Concordia University

Report presented to

M. Khashayar Khorasani

As a requirement for

Introductory Engineering Team Design Project [ENGR 290 - U]

By

Alexandre Vallières (4)

Andres-Antonio Camacho-Suarez (4)

Ravjotdeep Kaur (4)

Rifadul Haque (4)

Saro Nokhoudian (4)

Final Report for the Design of an Autonomous Hovercraft

Monday April 26th, 2021

1. Abstract

The objective of this project is to design a small-scale hovercraft that will complete an obstacle course within the allotted time of 60 seconds in a VR environment. Only the PDR part has been done yet. During that stage, we researched and designed 3 potential prototypes that could potentially accomplish the goal at hand. We then chose the third prototype to act as our main design with the help of SWOT and AHP analyses. After implementing it into CoppeliaSim and having worked on it for some time, we had to rethink our approach to the goal of the project to be able to deal more easily with the torque generated by the lift fan, which was causing a lot of problems with stabilization and turns. We settle on a simplistic, square-shaped design that would use the walls to stabilize itself. This design completed the track in 38 seconds with 9 components using the proximity sensors, earning a score of 3.65 for a track distance of 1250 cm. However, the hovercraft couldn't even move when using the proximity sensors.

Keywords: Hovercraft, Simulation, MATLAB

2. Table of Contents

1. Introduction

1.1. Operating Principles of a Hovercraft

A hovercraft is an air boosting vehicle, which can travel on land, on water, and some other surfaces. As the name of the vehicle implies - it is a mode of transportation that 'hovers' over various surfaces to move in a forward or backward direction. The main principle of the hovercraft is to produce a large volume of air below the hull or air cushion which produces lift and causes the vehicle to float a few centimeters above the surface. A hovercraft can be broken down into three significant parts: propeller, lift fan and skirt. Each part has its own distinctive role to play when it comes to operating the hovercraft:

Propeller - A mechanical device for propelling a boat or aircraft, consisting of a revolving shaft with two or more broad, angled blades attached to it, which creates the thrust required to move the hovercraft forward by pushing the air behind the vehicle [1]. The rotational motion of the blades creates thrust by creating a pressure difference between the two surfaces. This causes the air to blow in one direction making the hovercraft move in the opposite one. Bernoulli's principle and Newton's third law of motion are the two phenomenon that are responsible for operating a propeller [2].

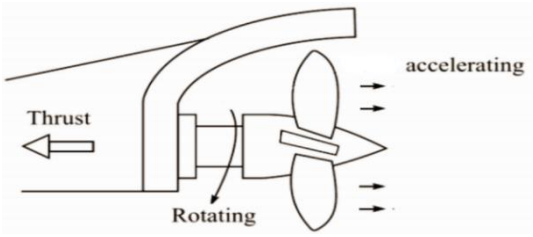

Figure 1: A typical propeller [4]

Lift fan - The lift fan is responsible for lifting the air-cushioned vehicle by pushing the air in the downward direction, thus keeping the craft afloat. The combination of two fans is responsible for the functioning of a hovercraft: the lift fan and the rear fan [3]. The central fan is responsible for creating the lift needed to keep the hovercraft float while the rear fan acts as the propeller that makes the vehicle move in a certain direction. Instead of a rotating rear fan, a hovercraft can also have two non-angled ones that can generate varying forces to make it turn [3].

Skirt - A skirt is a flexible piece usually made of rubber that traps the air underneath the craft or the hull, creating an air cushion and thus a constant pressure that lifts the vehicle by Newton's third law of motion [3]. It also helps clearing or removing any obstacles in the hovercraft's way and reduces the power needed to lift it by its efficiency at maintaining the air cushion. There is more than one way to use the lift fan and the skirt to create an air cushion, like the open plenum and momentum curtain configurations [3].

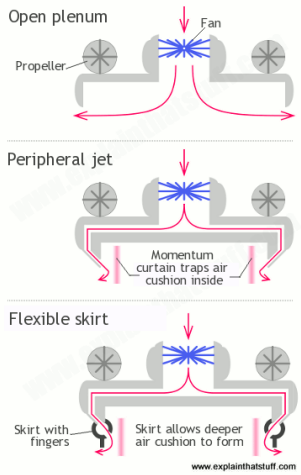

Figure 2: Multiple lift fan and skirt configurations [5]

1.2. Advantages and Disadvantages of a Hovercraft

One of the main advantages of the hovercraft is its ability to travel on sand, water, ice, mud and many other surfaces. Because it hovers over the surface, it has the ability of traveling the river up with same velocity as down, as there is almost no interference by the waves, and it can pass over debris and logs without any collision. This is very useful for travels over different terrains like in army deployments on a beach.

However, even though a hovercraft can travel any type of surface, it does not have the capability to travel slopes that are too high, so it is understandable that steep terrain will be a problem. It is also very loud because of its multiple fans and damaging its skirt can be catastrophic: it will lose its ability to hover and will stop moving forward. Hovercrafts are usually enormous and because of this they are challenging to control at high speed.

1.3. Project Goals

The main objective is to complete a virtual reality-based track from start to finish as quickly as possible with an originally developed and implemented autonomous hovercraft, using a minimum amount of resources. The simulation in which the hovercraft will complete the track will be implemented on CoppeliaSim and can be found in Figure 3. It is the same for every team and the goal is to compete against time itself, as we only have 60 seconds to complete it. The track is shown in Figure 3 and has a wall-to-wall segment length "x" of 275 cm, a wall-to-wall width "y" of 50 cm in the segments and 55 cm in the turning points and a wall height "z" of 20 cm. The 2nd, 3rd and 4th segments each have a bump of 1 mm, 2 mm and 3 mm respectively in the middle as obstacles.

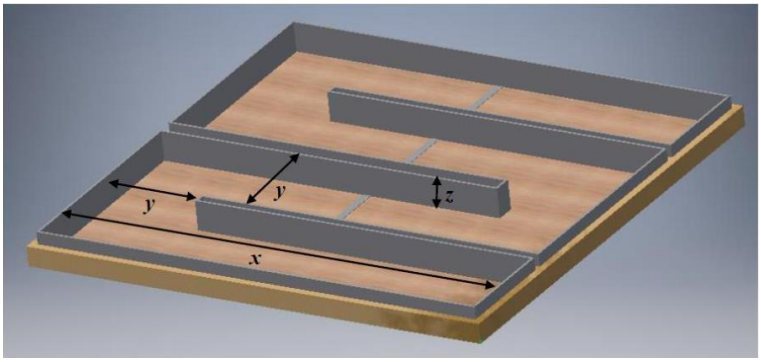

Figure 3: Track to be completed

2. Project Requirements

This project requires analyzing the physics behind the motion and operation of the hovercraft and taking quantitative measurements based on MATLAB simulations of the system of differential equations governing them. Furthermore, we need to research the advantages and disadvantages of all recorded team designs and develop an ideal model by continuously self-evaluating the hovercraft. After having proposed 3 different designs and chosen one of them as the final one, it will need to be analyzed in MATLAB and implemented it in CoppeliaSim for the final competition.

3. Design Process

3.1. Summary of the 3 Prototypes

The 3 prototypes that could possibly be implemented are listed in this section. It is important to keep in mind that in the drawings, the skirts have not bee implemented. However, all designs will have one.

3.1.1. Design 1

The base has a rectangular shape and has a bump on the front of the hovercraft with a 45-degree inclination. The bump on the front side allows for smoother rotational movements. With this diameter it also gets enough space to carry its components such as the batteries, sensors and fans.

It has 2 fans: one for lifting on the middle of the base and one in the back for propulsion. There are 4 sensors, two in the front and two in the back. Each one of them is at one corner. The batteries will be placed at the back of the hovercraft so that it doesn't completely wedge forward.

The advantage resides in the stability of the hovercraft since the center of mass is well adjusted with the base of the hovercraft (bump), thus steering will be easier. However, this increase in mass means that the vehicle is heavier, making its hovering height smaller and potentially preventing it from completing the obstacles in the course.

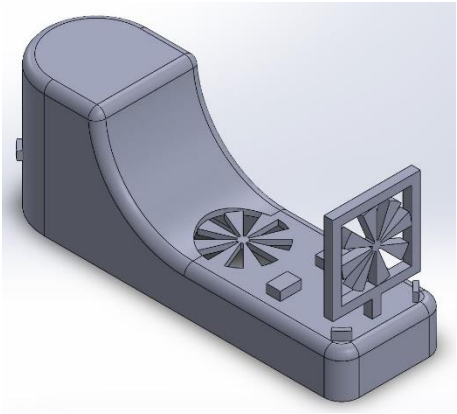

Figure 4: Preliminary drawings of design 1

3.1.2. Design 2

This design is essentially a workaround to the lifting problem from the first one. The body is the same, but the components change: there are two lift fans instead of one. This means that lifting will be easier, which makes less friction and thus less power required for movement, so there should be more maneuverability. This should be shown by the mathematical simulations.

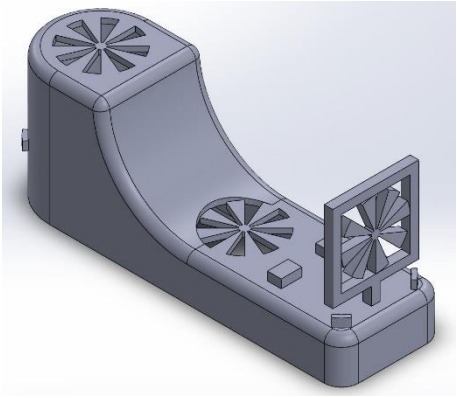

Figure 5: Preliminary drawings of design 2

3.1.3. Design 3

The base of this model has an overall rectangular shape with a single round edge on the front and square edges on the back. The design has 2 sensors in the front located at its edges and pointing at a 45-degree angle from the front of the vehicle with the same thing in the back, making a total of 4 sensors. There are 2 fans, one at the middle of the hovercraft for lifting and one at the back for propulsion. The batteries will be placed at the front to balance out the craft.

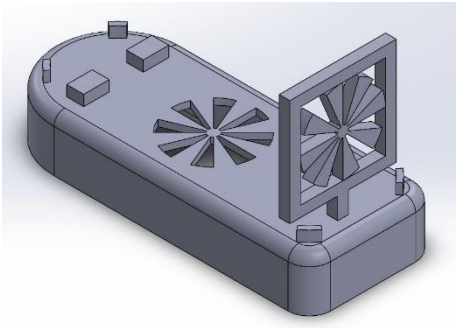

Figure 6: Preliminary drawings of design 3

The advantage of this hovercraft is the precision in movement. Because of its circular edge at the front, it is easier to move through obstacles. In addition, this model is light since it uses less

resources, making it faster and giving it more maneuverability than the other designs given the same thrust from the propulsion fan.

3.2. Calculations Based on the Proposed Designs

3.2.1. Proposed Properties

Although all 3 designs are different, they have a few things in common. For example, the types of components used are the same. This was meant to facilitate the comparison between them and remove the possible influence that different masses for the same things could have on the decision-making process. Nonetheless the designs are still different, and the way components impact the system of mechanical equations can be different too. Here are the proposed components and their location for each of the designs:

Component	Type	Mass (g)	X distance from the center (cm)	Y distance from the center (cm)
Propeller	AFB1212SH	200	9.75	0
Lift fan	AFB1212SH	200	0	0
Servo motor	HS422	45	13	0
4 x Sensors	GP2YOAO2YKOF	5	+/- 14	+/- 8.25
2 x Batteries	Rhino 360	23	-11.95	+/- 3.85
Bump	N/A	1 000	11.25	0

Table 1: Proposed properties for design 1

Component	Type	Mass (g)	X distance from the center (cm)	Y distance from the center (cm)
Propeller	AFB1212SH	200	9.75	0
2 x Lift fans	AFB1212SH	200	+/- 6	0
Servo motor	HS422	45	13	0
4 x Sensors	GP2YOAO2YKOF	5	+/- 14	+/- 8.25
2 x Batteries	Rhino 360	23	-11.95	+/- 3.85
Bump	N/A	1 000	11.25	0

Table 2: Proposed properties for design 2

Component	Type	Mass (g)	X distance from the center (cm)	Y distance from the center (cm)
Propeller	AFB1212SH	200	9.75	0
Lift fan	AFB1212SH	200	0	0
Servo motor	HS422	45	13	0
4 x Sensors	GP2YOAO2YKOF	5	+/- 14	+/- 8.25
2 x Batteries	Rhino 360	23	11.95	+/- 3.85

Table 3: Proposed properties for design 3

It is also worth noting that the coefficient of friction b is based on research on the average values it has for a normal hovercraft. Since there is no way to calculate it without more information like the materials that enter in contact with the air cushion, we needed to subjectively decide on a value depending on the properties of the HC in comparison to the ones found in our research. This gives the following variables for each design to be inserted in the equations described next:

Design	Total mass m (kg)	Total inertia I (kg*m^2)	Coefficient of friction b
1	1.542	0.010038	0.5
2	1.742	0.010978	0.3
3	0.542	0.004413	0.5

Table 4: Variables of each design for the equations

3.2.2. Differential Equations

We start getting the equations by getting the kinetic energy of the system with the following relation:

$$KE = \frac{1}{2} * (mu^2 + mv^2 + Ir^2)$$

Equation 1: Kinetic energy of the system

Since the potential energy isn't considered since the height of the system is constant, the Lagrangian is just equal to the kinetic energy. We then plug it into the following Euler-Lagrange equation:

$$\frac{d}{dt}\left(\frac{dL}{d\dot{q}}\right) - \frac{dL}{d\dot{q}} = F + Q$$

Equation 2: Euler-Lagrange generalized equation

In this equation, F is the force applied and Q is the force generated by friction. By differentiating this equation for each variable individually, we obtain the following system:

$$m\dot{u} = F_u - bu \rightarrow m\ddot{x} + b\dot{x} = F_u$$
$$m\dot{v} = F_v - bv \rightarrow m\ddot{y} + b\dot{y} = F_v$$
$$I\dot{r} = T_r - br \rightarrow I\ddot{\Psi} + b\dot{\Psi} = T_r$$

Equation 3: System of equations based on Euler-Lagrange

$\dot{x} = \dot{u} = acceleration\ on\ x$
$\dot{x} = u = velocity\ on\ x$
$\ddot{y} = \dot{v} = acceleration\ on\ y$
$\dot{y} = v = velocity\ on\ y$
$\ddot{\Psi} = \dot{r} = acceleration\ on\ \Psi\ (rotational)$
$\dot{\Psi} = r = velocity\ on\ \Psi\ (rotational)$
$b = friction$
$m = mass$
$I = inertia$

By isolating the acceleration in each equation, we obtain the following system that can be put in a MATLAB simulation with each design's properties:

$$\ddot{x} = \frac{1}{m} * (F_u - b\dot{x})$$
$$\ddot{y} = \frac{1}{m} * (F_v - b\dot{y})$$
$$\ddot{\Psi} = \frac{1}{I} * (T_r - b\dot{\Psi})$$

Equation 4: Final system of equations

3.2.3. MATLAB Mathematical Simulations

After finding the system of 2nd order equations of motion, they can be put into a simulation that can calculate the displacement, velocity and acceleration on the x, y and rotational axis. The following is the Simulink diagram used in MATLAB to generate these results:

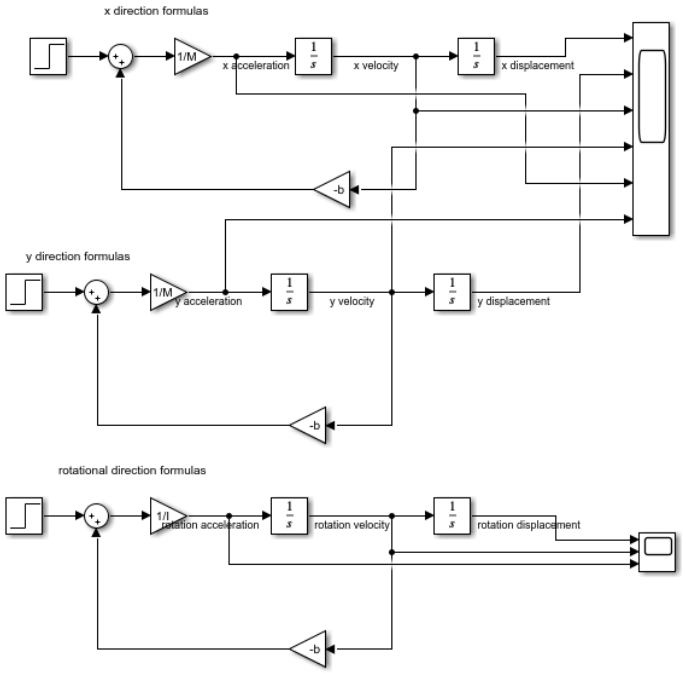

Figure 7: Simulink diagram of the MATLAB simulations

There is however something that is to be noted about this experiment. The fact is that no center of mass was used, meaning that everything was calculated with the center of the hovercraft acting as the center of mass, which is probably not true. This also brings the problem that when applying a force with the propeller, it is being applied to the craft as a whole and not at the back like it should be. This leads to things like the y axis getting overly affected by a force applied with the propeller turned at 90 degrees, while the only relation between this axis and the rotational one is their involvement in the calculation of torque as shown in ****. This makes rotational calculations only a ratio of the ones on the y axis instead of a separate thing entirely.

$$\tau = rF_{\perp} = rF_y = I\ddot{\Psi}$$

Equation 5: Torque equation

Even with these facts, we still went ahead and simulated the behaviour of this system with the propeller aimed at both 0 and 90 degrees to see the speed and maneuverability of the hovercraft when in a straightforward motion and when turning. Knowing what is wrong with the simulation still allowed us to properly analyze what we could from it. The time interval used for the simulations was 60 seconds.

3.2.3.1. Design 1

We simulated the system to get the following graphs with a propeller angle of 0 degrees:

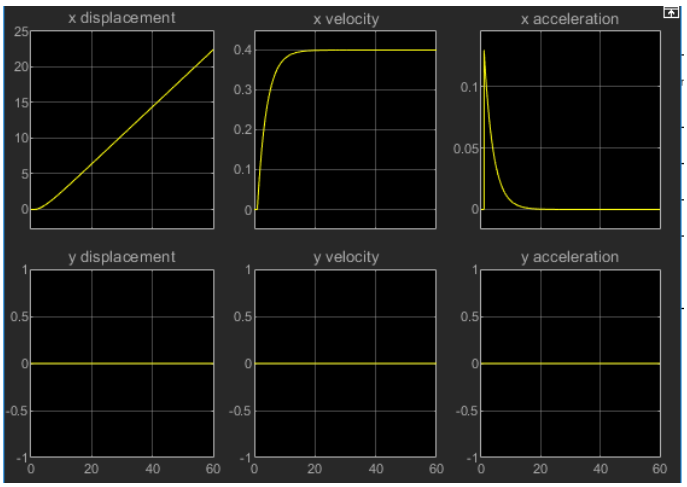

Figure 8: Simulation of design 1 with a propeller angle of 0 degree

It can be seen from the graph that all the thrust force only acts on the forward direction, which is exactly what was wanted. This design has a max acceleration and speed of approximately 0.13 m/s^2 and 0.4 m/s respectively, which provides a total forward displacement of about 22 degrees in one minute. Since r and y are linked in this experiment, all r calculations are also null. When turning the propeller at 90 degrees, the simulation gives the following results:

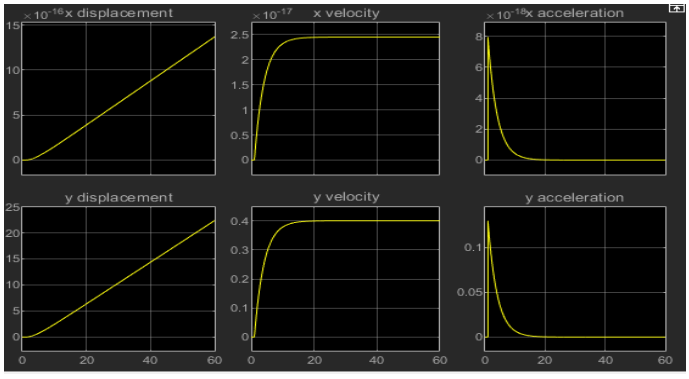

Figure 9: Simulation of design 1 with a propeller angle of 90 degrees

The results on the x axis of this simulation can be misleading since there is not supposed to be any forward displacement when aiming the propeller at that angle. However, the scale is so small and close to 0 that it won't have any impact on the system's movements. The y results are basically the same as the ones on the x axis with the last propeller angle because of our

experiment's flaw mentioned previously. On the other hand, we also get rotational values as in the following figure:

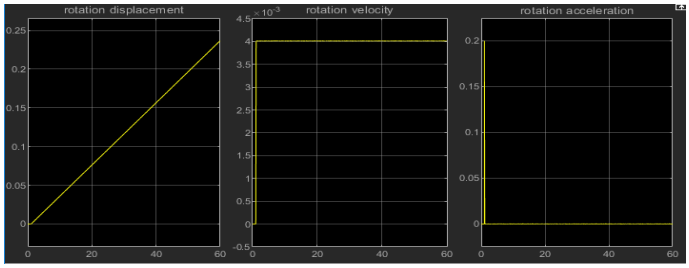

Figure 10: Rotational results of design 1 with a propeller angle of 90 degrees

For now, we cannot use these results in any way, but we can still compare them with the rotational values of the other designs. This will allow us to see how the ratio changes between y and r for each design and get an idea of how efficiently they handle turns compared to the others.

3.2.3.2. Design 2

Design 2 has less friction, which means maneuverability should increase in all ways even with the added weight of a fan, as seen with the following graphs showing the results of putting the propeller at a null angle:

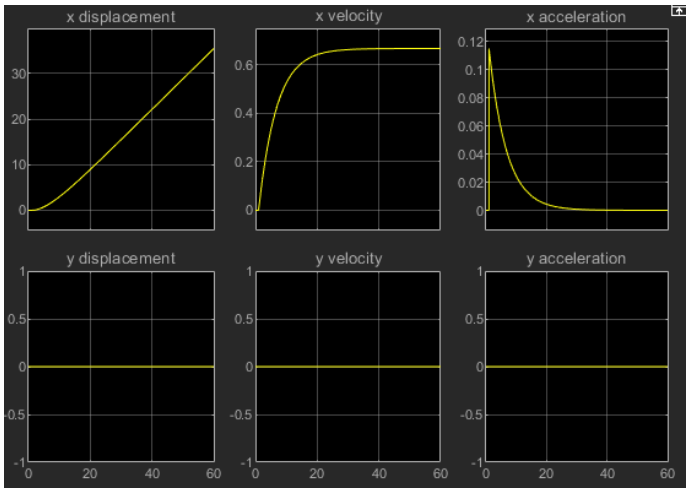

Figure 11: Simulation of design 2 with a propeller angle of 0 degree

This design being lighter, the results show that it has greater maximum velocity and acceleration, resulting in greater distance traveled, and less steeply decreasing acceleration. The increase in maneuverability is also found when plotting the results when the propeller is aimed at 90 degrees:

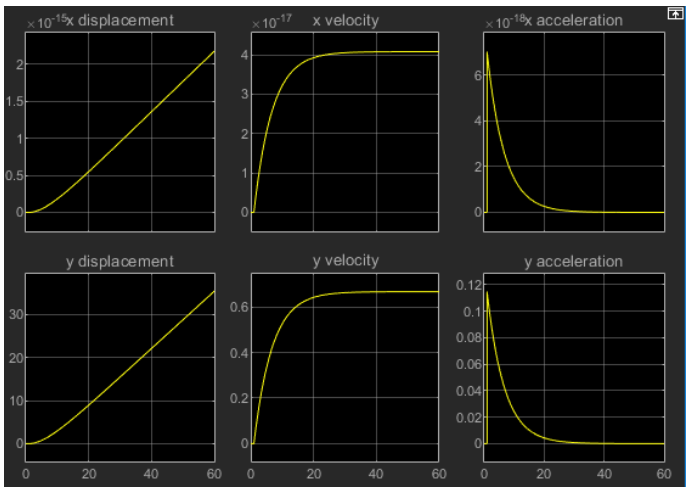

Figure 12: Simulation of design 2 with a propeller angle of 90 degrees

It can be seen that x displacement is also more affected than on the earlier designs, but the magnitude is still so small that it doesn't have any effect. The more interesting part after comparing between designs the ratio r shares with y. For this, the following graphs are needed:

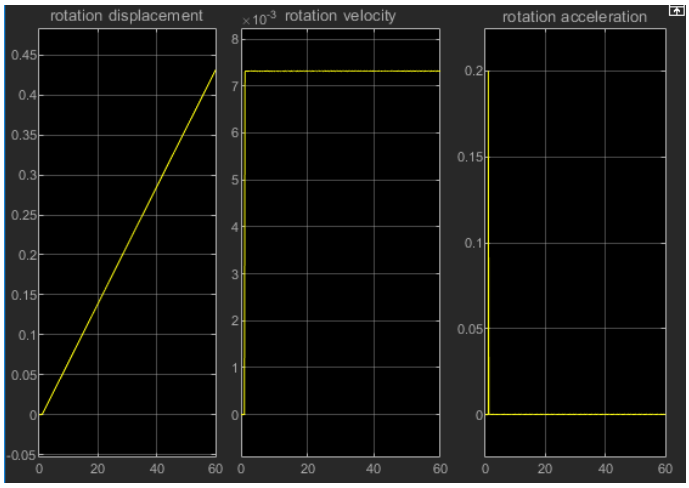

Figure 13: Rotational results of design 2 with a propeller angle of 90 degrees

What can be seen from this figure is that even though the maximum angular displacement and velocity greatly increase compared to the other designs, the maximum acceleration doesn't. Moreover, we can compare the increase in y and r displacements from the other designs. From design 1, the maximum y displacement in design 2 increases by an approximate ratio of 1.59 while the same ratio for r is 1.83. This means that even though we can't really see the exact way rotation will be affected on the design with this experiment, we can still assume that a thrust at

a right angle will affect the turning of this design way more than it affects design 1. This could be due to the rotational values having a greater ratio with the y axis values in their relation formed by the torque equation. On the other hand, that much speed isn't required in a small track like that and it might very well cause the autonomous craft to get stuck somewhere.

3.2.3.3. Design 3

This design has less weight on the front of the craft, so mass and inertia should decrease from other designs. This produces the following simulation results with a propeller angle of 0 degrees:

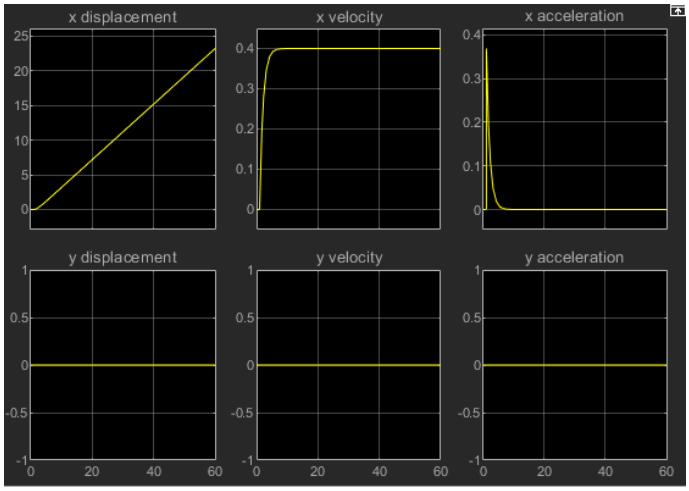

Figure 14: Simulation of design 3 with a propeller angle of 0 degree

These results are quite similar to the ones from design 1, except that acceleration is way steeper, meaning it will be able to travel more distance in total (approximately 23 meters instead of 22 in this simulation) and get back to cruising speed faster after a turn on the track. When putting the propeller at a 90-degrees angle, the simulation produces the following results:

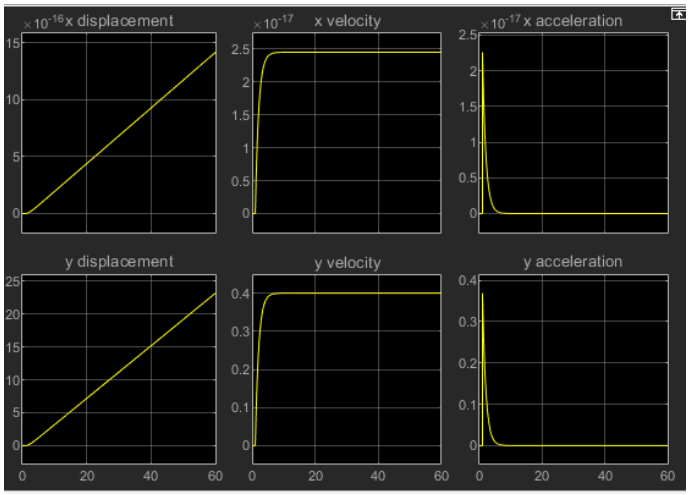

Figure 15: Simulation of design 3 with a propeller angle of 90 degree

As expected from what has been noted from previous designs, the force is essentially applied to y instead of r. Also, just like with the straightforward simulation, this one shows that the acceleration is greater. We can see how this translates to turning with the following rotational results:

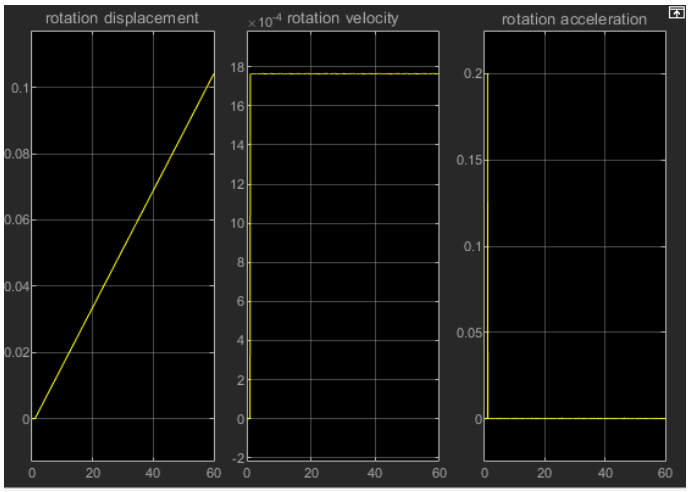

Figure 16: Rotational results of design 3 with a propeller angle of 90 degrees

These results show that rotations on this design are smoother than for all other designs. This makes sense, since their bump in the front give more impact to the force applied in the back. The ratio of increase in y distance from design 1 is close to 1 (1.045), while the same ratio for r is approximately 0.4468, meaning the force applied at a right angle doesn't translate as much into turning.

3.3. Selection Process

3.3.1. SWOT Analysis

The following table lists the strengths, weaknesses, opportunities, and threats of each design prototype based on the goal of the competition.

Design	Strengths	Weaknesses	Opportunity	Threats
Design 1 (bump, 1 lift fan)	-Quick rotational movements	-Increased friction -Propulsion might not be enough due to friction	-3 trial runs to complete the track -Turns can be handled quickly	-Failure of the autonomous system -Breakdown of electrical components -Inability to finish the track due to lack of lifting power
Design 2 (bump, 2 lift fans)	-Incredible maneuverability due to more lifting power -Very quick linear and rotational movements	-More components -Might be hard to control	-3 trial runs to complete the track -If controlled well, could complete the track very quickly	-Failure of the autonomous system -Breakdown of electrical components -Might not finish the track if not controlled properly
Design 3 (no bump)	-Smoother maneuverability -Good acceleration -Easily controllable	-Rotational movements might be too slow	-3 trial runs to complete the track -Very stable and easy to implement design	-Failure of the autonomous system -Breakdown of electrical components

3.3.2. AHP analysis

3.3.2.1. Characteristics and Their Weight

Since there is an equation that will be used to score our performance on the track, we decided to use its variables as our characteristics to be able to maximize the score obtained. The equation is as follow:

$$Score = \frac{D}{N \times T}$$

Equation 6: Score to be obtained on the course

In this equation, D is the distance completed, N is the number of components used and T is the time taken to complete the course. There are a few points to be noted, however. Most importantly, we have 3 attempts to complete the track in 60 seconds and the best time is taken out of all the completed attempts. If there is no completion at all, a score of 0 will be given for not meeting the objective. This brought us to two ways to analyze each variable's weight: mathematically and in terms of completion. We broke them down into the multiple impacts they can have and assigned them weight accordingly.

Starting with D, it is obvious that it is the most important one, since not completing the track results in a score of 0. However, we cannot give it a maximum weight score over all the other variables because of two things. Firstly, it is the numerator of the equation, meaning that it mathematically has less impact. Secondly, when completing the course, D is capped at the overall track's length, meaning that even though completing the objective is the main goal, there is no way to maximize this variable more than that by modifying the design: it is either 0 or the track's length, no matter what we do.

Next variable we looked at was T. This one is also a little bit tricky to analyze because its only impact is whenever you complete the course. However, it is part of the denominator and thus has more impact. Moreover, unlike N it is not a constant, meaning that if two designs were to use the same number of components, the one completing the track in less time would get a better score. However, it doesn't work the other way around: we cannot say that a design completing the track in the same amount of time than another one with less component would get a better score, since time is volatile and out of our control while the number of components

isn't. Even though this is a sentence that can be grammatically said and is objectively true, the implicit properties of both T and N don't allow to compare two different designs that would take the same time to complete the track, since the chances of that happening are slim compared to two designs with the same number of components completing it with different times.

Considering all those properties, the reciprocal matrix for the goal's AHP characteristics and its normalized form are as follows:

	D	N	T
D	1	5	3
N	1/5	1	1/3
T	1/3	3	1
Total	1 1/2	9	4 1/3

Table 5: Reciprocal matrix with respect to the goal

	D	N	T	Total	Weight
D	0.65	0.56	0.69	1.90	0.633346
N	0.13	0.11	0.08	0.32	0.106156
T	0.22	0.33	0.23	0.78	0.260498
Total	1	1	1	3.00	1.00

Table 6: Normalized reciprocal matrix with respect to the goal

This matrix gives a consistency index (CI) of 0.03, which paired with a random index (RI) when n=3 of 0.58 gives a consistency ratio (CR) of 0.05. This CR is smaller than 0.1 so we can state that these priorities are consistent enough within the margin of 10% of consistency to be considered viable. According to these results, D is more important than T, which in turn is more important than N.

3.3.2.2. Distance

When looking at each design's relationship with the distance completed, we did so thinking mostly about if the model could complete the course, but also about the fact that if we weren't sure that it would, we could maximize our chances by comparing which one could go the furthest. We never thought about the time it would take to complete the course in this one. This is also

due to the explicit meaning of distance traveled: we took it literally and tried to figure out, based on the definition, which design would go the furthest.

Keeping that in mind, design 1's main problem was that we weren't sure if it could actually overcome the growing obstacles in the track, so that one got the worst score of all. Since design 3 essentially has the same maneuverability but does not have the risk of not completing the track, we gave it a higher score. Design 2 was a little bit tricky to analyze, because its movements, including speed and acceleration, are faster than the others and it shouldn't get stuck on an obstacle, so of course if everything goes well it should definitely complete the track. If not, it should at least travel more distance than the others. However, specifically because of its faster movements, it makes it more volatile and subject to possibly get stuck somewhere, either spinning around or not able to get out of a certain point. Because of that, we couldn't give it a better score than design 3, which also has better acceleration than design 1, giving it a better score because of its ability to get out of turns. Considering all this, following are the reciprocal matrix and its normalized version:

	Design 1	Design 2	Design 3
Design 1	1	1/3	1/5
Design 2	3	1	1/3
Design 3	5	3	1
Total	9	4 1/3	1 1/2

Table 7: Reciprocal matrix with respect to the distance

	Design 1	Design 2	Design 3	Total	Weight
Design 1	0.11	0.08	0.13	0.32	0.106156324
Design 2	0.33	0.23	0.22	0.78	0.260497956
Design 3	0.56	0.69	0.65	1.90	0.63334572
Total	1	1	1	3.00	1.00

Table 8: Normalized reciprocal matrix with respect to the distance

The proportions of the scores in this matrix are the same as the ones in the characteristics' scores with respect to the goal, just in a different order. This means that is also has a CR of 0.05 and the

consistency is then acceptable. We can see from the results that we prioritized the fact that design 2's volatility might be a problem in a track where this kind of speed isn't required.

3.3.2.3. Time

To compare designs with respect to the time taken to complete the course, we needed to look at it as if every design would cross the finish line. Even if this could be false, it was still the only way to look at it, otherwise we would have designs getting poor scores based on more than just this characteristic.

This comparison was in the end very easy to make. We looked at the graphs generated by our simulations and scored the designs based on their speed and maneuverability. The only twist was that since design 2 could potentially get stuck somewhere, we took it into consideration and assumed that since we needed it to complete the track for this comparison to work, it would succeed into getting out of this situation and move on to the finish line. However, this would still take some time off its run and that is why, even with all its speed and maneuverability, we didn't give it as much of a high score as we could have without this in mind. The results of this comparison are contained in the following reciprocal matrix and its normalized version:

	Design 1	Design 2	Design 3
Design 1	1	1/3	1
Design 2	3	1	3
Design 3	1	1/3	1
Total	5	1 2/3	5

Table 9: Reciprocal matrix with respect to the time

	Design 1	Design 2	Design 3	Total	Weight
Design 1	0.20	0.20	0.20	0.60	0.2
Design 2	0.60	0.60	0.60	1.80	0.6
Design 3	0.20	0.20	0.20	0.60	0.2
Total	1	1	1	3.00	1.00

Table 10: Normalized reciprocal matrix with respect to the time

This matrix's CI is 3, which in turn gives a CR of 0. This is to be expected, since the proportion of the scores given is equal for all designs, meaning that the consistency is perfect. We can also see that design 2, even though it is very volatile, could potentially finish faster if it were to finish.

3.3.2.4. Number of Components

This comparison was very straightforward: the number of components is constant for each design and it is known. Of course, we could end up modifying it throughout the next stages of the design and making process, but for now all our prototypes have a constant number of components and all we had to do is give them a score accordingly. We didn't give scores that varied greatly, since in all three designs, there is only a difference of one component between the smallest and largest number of them. This forms the following reciprocal matrix and its normalized version:

	Design 1	Design 2	Design 3
Design 1	1	3	1
Design 2	1/3	1	1/3
Design 3	1	3	1
Total	2 1/3	7	2 1/3

Table 11: Reciprocal matrix with respect to the number of components

	Design 1	Design 2	Design 3	Total	Weight
Design 1	0.43	0.43	0.43	1.29	0.428571
Design 2	0.14	0.14	0.14	0.43	0.142857
Design 3	0.43	0.43	0.43	1.29	0.428571
Total	1	1	1	3.00	1.00

Table 12: Normalized reciprocal matrix with respect to the number of components

Just like with the matrix with respect to the distance, the proportion of the scores given is equal for all designs, meaning the comparison has a perfect CR of 0. Other than that, we can see that since designs 1 and 3 both have one less component than design 2, they lead neck to neck in weight.

3.3.2.5. Weighted Results

To choose the prototype that is to be used as the main competing design, we assembled all the respective priorities of the designs into their composite weight with respect to the goal. This gives the following table:

	D	N	T	Composite Weight
Weight	0.63334572	0.106156324	0.260497956	
Design 1	0.106156324	0.428571429	0.2	0.164828812
Design 2	0.260497956	0.142857143	0.6	0.336449228
Design 3	0.63334572	0.428571429	0.2	0.49872196

Table 13: Composite weight of all designs with respect to the goal

3.3.3. Final Selection

From the final AHP table, we can see that design 3 is the one that is to be considered according to our priorities in achieving the goal. Because of its stability, unlike design 2, this design will be the best to not only complete the track but also to do it in the least amount of time possible, unlike design 1 that might not be able to surmount the obstacles. This means that one of the goals in the testing phase will be to optimize its sense of direction so that it doesn't happen.

4. Simulation and Implementation

4.1. Functioning of the Simulation Environment

The project requires us to make a working model for both types of sensors, which are proximity and vision. The simulation environment, CoppeliaSim, uses the child script of a microcontroller to send data to either an Arduino board or a remote API (RAPI) written in either MATLAB or Python. We decided to use Python. The controller's child script is written in Lua.

At the beginning of the simulation, the initialization function in the child script as well as the setup function in the remote API are ran. After this, the simulation runs 3 functions at each iteration of 50 ms, which are the actuation and sensing functions in the child script and the loop function in the RAPI. At first, it reads data from the sensors then sends it to the RAPI. The latter uses the data in the algorithm and send the results back in the form of control variables for the motors and fans. The child script finally actuates the values and waits until the next iteration in the simulation.

4.2. Proximity Sensors

4.2.1. Version 1

The first version of our implementation of the hovercraft design in CoppeliaSim was about making it lift and giving it thrusting capabilities. The material used for the construction of the body would be Styrofoam, which usually has a density between 25 to 50 kg/m^3. To maximize the efficiency of our model, we decided to go with the lowest possible value, which is 25 kg/m^3. To implement it into the software, we needed to create the body as well as a thin layer that would go below it and for which the friction would change depending on the lifting capability. This part is called the contact patch and is used as the parent object of the whole structure, allowing the body to be attached to it as a child object by a force sensor to act as a body joint. The rest of the components are children of the body itself, except the propeller which is a child of the servomotor. The first implementation of our design is shown in the following figure:

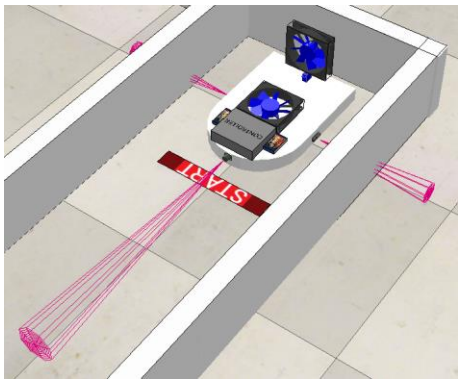

Figure 17: First implementation of the design with proximity sensors

As shown in the figure, there are two sensors at the sides to sense their respective walls and one at the front with a bigger range. The idea was that the side sensors would allow for stabilization by detecting where the hovercraft is positioned from the sides, while the front sensor would be able to detect when to start slowing down. The side sensors would also indicate when to turn, since if one didn't see a wall, it would mean that there is space on that side to make a turn. To make it run, the only code we had was a driver test that would make the hovercraft lift and go forward, nothing else.

4.2.2. Version 2

Before starting to make the algorithm to complete the track, our main concern after making it lift and propel forward was that the center of mass was misaligned by 7.43 mm, making it turn harshly to the left and straight into the wall. There was also the problem that even thought having two batteries would help balance out the weight once aligned well, they would still count as multiple components and thus make us loose more points in the competition. After many manipulations, we managed to move around the components in a way that would stabilize the hovercraft a lot more, giving a center of mass misaligned by only 1.34 mm. This meant putting the propeller in the front and the other components in the back while leaving the fan in the center of everything. We even managed to only take one more powerful battery instead of two smaller ones, as shown in the following figure:

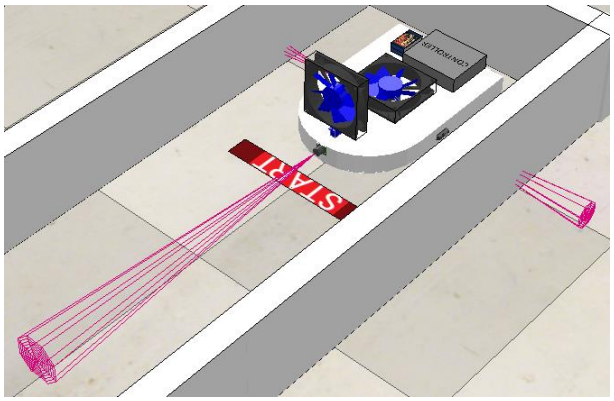

Figure 18: Second implementation of the design with proximity sensors

The main problem we have had with this version was that depending on the iteration of the simulation, it was either correcting itself to the right and even going further than just centered, or it was going straightforward even though the propeller was not pointing that way. We thought that it was not normal, but also that we could be able to fix it later on with the algorithm by adjusting parameters.

4.2.3. Version 3

We started making the algorithm in this version. While doing so, we realized that the side sensors wouldn't be able to stabilize the hovercraft properly because of their positioning: by being in the middle of each side, if the hovercraft was in the middle of the track and pointing in diagonal, the sensors would still read the same values because they would be on the same axis as the center of the structure. To make sure we could read different data, we moved them to the front of the hovercraft. This was also meant to facilitate the turns paired with our propeller being in the front: the sensors would be able to sense sooner when to turn and give more time to the algorithm to react accordingly. This version of the implementation is shown in the following figure:

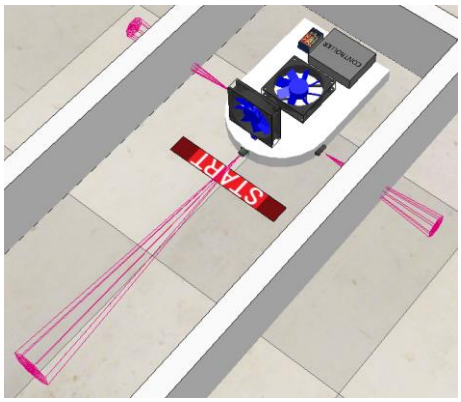

Figure 19: Third implementation of the design with proximity sensors

Our main goal during this implementation was to be able to stabilize the hovercraft when it was going forward, we didn't care about it making the turns yet. We quickly realized that the torque generated by the lift fan would pose a real challenge, as the craft was turning left way quicker than it was turning right. However, we were convinced that by modifying a few parameters, we could make it work and move on to the next version of the design afterwards.

4.2.4. Version 4

This version of the hovercraft contained what we thought to be all the possible scenarios of positioning. The design was the same as in the previous design, we only implemented more code into it. That being said, it still had a lot of difficulty stabilizing itself. Indeed, even when going in a straight line, it is travelling while facing in diagonal and the propeller barely manages to counteract the torque generated by the lift fan. This issue was even more relevant whenever it was trying turn even the slightest, as turning right was quite the struggle while left turns were too quick and made the craft misalign so much that it would think it was in a new turn. To move forward with our algorithm, we would have to be able to keep the hovercraft aligned and centered to make it turn at the right times.

4.2.5. Version 5

After seeing how the lift fan's torque from the previous version of the design was way harder to control than we thought, we had to rethink our approach to the whole design. We realized a few points while doing so:

1. There is no limit to how many times the hovercraft can touch the walls or bump into them.

2. The hovercraft only needs to complete the given track. What it would do in other situations is irrelevant.
3. Controlling the torque generated by the lift fan is way harder than anticipated.

Those three statements were the base of our thinking for this final design and implementation. The idea is that if the hovercraft can keep its direction while not having to worry about it being changed by the torque from the lift fan, the code would be significantly simpler to implement. Since we knew the dimensions of the track, we thought that it could control the direction for us while we would but all the power available from the propeller into thrust force. This required a design that would be large enough so that the maximum angle that would be generated by the lift fan's torque would be limited by the walls of the track. However, it would need to be small enough for it to still be able to go forward without its velocity being limited too much by sticking to the walls. This gave the following final design:

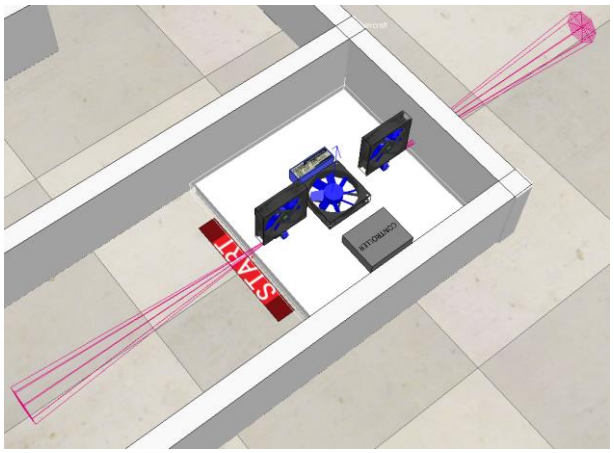

Figure 20: Fifth and final implementation of the design with proximity sensors

Viewed from the top, the body is a 45 cm square with the same density as our previous designs. This gives only a few centimeters on each side for it to have it turn a little bit sideways and only have its corners touching the walls. Because of its size and the fact that its velocity is still being limited by its contact with the walls, it needs two propellers. Since the hovercraft only needs to go in 3 directions, we put the servomotors facing towards the other lanes when it's at the start. This allows it to go forward, translate into the next lane, then go backwards, translate again and so on.

While trying to calibrate the center of mass, we realized that having it perfectly centered didn't matter, since as long as we had something that wasn't too big, the walls would retain the effect of its displacement. We then left it as 3.34 mm. We also realized when writing the algorithm that since the sensors were low-budget ones, they couldn't just be put on the front side of the hovercraft: they can detect anything if an object is too close. There is also the problem that when bumping into a wall, there would be a change that in a real-life situation, they would break. For the purpose of this simulation and because we ran out of time to put a small block at the front and back of the hovercraft the screw them on, we just placed them on the servomotors for now. If we had had more time, we would have added those blocks and put the sensors on them instead. The only thing that needs to be modified after this is the sensor distance in the code that makes the hovercraft stop going forward or backwards and start translating. One great thing about this design is also that it can be implemented with virtually no sensors by only using time.

This design is very effective: it completes the track in under 40 seconds. The code was extremely simple to implement since there is no stabilization to it. The only downside to it is that it is only useful for this exact track and nothing else. As soon as something changes, the design and code needs to be reimplemented. For example, changing the space between the walls would mean changing the size of the hovercraft and there is the possibility that it might not work if it needs to be too big. In the same way, adding length between lanes would mean changing the code, since it uses the simulation time to see when to exit the translation. This, however, allowed us to use less sensors and thus maximize our result during the competition. On the other hand, one great thing about this design is that it doesn't need any torque, surge, sway or yaw analysis, as those values don't matter.

4.2.6. Progress Logs

At the end of each version, we recorded our progress and uploaded them as unlisted videos on YouTube available only with the links, which are provided below:

- Version 1: https://youtu.be/xCG1uZlI_0k
- Version 2: https://youtu.be/v4GNxJTjfSM
- Version 3: https://youtu.be/Fj1ojgM0a1Q

- Version 4: https://youtu.be/MtEyoBGClEo
- Version 5: https://youtu.be/D5aKYcZU1bU
- Final Simulation: https://youtu.be/7jtDtOypKl8

4.3. Vision Sensors

After getting the final version for the proximity sensors, the team started switching them for visions sensors to implement the second required scenario. However, no progress has been achieved on this, since the hovercraft itself doesn't move at all. We have tried using the code provided in the example on vision control to manage the data from the sensors and determine the control values accordingly. Our hypothesis is that since the code for managing the data doesn't work, the control values revert to their default value, which is 0 for all of them. The following figure shows the concept that the team tried implanting. It uses the same structure as in the version with proximity sensors.

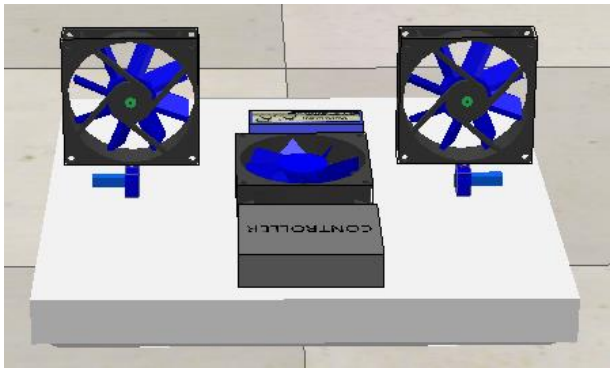

Figure 21: Implementation of the vision sensors

4.3.1. Progress logs

Since no progress has been achieved in the scenario with the vision sensors, the team has decided that no video needed to be recorded. Otherwise, it would have only shown a grounded hovercraft that doesn't move.

5. Progression Schedule

5.1. Desired Project Progression

5.1.1. PDR phase

Based on the tasks to be done during the project, the Gantt schedule in Figure 22 has been formed from breaking down these main tasks into smaller ones and assigning them to team members. The timing on this schedule has been decided at the beginning of the project from predictions on the amount of time each step had been thought to require.

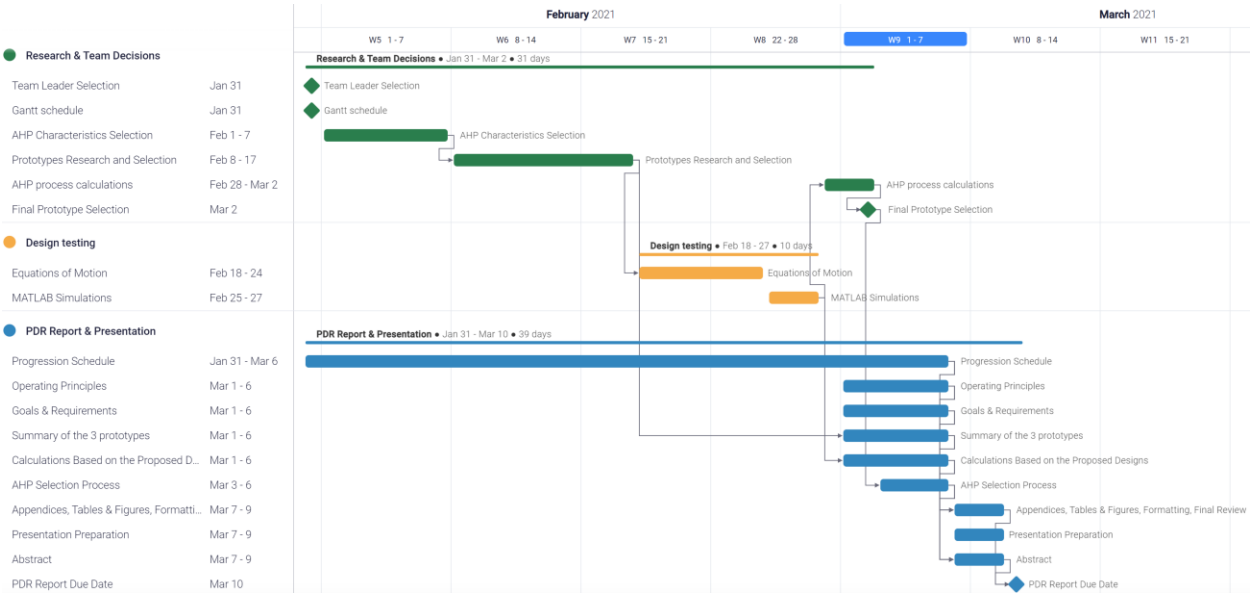

Figure 22: Initial PDR Gantt Schedule

5.1.2. Simulation and Implementation Phase

This phase was more about making our design work than the actual report itself, making it more practice-focus. The following Gantt schedule shows the predefined allowed time for each task at the beginning of the phase:

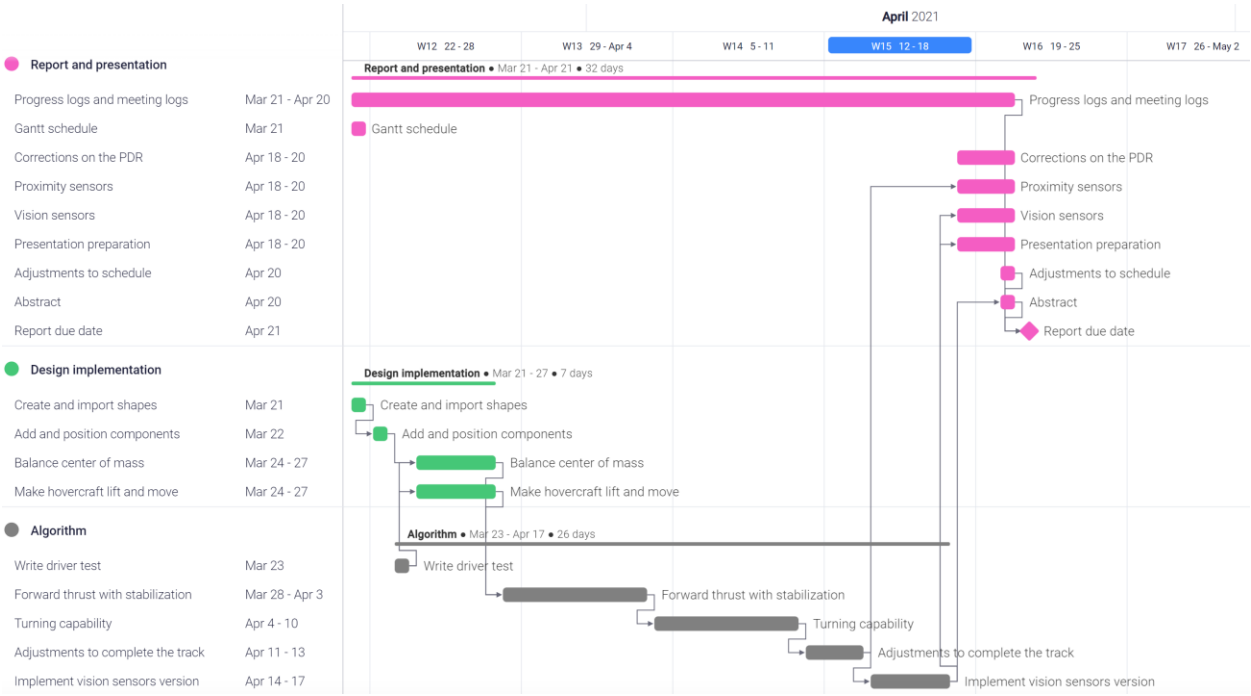

Figure 23: Initial Simulation and Implementation Gantt schedule

5.2. Necessary Adjustments

5.2.1. PDR Phase

The original schedule was made with having in mind only the time required for each step. Approximately at the end of February, the team had to readjust our schedule accordingly because all the members had been busy with other classes in the middle of the semester. The actual schedule went as in the following figure:

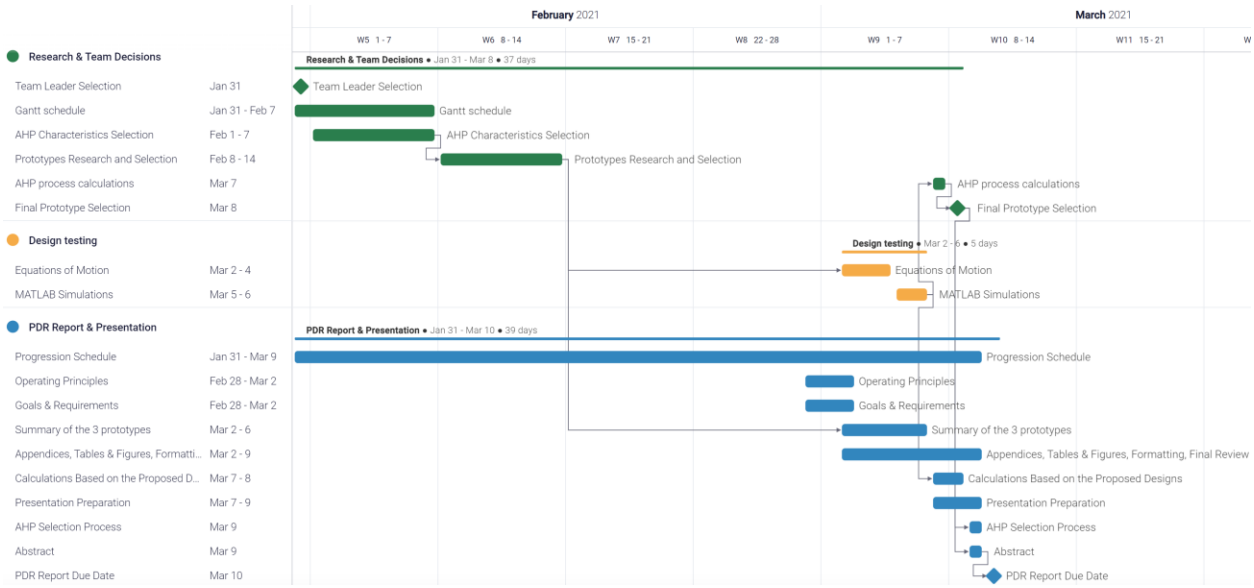

Figure 24: Adjusted PDR schedule

5.2.2. Simulation and Implementation Phase

For the simulation and implementation phase, the team has had a lot of difficulty understanding the code implementation and the relation between the simulation environment and the remote API. The report due date also changed and for these reasons, adjustments were necessary, as shown in the following Gantt schedule:

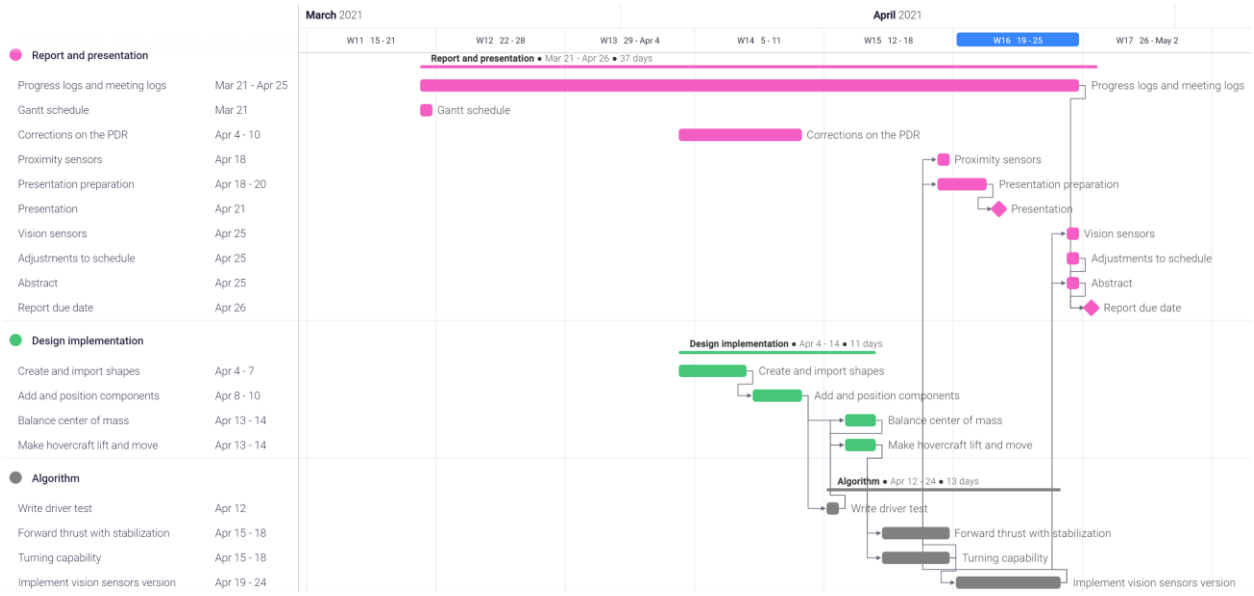

Figure 25: Adjusted Simulation and Implementation schedule

5.3. Meeting Logs

5.3.1. January 31st, 2021

Discussed:

- Designated a team leader (Alexandre)
- Chose a time in the week to meet if a second meeting is necessary (Wednesdays)
- Decided on questions to ask the teacher
- Made a Google doc to insert all ideas
- Decided briefly on the draft of the Gantt schedule

For next meeting (February 3rd, 2021):

- Think about how to separate the characteristics used in the AHP method
- Ask questions to the professor during office hours

For future meeting (February 7th, 2021):

- Do research on different HC models

5.3.2. February 3rd, 2021

Discussed:

- Decided on the AHP characteristics
- Formatted the Google doc to separate each section as per the report outline

For next meeting (February 7th, 2021):

- Do research on different HC models
- Ask questions to the professor during office hours

5.3.3. February 7th, 2021

Discussed:

- Reworked the importance of each of the AHP characteristics
- Finalized the Gantt schedule

For next meeting (February 14th, 2021):

- Do research on different HC models

5.3.4. February 14th, 2021

Discussed:

- Chose the 3 different designs for the AHP

For next meeting (February 21st, 2021):

- Research on how to use MATLAB for the PDR
- Ask questions to the professor during office hours

5.3.5. February 28th, 2021

Discussed:

- Procedure for the next week

For next meeting (March 1st, 2021):

- Rifadul and Ravjotdeep will work on the operating principles part of the introduction
- Antonio will work on the objectives and requirements
- Alexandre will work on the AHP characteristics description
- Saro will continue researching on the MATLAB simulations

5.3.6. March 2nd, 2021

Discussed:

- Revised the overall distribution of tasks

For next meeting (Undetermined):

- Saro and Antonio will work on the equations of motion and implement everything in MATLAB
- Alexandre will start transferring the report into a Word document for proper formatting and review
- Antonio, Rifadul and Ravjotdeep will work on the summary of the 3 designs

5.3.7. March 7th, 2021

Discussed:

- Completed the simulations and discussed about the results
- Decided on the AHP values for each design

For next meeting (March 8th, 2021):

- Antonio will provide clearer drawings of each design to insert into the report
- Antonio, Rifadul and Ravjotdeep will work on the presentation
- Alexandre will calculate the AHP weights and finish the AHP section of the report
- Saro will work on the calculations and simulations part of the report

For March 9th, 2021:

- Alexandre will review the calculations and simulations part of the report, insert the design drawings and write the abstract
- The rest of the team will review the final version and give feedback as soon as possible in case there are changes to be made

For March 10th, 2021:

- Alexandre will submit the PDR report

5.3.8. March 9th, 2021

Discussed:

- Everyone has their parts for the presentation
- Alexandre submitted the report

5.3.9. March 21st, 2021

Discussed:

- Defined the tasks the were to be done for the remaining of the project

For next meeting (March 30th, 2021):

- Ravjotdeep will work on building the course environment as well as helping Saro and Rifadul with the building of the hovercraft on CoppeliaSim
- Saro, Rifadul and Antonio will research on how to implement the components and their behaviour in CoppeliaSim
- Alexandre will work on the new and updated Gantt chart

5.3.10. April 4th, 2021

Discussed:

- What needs to be changed from the PDR report
- First simulation needs to be done as fast as possible

For next meeting (April 11th, 2021):

- Alexandre will rework the current PDR report and set up the YouTube account for the videos
- Saro, Ravjotdeep, Rifadul and Antonio will work on the first simulation

5.3.11. April 11th, 2021

Discussed:

- Shapes and components have been implemented
- Functions don't seem to work in the RAPI environment

For April 12th, 2021:

- Alexandre will ask questions to the TA in the tutorial

For next meeting (April 18th, 2021):

- A working version of the proximity sensors needs to be implemented
- When working on the project, members will be invited to join a Discord session, not as a meeting but as a work room environment with screen and idea sharing
- Videos will be recorded showing the progress throughout the implementation process

5.3.12. April 12th, 2021

Discussed:

- Alexandre explained his findings with the TA's answers
- The CoppeliaSim controller has been programmed to interact with the RAPI

For next meeting (April 18th, 2021):

- A working version of the proximity sensors needs to be implemented
    - A test bench needs to be implemented for the center of mass and lifting/thrusting
    - Forward movement
    - Turning

5.3.13. April 17th, 2021

Discussed:

- The implementation process is not going as planned, the approach to the design needs to be rethought all over for something simpler
- A new design has been thought about, which should be relatively simple to implement
- Saro has implemented the shapes of the new design into CoppeliaSim
- Alexandre has modified the placement of the components and tested the center of mass and lifting

For next meeting (April 18th, 2021):

- The new design needs to complete the track, Alexandre will implement the code

5.3.14. April 18th, 2021

Discussed:

- The new design completes the track, and everything has been recorded
- Alexandre has written the parts in the report about the different versions of the implementation
- The due date for the report has been changed to April 25th

For next meeting (April 24th, 2021):

- Rifadul, Ravjotdeep and Antonio will work on a version of the HC with vision sensors

For April 20th, 2021:

- Rabjotdeep and Antonio will work on the presentation PDF and Alexandre will review and adapt it to what he will say

5.3.15. April 24th, 2021

Discussed:

- The HC in the implementation of the vision sensors doesn't move at all, so the team decided to not record any progress, since there isn't any to show

For April 25th, 2021:

- Alexandre will complete the few parts left on the report and the team will review it

For April 26th, 2021:

- Alexandre will submit the report and the simulation files

6. Conclusion

Even with a few problems along the way, we still managed to make at least one off the two scenarios complete the track as required in the project goals. However, a few things could have been done better. For example, the MATLAB simulations didn't take into account the center of mass, so the results had to be interpreted in a different way when simulating a 90-degree force on the hovercraft. As for the design that completed the track, a different positioning of the components could be used to get the maximum out of the forces applied by the propellers.

In our design, the propellers loose of their efficiency when going forward in return of more power for translating from lane to lane. Our reasoning was that the hovercraft was sliding against the wall during the translations, which would create a lot more friction than when moving in the lanes. For this reason, we decided to put the propellers in the front and back of the craft so that both would contribute to translational movement at their maximum efficiency. However, this means that having one propeller push its air into the other one during forward and backward movement diminishes their efficiency, which in turn made passing through the obstacles more of a challenge.

Having a better positioning of propellers could be a possible future implementation for better performance. The other components could also be placed in another way so that the center of mass would be as close as possible to 0, because its current value of 3.34 mm could possibly still impair the craft's lifting and stabilizing performances.

7. Appendix A: References

[1] "Propeller," Encyclopaedia Britannica, [Online]. Available: https://www.britannica.com/technology/propeller. [Accessed 1 03 2021].

[2] N. Hall, "Propeller Propulsion," NASA Glenn Research Center, 05 05 2015. [Online]. Available: https://www.grc.nasa.gov/WWW/K-12/airplane/propeller.html. [Accessed 01 03 2021].

[3] C. Woodford, "Hovercraft," Explain That Stuff!, 19 03 2020. [Online]. Available: https://www.explainthatstuff.com/hovercraft.html. [Accessed 01 03 2021].

8. Appendix B: List of figures

9. Appendix C: List of tables

10. Appendix D: List of equations