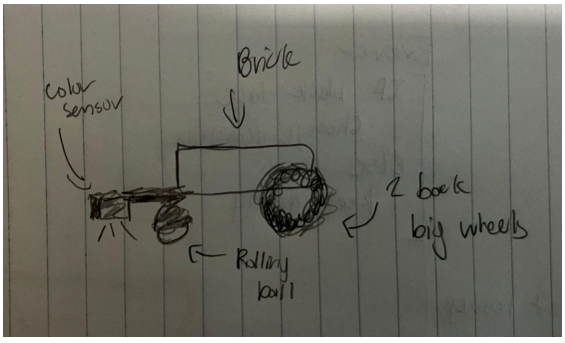
Group: Hotel
Group Members: Aarush Maddula, Kevin Zarate Nonato, Angel Edgardo Aguilar Gonzalez
School Year / Term: 2024 - 2025

# Line Follower Project Report

## Introduction

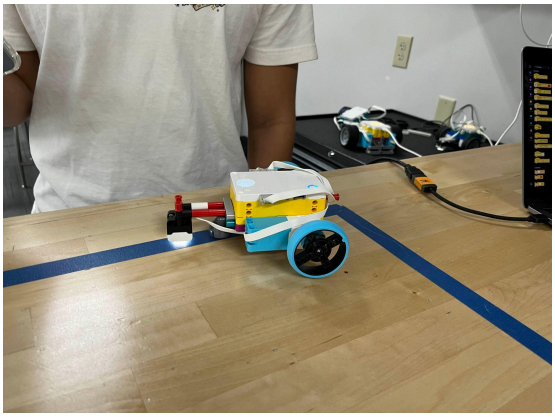
In this project, we (Aarush, Kevin, and Angel) were tasked with creating a Lego robot designed to navigate through a course with blue and green tape marking the course's outer and inner boundaries, respectively. To achieve this, we had to use a color sensor to detect the boundaries and adjust our robot accordingly. This was also the first time we needed to focus heavily on the coding aspect of our robot to utilize the functions of the color sensors. Throughout the process, we raced three times with the other robots our class made to see which robot could make it through the course the fastest.

## Brainstorming




To start, our group briefly discussed our initial design since we wanted to spend as much time on the code as possible to ensure we got the code working. We wanted to make a small, maneuverable robot, so we focused on making a small, compact design. We decided on having 2 medium-sized wheels, each controlled by a small motor, with a rolling ball in the front for extra maneuverability. The color sensor that we used to detect the tape was placed at the very front of the vehicle since we wanted to detect the tape ahead of time so the robot could turn before it went past the tape.

## Design Process



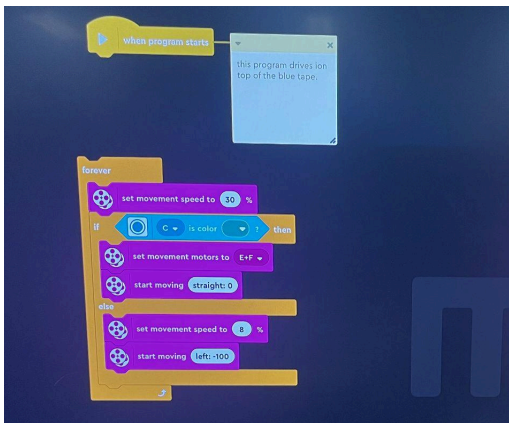
After brainstorming, we built the car as outlined in our brainstorming session, pictured to the right. After testing whether the motors functioned correctly, we started to code the robot.

### Programming

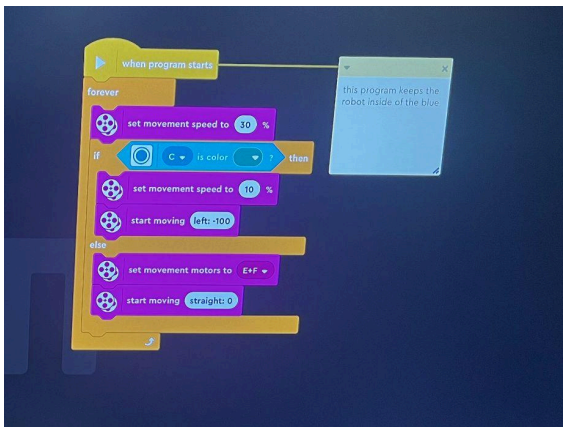
Our table had blue tape in the shape of a rectangle, which allowed us to quickly test and modify our robot instead of going back and forth to the center of the room where the actual course was.

In version 1 of our design, we have it set that when the program starts, the two motors go straight IF the sensor detects “teal blue.” We saw this as something easier to do so we have something to base our final design on. Then it will set the motors to move straight. However if the sensor detects any color other than “teal blue” (goes off the tape), the program is set to turn left at 8% speed (so that it turns slowly and precisely). The entire program is encapsulated by a forever command that makes the robot continuously check whether the robot is on the tape.

In version 2 of our design, we implemented the logic for the robot to stay inside the rectangle instead of driving along the edge. We flipped the logic of Version 1 so that when the color sensor detects the tape, it moves left until the sensor is off the tape. The code for version 2 worked well in our course



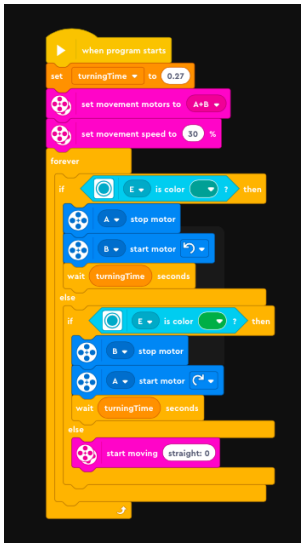

Version 1




Version 2

## Testing on the Track

After testing our car on our table, we moved on to the big track in the center of the room. After adding the code that turned right if the robot detected green tape, our robot adapted well to the track as it was able to navigate through most of the track. However, the robot often went off the edge when it encountered a corner. We thought this was because when the robot turned a corner, the sensor, which was in front of the car, went past the tape before the code to turn the car triggered and didn’t turn enough for the sensor to get back into the track, causing it to fall off the track.




As a temporary solution, we lowered the speed of the robot significantly, which seemed to fix the issue. However, our robot was extremely slow and certainly wasn’t going to beat the other robots, so we needed to come up with a more permanent solution.

On the right was our final code. The program continually checks if it is on the tape. If it isn’t, it goes straight at 100% speed (image has different values for testing). However, if the robot is on the tape,

only one of the wheels turns backward instead of both wheels turning. This change allows the sensor to reliably get back onto the course. Also, the wheels turn for a preset amount of time to ensure the sensor is on the track.

After making these changes and tweaking the speeds and timings, the robot didn't fall off when it encountered a corner.

# Test run #1

The day of the tournament came and we raced our robot with the other robots in the class. Here are our results:

Time: **24.39s**

Place: **1st**

In our first run, our robot crushed the other robots, with 2nd place finishing around 30s and half of the class getting DNF.

# Revisions

Since our robot did well the first time, we were able to freely experiment with iterating the design of the car to make it go faster through the course. Not completely sure what to do, we decided to move the motors from the sides of the Mindstorm brick to the bottom of it. We were not able to complete the design on time as the next race came quicker than expected. Since we were rushing to complete the design, we forgot to take a picture of the design, an oversight on our part.

# Test run #2

The second tournament happened with not as much success as #1 since we didn't finish revising our design in time. Here are our results:

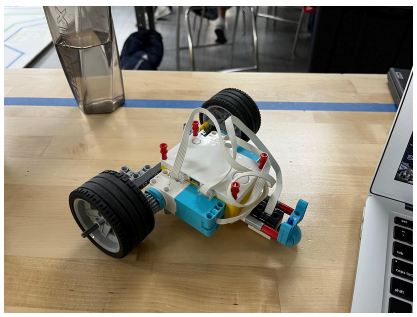
Time: **DNF**

Place: **N/A**

Our sensor wasn't detecting the tape properly as we didn't have time to properly align it and adjust the code, causing the car to go off the course. Overall, we still held the highest time out of all the trials.

# More revisions

After attempting to tweak the previous design, we decided to scrap the previous design to go back to our original design since in that design, the sensor was properly aligned. To make our car faster, we decided to use gear ratios in our design.



We tried using a 40:8 gear ratio, but we quickly realized that it was too fast for the sensor to react to the tape. We switched gears to a 24:16 ratio to give it a little boost in comparison to the competition. We also increased the size of the wheels to increase the speed of the car.

We realized we were running out of time until the third competition. We tried running the robot with the old code, but it didn't work since the faster speeds required us to tweak the timings and speeds of the motors. We unfortunately didn't have enough time to fix the code.

# Test Run #3

We raced with other robots in our class for the last time. Here are our results:

Time: **DNF**
Place: **N/A**

Our robot also drove off the edge since the car was too fast for the sensor to have enough time to detect and react to the tape.

After some fixes, we got the robot to work.
Video:
https://drive.google.com/file/d/1RkP-NPJ5PaxM8Mf-8BEBad5KdCZGRbjn/view?usp=sharing

# Conclusion

We were successful in our project since we were able to create a line follower robot that uses color sensors and code to navigate through a course the fastest. After creating a small, maneuverable robot, we spent most of our time coding the robot. Initially, we made the robot move along the edge of a test track we made on our table, but after modifying the code, we made it able to stay inside the box. After modifying the code and transitioning to the real course, we discovered that the robot would have trouble on corners, which we fixed by increasing the turning time and making only 1 wheel turn to ensure the sensor was always inside or on the boundaries. We won our first race with 24.39 seconds, unbeaten by any other robot in the classroom since. Since, we've tried to make the robot more compact by moving the motors and scraping that to add gear ratios, which caused us to get a DNF on the other 2 rounds as we didn't finish in time.

In the end, I've learned that having a small and compact car is best for a project like this, as well as having the sensor near the nose of the car is best for having the car turn and adjust in time. Also sacrificing some speed to have better consistent turns is always going to be worth it then risking not finishing the race at all. Also, in our case, having the line follower car along the blue tape is better than just being in the middle, which is best for its turns. Lastly, I've learned not to modify the design significantly before a race.

Next time, I'll make a design and stick to it, only changing it well before a race to ensure we have a quality product. I'll also spend more time optimizing the timings and speeds to make the robot finish as fast as possible.