, Prof. Giovanni Muscolo and Prof. Luigi Mazza, for continual drive to make me improve my work and for faith that it would always work better.

At this point I must thank not only those who have been close to me these last months during the development of the thesis, but above all those who have been with me for a longer time, until this important milestone has been reached.

I must thank my parents and my family in general, who has always let me follow my interests and has always supported me in my choices, both morally and economically. Finally, I must thank all the friends who had this adventure called university with me, with whom I spent moments of happiness and moments of despair, but that anyway never left me.

The last thanks go to myself, that I had the strength and the constancy of always going forward, so as to reach this important goal of my life.

Abstract

Mechanical legs are a key component of humanoid robots because they provide exceptional maneuverability and versatility, which determines core application performance like task adaptability, walking speed, and load capacity. On this foundation, we summarize and investigate the actuator and mechanical leg structure design. The design and simulation of a new-generation bipedal robot is presented in this paper. Its modeling and simulation were accomplished using MATLAB Simscape and Simulink, software for creating and analyzing dynamic movement simulations. It was also created using a CAD software. Taking into consideration that this topic has been conceived before. This model allows us to study its movement through experiments that have primarily focused on analyzing the behavior of the servomotor connected to the wheels axle of each leg by controlling the angular position and angular velocity using a PID Controller to achieve the desired wheel motion. Taking into account that the robot is conceived for usage in domestic environments and consequently improve the design of motion and balance controller in the real robot.

CHAPTER 1 -INTRODUCTION

1 INTRODUCTION

Since 1970, various researchers on biped walking robots have been conducted [1, 2, 3, 4]. During that time, technical advancements allowed biped walking robots to evolve into biped humanoid robots. Furthermore, in the intelligent robot research society, the biped humanoid robot has become a representative research issue. Many scholars believe that the humanoid robot business will be the industry leader of the twenty-first century, and that we will eventually enter an era where every home has at least one robot. A long-standing quest for human-like robots has led to a considerable concentration on biped humanoid robots. Furthermore, in a human-robot community, a human-like look is desirable for coexistence. Robots are artificial devices that are programmed to do a variety of jobs. They are often used to aid humans or, in some situations, to replace them. Manufacturing, building, and handling of large and dangerous products, as well as working in hazardous or non-human-compatible situations, are just a few examples. Because they lacked external sensing, the initial robots were utilized for simple tasks such as moving goods from one location to another. In 1956, George Devol, who had written a patent on a machine called a Programmed Transfer Article two years before, and Joseph Engelberger met and formed the first industrial robot enterprise. Unimation was the first company to put a robot in a General Motors (GM) factory in Trenton to serve a die casting process. These robots took over duties that were monotonous, repetitive, heavy, and dangerous for people. More advanced applications, such as welding, grinding, deburring, and assembly, arose when robots could handle both a more complex motion and external sensor capacity [5]. Advanced robots are no longer limited to factories and manufacturing duties; they are now being used in industries such as medicine, agriculture, exploration, maintenance, and aid. Until date, the majority of robots have been designed to be completely rigid, allowing them to move at fast speeds with high precision and repeatability. Rigid bodies, on the other hand, must be large and, above all, hefty. This means that in order for those robots to achieve great performance, they must consume a lot of energy. Furthermore, if the robots must interact with people, they must move at a very moderate speed to avoid injuring anyone [6]. As a result, their performance capabilities are severely limited. Humanoid robots are multi-body devices with a lot of moving parts. Many biped robot dynamic balance systems have been developed and tested in the literature, however the problem of falling down when walking has remained unsolved. The complexity of a legged machine's control causes this challenge [7] . The following are some important challenges for the control system: 1) The kinematics and dynamics of robots are challenging to model effectively; 2) The robot's dynamics are determined by which legs make contact with the ground. In other words, when the robot switches from a single support phase to a double support phase or a flight phase, and vice versa, the dynamics change. Furthermore, the change of leg support is accompanied by an impact that causes the robot's mobility to be disrupted;3) A legged robot is subjected to holonomic and nonholonomic limitations on a regular basis; 4) the environment is unknown and changing. The surface could be pliable, sticky, soft, or rigid.5) Vertical contact forces on the surface are unilateral, meaning they cannot pull the robot against the surface; 6) maintaining dynamic balance is difficult to deconstruct into actuator commands; 7) many degrees of freedom have to be controlled real-time. a large number of degrees of freedom must be regulated in real time. Ground robot locomotion can be divided into two categories: leg- and foot-based [8, 9, 10, 11] and wheel-based [12, 13]. While some indoor walking robots perform admirably while navigating difficulties such as stairs or slick terrain, they nevertheless require a significant amount of time to complete these complex actions. Robots having rotating parts, like as wheels, on the other hand, are well suited for flat terrain since they can move smoothly, efficiently, and quickly. Designing and controlling legged robots is far more difficult than designing and controlling wheeled robots. New tasks, such as construction, assembly, and human help, will be performed by robots. On future missions, human-robot cooperation and interaction may necessitate robot structural configurations and performances that are more human-like. As a result, the utilization of humanoid robotics technology in such missions and operations could be beneficial. While it is not difficult to create a human-like biped robot platform, achieving stable biped robot walking offers a significant problem. This is due to a lack of knowledge about how humans walk steadily. On legged robots, locomotion control algorithms such as the central pattern generator [14, 15] and the simplified Zero Moment Point (ZMP) technique for biped locomotion in a planar surface [16] have been applied. The dynamic balancing of biped robots has been addressed in recent works [17, 18], with the goal of bypassing the ZMP technique. However, dynamic balancing is a significant issue in biped locomotion, and some unique solutions, such as wheels on feet [19, 20] have been proposed to address the issue. The wheeled humanoid robot, which debuted lately, is a novel sort of humanoid robot in which the driving wheel replaces the standard plate foot structure used in humanoid or biped robots. As a result, instead of stepping alternately, wheeled humanoid robots must locomote by rotating their wheels. In the field of robotics, the Handle robot [21], introduced by Boston Dynamics in 2017, has gotten a lot of attention. Its outstanding agility and maneuverability on horizontal ground, as well as its adaptation in challenging terrain, were showcased in the released video. The antecedents of wheeled humanoid robots, on the other hand, may be traced back to 2007. Hitachi engineers created the EMIEW-2 [22] service robot to interact with humans in an indoor environment. ASCENTO [23], a wheeled humanoid robot being developed at ETH Zurich, aims to provide maximal indoor mobility. It is modest in size and only weighs 10 kg, but it has impressive mobility capabilities. Incorporating elastic components into the robot's structure could be a realistic option. When deformations are taken into consideration during the mechanical design process, smaller components can be used, resulting in a lighter construction. Elastic components can also store energy and then release it at a later time. Furthermore, by utilizing the resonance phenomena, the actuator's effort can be amplified. As a result, the overall system efficiency improves, as does their performance in terms of energy usage. Furthermore, an elastic frame can absorb some of the shock caused by the robot's interaction with the environment or objects, both intentionally and unintentionally. This makes interactions with people and the unstructured world safer, preventing both the robot and its surroundings from being damaged. Because their deformation smooths the impulsive torques imparted to the joints, the insertion of flexible parts can lessen the discontinuity created by the impact with the ground This filtering process benefits the structure as well, because it reduces the load on it and preserves its integrity. Finally, the use of flexible structural elements allows the robot to store energy without the addition of additional components, allowing the structure to be as light as feasible while enhancing total system efficiency. Flexible robotic structures have emerged in recent years as a result of the emergence of high-performance processors. When it comes to bipedal robots, there are numerous instances of both academic and commercial robots that have some degree of elasticity.

To summarize, controlling a bipedal walking robot that can navigate various terrains and walk/run at high speeds is still an unsolved problem. The most major research institutes and firms with backgrounds in legged robots presented alternate methods such as the use of wheels in place of or in combination with the legs to simplify biped locomotion control. The biped locomotion capabilities demonstrated by this robot proved that wheels could overcome many of the major issues in biped locomotion, allowing for more robust biped dynamic walking, running, and jumping. Recently, several of the current authors presented Rollo [24], a revolutionary biped wheeled robot with a flexible construction. Rollo is a basic robot that combines the features of biped wheeled robots with flexible legs to improve performance while reducing the number of actuators.

The paper is structured as follows: Sec. 2 shows the ROLLO humanoid robot with its conceptual and functional design; Sec. 3 shows ROLLO robot design; Sec. 4 shows ROLLO robot model; Sec. 5 shows motor model on matlab simulink; Sec. 6 shows servo motor control; Sec. 7 presents the results of experimental tests on the robot and relative discussions; finally, Sec. 8 presents conclusion and future works.

CHAPTER 2 -ROLLO HUMANOID ROBOT: CONCEPTUAL AND FUNCTIONAL DESIGN

2 ROLLO HUMANOID ROBOT: CONCEPTUAL AND FUNCTIONAL DESIGN

The proposed solution is based on a new humanoid robot engineering design that reduces complexity while allowing for the same operative functions in the actual world. The authors concentrated on a specific scenario, namely the operation of robots on planar surfaces, where a precise replication of human motion is not required, but it is sufficient that the robot resembles human elements in terms of movement and shape.

2.1 Rollo Robot

Rollo, shown in Figure 2-1, was created as an avatar robot for patients suffering from Amyotrophic Lateral Sclerosis (ALS), a chronic neurological disease that limits human movement and interaction. A biped robot could help individuals with ALS [25, 26], and the Rollo robot was built specifically for those patients, with its basic construction, low power supply, and great vertical stability.

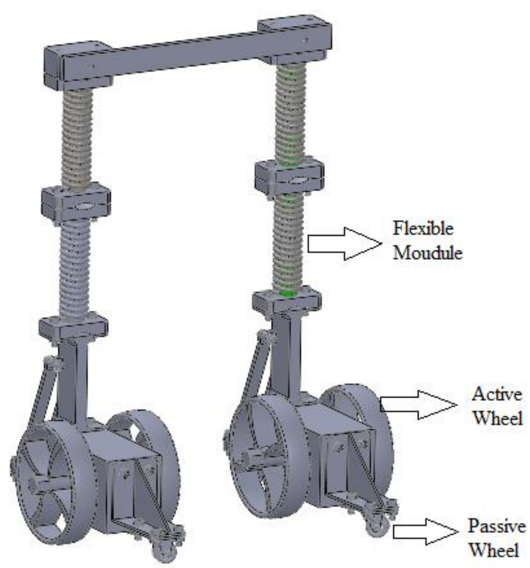

Figure 2-1 : Reduced Scale of the Flexible-wheeled biped robot Rollo.

ROLLO robot, that was conceived by previous project [27] [28] [29] [30], designed to be controlled using a human machine configuration. ROLLO Robot is a simple robot that is designed (in the first version) with a height around 1000 mm and weight around 10 kg. The speed, locomotion type, and direction are the input parameters required to drive the robot (forward, backward, turn left, turn right).

Rollo is a bipedal robot with flexible legs and wheeled feet. The two legs are joined at the top by a horizontal and parallel to the ground beam-like construction. On the bottom side, each leg is attached to the feet. On the beam that connects the two legs, the robot head is positioned.

The two legs are identical and are made up of two springs apiece. one gear motor, one battery, two side wheels (external and internal) connected to the gear motor shaft, and two free wheels (frontal and rear) to prevent falls. Each leg can have varied combinations of motion by adding springs or stiff component modules. One spring at each human-like joint (ankle, knee, and hip) and rigid parts near each human-like link can be used to give the robot a human-like motion (thigh and shin). ROLLO can be moved in a straight or curved path. The rotation can be achieved by either fixing the external wheel to the shaft and using a bearing between the central hole of the internal wheel and the shaft of the gear motor, or by fixing each wheel (external and internal) to the shaft of the gear motor and having two driving wheels in each leg; in this case, the bipedal rectilinear motion and rotation are both possible. Other two free wheels are present in each leg, in the frontal and rear parts, respectively; they are not fixed and may be moved around an axis perpendicular to the surface on which the robot is travelling, and they are only used to avoid falls. A battery is placed in each leg as a power source, allowing the robot to move at maximum speed for one hour. The robot can spin by turning each foot's wheels in the opposite direction. The quantity of elastic modules in each leg can also be changed to change the robot's altitude.

The basic elements of ROLLO's mechanical leg construction are springs, which are coupled in series in such a way that the robot may perform alternative biped walking patterns and remain in a standing position even when the motors are turned off. The motion is made possible by only two motors, which are located within the leg structure and are directly connected to the wheel rotation. The two gear motors must rotate in the same direction in order to drive the robot in a rectilinear direction at a consistent speed. In this situation, two motions are possible: one with the legs moving together, and another with the legs moving alternately in a pattern that resembles human walking. The robot will rotate along an axis perpendicular to the plane of motion if the speed value of the gear motors is the same and the direction of rotation of each motor is opposite, due to the overall construction that binds the legs together.

Rollo has two types of locomotion: human-like walking with alternating foot motions, which is the emphasis of the study, and simpler locomotion with the feet constantly parallel, as described in [25]. The flexible modules of each leg collect energy and give it back, net of dissipations, in the next step in the human-like locomotion type, just as they do in human walking. The human-like walking mechanism reduces the amount of energy expended during walking.

Because of the rollo robot's tiny size, simple mechanical design, and limited number of degrees of freedom, architectural, actuation, and control requirements are simplified. and the robot's ease of movement in a real setting could not limit its practical application. Another benefit was the safety of those who were in the vicinity of the rollo robot. A wheeled robot has numerous advantages in terms of energy efficiency, movement control, and motion stability.

CHAPTER 3 - ROLLO ROBOT DESIGN

3 ROLLO ROBOT DESIGN

The first version of the Rollo robot was presented in [29] [31] [30] [28].The design of a reduced scale of the Rollo robot is done with CAD software shown in Figure 2-1. The dimensions of the model are smaller respect to the first real version, and the total weight of the reduced scale wheeled legged robot of the Figure 2-1 is about 98 g. The purpose is to create and simulate a control logic in a small-scale prototype and then perform the simulation in order to control the servomotors, necessary for the movement of the robot.

3.1 Rollo In Details: Legs

3.1.1 Zero Moment Point

Bipedal walking robots are a subject of growing interest of humanoid robot development. Humanoid robots should have human-like features because they were designed to work in a human sense and interact with humans [33]. Legs are one of the main limbs that distinguishes human aspect, and they are a key component in the construction of humanoid robots. Recent study aims to create a leg mechanism with mechanical properties similar to human legs. In comparison to other forms of legged robot systems, bipedal robot locomotion control is highly difficult. Controlling biped robot mobility requires a high level of stability. To conduct stable locomotion, a bipedal robot must consider more dynamic factors (position, velocity, and acceleration) than quadrupedal and hexapod robots, which can adjust their stability using foot and center of mass (CoM) positions.

The walking gait is the most common mode of human mobility. When compared to the running gait, it has more stability, safety, and energy efficiency. The majority of bipedal robots move in a walking gait, however a handful may hop or run. The majority of this field's research focuses on the control problem as a critical part of developing highly efficient bipedal robot systems. The control methods for bipedal robot locomotion are based on body dynamics. One of the most well-known hypotheses for bipedal locomotion control is the zero moment point (ZMP) technique. ZMP is thought to be extremely stable and simple to use for foot trajectory planning, body position, and velocity. The ZMP approach, on the other hand, has the drawback of being inefficient in terms of energy expenditure because it specifies the foot and body trajectory regardless of joint stress.

The dynamic walking method is another bipedal locomotion approach. Tad McGeer coined the term "Passive Dynamic Walking" to describe this approach, which depends on the body's momentum to create the walking motion. It is thought to be extremely energy efficient and capable of mimicking human locomotion [34]. However, due of its poor motion control and susceptibility to ambient disturbances, this approach is not commonly used [35].

Given that the zero moment point (ZMP) control is the most widely utilized theory for stability in humanoid locomotion [36] , a continuum contact of the robot's foot with the ground could be a decent approximation of the real human swing, with the ultimate goal of duplicating biped locomotion. In this situation, the rapidity and durability of wheeled movement could be a viable option to the robot feet design.

On legged robots, locomotion control algorithms such as the central pattern generator [14, 15] and the simplified Zero Moment Point (ZMP) technique [37] for biped locomotion in a planar surface [16] have been developed. The dynamic balancing of biped robots has been addressed in recent works with the goal of bypassing the ZMP technique. [17, 18]. However, dynamic balancing is a significant issue in biped locomotion, and some unique solutions, such as wheels on feet have been proposed to address the issue [19, 20] . Several authors of the first version presented Rollo robot [29] [30] [28] [25], a revolutionary biped wheeled robot with a flexible construction. Rollo is a basic robot that combines the features of biped wheeled robots with flexible legs to improve performance while reducing the number of actuators.

3.1.2 Cylindrical Helical Spring

Within the past work [29] [30] [28] the usage of mechanical springs is demonstrated, which allows for lower actuator power requirements, energy storage and release when needed, and collision avoidance due to interactions with the environment. The authors show that spring stiffness affects both the response to external perturbation and the arc radius of arc-shaped feet. The spring absorbs the shock of the uneven surface, or step encounter, and the actuator corrects the robot's tilt as a result of the encounter. Furthermore, the idling control prevents the wheel from idling excessively when it leaps owing to such occurrences. The robot's lower body is made up of two flexible legs made up of cylindrical helical springs that are used in an unusual manner.

In more detail, cylindrical helical springs are typically employed for compression and extension in the same direction as the spring's axis (see Figure 3-1). The cylindrical helical springs, which are integrated into the leg construction, are used in the given robot to generate a passive motion by utilizing the springs' torsional and flexural properties. While the compression/extension behavior of cylindrical helical springs is well-studied in the scientific literature, torsional and flexural behavior necessitates a more complicated computation.

Della Pietra [38] investigated the relationship between torsional and flexural strains in cylindrical helical springs exposed to excitation along their own axes both theoretically and practically in 1976. The analytical examination, which lasted until 1976 and was based on the theory, revealed the existence of two sets of resonance conditions: one corresponds to the one already known for axial vibrations, and the other is related to the vibrations caused by coil rotations around the spring axis. Because of the presence of lateral deformations in the spring, the experimental research carried out by on two springs with different characteristics and subjected to different assembly preloading has shown that the coupling between torsional and flexural strains is much more complex than the one indicated by theory [38].

Recent publications [39, 40] offer a free vibration analysis of cylindrical helical springs with non-circular cross-sections and analyze helical springs under axial loads under various dynamic situations.

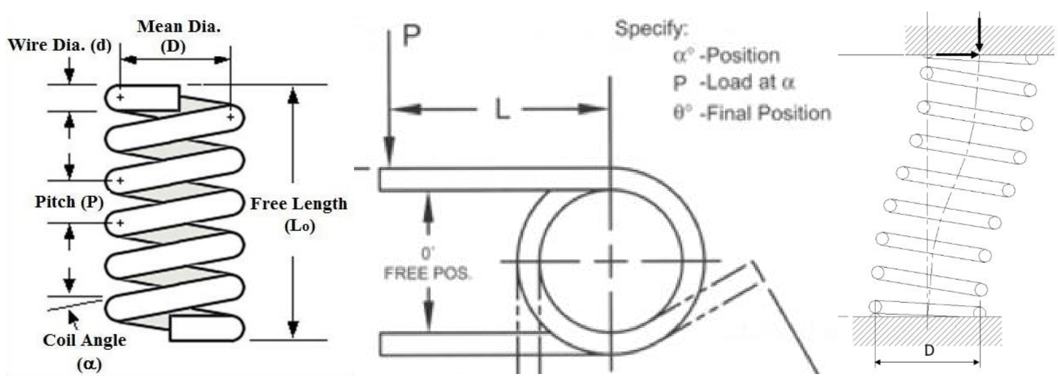

Figure 3-1 : Sketch of torsion, compression/extension and unconventional use of a cylindrical helical spring.

A typical helical spring is a spiral wire or rod with mean coil diameter of coil D = 2R, wire diameter d, number of active coils i, helical degree , uniform pitch of the helix p = $\pi * D * tan\, \alpha$, free length H = i $*$ p, length of the wire or rod L = π $*$D $*$ i., illustrated in Figure 3-1 . When an axial force is applied to the spring, the wire experiences a mix of torsional and flexural strains. [38, 39, 40] contains a wealth of information about the spring's nonlinear behavior in this and other situations. We illustrate some standard assumptions used to describe the stiffness of the spring (K) in the following to simplify the topic and generate a rough estimate of the stiffness of the ROLLO's legs (valid also if the spring is used in an unconventional way). We employed a cylindrical helical spring that was activated by a load applied perpendicular to the axial direction. The first common assumption is that trigonometric functions can be simplified ($cos\,\alpha = 1\,, \sin\alpha = 0$) with a 1% error if the helical degree α is minimal (<8°). The second common assumption utilized in many experimental applications is that the cylindrical helical spring's flexural and torsional behavior is similar to that of a torsional spring. Consider simply the torsional load M_t on the spring for a basic technique of calculating the behavior of the cylindrical helical spring. The maximum strain on the wire is τ_{max} where J_p is the inertial moment:

$$\tau_{max} = \frac{M_t}{J_p} \cdot \frac{d}{2} \quad \textit{Eq (3-1)}$$

The work generated by the force is compared to the elastic work accumulated on the wire in order to calculate the spring stiffness. F is the torsional force that causes a f displacement; ϕ is the rotation induced by M_t ; G is the tangential module.

$$M_t = F \cdot R \quad \textit{Eq (3-2)}$$

$$\frac{1}{2} \cdot F \cdot f = \frac{1}{2} \cdot M_t \cdot \phi = \frac{1}{2} \cdot M_t \cdot \frac{M_t \cdot L}{G \cdot J_p} = \frac{1}{2} \cdot F \cdot R \cdot \frac{F \cdot R \cdot 2 \cdot \pi \cdot R \cdot i}{G \cdot \frac{\pi \cdot d^4}{32}} \quad \textit{Eq (3-3)}$$

$$K = \frac{F}{f} = \frac{G \cdot d^4}{64 \cdot R^3 \cdot i} \quad \textit{Eq (3-4)}$$

The flexural degree φ of the spring, comprising Young module E and inertial moment J, may be determined using the same assumptions and referring to Figure 2-1 where a flexural moment M_f is applied on the spring:

$$M_f = P_1 \cdot a = P_2 \cdot b \quad \textit{Eq (3-5)}$$

$$\varphi = \frac{M_f \cdot i \cdot D \cdot \pi}{E \cdot J} \quad \textit{Eq (3-6)}$$

If $J = J_P$, the parameters for determining the springs for ROLLO's legs are as follows, and are functions of I ,D, and d:

$$f(i, D, d) = \frac{i \cdot D}{d^4} = \frac{E \cdot \varphi}{64 \cdot M_f} \quad \textit{Eq (3-7)}$$

Furthermore, whereas other biped robots on the market and in the literature require power to maintain a standing position, ROLLO can maintain an upright position without the use of a power supply because to the cylindrical helical springs included into the frame. This feature allows us to limit the amount of energy the robot expends when in motion. Another feature of the robot is that it combines the structure's flexibility with the biped wheeled system, allowing for a wide range of step combinations. The robot can, for example, move by alternating the steps of each foot with regard to the other, or by a common motion of the two legs, with the entire robot moving like a standard wheeled robot.

With legs in extension, energy storage and return can lead to intriguing behaviors, just as it might with compressive loading. Planting the distal end of the leg using contact normal or frictional forces while the system's kinetic energy is stored as potential energy in the spring is a popular approach to store energy in either direction. The external ground reaction force will point in towards the robot's center of mass, making this easier to perform with a leg in compression. The leg must be controlled in extension to maintain frictional contact while the body's inertia pushes the hip away from the toe. The depicted behavior applies a leg torque to keep the robot on the ground while the system's forward kinetic energy is stored in the leg spring and subsequently returned as rotational energy to flip the robot over.

3.1.3 Servomotors

Servo motors, or "servos," are electronic devices having rotary or linear actuators that precisely spin and push elements of a machine [41]. Servos are primarily used to control angular or linear position, velocity, and acceleration. Because of their compact size and power, servo motors are widely used by businesses. Despite its small size, it produces a significant amount of power and is noted for being extremely energy efficient. The majority of businesses that employ servos are those that require them to place control surfaces and spin items at precise angles and distances. The majority of organizations that use servo motors are manufacturing enterprises that use servo motors in their machinery. In the industrial field, there are two types of servo motors that are available and employed. The AC servo motor is the first. The majority of businesses now employ this form of servo. AC servo motors are typically seen in industrial settings. AC servo motors are AC motors that use encoders to control their speed. These servo motors are controlled by controllers that provide feedback and closed-loop control. They are well-known for their exceptional accuracy and ease of control. The DC servo motor comes in second. Fuji Electric employed these kind of servo motors in the past, but they are now rarely used because AC servo motors are easier to use, more effective, advanced, and reliable.

Advantages: Servos are noted for being frequent and consistent in their operation. As a result, if the motor is under a heavy load, the driver will increase the current to the motor coil as the motor turns. This essentially means that servo motors are supposed to be mechanically correct at all times. It also allows organizations to use it at a high-speed due to its precision.

We take the servomotor SG90 as the real model of the motors for potential future prototyping. In a first step, we planned to use the T-pro SG90 micro servomotor, as shown in Figure 3-2.

Figure 3-2 : T-pro SG90 micro servomotor.

3.2 Ground Friction

The main propelling force for mobility in nature is ground friction. The normal force acting vertically to the surface determines the friction force. The friction coefficient μ determines the tangent force as follows:

$$\mathrm{F_t} = \mu \cdot \mathrm{F_N} \qquad \textit{Eq (3-8)}$$

The value of the friction coefficient is a positive number not exceeding 1, that is:

$$0 < \mu \leq 1 \qquad \textit{Eq (3-9)}$$

The horizontal force must not surpass the friction force in order to maintain stable locomotion; otherwise, the foot will slip during locomotion. The following are the response forces on the foot to the surface:

$$R_x = m \cdot a_x \qquad \textit{Eq (3-10)}$$

$$R_z = m \cdot (a_z + \mathrm{g}) \qquad \textit{Eq (3-11)}$$

The reaction force angle rate that the tangent of the angle is determined by is:

$$\tan\ \theta = \left|\frac{R_x}{R_z}\right| \qquad \textit{Eq (3-12)}$$

The reaction force angle should be less than the friction coefficient in order to have stable walking. This is determined by:

$$\tan\ \theta < \mu \qquad \textit{Eq (3-13)}$$

Gravity determines the vertical response force. The vertical reaction force will be sufficient for steady locomotion if the gravity is higher. In low gravity, on the other hand, there may not be enough vertical response force to maintain stable locomotion. As a result, with low gravity, locomotion must be carefully managed. Due to the fixed height of the trunk, the ground vertical reaction force is assumed to be constant. However, as the mass of the object grows, so does the response force. The reaction force rate primarily increases with an increase in gravity in this basic calculation. However, there is some vertical motion in a genuine walking robot due to the mobility of its parts, indicating that the ground vertical response force is not constant. In decreased gravity, the effect of this motion is amplified. As a result, contrary the simple model's prediction, the reaction force rate is higher with lower gravity.

CHAPTER 4 - ROLLO ROBOT MODEL

4 ROLLO ROBOT MODEL

4.1 Matlab Simscape

Simscape allows us to quickly develop physical system models within the Simulink framework [43]. Simscape enables the creation of link-based physical component models that can be used in conjunction with block diagrams and other modeling paradigms. By arranging the main components in a schematic, we may represent systems like electric motors, bridge rectifiers, hydraulic actuators, and refrigeration systems. Simscape add-on solutions bring more advanced analytics components and features to Simscape. Simscape is a MATLAB-based language that allows text-based writing of physical modeling components, domains, and libraries. It may be used to construct control systems, test system-level performance, and generate bespoke component models. We'll be able to use MATLAB variables and expressions to parameterize our models, as well as create control systems for our physical system in Simulink. Simscape provides C code creation for deploying models to different simulation environments, such as hardware-in-the-loop (HIL) systems.

Simscape Multibody (a Matlab program) is a multibody simulation environment for 3D mechanical systems that allows us to build multibody systems with blocks that represent bodies, joints, constraints, force elements, and sensors. We offer a biped robot simulation model in Simscape Multibody in this study. The robot model is made up of many sorts of bodies and revolute joints, each with its own degree of rotational freedom.

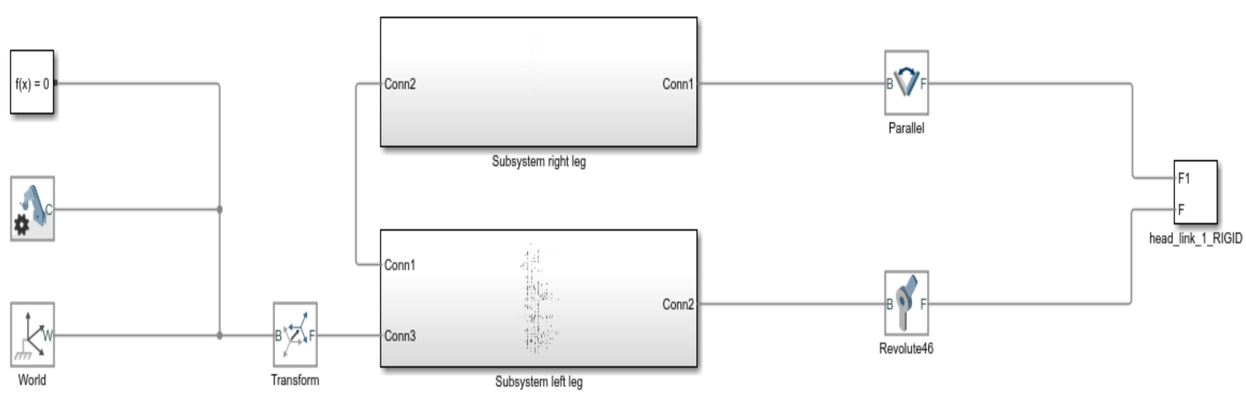

Figure 4-1 : Rollo Robot model on simscape

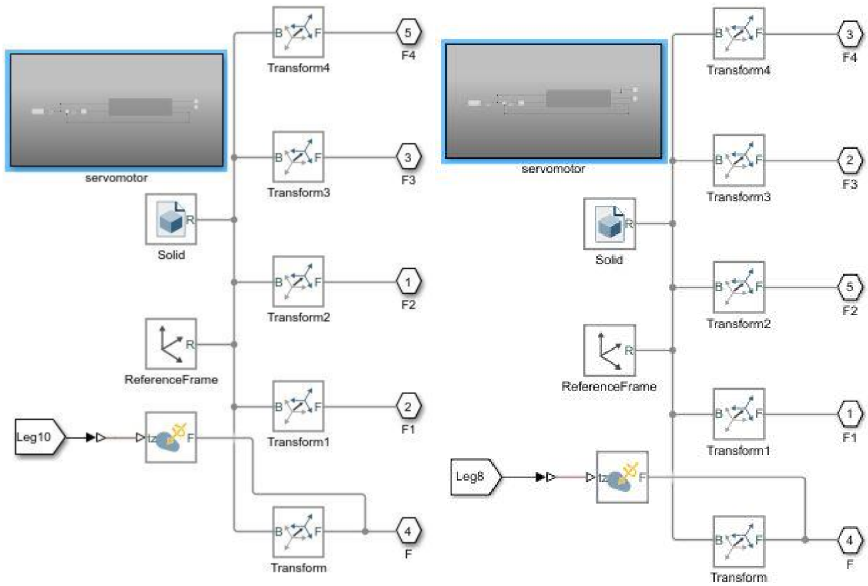

Figure 4-2: Simscape model showing the servomotors connected to the wheels axle of each leg.

We can see from this model Figure 4-1 there are blocks that are connected together. And they all start here at the bottom left. With the world frame, the solver configuration block that is needed for all models using simscape and the mechanism configuration block in which we can define things such as the direction and magnitude of gravity. After this we have blocks objects and rigid transforms that make it all the way to the robot. So, as we can see, the leg consists again of a sequence of solid blocks, rigid transformations and joint blocks, and all of these are going to define that 3D system. and there are some wheels, and the robot is attached to the wheel using a block called revolute joint. This makes sense because the wheel can move with respect to the body only through its rotation about the axle, so this is kind of how we build up that whole system, and most importantly is the articulations or joints. In the Simulink Library browser there is the Simscape Multibody library. Under joints there are lots of different ones available, like 6 degree of freedom, joints, Cartesian joints. Prismatic joints revolute joints. These are what define motion or degrees of freedom between rigid objects in our model and in the case of The Walking robot, we basically have a collection of revolute joints that define those motions for the leg. So, the idea is that we're just going to put together this model that consists of several rigid parts, and we're going to simulate them. When we run this, it opens up the mechanics Explorer and this is the visualization window for all simscape multibody models. So this is a way to debug whether our model was assembled correctly, is initialized correctly, etc. So, the model is updated by going to simulation update diagram. And this is going to bring up a window called mechanics Explorer, which is the built in 3d viewer from simechanics. The concept of physical modeling is introduced. Although we are looking at a model here, which is in the Simulink editor, all the blocks here belong to a subsystem called simechanics. Simechanics is a block set of simscape which is the physical modeling suit here. So it has a number of blocks called body gears and everything which relates to mechanical systems. Physical modeling helps to model multiple domains such as electrical, hydraulic and mechanical system in the same environment physical modeling is just like Simulink when it comes to simulating systems based on first principle equations in physical modeling or where we have more flexible parameters to tweak and attained different levels of fidelity that we want, without having to rewrite all the first principal equations with different sets of assumptions. In other words, test the robot for mathematical correctness by creating some test cases. Test one is an actuated test to see what the robot does under just the force of gravity. test two add a simple actuation to see what the robot does under actuation. Test for specific velocity conditions in test three, which are basically having the right and left wheel angular velocities, and test for different conditions to see when the robot moves straight and when the robot turns, etc. The wheel oscillated when running the model. This is not expected behavior because under gravity, the robot should just stay put. In the mechanism configuration block the gravity vector must be modified. The gravity was in the negative y direction. However, it should be in the negative z direction. Then the wheel doesn't move, and this is the expected behavior. This is test one. In second test, add actuation, it can be added by accessing the joint blocks and the actuation parameters. Provide motion as input, and automatically compute the torque that is required to get that motion. When this is chosen an extra signal port to give the input the revolute joint takes the angular position as input. Get a source which is a constant. if we feed in this constant to the revolute joint, this is going to make the robot move the wheel once and stop, but we want it to rotate at a constant rate. So also, we bring in an integrator which is going to integrate the constant rate and give a ramp kind of signal for the angle. So, the wheel moves at a constant rate. when the model becomes a little cumbersome to handle. And in order to make it more readable and easier to understand, select these blocks together. And create a subsystem. the robot functionally doesn't change. But it would be easier to maintain this system. Similarly, do this for the other wheel as well.

4.1.1 Blocks Properties

-Solver configuration: For simulation, each physical network represented by a connected Simscape block diagram requires information about solver settings. Before we can start simulation, the Solver Configuration section gives the solver parameters that our model requires. A single Solver Configuration block must be attached to each topologically distinct Simscape block diagram.

-Ports: There is only one preserving port in the block. By building a branching point and linking it to the Solver Configuration block's lone port, we may add this block to any actual network circuit.

-Mechanism Configuration: This block gives a mechanism, which is a self-contained group of interconnected Simscape Multibody blocks, mechanical and simulation characteristics. Gravity and a linearization delta for determining numerical partial derivatives during linearization are two parameters. Only the target mechanism, i.e., the mechanism to which the block links, is affected by these characteristics. It's not required to use the Mechanism Configuration block. The gravitational acceleration vector is set to zero if we leave it out. Use just one instance of this block per mechanism, and if the mechanism contains one or more Gravitational Field blocks, set uniform gravity to None.

-World Frame: In a model, this block represents the global reference frame. This frame is inertial and completely still. When a frame is rigidly connected to the World frame, that frame becomes inertial. The axes of the frame are orthogonal and arranged using the right-hand rule. The World frame is the ultimate reference frame in a frame network. All other frames are defined in relation to the World frame, either directly or indirectly. When numerous World Frame blocks connect to the same frame network, the frames are identified as one. If no World Frame block connects to a frame network, a copy of an existing frame is used as the World frame, frozen in its original position and orientation.

-Rigid Transform: This block transforms two frames in a time-invariant manner. With regard to the base port frame, the transformation rotates and moves the follower port frame (F) (B). When the frame ports are connected backwards, the transformation is reversed. During simulation, the frames remain fixed with relation to one another, moving only as a single unit. To model compound rigid bodies, combine Rigid Transform and Solid blocks.

4.2 Foot-Ground Contact Modeling

The behavior of the foot's contact with the ground was reproduced in this paper's model using the Simscape Multibody Contact Forces Library, which provides contact force models for intermittent contact [44]. A generic strategy was utilized to create correct contact between the sections of the system that collided during simulation:

1. Determine which components of the system will collide during simulation.
2. Determine which edges or surfaces will come into contact.
3. Draw reference frames for the intersecting lines and arcs.
4. Between the two frames, add a contact force model.

We'll include physical touch between the robot's feet and the ground, so that adjusting the robot's joints has some influence on the environment and the robot can walk.

The spring stiffness should be determined by the weight we're supporting, which we know is the robot's weight, as well as any allowable overlap between the foot and the ground. A few millimeters, for example. As a result, the stiffness coefficient, which is the weight divided by the displacement, was calculated. The damping should then be chosen based on the spring constant to ensure that the contact does not have too many or too few oscillations. Of course, we usually go at least one order of magnitude below the spring constant, but this isn't a hard and fast rule, so we should experiment with different values to find what works best. After that, there are friction parameters to consider. If we are aware of the nature of the contact. Static and dynamic friction coefficients that are reflective of our system can sometimes be found online.

CHAPTER 5 - MOTOR MODEL ON MATLAB SIMULINK

5 MOTOR MODEL ON MATLAB SIMULINK

5.1 Matlab Simulink

Simulink is a graphical programming environment that uses MATLAB to model, simulate, and analyze multidomain dynamical systems [45]. Its main interface consists of a graphical block diagramming tool and a set of block libraries that can be customized. It has a close interaction with the rest of the MATLAB environment and may be used to either drive or script MATLAB. Simulink is a multidomain simulation and model-based design tool that is widely used in automatic control and digital signal processing. It allows for the quick creation of virtual prototypes that may be used to explore design concepts at any level of detail with minimal effort. Simulink has a graphical user interface (GUI) for creating models in the form of block diagrams. It comes with a large library of prefabricated blocks that may be used to create graphical representations of systems by dragging and dropping them. The user can create an "up-and-running" model that would otherwise take hours to construct in a laboratory setting. It can represent linear and nonlinear systems in continuous time, sampled time, or a combination of both. Simulink's interactive nature encourages us to experiment; we can change settings "on the fly" and observe what happens right away, allowing for "what if" inquiry. Finally, Simulink is integrated with MATLAB, allowing data to be readily transferred between the two programs.

5.2 Servo Motor Used in Details

Two actuators are attached to the axles of each leg's wheels to allow robot movement, while the knees are passive joints with torsional springs. For prototyping, we use the servomotor SG90 as an actual model of the motors. It's little and light, yet it packs a punch. The servo can spin 180 degrees (90 degrees in each direction) and functions in the same way as the usual types, although it is smaller. To control these servos, we can use any servo code, hardware, or library. It's ideal for beginners who wish to move things without having to develop a motor controller with feedback and a gearbox, especially because it fits in small spaces. It includes three horns (arms) as well as hardware.

Specifications:

• Weight: 9 g

• Dimension: 22.2 x 11.8 x 31 mm approx.

• Stall torque: 1.8 kgf·cm

• Operating speed: 0.1 s/60 degree

• Operating voltage: 4.8 V (~5V)

• Dead band width: 10 µs

• Temperature range: 0 ºC – 55 ºC

Position "0" (1.5 ms pulse) is middle, "90" (~2ms pulse) is all the way to the left. ms pulse) is all the way to the right, ""-90" (~1ms pulse) is all the way to the left. as shown in Figure 3-2

The performance of servomotor under various conditions is simulated using MATLAB/SIMULINK environment and simulation result demonstrates the feasibility of the proposed system. See Figure 5-1

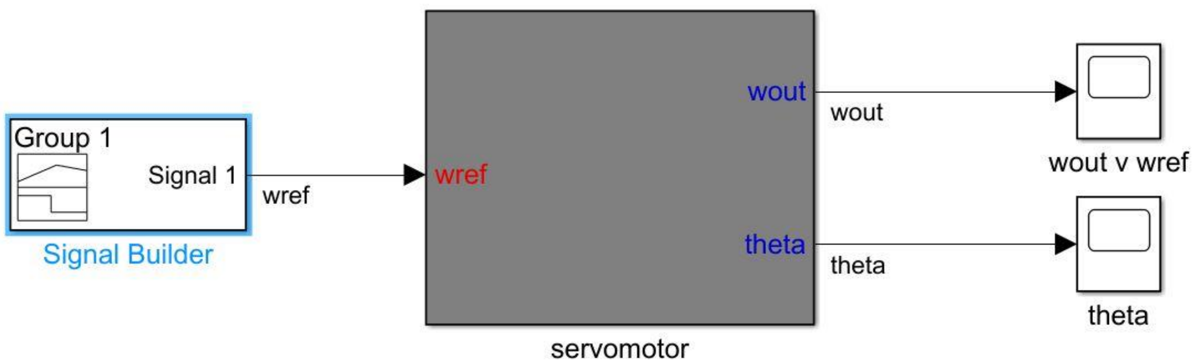

Figure 5-1 Servomotor model using Matlab/Simulink

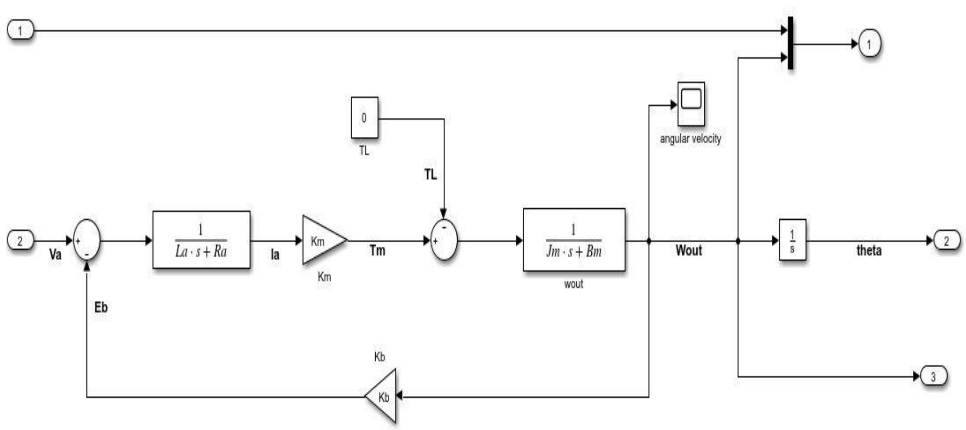

Figure 5-2: Detailed Servomotor.

The servo has a couple of terminals where voltage is applied, therefore some type of voltage source is required, denoted by Vs, and the motor itself can be described as having resistance and inductance which are commonly denoted by as armature resistance Ra and armature inductance La, As the motor spins, a back emf of back voltage is generated donated as Ve, and when voltage is applied to those terminals of the motor, it creates a current through this circuit. There are 2 equations working with servomotor that are the back emf Ve=$k_e \theta_m$ where θm is the rotational rate of the motors rotor, the rotor shaft the business end of the motor.

The other equation is that the torque imposed onto that shaft via the motor is $T_m = k_t i_a$T so when current run throught that motor, torque is generated onto the shaft that is proportional to it.

R_a L_a k_e k_t are parameters of the motor.

The different equations related to servo motor are given below:

$$e_m(\mathrm{t}) = K_m \frac{d\theta(t)}{dt} \qquad \textit{Eq (5-1)}$$

$$e_a(\mathrm{t}) = L_m \frac{di_a(t)}{dt} + R_m i_a(t) + e_m(t) \qquad \textit{Eq (5-2)}$$

$$T(t) = K_t i_a(\mathrm{t}) \qquad \textit{Eq (5-3)}$$

$$\mathrm{J}\frac{d\theta^2(t)}{dt^2} + \mathrm{B}\frac{d\theta(t)}{dt} = T(t) \qquad \textit{Eq (5-4)}$$

Where e_a(t) = armature voltage, e_m (t) = back emf, i_a(t) = armature current, $T(t)$ =developed torque, Θ$\theta(t)$ = motor shaft angle, $d\theta(t)/dt$ = ω(t) Θ= shaft speed, J= moment of inertia of the rotor, B = viscous frictional constant, Lm= inductance of armature windings, Rm= armature winding resistance, Kt= motor torque constant, Km= motor constant

Here the motor speed ω(t) is controlled by varying the armature voltage ea(t). Hence ea(t) is the input variable and ω(t) is the output variable.

We chose as the state variables

$$x_1(t) = \omega(\mathrm{t}) = \frac{d\theta(t)}{dt}\,,\; x_2(t) = i_a(t) \qquad \textit{Eq (5-5)}$$

The state equations will now be derived by using above equations.

$$\frac{dx_1(t)}{dt} = -\frac{B}{J}x_1(t) + \frac{K_t}{J}x_2(t) \qquad \textit{Eq (5-6)}$$

$$\frac{dx_2(t)}{dt} = -\frac{K_m}{L_m}x_1(t) - \frac{R_m}{L_m}x_2(t) + \frac{1}{L_m}e_a \qquad \textit{Eq (5-7)}$$

$$y(t) = \frac{d\theta(t)}{dt} = \omega(\mathrm{t}) = x_1(t) \qquad \textit{Eq (5-8)}$$

Hence state model of servo motor is derived from equations above as follows:

$$\begin{bmatrix} \frac{dx_1(t)}{dt} \\ \frac{dx_2(t)}{dt} \end{bmatrix} = \begin{bmatrix} -\frac{B}{J} & \frac{K_t}{J} \\ -\frac{K_m}{L_m} & -\frac{R_m}{L_m} \end{bmatrix} \begin{bmatrix} x_1(t) \\ x_2(t) \end{bmatrix} + \begin{bmatrix} 0 \\ \frac{1}{L_m} \end{bmatrix} \mathrm{u} \qquad \textit{Eq (5-9)}$$

$$\mathrm{y(t)} = \begin{bmatrix} 1 & 0 \end{bmatrix} \begin{bmatrix} \mathrm{x_1(t)} \\ \mathrm{x_2(t)} \end{bmatrix} \qquad \textit{Eq (5-10)}$$

so, this 1st order differential equation is the electrical side of the servo motor, its input is the voltage source Vs and current is created inside the motor and if the shaft is spinning then there is a back emf that opposes that input voltage to tune $k_e\theta m$.

for the mechanical side of the motor , it has an inertia associated with it that is part sticking out at the end of the motor ,but inside the motor housing itself is typically a little bit more inertia denoted as Jm that is a constant in addition to viscous damping coefficient, so the rotor could be spinning within the housing and one way to model the losses that occur is using a viscous damping effect Bm , so when voltage is turned on Vs it generates a current in this circuit and that is what causes this torque onto the motors rotor.

Let, the motor parameters (coefficient of differential equations) are assigned to be L_m = 1 H, k_m=1Nm/A, k_t=1 Vsec/rad, J=0.1 $Kg \cdot m^2$, B=0.5 Nmsec/rad, R_m= 5 Ω

.

CHAPTER 6 - SERVO MOTOR CONTROL

6 SERVO MOTOR CONTROL

Because of their simplicity, ease of application, reliability, and cost effectiveness, servo motors are employed in a variety of applications such as industries and robotics. The speed and position of the shaft of a servo motor may usually be controlled by adjusting the terminal voltage. The position control of a servo motor is critical in precision control applications. A motor position controller's job is to take a signal that represents the needed angle and operate a motor at that angle. A servo motor may be easily controlled with a microcontroller. An electronic component and a microprocessor make up a microcontroller-based position control system. There are a variety of servo motor drives that use power electronics to control the voltage and, as a result, the motor's speed, or position.

6.1 Proportional Integral Derivative Controller

The PID controller (proportional–integral–derivative) is a common control loop feedback mechanism in industrial control systems [46]. A PID controller tries to correct the difference between a measured process variable and a desired set point by calculating and then outputting a corrective action that can change the process. We were able to repair the error made by the motor and control the speed or position of the motor to the desired point or speed by integrating the PID controller with the motor.

Two types of locomotion are incorporated in Rollo within the scope of this work: human-like walking with alternating foot motion and simpler locomotion with the feet constantly parallel. As a result, a system with a controller is required to meet this need. For each leg, a PID controller is employed to react if there is a difference between the reference angular input and the output. PID tune in Simulink is used to calculate the PID controller's parameters. A general scheme of this controller is presented in Figure 6-1

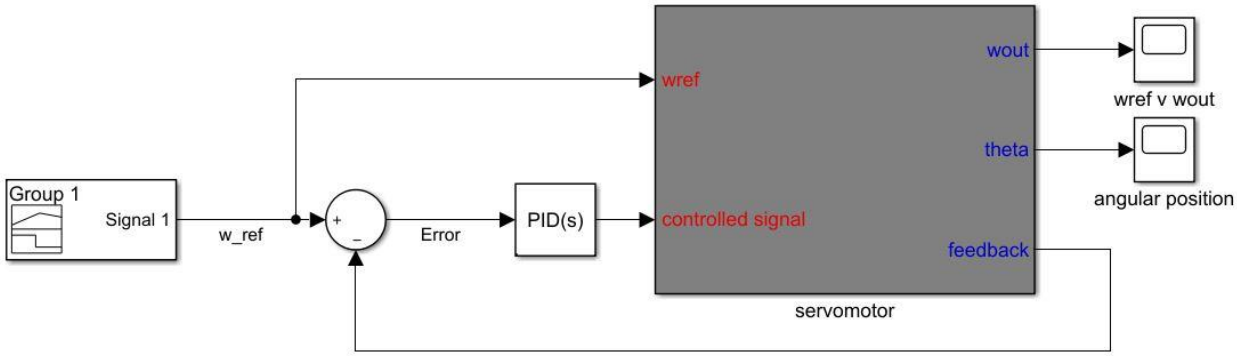

Figure 6-1 Scheme of the controller and servomotor in Matlab/Simulink

According to the PID tuning method, errors are not only solved but also taken to its minimal value with very low amount of error oscillations.

6.1.1 Equation

$$u(t) = K_p e(t) + K_i \int_0^t e(\tau)d\tau + K_d \frac{de(t)}{dt} \qquad \textit{Eq (6-1)}$$

Figure 6-2 shows the PID controller design for this control system. PID equation is calculated to get the control signal for servo motor.

From Eq (6-1), e(t) is position error value difference between set angle and output measured angle (actual angle), y(t) is the actual angle. Kp, Ki and Kd were respectively with the values of proportion, integral, and the differential coefficient.

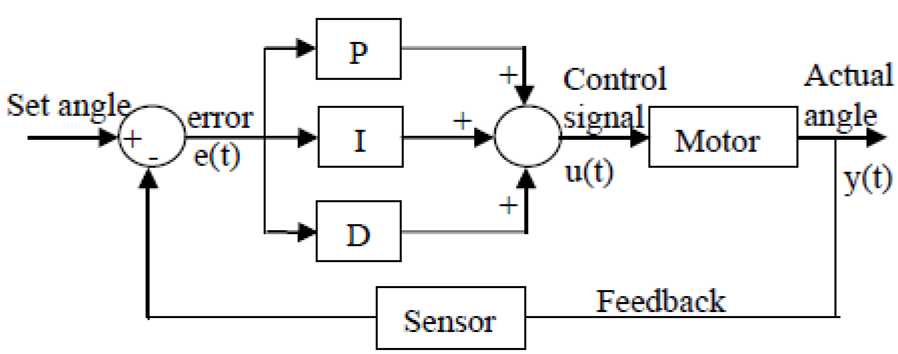

Figure 6-2 : Closed Loop Position Control of the servo Motor using PID Controller

The proportional gain (Kp) looks at the size of the error and responds proportionally. The magnitude of the motor position will obtain a large response, notwithstanding the large inaccuracy. The integral gain (Ki) tries to lower the steady state error. The derivative gain (Kd) aims to look at the error signal's rate of change. The overshoot will be reduced via derivative control, resulting in a larger system response curve of the motor position to a high rate of change.

Proportionate Reaction Only the difference between the set point and the process variable determines the proportional component. The Error word refers to this distinction. The proportional gain (Kc) is used to calculate the output response to error signal ratio. The speed of the control system response will be increased by raising the proportional gain. The process variable will begin to oscillate if the proportionate gain is too great. If Kc is raised even higher, the oscillations will grow greater, the system will become unstable, and it may even oscillate out of control.

Integral Reaction The error term is added up over time by the integral component. As a result, even a minor error term causes the integral component to steadily rise. Unless the error is zero, the integral response will continue to increase over time, driving the Steady-State error to zero. The final discrepancy between the process variable and the set point is known as steady-state error. When integral action saturates a controller without the controller moving the error signal toward zero, a phenomena known as integral windup occurs.

Derivative Reaction If the process variable is fast increasing, the derivative component causes the output to drop. The derivative response is proportional to the process variable's rate of change. By increasing the derivative time (Td) parameter, the control system will react more strongly to changes in the error term and the total control system reaction will be faster. Because the Derivative Response is particularly sensitive to noise in the process variable signal, most practical control systems use very short derivative times (Td). The derivative response can make the control system unstable if the sensor feedback signal is noisy or the control loop rate is too slow.

6.1.2 PID Tuning

Tuning is the process of determining the appropriate gains for P, I, and D in order to obtain an ideal response from a control system [47]. Trial and error can be used to determine the gains of a PID controller. This procedure becomes reasonably simple once an engineer understands the relevance of each gain setting. The I and D terms are set to zero first in this manner, then the proportional gain is increased until the loop's output oscillates. The system becomes faster when the proportional gain is increased, but care must be taken not to make the system unstable. The integral term is increased to end the oscillations once P has been set to the required quick response. The integral term decreases steady-state error while increasing overshoot. A rapid system must always have some amount of overshoot in order to respond to changes quickly. To get a small steady state error, the integral term is modified. The derivative term is increased until the loop is acceptable quick to its set point after the P and I have been set to get the required rapid control system with low steady state error. Increasing the derivative term reduces overshoot and offers higher gain with stability, but the system becomes extremely noise sensitive. Engineers frequently have to trade off one feature of a control system for another in order to meet their needs. According to the observed system response, the PID Gain tuning approach is a solid alternative for reducing oscillation.

After testing several experiments for tuning the PID controller, the best result for Kp, Kd and Ki values were achieved. In this condition, the values of the best result are Kp=83.2435, Kd=7.1338 and Ki=235.4019. But, the performance still occurs a little delay time between the desired and actual positions. See Figure 6-3

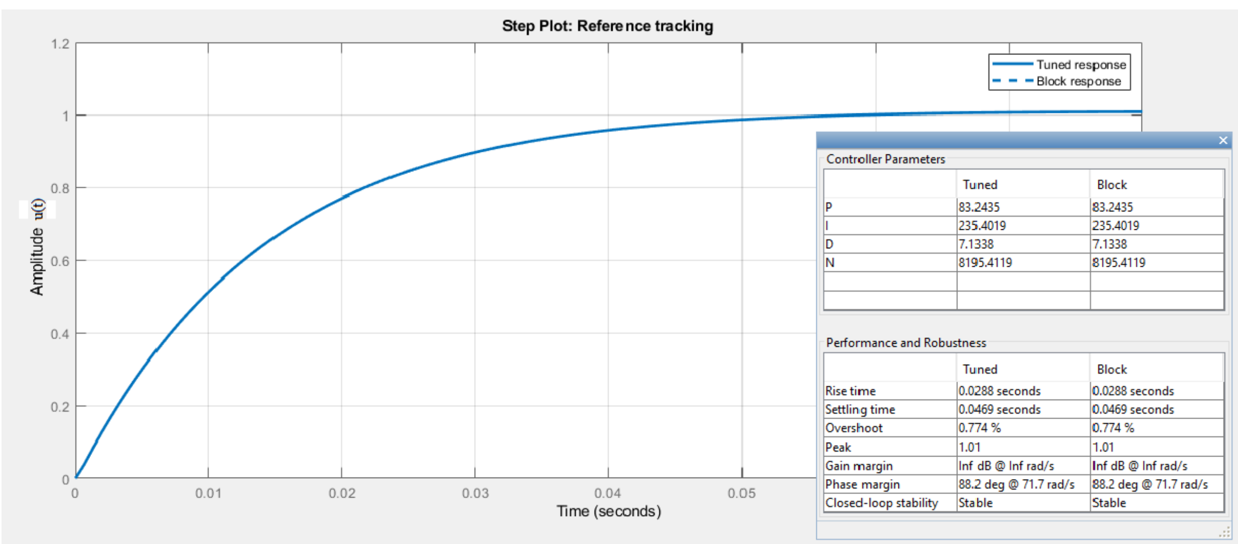

Figure 6-3 : Tuned PID parameters

The basic idea behind a PID controller is to read a sensor, then calculate the required actuator output by adding proportional, integral, and derivative responses. Before we begin to define the PID controller's parameters.

6.1.3 Closed Loop System

The process variable is the system parameter that needs to be controlled in a typical control system, and a sensor is used to measure the process variable and provide input to the control system. The set point is the desired or command value for the process variable. The control system algorithm (compensator) uses the difference between the process variable and the set point to determine the desired actuator output to drive the system at any given time (plant). Because the process of reading sensors to provide constant feedback and calculating the desired actuator output is repeated continuously and at a predetermined loop rate, this is referred to as a closed loop control system.

6.1.4 Control Design

The performance requirements are defined first in the control design process. The performance of a control system is frequently assessed by using a step function as the set point command variable and then observing the response of the process variable. Typically, the reaction is measured using waveform parameters that are prescribed. The rise time is the time it takes for the system to go from 10% to 90% of the steady-state, or final, value. The amount by which the process variable overshoots the final value, represented as a percentage of the final value, is known as percent overshoot. Settling time refers to the amount of time it takes for a process variable to settle to within a specified percentage (usually 5%) of its ultimate value. The final difference between the process variable and the set point is known as the steady-state error. After defining the performance criteria for a control system with one or more of these quantities, it's a good idea to identify the worst-case scenarios under which the control system will be required to achieve these design requirements. Frequently, there is a disturbance in the system that impacts the process variable or the process variable's measurement. It's critical to create a control system that works well even in the worst-case scenario. The disturbance rejection of the control system is a measurement of how well the control system can overcome the effects of disturbances. The system's response to a specific control output may vary over time or in proportion to some variable in some instances. A nonlinear system is one in which the control parameters that yield a desired response at one operational point may not yield a suitable response at another. The robustness of a control system refers to how effectively it can withstand shocks and nonlinearities. Deadtime is an unfavorable characteristic of some systems. A delay between when a process variable changes and when that change may be noticed is known as deadtime. A system or output actuator that is slow to respond to the control command can also produce deadtime. We begin with a plant, which we refer to as the system we wish to govern or whose behavior we want to influence. The actuation signal is the plant's input, and the regulated variable is the output. The primary concept behind a control system is to figure out how to provide the proper actuation signal, or input, in order for our system to produce the desired controlled variable, or output. The purpose is to provide the correct input into the system in order to obtain the desired output, which is referred to as the command, commanded variable, set point, reference, or desired value. In feedback control, the system's output is sent back (thus the name) and compared to the command to see how far away the system is from the desired state. The error term is the difference between the two. If the output was precisely what we wanted it to be, the error would be zero, which is exactly what we want. So, how do we take this error term and translate it into appropriate actuator commands so that the error is driven to zero over time? and the solution is to use a controller.

The following unity-feedback system will be considered:

The output of a PID controller, which is equal to the control input to the plant, is calculated in the time domain from the feedback error Eq (6-1)

First, using the diagram above, let's look at how the PID controller works in a closed-loop system. The tracking error (e) is the difference between the desired output (r) and the actual output (y). The PID controller receives this error signal e and computes both the derivative and integral of the error signal with regard to time. The proportional gain k_P times the magnitude of the mistake plus the integral gain k_i times the integral of the error plus the derivative gain k_d times the derivative of the error equals the control signal (u) to the plant.

The plant receives this control signal (u), and the new output (y) is obtained. The new output (y) is then sent back into the system and compared to the reference signal to determine the new error signal (e).

The transfer function of a PID controller is found by taking the Laplace transform of Eq (6-1)

$$K_p + \frac{K_i}{s} + K_d s = \frac{K_d s^2 + K_p s + K_i}{s} \qquad \textit{Eq (6-2)}$$

where k_p = proportional gain, k_i = integral gain, and k_d = derivative gain.

The Characteristics of the P, I, and D Terms

When the proportional gain (Kp) is increased, the control signal is correspondingly increased for the same degree of inaccuracy. Because the controller will "push" harder for a given degree of error, the closed-loop system will react faster, but it will also overshoot more. Another effect of increasing (Kp) is that it tends to lower the steady-state error, but not completely eliminate it.

The addition of a derivative term (Kd) to the controller increases the controller's ability to "predict" error. If (Kp) is fixed under basic proportional control, the only way the control will improve is if the error grows. Even if the amount of the error is still minor, the control signal can grow substantial with derivative control if the error begins to slant upward. This anticipatory behavior tends to dampen the system, reducing overshoot. The steady-state error is unaffected by the insertion of a derivative term. The addition of an integral term (Ki) to the controller reduces steady-state error. The integrator builds and builds if there is a continuous, steady error, raising the control signal and driving the error down. The integral term, on the other hand, has the disadvantage of making the system more slow (and oscillatory), because it takes a long time for the integrator to "unwind" when the error signal changes sign.

The general effects of each controller parameter (Kp, Kd, Ki) on a closed-loop system are summarized in the Figure 6-4 below. Note, these guidelines hold in many cases, but not all. If we truly want to know the effect of tuning the individual gains, we will have to do more analysis, or will have to perform testing on the actual system.

CL RESPONSE	RISE TIME	OVERSHOOT	SETTLING TIME	S-S ERROR
Kp	Decrease	Increase	Small Change	Decrease
Ki	Decrease	Increase	Increase	Decrease
Kd	Small Change	Decrease	Decrease	No Change

Figure 6-4 : Effects of the controller parameters

6.2 System Description

A system can be made up of many different components arranged in many different ways, but we'll start with the components and functions of a traditional closed-loop system [48].

Input: r - An input is a reference value that should be directly proportional to the output of the system. This may not always imply a voltage/power source, but rather a setting or switch for connecting a voltage/power source to the system. Our input signal in the system example we'll look at later will be a unitary step.

Controller: C - This is the PID controller that we shall design in our example. It's right after the input signal and feedback junction, just before the plant for which we're compensated.

Plant: G - This is a mathematical expression of all of our subsystems as a transfer function. If we're trying to control a motor, then the plant is the motor we're trying to control. We will induce the plant to react in such a way that the output value will be as close to our input as possible.

Output: y - This reading represents the system's real reaction to our desired response (input) after it has traveled through our built system (plant). Given that certain faults will occur and have been taken into consideration as acceptable, the system's performance is measured by comparing the output to the input.

Feedback: H - The feedback line in a system's equivalent block diagram introduces the system's output into the input. This means that any differences between the output (actual response) and input (desired response) may be assessed; in other words, the error between the input and output is what the controller is exposed to. The error of our system is consequently equal to input - output x H; if the system has unitary feedback (meaning H = 1), the error is just input minus output. When we describe how each aspect of the PID controller works, it's vital to keep this in mind.

It's worth noting that using the closed loop transfer function, the transfer function for the entire loop can be reduced to just one block with a single input and single output:

$$\text{Closed } - \text{Loop } (s) = \frac{C(s) \cdot G(s)}{1 + C(s) \cdot G(s) \cdot H(s)}$$

CHAPTER 7 - RESULTS

7 RESULTS

The simulation have been done on the servo motor with and without PID controller, our aim was to analyze the motion behavior of the servo motor before and after being controlled, by modelling it on simulink and adding the reference signal which is the angular speed or the angular position and start varying and changing the input signal according to the motion we want the rollo robot to achieve, for example In order to move the robot in a rectilinear direction and with a constant speed, the two gear motors must rotate with the same direction to achieve that the legs are moving together as shown in Figure 7-1

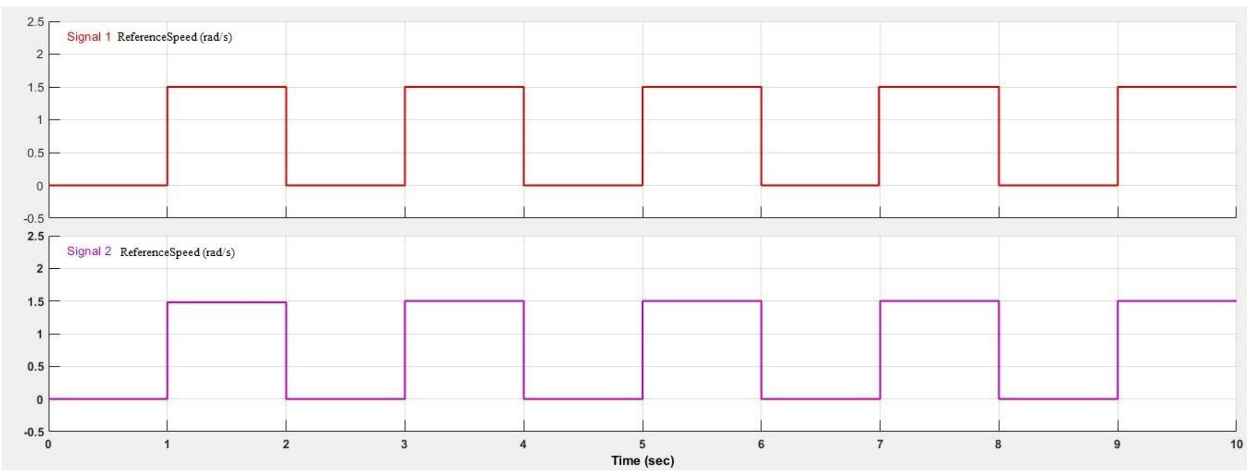

Figure 7-1 : Reference signal of the right and left leg to achieve a parallel rectilinear motion

the motors were stopped at the beginning and the robot was standing but at time 1 sec the motors start working and the angular speed start increasing form 0 to 1.5 rad/s and remains constant so that the wheels of the 2 legs start rotating together in the same direction and the robot start moving forward and in order to let the robot move backward the speed value of the motors must be the same and the direction of rotation must be the opposite).

and in order to achieve the alternate motion of the legs, in a way that resembles the human walking pattern the two gear motors must rotate with the same direction to move the robot in a rectilinear direction and with a constant speed as shown in Figure 7-2

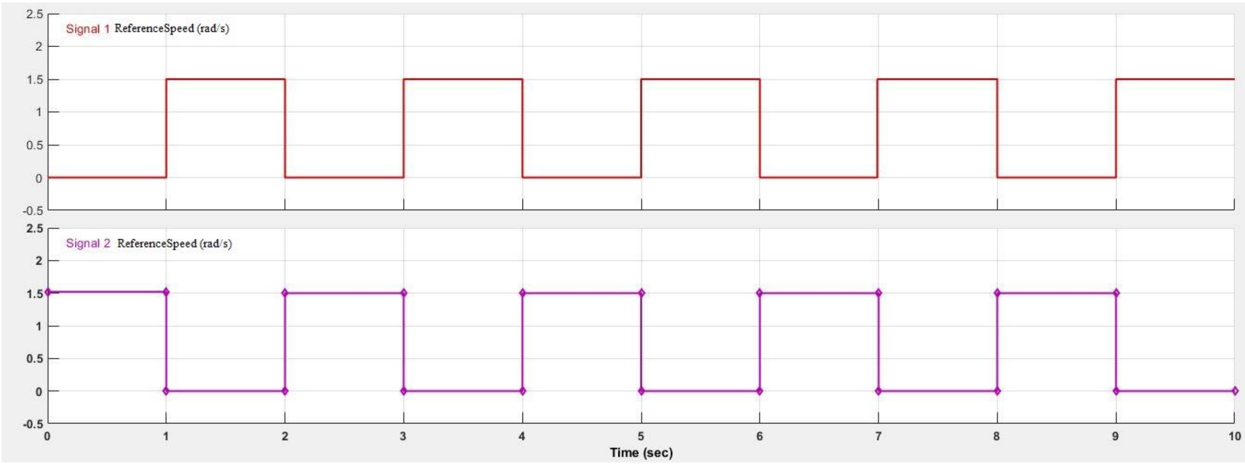

Figure 7-2 : Reference signal of the right and left leg to achieve a alternate motion

the motor of the left leg start working on time 0 sec and the speed decreases from 1.5 rad/s to 0 rad/s while the motor of the right leg was not working and at time 1 sec the speed increases to reach 1.5 rad/s the motor of the left leg stops and the motor of the right leg start working and so on , according to this process of the reference signal created the alternate motion of the legs will be achieved.

Also, the robot could rotate moving the wheels of each foot in the opposite direction. if the speed value of the gear motors is the same and the direction of rotation of each motor is opposite, the robot will rotate around an axis perpendicular to the plane of motion.

let us suppose that this rotation angle is theta θ. Now the actuation inputs to the robot or the individual wheel angular velocity is $\omega_L$and ω_r.

So, when we give ω_L and ω_r equal to each other, we are expecting the robot to move in a steady rate.

When the velocities are not equal to each other, we can imagine that the robot will rotate about an instantaneous center of curvature.

when ω_r is negative of. ω_L For this, we are expecting the robot to rotate about about its center of mass.

when ω_r in the same direction as ω_L, but of a different magnitude. Now we are expecting the robot to move in an arc in some sort of a circle

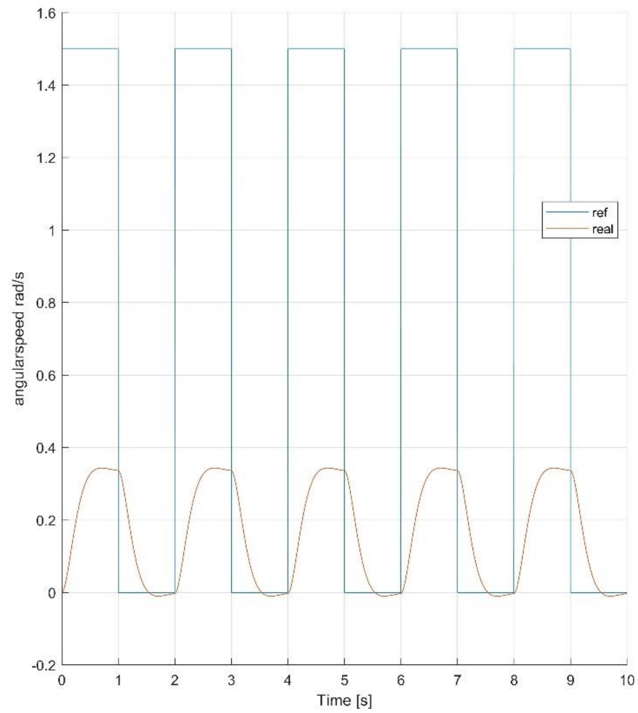

Figure 7-3 : Simulation without PID controller

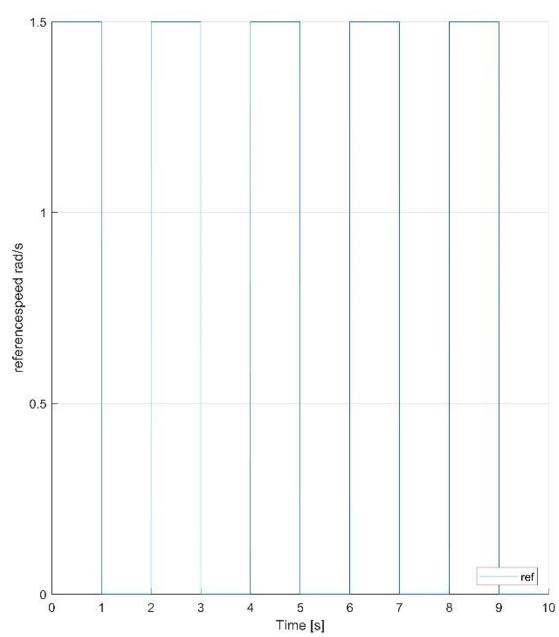

Figure 7-4 : Desired input signal

In Figure 7-3 which represent the reference angular speed and the output angular speed simulated without the PID controller, the desired input signal has the graphical description as shown in Figure 7-4, the output data is angular velocity on the scope compared to the reference signal. the reference angular velocity started at 1.5 rad/s at time 0 sec and it decreases to 0 rad/s at time 1 sec while the output angular velocity should track the input signal and should has damped behavior and should settle at certain steady state output but this never happened since the output signal started at 0 rad/s at 0 sec and increases till 0.3434 rad/s at 0.6 sec and didn't reach 1.5 rad/s as the desired input then it started decreasing at time 1.02 sec till it reaches -0.01 rad/s at time 1.7sec, and the same scenario continues, the output of the system is undamped and the system is not able to track the input angular speed , which means there is an error, so the difference between 2 lines is called steady state error (es) , the response is divided into transient response and steady state response . the transient response we can measure it by rise time, peak time, maximum overshot, and the steady state response we express it by steady state error.

To remove the error, add a PID to have zero steady state error and also closed loops system when we bring signal from the output and we make a negative feedback, we will have tracking of the reference speed, another property of the negative feedback closed loop system and the PID controller is rejection to any disturbance.

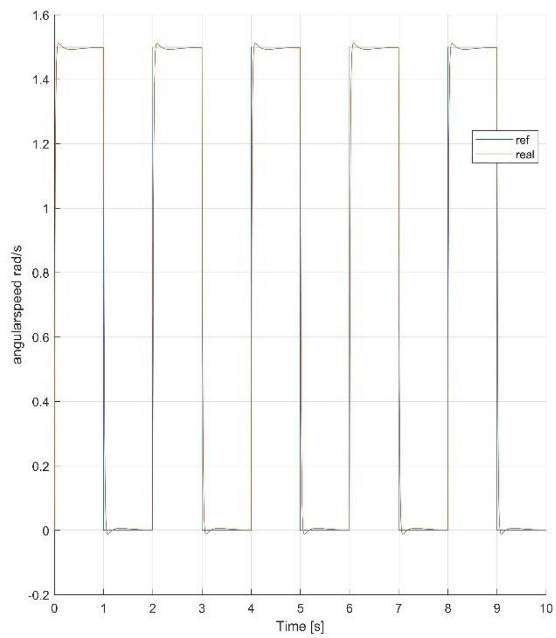

Figure 7-5 : Simulation with PID controller

Figure 7-5 shows the test result of the servomotor angular velocity controlled system. It can be seen the satisfied results. For this experiment, the input signal is used the magnitude of 1.5 rad/s for sqaure signal.

After testing several experiments for tuning the PID controller, the best result for Kp, Kd and Ki values were achieved. In this condition, the values of the best result are Kp=83, Kd=7 and Ki=235. But, the performance still occurs a little delay time between the desired and actual velocity.

The reference angular velocity started at 1.5 rad/s at time 0 sec and it decreases to 0 rad/s at time 1 sec while the output angular velocity started at 0 rad/s at time 0 sec and it increases till it reaches the maximum 1.512 rad/s at time 0.084 sec , and this slight increase above the reference signal represent the maximum overshot then the output signal at time 1 sec started decreasing to reach the minimum -0.012 rad/s at time 1.085 sec then the same scenario continues. The results were clearly seen, the controller output response curve is very well-matched to approach the desired speed.

so after adding a PID controller the output of the system is highly damped and that the system is able to track the control input, at the beginning there was a high overshoot that might destroy the motor so we tuned the parameter by changing transient behavior and response time which gave us the shape of the tracking response, the overshoot decreased. by varying the amplitudes of the different responses for our PID we can create a controller with the lowest overshoot and the best response time.

The PID output results were very precise to get the desired angle with stability performance.

After we make sure of the perfect simulation results of the servo motor, we connected it the axle of the wheels of each leg of rollo robot.

CHAPTER 8 - CONCLUSION

8 CONCLUSION

This work describes simple two-legged robot actuated by a commercial servomotor SG90. The purpose was to design a small version of Rollo robot with flexible legs and wheeled feet, then implementing it on simscape, modelling, controlling, and simulating the model of the servomotor and simulating the movement of the robot and this was achieved. Rollo is a biped, flexible-leg wheeled-feet robot that moved in a rectilinear and curvilinear manner over a planar surface. With only two actuated wheeled feet, proper dimensioning of flexible and rigid linkages provided additional advantages on biped locomotion, decreasing complexity of biped control, and allowing better stability during walking. The Rollo robot's reduced size, simple mechanical design, and minimal degree of freedom reduced the complexity of structures, actuation, and control needs. and the robot's ease of movement in a real setting could not limit its practical application.

9 References

[1] Yamaguchi, Takanishi and Kato, "Development of a biped walking robot compensating for three axis moment by trunk motion. Paper presented at the IEEE/RSJ international conference on intelligent robots and systems.," Yokohama, Japan, 26-30 July 1993.

[2] Nagasaka, Inoue and Inaba, "Dynamic walking pattern generation for a humanoid robot based on optimal gradient method. Paper presented at the IEEE international conference on systems, man, and cybernetics," 12-15 Oct. 1999.

[3] Sakagami, Watanabe, Aoyama, Matsunaga, Higaki and Fujimura, "The intelligent ASIMO: System overview and integration. Paper presented at the IEEE/RSJ international conference on intelligent robots and systems," Lausanne, Switzerland, 30 Sep.- 5 Oct. 2002.

[4] Kaneko, Kanehiro, Kajita, Hirukawa, Kawasaki, Hirata, Akachi and Isozumi, "Humanoid robot HRP-2. Paper presented at IEEE international conference on robotics and automation," New Orieans, LA, 26 April - 1 May 2004.

[5] I. Asimov, "Runaround," Astounding Science-Fiction, March 1942.

[6] Raibert, Blankespoor, Nelson and Playter, "Bigdog,the rough-terrain quadruped robot," IFAC Proc. 41, 10822–10825.doi: 10.3182/20080706-5-KR-1001.01833, (2008).

[7] Wang J, Lu K and Xu S, "Research situation and prospect on quadruped walking robot," Manuf Autom, 2009.

[8] Hutter, Gehring, Lauber, Gunther, Bellicoso, Tsounis, Fankhauser, Diethelm, Bachmann and Blösch, "“Anymal-toward legged robots for harsh environments,”," Advanced Robotics, vol. 31, no. 17, pp. 918–931, 2017.

[9] "Spotmini," [Online]. Available: https://www.bostondynamics.com/spot-mini.

[10] "Atlas," [Online]. Available: https://www.bostondynamics.com/atlas.

[11] "Asimo," [Online]. Available: http://www.asimo.honda.com.

[12] "Gita," [Online]. Available: http://www.vespa.com/us_EN/news-promo/Piaggio-Gita.html.

[13] "Rezero," [Online]. Available: http://www.rezero.ethz.ch.

[14] Righetti and Ijspeert, "Programmable central pattern generators: an application to biped locomotion control. In Proceedings of the 2006 IEEE International Conference on Robotics and Automation," pp. 1585–1590, Orlando, FL, USA, 15–19 May 2006.

[15] Ijspeert, "Central pattern generators for locomotion control in animals and robots: a review," Neural Netw. 21, 642–653, 2008.

[16] Ogura, Aikawa, Shimomura, Kondo, Morishima, Lim and Takanishi, "Development of a new humanoid robot WABIAN-2. In Proceedings of the 2006 IEEE International Conference on Robotics and Automation, ICRA 2006," pp. 76–81, Orlando, FL,USA, 15–19 May 2006.

[17] Muscolo, Recchiuto and Molfino, "Dynamic balance optimization in biped robots: physical modeling, implementation and tests using an innovative formula," Robotica 2015, 33, 2083–2099.

[18] Muscolo, Caldwell and Cannella, "Calculation of the center of mass position of each link of multibody biped robots.," Appl. Sci. 2017, 7, 724..

[19] "EMIEW 2-Hitachi's Humanoid Robot, Really Cute and Clever. 2010," [Online]. Available: https://www.youtube.com/watch?v=6Jb9oJY0Fe8 .

[20] "Dynamics, B. Introducing Handle. 2017," [Online]. Available: https://www.youtube.com/watch?v=-7xvqQeoA8c .

[21] "Dynamics, B. Introducing Handle," 2017. [Online]. Available: https://www.youtube.com/watch?v=-7xvqQeoA8c.

[22] "EMIEW 2-Hitachi's Humanoid Robot, Really Cute and Clever," 2010. [Online]. Available: https://www.youtube.com/watch?v=6Jb9oJY0Fe8 .

[23] Klemm, Morra, Salzmann, Tschopp, Bodie, Gulich, K¨ung, Mannhart, Pfister and Vierneisel, "“Ascento: A two-wheeled jumping robot,”," Proc. IEEE Int. Conf. Robot. Autom., pp 7515–7521,, May 2019..

[24] Oudeyer, O. Ly and P. Rouanet, "Exploring robust, intuitive and emergent physical human–robot interaction with the humanoid robot Acroban," in 11th IEEE–RAS International Conference on Humanoid Robots (Humanoids 2011), pp. 120–127.

[25] Muscolo and Recchiuto, "C.T. Flexible structure and wheeled feet to simplify biped locomotion of humanoid robots.," Int. J.Humanoid Robot. 2016, 14, 1650030., 2016.

[26] Calabrò, Portaro, Manuli, Leo, Naro and Bramanti, "Rethinking the robotic rehabilitation pathway for people with amyotrophic lateral sclerosis: A need for clinical trials," Innov. Clin. Neurosci. 2019, 16, 11., 2019.

[27] G. G. Muscolo and C. T. Recchiuto, "Propelled bipedal robotic apparatus. Property of the Humanots.r.l..," Patent Pending. 102015000049099, 2015.

[28] Muscolo G. G., Caldwell D. and Cannella F , "Biomechanics of Human Locomotion with Constraints to Design Flexible-Wheeled Biped Robots," in *AIM 2017 IEEE International Conference on Advanced Intelligent Mechatronics* , Munich, Germany, July 3-7,2017.

[29] Muscolo G. G. and Recchiuto C. T., "Flexible Structure and Wheeled Feet to Simplify Biped Locomotion of Humanoid Robots," *International Journal of Humanoid Robotics,* vol. 14, no. 1, p. 26, March 2017.

[30] Muscolo G. G., Caldwell D and Canella F, "Multibody Dynamics of a flexible Legged Robot with Wheeled Feet," in *ECCOMAS Thematic Conference on Multibodu Dynamics*, Prague,Czech Republic, June 19 - 22, 2017.

[31] G. G. Muscolo and C. T. Recchiuto, "Apparato robotizzato semovente bipede". Italy Patent 102015000049099 - ITUB20153452A1, 07/09/2015 - 7/03/2017.

[32] "Capitol Technology University," [Online]. Available: https://www.captechu.edu/blog/solidworks-mechatronics-design-and-engineering-program.

[33] Duindam and Stramigioli, "“Modeling and," Springer Tracts in Advanced Robotics (STAR), Vol. 53., 2009.

[34] McGeer, "“Passive dynamic walking”," International Journal of Robotics Research, 9(2): 62-82., 1990.

[35] Kajita and Espiau, "Legged Robots," Handbook of Robotics. Springer Part B, pp. 361-389..

[36] Vukobratov and Borovac, "Zero-moment point," Thirty-¯ve years of its life, Int. J. Hum. Robot. 1(1) (2004) 157–173..

[37] Vukobratovi´c and Borovac, "Zero-moment point - thirty five years of its life," Int. J. Hum. Robot 1, 157–173., 2004.

[38] L. Della Pietra, "The Dynamic Coupling of Torsional and Flexural Strains in Cylindrical Helical Springs," (Meccanica, Springer, New York, 1976)..

[39] A. M. Yu and Y. Hao, "Free vibration analysis of cylindrical helical springs with noncircular cross-sections," J. Sound Vib. 330 (2011) 2628–2639..

[40] C. J. Yang, W. H. Zhang, G. X. Ren and X. Y. Liu, "Modeling and dynamics analysis of helical spring under compression using a curved beam element with consideration on contact between its coils," Meccanica 49 (2014) 907–917, Doi: 10.1007/s11012-013-9837-1..

[41] "Fuji Electric," [Online]. Available: https://www.fujielectric.com/products/column/servo/servo_01.html.

[42] "Wheeled Robots," [Online]. Available: http://www.robotplatform.com/knowledge/Classification_of_Robots/Types_of_robot_wheels.html.

[43] "Simscape," [Online]. Available: https://it.mathworks.com/products/simscape.html.

[44] S. Miller, "Simscape Multibody Contact Forces Library. GitHub," 2020. [Online]. Available: https://www.github.com/mathworks/Simscape-Multibody-Contact-.

[45] "Simulink," [Online]. Available: https://it.mathworks.com/products/simulink.html.

[46] " PID Controller," [Online]. Available: https://www.elprocus.com/the-working-of-a-pid-controller/.

[47] "PID Theory Explained," [Online]. Available: https://www.ni.com/en-lb/innovations/white-papers/06/pid-theory-explained.html.

[48] "Control Systems," [Online]. Available: https://www.allaboutcircuits.com/technical-articles/an-introduction-to-control-systems-designing-a-pid-controller-using-matlabs/.

[49] Muscolo and Recchiuto, "C.T. Flexible structure and wheeled feet to simplify biped locomotion of humanoid robots," Int. J.Humanoid Robot. 2016, 14, 1650030.,, 2016.