Final Chapter - Project

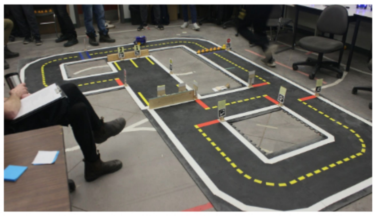


Image 1 - The project circuit

The end of the journey

As for the final task in the course, we were given a project where we had to complete three separate stages, where each stage highlights something we learned throughout the course. The first stage was simply applying lane following through a circular circuit of the track, where each april tag signed represented one of the three actions, left, right or straight. For this part, we already had code from the final lab that worked on that section of the course, but my partner Marcus decided to optimize and improve on that particular lane following code. Which functioned well in our testing, but like with everything, there is always room for fine-tuning.

The second stage was mostly focused on obstacle avoidance; where in this section of the circuit, there were two crosswalks where potential duckie pedestrians code cross at any of the two crosswalks as well as a broken-down robot in between these two crosswalks. This stage was tasked by my partner Marcus. For the crosswalks, he had some concerns about using the color-detection approach due to the fact that the ducks themselves had colors that we were already detecting, not only in our lane following procedure but also in terms of stopping. So initially attempted a different approach utilizing the fact that the pedestrians were rubber ducks. He opted to do SIFT descriptor, which utilizes reference pictures of other rubber ducks to see if what the robot saw matched the features of the reference. This approach didn't work because the images were too small to extract features. Then he tried template matching, but that, too, was computationally expensive, and it was slowing down the controller node. So we instead attempted the easier option of color detection; because if we carefully narrowed the HSV range enough, the robot could probably detect the ducks more accurately and not get affected by any miss detection from any other object.

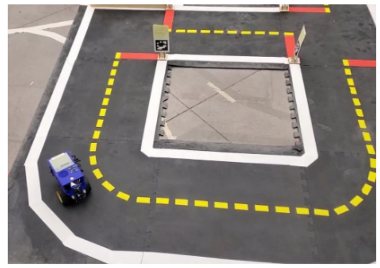
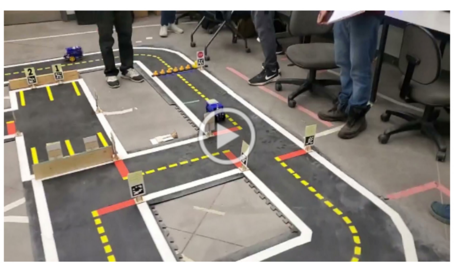
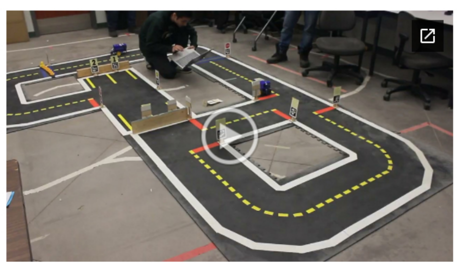


Image 2 - Snapshot of stage one

Next was the robot, and since in Lab 4, my partner knew that the grid sticker wasn't giving that consistent detection to the duckie bot from our implementation, he decided to use the time of flight sensor built into the duckiebot to properly stop within a safe distance from the broken down bot, the strategy we used was consistently based on hard coding our avoidance, English driver to proceed with the course and turning back onto the right lane. All of this occurs within a certain time frame. The final stage of this project was parking the duckie-bot in one of four pre-existing stalls at the end of the circuit, this was my part. For this task, I initially tried to use dead-reckoning based on the april tags that were in the parking lot, such that I could park the duckie-bot going backwards within the parking stall, but then I had to scrap that idea. The reason is that it was of greater relevance to get all the steps of forward parking first and then attempt the backward parking if we still had time to work on that. So instead, my strategy mostly relied on turning the correct amount toward the parking stall so that the lane following detection could kick in so that the duckie-bot would stay within the lines of the parking stall. I would use the april tag detection to stop the bot within the stall. Though by using the april tag detection, I was running through some problems with my implementation because whenever I was triggering the stop either through a threshold distance or threshold area within the frame of the duckie-bots' camera, it wasn't at all consistent, so just like my partner I ended up using the time of flight sensor to measure the distance to the april tag because the measurement was giving me more consistency. Also, my implementation was in a separate node then the rest of the two stages, so we found a much more reasonable solution to close the controller node and initiate the parking. The result of our work is shown below.



Video 1 - First demo attempt



Video 2 - Second demo attempt

Final Remarks for the Course

After a very interesting 3 months of hard work for this class, I can only say that I learned a lot from it, from dead-reckoning, localization, april tag detection, OpenCV, machine learning, state machines, SLAM and a lot of other stuff. This class reignited my interest in robotics again, and I wanted to pursue a career in this field. Also to take a deeper dive into some of the stuff we learned and start my own robotic-related projects. The experiences of triumph and hardship were definitely all worth it in the end. Also, the people I have met through this course were amazing, especially my lab partner Marcus. The TAs were also remarkable people, very insightful and engaging in helping the students learn and succeed. This course also helped me to believe a little more in myself and what I am capable of doing and not feel ashamed of asking for help. I still need to improve on time management and finding solutions more efficiently, but overall this course is definitely one for the books.

References :

- OpenCV:
  1. https://docs.opencv.org/3.3.0/dc/dbb/tutorial_py_calibration.html
  2. https://www.tutorialspoint.com/detection-of-a-specific-color-blue-here-using-opencv-with-python
  3. https://docs.opencv.org/3.4/d4/d73/tutorial_py_contours_begin.html (Others from the OpenCV documentation as well)
  4. https://docs.opencv.org/3.4/d4/dc6/tutorial_py_template_matching.html
- P.I.D controllers:

1. https://www.youtube.com/watch?v=y3K6FUgrgXw
2. https://github.com/vazgriz/PID_Controller/blob/master/Assets/Scripts/PID_Controller.cs

Received help from: Xiao (TA), Justin (TA), Marcus (Partner & student), Haruto (Student)