CZECH TECHNICAL UNIVERSITY

IN PRAGUE

FACULTY OF MECHANICAL ENGINEERING

BACHELOR'S THESIS

ROBALL ROBOT

2023

Juan Gustavo Maldonado Quispe

CTU CZECH TECHNICAL UNIVERSITY IN PRAGUE

BACHELOR'S THESIS ASSIGNMENT

I. Personal and study details

Student's name: **Maldonado Quispe Juan Gustavo** Personal ID number: **481750**

Faculty / Institute: **Faculty of Mechanical Engineering**

Department / Institute: **Department of Instrumentation and Control Engineering**

Study program: **Bachelor of Mechanical Engineering**

Branch of study: **Information and Automation Technology**

II. Bachelor's thesis details

Bachelor's thesis title in English:

Roball robot

Bachelor's thesis title in Czech:

Roball robot

Guidelines:

The goal is to redesign a Roball robot. The robot needs better mechanical solution, better internal arrangement.
The project involves those steps:
1) mechanical redesign (some parts). Parts can be 3D printed
2) electronics ? connection of stepper motors and servo, controlled by Arduino. Electronics will be provided

Bibliography / sources:

Bishop O.: Robots builders cookbook: Build and desing your own robots, 1st edition, Newnes, 2007, ISBN 978-0750665568

Name and workplace of bachelor's thesis supervisor:

doc. Ing. Martin Novák, Ph.D. Division of electrotechnics FME

Name and workplace of second bachelor's thesis supervisor or consultant:

Date of bachelor's thesis assignment: **27.10.2022** Deadline for bachelor thesis submission: **26.01.2023**

Assignment valid until: ___________

doc. Ing. Martin Novák, Ph.D.
Supervisor's signature

Head of department's signature

doc. Ing. Miroslav Španiel, CSc.
Dean's signature

III. Assignment receipt

The student acknowledges that the bachelor's thesis is an individual work. The student must produce his thesis without the assistance of others, with the exception of provided consultations. Within the bachelor's thesis, the author must state the names of consultants and include a list of references.

27-10-2022
Date of assignment receipt

Student's signature

CVUT-CZ-ZBP-2015.1

Table of Contents

Abstract

This thesis presents the design, development, and testing of a spherical robot with holonomic motion. It is discussed why a spherical shape robot is of particular interest among the shape of robots. The objective of the thesis is to design, build, and test a sphere robot that can move in any direction, and to evaluate different concepts developed in the past to achieve this objective, in order to have the most efficient one as a base for the design. Also, it is discussed all the issues and concerns during the process of the physical structure and the control systems design, as well as some solutions implemented or possible to implement in future designs.

Keywords: Sphere Robot, Rolling motion

Introduction

In the nowadays world, robots have become more and more common as they can be used to replace humans in many tasks that are sometimes too repetitive or even in many tasks that humans would not be able to perform by themselves. As a result, robots can be in many forms, many different shapes, and many different sizes as well. All depending on the purpose it is intended to be used at.

Mobility is one of the main aspects to consider in the design if the robot is intended to move. In this sense is that the spherical shape of a robot and its rolling motion offers some advantages over other types of motions, like walking sliding or hopping. They can be designed to be holonomic, hence move in any direction. This offers advantages over other designs that would require some space to turn or rotate before heading to the desired direction. Additionally, they cannot be overturned, or be upside down. This is a great advantage over other designs, like the wheeled ones or the walking robots, that need to be in the right position for it to work properly.

An additional advantage that this shape design offers is the possibility to have a hermetic shell inside which all the control system are located, which protects it from all the external environmental conditions like dust, humidity, temperature and even water, giving this kind of design a wider range of applications.

For these reasons is that the purpose of this thesis is to design, build and test a sphere robot capable of having the holonomic motion. However, there is more than one way to achieve this, in fact there are many different concepts developed over the past decades.

When it comes to the different concepts, the general objective is the same, to have a sphere shape robot capable of moving in different directions. This objective was persuaded by many different designers that attempted different concepts to achieve this. It is important to consider all of them to see what similarities they may share and what are the advantages that each of them may give in terms of design and efficiency in order to choose the most appropriate one for the purpose of this thesis.

1. Evaluation of different concepts

As mentioned before, there are a variety of different concepts or sphere robots designs that follow these different concepts, to achieve its motion objective, that will be analyzed in this paper prior to deciding one of them as the most suitable for our purpose. According to [1] it is possible to divide these mechanisms used for movement into 3 groups:

- control by moving the center of mass,

- control by generating variable gyrostatic momentum,

- control by deforming the shape of the ball.

In our case we will be concentrating on robots that fall into the first 2 categories as they are the more viable and realistic to achieve.

1.1 Spherical Robot Using rotors.

In this section will be described two robot designs that basically use the conservation of momentum, angular momentum, as a principle for motion of the sphere. In both cases, the layout of the robot was defined by the idea that the center of mass to be centered in the geometric center point of the ball, requiring symmetrical or at least balanced weight distribution. One uses 2 axes for the placement of its rotors while the other uses a 3-axial system.

In Figure 1 we can see a schematic of how the one using 2 axes would look like. As you can see, it does not have an exactly symmetric layout, but it does have a “dead weight” used to balance both sides of the ball and hence move the center of mass to the middle, which as mentioned before, is an important aspect for the correct functionality of the robot.

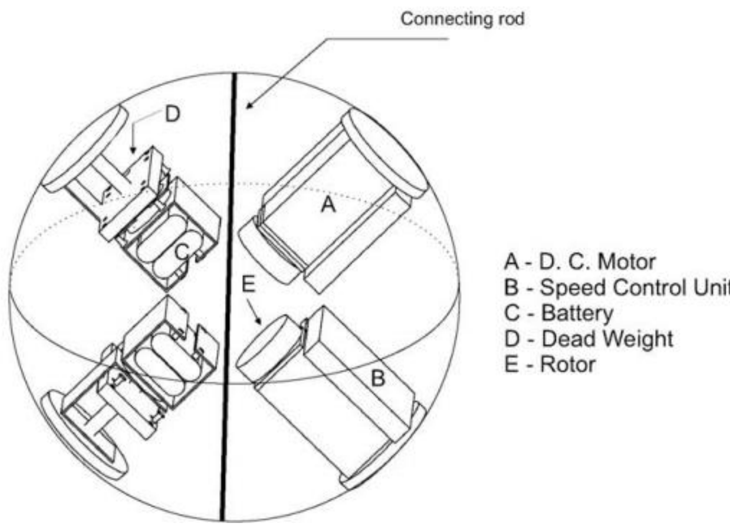


Figure 1 Schematic of a 2-axial system with 2 rotors details of its parts [2]

The way in which conservation of angular momentum is used for the movement of the robot is basically that once one of the motors start to rotate, it will make the rotors rotate, however because of the conservation of momentum, it will create an equal opposite reaction into the DC motor which is connected to the shell, hence making it turn.

Figure 2 Construction of the Robot Model [2]

In Figure 2 we can see the model that was built by [2]. The kinematic model of the system is developed using quaternions for the description of the orientation of the robot. This was later tested in two main areas where the expected movement can be predicted, when the rotor is in vertical position and in horizontal position. In their case, the vertical position worked as expected except at slow or gradual velocity changes. However, the results for horizontal were not as satisfactory when it should rotate in a straight line. This may have been due to imperfections of the sphere surface or other issues related to the design as the rotor parameters.

Another similar design can be seen in Figure 3 in which instead of 2 rotors it uses 3 rotors placed in the equatorial plane. The principle it uses for movement is the same as the previous shown design, however the equations used for the control of the movement of the robot are different. In this case the equations of motion are written in the form of Ferrers equations in quasi-velocities with undetermined multipliers [1]. The only difficulty on the design of this type of robot is, once again, to ensure the center of mass at the geometric center of the spheric shell.

Figure 3 A 3D design (left) and a full-scale photograph (right) of the design of the robot with 3 rotors [1]

1.2 Spherical robot using pendulums.

In this section it will be discussed few different designs that use pendulums as the basis of their principle of propulsion or turning. Even though among these designs the purpose of the pendulum as a feature of the design may be different, the basis is the same. Contrary to the concept of using rotors, these robots take advantage of a center of mass uncentered with the geometrical center of the sphere and by altering its relative position with the sphere shell is that they obtain their objective.

One of the robots that will be discussed is called "Rollo". This is a 2-dof motorized spherical robot which, as mentioned before, uses the concept of hanging pendulum for the purpose of its motion. It has basically two main motion mechanisms inside, one that it allows to rotate horizontally inside the shell and one which makes the shell roll over the surface. In Figure 4 we can see how the layout of such mechanism would look like. As you can see, it consists of a rail along the horizontal axis that allows the robot to rotate inside the shell to select the direction in which it will later roll.

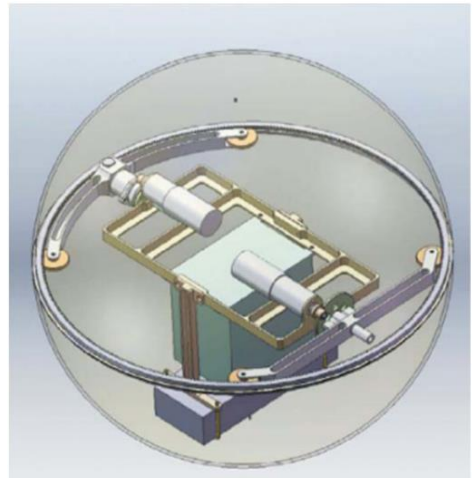


Figure 4 A three-dimensional model of the internal mechanism of a robot using pendulum [1]

In Figure 5 it shows the three generations of the Rollo and the latest one is the one that resembles the most the model shown in the Figure 4. In this image it can be seen better the gear on the sides of the robot which are part of the rolling mechanism of the robot.

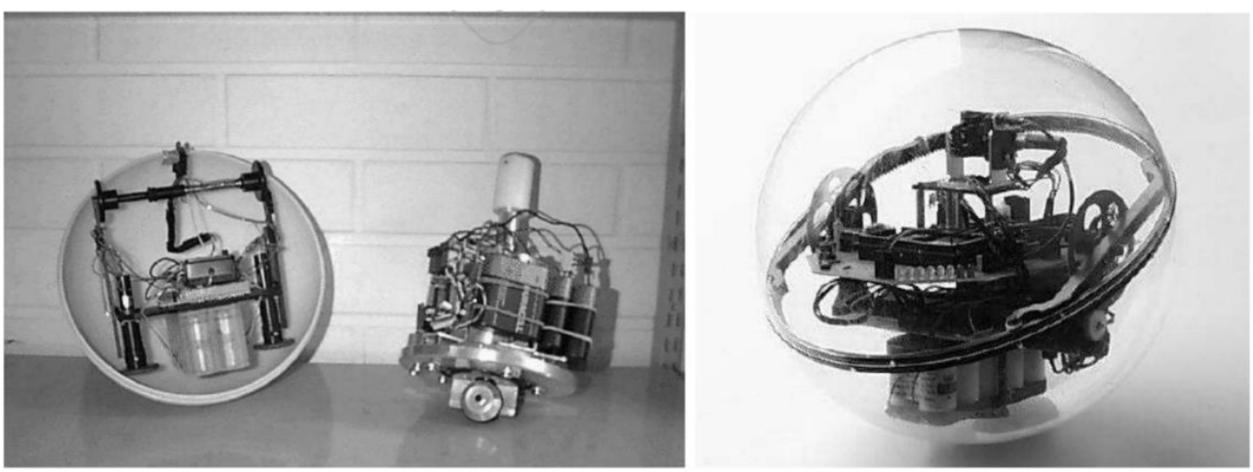


Figure 5 2nd, 1st, and 3rd generation of the Rollo [3]

When it comes to controllability, as already mentioned, selecting the direction is quite simple, however the challenging part is that the horizontal rim gear must be aligned with the horizontal plane, which is achieved every half-revolution, so every time it rolls it has to move a full number of them. These revolutions are counted by means of an inductive sensor which help determine the correct position of the mechanism. In theory, active controllability of the direction would be possible but that is a much more demanding task to achieve.

Another robot that uses the pendulum concept is the “Roball”(Figure 7) developed in Canada with the objective of being a children’s toy [4], due to many of the advantages that its sphere shape offer. In contrast to the Rollo, Roball takes advantage of the pendulum for both, the turning and the rolling mechanism of the robot. In Figure 6a. it can be seen the parts it contains inside the robot and how would activating the mechanism make the robot roll with the help of two DC motors. On the other image, Figure 6b., it can be seen a rear view of the robot and understand better how the angle at which the battery hangs from the plate is altered with the help of the servo motor to make the tilt so it turns while its rolling, showing how its steering mechanism works.

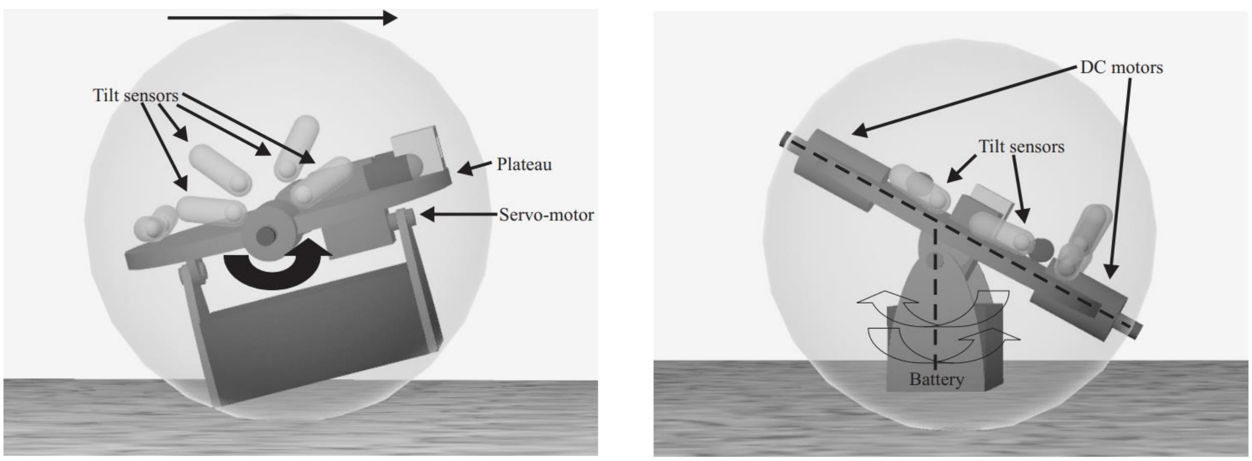


Figure 6 a) Right side view of Roball showing Propulsion mechanism. b) Rear view of the Robot showing the steering mechanism while it turns to the right [4]

The robot gets its data of the inclination via tilt sensors installed in it, four tilt sensor for the longitudinal inclination located on the right side of the robot (Figure 6a.). Meanwhile two tilt sensors located in the back for the purpose of measuring lateral inclination (Figure 6b.).

The main difference of the Roball and Rollo over its controllability is that, as seen before, the Rollo needs to be in an stationary position to select the rolling direction, while Roball needs to be in motion to be steering and change the direction of movement, now whether one or the other is better depends on which task it is designed for, but is something to take in consideration when deciding which design to go for.

Figure 7 First prototype of Roball [4]

1.3 Robots using a wheeled internal mechanism

In this section it will be discussed many different types of robots that has include wheels inside the sphere for the motion of the mechanism inside. Even though some of these designs may seem as if it consisted of a wheeled "vehicle" inside of a sphere, if we take the mechanism with the shell as a whole, they will fall into the same previous stated category of robots that moved their center of mass for the purposes of its motion.

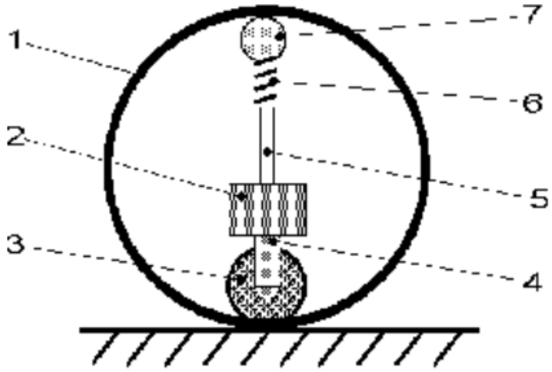


Figure 8 Structure of the Rolling Robot.1. robot body (case), 2. controlling box, 3. driving wheel, 4. steering axis, 5. supporting axis, 6. spring, 7. balance wheel [6]

In Figure 8 it is illustrated the idea of the first robot discussed in this part, the “Rolling Robot”. As seen in the illustration, the concept is quite simple, it consists of a driving wheel with contact to the bottom part of the sphere which when turned moves the internal mechanism forward, changing the center of mass and thus making the robot roll. The way the steering mechanism works shares the same principle as one of the mentioned in part 1.1, it uses the inertial moment to turn the shell. By making rotate a disk inside the robot, an inertial propulsive force is generated that results in the desired rotation. In Figure 9 is it possible to have a better look of the internal mechanism of a real-life prototype of this kind of robot, and the disk mentioned.

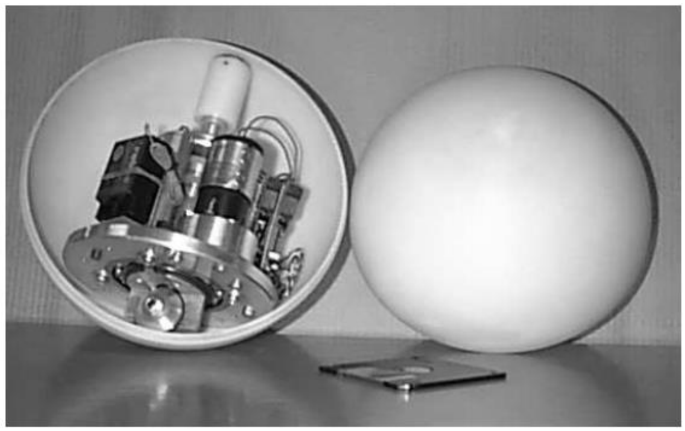


Figure 9 A prototype of spherical robot [6]

There are a couple of issues regarding this concept that must be taken in consideration when deciding for this concept for a sphere robot. One of them is that is not easy to control the direction in which the robot is pointing towards to, this is due to how friction plays a determining factor when differentiating an ideal theorical model and a real case scenario. In Figure 10 it can be seen a very similar design that addresses the problem that friction could represent on the inner surface of the shell by adding a fixed trail in which the wheel rolls and which material can be selected to have the desired friction.

Figure 10 Another more simplistic design of a spherical robot [1]

Even though one of the issues may be overcome in the later example, the friction value between the outer shell and the surface over which it moves is still a quite hard value to predict, which in turn will affect the prediction of which direction the robot is pointing towards to. However, it can still be measured the distance it travels by means of the number of revolutions the lower wheel has done.

The next wheeled robot that will be discussed is the "Sphericle". This robot concept of motion is basically the one of a "car" inside a wheel. This can be view more graphically in the Figure 11 which is part of the dynamic model used in [5] paper.

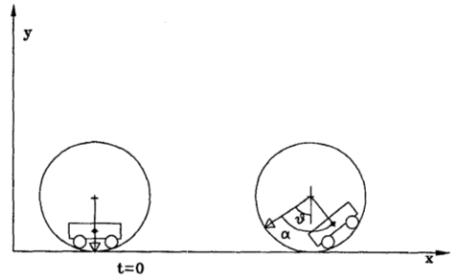


Figure 11 Dynamic model of the sphericle moving in a plane [5]

In this model, the inner vehicle consists of 2 driving wheels, which are connected to a stepper motor each through a belt reduction gear each. It has 2 additional support wheels which work as a suspension and also help maintain the balance of the robot which can be seen better in the Figure 12. A free pendulum is mounted in the cart, which angle is measured by an encoder to be used in a local feedback stabilization controller along with the wheel position. The steering mechanism is similar to the next and last robot that will be discussed.

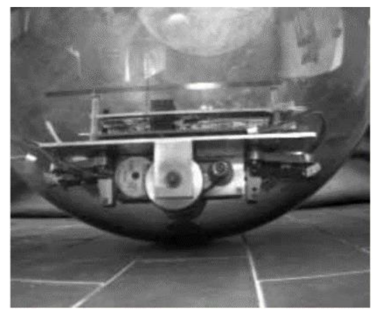


Figure 12 Prototype of the Sphericle [5]

Finally, the last robot concept discussed is the "Sphero". This design could be seen as a union of both previous two concepts discussed in this section of wheeled robots. Its propulsion and steering mechanism works the same way as the Sphericle but for helping of the stability it does not include the wheels in the front and the back, instead it has a similar design to one described by [6] in Figure 8, with 2 wheels in the top part and springs that ensure it is kept in place.

To have a better view, in Figure 13 it is possible to have a really clear picture of the parts that one of the models of Sphero includes: the springs, the wheels, the main motherboard, the DC motors, etc. To go deeper into the propulsion and steering mechanism, it is a very simple concept.

To move forwards or backwards, both motors make the wheels turns in the same direction, while to make it turn, they move in opposite directions depending on the direction towards which they want to turn. It also possible to observe the batteries and the coil in the bottom part used as an inductor to make it possible charging without disassembling the sphere.

Figure 13 Inside the Sphero SPRK edition [11]

This design proved to be very successful, as is one of the most well-known sphere robots nowadays. It also overcomes some issues that other concepts had, for example, unlike the Sphericle, this does not depend solely in the weight for keeping in touch with the surface of the shell, or it does not have the risk of being flipped over to an unrecoverable position.

2. Physical Design process

In this section, as the name suggest, it will be discussed mainly all things related to the design process of the physical part of the robot, so basically all the hardware. There are certain parameters that will determine how the design will go, starting with the concept that will be chosen to be followed. From there other limitations will guide the way on to an initial 3D model of the robot.

2.1 Concept chose decision

When it comes to the decision of choosing a concept to be followed, there are some key aspects to be taken in consideration. first of all, how simple is the maneuverability of the concept. Given that it will be the first prototype to be built by myself, it will be attempted to persuade a design as simple as possible, and the simpler it is to control the motion of the concept, also the easier it would be the design of it. In this sense, three of the concepts discussed previously stand out, the 3rd generation of the Rollo, the Sphericle and the Sphero.

The Rollo design is pretty straightforward, it contains rail on the horizontal axis over which it rotate to choose the direction in which it will rolling and then starts rolling. The main limitations are that it is simple to control the direction every half turn, limiting either the motion of the robot in this concept, or the simplicity of the maneuverability. The second main limitation is how simple or, on the other hand, how complicated it would be to have a rail on a sphere. For this project we will be counting on plastic balls available on the market and mounting a rail on one of them would not be the most suitable design.

In the case of the Sphericle and the Sphero, both share a very similar design when it comes to the propulsion and steerability given that both use the same concept of having 2 driving wheels at the bottom part. The main difference is what they both use to balance. The Sphericle has 2 support wheels in the bottom part too that give the robot a "car-like" concept inside a sphere, while the Sphero has 2 support wheels that go all the way to the top part, not only giving balance to the robot, but also helping in ensuring the driving wheels are in contact with the bottom part. That is where it could be considered a design advantage over the Sphericle which relies on its weight for that.

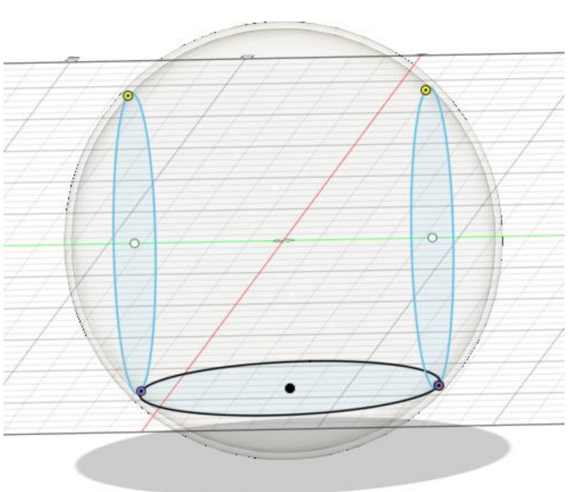


Figure 15 Rotation and Steering axes of a robot-like Sphero

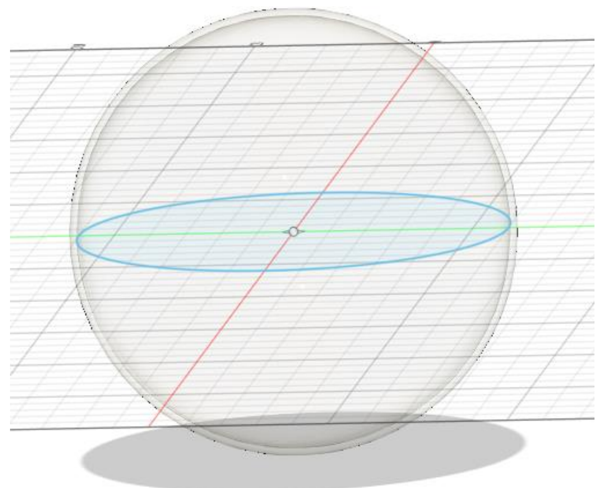


Figure 14 Steering axis of a robot like Rollo

If we compare both methods of steerability of the Rollo and the Sphero, we can identify two "steer axes". In Figure 14 is it possible to see the one of the Rollo, that as we know, is where the rail over which the robot rotates is located. It possible to identify such corresponding axis for the Sphero (Figure 15) which is where the wheels are located, which in consequence are also the trails of such wheels when the sphere is rolling. The main advantage of Sphero in this matter is that given axis is not fixed, so it moves as the robot rolls, maintaining it in parallel to the horizontal plane, while the Rollo depends on the position of the rail at any given moment, limiting its steerability.

In summary, a design concept similar to the Sphero (Figure 13) meets the criteria of simplicity we are looking for and has the advantage on the maneuverability over other designs, and for this reason is the concept it will be implemented for this first version of the Sphere Robot ball.

2.2 Limitations and things to have in consideration.

Given the wide range of possibilities that are possible to build for the first prototype, even after having quite a defined concept to follow, the way it will be approached this design process is to have some things in consideration, first and mainly in terms of dimensions, to have an starting point. To start with, the robot will be a mix of electronic components available either at the lab or at a conventional electric store, also the spheric shell will be purchased. While the structure and other necessary parts needed to hold the components together will be mainly 3D printed, given the access to one at the laboratory of the University.

To begin with, we determine the main components to be needed for the robot:

- A programable microcontroller board- this will be the brain of the robot that control all its functions
- A motor driver integrated circuit (IC)- this will be the intermediary between the controlling board and the DC motors to be able to control the DC motors in both directions (a dual H-bridge) and at higher current than supported by the controlling board
- DC motors- to drive the wheels of the robot
- Battery/batteries- to power the controlling board, motor driver, and DC motors
- Bluetooth module- to be able to communicate and control the robot wirelessly (since the board is encapsuled inside the shell)

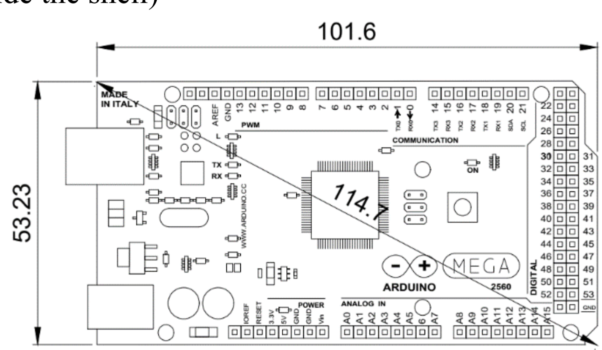


Figure 16 Arduino Mega 2560 and its dimensions in mm

For the programable microcontroller board, it will be used an Arduino mega 2560, its dimensions main dimensions can be seen in Figure 16 of which the most important is the diagonal (around 110 mm) since it is the one that will determine the largest dimension to fit inside the sphere. Other of the components which are already chosen, and of which are known the dimensions are the battery and the motor integrated circuit.

For the battery it was decided to use two 3.7V Li-po batteries with capacity of 800mAh. They were chosen given that both connected in series have the capacity of powering the Arduino board, given that this one counts with a voltage regulator, that allows the 7.4V to be suitable for it, and that it will be possible to obtain either 3.7V or 7.4V for the DC motors, which will be the last components to be chosen. The dimensions of the batteries are 52x29x6 mm. While for the motor integrated circuit it will be used the L293D (Figure 17), quite a common motor driver used for this type of applications. Given that it can supply up to 600mA of current to each motor, it will be also suitable for quite a range of DC motors. The dimensions of it are around 39x35x11mm.

Figure 17 Motor Driver L293D

The second part mentioned to have in consideration is that great part of the structure would be 3D printed, even the gears that eventually will be needed between the wheels and the DC motors. This is important to consider while designing the internal structure of the Robot because of 2 main reasons, the level of precision is limited to the one of the 3D printer and second, is that even if there is the possibility to print very complex structures with the 3D printer, ideally it will be aimed to use the less machining after printing, hence ideally have parts that do not require support while printing. To obtain that, the parts should have at least one flat face and the rest of the structure be an extrusion of it. An example of it is the Figure 18 where we can see how the body is formed from the extrusion of it.

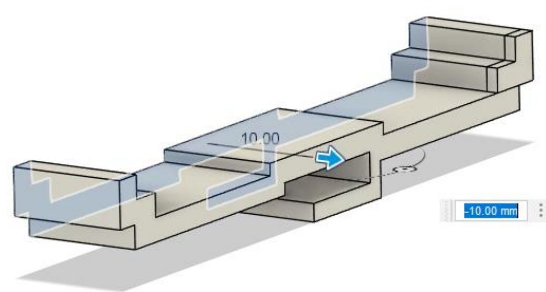


Figure 18 One of the parts of the structure formed from the extrusion of a 2D shape

2.3 Design of internal layout and structure.

Given that we have a desired concept to follow, we can have an idea of a rough layout of how we aim to have the parts inside the robot. In Figure 19 it is shown the first sketch of the internal parts of the robot, to have an idea of how it may look like. For the 3D model design of this it will be used 2 main 3D modeling software, 360 Fusion & Inventor, both by Autodesk.

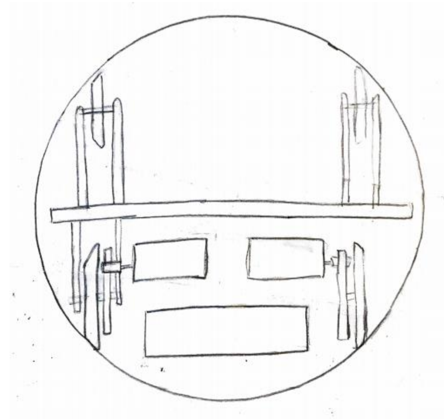


Figure 19 Initial sketch of the layout of the internal parts of the robot

After considering the largest component to be used inside the robot to be the microcontroller board, this dimension will be the one used to determine the size of the sphere shell to be chosen. As mentioned before, this shell would be also purchased and the available options are limited, there are shells of 10,12, or 14 cm and so on in smaller or larger diameters. The largest dimension of the board was around 11cm but the margins inside the sphere were too small if it was chosen a 12cm-diameter shell, so it was decided that a 14cm diameter shell would be more optimal.

The way the structure would be designed is to place the initial components inside the 3D modelling software in the position it would be desired and build the structure around it. In Figure 20 it can be seen these ones, which would be the wheels, the board, the batteries, and the DC motors. Their position where not randomly chosen but all have a reason for it. If we look back at Figure 15 (the rolling and steering axis), similar axes have been chosen for this robot. Both are they were determined by placing them in equidistance planes from the origin in both Horizontal and vertical way, (Figure 20) resulting in both, the steering and the rolling axes, to have same diameter and same circumference. This for instance, result in a similar force and torque needed when either rotating or rolling the ball. The wheels are consequently

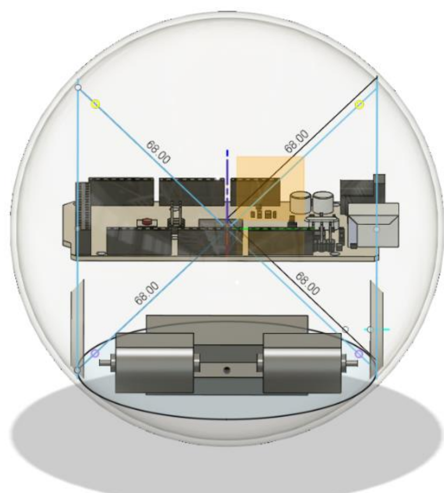


Figure 20 Placing of the main components of the robot and the wheels with reference to the steering and rolling axes

placed on top of those axes, as seen in the Figure 20. The board was placed in such a way that there would be the most space between the board and the shell, giving enough room for the future cables that will interconnect the components. The batteries were positioned in the lowest position possible, while still giving some margin for building a structure around it, to lower the general center of mass of the robot, in order to obtain the desired effect of it having a standing position at rest, meaning the sphere should balance itself.

The material to be used for 3D printing is PLA, and it is relatively light and of strong for the desired purpose in this case, given the low weights managed in the robot, it would be neglected the need of structural calculation of the parts, however a standard thickness of 2mm would be the common to be used in most parts of the structure. Having stablished that, it can be proceeded to build the structure around it, always having in mind the already stablished considerations given regarding the 3D printed structure. Most of the parts are designed in such a way that it would not require any extra material to join the parts together in place. Meaning that it is taken advantage of the material flexibility to obtain interference fitting among the parts, as well as other methods to ensure the parts are kept in position, as seen in the example Figure 21.

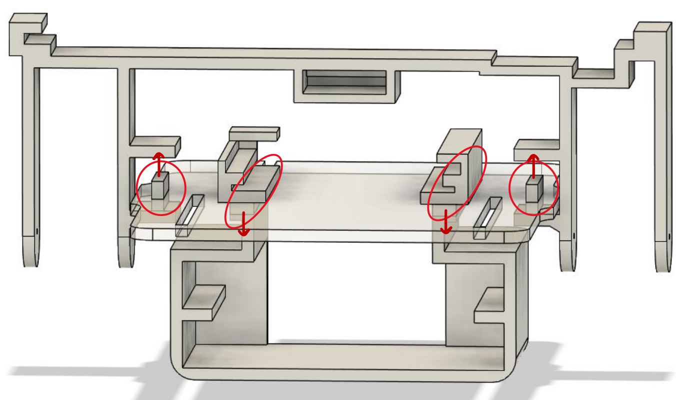


Figure 21 Example of how different parts are kept in position, in the image they both fall into place one into the other and weight keeps them in place

It was later proceeded to design the top part of the structure where the support wheels are placed. It was used the same principle as the driving wheels for the positioning of the support wheels, hence they were placed in the same rolling trail circumference but in the opposite side. These support wheels not only have the mentioned function of supporting, but also accomplish a similar one to a suspension, and as such, springs were implanted in its structure. To ensure the contact with the shell and the support wheels, they were designed in such a way that the height of the wheels will be larger than required when the springs are uncompressed.

2.4 Driving mechanism design

In this part, the driving mechanism will be designed, from the motor and gear ratios of the transmission. Firstly, to have a better understanding of how the motion of the sphere robot works, it can referred to the motion model explained by Halme in his paper [6]. In Figure 22 we can see some of the variables that are used in the motion model equation. The driving torque can be expressed as the following equation (1).

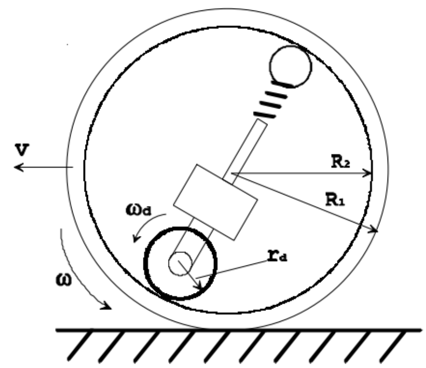


Figure 22 Model of a sphere robot by Halme [6]

$$I\alpha = l \times m = rm\sin\theta \qquad (1)$$

If now we implement the Inertial moment of the ball I_{ball} that can be obtained from eq (2), and combined with the previous stated driving torque, we can obtain that the angular acceleration α goes as the following equation (3).

$$I_{ball} = \frac{2}{5}M_{ball}\frac{R_1^5 - R_2^5}{R_1^3 - R_2^3} + \frac{M_{ball}}{4}(R_1 - R_2)^2 \qquad (2)$$

$$rm\sin\theta = \left[\frac{2}{5}M_{ball}\frac{R_1^5 - R_2^5}{R_1^3 - R_2^3} + \frac{M_{ball}}{4}(R_1 - R_2)^2\right]\alpha \qquad or \qquad \alpha = Ar\frac{m}{M_{ball}}\sin\theta \qquad (3)$$

Now, we only need to consider the first equation of the driving torque and implement it in the following model (Figure 23) to obtain how the driving torque and the motor torque M_{KM} are related. Also, it must be considered the inertia of the mass and the force m_I it exerts on the system in an acceleration case. The principle of transfer of the torque from the motor to the wheels is through a pinion and a gear, from which we obtain the transference number i_p depending on the number of teeth z_1, z_2 on them. Given two wheels are mounted in parallel using the same specification motors, the torque is multiplied by 2. Using the torque of the wheel and its radius we can obtain the force between the wheel and the sphere, which is equal to the one from the angular acceleration of the shell and the one by the acceleration of the mass in a horizontal plane. In eq. (7) can be seen the result of this simplified case. It is worth mentioning that in this simplified case, a non-slippery condition between the wheels and the inside surface of the shell of the robot is assumed. In this scenario, the torque of the DC motor implemented mainly should determine the acceleration that can be obtain from the ball, but in theory any torque would produce a motion of it. However, many factors were ignored during this process that would require a more extended research and testing for accurate results, such as the friction in shaft and wheels, the actual efficiency of the gears, the coefficient of frictions inside the ball. Also as seen from the model, the previously mentioned lower center of mass is an important determining factor in defining the motion of the sphere robot.

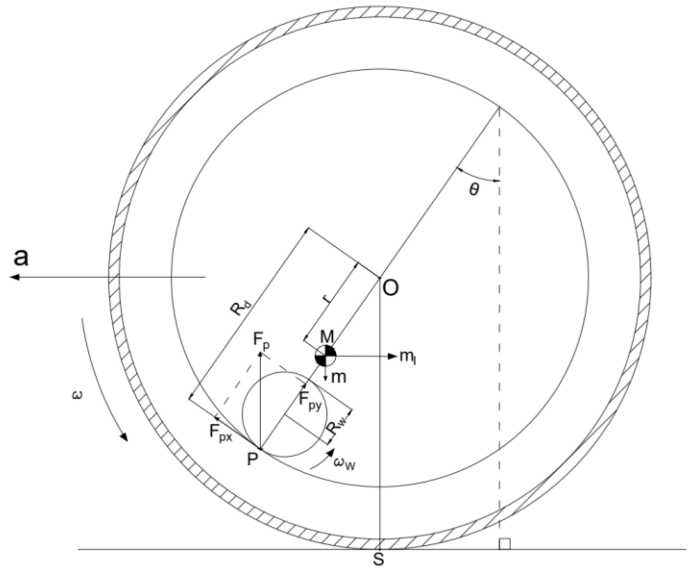


Figure 23 Model of the sphere robot adapted to the actual design

$$F_{px} = M_W \cdot R_W = M_{KM} \cdot i_p \cdot \eta_p \cdot R_W \qquad (4)$$

$$i_p = \frac{z_2}{z_1} \qquad (5)$$

$$m_{Ix} = m_I \cdot \cos\theta = \alpha \cdot R_1 \cdot m \cdot \cos\theta \qquad (6)$$

$$2 \cdot M_{KM} = \frac{\alpha}{R_d \cdot i_p \cdot \eta_p \cdot R_W}\left(\frac{M_{ball}}{A} + R_1 \cdot m \cdot \cos\theta \cdot r\right) \qquad (7)$$

The support and assembly for the motor and the gear was designed in such a way that it would be possible to have different motors implementation and gear ratios during or after testing, depending on the requirements needed.

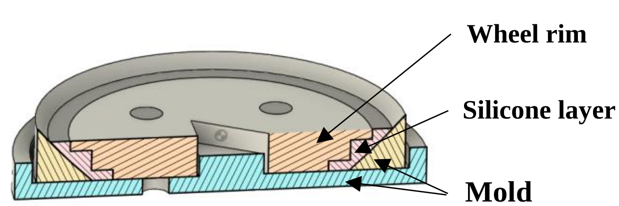


Figure 24 Half-view of the wheel rim, the mold and the silicon layer filling

When it comes to the wheels, the mounting to the gear was as shown in Figure 25, to ensure the transmission of torque between both parts. Additionally, it would be expected that a direct contact between the 3D printed wheel of PLA and the plastic shell would be of very low coefficient of friction, hence for the outer part of the wheels it was used a negative printed mold and a base rim over which silicone was filled as shown in the Figure 24, given the higher coefficient of friction between silicone and the plastic.

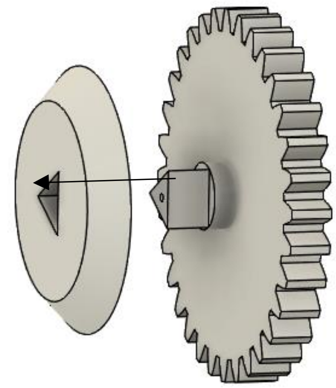


Figure 25 Mounting of the wheel into the gear structure

After all this, it is possible to build a 3D CAD assembly of how the sphere robot would look like with the addition of the main components as well, shown in Figure 26.

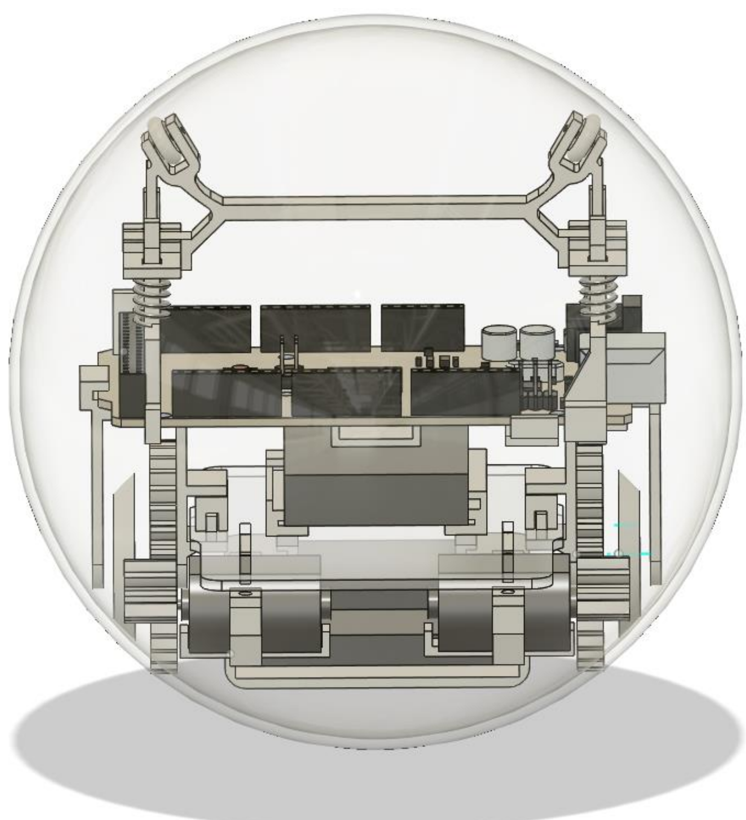


Figure 26 Completed 3D CAD model of the sphere robot

3. Design of the control system

For this part, the main focus is the design of the control system of the robot and all the components that form part of it. The design process is divided in two parts, the hardware part, and the software. The hardware part concentrates on the components needed for the control system and how they interconnect with each other while the software part is the designing of the control process and the programing of the control code that controls the robot.

3.1 Interconnection of the control parts.

It was previously mentioned the need of different parts for the control of the robot. The programmable microcontroller, the Bluetooth module and the motor driver, as well as DC motors that are the actual actuators for that give the motion to the robot. In the following diagram (Figure 27) we can see the block diagram of the communication between these parts.

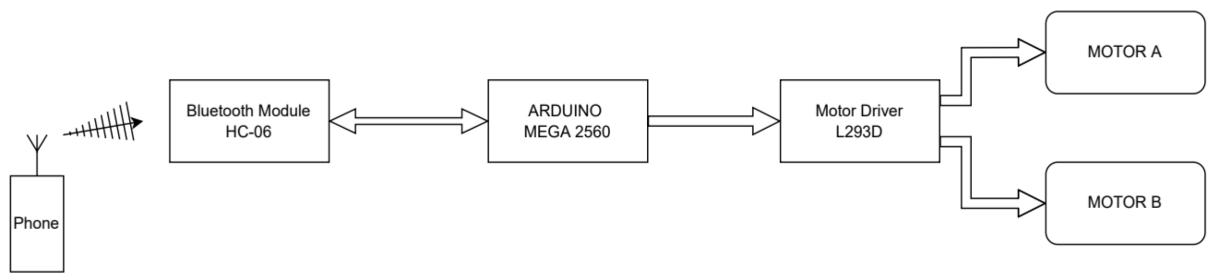


Figure 27 Block diagram of the communication between the components of the control system

As seen on such diagram, the only interconnection that goes both ways is the Bluetooth module and the microcontroller board, for the rest of the parts, communication only goes one way, even though the Bluetooth module and the phone have the capability of doing so, due to the controlling application interface shown later in Figure 31 there is no actual feedback that can be display in it during the remote controlling of the robot.

The shown communication block diagram is a quite a simplified view on how the components interact with each other, however in Figure 28 it can be seen in more detail of the connections that each of the components have among them, here it can be seen that the microcontroller is the main part of all it. Also, here it can be seen the connection with the batteries that feed this system.

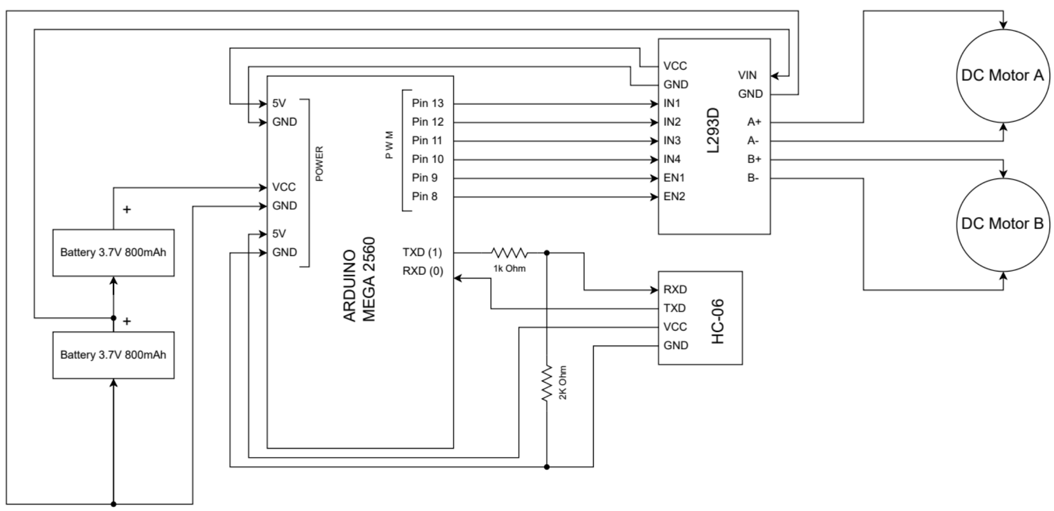


Figure 28 Detailed schematic of the connections between the components of the control system.

In such figure is possible to see two details that are important to remark. The two batteries are connected in series to power the Arduino in the voltage of 7.2V. This to meet the recommended voltage source range of 7-12V. The second detail that can be seen is the connection between the TXD pin on the Arduino and the RXD pin in the Bluetooth module. This is connected in a voltage divider arrangement using a 1K and a 2K Ohms resistors. This given that the signal output from the Arduino is in 5V, however the Bluetooth module recommended voltage input for such signal is 3.6V.

3.2 Controlling program.

As seen in the previous section, the Arduino is the part in charge of controlling the motor driver and hence the DC motors, but the microcontroller board cannot send signals arbitrarily to the motor driver, it must have some defined parameters with a define objective. For this reason, is that it was defined 5 states of motions. Each of them has defined parameters for the motor driver to control the motors. These are shown in the following diagram.

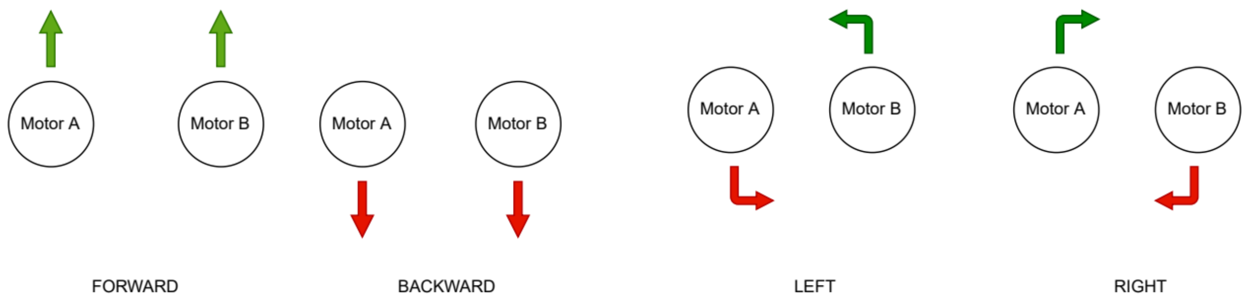


Figure 29 Combination of the motor actions for 4 of the different state of motions

As seen from it, to achieve the different motions, the motors must be activated in different ways. In the diagram it can be seen 4 of such motions, the fifth one which is not shown being STOP, for which both motors are required to have a resting state, (not moving).

The program that will control the signals send to the motor driver decides the state of motion according to the character available (or lack of it) sent by the Phone application through the Bluetooth module. The program then keeps running in a loop to keep awaiting for a new character to update the state of motion, or keep the last one running. This decision process can be visualize in the following flowchart.

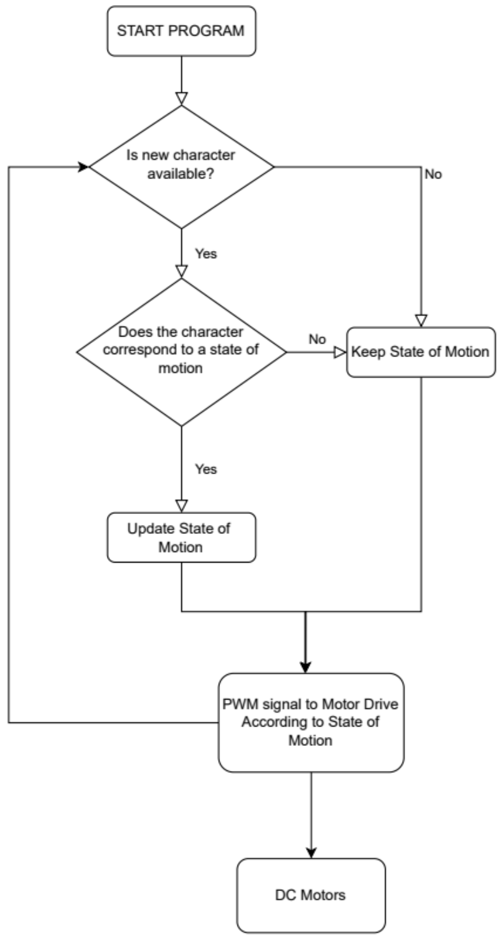


Figure 30 Flowchart of the controlling program

The application that is used to connect with the Bluetooth module counts with its own interface and allows for configuration of the sending signal according to the button pressed. As seen in the images, it also sends another signal ("0" by default) upon the release of the button. Consequently, as long as the button is kept being pressed, it will not change state.

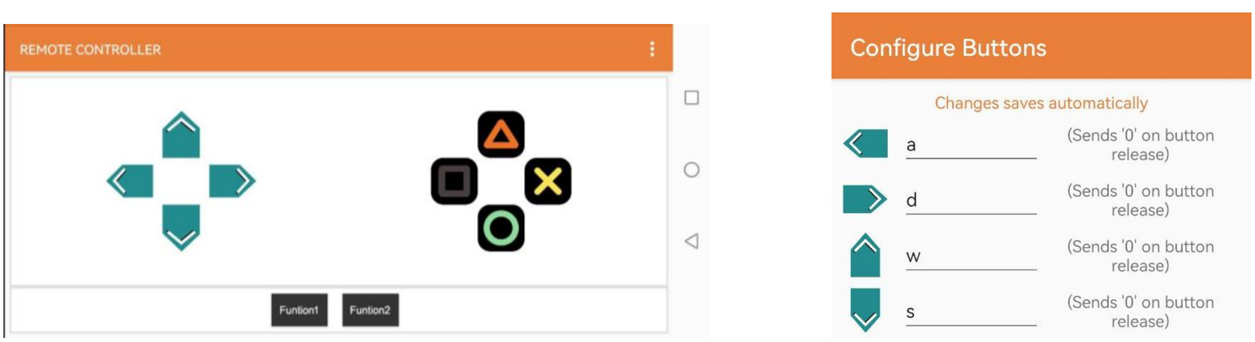


Figure 31 a) Interface of the controlling app of the phone (Left) b) Configuration of the buttons in the controlling app (Right)

The controlling board Arduino counts with its own cross-platform application for programming on desktop, and its programming language is similar to C++. In the Figure 32 it shows the code use for controlling the Robot, and it uses the logic shown in the flow diagram. In the first part of the program, it declares the pins to be used and to which connection with the motor driver it corresponds to, as well as the variables to be used. Also, the pins that connect with the Bluetooth module are declared.

```
// Motor A connections
int enA = 9;
int in1 = 12;
int in2 = 13;
// Motor B connections
int enB = 8;
int in3 = 11;
int in4 = 10;

char command;

#include <SoftwareSerial.h>

SoftwareSerial BTSerial(0, 1);  // RX, TX

void setup() {
  // Set all the motor control pins to outputs
  pinMode(enA, OUTPUT);
  pinMode(enB, OUTPUT);
  pinMode(in1, OUTPUT);
  pinMode(in2, OUTPUT);
  pinMode(in3, OUTPUT);
  pinMode(in4, OUTPUT);

  // Turn off motors - Initial state
  digitalWrite(in1, LOW);
  digitalWrite(in2, LOW);
  digitalWrite(in3, LOW);
  digitalWrite(in4, LOW);
  Serial.begin(9600);
  Serial.println("Serial started at 9600");

  BTSerial.begin(9600);
  Serial.println("BTserial started at 9600");
}
```

```
void loop() {

   if (Serial.available() > 0) {    // is a character available?
      command = Serial.read();
      Serial.print("You typed: ");
      Serial.println(command);
      switch(command){
      case 'w':
        forward();
        break;
      case 's':
         backward();
        break;
      case 'a':
        Left();
        break;
      case 'd':
        Right();
        break;
      case '0':
       Stop();
       break;
        default:
          Stop();
      // code to be executed if command is not a,w,s,d, or 0
      break;
   }
   }


}
```

Figure 32 Code of the controlling program, on the left the Setup part and on the right the loop section

In the 'void setup' it starts the communication with the Bluetooth module at a baud rate of 9600, as well as initially setting the pins that connect with the motor driver as outputs. It also setups the motors initial state to be turn off. Later in the 'void loop' it checks for any character available from the Bluetooth module and changes the value of the variable 'command' according to it. Depending on the character input, it will change the motion state according to it and run it in a loop until a different character that will change the value of 'command' and hence changing the state of motion.

```
void forward() {
  analogWrite(enA, 230);
  analogWrite(enB, 230);

  digitalWrite(in1, HIGH);
  digitalWrite(in2, LOW);
  digitalWrite(in3, HIGH);
  digitalWrite(in4, LOW);
}

void backward() {
  analogWrite(enA, 230);
  analogWrite(enB, 230);

  digitalWrite(in1, LOW);
  digitalWrite(in2, HIGH);
  digitalWrite(in3, LOW);
  digitalWrite(in4, HIGH);
}

void Left() {
  analogWrite(enA, 230);
  analogWrite(enB, 230);

  digitalWrite(in1, LOW);
  digitalWrite(in2, HIGH);
  digitalWrite(in3, HIGH);
  digitalWrite(in4, LOW);
}
```

```
void Right() {
  analogWrite(enA, 230);
  analogWrite(enB, 230);

  digitalWrite(in1, HIGH);
  digitalWrite(in2, LOW);
  digitalWrite(in3, LOW);
  digitalWrite(in4, HIGH);
}

void Stop() {
  digitalWrite(in1, LOW);
  digitalWrite(in2, LOW);
  digitalWrite(in3, LOW);
  digitalWrite(in4, LOW);
}
```

Figure 33 Part of the program where each state of the motion parameters is set

Lastly in the program, the parameters for each of the states of motion is set. The pins EN1 IN1, and IN 2 correspond for the motor A and the pins EN2, IN3 and IN4 correspond for the motor B. The EN pins control the speed of the motors by changing PWM at which the motor is enabled, this value can go from 0 being at stop to 255 being the maximum speed possible. In this case they were set at 230, meaning they were set at around 90% of the speed. The IN pins determine the direction in which it rotates. As seen on Figure 29, by the different combination of motion of both motors, different motion states can be achieved and according to it, it was the set the parameters for each one of the state of motions, including the STOP one which stops both motors from rotating.

3.3 Stability of the robot

In the part 2.4 it was shown how the center of mass and the angle θ at which it is an any moment with respect to the horizontal surface is directly related to the angular acceleration of the robot. After being in a state of motion being it rotation or rolling, due to the inertia the robot will likely continue its motion until finally it comes to a stop. Here is where this angular acceleration due to the center of mass plays an important role. The center of mass acts similar to a 'Proportional' controller, having an angle 0 as the objective, the greater the angle from it (equivalent of greater the error), the greater this angular acceleration is. At a moment in which the motors are in stop and the sphere shell is in motion, the mass will act in the opposite direction, bringing the robot to a stop.

3.3.1 PID Controller implementation

It is possible to just rely on the low center of mass to act as the stabilizer of the robot and bring it to a stop, however it is expected to have a lot of oscillation since it would act just like a pendulum. It is because of that it has be looked at other options to stabilize the robot an have a more smooth movement that could be implemented after testing. The most logic and common option would be the implementation of PID controller. PID stands for proportional integral derivative. In PID, error value is calculated as the difference between a reference point and the actual output. The error is minimized by adjusting the process control inputs. The PID algorithm involves three constant parameters. The proportional, the integral and derivative, denoted by P, I, and D. As mentioned in [7] paper, functions P,I and D can be described as:

‘P’ factor:

1. Based on current rate of change, depends on present
2. The product of gain and measured error.
3. large gain has fast response time and small steady state error
4. Causes overshoots.

‘I’ factor:

1. Based on current rate of change, depends on the history of errors.
2. Eliminates the steady state error.
3. Product of gain and summation of past errors.
4. Causes overshoots

‘D’ factor:

1. Based on current rate of change depends on the prediction of future errors.
2. Product of gain and rate of change of error.
3. Reduces the overshoots caused by proportional and integral factor.
4. Increases noise.
5. Too high gain will cause the system unstable.

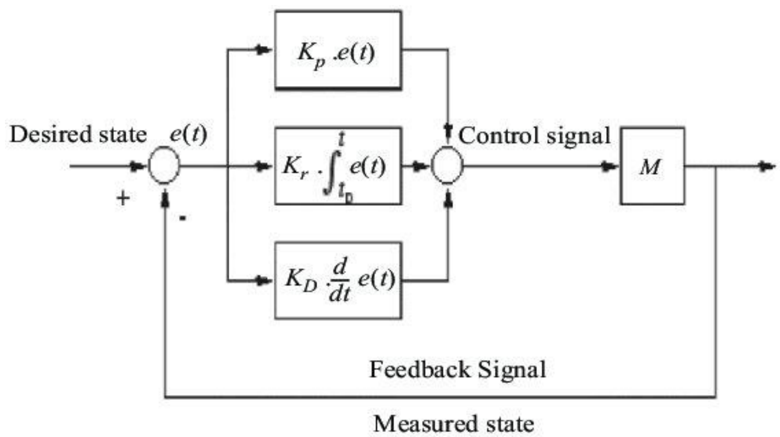


Figure 34 Loop in a PID controller

As seen on the diagram of the feedback loop, it would be needed a feedback signal into the system to determine the error and actuate according to it. In the case of the sphere robot, the error is the angle θ, which is the difference from the desired angle at rest and the actual angle at a given moment. For this is needed the implementation of a sensor capable of measuring such angle.

One of the most common sensors for this task is the MPU 6050. The MPU-6050 is a 6-axis MEMS (Microelectromechanical Systems) device, containing a 3-axis accelerometer and a 3-axis gyroscope. It can measure acceleration along X, Y, and Z axis as well as rotational motion about the X, Y, and Z axis. The device is built on top of a single chip that houses both the accelerometer and gyroscope, and includes an onboard Digital Motion Processor (DMP) to process and combine the data from the two sensors.

The programing of a code to extract the data of the sensor is not straightforward task, however it is possible to use already built libraries and just extract the data needed, which will depend on the mounting of the sensor to determine the angle from which axis is required.

In Figure 35 is an example of the program of the PID controller, where it can be seen each of the controllers' gains. As seen on the code, the proportional component is the direct product of the error and the proportional scalar. Hence, it is directly proportional to the error. The integral component, as its name suggest, keeps adding up itself in time, so stores the errors in the past and adds them up. In the example code, it has been set a limit as this component may wind up in the case of being out of the target for too long. And finally, the derivative component predicts the next error by comparing the previous error and the actual one, multiplying it by its scalar. Each of the components would require a necessary tuning and in this code the serial prints live data of each of them, helping to visualize each of them and the total output of the PID controller, which in this case would be the motor speed. Also is important to mention that the tuning of the parameters of the PID controller in the case of the sphere robot will vary depending on the motor to be used, the gear ratio, the wheel diameter, and even the battery charge would affect it.

```
void PID(){

 Serial.print("Error in angle: ");
 Serial.print(errorY);
 Serial.print("\t speedmotor: ");
//  Serial.println(motorspeed);
  errorY = pitchGyroAngle-goalY;
  if (errorY>= -2 && errorY <= 2) {
    // Set value to 0
    errorY = 0;
  }

  // calculate the proportional component (current error * p scalar)
  float p_scalar = 10;
  if(p_scalar < 0) p_scalar = 0;
  float proportional = errorY * p_scalar;

  // calculate the integral component (summation of past errors * i scalar)
  float i_scalar = 0.5;
  // if(i_scalar < 0) i_scalar = 0;
  static float integral = 0;
  integral += errorY * i_scalar;
  if(integral >  255) integral = 255; // limit wind-up
  if(integral < -255) integral = -255;

  // calculate the derivative component (change since previous error * d scalar)
  static float previous_error = 0;
  float d_scalar = 10;
  if(d_scalar < 0) d_scalar = 0;
  float derivative = (errorY - previous_error) * d_scalar;
  previous_error = errorY;
  int motorspeed=proportional + integral + derivative;
  if(motorspeed >  255) motorspeed  = 255; // limit wind-up
  if(motorspeed < -255) motorspeed = -255;
  Serial.print(motorspeed);
  Serial.print("\t Proportional: ");
  Serial.print(proportional);
  Serial.print("\t derivative: ");
  Serial.print(derivative);
  Serial.print("\t integral: ");
  Serial.println(integral);
   rotateMotor(motorspeed, motorspeed);
}
```

Figure 35 Programming of the PID controller

This code was implemented in the Arduino for having a little of the visualization of how each of the components is affected by the error in angle over time, also to visualize the final output. It is important to highlight that in this case, the actual motors where not in place, and the robot was not inside the sphere, so no actual action is being made, but helps visualize that the program is working properly.

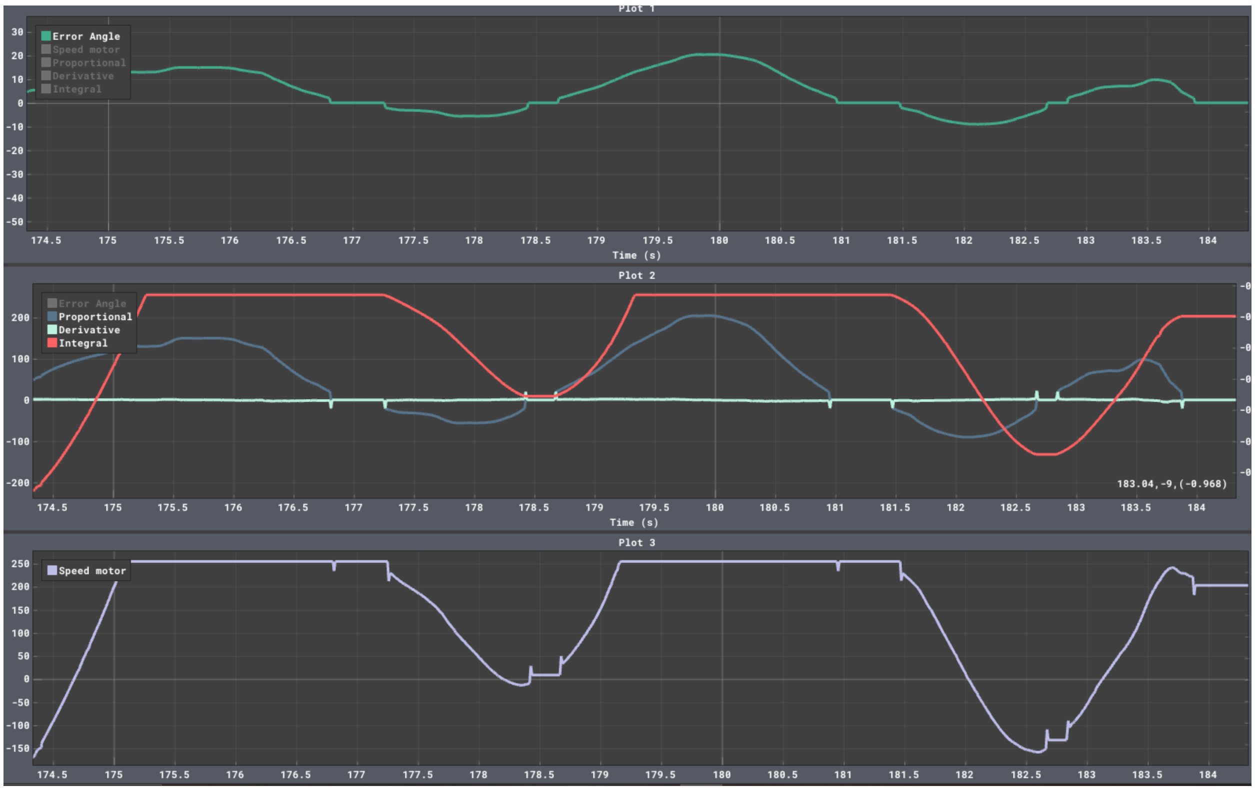


Figure 36 Visual representation of how the PID components change over time along the change in angle and the total outcome (Speed of motor)

4. Building and testing

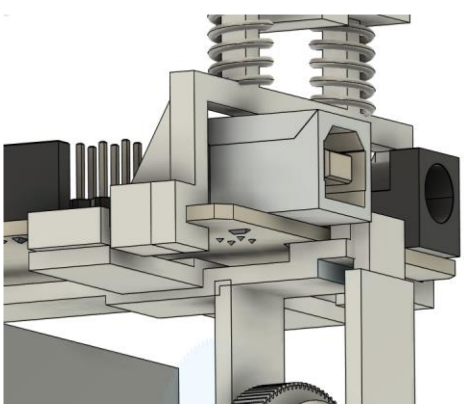
After all the process of designing of both the hardware and software part of robot, it comes the building of it and testing. It is in this part where it will be seen if the needed are changes in terms of the structural design or the programing part of it. Also given that the structure is composed of many parts, it will be seen how they interact with each other, as well as seen how it works as a whole.

4.1 Physical structure

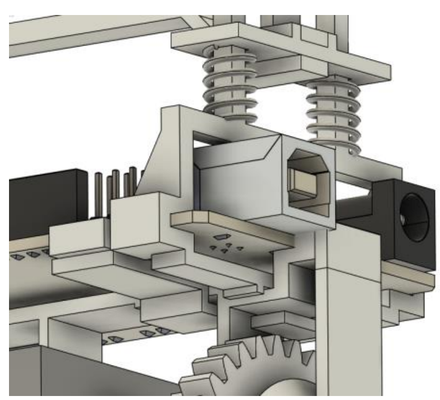
As it is known, the physical structure is composed of many 3D printed parts, of whose accuracy depends largely on the 3D printing process and parameters used in it. Also, the material used and other external factors like the ambient temperature affect this, consequently, it is not until start printing that it can be known for certain. After the first parts were printed with the PLA material, it was seen that the accuracy of the dimensions was of around ± 0.2 mm. Then it was when the first modifications to the parts that had to fit with each other came, to adapt to this margin of error.

Once all the part were ready for the physical construction is that it could be tested the physical stability of the structure in different situations. The initial design aimed to use the implementation of the springs not only as a suspension system, but also to keep together the upper and lower part of the structure, meaning the parts that are on top and below of the programmable microcontroller. This worked only to certain extend. The main issue being that to be able to stand the tangential forces acting on both ends of the structure, bigger perpendicular forces needed to be applied, or in other words, stronger springs would be needed. Even though that would work as a solution, the increase in this loads would also meaning higher friction in the rotating parts, such as the shaft with the wheels and the wheels with the inside surface of the shell. Consequently, the approach taken was to modify this parts to not depend on the perpendicular load to keep them in position. In the following figures is possible to appreciate these changes.

Before



After



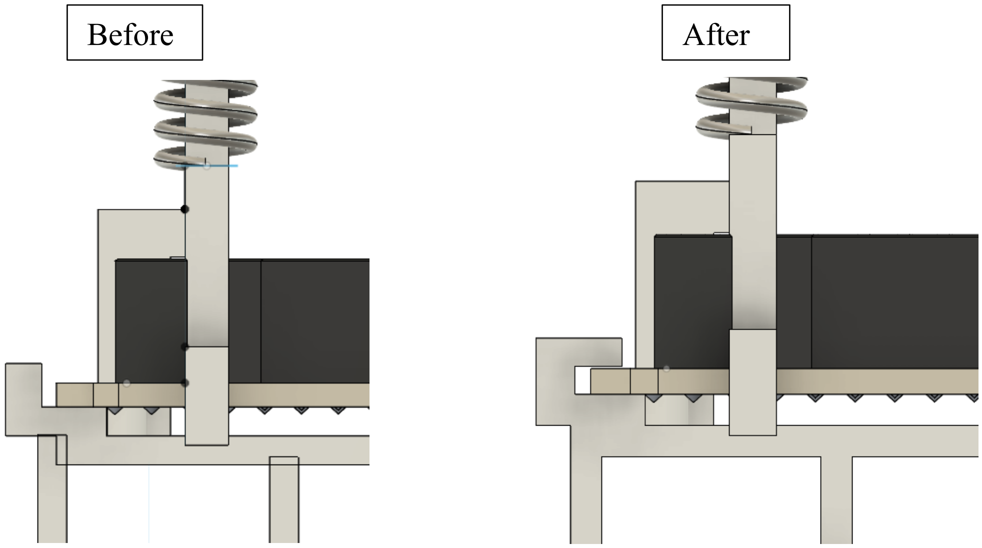


Figure 37 Before and after modifications were made to help keep the structure together

A similar issue was encounter with the platform that holds the batteries support, the DC motors and the lower part of the structure together, in which it was dependent on the weight of the battery to hold in position, which was not enough to withstand the reactive force from the torque of the DC motor. Also, another modification was made to ensure this part could be lock in place, shown in the following pictures.

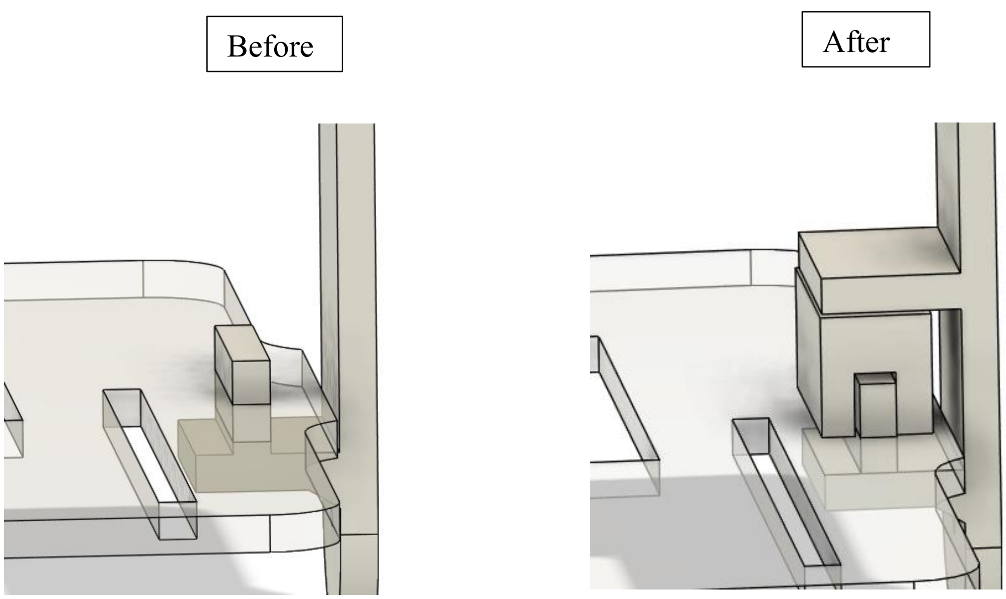


Figure 38 Before and after modifications to ensure the holding platform stays in place

It was not only this issues that were faced after building the structure, it was also notice a small defect in the design of the support wheels. The structure that holds the shaft in which these wheels are mounted were too close to the inside surface of the shell, and at moments when the wheel would not be centered, it would make contact with said surface. For ensuring this does not happen, it was angle at which it is mounted by 45 degrees, so the wheels are in a perpendicular position with the surface. As a result, either at turning or rolling, it will be at the same angle with respect to the motion.

Figure 39 Before and after change of the angle of support wheels

4.2 Testing of the mobility

For testing the mobility functions, it is necessary to set the propulsion system parameters to test, meaning a set pair of DC motors, and a set transfer number for the gears between the DC motor and the wheels. Depending on the availability on the market and the needs for the robot, it was decided to use in this case 5V/0.1A DC motor with incorporated gearbox, giving an output rotation speed of 110rpm. Given the low speed of the motor, it is not necessary to increase the speed or torque with the transfer between the gears, so it is used a ratio of 1:1 for this testing.

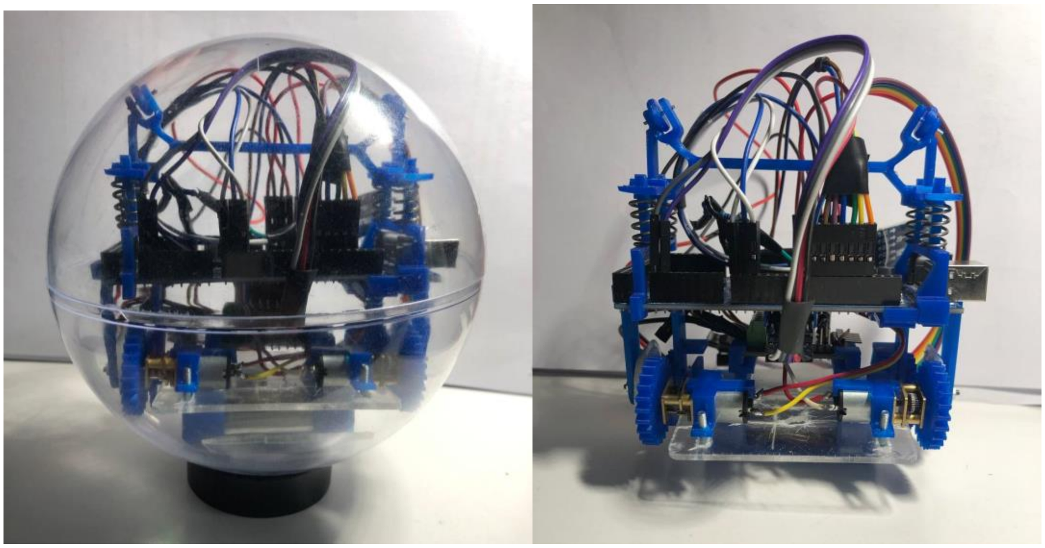


Figure 40 Roball Robot after being build, with the shell (Left) and without it (Right)

After successfully programming, compiling, and uploading of the control system into the Arduino with the simple motion controls it was possible to test the mobility of the robot. Unfortunately due to the previous mentioned interface of the Arduino application used for the control, it was not possible to retrieve live data while using the controls.

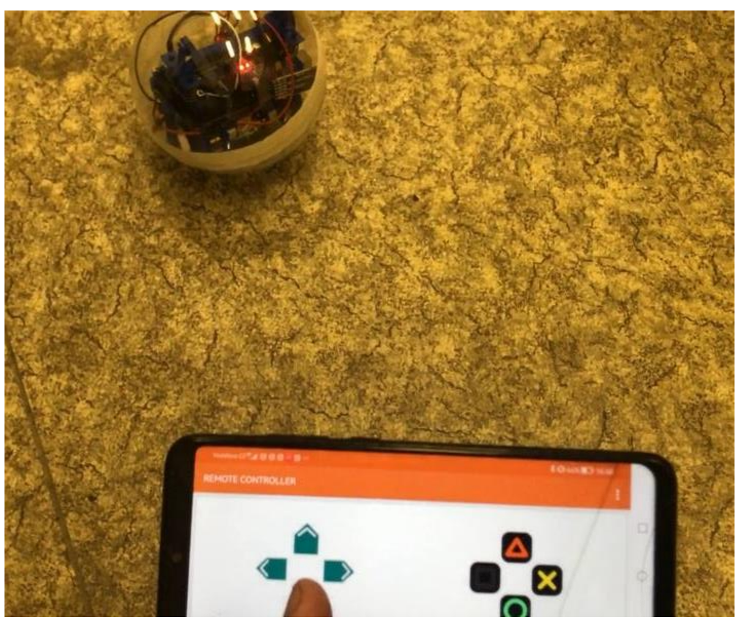


Figure 41 Sphere robot during testing

However, it was possible to note some aspects of it. As expected, the robot is capable of rotating and rolling in any direction, all the states of motions accomplish their functions, however the neither the rolling or rotating are completely smooth, meaning that none of them accomplish an steady state. This was identified to be a result of a not smooth surface transition on the inside surface of the spheric shell between the two semi-spheres that form part of it. Additionally, after the change of state of motion from either forward or backward to STOP, the lower center of mass help the robot stabilize without continuing the rolling but not without the expected oscillation, thus taking some time until it reaches a point of steady state. This stabilization also depend on the surface the robot is performing the robot. In Figure 42 can be seen the result of a small test that was made in two different surfaces, one in a completely solid and very smooth surface (Green) and another in a softer one (Blue). The results of each test was put in a overlapped graph to have a visual comparison of how long it takes to reach a steady state.

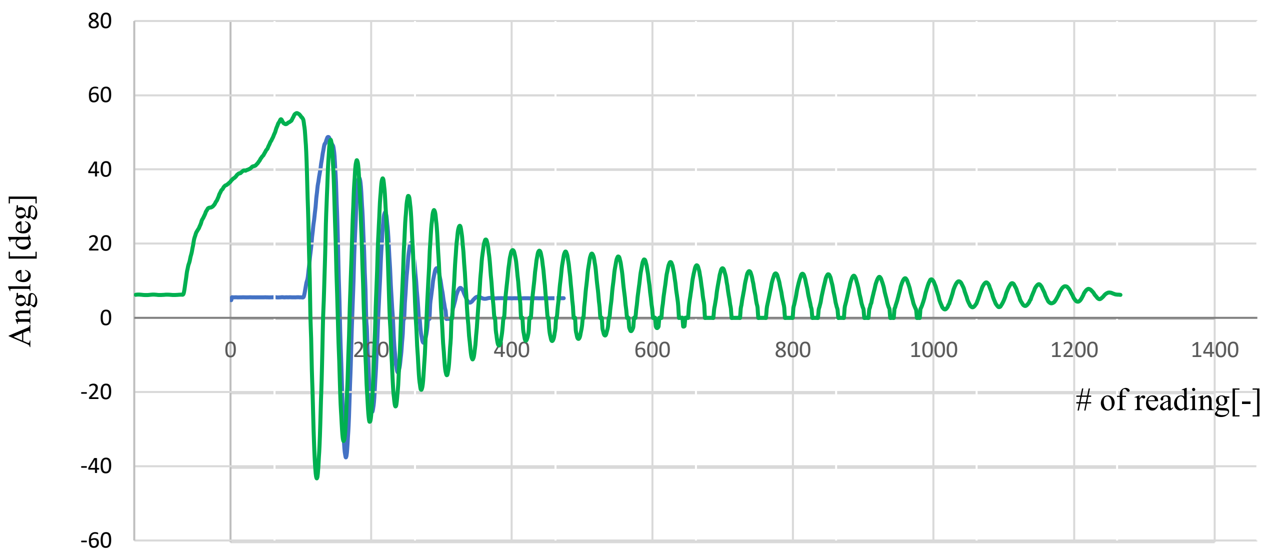


Figure 42 Graph of the angle of the robot at each reading as it reaches a steady state. Green for solid surface and Blue for a softer one. Each reading is made around every 2ms.

5. Conclusion

In summary, an extensive evaluation was conducted on various designs of previously built sphere robots and the concepts utilized for achieving their mobility objectives. These designs ranged from using their mass as a pendulum to wheeled robots inside a sphere. The most optimal and efficient concept was selected as the foundation for the design of a new sphere robot. To achieve this, the design process was divided into three main phases: the design of the physical structure, the design of the control system, and the building, testing, and redesign of components.

During this process, various considerations were taken into account to ensure the successful construction of the robot. The goal was to create a robot capable of moving in all directions and self-stabilizing, while also being adaptable for future improvements.

After building, testing, and redesign, it was determined that each step taken to achieve the main goal was successful. A stable and easily assembled and disassembled 3D printed structure was created, allowing for the change of its components for future improvements if necessary. Additionally, the control system was designed and programmed in a clear and concise manner, with the added capability of implementing a PID controller for future tests. Most importantly, testing revealed that the sphere robot was capable of rolling and rotating in any direction desired by the user, controlled through a Bluetooth application. The self-stabilization design using a lowered center of mass was also proven to be effective.

However, this initial design has significant room for improvement. The movement when either rotating or rolling is not smooth, and the self-stabilization is not the most efficient method for stopping the robot's motion. In future designs, improvements can be made by utilizing a shell with two semi-spheres of more precise shape and a smoother inside surface, which would greatly enhance the robot's movement. Additionally, the implementation of a PID controller with an appropriate propulsion system and proper tuning would improve both movement and stabilization but would require further testing and research.

References

[1] Y. L. a. M. I. a. K. A. a. P. E. Karavaev, "Spherical rolling robots: Different designs and control algorithms," in *Robots in Human Life: Proc. of the 23rd Internat. Conf. on Climbing and Walking Robots and the Support Technologies for Mobile Machines*, Moscow, Aug 24-26 2020.

[2] V. A. J. a. R. N. B. a. R. Hippalgaonkar, "Design and analysis of a spherical mobile robot," *Mechanism and Machine Theory,* vol. 45, pp. 130-136, 2010.

[3] J. a. Y. T. Suomela, "Ball-Shaped Robots: An Historical Overview and Recent Developments at TKK," *Springer Tracts in Advanced Robotics,* vol. 25, pp. 343-354, 01 2005.

[4] F. a. C. S. Michaud, "Roball, the Rolling Robot," *Auton. Robots,* vol. 12, pp. 211-222, April 2002.

[5] A. Bicchi, A. Balluchi, D. Prattichizzo and A. Gorelli, "Introducing the "SPHERICLE": an experimental testbed for research and teaching in nonholonomy," in *Proceedings of International Conference on Robotics and Automation*, New Mexico, USA, 1997.

[6] A. a. S. T. a. W. Y. Halme, "Motion control of a spherical mobile robot," *International Workshop on Advanced Motion Control, AMC,* vol. 1, pp. 259-264, 04 1996.

[7] O. Jamil, M. Jamil, Y. Ayaz and K. Ahmad, "Modeling, control of a two-wheeled self-balancing robot," in *2014 International Conference on Robotics and Emerging Allied Technologies in Engineering (iCREATE)*, 2014.

[8] V. A. Crossley, "A Literature Review on the Design of Spherical Rolling Robots," Pittsburgh, 2006.

[9] S. a. A. S. Bhattacharya, "Design, experiments and motion planning of a spherical rolling robot," in *Proceedings 2000 ICRA. Millennium Conference. IEEE International Conference on Robotics and Automation. Symposia Proceedings (Cat. No.00CH37065)*, San Francisco, CA, USA, 2000.

[10] A. V. Borisov, A. A. Kilin and a. I. S. Mamaev, "How to Control Chaplygin's Sphere Using Rotors," *Regular and Chaotic Dynamics,* vol. 17, pp. 258-272, May 2012.

[11] J. a. R.-R. I. a. C. A. a. M. M. J. a. M.-E. A. a. D. J. M. Rodríguez Corral, "A Study on the Suitability of Visual Languages for Non-Expert Robot Programmers," *IEEE Access,* vol. 7, pp. 17535-17550, January 2019.

[12] R. Rogers, "Building a Benchtop PID Controller," Cleveland, 2013.