MRSD Project Course

Team I – AIce

Autonomous Zamboni Convoy

Individual Lab Report 05

Author

Nick Carcione

Teammates

Rathin Shah

Yilin Cai

Jiayi Qiu

Kelvin Shen

April 7, 2022

Contents

1. Individual Progress

Since the last Progress Review, I have largely wrapped up the development and integration of the RC car. With the encoder mounted and integrated into the Arduino circuit, the actual speed of the DC motor, and thus the vehicle, could be determined. I used this encoder data to develop a new speed controller for the RC car, similar to the one developed for the DC motor in the Sensors and Motors Lab. Using the current RPM of the motor as measured by the encoder, the actual speed of the vehicle is calculated and compared to the desired speed from the trajectory planner. The error in the speed, defined as the difference between the desired and actual speeds, is then used with a PID controller to calculate an incremental change to the ESC command. If the actual speed is too low, the error is positive and the ESC command is increased, and vice versa. The error in the speeds is checked every iteration of the Arduino loop, and the ESC command is adjusted accordingly until the desired speed is reached.

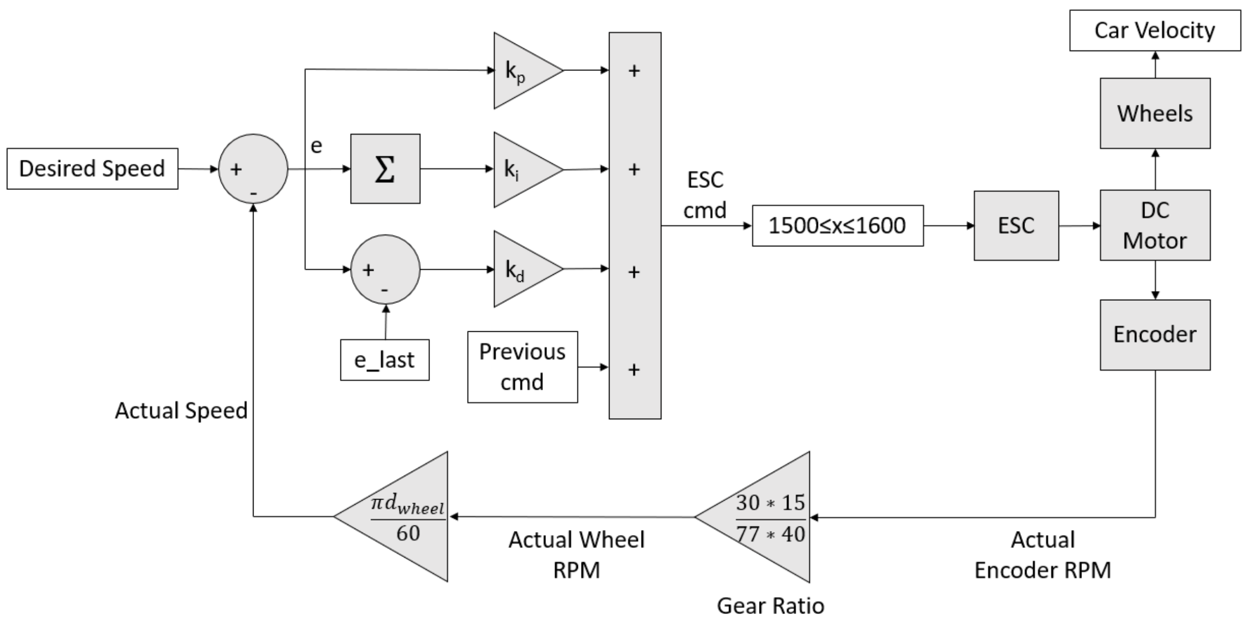

Figure 1: Block diagram of the RC car speed controller for a positive velocity

With such a controller, there is now more of a guarantee that the RC car is moving at the requested speed than before when an assumed linear mapping was used. The results of some initial testing with PID gains of $k_p = 1.0$, $k_i = 0$, and $k_d = 2.5$ are shown on the next page in Figure 2. With such a low proportional gain, there is some visible lag in the system response, especially at the start of the test. When the system first turns on, the vehicle is sent a brake command, which equates to a 1500 µs delay command to ESC. While commands just above 1500 µs are valid and should result in forward motion, it isn't until ~1534 µs that the DC motor is able to overcome static friction and start moving. This dead band causes the long delay at the start of the test. The test results also show that the actual speed is noisy and contains some significant spikes. These spikes are not so much an artifact of the controller, but the hardware. Even when sending a constant signal to the ESC, the speed of the DC motor will randomly spike.

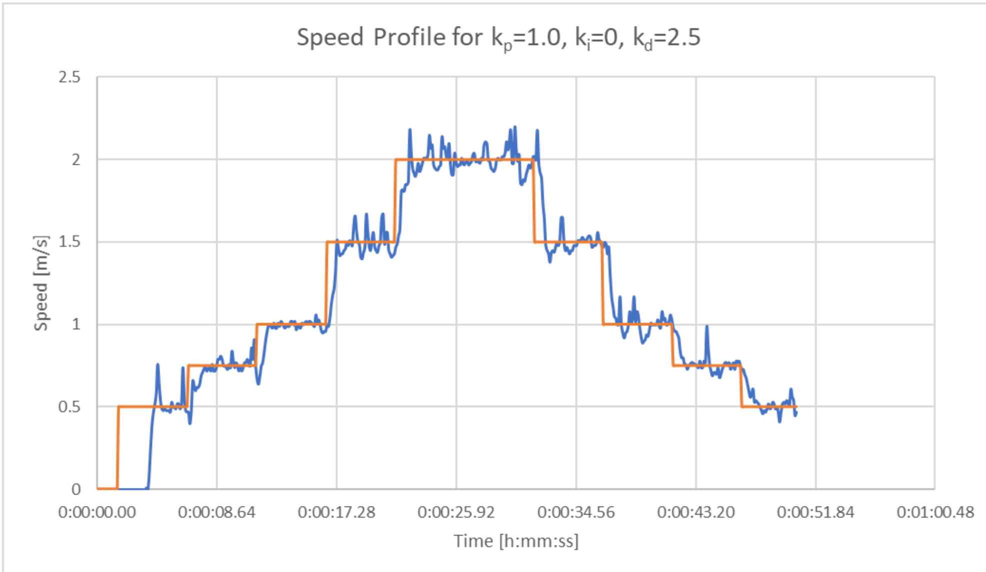

Figure 2: Initial RC car speed controller results for a pre-defined speed profile

As Table 1 below shows, the speed controller does a good job of maintaining the desired speed, with the average speed measured during each interval closely matching the set point.

Table 1: Average vehicle speed measured during testing compared to set speed

Avg. Actual Speed	Desired Speed	Difference
0.000	0	0.000
0.523	0.5	0.023
0.759	0.75	0.009
0.995	1	-0.005
1.496	1.5	-0.004
2.000	2	0.000
1.495	1.5	-0.005
1.008	1	0.008
0.760	0.75	0.010
0.515	0.5	0.015

I also worked on integrating the perception and software systems onto the RC car in preparation for final testing. The RealSense camera was mounted onto the car using the mount designed earlier this semester. The Jetson was also mounted onto the car. Initially, I had planned to design a mount to connect the Jetson to the top acrylic platform of the vehicle. However, after noticing that the Jetson came with 4 external mounting holes, Rathin and I decided to drill 4 through holes in the acrylic and mount the Jetson directly onto the platform without any intermediary. The power source for the Jetson was also included on the RC car and was attached using Velcro. Integrating the Jetson and its power source in these ways reduced the amount of space they would take up and eliminated the time it would take to design and integrate unnecessary mounts.

With all the required components mounted onto the RC car, we were able to begin doing some localization testing to validate the x and y output of the localization algorithm. The RC car was commanded to travel in a straight line at a constant speed of ~0.6 m/s for 2 seconds. The x and y position output by the localization algorithm was then compared to the actual distances the car had travelled. The x position was accurate to within 0.1 m. However, the y position was only accurate to within 0.3 m, which may be due to the car drifting despite the assumed straight-line path.

2. Challenges

While I was developing the speed controller for the RC car, I ran into an issue where, when a small, positive velocity was sent to the car, the wheels would begin moving forward before reversing direction and spinning backwards at maximum speed. In an attempt to fix this, I tried constraining the wheels to only spin forward or not at all when a positive velocity was given and vice versa for a negative velocity. This resulted in the wheel speed "bouncing." The wheels would quickly speed up to an RPM well above the desired RPM before dropping back down to 0 RPM. This up-and-down cycle would then repeat, and the speed would never converge to the set point. Eventually, I discovered that the reason for this came from how the RPM of the wheels was being calculated from the encoder data. The ticks counted by the encoder were only being checked once a second, causing the measured RPM to only be updated once a second. However, the output to the ESC that was calculated by the speed controller was being updated every loop. Changing the encoder code so that it updated the RPM measurement every loop iteration and adding a delay of 100 ms fixed this issue. It should also be noted that the delay of 100 ms proved to be necessary for accurate RPM measurements. Shorter delays (such as 10 ms) resulted in accurate readings, assumingly due to too few encoder ticks during the interval.

Some issues with the ground clearance of the RC car arose while integrating everything onto the car. The weight of the sensors, acrylic platforms, and especially the Jetson and its power source proved to be too heavy for the suspension system and would cause the bottom of the RC car to scrape the ground while running. To fix this, we exchanged the tires of the vehicle with slightly larger tires from another car in inventory. We also tightened the suspension system to add some pretension to the springs and changed where the suspension springs connect to the chassis so that the springs were more vertically oriented. These changes fixed the ground clearance issues and the bottom of the vehicle no longer scrapes along the ground.

Another challenge we faced was getting ROS and Arduino to communicate with each other after integrating them both onto the RC car. After integrating the systems together, we found that either the ESC could correctly send speed commands to the DC motor or the Arduino script could publish/subscribe to ROS nodes, but not both. After stripping down the Arduino code and testing it in small chunks, we found that the implementation of the encoder was the problem. The initial implementation included a check to see if 1 second had passed, and only then would the RPM be calculated. Replacing this with a simple delay solved the communication issue.

3. Teamwork

Rathin Shah

Rathin worked with Yilin and me on the integration of the Jetson Xavier with the Arduino and other sensors on the RC car. He also helped with the localization testing that was performed once the integration was completed. Rathin finished the development of the team's PCB and soldered the components on to finish its construction.

Yilin Cai

Yilin worked alongside Rathin and me to integrate the encoder and IMU on the RC car with the localization algorithm. He helped in testing the localization algorithm and visualizing the results as well. He also integrated remote teleoperation, perception, motion planning, and control code onto the car. Finally, Yilin worked with Kelvin to solve a communication issue that we ran into with the RealSense.

Jiayi Qiu

Jiayi made improvements to the waypoint generation algorithm. She used the time information of when a leader pose was detected to better estimate the speed that the follower would need to maintain a constant distance between the leader and follower. She also improved the initialization process for the leader-follower convoy. Finally, she worked with Kelvin to integrate the 2-camera setup with the waypoint generation algorithm so that the waypoints are based on the perceived position of the leader.

Kelvin Shen

Kelvin added a second camera onto the follower and two more ArUco markers boards onto the leader (one on the left and one on the right) in the simulation. This allowed to follower Zamboni to perceive the leader even when the leader would make sharp turns and U-turns. He then worked with Jiayi on integrating the pose estimation from this two-camera system with the waypoint generation algorithm. Finally, Kelvin got a Husky that will be used as a leader during testing running and operable through teleoperation.

4. Plans

In the next two weeks before the SVD, I will work with the rest of my teammates to finish integrating everything onto the RC car and do some final testing and validation of the various subsystems. This will include testing the leader detection and leader velocity estimation algorithms, validating the yaw component of the localization algorithm, and testing the leader-follower functionality of the full system. I will also work on tuning the gains of the RC car speed controller to get the car to react quicker while still maintaining accuracy with respect to the desired speed.