Term 3 Interim Project Report
B.A.T

Course: MEng Robotics and Artificial Intelligence
Team No.: 4
Team Members:

Rehan Agrawal	23025308
Aditya Bishnoi	23065221
Dami Ogunleye	23013102
Luoyu Zhang	23019184

Date of submission: 15/05/2024

Abstract

The BAT, which stands for the Bio-mechanical Aerial Tree-Climber, is a robot with the task of sensor deployment. This report details the process from start to finish, including the ideation and proof of concept stage, the iterative development cycle and the final product. The ideation stage of the report evaluates existing tree-climbing robot solutions and establishes the design concept. The report subsequently discusses various design ideas and documents the steps taken to manufacture and assemble the final project.

Contents

Introduction

The observation and conservation of wildlife are critical tasks that benefit greatly from advancements in robotic technology. Tree-climbing robots, in particular, can revolutionize the way we study animals to a certain extent. In addition to their ecological applications, tree-climbing robots are also crucial in the management and harvesting of trees, particularly in urban settings and agriculture. Their use in these areas increases efficiency and safety, making them indispensable tools for urban greenery maintenance and agricultural production.

With tree-climbing robots, animals can be observed non-invasively, facilitating the study of natural behaviors. These robots work uninterruptedly and independently, minimizing the need for human involvement and labor. These robots also contribute to the accessibility of useful tools for observing and protecting threatened species. Robots facilitate information flow, thus making data collection and analysis more efficient. This technology helps in designing better conservation strategies and allows knowing better the dynamics and patterns in the ecosystems.

Trees in the urban landscape should be maintained to have their health retained, thus serving aesthetic purposes. This has always demanded arborists to carry out perilous, high-level work that leads to the use of ladders and harnesses most of the time. Now, the advancement in this area with the help of tree-climbing robots provides a new alternative, much safer and demanding minimum human involvement.

These robots could be used by municipalities to decrease the labor cost of tree maintenance and even ensure workers' safety while doing such work. Another instance can be said when fruits such as coconuts are dependent on heavy skilled labor in an agricultural setting, posing risks for manually harvesting and being a physically demanding job. "Thus, robotic tree climbing has the potential to revolutionize this process by automating the harvest and increasing either safety and/or yield."

Robotic harvesters would give accuracy and efficiency, thus allowing fruits to be collected much faster and in a more uniform manner than could ever be realized with human labor, hence reducing time and labor costs and lessening the exposure of human beings to unsafe conditions. Besides, these robots have the best class advanced sensors and control systems that will enable them to perform tasks such as tree inspection, health monitoring down to the leaves and bark level. They can also do imaging and sampling tasks that are important for disease detection and growth assessment, respectively.

This capacity is useful not only for identifying any problems in time and providing them with proper treatment but also for the management of healthy trees on the whole, which makes the green places firmer and more productive. In addition, the possible combination of tree-climbing robots with data analytics tools might yield tree-growth pattern studies and environmental impact from this technology.

The data collected by these robots will help the researchers and urban

planners take decisions on species selection, how they have to be planted, and the schedule for maintenance which would actually lead to smart and sustainable city development. In conclusion, tree-climbing robots offer revolutionary technology for urban forestry and agriculture. In turn, their capability to execute such tasks as pruning, harvesting, or health monitoring effectively and safely guarantees not only operational efficiency but also the making of great contributions to safety and sustainability of practices. Further development of these technologies is expected to continue revolutionizing how tree populations are managed in an urban and rural set-up, thus ensuring continued provision of vital ecological and social infrastructure.

Building on what has been done in previous research in the field of robotic tree climbers, and motivated by the recent research and development in quadcopter drone technologies, our new robotic tree climber is proposed that uses rotors or propellers. This revolutionary method entails the use of aerial drones with dexterous and precisely controlled mobility adapted to the environmental challenges of arboreal systems. Our design will include propellers that will increase the robot's maneuverability over obstacles, such as branches and foliage, and a more dynamic and effective mechanism of climbing and descending trees.

In addition, the use of rotor-based propulsion allows the robot to better interact with both flat and surface areas, which are protruding from the tree and differently shaped than the cylindrical, which in traditional climbing robots grabbing or fully enclosing. Thus, the propellers allow the robot to exercise not only little pressure on the tree bark for the natural health of the tree but also to adapt very fast to different diameters and surface textures of the trunk.

After reviewing prior works of research in the following section, this paper brings out the technical details of the robot, its operating capabilities, and results obtained during pilot field tests—all of which describe the potential of this novel concept to reinvent the task of tree maintenance and harvest.

Case studies

TreeBot

The first existing solution for our case study analysis is "Treebot", a tree-climbing robot that came out of the Chinese University of Hong Kong. The main concept behind this design is a continuum robot, which means it has a compliant spine which allows it to bend and extend in different directions depending on the loads applied. Its locomotion can be compared to that of an inchworm whereby it does not rely on vision and is simple to process [7]. In essence, the robot alternates between the two grippers on the top and bottom of the robot. It extends and grips onto the surface with the top one and then it pulls up the bottom one and uses that to grip into the tree as well. This process is repeated, allowing the robot to inch its way up the tree. The Treebot can be split into two main components, the gripper and the continuum body. The continuum body consists of three bendable tendons or springs, 3 actuators to control the length of the springs and a rack and pinion driving mechanism. Each tendon is equidistant from the centre and is separated by an angle of 120 degrees [7]. The gripper uses 4 claws which are extended by a linear motor (pushing plate). The pushing plate, combined with some springs enables the grippers to get enough force to hold onto the tree. In addition, the ends of the grippers have surgical needles to create an adhesive force.

Implementing this concept would be suitable for the task we have been given due to its adaptability and flexibility. These attributes would be especially useful for us to avoid the branches and adapt to the curvature of the tree efficiently. On the other hand, while the researchers describe the design as lightweight, it is still too heavy given the task constraints. Additionally, the surgical needles that enable the robot to grip the tree cannot be implemented since that is invasive and could potentially damage the tree. Even though we will create a design that differs significantly from the Treebot, we aim to create a robot that will have a similar amount of adaptability and flexibility that this solution can encapsulate, while also keeping it lightweight.

Adhesive Adsorption

Many roboticists have looked to animals with expert climbing abilities as sources of inspiration for climbing robots. Animals such as geckos, cockroaches, spiders and more have been used in various designs. We will specifically analyse the Waalbot by Murphy and the Stickybot by Stanford which the special ability of the gecko has inspired. The gecko's unique climbing abilities come from the array of microscopic hairs on its feet called setae. According to an article from the University of Notre Dame (2021) [8], "when the gecko places its foot on a surface, these hairs cling to the surface and form intermolecular bonds, called Van der Waals bonds, with the molecules of that surface". Researchers behind the two mentioned bionic robots have tried to replicate gecko feet as a method of adhesion. The Waalbot is a lightweight robot that uses rotating footpads that

imitate the setae on gecko feet along with an autonomous adhesive recovery system to make its design more robust [3]. Similarly, the Stickybot uses a similar dry adhesive and even its locomotion is inspired by the gecko's gait [6].

While the way both robots move differs significantly, they are both effective at climbing smooth or nearly smooth surfaces due to the synthetic adhesion materials that the researchers have designed. While the gecko's feet are adaptable to rough and smooth surfaces, synthetic feet are only suitable on smooth surfaces. Furthermore, they are limited by a weak load capacity and slow movement speed [3]. However, with this method the advantages include no energy consumption when adhering to the surface and no noise. Overall, it would not be suitable for us to utilise this type of dry adhesion in our design for a few reasons. Firstly, the tree bark our robot would climb is rough, and the pads would struggle to adhere to it. Secondly, to try and source or produce our own microfiber pads is not feasible, given the constraints on our task. Lastly, our robot is purposed with sensor deployment and if this form of adhesion is not suitable for high loads, it would likely struggle with our mission. Even though our robot cannot incorporate dry adhesion like a gecko, looking to nature for inspiration is still useful for our design process.

Gecko and Cicada-inspired Design

In a similar vein, Bian et al created a gecko-inspired robot which mimicked the gait of a gecko but utilises an adhesion design inspired by both geckos and cicadas [2]. For smooth surfaces, they used a similar bionic material to the Waalbot and Stickybot. However, for rough surfaces, they designed a palm which incorporated design elements from the cicada. The palm had rows of bio-inspired hooks attached to them and were 3D printed using resin [2]. As opposed to other robots which penetrate the surface to climb, this innovative design uses small hooks to find leverage points on the surface. These hooks would exploit the nooks and crannies of a surface and friction would do the rest. This allows the robot to climb the surface in a non-invasive way. Therefore, this design could be implemented for our project because it prevents damaging the tree, a necessary criterion for our robot's success.

However, there are limitations to employing this kind of design because all tree bark is not equally rough. While most of a tree may have a surface that the palm could hook into, often in nature some parts are smoother than others. This robot is adaptable in a static nature such that you could either implement the gecko or the cicada-style palm, depending on the nature of the surface, but not during a climb or dynamically. This means its adaptability is flawed and could easily struggle in a real-life climb. While the researchers claimed relatively fast climbing speeds of 9.34 cm/s on glass and 9.14cm/s on stone [2], we veered away from this concept design due to the difficulty in controlling and synchronising the steps, with our current experience, without it moving at an incredibly slow speed.

VertiGo

VertiGo is not a bio-inspired robot but is equally effective in climbing walls. It is a robot project created in a collaboration between Disney Research Zurich and ETH. The robot can also drive on the ground and transition to climbing the wall. It can do this by utilising tiltable propellers that push the robot upwards and onto the walls, in addition to the four wheels helping it drive. Steering is possible due to the front pair of wheels being steerable and "each propeller has two degrees of freedom for adjusting the direction of thrust" [1].

The researchers minimised the robot's weight through numerous aspects such as the carbon fibre baseplate, and 3D-printed parts. These 3D-printed parts were used for structures like the wheel suspensions and the wheels themselves. The baseplate also acted as a point to mount the propellers and a carrier for all the electronic components. It uses two integrated servomotors to move in the inner and outer rings independently from each other, allowing it to drive on the floor, walls and theoretically on the ceiling. The full robot has 8 individually controlled actuators and is driven by a human operator in a similar way to RC cars. Furthermore, it also has an MCU, a 6-axis IMU and two infrared sensors to estimate its orientation in space [1]. We can integrate aspects of this solution into our design, such as the genius use of propellers as an extra thrust. This would be useful for our project since we must be non-invasive in our tree climb. While we may not incorporate such a complex implementation with the dual-axis propellers, due to challenges with the control, the concept could be useful in our climb.

Mechanical Design

Mechanism

General Dynamics

As shown in figure 1, the robot consists of three interconnected rectangular frames. The frames are designed to be rigid yet lightweight, ensuring the robot can easily maneuver around the tree. Four propellers are placed at each corner of the robot, providing thrust and stability. Each propeller provides a thrust (T1, T2, T3, and T4), with torque (M1, M2, M3, and M4) indicated in the schematic. The propellers work together to allow the robot to hover, ascend, descend, and yaw. There is a brush suspension system on each of the frames, which ensures stable contact with the tree surface, allowing the robot to maintain its position and adjust to varying tree diameters. Each of the three frames also houses an ultrasonic sensor to provide environmental data, especially for branch detection.

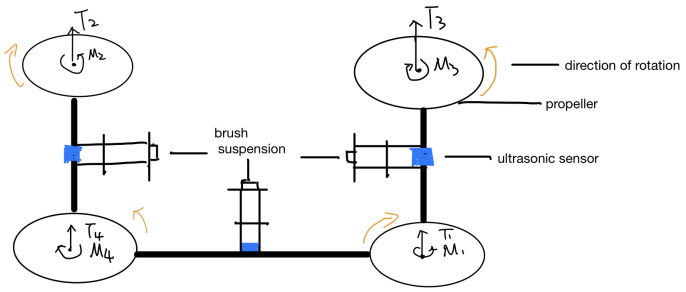


Figure 1: Schematics Diagram

From Newton-Euler equations, the motion of a rigid body can be seen as the combination of the translation motion of its center and rotational motion around it. The translation motion is described using Newton's second law: $F = m\frac{dv}{dt} = ma$, where F is the net force applied and v is the velocity. In the case of our robot,

$$T_1 + T_2 + T_3 + T_4 - W = ma$$

For rotational motion, the formula is:

$$M = J\dot{\omega} + \omega \times (J\omega)$$

Here, M represents the net torque applied, J is called the inertia matrix, and w is the angular velocity. The formula states that a net torque of M on the body causes it to rotation with angular velocity w. For our robot,

$$M = M_1 + M_2 - M_3 - M_4$$

By making M positive or negative, the direction of rotation is changed, and the magnitude of M determines the angular speed of rotation.

Propellers

Thrust T and torque M mentioned earlier can be explained with the mechanism of propellers.

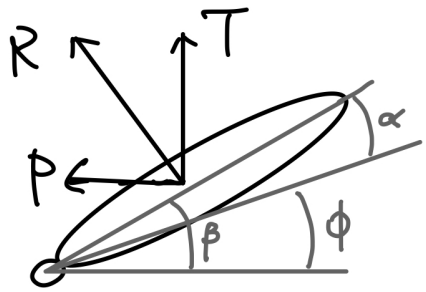


Figure 2

The angles labelled in figure 2 are:

- ϕ — airflow angle, the angle between the airflow and the plane of propeller rotation
- α — angle of attack of the blade section
- β — blade angle, the angle between the chord line of the blade section and the plane of rotation

It is evident that $\alpha + \phi = \beta$. As air flows over each segment of the blade, aerodynamic forces are generated, including drag ΔD and lift ΔL, which combine to form the total aerodynamic force ΔR. The component of ΔR along the direction of flight is the thrust ΔT, while the force ΔP, acting opposite to the direction of propeller rotation, resists the rotation of the propeller. Summing the thrust and the resistance from all segments of the blade yields the overall thrust T and torque M resisting the rotation of the propeller. To make the robot hover, there should be no overall torque, $M = 0$. This is why propeller 1 and 2 are in the opposite direction of 3 and 4, so that the torques act oppositely and counteracts each other.

Ultrasonic Sensor

To detect branches, ultrasonic sensors are used. To initiate distance measurement, a 10us TTL pulse is transmitted. Following this pulse, the sensor proceeds to measure distance, with the results conveyed via an echo signal. The duration

t of the echo signal's TTL level indicates the total time taken for the ultrasound waves to travel from the HC-SR04 component, encounter an obstacle, and return to the component. The distance is calculated as:

$$d = t \times c_s/2$$

where c_s is the speed of sound.

Stages of motion

- Hovering
  As previous mentioned, for our robot to hover stably, the total thrust gen-

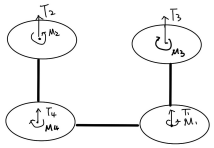


Figure 3

erated by the four motors must exactly counterbalance the gravitational force. This condition can be described by the equation:

$$T_1 + T_2 + T_3 + T_4 = W$$

Additionally, to ensure there is no rotational motion about the center of mass in the horizontal plane, the moments generated by the motors must be balanced:

$$M_1 + M_2 = M_3 + M_4$$

- Ascending or Descending
  When the robot needs to ascend or descend, the total thrust must be

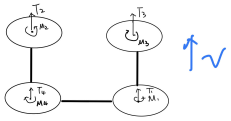


Figure 4

either greater or less than the weight, respectively. This dynamic can be captured by the equation:

$$T_1 + T_2 + T_3 + T_4 - W = ma$$

If $a > 0$, the robot is ascending; if $a < 0$, it is descending.

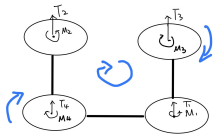


Figure 5

- Rotation (Yaw)
  To change the heading of the robot without moving sideways, a differential thrust setup is used. This can be achieved by adjusting the thrust of the motors to create a net torque about the vertical axis:

$$\tau = (M_1 + M_2) - (M_3 + M_4)$$

  A positive τ results in clockwise rotation, and a negative τ results in counterclockwise rotation. If obstacles are detected, the control system may adjust the motor speeds such that:

$$M_1, M_2 < M_3, M_4 \quad \text{or} \quad M_1, M_2 > M_3, M_4$$

  depending on the desired direction of rotation.

CAD

First Iteration

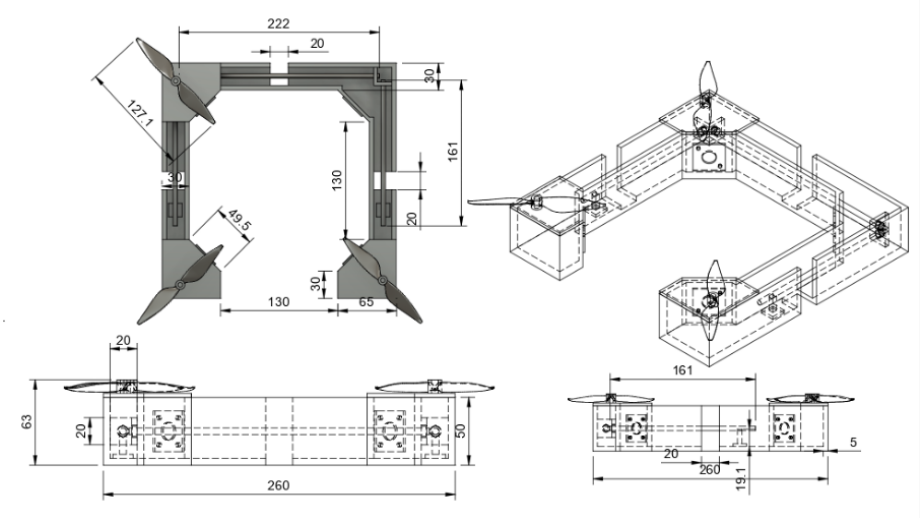


Figure 6: A CAD Model of the First Iteration of Our Design

Figure.6 presents schematic diagrams of the first iteration of our design. It includes a square frame with a length of 26 mm and a height of 50 mm. Please note that the design does include a fourth propeller on the remaining corner since it has been hidden for ease of seeing all the interior components. This horseshoe-style design is 30 mm thick and has walls which are 5 mm thick. As shown in the diagrams the inside of the frame is hollow to allow space for the mechanical and electrical components. Each propeller has a diameter of 127.1 mm. We chose to go for two-blade propellers as opposed to three-blade propellers because they are generally more efficient. Each propeller is mounted on a plate where the motors and the frames can be inserted into. In this design, the suspension discussed below would be attached to the four inner corners.

To move up the tree and detect branches we have proposed to mount three ultrasonic sensors on the three sections of the frame. These sensors would be connected to the rods that go through the robot's frame at the centre points, where the 20 mm gaps are in the frame. For our robot to traverse down the tree we must be able to detect where obstacles may be. Instead of fitting another three sensors on the bottom of the robot because of the unnecessary weight, we have created a system that would allow us to rotate the ultrasonic sensors from the top to the bottom of the robot. A motor would drive the rod on the left side of the robot. This rod is connected to the other rods throughout the robot via bevel gears, found at the top two corners (See fig.6). The rods are supported by holders located by the centre gaps and a bevel gearbox at the corners. As

the driving rod rotates, it would cause the other rods to rotate, allowing the robot to switch the motors from the top to the bottom with a single motor. This decision was made due to weight considerations and ease of programming.

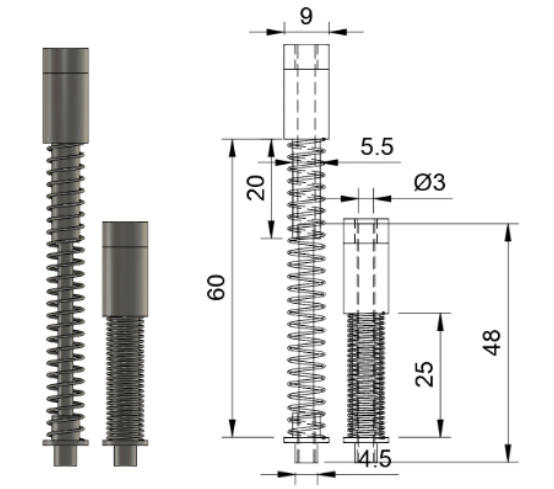


Figure 7: CAD Model of our Suspension

The above figure presents an RC car suspension that we modelled on Fusion 360. We have presented the spring in its max extension (60 mm) and max compression states (25 mm). In the lab, we had access to 30 mm springs as opposed to 60 mm springs so in the manufacturing of the suspension we chose to assemble two 30 mm springs in series. The design works whereby there is a centre rod fixed to the bottom cap which moves linearly through the top cap. The springs work to absorb the shocks and adapt to changes in tree diameter. To manufacture this suspension, we 3D printed the top and bottom caps and cut a 3 mm diameter metal dowel to the right length (48 mm). The springs were glued to the two caps to create a working suspension. The end attachments for such a suspension could be brushes of some sort or a ball caster because that would limit the friction and allow for omnidirectional movement. After testing the suspension, the force required to achieve a compression of 35mm was noted as too substantial. The springs that were available in the lab, had a spring coefficient that was not suitable to be used in our robot ($0.546\,\mathrm{N\,mm^{-1}}$). When using two 30 mm springs in series the force required was 8.19 N as seen below.

$$\frac{1}{K_eff} = \frac{1}{K_1} + \frac{1}{K_2} + \dots \tag{1}$$

Equation.1 states how the effective spring coefficient can be calculated when springs are in series. This equations suggests that when using two identical springs in series, the effective spring coefficient is halved. Therefore the spring

coefficient for our suspension is $0.273\,\mathrm{N\,mm^{-1}}$. Below we have calculated the force required to achieve a compression of 30 mm.

$$0.273\,\mathrm{N\,mm^{-1}} \times 30\,\mathrm{mm} = 8.19\,\mathrm{N}$$

Although this suspension would have been suitable for our robot, we chose not to use it because springs of the right specification were not easily available, and a lighter design could also be implemented. In the latest design, one can see a simpler suspension that we have designed.

Design 2.0

Through a process of iterative development and constant evaluation, we were able to develop our design so that it would be more lightweight while still having a suitable amount of rigidity. The main challenge with this design concept is the trade-off between a lightweight robot and a sturdy robot. Our current design aims to find that right balance and with further tests of the design, we will be able to refine it further.

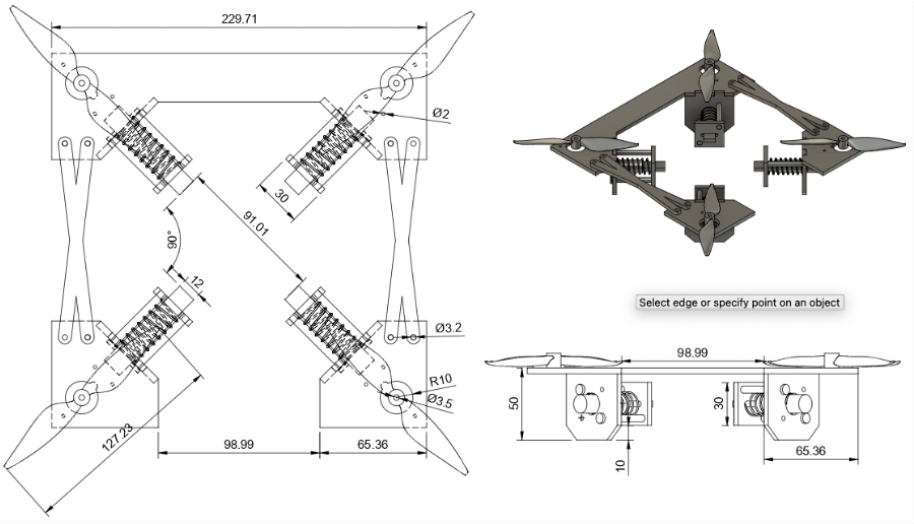


Figure 8: CAD Model of Current Design

Figure.8 presents the schematics for our updated design. The main advantage that this design has over the previous design is that it is significantly lighter. The design is conceptually the same as the last one but unnecessary aspects from the frame were removed to save weight. In addition, this design is smaller due to the change in challenge specifications, allowing us to cut some weight as well. This outer diameter of the frame is now 229.71 mm in contrast to 260 mm. It includes four identical corners where holes have been drilled for

the motor frames to be screwed into. The motor frame utilises M2 bolts, and we have also drilled a larger hole of 20 mm diameter, where the gear attached to the propeller would sit flush in. The gaps between the corner plates are approximately 99 mm. To connect the corner plates to the central frame, our design includes cross-shaped rigid supports that are fixed to the corners with M3 bolts. This design was chosen in an attempt to reduce some of the excess weight. In addition, cross-bracing is also useful in maintaining a stable structure [9], especially concerning earthquakes. Hence, we decided to implement a similar idea with our frame since we are dealing with significant vibration levels from the motors that could damage our frame. Moreover, the main electronic components will be mounted on the central frame, however, with further testing of the weight distribution this may change in future iterations.

As with the first design, we are using plates angled at 45° to hold the suspension mechanism. In their extended state, the distance between opposite brushes is around 91 mm. This means for the minimum diameter of the tree (90 mm) the brushes should contact the tree enough to use it as a guide for traversing up and down the tree while still restricting excess friction. As you can see in figure.8, the ends of the brush go through the middle of the suspension plates and are supported by two 5mm dowels with stoppers at the end. These dowels are also rigidly fixed to the plates at the brush tips. The stoppers help to limit the max extension of the brush and to prevent unnecessary lateral movement during locomotion. In this figure, you can also see the springs which we manufactured ourselves. The springs have a lower spring coefficient than those that we had access to in the lab since they are custom-made. These springs would also be fixed to the brush tip and suspension plates. Moreover, the propellers are mounted above the suspension in a similar orientation to the first design.

Concerning sensing, we have attempted to reduce the robot's weight. In the first iteration, we proposed using three sensors and a sensor-rotating system to detect obstacles. This current design only uses one sensor. To detect branches, the ultrasonic sensor is mounted on the frame opposite the gap and is used to scan the entire circumference of the tree to detect the closest obstacle. Once the obstacle is detected, the robot would rotate 180° to avoid it. This method reduces the mechanical burden on the robot but increases the necessity for a solid control system, but most importantly it helps us to reduce the robot's mass.

Exploded Design 2.0

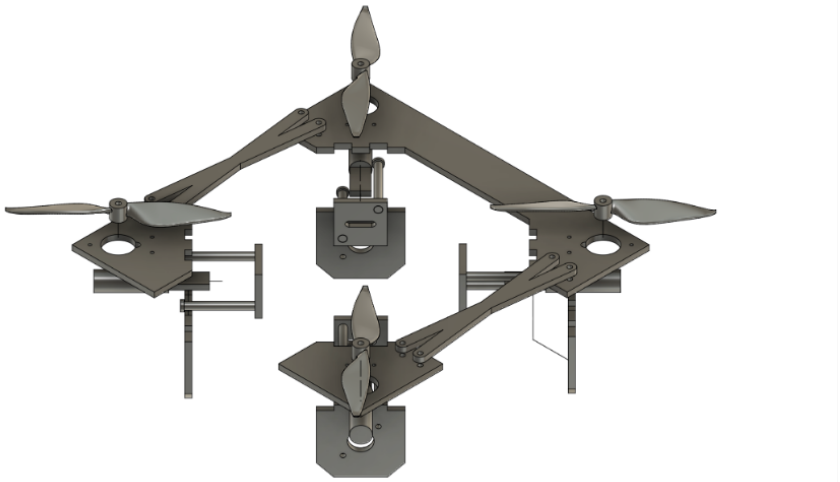


Figure 9: Exploded Model of our Suspension

The above figure depicts an exploded view of our mechanism. In this figure, the frames for the motor have not been included because we are still in the prototyping and testing stage. At this point in the design process, we are utilising components taken from a drone to help us figure out the higher-level elements such as control and weight distribution. Once further testing has been completed, we will use our own propeller models and motor frames. This will be reflected later in this report. Figure.9 also highlights the use of finger joints to connect the corner and suspension plates. This was done to increase the structural integrity of our suspension. Depending on the springs used and the amount of compression, the spring force could break the joint between the two plates if they were not meshed in this way. As our robot is lightweight, we need a strong structure for our frame. Also please note that while the springs are not featured in this exploded design, they are a part of our design and will be seen in the manufacturing stage.

Sensor Deployment

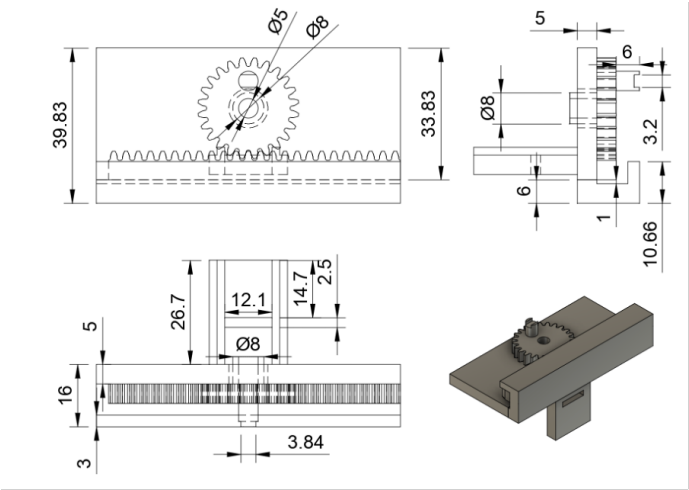
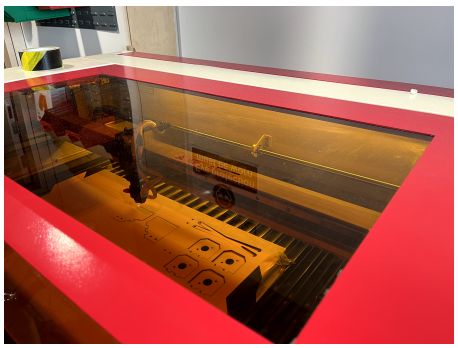


Figure 10: Rack and Pinion Mechanism for Sensor Deployment

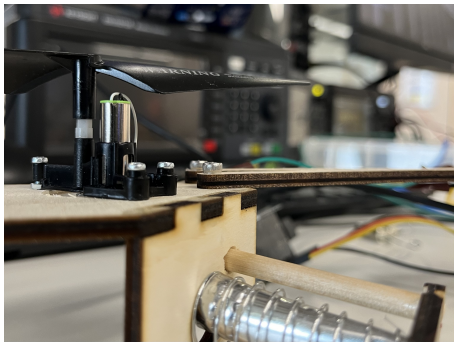
Figure.10 presents the CAD for our sensor deployment system. The concept is that a 360° rotating servo motor will be connected to the driver spur gear at the top from behind the base. After rotating 180° or until the servo experiences resistance and starts drawing more current, the spur gear will push the rack linearly onto the tree. The sensor, which will be attached to the end of the rack loosely, will stick to the metal strip, coming off the rack. The servo motor will wait for a few seconds then pull back the angle it originally turned. After this the servo motor will rotate in the opposite direction 180° to flip the ultrasonic sensor that will be attached to the spur gear. Now the drone is ready to climb down the tree. This component was not manufactured in the end due to concerns that the sensor being deployed could interfere with the ultrasonic sensor. However, our reason behind designing this mechanism was to reduce the need for several servo motors and consequently lower the weight.

Assembly

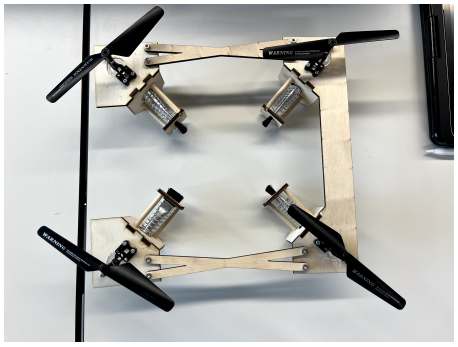
(a) Manufactured Springs



(b) Laser Cutting of Frame



(c) Finger Joints



(d) Assembled Frame

Figure 11: Snapshots of the Assembly Process

Figure.11 presents different snapshots from our design process's assembly and manufacturing stages. Figure.11a illustrates two iterations of the suspension used in our design. The suspension on the left reflects the manufactured suspension found in figure.7 while the one on the right reflects the one found in figure.8 . The RC-inspired suspension utilises 3D-printed top and bottom caps in white and a 3 mm metal dowel that goes through them. The metal dowel is fixed to the bottom cap with super glue and can move freely through the top cap. Around the metal dowel, two 30 mm springs have been attached and fixed to both the top and bottom caps. On the right, we have a simpler suspension model whereby we have fixed a 4 mm thick plywood plate to the tips of paint brushes found in the lab. The other plate found in that figure was later discarded as seen in figure.11c. Moreover, the plate fixed to the brush tip had two 5 mm holes drilled into them for wooden dowels to be fixed into with glue. These wooden dowels would act as supports for the suspension to prevent too much movement. The spring used in that suspension design was

custom-made by wrapping metal wire around a cylinder and was a result of various attempts.

Figure.11b demonstrates how laser cutting was used in the manufacturing of our design. Since our design could easily be separated into different flat plates it was logical to use laser cutting as opposed to 3D printing for manufacturing the frame. Furthermore, since this design is a prototype, optimising in a way that could be done with 3D printing was not necessary. Also, as laser cutting is a much faster process it was an efficient and precise method for us to manufacture the different frame components. To achieve the flush fit found in the finger joints (see figure.11c) we applied a kerf offset setting of 0.15 mm. This technique was done to counteract the inaccuracies that occur as the laser cutter does not account for laser beam thickness. Using these settings, the corner and suspension pieces meshed together very well, and the superglue was only needed to hold them in place.

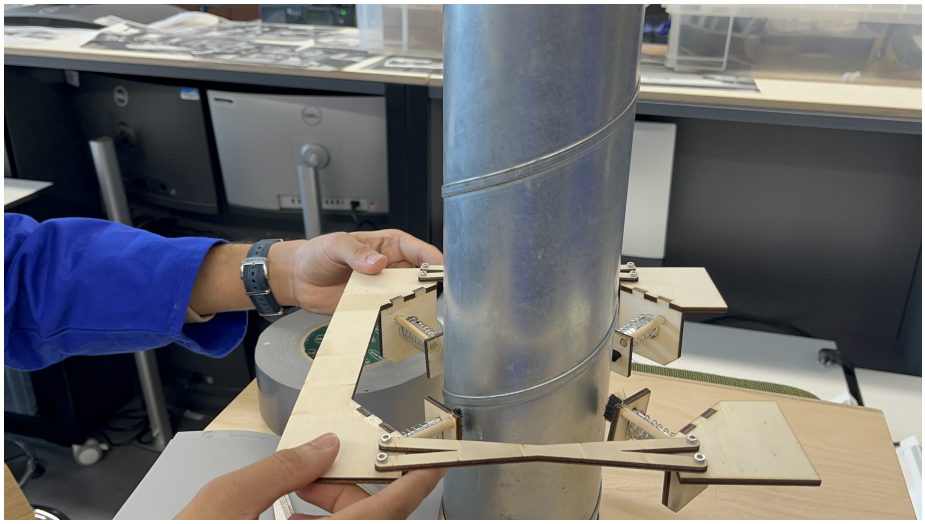


Figure 12: Test of Frame

Figure.12 presents a test of our frame on a cylindrical pipe of 110 mm. At this stage, the propellers were not mounted to the frame as seen in figure.11d. This test was useful for us to understand potential difficulties with our suspension. Additionally, it highlighted a slight disparity between our CAD model and the physical prototype because the brushes were not always in contact with the tree despite that being the case in our model. As a result, we created a slightly smaller frame, and this decision was further supported by changes in the challenge specification where we would face a smaller tree. A conclusion drawn from this test is that the suspension works appropriately and the springs have a suitable amount of stiffness. On the other hand, we envisage some problems with the friction between the brushes and the rough tree bark that could

cause the robot to rotate unexpectedly and potentially crash. While this is a potential problem, we will only be able to confirm this with further testing.

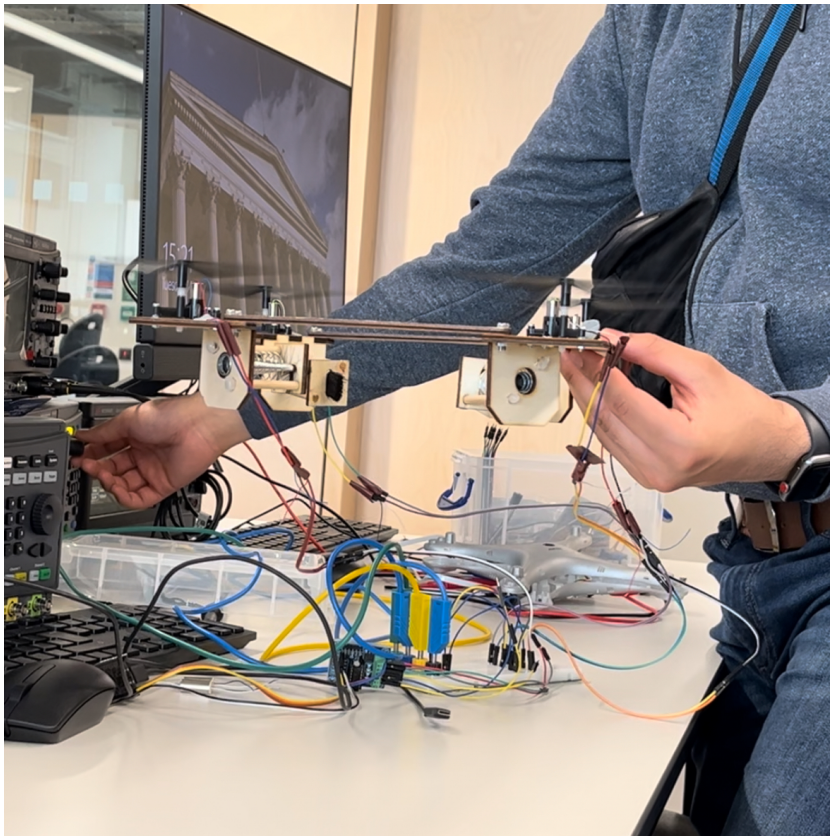


Figure 13: Test of Flight

The above figure presents a test of the thrust from our propellers. The arrangement shown above includes motors attached at opposite corners connected to the DC power source in parallel through a breadboard. Furthermore, in this figure, one can see how the motors have been fitted into the frames and the wires fixed to the plate with adhesive. The conclusion drawn from this test was that our frame was too heavy for us to achieve an optimal weight-to-thrust ratio with the motors and propellers available. Hence, to solve this issue we must either reduce the weight frame or create more thrust from the propellers. Additionally, we noted from the test that our robot was drawing a large amount of current, so attempting to increase the thrust is likely an unfeasible solution. With the current iteration, our robot was almost able to achieve lift but not to a suitable level where we could have efficient control of the robot.

Electrical Design

Circuit Diagram

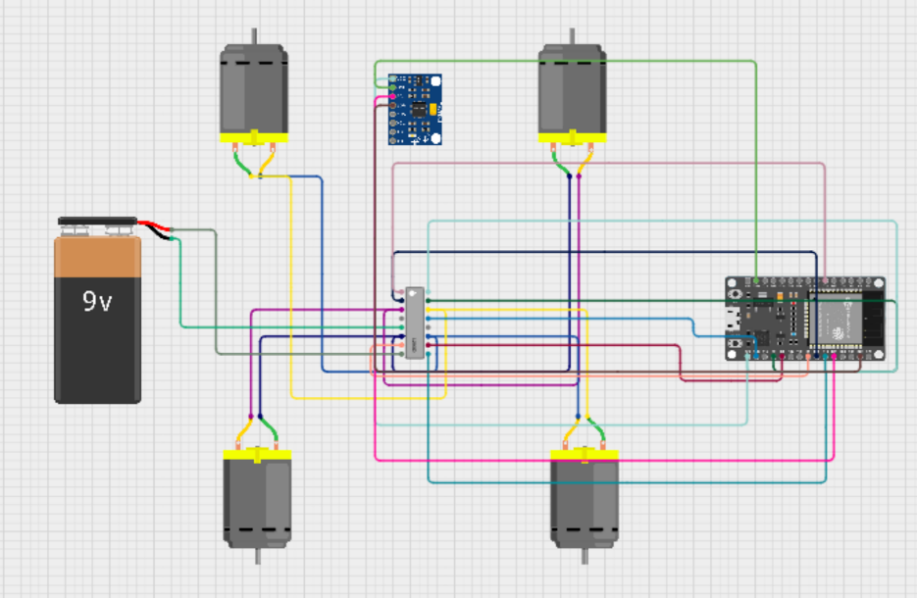


Figure 14: Circuit Diagram

In the figure above we can see a circuit diagram of the electronic components used in our robot. This circuit is designed to achieve synchronous propeller rotation with the diagonal motors. This is with the aim of achieving the motion discussed in the mechanical section of this report. Our design incorporates an L293D H-bridge motor driver IC. This allows us to use an external power supply to power the robot. Furthermore, our design employs PWM signals from the esp32 to control the speed and directions of the motors.

The motors used in our design are hollow cup cordless DC motors requiring 0.250 A at 3.7 V. The ultrasonic sensor mounted on top of the servo motor is purposed to avoid branches. To traverse down the tree, the device will rotate the ultrasonic sensor using the rack and pinion mechanism for sensor deployment.

Programming Logic

As the robot moves upward it continuously scans by performing a 360° turn to scan for the closest branch. It locks that distance on and moves upward. It aligns the slit in the frame to accommodate passing the branch. Once it

successfully passes a branch it then performs another 360° turn to scan for the closed branch and repeats the mentioned steps.

Once the robot reaches the top and deploys the 30 g sensor via magnets, the servo motor then rotates 180° pointing the ultrasonic sensor downward and then proceeds the steps in the previous paragraph again.

Signals

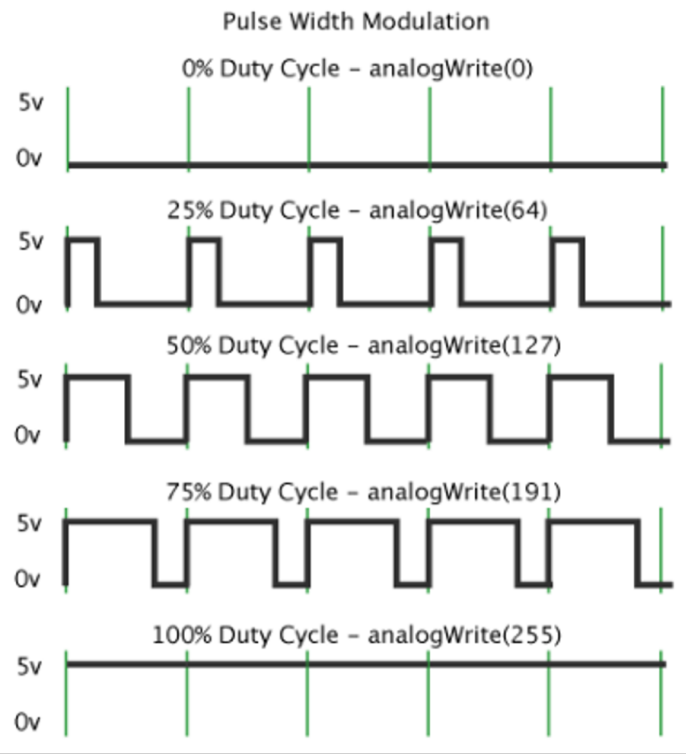


Figure 15: PMW Signals

The above figure conveys how PWM communication between our MCU and our motor driver will work. We will program the MCU using the Arduino IDE and thus utilize the 'analogWrite' function along with values between 0 and 255 to determine the percentage of 5V we want to supply from the MCU. This will enable control of the speed and thrust of each propeller. The percentage is calculated by with the below equation.

$$PWM_{Percentage} = \frac{input}{256} \times 100\% \tag{2}$$

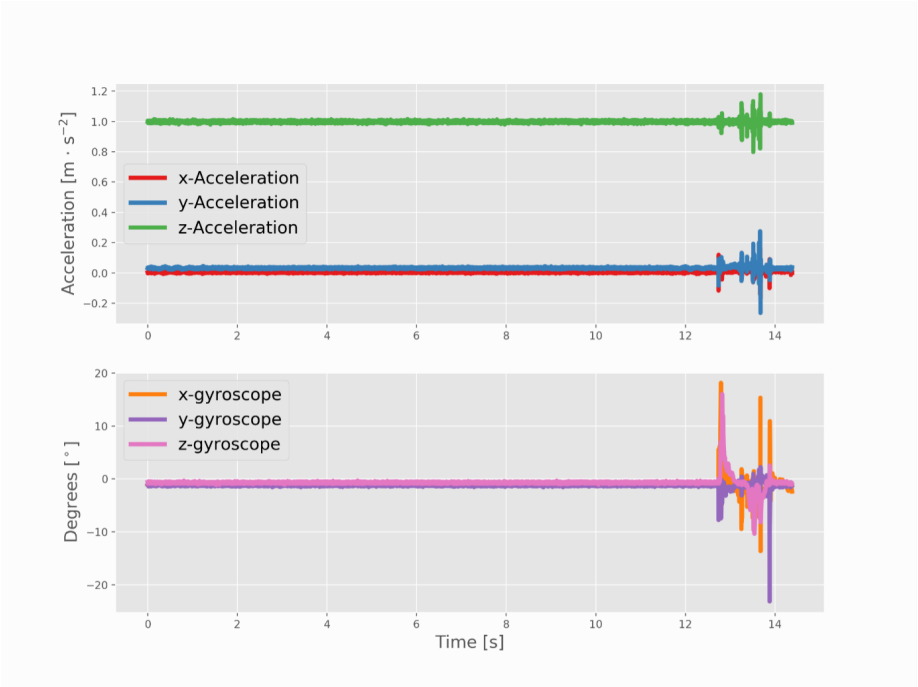


Figure 16: Signals from the MPU6050 Gyroscope

The MPU-6050, a 6-axis motion tracking device, combines a 3-axis gyroscope and a 3-axis accelerometer to detect changes in motion, acceleration, and rotation. We are using it because of its high accuracy and affordability. In a sensor module, it comprises a 3-axis accelerometer and a 3-axis gyroscope [4].

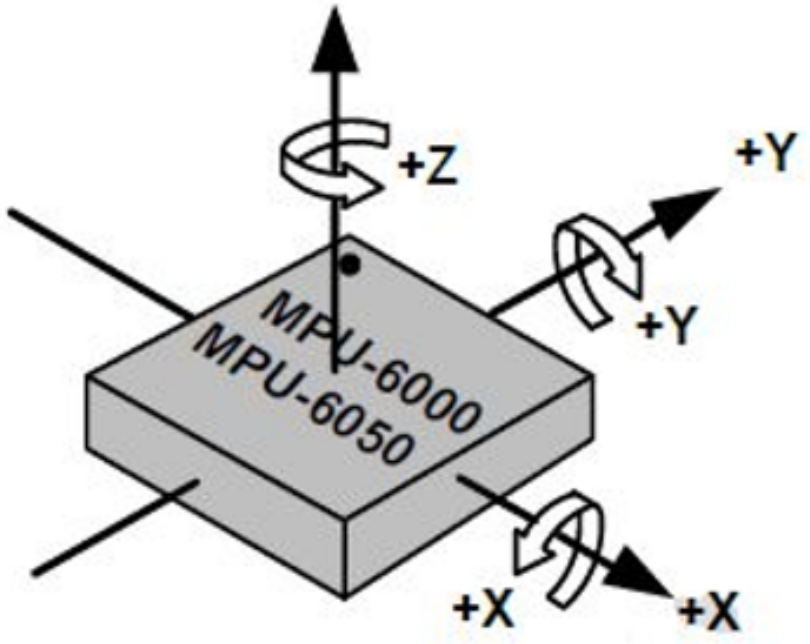


Figure 17: Axis Orientation with MPU6050 Module

The module's coordinate system is established by placing the MPU6050 flat on a table, ensuring the labeled face is upward with a dot in the top left corner. The z-axis extends upward, the X-axis from left to right, and the Y-axis from back to front (see figure.17).

Electronics Composition

A comprehensive list on most of the electrical components that will be utilised during the course of the project.

Table 1: Electronics Composition

Component	Description
1 x L293D H-Bridge Motor Driver	This module is used to drive the 4 drone motors. This module was chosen due to its lightweight nature and ability to control all the opposing motor pairs separately

Continued on next page

Table 1: Electronics Composition (Continued)

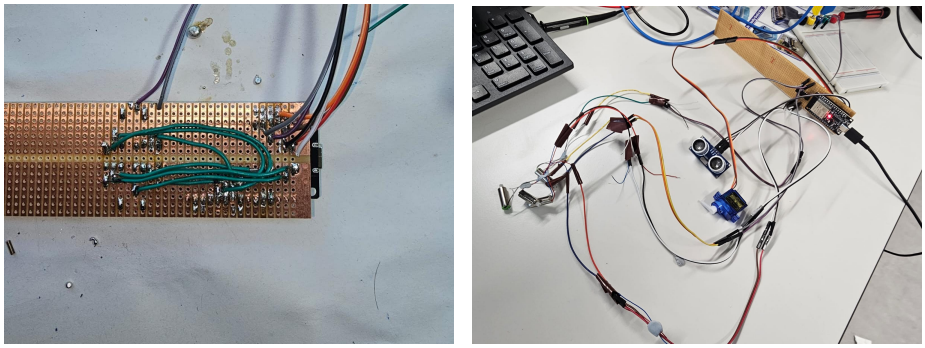
Component	Description
4 x Hollow-Cup cordless DC motors	These motors were selected because they provide 50000 rpm. Motors with enough speed are vital for any drone to work, so these motors were necessary for us to achieve liftoff.
1 x SG90-HV	The SG90-HV is a servo motor that is capable of 360° continuous movement. This component is purposed for the sensor deployment and ultrasonic sensor rotation systems. It was also chosen because it is a small module, weighing only 9 g, making it suitable for a product where every gram matters and because of its precise position control.
1 x HC-SR04	The HC-SR04 is an ultrasonic sensor that we will use for precise distance mapping of the branches. It was the selected ultrasonic for this prototype because it is the most common ultrasonic sensor with widely available libraries. Furthermore, through prior tests we concluded that it could detect branches of 20 mm diameter at a distance up to at least 500 mm away, making it suitable for this task.
1 x ESP-32	The ESP-32 is the micro-controller unit that was selected for this project due to its Bluetooth and WIFI capabilities. These aspects of this MCU are beneficial for debugging of our control system as we would be able to send commands through Bluetooth. Additionally, it is relatively small and lightweight while still being powerful, making it suited for our project.

Continued on next page

Table 1: Electronics Composition (Continued)

Component	Description
1 x External Power Supply	For our robot to be powered we will use an external power supply of 30W. Even though drones typically use lithium-ion batteries as their power source, it is unrealistic for us to use such a power source because of the capacity required. As mentioned, our drone motors draw a large current and would likely drain widely available batteries very quickly. Since our robot, is not required to fly at a very high altitude at this stage in the design, we would be able to use wires connected to an external power supply to power the robot. As a result, we avoid the capacity limitations that we may face with a li-ion battery.

Electronics Assembly



(a) Soldering of Components (b) Electrical Components in a Circuit

Figure 18: Electronics Assembly

Figure.18 depicts how the circuit was assembled before being attached to the frame. In figure.18b, one can see the ultrasonic sensor, the four DC motors and the servomotor. This circuit is a reflection of figure.14. The mentioned components have been soldered unto their respective pins on the ESP-32 using a PCB prototyping board (see figure.18a)

Ethical Considerations

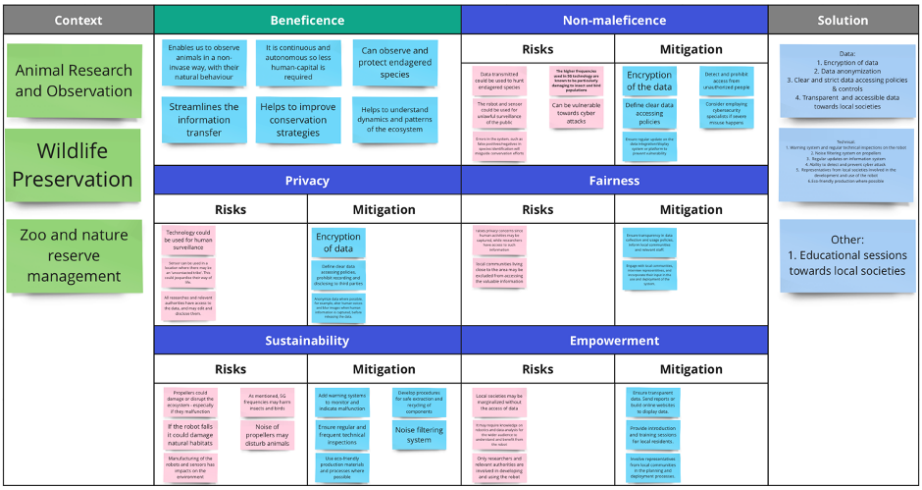


Figure 19: Summary of Ethical Considerations

The following is the URL to our ethics board (seen above): `https://miro.com/app/board/uXjVKLun_zM=/?share_link_id=691757773349`

Control System

Figure 20 is an overview flow chart of the control algorithm used for our robot.

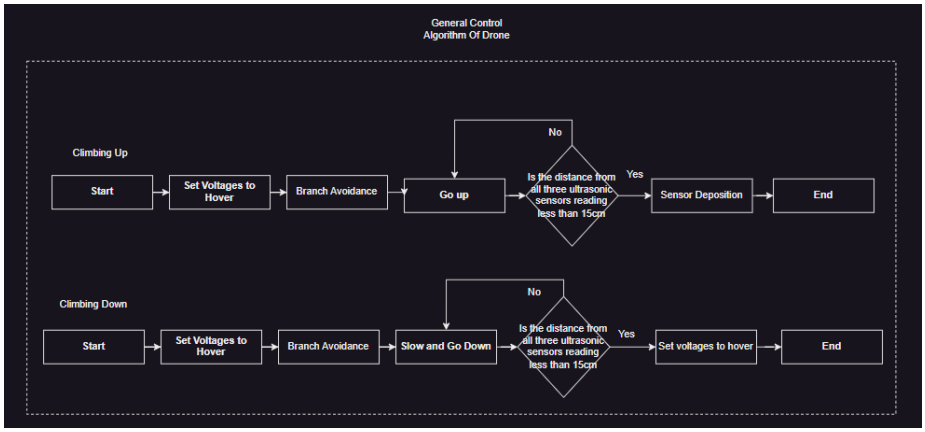


Figure 20

Control Algorithm Overview

Our control algorithm consists of two parts: one for climbing up and one for climbing down. There are five main actions our drone can perform: move up, move down, hover, deposit a sensor, and avoid branches.

Figure 21 is where we break down some of the main functions in our program.

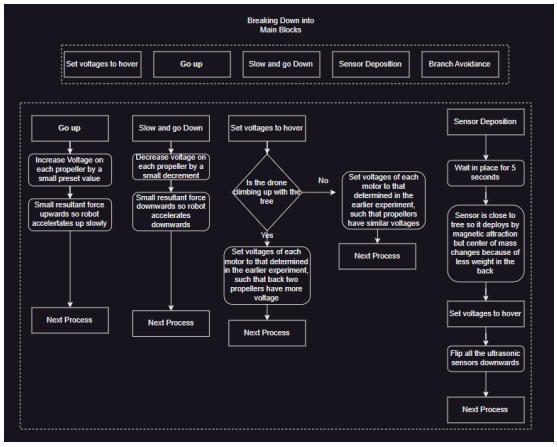


Figure 21

Moving Up

To move up, the voltage is incrementally and evenly increased to each motor, which increases the thrust upwards. The friction at the contact points allows the drone to travel upward at a constant velocity.

Moving Down

To move down, the voltage at each motor is evenly and slightly decreased, allowing the drone to slide down. The friction at the contact points slows the descent.

Hovering

Hovering occurs when the voltage supplied to each motor is set to a baseline. There are two baseline settings:
With Sensor Box Attached: The back two motors receive more voltage to balance the weight of the sensor box.
Without Sensor Box: The back two motors receive similar voltage to the front two motors since there is no additional weight.

Depositing the Sensor

The depositing sensor action is triggered when all three ultrasonic sensors detect that the drone has reached the top of the tree, indicated by recording a similar distance of less than 15 cm. The drone then hovers in place long enough for the sensor box to attach to the top of the tree. After the sensor is deposited, the drone's balance shifts, and the voltage settings are reset to account for the reduced weight. Additionally, the ultrasonic sensors are flipped downward using servo motors to prepare for descent.
Figure 22 is a flowchart for branch avoidance

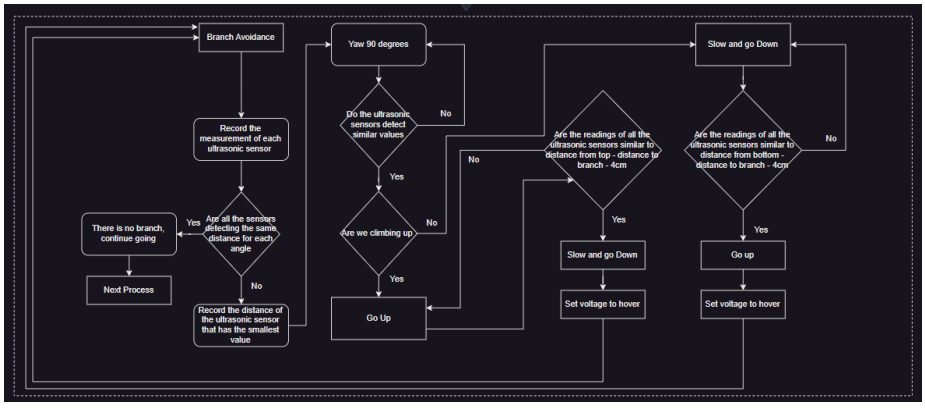


Figure 22

Branch Avoidance

The branch avoidance algorithm activates when an anomalously close distance is detected by one of the sensors, indicating a branch. The drone then rotates 90 degrees from its original orientation by increasing the voltage to two diagonal motors. The angle of rotation is measured by the MPU6050 sensor, which stops the rotation once the desired angle is reached. Due to the friction at the contact points and the slow movement of the drone, the rotation is accurate. The drone continues scanning and rotating until all sides are clear of branches. Once the path is clear, the drone can safely ascend to the top of the tree.

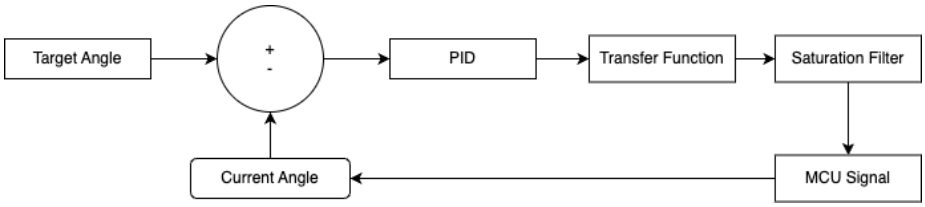


Figure 23: PID Control for Branch Avoidance

Figure.23 shows an example PID controller which could be used for the branch avoidance algorithm. The controller would take the target angle as input and adjust the voltage provided to the relevant motors as needed. The transfer function will be used to translate the PID value into a useful signal for the MCU. The saturation filter was added to prevent the signal from being too large or small that it would cause the roboto to act in an unintended way. Without the filter, the MCU could want to change the voltage to the motors and cause the robot to become unstable. The filter helps keep the signal in the right range that prevents instability but allows the robot to still get rotation in the yaw axis.

Ultrasonic Sensor Field of View

The ultrasonic sensors have a 30-40-degree effective field of view (FOV). With three sensors, their FOVs overlap, allowing them to detect branches above the drone on all three sides, provided the branches are 15-20 cm away from the drone. This meant, one scan from all sides can identify any branch no matter its angle.

Stabilization and Control

A control algorithm for stabilizing the drone's movement in the x, y, and z directions is unnecessary. The tree stabilizes and anchors the drone due to the tight contact with its brushes. Additionally, the tree supports and suppresses the drone's rotation in all axis. Therefore, developing a PID control algorithm to ensure specific thrust from the propellers is not required.

Power and Maneuverability

Given the weak power source and limited resources, the thrust provided by the drone's propellers is minimal. Designing a PID controller to control the drone's acceleration up and down the tree is unnecessary since the drone barely lifts itself even at near full power. The contact points mean that the friction increase with speed causing the drone's slow movement which prevents it from overshooting its target position as it breaks to a stop very quickly. Its lightweight, has low inertia, and so minimal power and force are required for direction changes, making it easily maneuverable.

Experiment for Motor Voltage Calibration

Before flying the drone up the tree, we designed an experiment to generate voltage values for each motor. This ensured the drone could counterbalance the displaced center of mass near the back when carrying the sensor box, which accounts for almost 30 percent of its weight. We generated a new set of voltage values to supply the motors when the center of mass shifted as the sensor box deployed.
We used a clamp stand with a swivel clamp along the roll axis of the drone. We clamped the drone at the intended center of mass. Each motor was individually connected to a power source that measured the voltage and current supplied. Using trial and error and observation, we determined a set of values for the shifted weight scenarios. We then hard-coded these values into the drone's code, increasing the magnitude equally to achieve the necessary thrust.

Programming

Link to code

The following is the github page for our code: `https://github.com/zlyksy/BAT`

Algorithm Explanation

Figure 24 is an overview flow chart of the algorithms used for our robot. First,

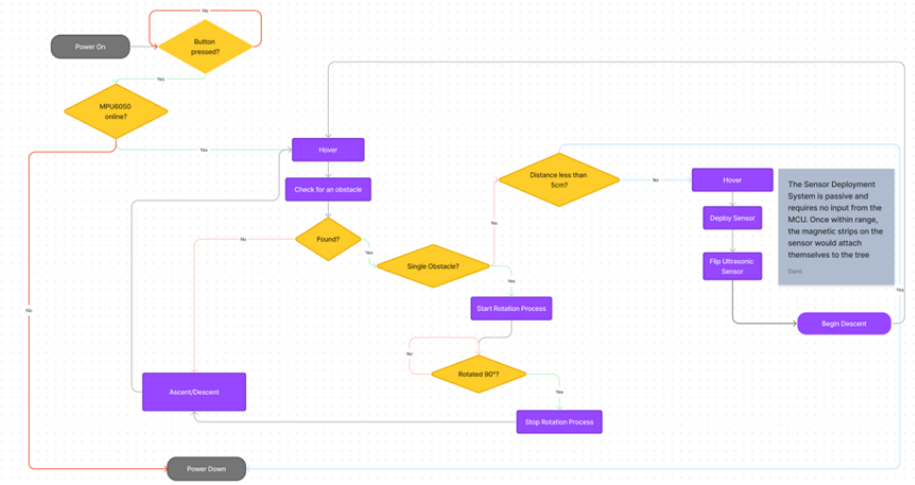


Figure 24

the robot needs to determine if the button has been pressed. To do this, it reads the current state of the button. If the button is pressed, the variable buttonPressed is set to 1. After this, the next loop begins. In the next loop, the button state read should be released. The state of the button being pressed and then released is interpreted by the algorithm as a signal to start the program. A "program start" signal is then outputted. Starting from the next loop, the program will execute the part after if start, which is the main program part.

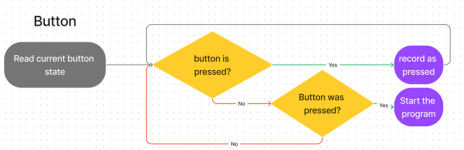


Figure 25

In the main program, the robot reads distances from the ultrasonic sensors and adjusts the robot's movements accordingly. The robot hovers initially and

checks for obstacles. If obstacles are detected at a close range, the robot tells if it's a branch or the ceiling by checking the data from all three sensors. If there is obstacle in only one side, then the robot understands that it is a branch and rotates to avoid it. If all 3 sides are blocked, which means the robot reaches the ceiling or floor, it either hovers and changes the direction flag, or powers down. Otherwise, it just keeps moving in the same direction Figure 26 is a simple chart that shows how the robot react when it reaches the floor and ceiling.

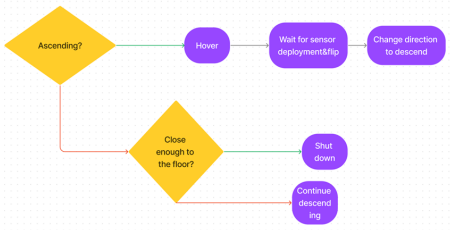


Figure 26

The rotate function is responsible for rotating the robot until it achieves a yaw angle of 90 degrees. It reads sensor data from the MPU6050, calculates yaw angle from both accelerometer and gyroscope data, and applies a complementary filter to smooth the data and ensure the accuracy. The motors are controlled with the motor driver to achieve the desired rotation speed and direction. After completing the rotation, the function ensures that the robot hovers.

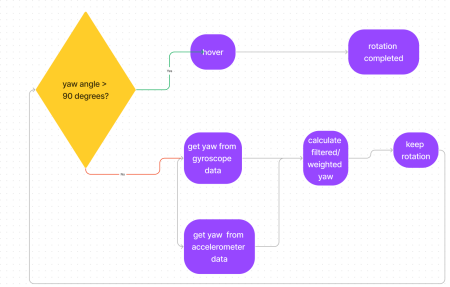


Figure 27: algorithm for rotation

The move function controls the robot's movement based on the given direction. It sets the motor speed according to whether the robot should move up, down, or hover. The motor speeds are set by writing appropriate values to the motor control pins.

Discussion

Due to issues with electronics, we were unable to attempt Task 3, branch avoidance, and therefore could not test our obstacle avoidance algorithm rotate().

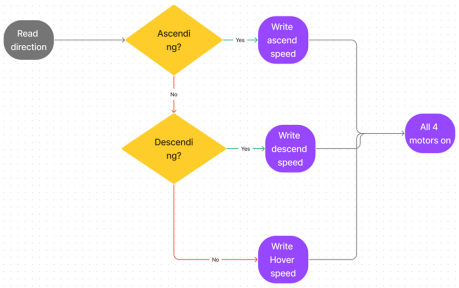


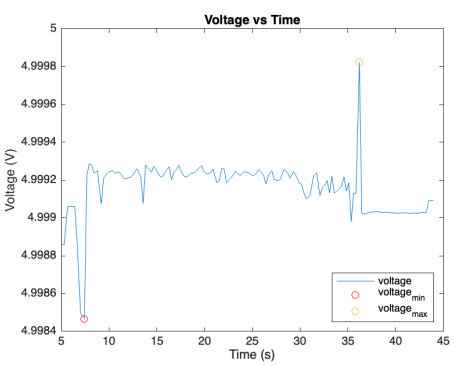
Figure 28

Nevertheless, in one of the initial testings, using lighter-weight electronic components just to test the code first, the get distance function, which controls the ultrasonic sensor, worked correctly, and the move function effectively controlled the robot.

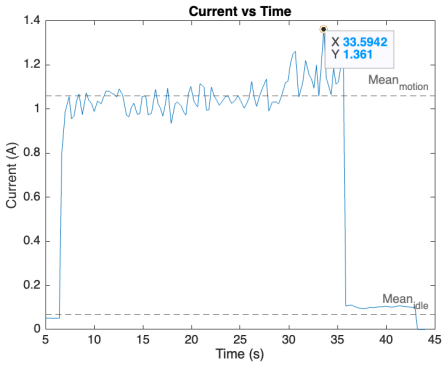
In reality, one side of the aircraft is often tilted, the center of mass is not at the exact center and there is sometimes difference in the performance of the motors. The flight can still be achieved, but if to have perfect level, it is not enough to just control the diagonal motors at the same speed; the motors on the lower side must operate at a consistently higher speed to provide extra lift, which is easy to achieve by writing the speeds of different motors separately. Since we ultimately decided not to attempt Task 3, we removed the two extra ultrasonic sensors to reduce weight. However, we still uploaded the complete code as it would be used under normal circumstances. After adding other components, the load increased from our early tests, and we did not conduct further tests with all three ultrasonic sensors due to not attempting task 3. If using three sensors caused excessive weight, we could reduce the load by using only one sensor and having the robot constantly rotating to check around. However, this would require increasing the detection distance and decreasing the movement speed, so that there is enough time allowed for the robot to react. Another thing being noticed is that the power-down process is quite abrupt, so the distance to floor when the robot should be shutting down is set to be very small. However, if the distance to the floor is increased for future optimization such as mentioned above, collisions during landing could cause some damage to the robot. A feasible improvement would be to introduce a hover state before powering down.

Figure 29: The robot was able to fly up and touch the top with the move function

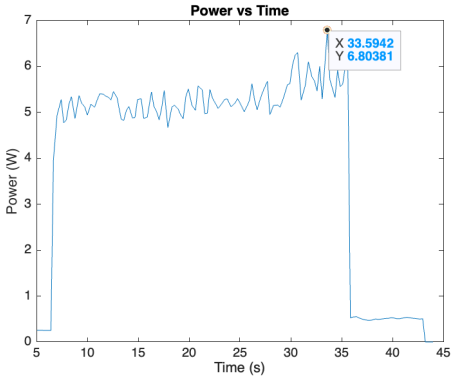
Mathematical Evaluation



(a) Voltage-Time Graph



(b) Current-Time Graph



(c) Power-Time Graph

Figure 30: Plotted Graphs of Electrical Quantities

Figure.30 presents the voltage, current and instantaneous power levels during a trial run of a wheeled robot climbing up and down the tree. Throughout the trial, the voltage levels are fairly consistent despite the fluctuations illustrated in figure.30a . The maximum and minimum voltages were 4.9998 V and 4.9985 V, respectively, with a range of 0.0013 V. This demonstrates that during the different stages of the trial, the voltage provided to the system was consistent. On the other hand, there is a significantly larger disparity seen in the current readings and consequently the instantaneous power readings. In figure.30b, the current ranges between 0 A and 1.3610 A. From figures.30b and 30c we can identify to operating states during the trial. Between the time of 6.41 s and 35.6 s, the robot was in its locomotion state while outside of that timespan, it was in an idle state. The locomotion state includes both the ascending and descending of the robot and has been identified by the

significant current consumption compared to the idle states. In the locomotion state, the mean current being consumed was 1.05 A while in the idle state it was 0.0692 A shown in figure.30b.

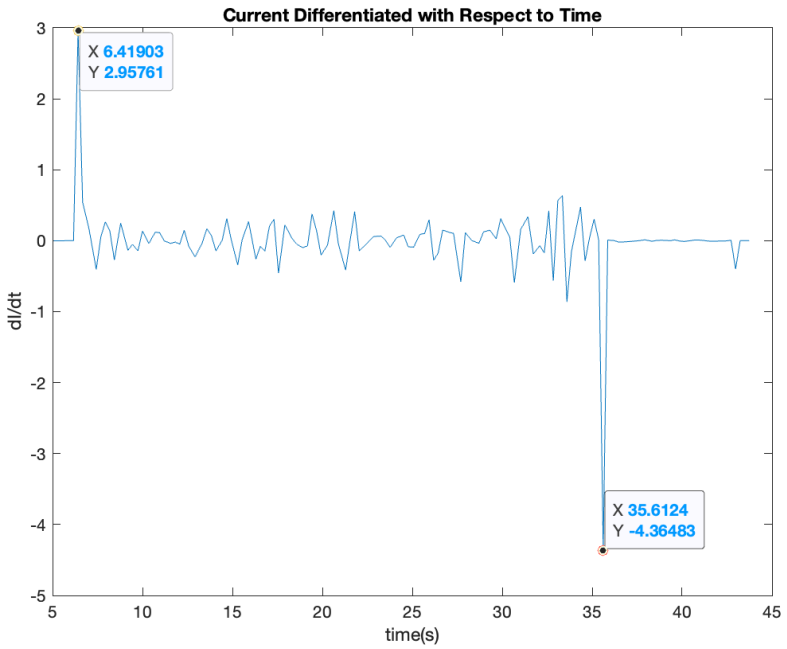


Figure 31: A Graph of Current Consumption Differentiated with Regard to Time

Numerical differentiation has been used in conjunction with the raw data, to identify the timestamps of changes in operation states (figure.31). For this analysis, we have considered the points with the greatest change in current as the points when the operating state changes. As mentioned above the timestamps for changes in operating state are 6.41 s and 35.6 s.

The max current and power has been depicted in figure.30b and figure.30c. For this trial the max power and current were 6.80 W and the 1.36A respectively. Throughout the motion state, there is continuous fluctuation in the current being consumed by the robot. There are various reasons for why the current and power vary throughout the trial. This could occur because of load variations whereby as the robot climbs it could be affected by the non-uniform surface. If there is more grip at different points or the diameter of the tree varies slightly, the robot has to adapt appropriately by drawing more or less power and this results in the graphs that we see above. As the wheels rotate up the tree, they can also slip or get stuck momentarily which would also produce slight variations.

To find the average power consumed, we have calculated the root mean square (RMS) power using equation.3 . RMS can be calculated by finding the square root of the average of a set's squared values, see below.

$$P_{RMS} = \sqrt{\frac{1}{N}\sum_{i=1}^{N} P_i^2} \tag{3}$$

For a sinusoidal waveform like an alternating current, the RMS can be found by dividing the amplitude by root 2.

$$\frac{A_0}{\sqrt{2}}$$

Using equation.3, the RMS power for this trial is 4.59 W. Please see Appendix.A for the detailed mlx file used for this mathematical evaluation section.

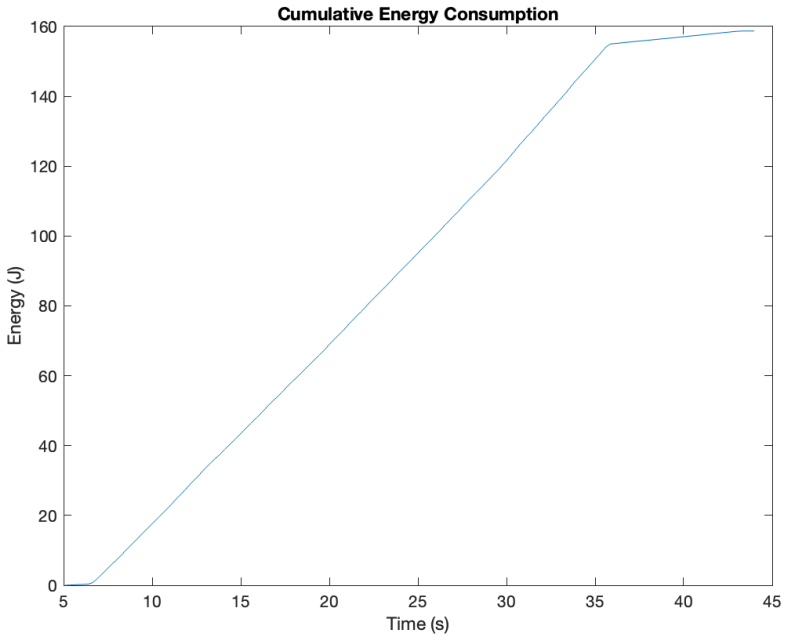


Figure 32: Cumulative Energy Consumption During the Trial

The total energy consumed during the trial was 158.7 J. In figure.32, the cumulative energy consumed has been plotted. The energy was computed by finding the numerical integral of the power with respect to time due to the relationship between power and energy.

$$Energy = Power \times Time$$

Figure.32 presents a steady and linear increase in the energy consumption of the robot. This steady increase is attributed to the stable power consumption throughout the trial. Interestingly, the power is stable throughout the trial even though the robot ascended and descended. While there was a gripping force that likely reduced the effect of gravity on the work done by the motors, one would expect the power consumption to decrease as the robot descends from the tree. This is because the robot is moving in a direction that is aided by gravity and consequently the motors should be doing less work to rotate the wheels. Since this is not the case it is likely the gripping force and friction on the tree is significantly greater than gravity and this alone can hold the robot up. As a result, as the robot moves, the work due to gravity is counteracted by the gripping force and the motors must work similarly as hard regardless of the direction of motion.

In order to evaluate the efficiency of this robot we can compare the useful mechanical energy exerted by the robot to the supplied electrical energy. Above we have computed that the total electrical energy supplied to the robot during its motion operating state was 158.7 J.

$$E_{mechanical} = U + K$$

Due to the nature of the robot's motion our useful mechanical energy is made of changes in the gravitational potential energy: calculated using the formula below.

$$U = mgh$$

Since the robot has climbed and descended the tree it covers a total distance of 4 m. Moreover, as the work due to gravity is being countered by the gripping force from the springs, one can assume the mechanical work is not being aided or hindered by gravity depending on direction. Therefore, the mechanical power is relatively constant throughout the run. Using the above formulae, the useful mechanical energy is 11.7 J.

$$\eta = \frac{Output_{Useful}}{Input_{Total}}$$

The efficiency of this robot is computed to be 7.42%. With such a low efficiency, it raises questions on how suitable this solution is for a tree-climbing robot. There is a significant loss of energy which can arise for various reasons. The main reason for these losses could be a result of the wheel-tree surface friction. If there is improper contact with the tree, then not all the electrical power is being converted into useful mechanical power. Moreover, the texture and shape of the tree bark can cause varying levels of friction, so as the robot tries to adjust to the varying grip levels there may be power losses. With such a

tight gripping force from the mechanical design, some energy was likely wasted in trying to overcome the inwards force on the tree.

Energy was also likely lost to vibrations, noise and heating. There may have been inefficiencies with the motors used in this robot as some of the electrical energy may have been converted into heat due to resistance in the wiring. As the temperature of the wiring may have increased, the resistance would also change, increasing the power being consumed. There may have been internal components whose resistance decreases with heating, consequently increasing the current being consumed. This hypothesis is supported by figure.30b where we can see an increase in the current being drawn by the circuit, while the voltage stays relatively constant in figure.30a. This is also inline with Ohm's law and the relationship between power, voltage and current (see below).

$$V = I \times R$$

$$P = V \times I$$

It is also possible that the variable current being drawn could be a result of faulty connections. Faulty connections can also cause intermittent contact which could act as a source for some of the peaks in the current that can be seen in figure.30b.

Throughout the subfigures in figure.30, there are some noticeable outliers. The most noticeable of these were the two peaks in figure.30a, while they only differ by 0.0013 V, they seem significant on the figure. One reason that these two outliers occurred was possibly due to minor voltage variations as the system was activated and subsequently switched to idle mode. These peaks also coincide with the changes in operation modes discussed before. The minor voltage drop is likely a result of the rapid increase in current from when the motors were switched on. Whereas the voltage peak is possibly due to the inductive kickback where a sudden change in current can cause the motor to create a "large voltage swing" in attempt to keep things "normal", analogous to inertia [5].

Discussion

Key takeaways

Undertaking the development of a unique tree-climbing robot with an air-propelled system required substantial confidence in our design and innovation. Initially, we considered having a backup plan in case of failure, but we successfully transformed our idea from imagination into reality. Key takeaways include:

- Time constraints and limited resources were significant challenges, affecting our ability to acquire necessary parts.
- Weight sensitivity was a critical issue; even minor weight increases (e.g., 5 grams) impacted performance.
- Our suspension system, even with modifications using makeup brushes, faced excessive friction, highlighting the need for a more pliable yet resistant mechanism in future designs.
- Weight management and distribution were crucial, as drones are particularly sensitive to these factors.
- Cardboard, as a lightweight and rigid material, proved surprisingly effective for certain components.
- A robust mechanical design can simplify coding complexity. For instance, we initially considered ultrasonic sensors for branch avoidance but found that the MPU6050 sensor provided faster and more efficient results, reducing the scenarios our code needed to handle.

What Worked and What Didn't!

The combination of springs and brushes effectively gripped trees of varying diameters, successfully addressing tasks 2 and 3. Housing the electronics at the base of the tree mitigated some weight issues. Our air-propelled system demonstrated feasibility.

However, we spent excessive time on tasks 2 and 3 without perfecting task 1. The electronics posed significant challenges. Sourcing extremely lightweight H-bridge motor drivers was problematic, and our fallback on L293D drivers required extensive adaptation. Developing our own motor drivers was impeded by a lack of necessary components like MOSFETs and diodes. Additionally, we couldn't fully test the branch avoidance system due to unresolved electronics and voltage distribution issues.

Surprising Discovery

A surprising discovery occurred three weeks into the project during motor testing. Initially, we powered the motors via an external power supply, assuming the transition to an H-bridge motor driver would be straightforward. However, each motor required 1.2A at 3.4V, exceeding the 600mA capacity per channel of the standard L293D. We resolved this by engineering a custom soldered PCB that combined two channels to support 1.2A per motor, using four L293Ds for four motors. Although this workaround was effective, in hindsight, ordering more capable motor drivers like the TMC2208, which supports 1.4A per channel, would have been preferable. Unfortunately, time constraints prevented this option, leaving us to work with the L293Ds we had.

Conclusion

If we could redo this project from scratch, several key changes would improve our process. First, starting with cardboard as our primary material instead of wood would have optimized our limited time and resources. Cardboard proved to be surprisingly effective—lightweight yet rigid enough for our needs—allowing us to make quicker progress on the structural components.

Early recognition of the motors' 1.2A current requirement would have enabled us to procure the appropriate motor drivers from the beginning, avoiding the need for extensive adaptations later. This foresight would have saved us considerable time and effort, allowing us to focus more on other critical aspects of the project.

Additionally, having access to pre-calculated springs with the correct coefficients would have streamlined our design process. Our efforts to manually coil steel wire to create springs not only delayed our mechanical development but also introduced variability and inefficiency into the system. With the right springs, we could have expedited our design progress and improved the overall performance of the robot. Although our programming and control systems were mostly ready, we lacked the opportunity to test them due to delays in the physical hardware. Procuring the necessary MOSFETs, diodes, and other electronic components at the outset would have allowed us to integrate and test our systems more reliably. This would have shifted our focus from struggling with hardware issues to refining our programming and control, leading to a more robust and reliable robot.

By making these adjustments, we would have better allocated our time and resources, improving both the efficiency and effectiveness of our project development. This experience was a great one teaching all of us many things in just a months time.

Appendices

A Matlab Code

```
clear; clc;

%import the data

data = readmatrix("sample_robot_run.csv");
% TIME - VOLTAGE - CURRENT


Data Analysis

%1.1 Plot voltage , current and instantaneous power as functions of time

t = data(:, 1); % time
v = data(:, 2); % voltage
i = data(:, 3); % current
power = v .* i; % calculate instantenous power


% find peak current and power
[current_peak, current_peak_idx] = max(i);
[power_peak, power_peak_idx] = max(power);

% find min and max voltage
[min_v, min_v_idx] = min(v);
[max_v, max_v_idx] = max(v);
mean_voltage = mean(v)

% find max current drawn
current_range = range(i);


RMS
N = length(power); % Number of samples
squared_sum = sum(power.^2);

rms = sqrt(1/N * squared_sum) % calc RMS

Energy Calculation
E_total = trapz(t, power) % integrate the E = P*t
```

```
E_total_cum = cumtrapz(t, power);  % get cumulative intergral for graph

figure(1)
plot(t, E_total_cum)
xlabel('Time (s)')
ylabel('Energy (J)')
title('Cumulative Energy Consumption')

% EFFICIENCY
Emech = 0.3 * 9.81 * 4;

Efficiency = Emech / E_total


Average Current
% get average current for locomotion state
motion_avg_i = mean(i(6:104))

% avg for idle state
idle_avg_i1 = mean(i(1:5));
idle_avg_i2 = mean(i(105:end));
idle_avg_i = (idle_avg_i1 + idle_avg_i2)/2 % idle average

% get the max and min gradient
current_diff = diff(i) ./ diff(t);
[max_grad, max_grad_idx] = max(current_diff);
[min_grad, min_grad_idx] = min(current_diff);


PLOTTING
% voltage - time
figure(2)
plot(t, v, t(min_v_idx), min_v, 'ro', t(max_v_idx), max_v, 'o')
xlabel('Time (s)')
ylabel('Voltage (V)')
legend({'voltage', 'voltage_{min}', 'voltage_{max}'}, 'Location','southeast')
title('Voltage vs Time')

% current - time
figure(3)
plot(t, i, t(current_peak_idx), current_peak, 'o')
xlabel('Time (s)')
ylabel('Current (A)')
title('Current vs Time')
yline([motion_avg_i, idle_avg_i], '--', {'Mean_{motion}', 'Mean_{idle}'})
```

```
xlim([5.0  45.0])
ylim([0.00  1.40])

ax2 = gca;
chart2 = ax2.Children(3);
datatip(chart2,33.59,1.361);

%power - time
figure(4)
plot(t, power, t(power_peak_idx), power_peak, 'o')
xlabel('Time (s)')
ylabel('Power (W)')
title('Power vs Time')

ax2 = gca;
chart2 = ax2.Children(1);
datatip(chart2,33.59,6.804);

% dI/dt
figure(5)
plot(t(1:end-1), current_diff, t(min_grad_idx), min_grad, 'ro', t(max_grad_idx),
title('Current Differentiated with Respect to Time')
xlabel('time(s)')
ylabel('dI/dt')

xlim([5.0  45.0])
ylim([-5.0  3.0])

ax = gca;
chart = ax.Children(1);
datatip(chart,6.419,2.958);
chart = ax.Children(2);
datatip(chart,35.61,-4.365);
```

References

[1] P. Beardsley et al. “Vertigo: Autoencoder-based Image Compression”. In: *Disney Research* (2019). URL: https://la.disneyresearch.com/publication/vertigo/.

[2] Shunzhi Bian et al. “A Novel Type of Wall-Climbing Robot with a Gear Transmission System Arm and Adhere Mechanism Inspired by Cicada and Gecko”. In: *Applied Sciences* 11.9 (2021), p. 4137. DOI: 10.3390/app11094137.

[3] G. Fang and J. Cheng. “Advances in Climbing Robots for Vertical Structures in the Past Decade: A Review”. In: *Biomimetics* 8.1 (2023), p. 47. DOI: 10.3390/biomimetics8010047.

[4] Joshua Hrisko. *MPU6050 Arduino High-Frequency Accelerometer and Gyroscope Data Saver*. Maker Portal. 2019. URL: https://makersportal.com/blog/2019/8/17/arduino-mpu6050-high-frequency-accelerometer-and-gyroscope-data-saver (visited on 05/13/2024).

[5] Inductive Kickback. *Inductive Kickback Made Simple to Grasp, Easy to Handle*. 2019. URL: https://inductive-kickback.com/2019/04/inductive-kickback-made-simple-to-grasp-easy-to-handle.

[6] Sangbae Kim et al. “Whole body adhesion: hierarchical, directional and distributed control of adhesive forces for a climbing robot”. In: *Proceedings 2007 IEEE International Conference on Robotics and Automation*. 2007, pp. 1268–1273. DOI: 10.1109/ROBOT.2007.363159.

[7] Tin Lun Lam and Yangsheng Xu. “A flexible tree climbing robot: Treebot - Design and implementation”. In: May 2011, pp. 5849–5854. DOI: 10.1109/ICRA.2011.5979833.

[8] University of Notre Dame. *The Hairy Feet of the Gecko*. 2021. URL: https://sites.nd.edu/biomechanics-in-the-wild/2021/04/07/the-hairy-feet-of-the-gecko/.

[9] N. Schneider. *Cross Bracing: Why It's Important*. URL: https://origindesign.com/articles/cross-bracing-why-its-important (visited on 05/13/2024).