PEOPLE'S DEMOCRATIC REPUBLIC OF ALGERIA

MINISTRY OF HIGHER EDUCATION AND SCIENTIFIC RESEARCH

UNIVERSITY OF GHARDAIA

N° d'ordre :

N° de série :

Science and Technology faculty

Department of Automatics and Electromechanics

A thesis submitted in fulfillment of the requirements for the degree of

MASTER

Theme:

STUDY AND DESIGN OF SELF-BALANCING ROBOT

by

Omar ABDESSELAM

Field: *Sciences and Technologies*

Sector: *Automatic*

Specialty: *Automation and systems*

Presented publicly on 15/06/2021

In front of the jury:

		Univ. Ghardaïa	**President**
Charaf Abdelkarim Mosbah	MCB	Univ. Ghardaïa	**Examiner**
FIHAKHIR Amine	MCB	Univ. Ghardaïa	**Examiner**
HACENE Nacer	MCB	Univ. Ghardaïa	**Supervisor**

Academic year 2020/2021

Dedication

To my parents for being the best parents I could ever ask for..

To my teachers whom I owe everything I've learned throughout my studying career..

To my friends and family for being supportive..

Acknowledgement

""I would like to thank ***Dr. HACENE Nacer*** *for his expert advice and encouragement throughout this challenging project.*

I should also mention all the teachers and professors of Ghardaia University that have taught me through the past five years.

I'd like to also thank ***Erinov foundation*** *for their supply with the needed materials and equipment*

And whoever provided help what so ever in this thesis process and project, and my colleagues as well"

ملخص

تهدف هذه الأطروحة إلى تصميم وبناء روبوت ذاتي التوازن.

تم تصميم الروبوت، حيث تم تطوير الحركية والديناميكيات بطريقة مفصلة. تعتمد ديناميكيات الروبوت على ميكانيكا لاغرانج. تم بناء واختبار حركيات وديناميكيات الروبوت باستخدام برنامج الماتلاب،.وقد تم اشتقاق معادلات الحركة الخطية ونمذجة النموذج العام للروبوت ذي العجلتين ذاتي التوازن في فضاء الحالة.

تم استخدام نوعين من وحدات التحكم ؛ وحدة تقوم على المتحكم PID و أخرى على المنطق الغامض، وكلاهما له إيجابياته وسلبياته.

تم بناء نموذج أولي للروبوت حيث يتم تشغيل الروبوت بواسطة لوحة آردوينو.

الكلمات المفتاحية: روبوت ذاتي التوازن، بندول عكسي، النموذج الحركي، النموذج التحريكي، فضاء الحالة، متحكم PID، التحكم الغامض، أردوينو.

Abstract

This thesis aims to design and build a self-balancing two-wheeled robot. The robot has been modeled, where kinematics and detailed dynamics are developed.

The robot's dynamics are based on Lagrangian mechanics. The kinematics and dynamics of the robot were built and tested using Matlab Simulink. Linearized equations of motion are derived and the overall model of the two-wheeled self-balancing robot is represented in state space.

Two types of controllers were used; PID and Fuzzy Controller, and both had their pros and cons.

A prototype of the robot has been built. the robot is driven by an Arduino board.

Keywords: Self-balancing robot; inverted pendulum; kinematics; dynamics; state space; PID; Fuzzy control, Arduino.

Résumé

Ce mémoire vise à concevoir et construire un robot à deux roues auto-équilibré.

Le robot a été modélisé, où la cinématique et la dynamique détaillée sont développées. La dynamique du robot est basée sur la mécanique lagrangienne. La cinématique et la dynamique du robot ont été construites et testées à l'aide de Matlab Simulink. Des équations de mouvement linéarisées sont dérivées et le modèle global du robot auto-équilibré à deux roues est représenté dans l'espace d'état.

Deux types de contrôleurs ont été utilisés ; PID et Fuzzy Controller, et les deux avaient leurs avantages et leurs inconvénients.

Un prototype du robot a été construit. Le robot est piloté par une carte Arduino.

Mots clés : Robot auto-équilibré ; pendule inversé ; cinématique ; dynamique ; espace d’état ; PID ; Contrôle flou, Arduino.

Table of Contents

Table of Abbreviations and symbols

TWSBR	Two-Wheel Self-Balancing Robot
PID	Proportional Integral Derivative
WMR	Wheeled Mobile Robots
AIV	Autonomous Intelligent Vehicles
DOF	Degree of Freedom

General Introduction

General introductory

The self-balancing robot is a two-wheeled structure. The idea behind it can be presented this way; When the structure leans forward or backward at a certain angle, it should drive both wheels in the same direction (i.e., forward or backward) at a speed and acceleration proportional to this angle to stay in the vertical position. The robot can detect the tilt using different sensors and then drives the wheels using two motors. All calculations and processing are done by some kind of digital device like a microcontroller unit. Another feature of the self-balancing robot is that it can drive the two wheels independently of each other. In this case, it can turn left or right or even turn around its own center.

These robots have the ability to carry and balance different objects on them without losing their balance. And some other designs can provide wireless communication for remote control. These robots are commercially available as human powered vehicles well known as Segway's PT (the name Segway is derived from the word segue, which means smooth transition. PT is short for Personal Transporter). Another example could be the hoverboard for entertainment purposes.

The idea of the self-balancing robot is related to the inverted pendulum carriage (carriage and pole) which is a classic problem in dynamics and in control theory. While a normal pendulum is stable when suspended downward, an inverted pendulum is inherently unstable and must be balanced actively, precisely, and quickly in order to remain upright. This is the case with the self-balancing robot.

Trying to stay stable and recovers its old position when opposed to an external force. Seeking stability while moving between two points is another difficult question and can lead to a loss of consistency and as a result everything will fall apart. The problem is to ensure a fast and precise response in order to obtain high stability and performance and also to reliably find its previous location. The two-wheeled and self-balancing robot belongs to a multivariable, nonlinear, high order, strong coupling, and unstable essential motion control system, and it is a typical device of testing various control theories and control methods; therefore, the research has great theoretical and practical significance. Because it has the advantages of simple structure, stable running, high energy utilization rate, and strong environmental adaption, it has the broad application prospects whether in the military field or in the civilian field. Since 1980s, the scholars of various countries have conducted the system research on the two-wheeled self-balancing robot.

The objective of this thesis is to design and build a self-balancing two-wheeled robot. First, the mathematic modeling of the robot has been established. Then, two types of control are proposed: PID and fuzzy control. A prototype was designed and built based on an Arduino board and a gyroscope for sensing. The thesis is organized as follows:

After the introduction, this thesis is broadly structured in four chapters:

The first chapter gives a brief overview of the fundamentals of the navigation, perception, localization, mapping and navigation (path planning) of mobile robots.

The second chapter deals with the modeling of the two-wheeled self-balancing robot, where the kinematics and dynamics are established. the robot model is built and tested in Matlab.

The third chapter proposes two controllers to stabilize the robot namely a PID and a fuzzy controller.

The last chapter is dedicated to the realization of a prototype of the robot based on an Arduino board.

Finally, the conclusions of this work and the possible extensions were discussed.

CHAPTER I

A REVIEW OF MOBILE ROBOTS: CONCEPTS, METHODS, AND APPLICATIONS

Chapter I A review of mobile robots: Concepts, methods, and applications

I.1 Introduction

Nowadays, mobile robotics is one of the fastest expanding fields of scientific research. Due to their abilities, mobile robots can substitute humans in many fields. Applications include surveillance, planetary exploration, patrolling, emergency rescue operations, reconnaissance, petrochemical applications, industrial automation, construction, entertainment, museum guides, personal services, intervention in extreme environments, transportation, medical care, and so on, as well as many other industrial and nonindustrial applications. Most of these are already available on the market.

Mobile robots can move autonomously (in an industrial plant, laboratory, planetary surface, etc.), that is, without assistance from external human operators. A robot is autonomous when the robot itself has the ability to determine the actions to be taken to perform a task, using a perception system that helps it. It also needs a cognition unit or a control system to coordinate all the subsystems that comprise the robot. The basics of mobile robotics consist of the fields of locomotion, perception, cognition, and navigation. [1]

Locomotion problems are solved by understanding the mechanism and kinematics, dynamics, and control theory. Perception involves the areas of signal analysis and specialized fields such as computer vision and sensor technologies. Cognition is responsible for analyzing the input data from sensors and taking the corresponding actions to achieve the objectives of the mobile robot. It is in charge of the control system scheme. Navigation requires knowledge of planning algorithms, information theory, and artificial intelligence.

I.2 Locomotion

The robot's locomotion system is an important aspect of the mobile robot design and it depends not only on the medium in which the robot moves (on the Earth's surface, under water, in the air, etc.) but also on technical criteria such as maneuverability, controllability, terrain conditions, efficiency, stability, and so on.

Depending on it, robots can mainly walk, roll, jump, run, slide, skate, swim, and fly. According to their locomotion system, mobile robots can be classified into the following major categories:

I.2.1 Stationary robot (arm/manipulator)

Manipulators and industrial robots are examples of this type. The robot's base is fixed and they consist of an open kinematic chain, mainly with an end-effector with special tools which not only handle objects but can also perform tasks such as welding, painting, assembling, machining, and so on.

Figure I.1: *Industrial arm robot*

I.2.2 Land-based robots

These can be classified as follows:

Wheeled mobile robots

Wheels are one of the most important systems for robot locomotion, and autonomous intelligent vehicles (AIVs) are part of a challenging research field in mobile robotics, which relies on principles such as pattern recognition and signal–image processing. They will play an important role in transport, logistics, and distribution. The use of wheels is simpler than using treads or legs and is easier to design, build, and program when the robot is moving on flat, nonrugged terrain. They also tend to be much cheaper than their legged counterparts. Wheel control is less complex and they cause less wear and tear on the surface where they move in comparison with other solutions. Another advantage is that they do not present any great difficulty in terms of balance issues, since the robot is usually in contact with a surface. The main disadvantage of wheels is that they are not very good at navigating over obstacles, such as rocky terrain, sharp surfaces, or areas with low friction. There are four basic wheel types: [2] [3]

- Fixed standard wheel: These are conventional wheels with one degree of freedom (DOF), rotation around the contact point.
- Castor wheel: It has two DOF and turning around an offset steering joint.
- Swedish wheel: It has three DOF, revolving around the driven wheel axle, around the contact point and the rollers.
- Ball or spherical wheel: Its implementation is technically complex.

Figure I.2: *Fixed standard wheel*

Figure I.3: *Castor wheel*

Figure I.4: *Swedish wheel*

Figure I.5: *Spherical wheel*

It is very important to know the number and type of wheels in the robot to model the kinematics and dynamics. [4]

In general, wheeled robot research tends to focus on the problems of traction and stability, maneuverability, and control. [5] Stability is not usually a great problem in wheeled robot, because they are almost always designed so that all wheels are in contact with the ground at all times. Thus, three wheels are sufficient to guarantee stable balance, although two-wheeled robots can also be stable. [6] [7]

WMRs can also be classified according to the drive system:

- Differential drive WMRs
- Car-type WMRs
- Omnidirectional WMRs
- Synchro drive WMRs

It is very important when modeling any robot to know the number and types of wheels. According to the number of wheels, the robots can be classified as follows:

Single-wheeled robots

Unicycle robots have only one fixed or conventional wheel. The unicycle system is an inherently unstable system. Both longitudinal and lateral stability controls are needed simultaneously to maintain the unicycle's posture. [8]

Figure I.6: *Unicycle robot*

Two-wheeled robots

These have two alike parallel, conventional wheels (linked to both sides of the robot), which are controlled by two independent actuators. It is also considered that every wheel is perpendicular to the ground and the contact between the wheels and the ground is non-slipping and pure rolling. [9] The Roomba vacuum cleaner is a two-wheeled robot which is a good exponent of mobile robotics in recent household applications. It utilizes a contact sensor at the front and an infrared sensor on top. Segways can also be considered a self-balancing, two-wheeled robot.

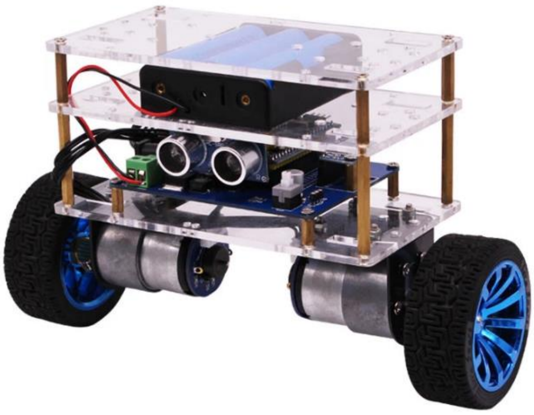

Figure I.7: *two-wheeled self-balancing robot*

Three-wheeled robots

Two types of three-wheeled robots can be distinguished: firstly, differentially steered (two driven wheels with an additional free turning wheel to maintain the vehicle in balance) and secondly, two wheels driven by a single actuator and a driven steering for the third wheel. [10]

Figure I.8: *Three-wheeled robots*

Four-wheeled robots

These robots are more stable than the three-wheeled counterpart, because the center of gravity (COG) is located inside the rectangle formed by the four wheels rather than a triangle. The wheels can be differentially steered (like rovers), two-by-two powered wheels (like a tank motion), or can have car-like steering. Car-like mobile robots are very important. Google's self-driving car, AIVs. They are acquiring a great importance in transports, logistics, food industry, and food processing

Figure I.9: *Four-wheeled robot*

Six-wheeled robots

When more powered wheels are used, the design becomes much more complex. The Sojourner robot from the Mars Pathfinder mission in 1997, [11] Spirit and Opportunity in 2004 and Curiosity in 2012, [12] are good examples of six-wheeled robots. They have a suspension system which keeps all six wheels in contact with the surface and helps them go over slopes and sandy terrain. Good control is needed to avoid slipping.

Figure I.10: *mars rover "curiosity"*

More than six wheels

The Octopus robot is a WMR that is able to deal with obstacles autonomously on rough terrain without getting stuck. It is equipped with tilt sensors and tactile wheels. [13]

Walking or legged mobile robots

Legs are another common form of locomotion, giving rise to walking robots. Although they are usually more expensive than wheels, legs have several advantages over wheels. The greatest advantage is their transversality and efficiency and the fact that they can also move on soft and uneven terrain, better mobility, better energy efficiency, better stability, and a smaller impact on the ground. Walking robots also have the advantage of easily coping with obstacles or cracks found in the environment [14]; in short, adaptability and maneuverability on rough terrain. There are many types of walking robots depending on the number of legs. Among the most important are biped (humanoids), four-legged (quadruped), six-legged, and so on.

Nowadays, there are many biped robots that possess the capability to climb over surfaces with different slopes. Here are some of the most common legged robots:

1) One-legged robot

“Hoppers” are one-legged robots. The greatest challenge with hoppers is that they cannot stand still; they need to keep hopping in order to maintain their balance. Uniroo is an example of a one-legged robot.

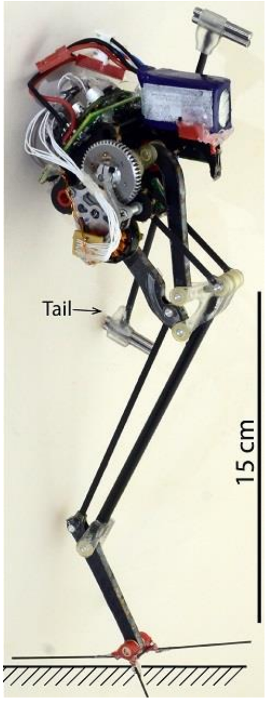

Figure I.11: *Uniroo one-legged robot*

Two-legged or humanoid robots

One of the most important types of walking robots is humanoid robots. In order to 4 International Journal of Advanced Robotic Systems reproduce human capabilities, they make good use of sensors. They can walk, talk, reproduce emotions, and so on. One of the most important problems from the point of view of locomotion is loss of balance.

The motion of biped robots is dependent upon dynamic stability. Two-legged (biped) robots can walk, run, travel up and down stairs, jump, and even do somersaults. The motions are sophisticated because the feet are quite small and the balance has to be dynamic at all times; even standing still requires sophisticated control. Humanoid robots that present autonomous locomotion are made of the following modules: local path planning based on observation of the environment, global path planning using given geometrical information, footstep planning, and a motion planner. [15]

Figure I.12: *Two-legged robot*

Three-legged robots

These are not very common because of the odd number of legs. STriDER is an example of a three-legged robot. [16] It presents a straightforward kinematic structure that is inherently stable like a camera tripod, it is easy to control, energy efficient, and lightweight. In the course of a step, two legs behave as supportive legs while the other works as a swing leg. The aim of the legs is to push the COG outside of the supportive legs to start a step. While the robot´s body falls forward, the swing leg naturally swings in between the two supportive legs and catches the fall.

Figure I.13: *STriDER robot*

Four-legged (quadruped) robots

When increased safety or payload capability is needed, quadrupeds or robots with a larger number of legs are used. They have the advantage of being statically stable when not moving, but they require dynamic walking control, as the robot's COG must be readily shifted during gait. The control and leg coordination of these larger robots is, however, more complicated. These systems require a high computational speed. The motors and power storage system required for these systems are highly expensive. A good example of a quadruped robot is BigDog. [17] BigDog can run at 6.5 km per hour, climb 35 slopes, and carry 150 kg. But the most impressive feature is its dynamic walking; BigDog can recover from slipping and even being pushed.

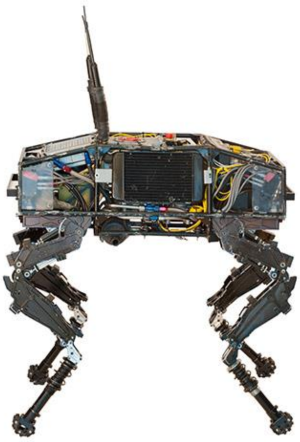

Figure I.14: *BigDog robot*

Five-legged (quadruped) robots

This kind of robots are rarely developed due to the odd number of legs. A five-legged robot inspired from starfish is presented by Besari et al. [18] One of the objectives of the article is to find the optimal gait when robot is walking. Trial and error have been used to provide learning through an interaction between the robot and the work environment.

Six-legged (hexapod) robots

A robot can be statically stable on three or more legs, so a robot with six or more legs can be controlled with static walking techniques rather than dynamic walking, thus reducing the control complexity. A common architecture for walking robots is the hexapod. These kinds of robots have been very popular in mobile robotics. [19] Hexapod gaits are typically stable, even on slightly rocky and uneven terrain. There are several hexapod gaits which can be suitable in different environments; for example, one leg at a time or quadruped gait. If one or two legs become disabled, the robot is still able to walk.

Figure I.15: *Six-legged robot*

More than six-legged robots

The most commons in this category are the eight-legged robots. They are inspired by spiders, underwater walkers, and other arachnids. An example is the Dante robot. It was designed with the aim to rappel into and explore active volcanic craters. It was used in an expedition to explore Mount Erebus in Antarctica. The objectives were to demonstrate its capabilities in a real exploration mission, in rough terrains, and its survival in the cold, windy, bright, rugged Antarctic environment.

Figure I.16: *the Dante robot*

Tracked robots

These are a type of robots that uses treads or caterpillar tracks instead of wheels. Other possible solution to steer, might be used to redirect the robot by spinning wheels with the same direction at different speeds or in opposite directions. The Nanokhod rover [20] operates in this way. Tracked robots have much larger ground contact patches, and this fact plays a major role to improve their maneuverability on loose surface in comparison to conventional wheeled robots. Nevertheless, because this large ground contact patch, changing the direction of the robot normally needs a skidding turn, and therefore a large portion of the track must slide against the surface. The skid/slip steering is also a disadvantage of such configurations. Due to the great amount of skidding in the course of a turn. Power efficiency of this approach is reasonably acceptable on loose surfaces but very inefficient otherwise.

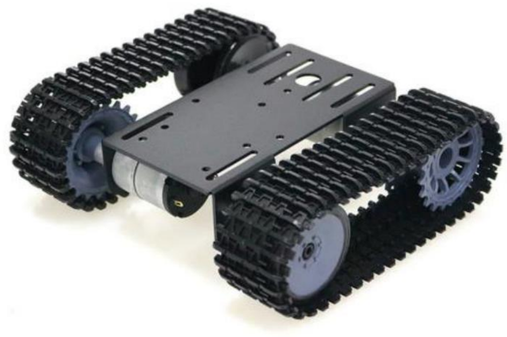

Figure I.17: *Tracked robot*

Hybrid robots

These are robots whose structure consists of a combination of any of the above-described types. For example, a segmented articulated and wheeled device. Hybrid solutions combine the suitability of wheels with the adaptability of legs. According to Bruzzone1 and Quaglia, [20] four categories of hybrid mobile robots can be distinguished:

- leg-wheel hybrid locomotion systems
- leg-track hybrid locomotion systems
- wheel-track hybrid locomotion systems
- leg-wheel-track hybrid locomotion systems

AIV-based robots

An unmanned aerial robot commonly known as “drone” is a machine that performs a preprogrammed task with or without human interaction and it is inspired by an airplane’s operation. The most advanced ones can now take off and land completely independently of the actions of their operators.

Figure I.18: *drone robot*

Water-based robots

Many devices have been built for this, including robotic systems. OceanOne is an example of a submarine robot. It is a humanoid robot that explores the seabed. It takes advantage of the best of remotely operated vehicles and the advantages of humanoid robots, such as having a robotic hand with which to rescue objects as if it were a human being. [21]

Figure I.19: *OceanOne robot*

Other robots

There are other robots, which are difficult to frame from the point of view of their motion, such as:

- Snake-like robots
- Worm-like robots
- Nanorobots
- Cooperative robotics
- Cooperative nanorobotics

I.3 Perception

It is vital for an autonomous mobile robot to acquire knowledge about its work environment and itself. This is achieved by means of sensors and subsequently extracting relevant information from those sensor's measurements. The use of sensors makes it possible to perform robot positioning and localization tasks. They are also used for mapping and representation. In addition, they are very useful in other robotic applications, such as object recognition. The latest advances in sensoring and artificial intelligence are being used in speech recognition systems, which are very important to reproduce human capabilities.

I.3.1 Sensor classification

There are a great variety of sensors that can be used in a robot for data collection. They can be grouped into:

a) Proprioceptive and Exteroceptive sensors:

Proprioceptive sensors read values internal to the robot, such as motor speed, wheel load, joint angles, battery voltage, and so on. Exteroceptive sensors acquire information from the robot's environment, such as distances, light intensity, and sound amplitude.

b) Passive and Active sensors:

Passive sensors measure ambient environmental energy entering the sensor, such as microphones, temperature probes and cameras. Active sensors radiate energy into the surroundings and then measure the reaction. They can deal with more controlled interactions with the work environment, thus achieving greater performance, although they may suffer from interference between their own signal and external ones.

I.3.2 Types of sensors

The most common sensors used in robotics are as follows:

- **Tactile sensors:**

These are designed to sense objects at a small distance with or without direct contact and are used to detect physical contact or closeness.

- **Force torque sensors:**

The robot uses a force torque sensor to know what force the robot is applying. Different robot tasks such as assembly, hand-guiding, teaching, and force limitation can be performed with this device.

- **Encoders:**

To know the robot's part position and speeds. There are a great variety (such as optical encoders, potentiometers, resolvers, inductive encoders, magnetic encoders, capacitive encoders).

- **Infrared sensors:**

These are light-based sensors.

- **Ultrasonic sensors:**

These are sound-based sensors and are used as distance meters. These sensors are designed to generate high-frequency sound waves and receive the echo reflected by the target.

- **Sonar:**

This can be used primarily for object detection. This sensor offers high performance on land and in water.

- **Active beacons:**

To help the robot navigate. There are two distinctive kinds of beacon systems: trilateration and triangulation.

- **Accelerometers:**

To measure acceleration, from which velocity can be obtained by integration. Humanoid robots use accelerometers

- **Gyroscopes:**

These are well-known and reliable rotation sensors, which measure angular velocities and orientation.

- **Laser range finder:**

This is a device which uses a laser beam to generate highly precise distance measurements. The distance between sensor and target is measured by calculating the speed of light and the time it takes for emitted light to return to the receiver.

- **Vision-based sensors:**

These process data from any modality and use the electromagnetic spectrum to produce an image. The two current technologies for devising vision sensors are CCD and CMOS.90

- **Color-tracking sensors:**

These make it possible to detect and track color in the environment.

- **Contact and proximity sensors:**

To measure the force of contact with the environment.

- **Pressure sensors**

- **Depth sensors:**

They are used in object detection, scene reconstruction, 3-D inspection, and so on. Two elements must always be present in a depth sensor: an infrared (IR) projector and an IR camera.

Figure I.20: *some robot sensors*

I.4 Cognition and control system

The mechanical structure of a mobile robot must be controlled to perform tasks and achieve its objectives. The control system involves three different pillars:

- perception
- processing and cognition
- action

The perception system gives information about the environment, the robot itself, and the relationship between robot and environment. This information is processed, then the appropriate commands are sent to the actuators, which move the mechanical structure. Once the environment and the robot’s direction, destination, or purpose are known, the cognitive architecture of the robot must plan the path that the robot must take to achieve its objectives.

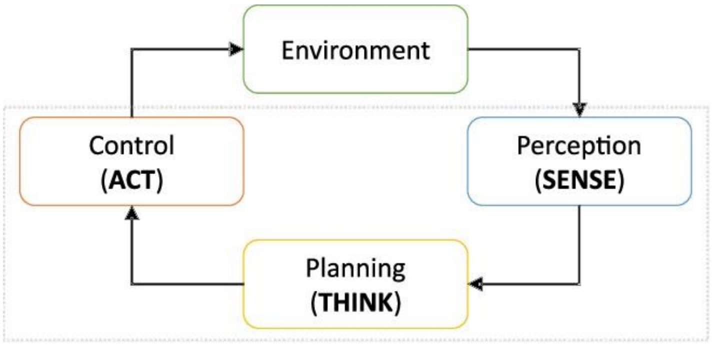

Figure I.21: *robot perception, cognition and action scheme*

The best-known control strategies are described next:

- Computed torque control methods
- Robust control methods
- Sliding mode control methods
- Adaptive methods
- Neural networks methods
- Fuzzy logic methods
- Invariant manifold method
- Zero moment point control

I.5 Navigation

The most important aspect in the design of a mobile robot is navigation skills. The objective is for the robot to move from one place to another in a known or unknown environment, taking into account the values of the sensors to achieve the desired targets. This means that the robot must rely on its other aspects, such as perception (the robot must use its sensors to obtain valuable data), localization (the robot must know its position and configuration), cognition (the robot must decide what to do to achieve its goals), and motion control (the robot must calculate its input forces on the actuators to achieve the desired trajectory). Most of the time, the mobile robot cannot take the direct path from its initial position to the final goal, which means that motion planning techniques must be used. Mobile robot navigation is categorized into the following tasks:

- Generating a model of the world in the form of a map.

- Computing a collision-free trajectory from a starting position to a target position.
- Moving along the calculated trajectory, avoiding collision with obstacles.

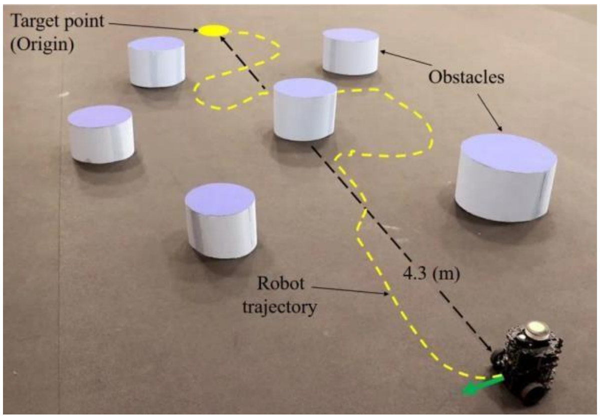

Figure I.22: *robot trajectory*

I.5.1 Navigation skill

Some essential skills are required for mobile robot navigation. The first of these is trajectory planning; Given a map and an objective location, it involves obtaining the trajectory that the robot must follow in order to reach the objective location. The second skill is obstacle avoidance; This plays an important role in trajectory planning in order to avoid collisions. The data from the sensors may modulate the robot's trajectory for avoiding collisions.

I.5.2 Localization and mapping

In order for the robot to navigate successfully, it must determine its position in the workplace. So, localization together with perception and motion control are key issues in robot navigation. Localization is closely related to representation. If an accurate GPS system could be installed on a robot, the localization problem would be solved. The robot would always know where it was. But at the moment, this system is not available or is not accurate enough to work with. In any case, localization implies not only knowing the robot's absolute position on Earth but also its relative position with respect to a target.

a) Positioning

The Exact knowledge of the position of a moving particle is a key issue in mobile robot applications. In seeking a solution, researchers and engineers have elaborated a diversity of systems, sensors, and methods for mobile robot positioning. Techniques and technologies used for positioning are as follows:

- odometry
- inertial navigation
- magnetic compasses
- active beacons
- global positioning systems
- landmark navigation
- model matching

Map representation

There are many techniques for creating a map representation. Some are closely related to path planning algorithms. The occupancy grid (quadtree) representation or a topological approach are two important techniques for map representation. A continuous-valued map is also one technique for a decomposition of the environment.

Many map representation techniques have been used in path planning. A set of techniques are as follows:

- Probabilistic map–based localization
- Markov localization
- Kalman filter localization
- Monte Carlo localization
- Landmark-based navigation
- Globally unique localization
- Positioning beacon systems
- Route-based localization
- Autonomous map building
- The stochastic map technique

- Cyclic environments
- Dynamic environments

I.5.3 Path, trajectory, and motion planning

Path planning is concerned with finding the best path in order for the mobile robot to reach the target without collision, thus allowing a mobile robot to navigate through obstacles from an initial configuration to another configuration. The trajectory planning entails to find out the force inputs (control u(t)) to move the actuators so that the robot follows a trajectory q(t) that enables it to go from the initial configuration to the final one while avoiding obstacles. It takes into account the robot's dynamics and physical characteristics to plan the trajectory. In short, the temporal evolution of the motion is calculated as well as the forces needed to achieve that motion.

I.5.4 Tracking planning

Tracking is very closely related to the control system. It is necessary to record how a mobile robot moves around. In many applications, mobile robots are required to follow a predefined trajectory. Trajectory tracking is a field of robotics that can be considered part of the motion planning problem. It can also be seen as a type of control problem which consists of planning and following a trajectory in the presence of noise and uncertainty. The trajectory tracking problem is defined as finding a control law such that the difference between the planned trajectory and the real trajectory followed by the mobile robot must be zero. It seeks to determine the control inputs needed to follow, or track, a desired trajectory that has been previously planned for the mobile robot, while at the same time avoiding disturbances due to unmodeled dynamic effects as friction and noise.

I.5.5 I.5.5 Obstacle avoidance

Collisions between the mobile robot and obstacles must be avoided during robot motion. Robot navigation is related to a mobile robot's ability to move around the environment (known or unknown) to achieve a goal without hitting any obstacles. The robot's motion from its current location to the goal involves calculating a trajectory. The process needs a map, a goal location, and the robot's current location using sensors or another location system. In addition, a good motion planner must be able to detect collisions between the robot and obstacles in the work environment so that the robot can change its trajectory or stop before a collision occurs. Obstacle avoidance algorithms help prevent collisions. They involve obstacle detection and obstacle avoidance itself.

I.6 Conclusion

In this chapter, an overview of mobile robotics has been given. We have identified mobile robots and classify them according to their working environment, and we have also introduced the fundamental characteristic of mobile robots, which is autonomous navigation, i.e., the ability to navigate , locate and map based on a set of internal and external sensors that allow the perception of this environment.

The next chapter deals with the mathematical modeling of the two-wheeled self-balancing robot.

CHAPTER II

MODELING OF A TWO-WHEELED SELF-BALANCING ROBOT

Chapter II Modelling of a Two-Wheels Self-Balancing Robot

II.1 Introduction

The two-wheeled self-balancing robot represents a robotic platform with two independently actuated wheels and center of gravity above the axis of the wheels’ rotation. The behavior of the robot is similar to the classical mechanical system of an inverted pendulum. It is an interesting system to control since it is inherently unstable and non-linear.

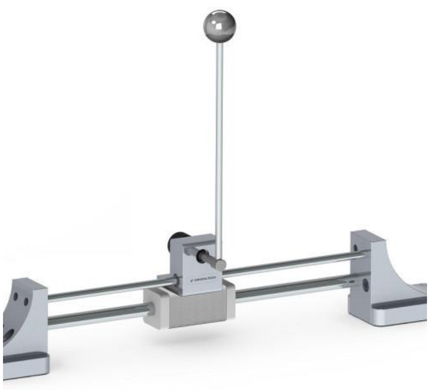

Figure II.1: *inverted pendulum*

Such robots have made a rapid advancement over the last decade and appeared in many areas of people’s daily life. The best-known examples of this robotic platform are modern vehicles such as Segway or Hoverboard. Also, there are a lot of concepts based on the above-mentioned robotic platform which can help people in their daily life, like for example assistant robots, baggage transportation robots or wheelchairs for handicapped people. A wide application range of these robots stems from their fundamental characteristics including compact structure and good maneuverability with zero turning radius.

II.2 System modeling

In this section, the mathematical model of the robot is derived into two subsystems:

II.2.1 Mechanical subsystem

A mechanical subsystem consists of robot's body and two wheels. The body can be modelled as an inverted pendulum with the mass concentrated in the center of gravity and the axis of rotation above the axis of wheels. For derivation of motion equations, a planar model is used where robot moves along a horizontal axis, as is shown in **Figure II.2** Suppose that there is no slip between the wheels and the ground. The Lagrange's second order equations are used for modelling of a mechanical subsystem [21].

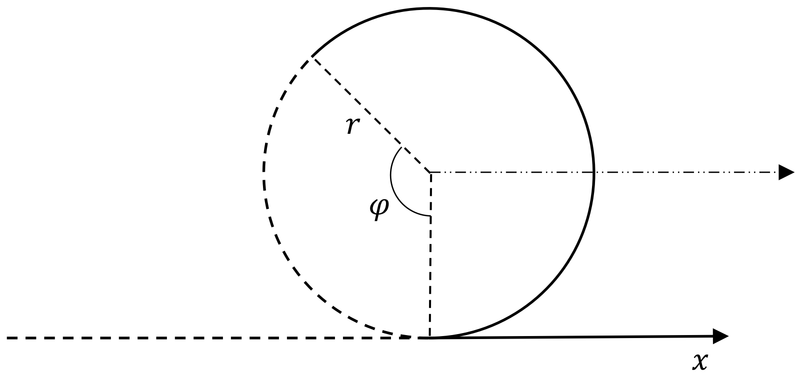

Figure II.2: *free-body diagram of the wheel*

The no slip condition of movement is given by

$$x = r \times \varphi \qquad II.1$$

Where $\boldsymbol{x}$ is the displacement, $\boldsymbol{r}$ is the radius of the wheel and $\boldsymbol{\varphi}$ is rotation angle

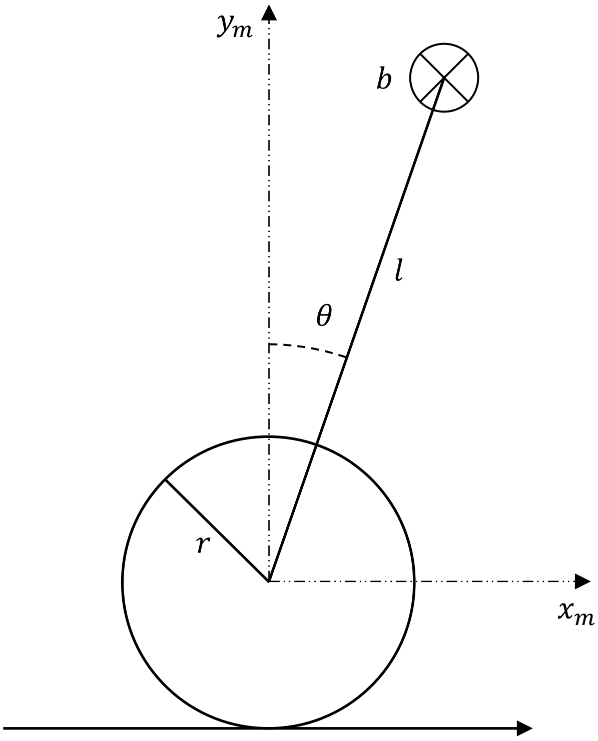

Figure II.3: *Free body diagram of robot's mechanical subsystem*

The movement of the robot's body's center of gravity is given by

Center of gravity's coordinates:

$$x_b = x + l \times sin(\theta)$$
$$z_b = l \times cos(\theta) \qquad II.2$$

Center of gravity's velocity:

$$\dot{x}_b = \dot{x} + l\,cos(\theta)\,\dot{\theta}$$
$$\dot{z}_b = -l\,sin(\theta)\,\dot{\theta} \qquad II.3$$

$$v_b{}^2 = \dot{x}_b{}^2 + \dot{z}_b{}^2$$
$$v_b{}^2 = (\dot{x} + l\,cos(\theta)\,\dot{\theta})^2 + (-\,l\,sin(\theta)\,\dot{\theta})^2$$
$$v_b{}^2 = \dot{x}^2 + 2\dot{x}l\,cos(\theta)\,\dot{\theta} + l^2\,cos^2(\theta)\,\dot{\theta}^2 + l^2\,sin^2(\theta)\,\dot{\theta}^2 \qquad II.4$$
$$v_b{}^2 = \dot{x}^2 + 2\dot{x}l\,cos(\theta)\,\dot{\theta} + l^2\dot{\theta}^2(cos^2(\theta) + sin^2(\theta))$$
$$v_b{}^2 = \dot{x}^2 + 2\dot{x}l\,cos(\theta)\,\dot{\theta} + l^2\dot{\theta}^2$$

Kinetic energy (translation kinetic energy + rotation kinetic energy) of the body is given by:

$$KE = \frac{1}{2}mv^2 + \frac{1}{2}I\dot{\theta}^2 \quad II.5$$

$$KE_b = \frac{1}{2}m_b{v_b}^2 + \frac{1}{2}I_b\dot{\theta}^2 \quad II.6$$

$$I_b = m_b l^2 \quad II.7$$

$$KE_b = \frac{1}{2}m_b(\dot{x}^2 + 2\dot{x}l\,cos(\theta)\,\dot{\theta} + l^2\dot{\theta}^2) + \frac{1}{2}I_b\dot{\theta}^2 \quad II.8$$

Where $\boldsymbol{m_b}$ is the mass of the body, $\boldsymbol{v_b}$ is the speed and $\boldsymbol{I_b}$ is the inertia.

Kinetic energy of the wheel :

$$KE_w = \frac{1}{2}m_w\dot{x}^2 + \frac{1}{2}I_w\dot{\varphi}^2 \quad II.9$$

From II.1 we have:

$$\dot{x} = r \times \dot{\varphi} \Longrightarrow \dot{\varphi} = \frac{\dot{x}}{r} \quad II.10$$

$$KE_w = \frac{1}{2}m_w\dot{x}^2 + \frac{1}{2}\frac{I_w}{r^2}\dot{x}^2 = \frac{1}{2}(m_w + \frac{I_w}{r^2})\dot{x}^2 \quad II.11$$

Where $\boldsymbol{m_w}$ is the mass of the body, $\boldsymbol{\dot{x}}$ is the linear speed, $\boldsymbol{I_w}$ is the inertia and **r** is the radius of the wheel.

Potential energy of the body :

$$PE = mgh \quad II.12$$

$$h = l\,cos(\theta) \quad II.13$$

$$PE_b = m_b gl\,cos(\theta) \quad II.14$$

Potential energy of the wheel :

$$PE_w = 0 \quad II.15$$

Rayleigh dissipation function for both wheels :

$$D = \frac{1}{2}\sum_{i=1}^{n} c_i\dot{q}^2$$
$$D = 2\left(\frac{1}{2}b_w\dot{\varphi}^2\right) = \frac{b_w}{r^2}\dot{x}^2 \quad II.16$$

$\boldsymbol{b_w}$:damping constant

The Lagrangian of the system is defined as the difference between the kinetic and potential Energy :

$$L = KE - PE = KE_b + 2KE_w - PE_b - 2PE_w \qquad II.17$$

$$L = \frac{1}{2} m_b\left(\dot{x}^2 + 2\dot{x}l\,cos(\theta)\,\dot{\theta} + l^2\dot{\theta}^2\right) + \frac{1}{2} I_b \dot{\theta}^2 + \left(m_w + \frac{I_w}{r^2}\right)\dot{x}^2 - m_b g l\,cos(\theta) \qquad II.18$$

The Lagrange second order equations with the dissipation function are defined as :

$$\frac{d}{dt}\left(\frac{\partial L}{\partial \dot{q}_i}\right) - \frac{\partial L}{\partial q_i} = Q_i - \frac{\partial D}{\partial \dot{q}_i} \qquad II.19$$

where $\boldsymbol{q_i}$ represents generalized coordinates and $\boldsymbol{Q_i}$ represents generalized forces. Generalized forces are torques from both wheels

$$Q_i = \tau_R + \tau_L \qquad II.20$$

Position $\boldsymbol{x}$ and angular position $\boldsymbol{\theta}$ were chosen as system coordinates Substituting system coordinates into generalized coordinates, we can rewrite the Lagrange second order equations and get equations of motion.

For the $\boldsymbol{x}$ coordinate we obtain

$$\frac{d}{dt}\left(\frac{\partial L}{\partial \dot{x}}\right) - \frac{\partial L}{\partial x} + \frac{\partial D}{\partial \dot{x}} = \frac{\tau_R + \tau_L}{r} \qquad II.21$$

So, the generalized force in this case is:

$$q_1 = x; \tau = rF \Rightarrow F = \frac{\tau}{r} \Rightarrow Q_1 = \frac{\tau_R + \tau_L}{r} \qquad II.22$$

$$\frac{\partial L}{\partial \dot{x}} = \frac{1}{2} m_b\left(2\dot{x} + 2l\,cos(\theta)\,\dot{\theta}\right) + 2\left(m_w + \frac{I_w}{r^2}\right)\dot{x} \qquad II.23$$

$$\frac{d}{dt}\left(\frac{\partial L}{\partial \dot{x}}\right) = \frac{1}{2} m_b\left(2\ddot{x} + 2l(-sin(\theta)\,\dot{\theta}^2 + cos(\theta)\,\ddot{\theta})\right) + 2\left(m_w + \frac{I_w}{r^2}\right)\ddot{x} \qquad II.24$$

$$\frac{\partial L}{\partial x} = 0 \qquad II.25$$

$$\frac{\partial D}{\partial \dot{x}} = 2\frac{b_w}{r^2}\dot{x} \qquad II.26$$

We substitute the equations II.24, II.25 and II.26 in II.21 we get:

$$m_b\left(\ddot{x} + l\left(-sin(\theta)\,\dot{\theta}^2 + cos(\theta)\,\ddot{\theta}\right)\right) + 2\left(m_w + \frac{I_w}{r^2}\right)\ddot{x} + 2\frac{b_w}{r^2}\dot{x} = \frac{\tau_R + \tau_L}{r} \qquad II.27$$

$$\left(m_b + 2m_w + 2\frac{I_w}{r^2}\right)\ddot{x} + \left(2\frac{b_w}{r^2}\right)\dot{x} + m_b l\left(-sin(\theta)\,\dot{\theta}^2 + cos(\theta)\,\ddot{\theta}\right) = \frac{\tau_R + \tau_L}{r} \qquad II.28$$

For the $\boldsymbol{\theta}$ coordinate we obtain

$$\frac{d}{dt}\left(\frac{\partial L}{\partial \dot{\theta}}\right) - \frac{\partial L}{\partial \theta} + \frac{\partial D}{\partial \dot{\theta}} = -(\tau_R + \tau_L) \qquad II.29$$

$$q_2 = \theta; Q_2 = -(\tau_R + \tau_L) \qquad II.30$$

$$\frac{\partial L}{\partial \dot{\theta}} = m_b\left(\dot{x}l\,cos(\theta) + l^2\dot{\theta}\right) + I_b\dot{\theta} \qquad II.31$$

$$\frac{d}{dt}\left(\frac{\partial L}{\partial \dot{\theta}}\right) = m_b\ddot{x}l\,cos(\theta) - m_b\dot{x}l\,sin(\theta)\,\dot{\theta} + m_b l^2\ddot{\theta} + I_b\,\ddot{\theta} \qquad II.32$$

$$\frac{\partial L}{\partial \theta} = -m_b\dot{x}l\,sin(\theta)\,\dot{\theta} + m_b gl\,sin(\theta) \qquad II.33$$

$$\frac{\partial D}{\partial \dot{\theta}} = 0 \qquad II.34$$

By substituting the equations II.32, II.33, II.34 in II.29 we get:

$$m_b\ddot{x}l\,cos(\theta) + m_b l^2\ddot{\theta} + I_b\ddot{\theta} - m_b gl\,sin(\theta) = -(\tau_R + \tau_L)$$
$$(m_b l^2 + I_b)\ddot{\theta} + m_b l\,cos(\theta)\,\ddot{x} - m_b gl\,sin(\theta) = -(\tau_R + \tau_L) \qquad II.35$$

The derived dynamic equations of motion are nonlinear. Assume that while the robot moves along the $\boldsymbol{x}$ axis only small deviations in the angular position $\boldsymbol{\theta}$ are obtained. This means that we can make a linearization around an unstable equilibrium point which makes the model more suitable for controller design. Using this fact, the approximations are :

$$\begin{cases} cos(\theta) \approx 1 \\ sin(\theta) \approx \theta \\ \dot{\theta}^2 \approx 0 \end{cases} \qquad II.36$$

and linearized equations of motion are :

$$\begin{cases} \left(m_b + 2m_w + 2\frac{I_w}{r^2}\right)\ddot{x} + \left(2\frac{b_w}{r^2}\right)\dot{x} + m_b l\ddot{\theta} = \frac{\tau_R+\tau_L}{r} \\ (m_b l^2 + I_b)\ddot{\theta} + m_b l\ddot{x} - m_b gl\theta = -(\tau_R + \tau_L) \end{cases} \qquad II.37$$

II.2.2 Actuator subsystem

The brushed DC gearmotors are used as the actuator subsystem of the robot. The subsystem directly provides rotary motion and coupled with the wheels allows the robot to make movement. The input to this subsystem is voltage source $\boldsymbol{v_s}$ applied to the motor's armature, while the output is the rotational speed $\dot{\boldsymbol{\varphi}}_{\boldsymbol{W}}$ and torque $\boldsymbol{\tau_W}$ of the gearbox shaft.

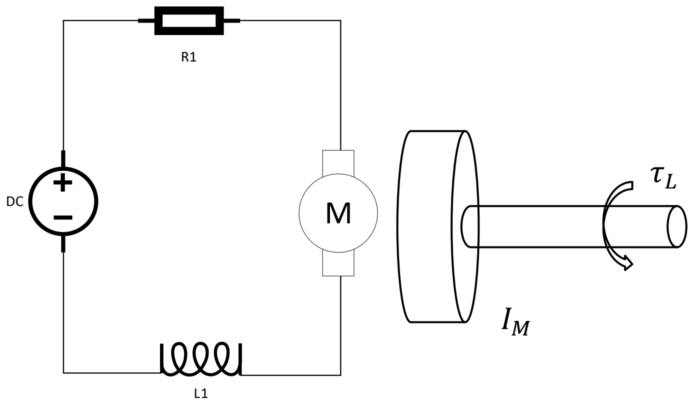

Figure II.4: *Electric equivalent circuit of the armature and the free-body diagram of the rotor*

The torque $\boldsymbol{\tau}$ generated by a DC motor is in general proportional to the armature current $\boldsymbol{i}$ and strength of the magnetic field. Assume that the magnetic field is constant. In that case the motor torque is proportional to the armature current by constant factor $\boldsymbol{k_t}$

$$\tau = k_t i \qquad II.38$$

While the shaft is rotating the back emf voltage $\boldsymbol{v_{emf}}$ is generated and is proportional to the angular velocity of the shaft by constant factor $\boldsymbol{k_e}$

$$v_{emf} = k_e \dot{\varphi}_M \qquad II.39$$

The motor torque and back emf constant have the same numerical value , therefore we will use $\boldsymbol{K}$ to represent them both.

From the electrical equivalent circuit of the armature, we can derive the equation

$$v_s - Ri - L\frac{di}{dt} - v_{emf} = 0 \qquad II.40$$

Given that the value of inductance $\boldsymbol{L}$ of a small DC motor is in general a very low number compared to its resistance $\boldsymbol{R}$, we can neglect it.

From the free-body diagram of the rotor we can derive the equation

$$I_M\ddot{\varphi} + b_M\dot{\varphi}_M + \tau_L = \tau \qquad II.41$$

By putting the Equations II.38 and II.41 together we obtain the equation describing the overall DC motor characteristics:

$$I_M\ddot{\varphi}_M + \left(b_M + \frac{K^2}{R}\right)\dot{\varphi}_M + \tau_L = \frac{K}{R}v_s \qquad II.42$$

The torque $\boldsymbol{\tau_W}$ required by the robot's wheel is much higher as compared to the nominal torque $\boldsymbol{\tau}$ of the DC motor. In this case a gearbox needs to be applied between the motor and the wheel.

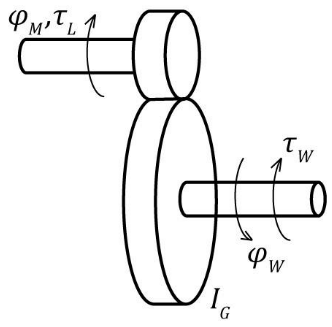

Figure II.5: *Free-body diagram of the gearbox*

Power exerted by the DC motor is the same at the input and at the output of the gear. It can be written as

$$\tau_L\dot{\varphi}_M = \tau_W\dot{\varphi}_W \qquad II.43$$

and then by using the gear aspect ratio $\boldsymbol{n}$ we obtain

$$\dot{\varphi}_M = \dot{\varphi}_W n \qquad II.44$$

$$\tau_L = \frac{\tau_W}{n} \qquad II.45$$

Substituting the above equations into Equation II.42 and denoting by $\boldsymbol{I_G}$ the internal inertia of the gear and by $\boldsymbol{b_G}$ the internal damping of the gear, we obtain the equation of the actuator subsystem

$$(I_M n^2 + I_G)\ddot{\varphi}_W + \left(\left(b_M + \frac{K^2}{R}\right)n^2 + b_G\right)\dot{\varphi}_W + \tau_W = \frac{Kn}{R}v_s \qquad II.46$$

II.2.3 The overall model of the robot

DC gearmotors are located between the wheels and the robot's body. As shown in **Figure II.2**, the relation between angular coordinates is:

$$\varphi - \theta = \varphi_W \qquad II.47$$

Consider that both gearmotors generate similar torque, then we can write:

$$\tau_R + \tau_L = 2\tau_W \qquad II.48$$

By substituting Equations II.1, II.46, II.47 and II.48 into the equation II.37 we obtain:

$$\begin{cases} \left(\frac{m_B r^2}{2} + m_W r^2 + I_W + I_M n^2 + I_G\right)\ddot{\varphi} + \left(\frac{m_B r l}{2} - I_M n^2 - I_G\right)\ddot{\theta} + \left(b_W + b_M n^2 + \frac{K^2 n^2}{R} + b_G\right)\dot{\varphi} \\ \qquad - \left(b_M n^2 + \frac{K^2 n^2}{R} + b_G\right)\dot{\theta} = \frac{Kn}{R} v_s \\ \left(\frac{m_B r l}{2} - I_M n^2 - I_G\right)\ddot{\varphi} + \left(\frac{m_B l^2}{2} + \frac{I_B}{2} + I_M n^2 + I_G\right)\ddot{\theta} - \left(b_M n^2 + \frac{K^2 n^2}{R} - b_G\right)\dot{\varphi} \\ \qquad + \left(b_M n^2 + \frac{K^2 n^2}{R} + b_G\right)\dot{\theta} - \frac{m_B g l}{2}\theta = -\frac{Kn}{R} v_s \end{cases}$$

II.49

II.3 Simulation

After deriving the TWSBR equations of motion in the previous section, the model is tested in order to obtain its response and to manage to control it.

Here we'll simulate the robot dynamics and kinematics based on Lagrange equation in Matlab SIMULINK, but first we should define the robot parameters

II.3.1 Robot parameters

The following table contains parameters of the modelled self-balancing robot.

Table 1: *Robot parameters*

m_B	Body weight	$1.2\ kg$
m_W	Wheel weight	$2 \times 10^{-2}\ kg$
I_B	Body inertia	$1.5 \times 10^{-2}\ kgm^2$
I_W	Wheel inertia	$2 \times 10^{-5}\ kgm^2$
I_M	Rotor inertia	$1 \times 10^{-6}\ kgm^2$
I_G	Gearbox inertia	$1 \times 10^{-4}\ kgm^2$

b_M	Motor damping	$1 \times 10^{-3}\ Nms/rad$
b_W	Wheel damping	$1 \times 10^{-3}\ Nms/rad$
b_G	Gearbox damping	$1 \times 10^{-3}\ Nms/rad$
n	Gearbox ratio	30
r	Wheel diameter	$3.2 \times 10^{-2}\ m$
l	COG distance	$7.5 \times 10^{-2}\ m$
R	Armature resistance	$2.4\ \Omega$
K	Motor constant	$1 \times 10^{-2}\ Nm/A$

Both robot models (kinematics & dynamics) were built on the Simulink MATLAB environment.

II.3.2 Kinematics

$$x = r \int \omega dt = r \int \dot{\varphi} dt \qquad II.50$$

The input to the kinematics of the robot is the speed of the wheels $\dot{\boldsymbol{\varphi}}$ and the output is its location $\boldsymbol{x}$.

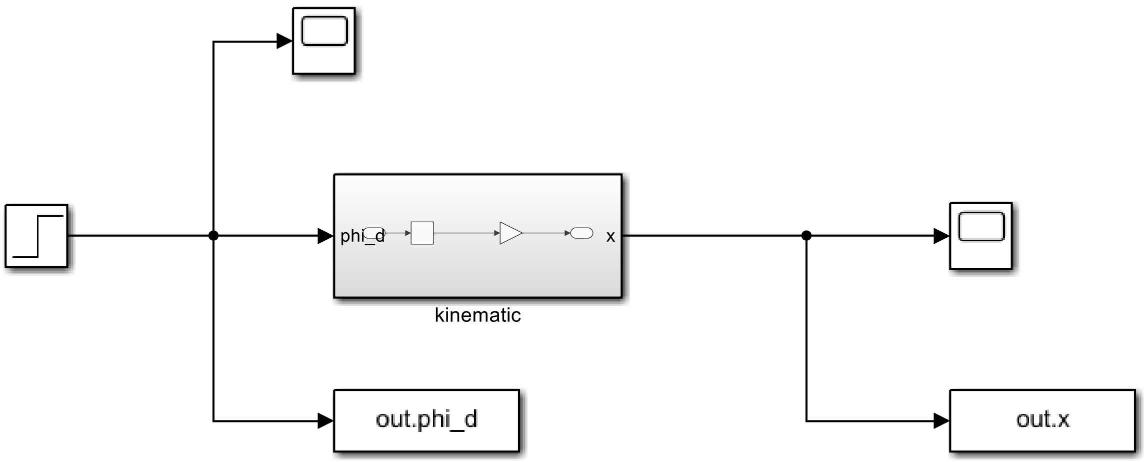

Figure II.6: *the kinematics of the robot on SIMULINK*

So, we tested the kinematics of the robot and we got the following results:

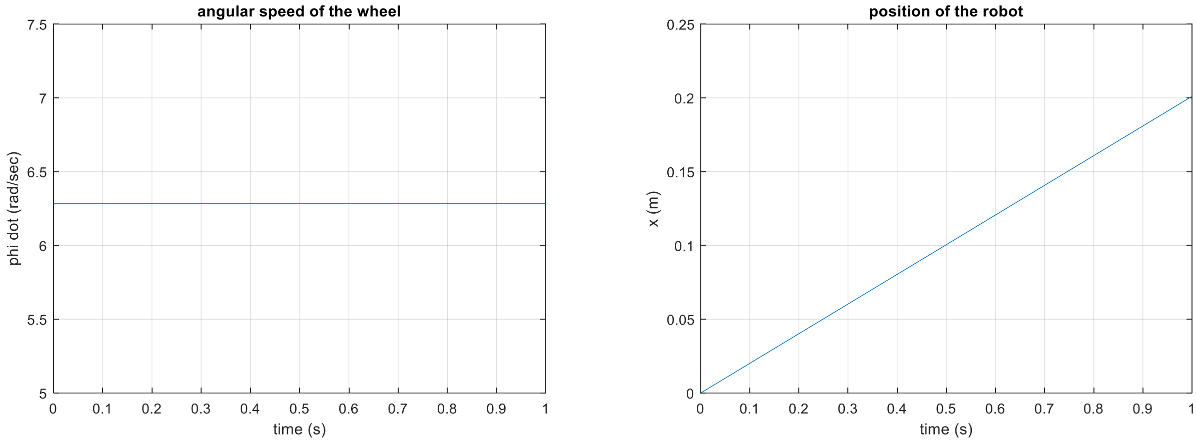

Figure II.7: *input (angular speed) and output (position) of the robot kinematics*

The position of the robot is increasing linearly because the speed of the wheels is constant.

II.3.3 Dynamics

We can rewrite the equation II.49 into the following form:

$$\begin{cases} E1\ddot{\varphi} + E2\ddot{\theta} + F1\dot{\varphi} - F2\dot{\theta} = \frac{Kn}{R}v_s \\ E3\ddot{\varphi} + E4\ddot{\theta} - F3\dot{\varphi} + F4\dot{\theta} - \frac{m_B g l}{2}\theta = -\frac{Kn}{R}v_s \end{cases} \qquad II.51$$

$$\begin{cases} \ddot{\varphi} = \frac{1}{E1}\left(-E2\ddot{\theta} - F1\dot{\varphi} + F2\dot{\theta} + \frac{Kn}{R}v_s\right) \\ \ddot{\theta} = \frac{1}{E4}\left(-E3\ddot{\varphi} + F3\dot{\varphi} - F4\dot{\theta} + \frac{m_B g l}{2}\theta - \frac{Kn}{R}v_s\right) \end{cases} \qquad II.52$$

Where:

$$\begin{aligned} E1 &= \frac{m_B r^2}{2} + m_W r^2 + I_W + I_M n^2 + I_G \\ E2 &= \frac{m_B r l}{2} - I_M n^2 - I_G \\ E3 &= \frac{m_B r l}{2} - I_M n^2 - I_G \\ E4 &= \frac{m_B l^2}{2} + \frac{I_B}{2} + I_M n^2 + I_G \\ F1 &= b_W + b_M n^2 + \frac{K^2 n^2}{R} + b_G \\ F2 &= b_M n^2 + \frac{K^2 n^2}{R} + b_G \\ F3 &= b_M n^2 + \frac{K^2 n^2}{R} - b_G \\ F4 &= b_M n^2 + \frac{K^2 n^2}{R} + b_G \end{aligned} \qquad II.53$$

The input to the dynamic model of the system is the voltage $\boldsymbol{v_s}$, and the output is the angular speed of the wheel $\dot{\boldsymbol{\varphi}}$ and the angle of the body of the robot $\dot{\boldsymbol{\theta}}$.

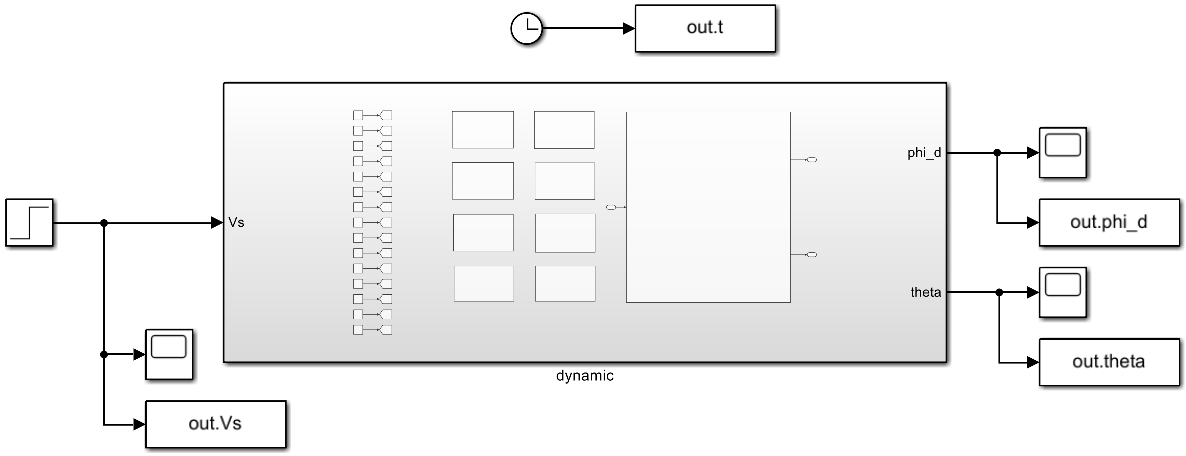

Figure II.8: *robot dynamics test*

We tested the robot dynamics and the results were as expected, the robot is not stable without a controller.

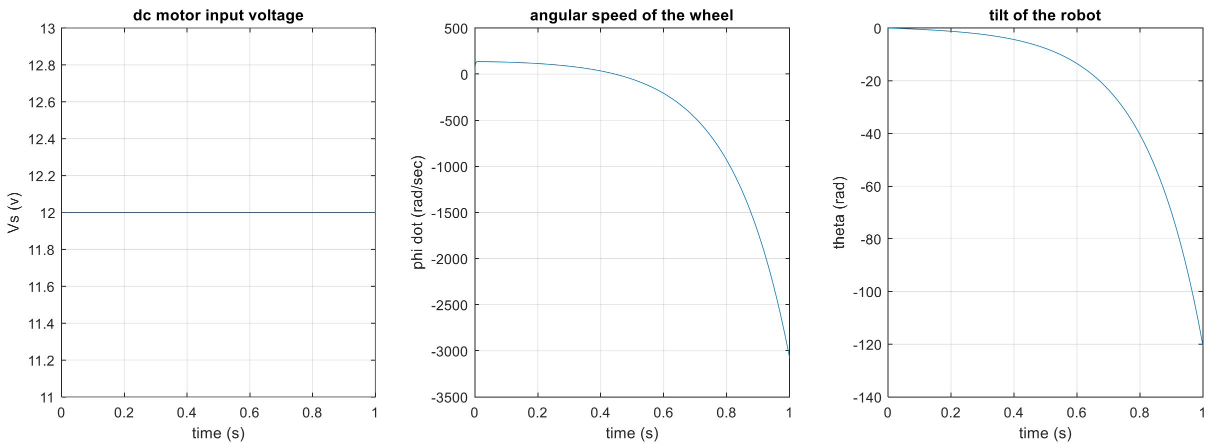

Figure II.9: *input ($\boldsymbol{v_s}$) and output (angular velocity $\dot{\boldsymbol{\varphi}}$ and tilt angle $\boldsymbol{\theta}$) of the robot dynamics*

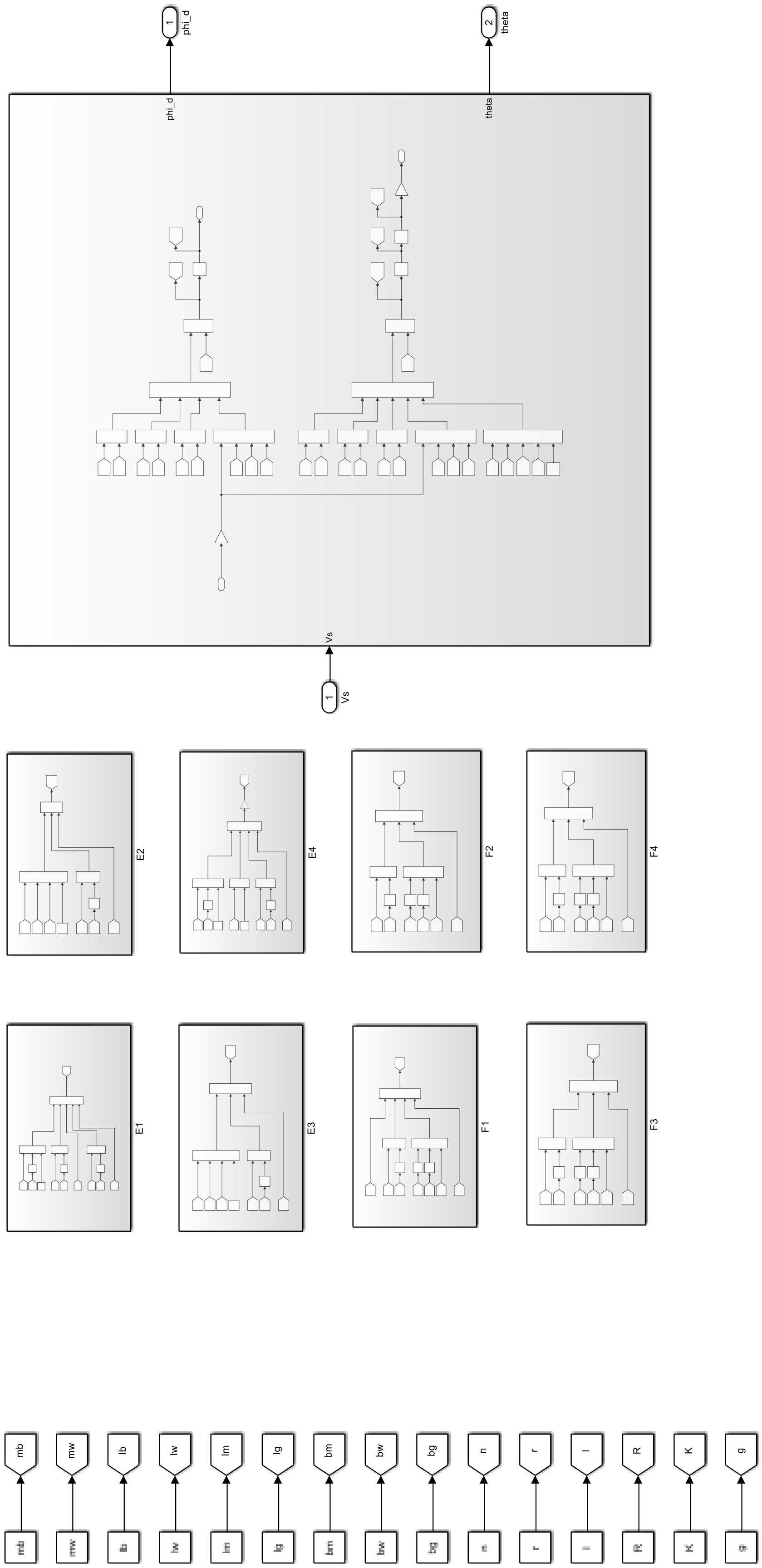

Figure II.10: *the dynamics of the robot on SIMULINK*

II.3.4 State-space module

The above equation II.49 can be rewritten into a matrix form as

$$E\begin{bmatrix}\ddot{\varphi}\\ \ddot{\theta}\end{bmatrix}+F\begin{bmatrix}\dot{\varphi}\\ \dot{\theta}\end{bmatrix}+G\begin{bmatrix}\varphi\\ \theta\end{bmatrix}=Hv_s \qquad II.54$$

$$E\begin{bmatrix}\ddot{\varphi}\\ \ddot{\theta}\end{bmatrix}=-F\begin{bmatrix}\dot{\varphi}\\ \dot{\theta}\end{bmatrix}-G\begin{bmatrix}\varphi\\ \theta\end{bmatrix}+Hv_s \qquad II.55$$

$$\begin{bmatrix}\ddot{\varphi}\\ \ddot{\theta}\end{bmatrix}=E^{-1}\left(-F\begin{bmatrix}\dot{\varphi}\\ \dot{\theta}\end{bmatrix}-G\begin{bmatrix}\varphi\\ \theta\end{bmatrix}+Hv_s\right) \qquad II.56$$

$$\begin{bmatrix}\ddot{\varphi}\\ \ddot{\theta}\end{bmatrix}=-E^{-1}G\begin{bmatrix}\varphi\\ \theta\end{bmatrix}-E^{-1}F\begin{bmatrix}\dot{\varphi}\\ \dot{\theta}\end{bmatrix}+E^{-1}Hv_s \qquad II.57$$

And finally, the state-space model of self-balancing two-wheeled robot is given as:

$$\begin{cases}\begin{bmatrix}\dot{\varphi}\\ \dot{\theta}\\ \ddot{\varphi}\\ \ddot{\theta}\end{bmatrix}=\begin{bmatrix}0 & 0 & 1 & 0\\ 0 & 0 & 0 & 1\\ \hline \multicolumn{2}{c}{-E^{-1}G} & \multicolumn{2}{c}{-E^{-1}F}\end{bmatrix}\begin{bmatrix}\varphi\\ \theta\\ \dot{\varphi}\\ \dot{\theta}\end{bmatrix}+\begin{bmatrix}0\\ 0\\ \hline E^{-1}H\end{bmatrix}v_s\\ y=\begin{bmatrix}r & 0 & 0 & 0\\ 0 & 1 & 0 & 0\end{bmatrix}\begin{bmatrix}\varphi\\ \theta\\ \dot{\varphi}\\ \dot{\theta}\end{bmatrix}\end{cases} \qquad II.58$$

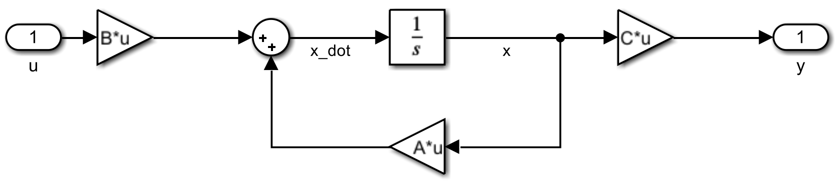

Figure II.11: *the state-space model of the robot on SIMULINK*

II.4 Conclusion

Modeling is very important in the study of the movement of robots, especially in the field of research and development of robots.

In this chapter, we focused on the compute of the kinematic model and the dynamic model of a two-wheeled self-balancing robot, and we tested the validity of the modeling by simulation in Simulink Matlab. Kinematic modeling is the study of the movement of mechanical systems without taking into account the forces that affect the movement, and dynamic modeling is the study of the movement of the mechanical system, taking into account the forces affecting the movement.

In the next chapter, we will propose a control system based on PID and fuzzy control.

CHAPTER III

CONTROL SYSTEM

Chapter III Control system

III.1 Introduction

After deriving the TWSBR equations of motion in the previous section, the model is tested in order to obtain its response and to manage to control it. Different control strategies are implemented and compared for the purpose of obtaining a suitable response for the system.

III.2 Open loop system

An open-loop system response has to be investigated in order to study the behavior of the developed model. Employing the simulation parameters listed in **Table 1:** Robot parameters, the model is simulated in Matlab Simulink® environment and the simulation results are illustrated in **Figure III.2**.

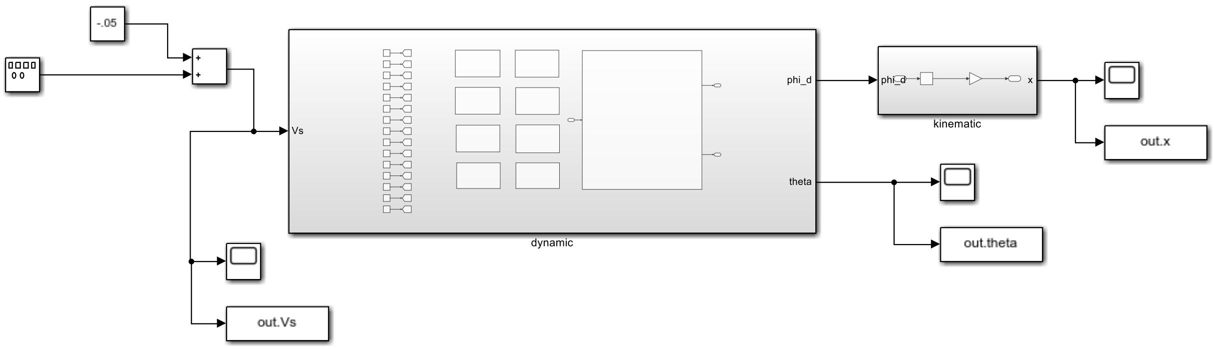

Figure III.1: *Open loop command scheme*

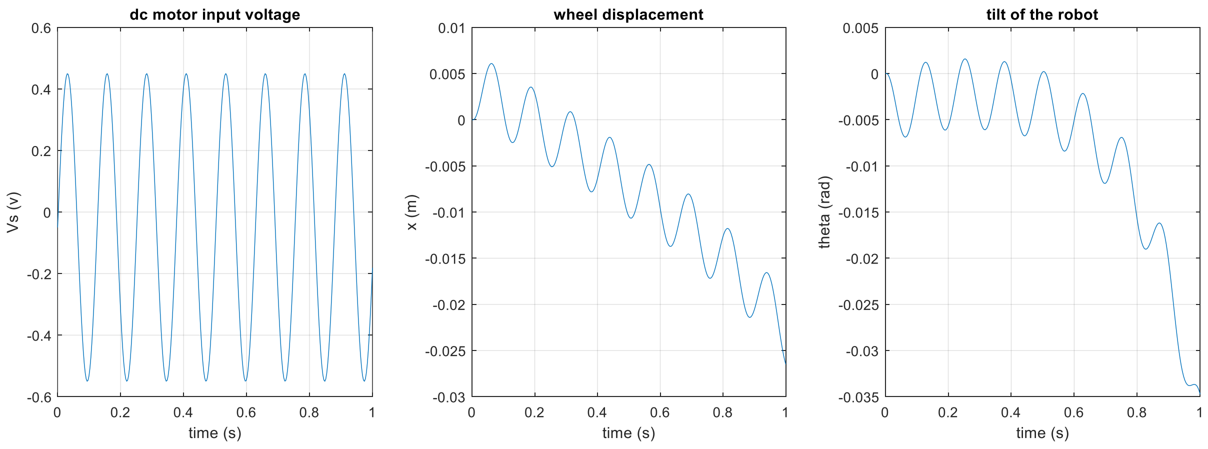

Figure III.2: *open-loop response*

It is clear from the response of pitch angel $\boldsymbol{\theta}$, wheel displacement $\boldsymbol{x}$ that the system is an unstable nonlinear system. The obtained response in **Figure III.2** is actually expected. Although the angle of IP moves the same as the chassis movement, they are different in terms of magnitude. When the tilt angle starts oscillating to the left (positive tilt angle), it affects the wheels and forces them to maneuver in the direction of the frame of reference. The same occurs on the other side as well but with negative sign (the wheels maneuver in the opposite side of the frame of reference when the chassis tilts with a negative angle). These oscillations are very small at the beginning and cannot be visualized if an actual system was behaving like that. With time, the amplitude of these oscillations increases until the system loses its balance completely. Based on the previous analysis and the fact that the system outputs reach infinity, a closed loop system is essential for stabilizing the system and improving its performance.

III.3 Closed-loop system

A closed-loop control system is a type of control system in which the controlling action shows dependency on the generated output of the system. In simple words, in these systems, the output of the system controls the input applied to the system.

The variation in input according to the output leads to produce more accurate system output. Thus, controllability in the closed-loop system is achieved through the output generated by utilizing a feedback path. [22]

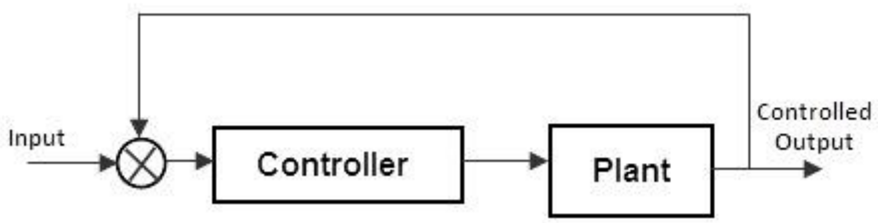

Figure III.3: *close-loop system module*

In our case the Input of the system is the DC motor voltage ($\boldsymbol{v_s}$) and the Output is the tilt angle of robot ($\boldsymbol{\theta}$), so if the robot leans forward, the control system must feed more positive voltage to the DC motor to go forward and keep the robot standing vertically, same thing goes for the other case, if the robot leans backwards the wheels must rotate backwards to keep the balance.

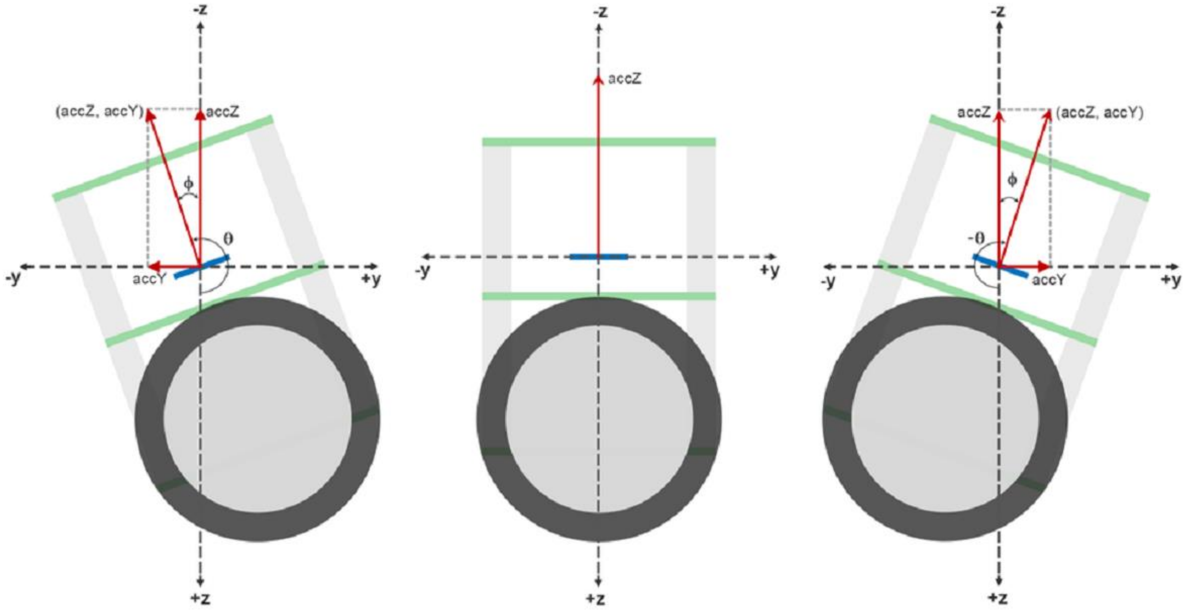

Figure III.4: *three possible cases*

Several control algorithms were considered and tested for stabilizing the robot. The first one considered was the PID controller which controls the robot using only one feedback variable $\boldsymbol{\theta}$ the tilt angle. After we're done with it, we'll do the fuzzy logic controller afterwards.

III.3.1 PID controller

PID stands for Proportional, Integral, and Derivative. Each of these terms provides a unique response to our self-balancing robot.

The proportional term, as its name suggests, generates a response that is proportional to the error. For our system, the error is the angle of inclination of the robot.

The integral term generates a response based on the accumulated error. This is essentially the sum of all the errors multiplied by the sampling period. This is a response based on the behavior of the system in past.

The derivative term is proportional to the derivative of the error. This is the difference between the current error and the previous error divided by the sampling period. This acts as a predictive term that responds to how the robot might behave in the next sampling loop.

Multiplying each of these terms by their corresponding constants (i.e. $\boldsymbol{K_p}$, $\boldsymbol{K_i}$ and $\boldsymbol{K_d}$) and summing the result, we generate the output which is then sent as command to drive the motor.

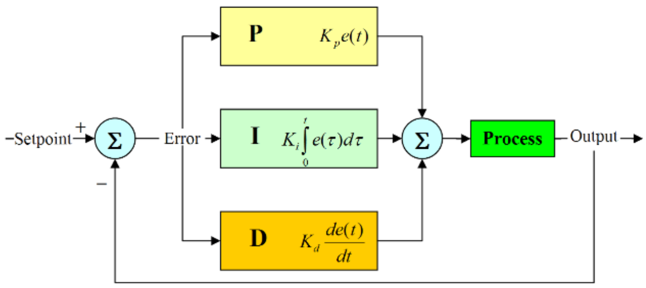

Figure III.5: *PID controller scheme*

Simulation

So, as you can see, we used two PI controllers one to control the robot tilt angle, and the second one to control the robot position.

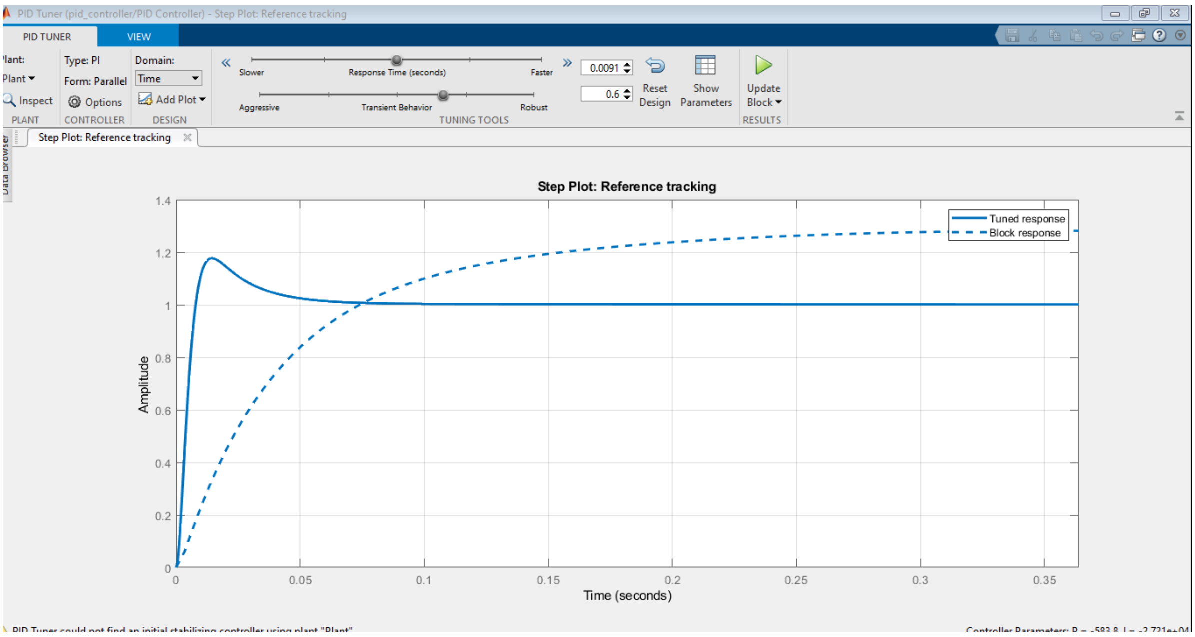

Figure III.6: *Matlab integrated PID tuner*

We used the Matlab integrated tuning method **Figure III.6** to set the K_p and K_i coefficients which they were (-75) and (-150) for the angle controller respectively, and (-1.88) and (-5.5) for the position controller.

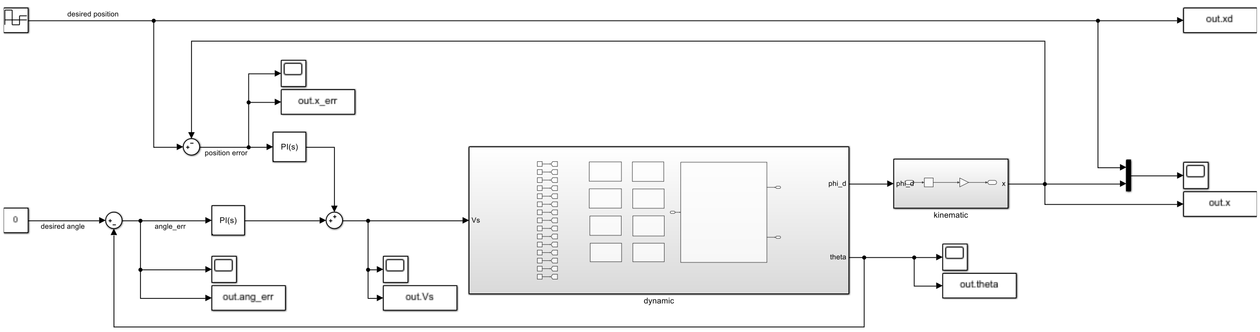

Figure III.7: *SBR with PID feedback control*

We set the timer for 30 seconds, and the robot was following the desired position while keeping its balance as we can see in the following **Figure III.8**.

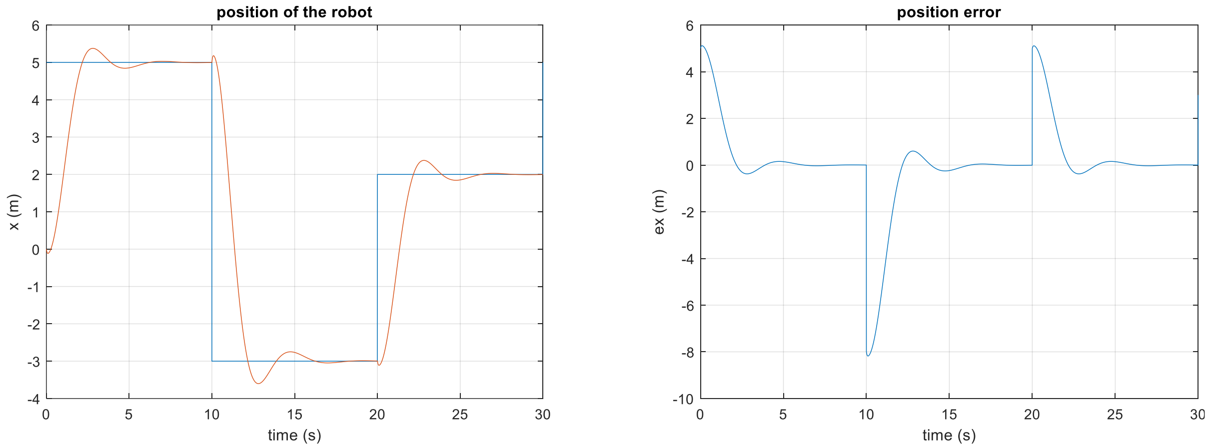

Figure III.8: *the robot position together with the desired position, and the position error*

The robot trying to keep the tilt angle to zero (vertical) and at the same time following its desired position, as we ca see in the following **Figure III.9**.

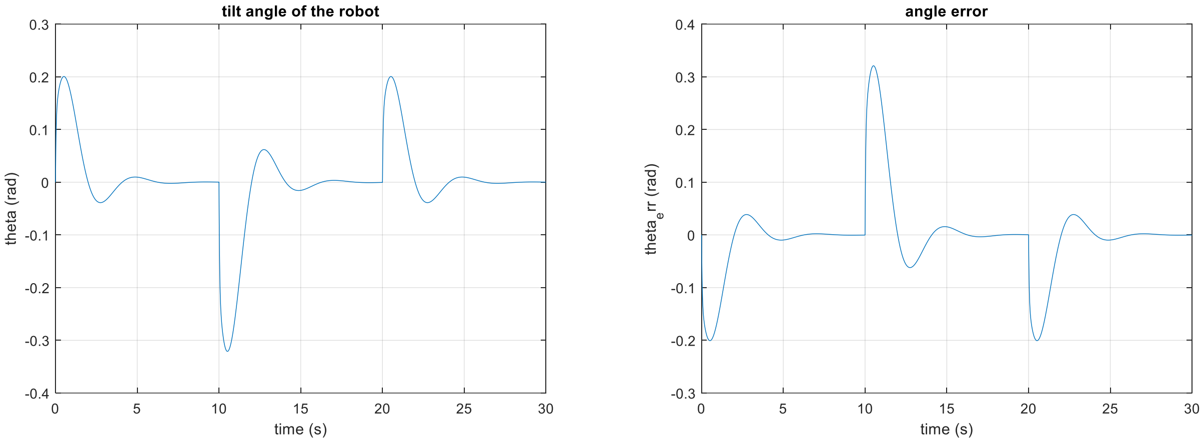

Figure III.9: *the tilt angle of the robot and its error*

we can notice that whenever there is an error in the robot position, the controller gives order to the wheels to correct that error, but in order to move without falling to the other side the robot must also be tilted a little bit. We can also notice a little bit of overshooting; we could decrease it but it would take more time to get to the desired destination.

That overshooting (decreasing oscillation) mean that when the robot arrives at its desired position it passes it a little and then comeback to it.

III.3.2 Fuzzy logic controller

The Fuzzy method gives a human like intuition to the control strategy and is self-tolerant to inputs which are no so precise. The Fuzzy Logic Controller contains different components like Fuzzification, Defuzzification and Fuzzy Rule inference. The Objective is to understand the Fuzzy Rule base and inference methods and employ them in controlling the speed of the motor. It is very efficient where the precision required is not too high. It is a robust, easily controllable strategy. It is capable of realizing multiple inputs and producing different numerous outputs. [23]

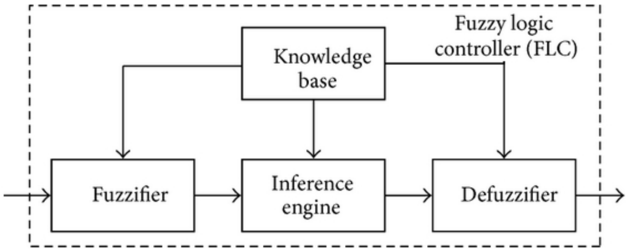

Figure III.10: *fuzzy controller structure*

Simulation

Here is the schematics of the robot being controlled by a fuzzy logic controller in its input we have the position error and the angle error and the output gives us how much voltage we should feed the DC motors.

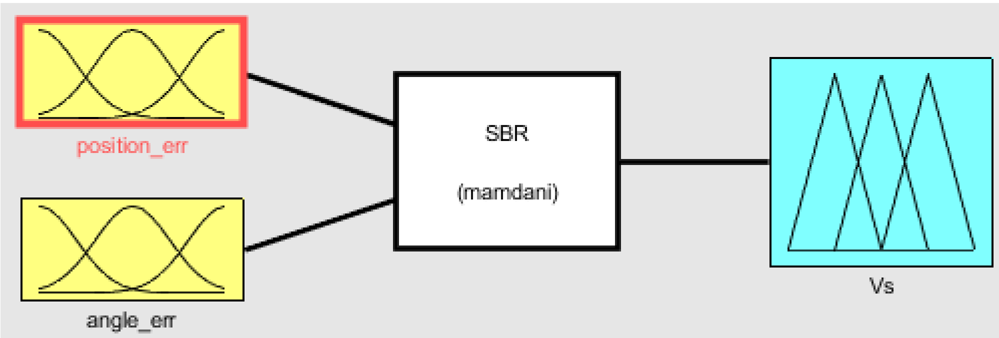

Figure III.11: *inputs and outputs*

Fuzzification

So, after figuring out the state variables, they are to be passed through the fuzzification block to fuzzify the inputs. As the Fuzzy Rule base uses rules on only linguistic variables, the numerical inputs have to be converted to fuzzy linguistic variables first.

Defuzzification

The inverse of Fuzzification is called Defuzzification. The Fuzzy Logic Controller (FLC) produces output in a linguistic variable, and the linguistic variables must be changed to crisp output. Centre of gravity strategy is the best Defuzzification system and we have used it in our controller. A fuzzy control system has certain rules that change various variables into a "fuzzy" form, that is, the outcome is shown as membership functions and their degree of membership in fuzzy sets. The most average fuzzy set enrolment capacity has the shape of a triangle. Finally, the centroid of this is computed. The abscissa of the centroid gives the defuzzified output.

Membership functions

The membership function is the graphical representation of the degree of belonging of an element to the fuzzy set. We can use different membership functions for an input and output depending upon the requirement of the precision to be provided. Generally used membership functions are triangular and trapezoidal membership functions.

In our case three membership functions for the "position error" and five membership functions for the "angle error" have been considered and the output has been given five membership functions.

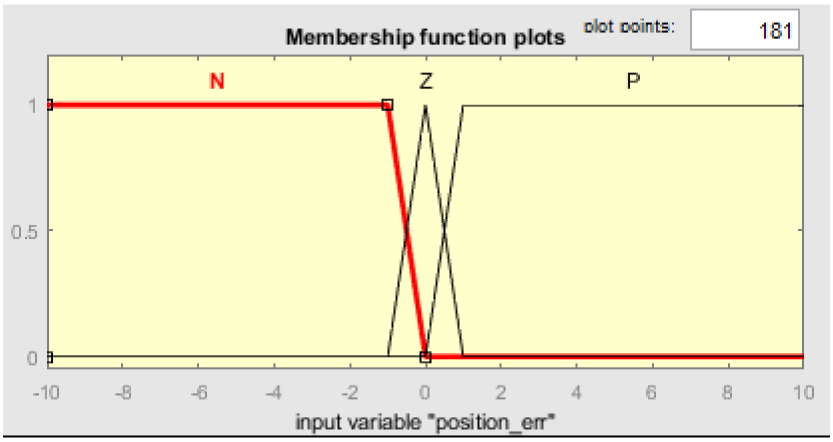

Figure III.12: *membership function of the input*

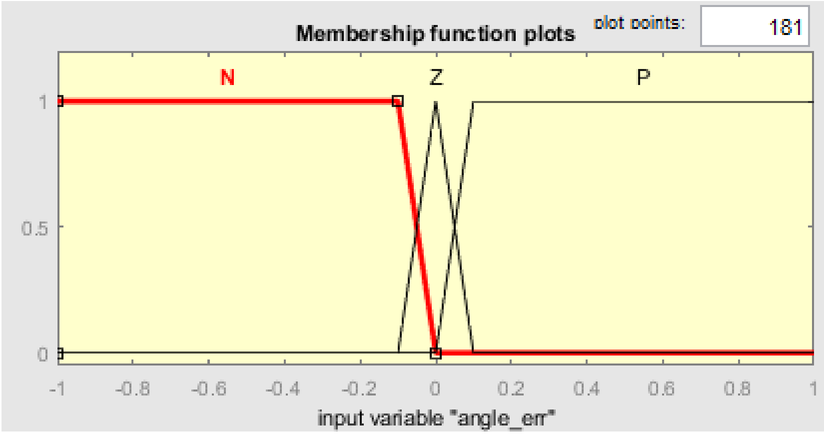

Figure III.13: *membership function of input (angle error)*

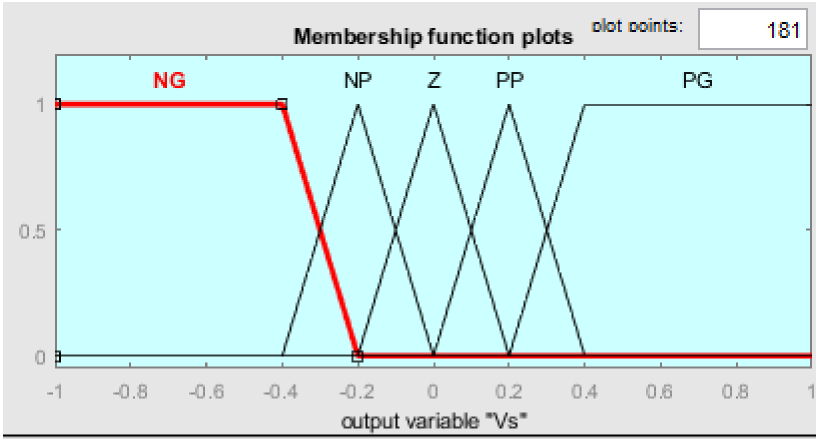

Figure III.14: *membership function of the output* (v_s)

Rules

General Interpretation of the control rules to be set to the Fuzzy control:

Rule #1: If position Error in negative (N) and angle error is negative (N), then Vs is (NG) should get -12 v.

Here is an example with **Rule #9**:

If position error is positive (P) and the angle error is positive (P), then Vs is (PG) should get +12 v.

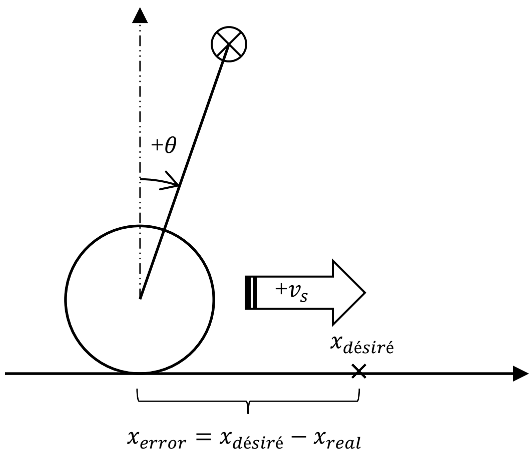

Figure III.15: *rule 9 demonstrated*

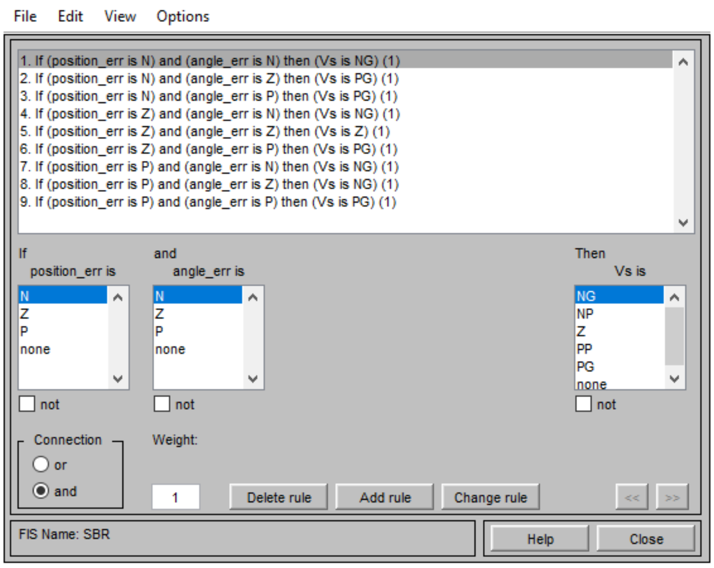

Figure III.16: *rule editor in Simulink*

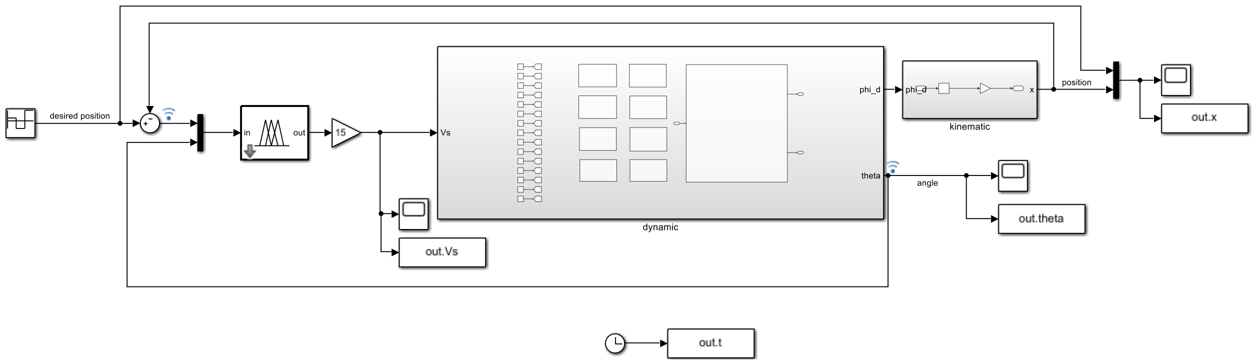

Figure III.17: *SIMULINK schematic with fuzzy controller*

After picking the membership functions and the rules for our FLC we tested it to see if follows the desired position and these were the results **Figure III.18**

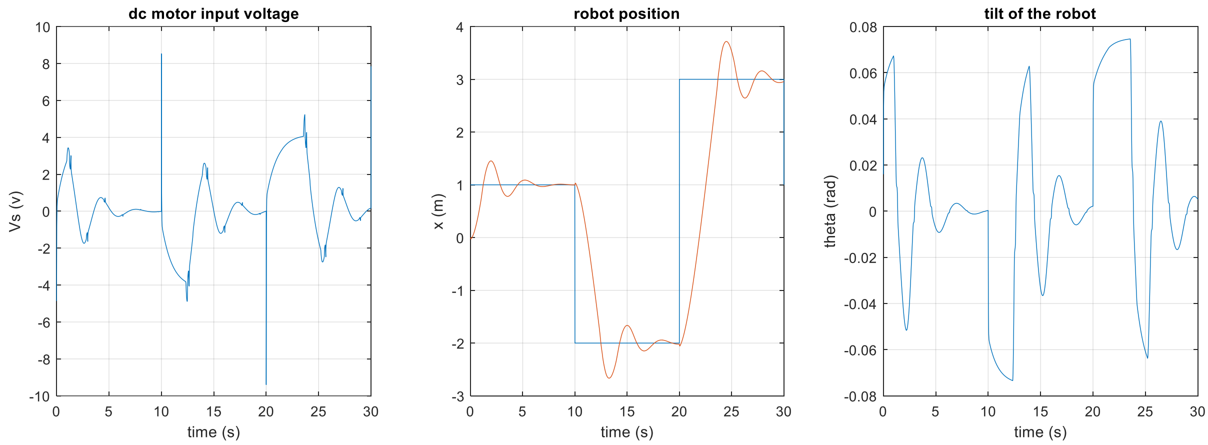

Figure III.18:.*fuzzy controller simulation result*

III.4 Comparison and conclusion

To compare our chosen controllers, we should put them in the same experiment.

In our experiment we gave the robot to be in certain positions in certain times and this was its response :

The PID command was smoother and in a smaller range compared to FLC.

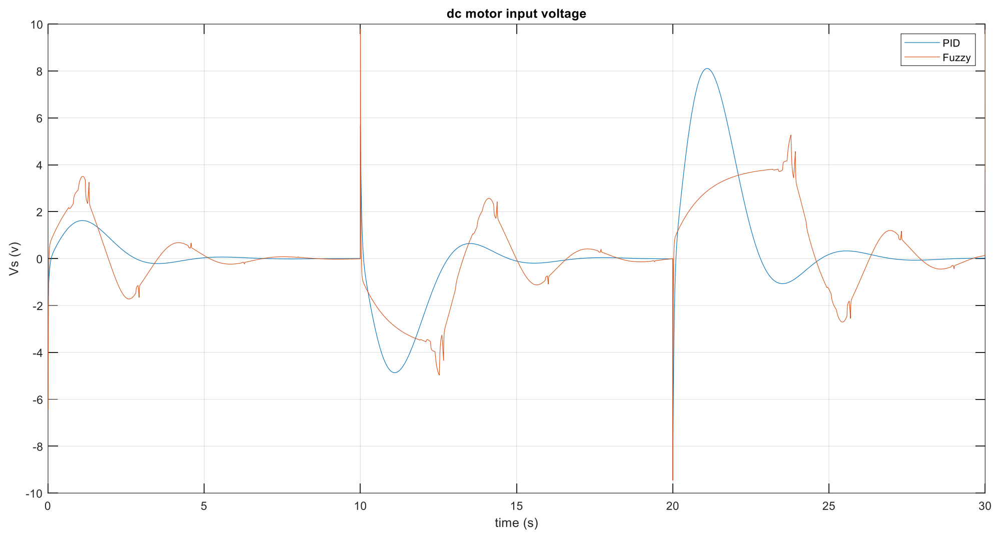

Figure III.19: *DC motor command* (v_s)

The PID controller was faster in following the robot position in long distances, and also more stable.

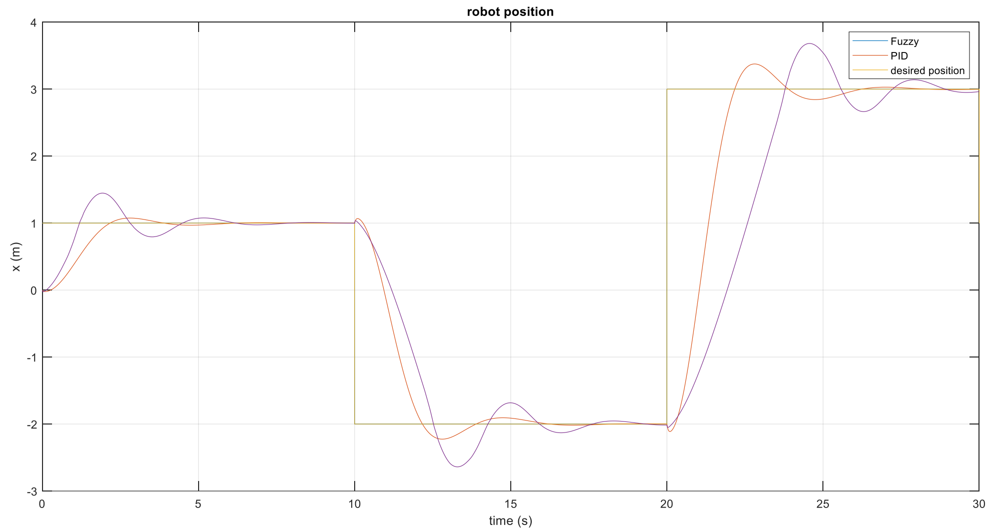

Figure III.20: *robot position* (x)

For the tilt angle its was smaller in FLC.

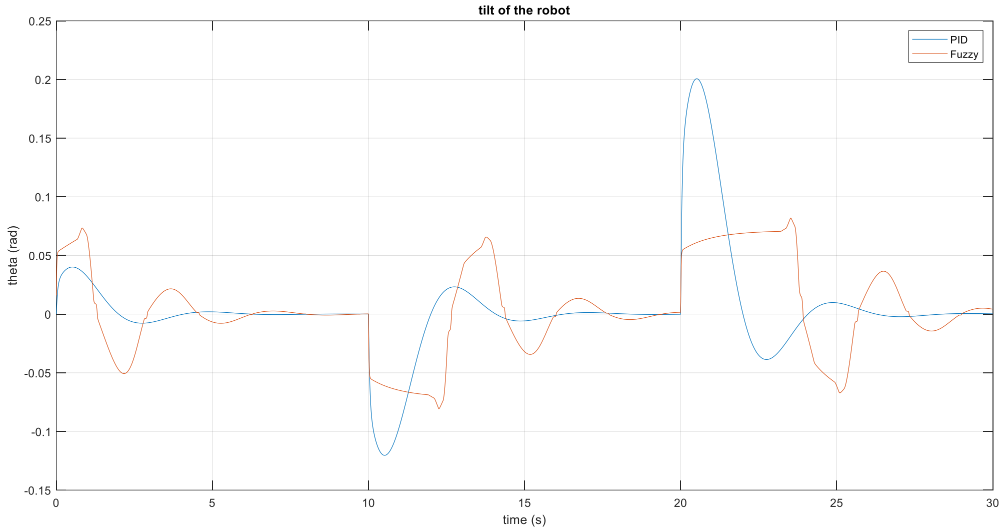

Figure III.21: *robot tilt angle ($\boldsymbol{\theta}$)*

III.5 Conclusion

In this chapter, we have proposed two types of control to stabilize the two-wheeled self-balancing robot. The simplest choice is to use a PID controller. the other choice is the fuzzy controller.

The results obtained show the efficiency of the controllers, which allowed the robot to stabilize in the vertical position.

We conclude that in our case, the PID controller was better and more efficient than the fuzzy logic controller.

CHAPTER IV

IMPLEMENTATION OF THE PROTOTYPE

Chapter IV Implementation of the prototype

The self-balancing robot is similar to an inverted pendulum. Unlike a normal pendulum which keeps on swinging once given a nudge, this inverted pendulum cannot stay balanced on its own. It will simply fall over. Then how do we balance it? Consider balancing a broomstick on our index finger which is a classic example of balancing an inverted pendulum. We move our finger in the direction in which the stick is falling. Similar is the case with a self-balancing robot, only that the robot will fall either forward or backward. Just like how we balance a stick on our finger, we balance the robot by driving its wheels in the direction in which it is falling. What we are trying to do here is to keep the center of gravity of the robot exactly above the pivot point.

IV.1 The components

Chassis

The robot is built on two layer of plastic sheets and a metal layer that are spaced 11 cm apart using metal threaded rods and bolts. The bottom layer contains the two DC 12v motors and the motor driver and the Arduino uno board. The middle layer has the battery holder and the MPU.

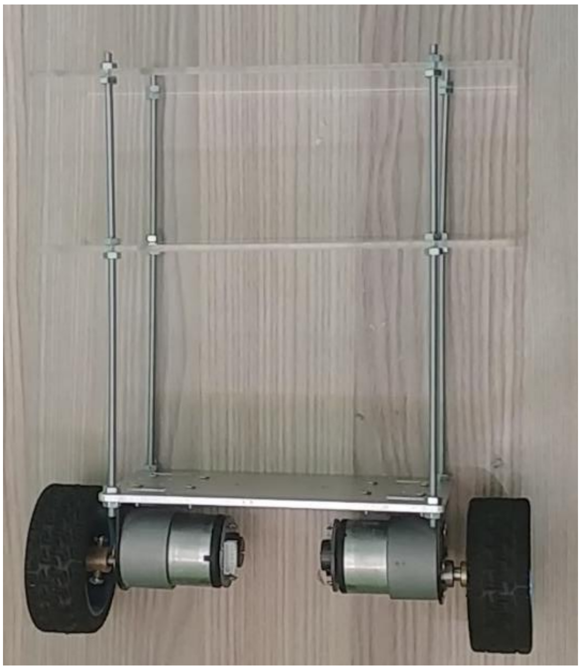

Figure IV.1: *robot chassis*

Arduino UNO

The Arduino board is the main controller for the robot it does all the calculations and commands, it has inputs to get data from sensors, and outputs to give orders to actuators, DC motors in our case.

Figure IV.2: *Arduino UNO board*

Geared 12v DC motors and a pair of wheels

12v DC motors and a pair of wheels that does all the movements this dc motor is attached to a gear box that reduces speed and increases the torque force, it has a reduction of 1to30 ratio. The wheel is made of rubber to have the most friction (prevent sliding)

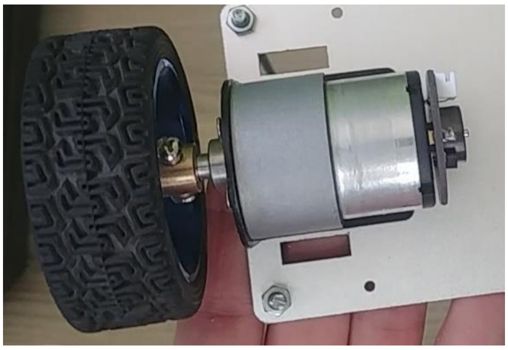

Figure IV.3: *12v DC motor + Wheel*

L298N Motor Driver Module

This module the L298N is a dual H-Bridge motor driver which allows speed and direction control of two DC motors at the same time. The module can drive DC motors that have voltages between 5 and 35V, with a peak current up to 2A.

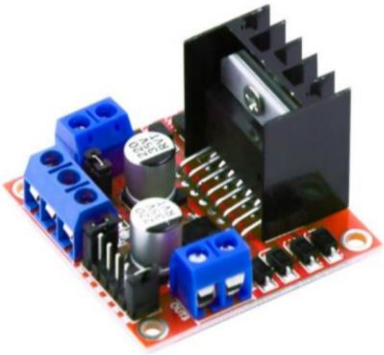

Figure IV.4: *L298N Motor Driver Module*

MPU6050

To drive the motors, we need some information on the state of the robot. We need to know the direction in which the robot is falling, how much the robot has tilted and the speed with which it is falling. All these information can be deduced from the readings obtained from the gyroscope MPU6050. We combine all these inputs and generate a signal which drives the motors and keeps the robot balanced.

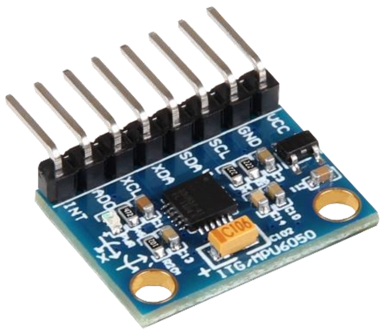

Figure IV.5: *Gyroscope MPU6050*

Three 3.7V Li-ion Batteries

I used three 3.7 V Li-ion Batteries to power the Arduino board that needs at least 5V, and also used to power the motor driver.

Figure IV.6: *3.7V Li-ion Batteries*

IV.2 How does it function

The 3-axis gyroscope of MPU6050 measures angular rate (rotational velocity) along the three axes. For our self-balancing robot, the angular velocity along the x-axis alone is sufficient to measure the rate of fall of the robot.

The data is then fed back to the Arduino board to calculate the command needed using a PID controller, to drive the DC motors.

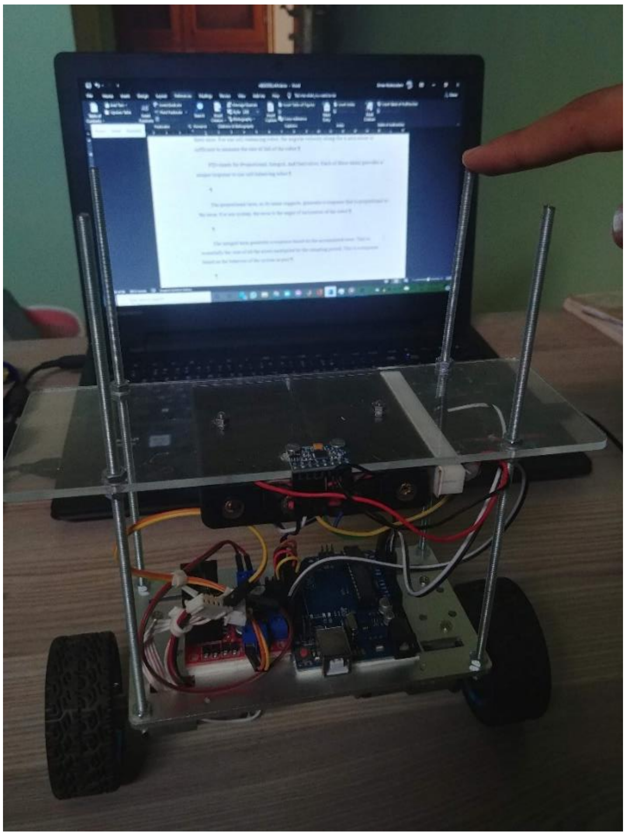

Figure IV.7: *the robot after assembly*

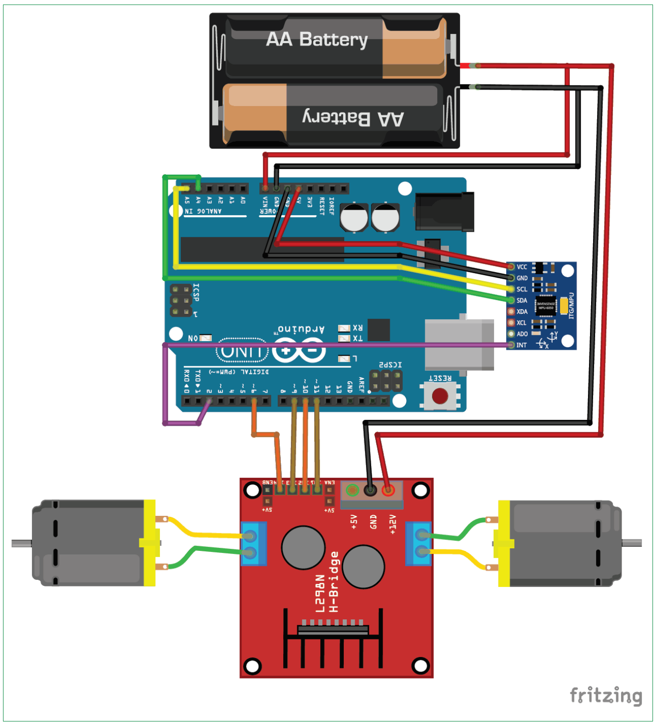

Figure IV.8: *robot wiring*

IV.3 Conclusion

A prototype of the two-wheeled self-balancing robot was designed and built, based on an Arduino board for control and a gyroscope sensor for perception. The prototype has been tested to show its ability to stabilize autonomously.

CONCLUSION AND FUTURE WORK

Conclusion and future work

The objective of the thesis is the design and implementation of a self-balancing two-wheeled robot. The self-balancing robot turned out to be a difficult project and although the system seemed straightforward at first, it allowed many different techniques and concepts to be studied and illustrated, from sensing to modeling to modeling. control and implementation.

First, we introduced some concepts about mobile robots and their types, and gave an overview of the essential features that should be available in all mobile robots, namely autonomous navigation. The basic elements for achieving autonomous navigation are navigation, location, mapping and perception.

After a comprehensive overview of the navigation of the mobile robot, a model of a two-wheeled self-balancing robot was developed, where the kinematics and dynamics of the robot were derived in detail. The self-balancing two-wheeled robot is an example of the inverted pendulum. The robot was modeled based on the inverted pendulum. The results shows that the system is unstable without a controller.

Two controllers: PID and Fuzzy, have been designed to control the robot based on the acquired information from a six DOF gyroscope which supplies the position and the orientation of the robot. The results showed the efficiency of the designed PID compared to the designed fuzzy controller. The fuzzy controller can be improved by adjusting its parameters, namely membership functions and the rule base.

Finally, a prototype of the robot was designed and built. the robot is controlled by an Arduino board, and equipped with a gyroscope allows it to acquire position and orientation information in order to stabilize itself.

A possible direction of future work is to add an inverted pendulum to the robot to obtain a double inverted pendulum system.

REFERENCES

References

[1] S. R. a. N. IR, Introduction to autonomous mobile robots, Massachusetts London, England, 2004.

[2] C. G. a. C. W, "Springer Handbook of Robotics," 2008, pp. 391-410.

[3] R. B. a. C. G. Ferrie`re L, "Design of omnimobile robot wheels," in *Proceedings of the 1996 IEEE international conference on robotics and automation*, Minneapolis, Minnesota, May 1996, p. 3664–3670.

[4] B. G. a. A.-N. B. Campion G, Structural properties and classification of kinematic and dynamic models of wheeled mobile robots, 1996.

[5] C. Y. M. Z. e. a. Shiroma N, "Development and control of a high maneuverability wheeled robot with variable-structure functionality," in *Proceedings of the IEEE/RSJ International Conference on Intelligent Robots and Systems*, Beijing, China, October 2006, p. 9–15.

[6] J. D. a. S. KA, "Modelling and stability control of two-wheeled robots in low-traction environments," Brisbane, Australia, December 2010, pp. 1-9.

[7] G. X. G. W. Cui D, in *Design and stability analysis of a wheel-track robot*, Beijing, China, July 2016, pp. 918-922.

[8] A. J. a. L. J. Han W, "Balancing control of unicycle robot," in *Intelligent autonomous systems*, Heidelberg, Berlin, 2013, p. 663–670.

[9] S. K. a. H. C. Chan RBM, "Review of modelling and control of two-wheeled robots," in *Annu Rev Control*, 2013, p. 89–103.

[10] K. H. a. K. BK, "Online minimum-energy trajectory planning and control on a straight-line path for three wheeled omnidirectional mobile robots," in *IEEE T Ind Electron*, 2014, p. 4771–4779.

[11] M. M. a. H. D. Bajracharya M, "Autonomy for mars rovers: past, present, and future," in *Computer*, 2008, p. 44–50.

[12] G. JP, "Mars science laboratory mission and science investigation," in *Space Sci Rev*, 2012, p. 5–56.

[13] P. Y. a. S. R. Lauria M, "Octopus: an autonomous wheeled climbing robot," in *Proceedings international conference on climbing and walking robots*, Paris, France, May, 2002, p. 315–322.

[14] Y. K. A. K. e. a. Hirose S, "Design of prismatic quadruped walking vehicle TITAN VI," in *Fifth International conference on advanced robotics*, Pisa, Italy, June 1991, p. 723–728.

[15] Y. T. a. I. H, " Evolutionary motion design for humanoid robots," in *GECCO '06 Proceedings of the 8th annual conference on Genetic and evolutionary computation*, Seattle, Washington, USA, July 2006, p. 1825–1832.

[16] H. D. M. I. e. a. Heaston J, "STriDER: self-excited tripedal dynamic experimental robot," in *IEEE International Conference on Robotics and Automation*, Roma, Italy, April 2007, p. 2776–2777.

[17] A. K. a. H. S, "Development of quadruped walking robot TITAN-VIII," in *Intelligent Robots and Systems '96 IROS 96, Proceedings of the 1996 IEEE/RSJ International Conference*, Osaka, Japan, November 1996, p. 208–214.

[18] Z. R. P. A. e. a. Besari ARA, "The study on optimal gait for five-legged robot with reinforcement learning," in *Lect Notes Artif Int*, Berlin, Heidelberg, 2009, p. 1170–1175.

[19] T. A. M. P. e. a. Barreto JP, "FED-the free body diagram method. Kinematic and dynamic modeling of a six leg robot," in *5th International workshop on advanced motion control*, Coimbra, Portugal, July 1998, p. 423–428.

[20] R. J. L. C. e. a. Schiele A, "NanoKhod exploration rover - a rugged rover suited for small, low-cost, planetary lander mission," in *IEEE Robot Autom Mag*, 2008, p. 96–107.

[21] J. S. a. T. T, "Wheeled inverted pendulum type assistant robot: design concept and mobile control," in *Intelligent Service Robotics*, 2008, pp. 313-320.

[22] [Online]. Available: https://electronicscoach.com/closed-loop-control-system.html.

[23] T. Y. R. KRISHNA, in *Speed Control of Separately Excited DC Motor using*, p. 7.